

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

รณสารวจผ่านอินเทอร์เน็ต

Internet Controlled Robot



โดย

นายเจษฎา วาจาสิทธิศิลป์
นายชูเกียรติ เต็มเกรียงไกร

อาจารย์ที่ปรึกษา

อ.สุมิตร พนาอุดมทรัพย์

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2541

เลขที่.....

เลขทะเบียน..... 33997

วัน, เดือน, ปี 27 ก.ย. 2542

สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ปีการศึกษา 2541
ภาควิชา วิศวกรรมระบบควบคุม
คณะวิศวกรรมศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง รถสำรวจผ่านอินเตอร์เน็ต
(Internet Controlled Robot)

ผู้จัดทำ

- | | | |
|----------------|----------------|----------|
| 1.นายเจษฎา | วาจาสิทธิศิลป์ | 38014086 |
| 2.นายชูเกียรติ | เดิมนเกรียงไกร | 38014125 |

.....อาจารย์ที่ปรึกษา
(อ.สุมิตร พนาอุดมทรัพย์)

INTERNET CONTROLLED ROBOT

โดย

นายเจษฎา วาจาสิทธิศิลป์ 38014086

นายชูเกียรติ เดิมเกรียงไกร 38014125

อาจารย์ที่ปรึกษา

อาจารย์สุมิตร พนาอุดมทรัพย์

บทคัดย่อ

ปริญญานิพนธ์นี้เป็นการออกแบบและสร้างระบบควบคุมหุ่นยนต์โดยผ่านทางอินเทอร์เน็ต ซึ่งใช้ไมโครคอนโทรลเลอร์ 89C51 เป็นตัวควบคุมการเคลื่อนที่ของหุ่นยนต์ โดยมีกล้องวิดีโอไว้สำหรับรับภาพจากตัวหุ่น เพื่อใช้ในการสังเกตและแสดงผล โดยมีผู้ใช้ที่ควบคุมอยู่ทางหน้าจอคอมพิวเตอร์บนเว็บไซต์ทางอินเทอร์เน็ตเป็นผู้ทำการป้อนคำสั่งเพื่อที่จะทำการควบคุมให้หุ่นยนต์ไปตามความต้องการ

Abstract

This thesis presents the design and construction of the robot that is controlled via the Internet. The movement of the robot is controlled by an 80C51 Micro Controller and a VDO camera is installed to observe and display the movements to the human controller who is accessing the website could receive and send the movement orders and then process and issue the commands to the robot accordingly.

6.2 . ส่วนตัวกลางการประมวลผล	60
6.3 . ส่วนการติดต่อผู้ใช้	63
ภาคผนวก ก. โปรแกรมควบคุมการทำงาน, รูปวงจรอิเล็กทรอนิกส์ และรายละเอียดของอุปกรณ์ (Data Sheets)	65
ภาคผนวก ข. แนวทางการพัฒนา	66
เอกสารอ้างอิง	67



รูปที่	หน้า
5.22 แสดงการดูโครงสร้างของเทเบิล	46
5.23 ตัวอย่างการสร้างเทเบิล	46
5.24 การใช้ DataModule	48
5.25 การเซฟพอฟเพอร์ตี้ของคาตาซอร์ทและคาตาเซท	49
5.26 การนำ DataModule มาใช้	49
5.27 การคลิกโปรแกรม	51
5.28 Add Watch	51
5.29 Web Application	53
5.30 ไคลเรคทอรี Scripts	53
5.31 การทำงานของ ActionItem	54
5.32 ActiveX	54
5.33 การสร้าง ActiveX (ขั้นตอนที่ 1)	55
5.34 การสร้าง ActiveX (ขั้นตอนที่ 2)	55
5.35 การสร้าง ActiveX (ขั้นตอนที่ 5)	56
5.36 การสร้าง ActiveX (ขั้นตอนที่ 6)	57
5.37 การสร้าง ActiveX (ขั้นตอนที่ 10)	57

บทที่ 1

บทนำ

ในปัจจุบันนี้เทคโนโลยีทางด้านหุ่นยนต์และการนำคอมพิวเตอร์มาประยุกต์ใช้ในการควบคุมการทำงานของเครื่องจักรต่างๆ ได้เจริญก้าวหน้าอย่างรวดเร็ว จึงมีการนำหุ่นยนต์มาใช้ในการทำงานต่างๆ มากมายไม่ว่าจะเป็นงานที่เสี่ยงต่ออันตรายต่างๆ ภายในโรงงานอุตสาหกรรมแทนที่คน เพื่อเพิ่มความรวดเร็วและความแน่นอนในการทำงานและช่วยลดต้นทุนในการผลิตอีกด้วย เป็นต้น ด้วยเหตุนี้จึงมีความจำเป็นอย่างยิ่งที่จะต้องมีการพัฒนาประสิทธิภาพของหุ่นยนต์ไม่ว่าจะเป็นในด้านการติดต่อ, วิธีการการควบคุมรวมถึงความสามารถในการทำงานเพื่อที่จะทำให้หุ่นยนต์มีความสามารถในการทำงานได้หลายอย่างในตัวเดียวกัน และในโครงการนี้ก็เป็นส่วนหนึ่งในการพัฒนาการควบคุมการติดต่อกับหุ่นยนต์โดยใช้อินเทอร์เน็ต(Internet)เข้ามามีส่วนร่วมในการควบคุมและติดต่อ

หลักการการทำงานของ INTERNET CONTROLLED ROBOT

หุ่นยนต์ที่ใช้ในการควบคุม(ซึ่งในที่นี้เราขอใช้คำว่ารถแทน)จะเคลื่อนที่ไปตามทางที่เราต้องการ โดยที่รถนั้นจะมีกล้องวิดีโอไว้สำหรับจับภาพตำแหน่งที่รถอยู่ในขณะนั้นและส่งภาพกลับไปบนหน้าจอคอมพิวเตอร์เพื่อให้ผู้ใช้ที่อยู่หน้าคอมพิวเตอร์รู้ว่าตอนนี้ตัวรถอยู่ที่ใด เมื่อต้องการให้รถไปทางใดก็ป้อนคำสั่งที่อยู่บนหน้าจอคอมพิวเตอร์(ซึ่งในการควบคุมนั้นจะควบคุมบนอินเทอร์เน็ตโดยเข้าไปควบคุมและดูภาพตำแหน่งของรถในขณะนั้นได้โดยการเรียกเข้าไปยังเว็บเพจหรือที่เรียกว่าเอกสารที่อยู่บนอินเทอร์เน็ต) จากนั้นคอมพิวเตอร์ก็จะรับค่าข้อมูลที่เกิดจากการป้อนคำสั่งทางอินเทอร์เน็ต โดยนำค่านั้นไปประมวลผลเพื่อที่จะได้ทราบว่าผู้ใช้ที่อยู่บนหน้าจอคอมพิวเตอร์นั้นต้องการอะไรแล้วก็ส่งค่าที่ได้จากการประมวลผลนั้นส่งไปยังไมโครคอนโทรลเลอร์ ซึ่งเป็นตัวที่ใช้ในการควบคุมมอเตอร์เพื่อให้รถนั้นเคลื่อนที่ตามต้องการ โดยส่งค่าโดยผ่านทางพอร์ตอนุกรม RS-232

วัตถุประสงค์

- 1.เพื่อศึกษาถึงการควบคุมมอเตอร์โดยใช้ ไอซีเบอร์ LMD 18201
- 2.เพื่อศึกษาถึงการใชไมโครคอนโทรลเลอร์ 89C51 ในการควบคุมมอเตอร์
- 3.เพื่อศึกษาและเรียนรู้การใช้งานในการออกแบบเว็บกราฟิกด้วย HTML
- 4.เพื่อศึกษาถึงการติดต่อระหว่างอินเทอร์เน็ต

ขอบเขตงาน

ในโครงการนี้ได้แบ่งงานออกเป็น 2 ส่วน คือ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1. ส่วนของ Hardware

ประกอบด้วย

- ตัวรถ
- ตัวกล้อง VDO
- Power Supply
- วงจรอิเล็กทรอนิกส์
- เครื่องคอมพิวเตอร์

2. ส่วนของ Software

ประกอบด้วย

- โปรแกรมการออกแบบเว็บเพจกราฟิกด้วย HTML
- โปรแกรมการรับค่า, ส่งค่าและประมวลผลระหว่างอินเทอร์เน็ตด้วยโปรแกรม DELPHI
- โปรแกรมการควบคุมมอเตอร์โดยใช้โปรแกรม ASSEMBLY

เนื้อหาของปริญญานิพนธ์

บทที่ 1 กล่าวถึง หลักการทำงานของ INTERNET CONTROLLED ROBOT, วัตถุประสงค์, ขอบเขตงานและเนื้อหาของปริญญานิพนธ์

บทที่ 2 กล่าวถึง รายละเอียดและทฤษฎีต่างๆที่เกี่ยวกับส่วน Hardware, ไมโครคอนโทรลเลอร์

บทที่ 3 กล่าวถึง รายละเอียดและทฤษฎีการออกแบบเว็บเพจด้วย HTML

บทที่ 4 กล่าวถึง รายละเอียดการส่งข้อมูลผ่านพอร์ตอนุกรม

บทที่ 5 กล่าวถึง รายละเอียดและทฤษฎี Delphi

บทที่ 6 กล่าวถึง รายละเอียดการทำงานของระบบ

ภาคผนวก ก. แสดงรายละเอียดของ โปรแกรม โดยสมบูรณ์

ภาคผนวก ข. กล่าวถึงแนวทางในการพัฒนาโครงการนี้ในขั้นต่อไป

บทที่ 2

Hardware

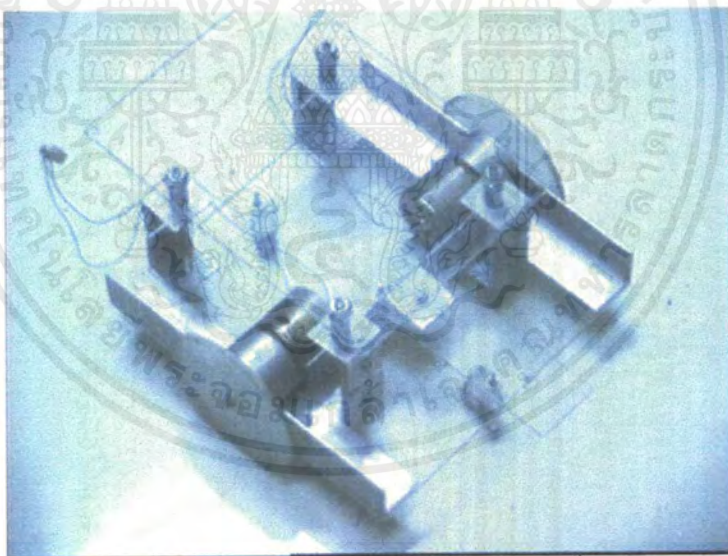
2.1 ส่วนประกอบต่างๆทาง Hardware

ในโครงงานนี้นั้นจะประกอบไปด้วยส่วนต่างๆทาง Hardware ดังนี้

1.ตัวรถ : ซึ่งใช้วัสดุดังนี้

- 1.1 แผ่นพลาสติก : เพื่อนำมาทำเป็นตัวโครงรถร่วมกับโครงอลูมิเนียม และนำมาทำเป็นล้อ
- 1.2 อลูมิเนียมฉาก : เพื่อนำมาทำเป็นโครงรถ
- 1.3 น็อตขนาดต่าง : เพื่อใช้ยึดติดอุปกรณ์ต่างๆเข้าด้วยกัน
- 1.4 มอเตอร์ : เพื่อใช้ในการขับเคลื่อนล้อรถ

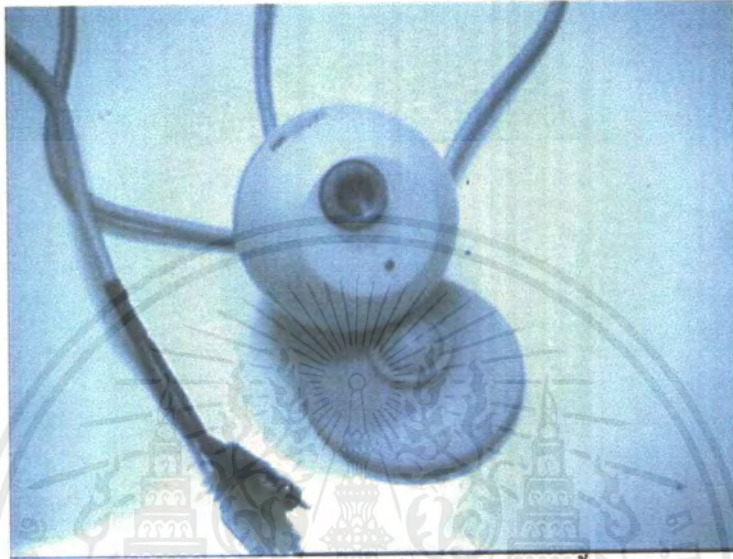
เมื่อประกอบออกมาดังรูป



รูปที่ 2.1 ภาพตัวรถจริงที่ใช้ในโครงงานนี้

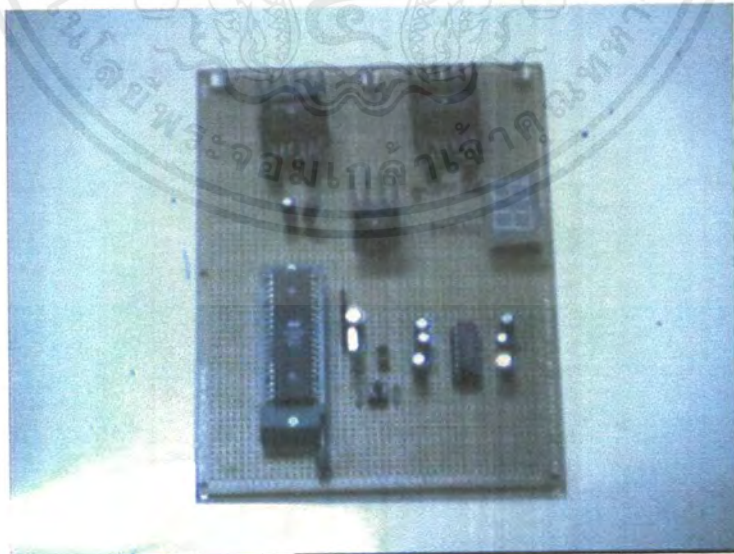
2.เครื่อง Computer : เพื่อใช้เป็นตัว Server

3. ส่วนรับภาพ : ซึ่งในที่นี้เราใช้กล้อง Digital Camera ยี่ห้อ CUCUS ซึ่งจะมีการ์ดใช้ในการ Capture ภาพประกอบมาด้วยทั้งคู่



รูปที่ 2.2 รูปตัวกล้องที่ใช้ในโครงการนี้

4. ส่วนที่ใช้ในการจ่ายไฟให้กับวงจร และอุปกรณ์ต่างๆ : ในโครงการนี้เราจะใช้การจ่ายไฟโดยใช้ Power Supply โคนจะจ่ายแรงดันไฟขนาดประมาณ 12 V



รูปที่ 2.3 รูปแผงวงจรทั้งหมดที่ใช้ในโครงการนี้

5. วงจร : วงจรของเรานั้นแบ่งออกได้เป็นส่วนๆดังนี้

5.1 วงจรขับเคลื่อนมอเตอร์ ซึ่งเราจะใช้ ไอซี LMD 18201 ในการขับเคลื่อนมอเตอร์ของรถ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2 วงจรการทำงานของหน่วยประมวลผล (ไมโครคอนโทรลเลอร์)

5.3 วงจรแปลงแรงดันให้เป็นไปตาม TTL และ RS 232

5.4 วงจรแสดงผล 7 Segment

2.2 ไมโครคอนโทรลเลอร์

การเลือกใช้ไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ MCS-51 เป็นไมโครคอมพิวเตอร์แบบจีพียูเดียว (ไม่ต้องต่อกับอุปกรณ์ภายนอกก็สามารถทำงานได้) มีความสะดวกในการใช้งานและเขียนโปรแกรมควบคุมด้วยภาษาเบสิกได้ โดยไม่ต้องศึกษาการทำงานของวงจรเหมือนกับภาษาแอสเซมบลี แต่ภาษาแอสเซมบลีก็ยังคงสามารถใช้ในการเขียนโปรแกรมได้เช่นกัน MCS-51 นี้เป็นอุปกรณ์ที่ออกแบบเพื่อตอบสนองความต้องการของผู้ใช้คือ มีสายอินพุต และเอาต์พุตภายในตัวเอง พอร์ตของอินพุต และเอาต์พุตบัฟเฟอร์ อินเตอร์เฟส และสายควบคุมอื่นๆที่ใช้สำหรับแลกเปลี่ยนข้อมูลกับแอสเซอเรตออกจากรันและยังมีชุดคำสั่งเพิ่มขึ้นเป็นพิเศษ เพื่อจัดการข้อมูลพร้อมด้วยวงจรตั้งเวลาและวงจรรนับด้วยสำหรับในโครงการนี้ได้เลือกใช้ ไมโครคอนโทรลเลอร์เบอร์ 89C51 ซึ่งมีคุณสมบัติดังนี้

- ต้องการแหล่งจ่ายไฟ 5 โวลต์ เพียงชุดเดียว
- มีหน่วยความจำสำหรับเก็บโปรแกรมควบคุมการทำงานอยู่ภายในชิปจำนวน 4 กิโลไบต์ (เบอร์ 8031, 8032 ไม่มีหน่วยความจำส่วนนี้ ส่วนเบอร์ 8052 มีหน่วยความจำส่วนนี้ 8 กิโลไบต์ และสำหรับเบอร์ 83C51FB จะมีหน่วยความจำส่วนนี้รวมทั้งสิ้น 16 กิโลไบต์)
- มีหน่วยความจำสำหรับเก็บข้อมูลทั่วไป (RAM) อยู่ภายในชิปจำนวน 128 กิโลไบต์ สามารถใช้หน่วยความจำสำหรับโปรแกรมและข้อมูลที่อยู่ภายในชิปได้อย่างละ 64 กิโลไบต์ แยกจากกัน
- คำสั่งส่วนใหญ่ใช้เวลาทำงาน 1 ไมโครวินาที เมื่อใช้คริสตอลความถี่ 12 เมกะเฮิร์ตซ์
- มีพอร์ตที่สามารถรับหรือส่งข้อมูลได้ทั้ง 2 ทิศทาง จำนวน 4 พอร์ตๆ ละ 8 บิต หรือสามารถใช้งานเป็นพอร์ตขนาด 1 บิตแยกจากกัน ทำให้เสมือนมีพอร์ตขนาด 1 บิตใช้งานรวมทั้งสิ้น 32 พอร์ต
- รับและส่งข้อมูลแบบอนุกรมได้ในตัว โดยสามารถกำหนดอัตราเร็วในการรับและส่งข้อมูล (baud rate) ได้ตั้งแต่ 300 ถึง 375 กิโลบิตต่อวินาที
- จัดลำดับความสำคัญของสัญญาณอินเทอร์รัปต์ได้ 2 ระดับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- มีรีจิสเตอร์สำหรับใช้งานเป็นไทม์เมอร์หรือเคาน์เตอร์เพื่อบันทึกจำนวนสัญญาณนาฬิกาภายในชิป หรือนับการเปลี่ยนสถานะของสัญญาณภายนอกขนาด 16 บิต จำนวน 2 ตัว เพื่อใช้สำหรับนับจำนวนพัลส์ วัดความกว้างของพัลส์หรือใช้วัดช่วงเวลา หน่วยความจำสำหรับเก็บข้อมูลภายในบางส่วนสามารถเข้าถึงข้อมูลได้ทั้งระดับไบต์และระดับบิตเพื่อให้การออกแบบโปรแกรมและการควบคุมระบบทำงานได้ง่ายขึ้น
- มีคำสั่งคูณและหารเลขขนาด 8 บิตในตัวเอง
- สามารถประมวลผลแบบบูลีน เพื่อใช้ในงานควบคุมโดยเฉพาะ
- ใช้โปรแกรมของไมโครคอนโทรลเลอร์ตระกูล MCS-48 (upwardly compatible) ได้

2.3 การขับเคลื่อนและการเลี้ยง

การเคลื่อนที่ของรถนั้นเราใช้พลังงานในการขับเคลื่อนจาก Power Supply และขับเคลื่อนโดยใช้มอเตอร์กระแสตรง ซึ่งมีการควบคุมทิศทางการหมุน, ความเร็วและตำแหน่งโดยใช้คอมพิวเตอร์ สำหรับเทคนิคการเลี้ยงนั้นใช้ระบบ 4 ล้อ มีล้อหน้าและล้อหลังเป็นตัวช่วยพยุงส่วนการเลี้ยงนั้นเราใช้ล้อข้างทั้ง 2 ข้างนั้นเป็นตัวบังคับทิศทาง โดยที่ถ้าเราต้องการไปทางด้านซ้ายเราก็สั่งให้ล้อทางด้านซ้ายนั้นหยุด ล้อทางขวาทำงาน ในทางกลับกันถ้าต้องการเลี้ยวขวาก็สั่งให้ล้อทางด้านขวาหยุด ล้อทางซ้ายทำงาน ถ้าเราต้องการไปทางข้างหน้าเราก็ให้ล้อทั้ง 2 ข้างทำงานพร้อมกัน ถ้าเราจะถอยหลังก็เพียงสั่งให้มอเตอร์ทำงานกลับทิศทางที่เราต้องการไปข้างหน้า แต่ถ้าเราต้องการหมุนรอบตัวเองก็เพียงแต่สั่งให้ล้อข้างหนึ่งหยุดอีกข้างหนึ่งทำงาน

บทที่ 3

การออกแบบเว็บเพจกราฟิก

อินเทอร์เน็ต เป็นคำศัพท์ที่เราคงคุ้นเคยกันมาบ้างแล้ว กล่าวโดยง่ายก็คือเป็นการเชื่อมคอมพิวเตอร์จากสมาชิกทั่วโลกเข้ามาไว้ด้วยกัน โดยจะมีการแลกเปลี่ยนข้อมูลซึ่งกันและกัน แต่บุคคลทั่วไปคงเชื่อมกันเองไม่ได้ ต้องเชื่อมกับ “คนกลาง” ซึ่งเรียกว่า “Internet Service Provider (ISP)” หรือผู้ให้บริการอินเทอร์เน็ต ซึ่งปัจจุบันมีอยู่มากมาย

เมื่อต่อเชื่อมได้แล้ว เราจะเห็นข้อมูลต่างๆบนจอภาพ ข้อมูลเหล่านี้เกิดจากบุคคลต่างๆสร้างไว้และนำมาเสนอไว้ในส่วนกลาง เพื่อให้ผู้ใช้คนอื่นๆอ่าน . ซึ่งอาจมีทั้งการโฆษณาประชาสัมพันธ์ การให้บริการทางสังคมและการบริการทางธุรกิจ แต่การจะนำข้อมูลมาเสนอนี้ จะต้องมีความรู้มาตรฐานสากล มิฉะนั้นจะเกิดความวุ่นวายและยุ่งยากมาก มาตรฐานสากลที่กำหนดไว้ก็คือ ให้เขียนเอกสารที่จะนำมาเสนอบนอินเทอร์เน็ตด้วยภาษาที่เรียกว่า “HTML” (HyperText Markup Language) ซึ่งเป็นภาษาที่ไม่ยาก เพียงแต่มีรูปแบบและกฎเกณฑ์บางอย่างที่เราต้องปฏิบัติตามเท่านั้น เครื่องมือหรือโปรแกรมที่จะเขียนภาษา HTML เรียกว่า “HTML Editor” ซึ่งปัจจุบันก็มีมากมายหลายยี่ห้อ หรืออาจใช้โปรแกรม Text Editor ทั่วๆไปก็ได้ (แต่ต้อง Save ให้อยู่ในรูปของ ASCII หรือ TEXT เท่านั้น) เมื่อเขียนเสร็จเรียบร้อยแล้วก็นำข้อมูลนั้นไปใส่ในส่วนกลาง (ซึ่งก็คือใส่ไว้ใน Server ของ ISP ที่เราเป็นสมาชิกอยู่นั่นเอง) ข้อมูลที่เรานำไปใส่นี้เรียกว่า “Web Page” (บางครั้งเราก็เรียกว่า “Home Page”)

Web Page ที่แสดงอยู่บนอินเทอร์เน็ตนี้มีใช้อยู่ก็สามารถเรียกออกมาดูได้ แต่จะต้องใช้เครื่องมือหรือโปรแกรมมาดู เครื่องมือหรือโปรแกรมที่วันนี้เราจะเรียกว่า “Web Browser” ซึ่งปัจจุบันนี้มีอยู่มากมายแต่จะมีที่นิยมมากที่สุดเห็นจะเป็น Netscape Navigator และ Microsoft-Internet Explorer

- | | |
|-------------|--|
| HTML | – ภาษาที่ใช้สร้างเอกสารเพื่อแสดงบนอินเทอร์เน็ต |
| HTML Editor | – โปรแกรมที่ใช้ในการสร้างภาษา HTML ให้ง่ายขึ้น |
| Web Browser | – โปรแกรมที่ใช้ในการดูเอกสารบนอินเทอร์เน็ต |
| Web Page | – เอกสารที่อยู่บนอินเทอร์เน็ต |

3.1 Web Page

ก่อนที่เราจะไปทำการศึกษาถึงการออกแบบเว็บเพจเรามาทำความเข้าใจถึงส่วนประกอบต่างๆที่บรรจุอยู่ในเว็บเพจโดยคร่าวๆกันก่อน เช่น

Text เป็นข้อความปกติ โดยเราสามารถตกแต่งให้สวยงาม และมีลูกเล่นต่างๆคล้ายกับเอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การใช้งาน ด้วย Word Processing

Graphics	มีรูปภาพ (Picture),ลายเส้น (Lines),ลายพื้น(Background) ต่างๆมากมายแล้วแต่จะเลือกใส่.
Multimedia	เป็นรูปภาพและเสียงประกอบกัน ซึ่งเราจะเรียกว่า Audio และ Video
Counter	ใช้นับจำนวนคนที่เข้ามาเยี่ยมชม Web Page ของเรา
Cool Links	ใช้ต่อเชื่อมไปยัง Web Page หรืออื่นๆของเรา หรือของคนอื่นก็ได้
Forms	เป็นแบบฟอร์มที่ให้ผู้เข้าชม กรอกรายละเอียด แล้วส่งกลับมายังเรา
Frames	เป็นการแบ่งจอภาพเป็นส่วนๆแต่ละส่วนก็จะแสดงข้อมูลที่แตกต่างกัน และเป็นอิสระจากกันต่างกันไป
Image Maps	เป็นภาพขนาดใหญ่ ที่กำหนดให้ส่วนต่างๆบนรูป เชื่อมโยง (Link) ไปยัง Web Page อื่นๆที่แตก Java Applets เป็นโปรแกรมสำเร็จรูปเล็กๆที่ใส่ลงไปใน Web Page เพื่อให้การใช้งาน Web Page มีประสิทธิภาพมากยิ่งขึ้น

3.2 รูปแบบของ HTML

มีศัพท์ 2 คำที่ควรทราบเกี่ยวกับการสร้าง Web Page ด้วย HTML ได้แก่

Text หมายถึง ข้อความทั่วไป

Tag หมายถึง คำสั่งที่ใช้ในการกำหนดรูปแบบของ Text โดยคำสั่งของ HTML จะอยู่ภายในเครื่องหมาย < >

กฎในการใช้ HTML

- แต่ละ Tag จะต้องอยู่ในเครื่องหมาย < > เท่านั้น
- จะใช้ตัวพิมพ์เล็กหรือพิมพ์ใหญ่ก็ได้มีความหมายเดียวกัน เช่น <title>,<Title> หรือ <TITLE> ต่างก็ให้ความหมายเดียวกัน
- แทบทุก Tag จะมี Tag เปิด และ Tag ปิด โดย Tag ปิดจะมี / รวมอยู่ด้วย แต่ก็มีบาง Tag ที่ไม่มีตัวปิด เช่น
,<P> เป็นต้น
- ในการพิมพ์เอกสาร HTML นั้นจะต้องพิมพ์ติดต่อกันทั้งข้อความและคำสั่งหรือพิมพ์แยกบรรทัดหรือเคาะช่องว่างก็ได้แล้วแต่ผู้พิมพ์ แต่การพิมพ์ให้ดูสวยงามจะทำให้ง่ายต่อการอ่านและแก้ไขในภายหลัง

การทำงานของ HTML กับ Browser

ปกติถ้าเรานำข้อมูลที่สร้างจากโปรแกรมต่างๆไปที่ไม่ใช่ HTML มาแสดงใน Web-Browser ผลลัพธ์ที่ได้จะไม่ถูกต้องตามเอกสารต้นฉบับ เนื่องจาก Browser ไม่เข้าใจในโครงสร้างของข้อมูล แต่ก็ก็จะแสดงผลให้Browserเป็นตัวแปรภาษา HTML ที่แปลงโดยตรงไปตรงมา ถ้าโครงสร้างเอกสารนี้เป็นเอกสารที่สว่นไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่นอนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สร้างผิดๆ ก็จะแสดงแบบผิดๆ ถ้าถูกก็จะแสดงผลตรงตามที่ต้องการ ไม่มีการบอกว่าผิดพลาด (Error) ที่ตำแหน่งใด เหมือนกับตัวแปลภาษาอื่นๆ แต่จะแสดงผลลัพธ์ให้เห็นเลยว่าจะผิดหรือถูก ดังนั้นผู้เขียนจึงจำเป็นต้องค้นหาส่วนที่ผิดให้ได้เอง

การจัดรูปแบบให้เป็น Paragraph และเป็นบรรทัด

จะใช้ `<HTML>` วางไว้ต้นเอกสารและ `</HTML>` ไว้ที่ท้ายเอกสารเพื่อให้ Browser ทราบว่าเป็นเอกสาร HTML

จะใช้ `<P>` เป็นตัวจัดให้ข้อความที่ตามหลังมาขึ้นย่อหน้า (Paragraph) ใหม่ เพื่อความเข้าใจ ไม่มีคำสั่งปิดท้าย

จะใช้ `<BR>` เพื่อให้ข้อความขึ้นบรรทัดใหม่

การจัดรูปแบบข้อความให้ตรงตามเอกสารเดิม

ในบางครั้งเราจะได้ข้อมูลที่โอนมาจากโปรแกรมอื่น เช่น Excel ซึ่งจัดรูปแบบไว้เรียบร้อยแล้ว จึงเป็นการเสียเวลาถ้าต้องมาใช้คำสั่ง HTML จัดซ้ำใหม่ หรือบางครั้งเราอาจต้องการจัดให้เสร็จเรียบร้อยใน Editor' เลยก็ได้ ลักษณะนี้จะใช้คำสั่ง `<PRE>` และ `</PRE>` มาช่วย โดยนำ `<PRE>` วางไว้หน้าข้อมูล และ `</PRE>` ไว้ท้ายข้อมูล

การแบ่งเอกสารออกเป็นส่วนๆ (Section)

กรณีที่เอกสาร HTML ที่สร้างมามีขนาดใหญ่ เราอาจ แบ่งเอกสารออกเป็นส่วนๆ เพื่อให้การค้นหาหรือแก้ไขได้ง่ายขึ้น (เอกสารขนาดเล็ก ก็นิยมทำเช่นเดียวกัน) การใส่ชื่อเรื่อง และขอบเขตของเนื้อหาลงไป

`<HEAD>`, `</HEAD>` ใช้กำหนดชื่อเรื่องให้กับส่วนเอกสาร แต่จะไม่แสดงให้เห็นใน Browser

`<BODY>`, `</BODY>` ใช้กำหนดขอบเขตของเนื้อหาในเอกสารส่วนนั้น (เนื้อหาที่แสดงใน Browser)

การกำหนดชื่อเอกสารให้แสดงบน Browser

เราสามารถแสดงเนื้อหาเรื่องไว้ใน Browser ได้ด้วยคำสั่ง `<TITLE>` และ `</TITLE>` โดยจะต้องวางไว้ระหว่างคำสั่ง `<HEAD>` และ `</HEAD>` เสมอ

การกำหนดขนาดของตัวอักษรในแต่ละ Section (Headline)

กรณีที่เอกสารถูกแบ่งออกเป็นหลายๆ ส่วน และเราต้องการกำหนดขนาดของตัวอักษรในแต่ละ Section ให้แตกต่างกันออกไป ก็ทำได้ง่ายๆ โดยใช้คำสั่ง `<Hn>` และ `</Hn>` โดย n แทนตัวเลข 1 ถึง 6 1 คือขนาดเล็กที่สุด และ 6 คือขนาดใหญ่สุด แก๊ไขขนาดเองไม่ได้ เนื่องจากถูก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กำหนดมาแล้ว โดยผู้ออกแบบ HTML ข้อความที่กำหนดด้วย Headline จะเริ่มต้นที่บรรทัดใหม่ทุกครั้ง

การกำหนด Footer

เป็นข้อความที่นิยมใส่ไว้ด้านล่างเอกสาร เช่น ที่อยู่ที่เราจะติดต่อได้ ใช้คำสั่ง `<ADDRESS>` และ `</ADDRESS>` ข้อความที่อยู่ระหว่าง Tags นี้จะถูกกำหนดให้เป็นตัวเอน (Italic) โดยอัตโนมัติ

การสร้างเส้นตรงแนวนอน (Horizontal Rule)

สร้างเส้นแนวนอนง่ายๆ ด้วยคำสั่ง `<HR>`

การจัดรูปแบบของตัวอักษร (Styles)

รูปแบบของตัวอักษรมี 2 แบบคือ Physical Styles และ Logical Styles

- Physical Styles เป็นรูปแบบที่ Browser บางตัวอาจไม่รู้จักร ทำให้แสดงออกไม่ได้
- Logical Styles จะถูกแปลงให้ไปใช้ตามรูปแบบของ Browser ที่เรียกมาใช้งานทำให้ใช้งานได้ทุก Browser

Physical Style ได้แก่

- Bold/Italic เป็นการกำหนดข้อความให้เป็นตัวเข้ม (Bold) โดยใช้คำสั่ง `<B>` และ `</B>` และกำหนดข้อความให้เป็นตัวเอน (Italic) โดยใช้คำสั่ง `<I>` และ `</I>`
- Blink กำหนดให้ตัวอักษรกระพริบได้ ใช้คำสั่ง `<BLINK>` และ `</BLINK>`
- Underline ใช้ขีดเส้นใต้ข้อความ (เส้นเดี่ยว) ใช้คำสั่ง `<U>` และ `</U>`
- Typewriter Text เป็นลักษณะตัวอักษรแบบพิมพ์ดีด คือจะมีความกว้างของแต่ละตัวอักษรเท่ากันหมดใช้คำสั่ง `<TT>` และ `</TT>`
- Superscript/Subscript เป็นตัวอักษรยกขึ้นและห้อยลงตามลำดับ ใช้คำสั่ง `<SUP>`, `</SUP>` และ `<SUB>`, `</SUB>`
- Strike เป็นเส้นตรงที่ใช้ขีดทับกลางข้อความ คำสั่งที่ใช้คือ `<STRIKE>` และ `</STRIKE>`

Logical Style ได้แก่

รูปแบบตัวอักษร

รูปแบบคำสั่ง

Citations

`<CITE>`, `</CITE>`

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Compute Code	<CODE>, </CODE>
Definition	<DFN>, </DFN>
Emphasis	,
Keyboard	<KBD>, </KBD>
Samples	<SAMP>
Strong	
Variable Name	<VAR>

การกำหนดให้ข้อความอยู่ในรูปของรายการ (List)

แบ่งออกเป็น

- Unordered Lists แสดงข้อความแยกเป็นบรรทัดๆ ด้วย Bullet (.) ใช้คำสั่ง
- Ordered Lists เหมือน Unordered List แต่แทน Bullet ด้วยตัวเลขใช้คำสั่ง
- Menu Lists เหมือน Order Lists แต่ข้อความต้องมีความยาวไม่เกิน 1 บรรทัด (จะแสดง Bullet หรือ ไม่ก็ขึ้นอยู่กับความสามารถ Browser) ใช้คำสั่ง <MENU>
- Directory Lists เหมือน Menu List แต่ใช้ได้เฉพาะข้อความสั้นๆ ไม่เกิน 20 ตัวอักษร ในแต่ละบรรทัด บาง Browser แสดงเป็น Bullet บาง Browser แสดงเป็น Bullet บาง Browser ก็แสดงเป็น Column ให้ใช้คำสั่ง <DIR>
- Definition Lists ใช้สำหรับการนิยามความหมายของคำใดๆ โดยคำและความหมายจะอยู่แยกบรรทัดกัน ใช้คำสั่ง <DL>, <DT> และ <DD>
 - <DL> และ </DL> สำหรับบอกว่าจะเริ่มต้นใช้งาน Definition Lists
 - <DT> และ </DT> Definition Term ใช้กำหนด “คำ” (Word)
 - <DD> และ </DD> Definition Description ใช้กำหนดคำอธิบายของคำใน Definition-Term

ในการสร้าง Lists เราจะใช้
 มาช่วยแยกบรรทัด หรืออาจใช้ ก็ได้เช่นกัน ส่วน

<LH> ใช้สำหรับ Header ให้กับ List (List Header)

3.3 การสร้าง Link

ความหมายและการทำงานของ Link

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Link เป็นการเชื่อมโยงจากจุดหนึ่งของเอกสารในปัจจุบันไปยังอีกจุดของเอกสารเดียวกันหรือคนละเอกสารก็ได้ ซึ่งการ Link อาจจะเป็นการ Link กันระหว่าง Text กับรูปภาพก็ได้ จุดที่ Link กันนี้ เราเรียกว่า “Anchor” ใช้คำสั่ง <A> และ โดยจะมีตัวเลือก “HREF= “ อยู่รวมด้วยการ Link ไปยัง File อื่น

สำหรับการระบุการ Link ไปยังสถานที่อื่นๆที่ไม่ใช่ระบบของเรา จะต้องระบุรูปแบบของ URL ให้ถูกต้อง เช่น <http://overworld.compuserve.com/homepages.nat.do.htm> แต่ถ้า Link ไปยังเอกสารที่อยู่ภายในระบบของเราเอง ให้ทำในรูปแบบดังนี้

- อยู่ใน Directory เดียวกัน ให้ใส่เฉพาะชื่อ File
- อยู่ใน Subdirectory ของ Directory ปัจจุบัน เช่น ถ้าปัจจุบันอยู่ที่ me/homepage/recipes.htm ต้องการ Link ไปที่ me/homepage/cook/cookie.htm ให้กำหนดเป็น

- อยู่ใน Directory ที่อยู่สูงกว่า Directory ปัจจุบัน เช่น ปัจจุบันอยู่ที่ me/homepage/recipts.htm ต้องการ Link ไปที่ tetra.htm อยู่ที่ Root Directory ให้กำหนดเป็น

การ Link ไปยังส่วนอื่นภายในเอกสารเดียวกัน

บางครั้งอาจต้องการ Link ไปยังตำแหน่งอื่นๆแต่ยังคงอยู่ในเอกสารเดียวกัน ให้กำหนด Label หรือชื่อให้กับตำแหน่งที่ต้องการจะ Link ไป โดยนำตัวเลือก NAME มาใช้ร่วมกับคำสั่ง <A> การ Link ไปยังส่วนใดๆใน File อื่น

ถ้าระบุการ Link ภายในเอกสารเดียวกัน ก็เพียงระบุชื่อ (NAME) ของส่วนนั้นลงไป แต่ถ้าเป็นเอกสารอื่น เราจะต้องระบุชื่อเอกสารลงไปด้วย พร้อมทั้งอย่าลืมตั้งชื่อ Label ให้กับส่วนของเอกสารนั้นๆด้วย

การ Link ไปยังบริการอื่นๆบนอินเทอร์เน็ต

- การ Link ไปยัง FTP

สามารถ Link ไปยัง FTP เพื่อขอใช้บริการการโอนข้อมูลได้ โดยระบุ Address ของ FTP ที่ต้องการลงไป

ให้กำหนดรูปแบบดังนี้

ในกรณีที่ Site นั้นจำเป็นต้องใช้ User ID และ Password จึงจะใช้ได้เพื่อความสะดวก เราควรระบุ User ID และ Password ลงไปเลย (แต่เราต้องรู้ค่าของทั้งสองตัวนี้ด้วย)

ให้กำหนดรูปแบบดังนี้

ftp://userid:password@address

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เพื่อให้สามารถควบคุมการแสดงส่วนต่างๆของเพจ HTML ได้เพิ่มค่าเสริมที่ใช้สำหรับกำหนดระยะระหว่างภาพกับข้อความ และรูปภาพกับขอบหน้าต่าง ทั้งแนวนอน และทางแนวตั้ง โดยกำหนดหน่วยเป็นพิกเซล

รูปแบบคำสั่ง

VSPACE : กำหนดระยะห่างแนวตั้ง

HSPACE : กำหนดระยะห่างแนวนอน

คำสั่งเปลี่ยนสีพื้นหลังของเพจ

รูปแบบคำสั่ง

<BODY Background = [URL] Filename>

BODY :เป็นคำสั่งกำหนดเนื้อหาของไฟล์ HTML

Background :เป็นค่าเสริมเพื่อกำหนดภาพพื้นหลัง

Filename :ไฟล์ภาพกราฟิกที่เราต้องการนำมาเป็นภาพพื้นหลัง

คำสั่ง <FRAME>

เราสามารถแบ่งพื้นที่ของ WebPage ออกเป็นส่วนๆโดยที่แต่ละส่วนมีข้อมูลอิสระการทำงานโดยไม่ขึ้นแก่กันส่วนของจอภาพที่ถูกแบ่งออกนี้ เราเรียกว่า เฟรม (Frame)

โครงสร้างของเฟรม

รูปแบบโครงสร้างของเฟรม

<HTML>

<HEAD>.....</HEAD>

<Frameset>

<Frame>

Other HTML Tag

<No Frame>

</Frameset>

</HTML>

คำสั่งหลักของการสร้างเฟรมมี 3 คำสั่ง คือ

<FrameSet Attribute> เป็นคำสั่งให้ Browser แบ่งหน้าจอเป็นเฟรมๆคำสั่งนี้จะใช้แทนที่คำสั่ง

<Body>

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- <Frame Attribute> เป็นคำสั่งกำหนดรายละเอียดของแต่ละเฟรม
- <No Frame> คำสั่งที่จะตีพิมพ์ข้อความให้ผู้ใช้ได้ทราบ เมื่อ Browser นั้นไม่สนับสนุนการทำงานใน รูปแบบเฟรม

รูปแบบคำสั่ง

<FRAME SRC = [URL] Name =Frame_name MarginWidth = value1 Marginheight = value2 Scrolling = choice1 NoResize FrameBorder = choice2 FrameSpacing = value2>

- SRC : เป็นค่าเสริมเพื่อ Load รูปภาพ หรือ File HTML
- URL : Address
- Name : เป็นคำสั่งบอกให้ Browser รู้ว่าเฟรมนี้จะมีชื่ออะไร รูปภาพและไฟล์ HTML ที่เรียกมาจะแสดงในเฟรมที่มีชื่อตามนี้
- Frame_name : ชื่อของเฟรมที่ตั้งขึ้น
- MarginWidth : การกำหนดระยะห่างทางซ้ายและขวา ระหว่างเฟรมกับข้อมูล
- MarginHeight : การกำหนดระยะห่างด้านบนและล่าง ระหว่างเฟรมกับข้อมูล
- Value1 : ค่าจำนวน Pixel
- Scrolling : การกำหนดการเลื่อนหน้าจอ
- Choice1 : คำที่กำหนดสำหรับคำสั่ง Scrolling
- NoReSize : เป็นค่าเสริมที่ทำให้ผู้ใช้ไม่สามารถปรับเปลี่ยนขนาดของเฟรมได้ตามต้องการ
- FrameBorder : คำสั่งสร้างกรอบของเฟรม
- Choice2 : คำกำหนดของกรอบเฟรม
- FrameSpacing : กำหนดระยะห่างระหว่างเฟรม
- Value2 : ค่าระยะระหว่างเฟรม

บทที่ 4

การส่งข้อมูลผ่านทางพอร์ตอนุกรม

4.1 รูปแบบข้อมูลในคอมพิวเตอร์

การที่จะทำความเข้าใจการส่งผ่านข้อมูล สิ่งแรกคือต้องทำความเข้าใจกับวิธีที่ข้อมูลถูกเก็บไว้ภายในคอมพิวเตอร์ก่อน

บิตและไบต์

ในเลขฐานสิบ มีตัวเลขอยู่สิบตัว คือ 0-9 การเพิ่ม 0 หนึ่งตัวเข้าทางซ้ายเป็นการคูณจำนวนด้วย 10 ในเลขฐาน 2 มีเพียงตัวเลข 2 ตัวคือ 0 และ 1 การเพิ่ม 0 เข้าทางซ้ายจำนวนเป็นการคูณจำนวนด้วย 2 ตัวเลข 0 หรือ 1 แต่ละตัวในเลขฐาน 2 เรียกว่า บิต 8 บิตเป็นหนึ่งไบต์ ผลที่ตามมาคือค่าของ 1 ไบต์จึงเป็นได้ตั้งแต่ 00000000 ถึง 11111111 หรือ 0-255 ในเลขฐาน 10 บิตที่อยู่ทางขวาสุดของไบต์เรียกว่า บิต 0 บิตที่อยู่ทางซ้ายสุดเรียกว่า บิต 7 บิต 0 เรียกว่าบิตที่มีนัยสำคัญต่ำสุด(least-significant bit) และ บิต 7 เรียกว่าบิตที่มีนัยสำคัญสูงสุด(most significant bit)

หมายเลขบิต	7	6	5	4	3	2	1	0
ค่าถ่วงเขต	128	64	32	16	8	4	2	1
การเขต	0	0	1	0	0	0	1	1
ค่าตามที่เขต	0	0	32	0	0	0	2	1

รูปที่ 4.1 แสดงจำนวน 35 ในฐานสอง

คอมพิวเตอร์เกือบทั้งหมดทำงานในระบบเลขฐานสอง เพราะว่ามันเป็นการง่ายที่จะแปลงรหัส 0 และ 1 เป็นแรงดันไฟฟ้าบวกและลบ ในคอมพิวเตอร์ส่วนใหญ่หน่วยเล็กที่สุดของหน่วยความจำที่อ้างถึงได้โดยการอ้างแอดเดรสคือไบต์ ดังนั้นเมื่อข้อมูลถูกเก็บและจัดการในคอมพิวเตอร์ตามปกติมันจึงถูกแปลงให้เป็นไบต์ที่เรียงลำดับกัน

4.2 การเข้ารหัสข้อความ

เมื่อข้อความ (อักขระ เครื่องหมายวรรคตอน และอื่น ๆ) ถูกเก็บในคอมพิวเตอร์ แต่ละตัวอักษรที่แตกต่างกันจะถูกแทนด้วยจำนวนที่ต่างกัน จำนวนเหล่านี้โดยปกติมีค่าจาก 0 ถึง 127 หรือจาก 0 ถึง 255 เนื่องจากไบต์หนึ่งสามารถมีค่าจาก 0 ถึง 255 มันจึงเป็นธรรมชาติที่จะให้หนึ่งไบต์แทนตัวอักษรหรือเครื่องหมายวรรคตอนแต่ละตัวในข้อมูลที่เป็นข้อความ

มีสองวิธีที่ต่างกันสำหรับการจับคู่ตัวอักษรกับจำนวน คือ EBCDIC (Extended Binary-Coded Decimal Interchange Code) ซึ่งถูกใช้ในคอมพิวเตอร์ชนิดอื่นของไอบีเอ็มยกเว้นไอบีเอ็มพีซี และ ASCII (American Standard Code for Information Interchange) ซึ่งถูกใช้ในคอมพิวเตอร์อื่น ส่วนใหญ่ เราจะเกี่ยวข้องกับ ASCII เท่านั้นในหนังสือเล่มนี้

ตาราง ASCII อย่างเป็นทางการให้จำนวนระหว่าง 32 ถึง 126 แทนตัวเลข ตัวอักษร เครื่องหมายวรรคตอนและสัญลักษณ์ที่ใช้กันทั่วไปอื่น ๆ จำนวนจาก 0 ถึง 31 และ 127 มีความหมายพิเศษ เช่น Carriage return, Line feed และตัวอักษรที่ไม่สามารถแสดงผลได้อื่น ๆ

ตัวอย่างเช่น ตัว A ถูกเก็บเป็นเลขฐานสิบ 65 ในฐานสอง คือ 01000001 คอมพิวเตอร์เก็บในเลขฐานสิบ 44 ซึ่งเป็น 00101100 ในฐานสอง

เนื่องจากจำนวน 127 ในฐานสองใช้เพียงเจ็ดบิต ตัวอักษรทั้งหมดถูกแทนด้วย 0 ถึง 127 สามารถถูกเก็บในหนึ่งไบต์ โดยจะเหลืออีกหนึ่งบิต เนื่องจากเราให้ชื่อบิตในไบต์หนึ่งตั้งแต่ศูนย์ถึงเจ็ด จะเห็นได้ว่ารหัส ASCII ใช้เพียงบิตศูนย์ถึงหก บิตเจ็ดถูกสำรองไว้

คอมพิวเตอร์หลายชนิดใช้เต็มที่ทั้งแปดบิตสำหรับการเข้ารหัส ทำให้มีรหัสที่แตกต่างกัน 256 ตัว 128 ตัวแรกเป็นไปตาม ASCII และส่วนที่เหลือถูกใช้สำหรับอักขระต่างชาติ สัญลักษณ์ทางคณิตศาสตร์ อักขระ กราฟิก และอื่น ๆ ตามแต่การออกแบบ โชคไม่ดีที่ไม่มีมาตรฐานสำหรับอักขระเพิ่มเติม (extended character) เหล่านี้ ซึ่งมักจะมีความหมายแตกต่างกันบนคอมพิวเตอร์คนละชนิด

รหัส ASCII ชนิดพิเศษ

รหัส 32 ตัวแรกในตาราง ASCII มีความหมายพิเศษ ดังในตารางที่ 4.1 มีหลายตัวได้รับการออกแบบเพื่อวัตถุประสงค์ทางการสื่อสาร โดยเฉพาะ

รหัส	อักขระ	ความหมาย
0	NULL	วิธีหนึ่งที่จะทำให้เกิดการหน่วงเวลาอย่างจงใจ ในอดีตมันมีความจำเป็นที่จะส่ง null หลังจาก Carriage return เพื่อให้เครื่องพิมพ์บัตรไปทางซ้ายสุดของหน้ากระดาษ ปัจจุบันเครื่องพิมพ์ทำงานได้เร็วขึ้น null จึงถูกใช้สำหรับจุดประสงค์อื่นหลายอย่าง

ตารางที่ 4.1 รหัส ASCII ชนิดพิเศษ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รหัส	อักขระ	ความหมาย
1	SOH	Start of heading แสดงว่าข้อความที่ตามมาเป็นส่วนหนึ่งของหัวข้อ
2	STX	Start of text แสดงจุดเริ่มต้นของข้อความจริงของข่าวสาร
3	ETX	End of text แสดงจุดสิ้นสุดของข้อความ
4	EOT	End of transmission แสดงการสิ้นสุดของการส่ง
5	ENQ	Enquiry โดยปกติถูกใช้เป็นส่วนหนึ่งของซอฟต์แวร์แฮนด์เช็กกิ้ง ในการขอให้คอมพิวเตอร์ฝ่ายรับตอบรับการได้รับข่าวสาร
6	ACK	Acknowledge การตอบรับการได้รับข่าวสาร
7	BEL	ส่งเสียงออกทางเทอร์มินัล
8	BS	Backspace
9	HT	Horizontal tab
10	LF	Line feed ทำให้ขึ้นบรรทัดใหม่ในตำแหน่งเดิม
11	VT	Vertical tab
12	FF	Form feed เลื่อนหน้ากระดาษไปหนึ่งหน้า
13	CR	Carriage return เลื่อนไปที่ต้นบรรทัด บางครั้งทำให้เกิด Line feed ด้วยเช่นกัน
14	SO	Shift out กำหนดจุดเริ่มต้นของรหัสควบคุมพิเศษบ่อยครั้งที่ใช้ Esc แทน
15	SI	Switch in กำหนดจุดสิ้นสุดของรหัสควบคุมที่เริ่มต้นโดย SO
16	DLE	Data link escape เหมือนกับ Esc
17	DC1	
18	DC2	Device control 1 ถึง 4 รหัสที่สำรองไว้ให้ใช้ตามความต้องการบางครั้งใช้ในซอฟต์แวร์แฮนด์เช็กกิ้ง
19	DC3	
20	DC4	

ตารางที่ 4.1 (ต่อ) รหัส ASCII ชนิดพิเศษ

รหัส	อักขระ	ความหมาย
21	NAK	Negative acknowledgement บ่งชี้ว่าข้อมูลที่ส่งนั้น ไม่ได้ถูกรับอย่างถูกต้อง ตัวอย่างเช่น พบความผิดพลาด ทางพาริตี
22	SYN	Synchronous idle เหมือนกับ NULL แต่ถูกใช้ในการ สื่อสารแบบซิงโครนัส เพื่อดูแลให้อุปกรณ์สองตัว ซิงโครไนซ์กันระหว่างการส่ง
23	ETB	End of transmission block ถูกใช้ในที่ซึ่งการส่งข้อมูล ถูกแบ่งเป็นบล็อก เพื่อวัตถุประสงค์ในการตรวจสอบ ข้อผิดพลาด
24	CAN	Cancel บ่งชี้ว่า ข้อมูลที่ถูกส่งไปควรถูกทิ้งไป
25	EM	End of medium บ่งชี้ว่ามาถึงปลายของเทปกระดาษ
26	SUB	Substitute แก้ไขตัวอักษรที่ถูกส่งมาผิดพลาด ถูกใช้ เพื่อป้องกันจุดสิ้นสุดของการส่งด้วยเช่นกัน
27	Esc	Escape บ่งชี้จุดเริ่มต้นของตัวอักษรที่ติดตามมาว่ามี ความหมายพิเศษ
28	FS	
29	GS	File group, Record และ Unit separator ตามลำดับ ใช้เพื่อกำหนดขอบเขตระหว่างส่วนของข้อความ
30	RS	
31	US	
32	DEL	บ่งชี้ว่า ตัวอักษรที่มาก่อนมันควรถูกลบ

ตารางที่ 4.1 (ต่อ) รหัส ASCII ชนิดพิเศษ

รหัส 1 ถึง 26 ถูกอ้างอิงถึงเป็น Ctrl-A ถึง Ctrl-Z ด้วยเช่นกัน และพวกมันสามารถถูกสร้าง
ด้วยแป้นพิมพ์ของคอมพิวเตอร์ โดยการกดปุ่ม Ctrl ค้างไว้ และกดปุ่มตัวอักษรที่เหมาะสมพร้อมกัน
(ดังนั้น 1 = Ctrl-a 2= Ctrl-B เป็นต้น) บางรหัสสามารถถูกป้อนเข้าโดยการกดปุ่มเฉพาะ เช่น Tab
สำหรับรหัส 9 หรือ Return สำหรับรหัส 13

การเข้ารหัสข้อมูลที่ไม่ใช่ข้อความ

แน่นอนว่าทุกอย่างที่ถูกเก็บในคอมพิวเตอร์ไม่ได้อยู่ในรูปของข้อความเสมอไป คำสั่งของ
โปรแกรม ข้อมูล และกราฟิกอิมเมจ เป็นตัวอย่าง ข้อมูลที่ไม่ได้ถูกเก็บในรูปแบบ ASCII

ข้อมูลประเภทนี้โดยปกติถูกเข้ารหัสให้ใช้ทุกค่าที่เป็นไปได้ของหนึ่งไบต์ จำนวนถูกเก็บในรูปแบบไบนารี และสามารถขยายไปเป็นหลายไบต์ คำสั่งของโปรแกรมมักจะประกอบด้วยหนึ่งหรือสองไบต์ เราเรียกข้อมูลประเภทนี้ว่า ข้อมูลไบนารี (Binary data) แม้ว่าข้อความจะถูกเก็บในรูปแบบไบนารีเช่นกัน

เนื่องจากไบต์ที่เก็บข้อมูลซึ่งไม่ใช่ข้อความสามารถเป็นค่าใด ๆ ก็ได้ ในเวลาที่มันตรงกับค่าที่มีความหมายพิเศษในตาราง ASCII ทำให้เกิดความยุ่งยากในการส่งข้อมูล ถ้าอุปกรณ์ฝ่ายรับเกิดแปลไบต์ที่ไม่ใช่ข้อความว่าหมายถึงสิ้นสุดข่าวสาร ในกรณีนี้ข้อมูลไม่สามารถถูกส่งในรูปแบบข้อมูลดิบ เพราะว่าไบต์ที่อยู่กลางข่าวสารอาจตรงกับสัญลักษณ์สิ้นสุดข่าวสารโดยบังเอิญ และทำให้อุปกรณ์ฝ่ายรับหยุดรับข้อมูล

4.3 การแปลงเป็นรูปแบบอนุกรม

คอมพิวเตอร์เก็บทั้งหมดเก็บและจัดการข้อมูลในแบบขนาน หมายความว่าเมื่อไบต์หนึ่งถูกส่งจากส่วนหนึ่งของคอมพิวเตอร์ไปยังส่วนอื่น มันไม่ได้ถูกส่งไปส่งครั้งละหนึ่งบิต แต่จะถูกส่งไปหลายบิตพร้อมกันผ่านตัวนำในแบบขนาน จำนวนบิตที่ถูกส่งในครั้งหนึ่งแปรผันไปตามเครื่อง แต่โดยปกติจะเป็นแปดหรือทวิคูณของแปด เพราะฉะนั้น คอมพิวเตอร์สามารถทำงานกับหนึ่งไบต์เป็นอย่างน้อยในครั้งหนึ่ง

เนื่องจากการสื่อสารจากคอมพิวเตอร์ไปยังอุปกรณ์อื่นหลายชนิดเป็นแบบอนุกรม หมายความว่าข้อมูลถูกส่งไปทีละหนึ่งบิต ตัวเชื่อมต่อการสื่อสารต้องสามารถนำไบต์ที่รับมาแบบขนานส่งออกไปทีละบิตได้

จากที่กล่าวมาแล้วว่าสายข้อมูลในการสื่อสารแบบอนุกรม มีเพียงสถานะ MARK และ SPACE ซึ่งในกรณีของการเชื่อมต่อโดยตรงเท่ากับแรงดันไฟฟ้าลบหรือบวกตามลำดับ ข้อมูลใด ๆ ที่ถูกส่งต้องถูกแปลงให้เป็นลำดับของ MARK และ SPACE ก่อน สำหรับการส่งข้อมูล MARK แทนค่าหนึ่ง และSPACE แทนค่าศูนย์

4.4 การสื่อสารแบบซิงโครไนซ์และอะซิงโครไนซ์

เมื่อข้อมูลถูกแปลงให้เป็นรูปแบบอนุกรมแล้ว มีวิธีการส่งข้อมูลอยู่สองอย่างคือ ซิงโครไนซ์(Synchronous) และ อะซิงโครไนซ์(Asynchronous)

เมื่อข้อมูลถูกส่งมาจากการพิมพ์ที่เป็นพิมพ์ การส่งและรับจะเป็นแบบอะซิงโครไนซ์ คือคนที่พิมพ์ไม่สามารถที่จะพิมพ์ได้อย่างต่อเนื่อง ดังนั้นเมื่อคอมพิวเตอร์รับตัวอักษรแต่ละตัวจะมีช่องว่างระหว่างตัวอักษรที่ไม่สม่ำเสมอ ทำให้อุปกรณ์ฝ่ายรับไม่อาจคาดหมายได้ว่า ตัวอักษรต่อไปจะมาถึงเมื่อใด จากการขาดความต่อเนื่องนี้จึงมีความจำเป็นต้องใส่บิตพิเศษก่อนและหลังตัวอักษรแต่ละตัวเพื่อบ่งบอกจุดเริ่มต้นและสิ้นสุดของตัวอักษรบิตพิเศษนี้ว่า บิตเริ่มต้น (Start bit) นอกจากนี้

ยังมีอีกหนึ่งคือ บิตพาริตี(Parity bit) ที่มักจะถูกใส่เพิ่มเข้าไปเพื่อใช้ตรวจสอบความผิดพลาด วิธีนี้เรียกว่า การสื่อสารแบบอะซิงโครนัส(Asynchronous Communication)

เมื่อตัวอักษรถูกส่งไปเป็นกลุ่มตามความเร็วของเครื่อง ช่วงห่างระหว่างกันก็จะสม่ำเสมอ จึงไม่มีความจำเป็นต้องมีบิตเริ่มต้นและบิตจบสำหรับตัวอักษรแต่ละตัว เพราะว่าเมื่อตัวอักษรแรกถูกรับไป อุปกรณ์ฝ่ายรับสามารถคาดหมายการมาถึงของตัวอักษรถัดไปได้ กล่าวอีกหนึ่งคือ มันสามารถเข้าจังหวะตัวมันเองกับคอมพิวเตอร์ฝ่ายส่งได้ วิธีแบบนี้เรียกว่า การสื่อสารแบบซิงโครนัส (Synchronous communication)

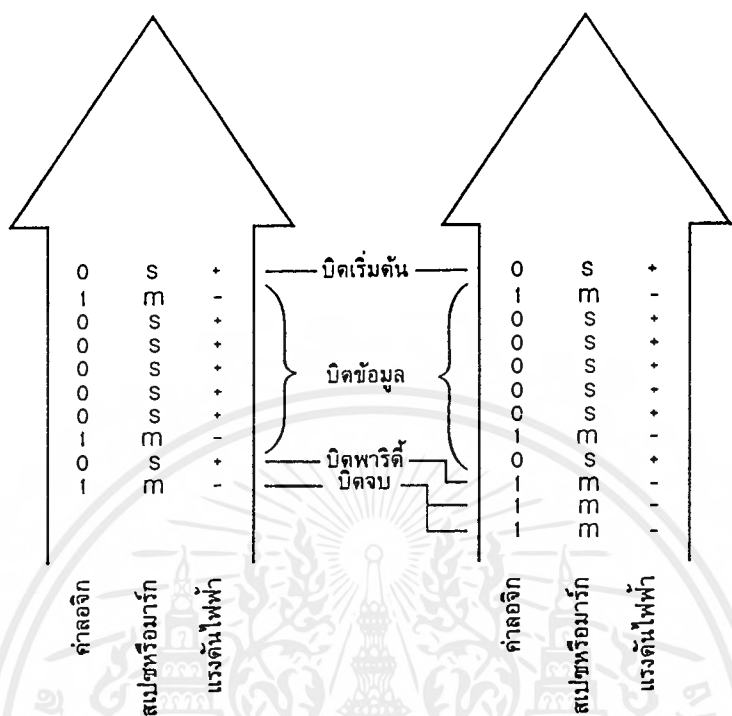
เนื่องจากการสื่อสารแบบอะซิงโครนัสต้องการบิตเริ่มต้นและบิตจบเพิ่มเข้าไปในแต่ละตัวอักษร จึงมีความยาวในการส่งไฟล์มากกว่าการสื่อสารแบบซิงโครนัส ประมาณ 20 เปอร์เซ็นต์ ความแตกต่างนี้อาจสังเกตไม่เห็นเมื่อแหล่งข้อมูลที่ส่งมาจากการพิมพ์ที่เทอร์มินัล

นอกจากในโลกของไอบีเอ็มเมนเฟรมซึ่งเทอร์มินัลแบบซิงโครนัสเป็นอุปกรณ์สามัญ การสื่อสารแบบอนุกรมส่วนใหญ่เกิดขึ้นในแบบอะซิงโครนัส ซึ่งประยุกต์เข้ากับการสื่อสารเกือบทั้งหมดระหว่างไมโครคอมพิวเตอร์ เทอร์มินอล และระบบยูนิกซ์ ด้วยเหตุนี้ส่วนที่เหลือของบทนี้จะให้ความสนใจที่การสื่อสารแบบอะซิงโครนัสเป็นหลัก

การจัดเฟรม

ในกรณีการสื่อสารแบบอะซิงโครนัส บิตที่เป็นตัวแทนของหนึ่งไบนารี ซึ่งเรียกว่า บิตข้อมูล (Data bit) จะถูกนำและตามด้วยบิตเริ่มต้น บิตจบและบิตพาริตี กระบวนการนี้เรียกว่า การจัดเฟรม (Framing)

จำนวนของบิตที่แทนหนึ่งตัวอักษรแปรผันไปตามโปรโตคอลสื่อสารที่ใช้ จำนวนที่ว่าหมายถึงจำนวนของบิตของข้อมูล หรือความยาวเวิร์ด (Word length) โดยปกติจะเป็นเจ็ดหรือแปดบิต แต่ละตัวอักษรจะถูกส่งออกไปเป็นกลุ่มที่ประกอบด้วยบิตเริ่มต้น ตัวอักษร (บิตข้อมูล) บิตพาริตีซึ่งสามารถเลือกได้ และบิตจบหนึ่งหรือสองบิต เพื่อความชัดเจน เราจะเรียกกุ่มของตัวอักษรและบิตเหล่านี้ว่าเฟรม (Frame) เพื่อหลีกเลี่ยง ความสับสนกับคำว่าตัวอักษร ที่บางครั้งอ้างถึงบิตข้อมูล และบางครั้งอ้างถึงทั้งกลุ่มพร้อมด้วยบิตเริ่มต้น บิตจบ และบิตพาริตี ตัวอย่างของเฟรมที่ถูกส่งแสดงไว้ในรูปที่ 4.2



รูปที่ 4.2 ตัวอย่างการส่งตัวอักษร A สองแบบ

บิตเริ่มต้น

บิตเริ่มต้นถูกใส่เพิ่มที่จุดเริ่มต้นของเฟรมเสมอ เพื่อเตือนอุปกรณ์ฝ่ายรับว่าข้อมูลกำลังมาถึง และเพื่อเข้าจังหวะกลไกที่แยกแต่ละบิต บิตเริ่มต้นคือ Space หรือ ไบนารี 0

ในการเชื่อมต่อโดยตรง SPACE หรือ 0 ถูกส่งเป็นแรงดันไฟฟ้าบวก แรงดันไฟฟ้าระหว่างเฟรมจะเป็นลบ ดังนั้นที่จุดเริ่มต้นของแต่ละเฟรมแรงดันไฟฟ้าจะเปลี่ยนจากลบเป็นบวก

บิตข้อมูล

มาตรฐานหรือโปรโตคอลการสื่อสารแบบอนุกรม ทำให้เกิดการส่งตัวอักษรที่ยาวต่างกัน เมื่อซอฟต์แวร์สื่อสารให้คุณเลือกความยาวเวิร์ด มันกำลังถามว่าคุณต้องการส่งตัวอักษรเจ็ดบิตหรือแปดบิต (บางครั้งความยาวอื่นก็ถูกใช้แต่แทบไม่ค่อยมี) ถ้าข้อมูลทั้งหมดถูกส่งในรูปแบบ ASCII เวิร์ดขนาดเจ็ดบิตก็เพียงพอ จำไว้ว่าตาราง ASCII กำหนดจำนวนจาก 0 ถึง 127 ซึ่งทั้งหมดสามารถแทนได้ด้วยเจ็ดบิต

ถ้าข้อมูลที่ส่งไม่ใช่ ASCII (เช่นข้อความที่ใช้ชุดอักขระเพิ่มเติมหรือข้อมูลไบนารี) ทั้งแปดบิตของแต่ละไบต์จึงมีความจำเป็น คุณไม่สามารถใช้โปรโตคอลเจ็ดบิตได้ ถ้าข้อมูลไม่ถูกแปลงเป็นรูปแบบเจ็ดบิตเสียก่อน

บิตพาริตี

การตรวจสอบพาริตีเป็นวิธีหนึ่งในการทดสอบว่า ข้อมูลที่ส่งได้ถูกรับไปอย่างถูกต้องหรือไม่ อุปกรณ์ฝ่ายส่งจะเพิ่มบิตพาริตีอีกหนึ่งบิตเป็นค่า 0 หรือ 1 ขึ้นอยู่กับบิตข้อมูล อุปกรณ์ฝ่ายรับจะตรวจสอบว่าบิตพาริตีมีความสัมพันธ์ที่ถูกต้องกับบิตอื่นหรือไม่ ถ้าไม่แสดงว่าบางสิ่งต้องผิดพลาดในระหว่างการส่ง พาริตีสามารถคำนวณได้จากวิธีต่อไปนี้

พาริตีคู่(Even Parity) หมายความว่า จำนวนของบิตข้อมูลที่เป็น 1 และค่าของบิตพาริตีรวมกัน เป็นจำนวนคู่ เช่น ตัว A ในฐานสองคือ 01000001 เมื่อนับจำนวนของบิตที่เป็น 1 จะได้ 2 ซึ่งเป็นเลขคู่ ดังนั้นบิตพาริตีต้องเป็น 0 ถ้าตัวอักษร A ที่รับได้มีพาริตีเป็น 1 แสดงว่าเกิดความผิดพลาดในระหว่างการส่ง

พาริตีคี่(Odd Parity) หมายความว่า จำนวนทั้งหมดของบิตข้อมูลที่เป็น 1 บวกกับค่าของบิตพาริตีเป็นจำนวนคี่ ดังนั้นสำหรับตัวอักษร A บิตพาริตีควรถูกเซตเป็น 1 เพื่อให้จำนวนของบิตที่เป็น 1 ทั้งหมดเป็น 3 ซึ่งเป็นจำนวนคี่

- **ไม่มีพาริตี(Null Parity)** หมายถึง ไม่มีบิตพาริตี

SPACE (บางครั้งเรียกว่า Bit Trimming) คือบิตพาริตีที่เป็น 0 เสมอ มีประโยชน์ในการตรวจสอบข้อผิดพลาดบางอย่าง เมื่อการส่งข้อมูลเป็นขยะมาก บางครั้งบิตพาริตีอาจกลายเป็น 1 แสดงว่า เกิดข้อผิดพลาด พาริตีแบบนี้สามารถใช้เพื่อส่งอักษรเจ็ดบิตให้กับอุปกรณ์ที่ต้องการตัวอักษรแปดบิตได้เช่นกัน อุปกรณ์ฝ่ายรับจะถือว่าบิตพาริตีเป็นบิตสุดท้ายของข้อมูล

MARK (บางครั้งเรียกว่า Bit Forcing) ทำงานเหมือนกับพาริตีแบบ SPACE ยกเว้นแต่บิตพาริตีจะเป็น 1 เสมอ เนื่องจาก 1 ในตำแหน่งนั้นสามารถที่จะถูกตีความรวมเข้ากับค่าของจำนวนได้ อุปกรณ์ หรือ คอมพิวเตอร์ฝ่ายรับต้องถูกโปรแกรมไม่ให้สนใจมัน

บิตจอบ

ที่ท้ายของแต่ละเฟรม บิตจอบจะถูกส่งออกมา บิตจอบมีทั้งแบบหนึ่งบิต หนึ่งบิตครึ่ง หรือ สองบิต อย่างน้อยต้องมีหนึ่งบิตเสมอ เพื่อประกันว่ามีแรงดันไฟฟ้าลงอย่างน้อยเป็นช่วงเวลาหนึ่ง ก่อนที่เฟรมถัดไปจะมาถึง เพื่อที่จะสามารถแยกแยะเฟรมถัดไปได้จากบิตเริ่มต้นที่เป็นบวกของมัน บิตจอบมากกว่าหนึ่งบิตโดยทั่วไปจะใช้เมื่ออุปกรณ์ฝ่ายรับต้องการเวลาเพิ่มขึ้นก่อนที่มันจะสามารถจัดการกับตัวอักษรที่เข้ามาตัวถัดไปได้

หนึ่งบิตครึ่ง หมายความว่า ความยาวของบิตนั้นมากกว่าบิตปกติ บิตจอบบังคับให้มีช่องว่างอย่างน้อยระหว่างเฟรม พวกมันถูกส่งเป็นไบนารีหนึ่งซึ่งในการเชื่อมต่อโดยตรงจะเป็นแรงดันไฟฟ้าลบ

บิตจบสองบิตมักจะถูกใช้ที่อัตราบอด 110 ซึ่งเป็นอัตราการส่งข้อมูลต่ำสุดที่ใช้กันทั่วไป เพื่อให้สอดคล้องกับความต้องการของเครื่องโทรพิมพ์รุ่นเก่า ซึ่งใช้อัตราบอดต่ำและต้องการเวลาพิเศษเพื่อประมวลตัวอักษร

เบรก

ดังที่ได้อธิบายมาก่อนเมื่อกล่าวถึงบิตเริ่มต้นว่า ระหว่างตัวอักษร สายข้อมูลโดยปกติอยู่ในสถานะ MARK (แรงดันไฟฟ้าลบ, ไบนารีหนึ่ง) ถ้าตัวอักษรประกอบด้วยศูนย์ทั้งหมด พร้อมด้วยแปดบิตข้อมูลและพาริตีคู่ สถานะ SPACE จะปรากฏอยู่สิบบิตคือ บิตเริ่มต้น บิตข้อมูลทั้งแปดและบิตพาริตี ซึ่งเป็นสถานะ SPACE ที่ยาวที่สุด ก่อนจะสิ้นสุดเมื่อถึงบิตจบ ดังนั้นที่อัตรา 150 บิตต่อวินาที สถานะ SPACE ตามปกติจะไม่มากไปกว่า $1/15$ วินาที หรือ 66.67 มิลลิวินาที

สถานะ SPACE ที่นานกว่านี้ โดยปกติเป็น 100 ถึง 600 มิลลิวินาที ถูกใช้เป็นสัญญาณพิเศษเรียกว่า เบรก (Break) เบรกบางครั้งถูกใช้เสมือนกับ Ctrl-C ของเมนเฟรมบนพีซี มันจะขัดจังหวะไม่ว่าโปรแกรมอะไรกำลังทำงานอยู่ และกลับคืนสู่ระบบปฏิบัติการ หรือ เมนูที่อยู่ในระดับบนภายในโปรแกรม เช่นเดียวกับ Ctrl-C หรือ Break มันมีประโยชน์สำหรับการหนีออกจากโปรแกรมที่เข้ามาลูบไม่รู้จบ

อัตราบอด

อัตราบอด (Baud rate) แสดงจำนวนของสัญญาณแต่ละหน่วยในหนึ่งหน่วยวินาที มันถูกตั้งชื่อตาม Baudot ซึ่งเป็นผู้บุกเบิกการสื่อสารชาวฝรั่งเศส ในการส่งแบบไบนารีมันเป็นสิ่งเดียวกับบิตต่อวินาที (bps) หรือ จำนวนของเลขฐานสองที่ถูกส่งในหนึ่งวินาที ทั้งสองคำนี้มีความแตกต่างกัน แต่มันมักจะทำให้สับสน ผู้คน 200,000 คน อาจบอกว่าพวกเขามีโมเด็ม 1200 บอด และไม่มีสักคนที่จริง ๆ ที่จริงแล้วพวกเขามีโมเด็ม 1200 bps

ในการเชื่อมต่อ RS-232 โดยตรง สัญญาณจะเป็นหนึ่งในสองสถานะ ในเวลาขณะใดขณะหนึ่ง อัตราบอดและ bps จึงเท่ากัน อย่างไรก็ตาม ในบทที่ 4 เราจะเห็นว่าเมื่อสัญญาณหนึ่งถูกส่งผ่านระหว่างโมเด็ม มันสามารถเป็นหนึ่งในหลายสถานะ ความยาวของสัญญาณอาจเป็น $1/600$ วินาที (600 บอด) แต่เนื่องจากมากกว่าสองบิตของข้อมูลสามารถถูกส่งไปพร้อมกับการเปลี่ยนแปลงแต่ละสถานะ อัตราบิตต่อวินาทีจะสูงกว่าอัตราบอด

มีจุดน่าสังเกตคือทั้งอัตราบอด และ bps อ้างถึงอัตราที่บิตภายในหนึ่งเฟรมถูกส่ง ช่องว่างระหว่างเฟรมอาจมีความยาวแปรเปลี่ยนได้ เช่น จากการพิมพ์ตัวอักษรด้วยอัตราแตกต่างกัน ดังนั้นทั้งอัตราบอดและ bps จึงไม่ได้หมายถึงอัตราที่ข้อมูลถูกส่งไปจริง ๆ

การแก้ปัญหา

เมื่ออุปกรณ์สองตัวสื่อสารซึ่งกันและกัน พวกมันต้องตกลงกันในเรื่องอัตราบอด ความยาวเวิร์ด จำนวนบิตจบ และ พาริตี ถ้าพบว่าไม่ได้รับอะไรเลย ความผิดพลาดอาจอยู่ที่การเชื่อมต่อทางกายภาพ เช่น ข้อมูล กำลังถูกส่งบนสายผิดเส้น สายขาด หรือไม่ได้รับสัญญาณแฮนด์เช็กที่ถูกต้อง ถ้าได้รับขยะ ความผิดพลาดอาจอยู่ในหัวข้อที่จะกล่าวต่อไปนี้

อัตราบอดไม่ตรงกัน

ถ้าอุปกรณ์สองตัวถูกตั้งอัตราบอดต่างกัน อุปกรณ์ฝ่ายรับอาจพยายามที่จะแปลข้อมูล (ถ้ามันไม่ได้ถูกโปรแกรมให้รายงานข้อผิดพลาดทางพาริตีและทางเฟรม) โดยปกติคุณจะเห็นว่าจำนวนข้อมูลที่รับได้แตกต่างจากที่ถูกส่งมา

ความผิดพลาดทางพาริตี

ความผิดพลาดทางพาริตี(Parity error) บ่งบอกว่าข้อมูลถูกทำลายในระหว่างการส่ง อย่างไรก็ตาม มันอาจหมายความว่าอุปกรณ์ทั้งสองไม่ได้ถูกตั้งให้มีพาริตี (คู่,คี่, หรือ ไม่มี) หรือความยาวเวิร์ดตรงกัน

ความยาวเวิร์ดไม่ตรงกัน

ถ้าเวิร์ดขนาดแปดบิตกำลังถูกส่งและอุปกรณ์ฝ่ายรับคาดหวังที่จะรับเวิร์ดขนาดเจ็ดบิต คุณอาจไม่พบความแตกต่างในการส่งข้อความ เพราะว่าเพียงแค่เจ็ดบิตแรกที่มีนัยสำคัญ เนื่องจากบิตศูนย์ถูกส่งก่อน และบิตเจ็ดไม่ถูกใช้ในการส่ง ASCII ปกติการขาดหายไปของมันจึงไม่มีความสำคัญอย่างไรก็ตาม อุปกรณ์ฝ่ายรับอาจพยายามแปลความหมายบิตที่เกินมาเป็นบิตพาริตีคี่ และรายงานข้อผิดพลาด ดังนั้นข้อผิดพลาดทางพาริตีจึงไม่จำเป็นที่จะต้องหมายถึงข้อมูลถูกทำลายในการส่ง มันอาจบอกถึงความยาวเวิร์ดไม่ตรงกันก็ได้

ถ้าส่งเวิร์ดขนาดเจ็ดบิต โดยที่อุปกรณ์ฝ่ายรับต้องการเวิร์ดขนาดแปดบิต บิตพาริตีอาจถูกนำไปรวมเป็นบิตที่แปด เนื่องจากบิตพาริตีอาจเป็น 1 สำหรับตัวอักษรครึ่งหนึ่ง และเป็น 0 สำหรับครึ่งหนึ่ง จึงพบได้บ่อยครั้งว่าอุปกรณ์ฝ่ายรับจะแสดงอักขระเพิ่มเติม เช่น อักขระกราฟิกในจำนวนครึ่งหนึ่งของตัวอักษรที่รับได้

บิตจบ

ไม่ควรจะมีปัญหาถ้าบิตจบสองบิตถูกส่งมา แม้มีเพียงบิตเดียวที่ต้องการ บิตจบที่เกินมาเพียงแค่รวมเข้าในช่องว่างระหว่างตัวอักษร อย่างไรก็ตามการส่งหนึ่งบิตจบ เมื่อต้องการสองบิต อาจทำให้เกิดปัญหาขึ้นอยู่กับคุณลักษณะของอุปกรณ์ฝ่ายรับ เรื่องนี้ไม่เป็นปัญหากับอุปกรณ์โมเด็ม

ความผิดพลาดทางเฟรม

ความผิดพลาดทางเฟรม บ่งบอกความไม่ตรงกันของจำนวนบิต ซึ่งมักจะถูกรายงาน เมื่อไม่ได้รับบิตจบตามที่คาดหวัง

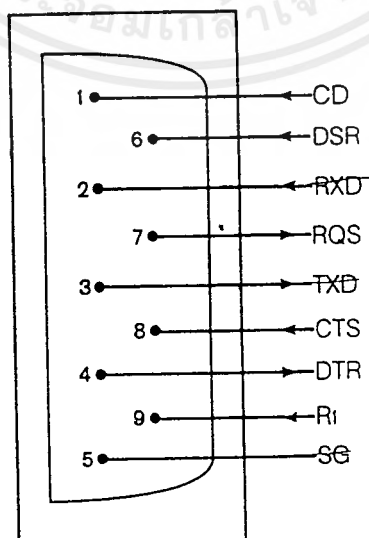
4.5 สัญญาณทางไฟฟ้า

มาตรฐาน RS-232-C กำหนดคุณลักษณะของสัญญาณทางไฟฟ้าที่ใช้ในการเชื่อมต่ออนุกรมโดยตรง มีเพียงสองลักษณะคือ SPACE แสดงถึงไบนารี 0 หรือแรงดันไฟฟ้าบวกและ MARK แสดงถึงไบนารี 1 หรือแรงดันไฟฟ้าลบ

บนสายข้อมูล(เช่นสาย 2 และ 3)แรงดันไฟฟ้าบวกแสดงถึงค่าลอจิก(logic) 0 และแรงดันไฟฟ้าลบแสดงถึงค่าลอจิก 1 บนสายแฮนด์เช็กกิ้ง(เช่น DTR และ DRS)แรงดันไฟฟ้าบวกแสดงว่าส่งข้อมูลได้ ส่วนแรงดันไฟฟ้าลบหมายถึงหยุดส่งข้อมูล

แรงดันไฟฟ้าบวก(สถานะ SPACE)อยู่ระหว่าง +5 ถึง -5 V สำหรับเอพาร์ทูท และระหว่าง +3 ถึง +15 V สำหรับอินพุท ความแตกต่างมีไว้เพื่อกรณีที่แรงดันไฟฟ้าสูญหายเนื่องจากความยาวของสายสัญญาณ ในทำนองเดียวกัน แรงดันไฟฟ้าลบ(สถานะ MARK)ถูกกำหนดไว้ระหว่าง -5 ถึง -15 V สำหรับเอพาร์ทูท และ -3 ถึง -15 V สำหรับอินพุท

สังเกตว่า ถ้าสัญญาณยาวเกินไป ระดับแรงดันไฟฟ้าจะตกลงเกินขอบเขตที่ยอมรับได้นอกจากนี้ ความจุไฟฟ้าที่เกิดขึ้นจะมีผลกับคุณภาพของสัญญาณ โดยทำการเปลี่ยนสถานะจากแรงดันไฟฟ้าบวกไปลบไม่ชัดเจน RS-232-C ไม่ได้มุ่งหวังให้นำไปใช้กับระยะไกล และโดยทั่วไป 50 ฟุตเป็นระยะทางไกลที่สุดในการใช้สัญญาณปกติที่อัตราการส่งข้อมูลปกติ ถ้าอุปกรณ์อยู่ห่างกันมาก อาจจำเป็นต้องใช้โมเด็มหรือวิธีการอื่น



รูปที่ 4.3 หัวต่อแบบ DB-9 ในไอบีเอ็มพีซีเอที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5

Delphi

5.1 โครงสร้างทั่วไปของเดลไฟ

เดลไฟเป็นคอมไพเลอร์ที่มีจุดเริ่มต้นมาจากภาษาและพัฒนามาเป็นออบเจกต์ปาสคาลและสุดท้ายเป็นเดลไฟซึ่งได้นำความสามารถในการจัดการฐานข้อมูลเอาไว้ด้วยจึงเหมาะสำหรับผู้ที่ต้องการพัฒนาโปรแกรมต่างๆ ไปและโปรแกรมเกี่ยวกับการจัดการฐานข้อมูล สำหรับผู้ที่มีความรู้ทางด้านภาษาปาสคาลแล้วสามารถศึกษาโครงสร้างของออบเจกต์ปาสคาลเพิ่มเติมและศึกษาโครงสร้างการทำงานของเดลไฟก็จะสามารถพัฒนาโปรแกรมได้เลย

ภาษาปาสคาลเป็นภาษาคอมพิวเตอร์ที่มีโครงสร้างที่ตายตัวแล้วเมื่อมีการปรับปรุงให้มีการเขียนโปรแกรมในลักษณะของออบเจกต์ด้วยแล้วทำให้คอมไพเลอร์ตัวนี้มีความสามารถอย่างไม่มีขีดจำกัดบนระดับปฏิบัติการวินโดวส์ อยู่กับโปรแกรมเมอร์จะดึงเอาความสามารถของคอมไพเลอร์ออกมาได้มากหรือน้อยเท่านั้นเอง

การเขียนโปรแกรมแบบออบเจกต์จะช่วยลดระยะเวลาในการเขียนโปรแกรมเพราะสามารถดึงโมดูลหรือฟังก์ชันกลับมาใช้ใหม่ได้เรื่อยๆ ซึ่งออบเจกต์ต่างๆ จะถูกเก็บเอาไว้เป็นระดับเดียวที่เรียกว่า Class ซึ่งจะเก็บฟังก์ชันและโมดูลต่างๆ เอาไว้และเมื่อต้องการนำมาใช้จะต้องใช้สร้างออบเจกต์ขึ้นมาซึ่งมีได้ของการสร้างออบเจกต์ดังนี้

Type

Tform1=class (Tform)//กำหนดชนิดของออบเจกต์

Tform1เป็นออบเจกต์ที่สร้างมาจาก Class Tform ซึ่ง Tform1ก็จะกลายเป็นออบเจกต์ที่สามารถนำไปสร้างเป็นแอปพลิเคชันได้โดยจะต้องจองตัวแปรชนิดออบเจกต์ขึ้นมาก่อน เช่น

Var

Form1:Tform1; //Form1 เป็นตัวแปรออบเจกต์ชนิด Tform1ซึ่งภายในออบเจกต์ที่กำหนดมาสามารถมีฟิลด์(ฟิลด์จะเป็น ออบเจกต์ โมดูล หรือ ตัวแปรก็ได้)ซึ่งจะต้องกำหนดต่อจากการกำหนดชนิดของออบเจกต์ เช่น

Type

Tform1=class(Tform)

Button1:Tbutton;

Procedure Button1Click(Sender:Tobject);

จากตัวอย่างดังกล่าว TForm1 ออบเจกต์ประกอบด้วยฟอยล์ Button ซึ่งเป็นออบเจกต์ชนิด Tbutton และ โมดูล(โพรซีเยอร์) Button1Click

ความจริงแล้วโปรแกรมเมอร์จะต้องสร้างโค้ดขึ้นมาเองทั้งหมดซึ่งอาจจะทำให้เสียเวลาเคลฟจึงสร้างโค้ดขึ้นมาให้อัตโนมติ โปรแกรมเมอร์เพียงแต่เลือกออบเจกต์มาใช้นั่นเองซึ่งทำให้โปรแกรมเมอร์สมัครเล่นสามารถฝึกทักษะตัวเองให้เป็นโปรแกรมมีอาชีพได้ด้วยเคลฟสร้างออบเจกต์เอาไว้มากมายเพียงพอสำหรับการพัฒนาโปรแกรมต่างๆไปและโปรแกรมทางด้านคอมพิวเตอร์ (คอมพิวเตอร์ก็คือออบเจกต์นั่นเองเมื่อนำมาวางในแอปพลิเคชัน)แต่ละตัวก่อน

5.1.1 การเขียนโปรแกรมแบบ Event-Driven

คือการเขียนโปรแกรมเพื่อรองรับเหตุการณ์ต่างๆเช่นการคลิกเมาส์ หรือการกดคีย์บอร์ดซึ่งเป็นหน้าที่ของโปรแกรมเมอร์ที่จะต้องเขียนโค้ดลงไปเพื่อควบคุมการทำงานของโปรแกรม

Visual two Way คือการเขียนโปรแกรมที่สามารถทำได้ทั่วไปก็ได้ ซึ่งโดยทั่วไปแล้วจะทำทั้งสองอย่างจะขึ้นอยู่กับความถนัดของแต่ละคนด้วย

การสร้างแอปพลิเคชันสำหรับจัดการกับฐานข้อมูลนั้น โปรแกรมเมอร์จะต้องเข้าใจการทำงานของดาต้าเซต(Table,Query StoredProc)และ Data Control คอมพิวเตอร์ก่อน

Code Wizard ช่วยให้ทราบถึงฟังก์ชันและโพรซีเยอร์ภายในออบเจกต์นั้นได้เป็นอย่างดีโดยที่เราไม่ต้องไปจำในส่วนนั้นๆ Code Wizard จะปรากฏขึ้นเมื่อเขียนเครื่องหมาย (.) ลงไปหลังชื่อออบเจกต์นั้นๆ

Help ช่วยได้เสมอเมื่อเกิดคีย์ F1 โดยเฉพาะอย่างยิ่งช่วยในการศึกษาฟังก์ชันและโพรซีเยอร์ซึ่งมีมากมายจนไม่สามารถที่จะจำได้หมด อินเทอร์เน็ตแหล่งข้อมูลมากมายช่วยให้คุณได้เสมอ ทั้ง Homepage หรือ NewsGroup ต่างๆ

การนำคอมพิวเตอร์มาวางบนฟอร์ม

เมื่อต้องการสร้างแอปพลิเคชันจากเคลฟคุณจะต้องนำคอมพิวเตอร์ที่มีอยู่แล้ววางบนฟอร์มเซตพารามิเตอร์และเขียนโค้ดลงในอีเวนท์ของแต่ละคอมพิวเตอร์นั้นๆ คอมพิวเตอร์ก็คือส่วนที่มองเห็นได้ในแต่ละแอปพลิเคชันเช่น บัทตอน สลอลด์บอช และบางคอมพิวเตอร์ที่ไม่สามารถเห็นได้แต่มีส่วนช่วยในการรันแอปพลิเคชันเช่นการเรียกข้อมูลจากฐานข้อมูล การคำนวณและการติดต่อภายในระบบปฏิบัติการทุกคอมพิวเตอร์จะปรากฏอยู่ใน Delphi Visual Component library การเปลี่ยนคุณสมบัติและพฤติกรรมของแต่ละคอมพิวเตอร์

การเปลี่ยนพารามิเตอร์และอีเวนท์ของแต่ละคอมพิวเตอร์สามารถทำได้โดยการใช Object Inspector เมื่อคอมพิวเตอร์ถูกไฟกัสที่ฟอร์ม(คอมพิวเตอร์ดังกล่าว Active อยู่ในขณะรัน-โปรแกรม)

การดูอีเวนท์(พรอพเพอร์ตี้และอีเวนท์)

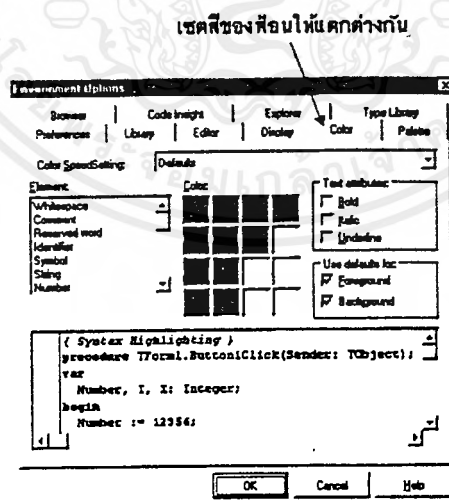
คลิกที่อีเวนท์แท็บ (EvenTab) ในออบเจกต์อินสเปกเตอร์เพื่อสำรวจว่าภายในคอมโพเนนท์นั้นมีอีเวนท์อะไรบ้างตัวอย่างเช่น Button4 เมื่อมีการคลิกให้ถูกโฟกัสบนฟอร์มแล้วชนิดของออบเจกต์ดังกล่าวจะแสดงออกมาเป็น Tbutton ที่ออบเจกต์เซเลกเตอร์ภายในออบเจกต์อินสเปกเตอร์ เมื่อต้องการเขียนโค้ดลงในแต่ละอีเวนท์ให้ดับเบิลคลิกที่ช่องขวาของแต่ละอีเวนท์นั้นๆหรือดับเบิลคลิกออบเจกต์ที่อยู่บนฟอร์มเคลไฟจะสร้างโค้ดที่เซตเอาไว้ในเบื้องต้น (Default) ขึ้นมาในโค้ดอีดิเตอร์ให้อัตโนมติ

5.1.2 การเขียนโค้ดในโค้ดอีดิเตอร์

เมื่อเราสร้างแอปพลิเคชันขึ้นมา สร้างโค้ดอัตโนมัติสำหรับแอปพลิเคชันนั้นๆซึ่งเป็นภาษาปาสคาล แต่เราสามารถแก้ไขพรอพเพอร์ตี้ หรือสร้างเงื่อนไขของแอปพลิเคชันโดยการใส่โค้ดลงในโค้ดอีดิเตอร์ นอกจากนั้นแล้วยังสามารถใช้ Help ได้อย่างรวดเร็วโดยการกดคีย์ F1 หรือต้องการทราบรายละเอียดของคำต่างๆภายในโค้ดอีดิเตอร์ก็ให้เลื่อนไปที่หน้าค่านั้นๆแล้วกดคีย์ F1

การแก้ไขรูปแบบการแสดงผลของโค้ดอีดิเตอร์

เนื่องจากโค้ดของเคลไฟมีคำสั่งที่มีชนิดไม่เหมือนกันคุณสามารถให้แสดงคำจำพวกนั้นให้มีคุณสมบัติของตัวอักษรได้เช่นการตั้งสีของคำศัพท์สวจนให้มีสีแตกต่างจากคำที่เป็นสตริง หรือตั้งขนาดของตัวอักษรให้มีความแตกต่างกัน



รูปที่ 5.1 แสดงการเซตสีของฟอนท์ให้แตกต่างกัน

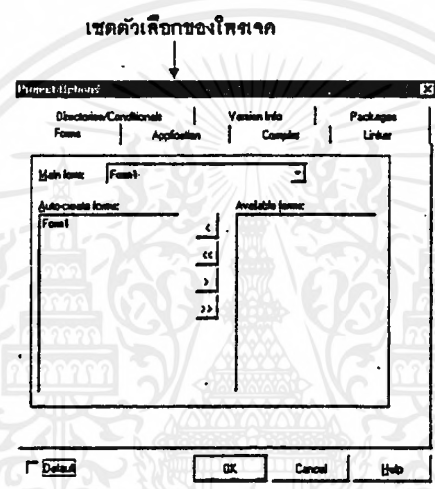
5.1.3 สำรวจฐานข้อมูล

ดาตาเบสเอกโพรเลอร์ (DataBase Explorer) ให้คุณตรวจสอบฐานข้อมูลไม่ว่าจะเป็นการ สร้างเอเลียส(Alias) การสร้างฐานข้อมูลเช่น Table Stored Procedure และ Triggers คู่ออบเจกต์ของเอกสารนี้เป็นเอกสารที่สวจนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ไฟล์จะเก็บซอร์ทโค้ดเอาไว้ และในหนึ่งโปรเจกจะมีโปรเจกไฟล์เดียวเท่านั้นส่วนยูนิคไฟล์จะมีมากกว่าหนึ่งไฟล์ก็ได้

การตั้งตัวเลือกของโปรเจก(Setting Project Option)

คุณสามารถเปลี่ยนค่าตัวเลือกภายในโปรเจกเพื่อสร้างแอปพลิเคชันให้เหมาะสมได้โดยการเลือกเมนู Project Option การเซตค่าต่างๆจะถูกเก็บไว้ในไฟล์ไฟล์หนึ่งซึ่งจะมีนามสกุลเป็น .DOF ซึ่งเมื่อคุณเปิดโปรเจกจะดูค่าตัวเลือกต่างๆภายในโปรเจกจากไฟล์นี้



รูปที่ 5.4 แสดงการเซตตัวเลือกของโปรเจก

5.1.5 แอปพลิเคชัน(Application Options)

ในหน้าของแอปพลิเคชันคุณสามารถกำหนดไทเทิล(Title)ของแอปพลิเคชัน Help ไฟล์ และไอคอนของโปรเจกไทเทิล ข้อความในไทเทิลจะปรากฏข้างล่างของแอปพลิเคชันเมื่อคุณ minimize แอปพลิเคชันในขณะรัน แต่ถ้าในไทเทิลไม่มีการกำหนดค่าให้ส่วนที่ปรากฏก็จะเป็นชื่อของแอปพลิเคชันนั้นซึ่งเป็น .EXE ไฟล์ ถ้าสร้าง Help file (.HLP)

สำหรับโปรเจกจะต้องกำหนดจากHelp fileอีติคบอกซ์ในตัวเลือกของโปรเจก Helpfilenameจะถูกใช้โดยhelpCommand, HelpContext และHelpJump method ของ Application-object ซึ่ง Help ของแอปพลิเคชันจะเรียก Windows API ฟังก์ชันโดยการส่งชื่อ Help ไฟล์ลงไป

Compiler Options

ในหน้าของตัวเลือกของคอมไพเลอร์จะมีตัวเลือกเพื่อใช้ประกอบในการสร้างโค้ด, ไวยากรณ์, runtime error, debugging และ message

Directory และ Conditional

ในส่วนนี้จะมีตัวเลือกสำหรับการกำหนดที่อยู่ของ Resource ที่ต้องการนำมาคอมไพล์ด้วย ลิงค์และการแบ่งแอฟลิเคชัน

Output Directory

การคอมไพล์โปรแกรมจะต้องมีการกำหนด Output Directory เพื่อเก็บ .EXE File หากไม่กำหนดในช่องของ Output Directory แล้วโปรแกรมจะใช้ไดเรกทอรีเดียวกันกับที่อยู่ของโปรเจคไฟล์

Unit Output Directory

หากมีการนำยูนิตอื่นไปคอมไพล์ด้วยจะต้องกำหนดที่อยู่ของยูนิตนั้นด้วยที่ UnitOutput-Directory อีดิเตอร์

Search Path

Search Path อีดิเตอร์เป็นที่กำหนดที่อยู่ของไลบรารีที่ไม่ใช่ไลบรารีของเดลไฟ เดลไฟคอมไพเลอร์จะเซตไดเรกทอรีที่ไม่ได้อยู่ในไดเรกทอรีดังกล่าวแล้วจะต้องกำหนดเองโดยใส่ไว้ใน

Search Path Directory

Version Information

Version Information ให้คุณกำหนดเวอร์ชันของโปรแกรม

Setting environment preferences

เลือกเมนู Tool Environment Options สำหรับการตั้งค่าให้กับโปรเจคไม่ว่าจะเป็น editor-designer debugger และ compiler

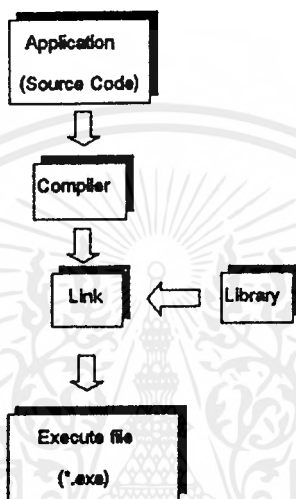
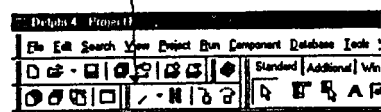
5.1.6 Compiling, Building และ Running โปรเจค

คุณสามารถตรวจสอบไวยากรณ์ในโปรเจคได้โดยไม่ต้องมีการคอมไพล์ หากตรวจพบการผิดพลาดของโค้ด โปรแกรมจะฟ้องเป็นเมสเสจออกมาโดยที่ไม่มีการสร้าง ออบเจกต์และ execute file การตรวจสอบไวยากรณ์ทำได้โดยเลือกเมนู Project Syntax Check หากตรวจพบการผิดพลาดของไวยากรณ์เกิดขึ้น โปรแกรมจะมีเมสเสจข้างล่างปรากฏขึ้นมา และบรรทัดภายในโค้ดอีดิเตอร์จะมีแถบสี หากต้องการทราบรายละเอียดของความผิดพลาดในแต่ละบรรทัดในโค้ดอีดิเตอร์ให้กดคีย์F1

5.1.6.1 การคอมไพล์โปรเจค

การคอมไพล์ทุกซอร์สโค้ดของโปรเจคจะต้องเลือกเมนู Project Compile เมื่อเลือกคำสั่งนี้ คอมไพเลอร์จะทำการแปลคำสั่งทุกคำสั่งในโค้ดอีดิเตอร์ ซึ่งวิธีดังกล่าวจะสร้างไฟล์ที่มีนามสกุลเป็น .DCU(Delphi compiled unit) หลังจากนั้นก็สร้างไฟล์ที่มีนามสกุลเป็น .OBJ ซึ่งเป็นซอร์สโค้ด

คลิก Run เพื่อทดสอบรันโปรแกรมไฟล์



รูปที่ 5.7 แสดงการรันโปรแกรม

5.1.7 ความหมายของคอมโพเนนต์

การเขียนโปรแกรมเซลล์ไฟแบบวิซวลทำให้คุณสามารถนำออบเจกต์ต่างมาประกอบกันเป็นแอปพลิเคชันได้อย่างรวดเร็ว การนำออบเจกต์ที่มีอยู่แล้วกลับมาใหม่ทำให้ไม่เสียเวลาในการพัฒนาโปรแกรม

5.1.8 ขั้นตอนการเขียนโปรแกรม

1. สร้างฟอร์ม
2. วางคอมโพเนนต์บนฟอร์ม
3. เซตพรอพเพอร์ตี้ของแต่ละคอมโพเนนต์
4. เขียนโค้ดลงในอีเวนต์ของแต่ละคอมโพเนนต์

ให้หัวข้อนี้จะอธิบาย

- อธิบายส่วนต่างๆที่อยู่บนฟอร์มและคอมโพเนนต์
- การเลือกใช้คอมโพเนนต์ได้ถูกต้องและเหมาะสม

ฟอร์ม

การออกแบบโปรแกรมทั้งหมดจะต้องกระทำลงบนฟอร์มโดยการวางคอมโพเนนต์ต่างๆ บนฟอร์มซึ่งฟอร์มที่คุณกำลังออกแบบอยู่จะกลายเป็นวินโดวส์และไคอะลอกซ์บ็อกซ์ของแอปพลิเคชัน บางคอมโพเนนต์ที่วางบนฟอร์มทำให้ผู้ใช้โปรแกรมสามารถตั้งงานกระทำกับแอปพลิเคชัน-คอมโพเนนต์บางส่วนก็จะกลายเป็นส่วนหนึ่งของแอปพลิเคชันไป เช่น ฐานข้อมูลเป็นต้น คอมโพเนนต์บางอย่างก็เป็นคอมโพเนนต์ชนิดพิเศษที่เรียกว่า Data Modules ซึ่งไม่สามารถมองเห็นได้ในขณะที่รัน โปรแกรมแต่ละตัวติดต่อกับฐานข้อมูลซึ่งทำให้ผู้ใช้โปรแกรมกระทำกับข้อมูลดังกล่าวได้

ดูเหมือนว่าจะมีความแตกต่างกันระหว่างฟอร์มและคอมโพเนนต์อื่นๆ ซึ่งความเป็นจริงไม่ใช่ ฟอร์มก็เป็นคอมโพเนนต์อันหนึ่งซึ่งสามารถเซตพรอพเพอร์ตี้และเขียนโค้ดเพื่อตอบโต้กับฟอร์มได้เช่นเดียวกับคอมโพเนนต์อื่นๆ

ภาพทั่วไปของคอมโพเนนต์

ทุกคอมโพเนนต์ที่วางบนฟอร์มก็จะกลายเป็นออบเจกต์ที่จะต้องติดต่อกับผู้ใช้โปรแกรม-ออบเจกต์ที่เป็นฐานข้อมูลหรือฟังก์ชันของระบบเช่น Timer ก็เช่นเดียวกันเมื่ออยู่บนฟอร์มแล้วก็จะทำให้ยูสเซอร์สามารถติดต่อกับแอปพลิเคชันได้ด้วย

คุณสมบัติของออบเจกต์ดังนี้

- เก็บรวบรวมฟังก์ชันและโพรซีเยอร์
- สามารถสืบทอดพฤติกรรมจากบรรพบุรุษได้
- สามารถแลกเปลี่ยนข้อมูลกับออบเจกต์อื่นๆ ได้

ถึงแม้ว่าคอมโพเนนต์แต่ละตัวจะมีรูปแบบที่เป็นลักษณะเฉพาะในแต่ละคอมโพเนนต์นั้นๆ แต่ทุกๆ คอมโพเนนต์จะมีคุณสมบัติบางอย่างที่เหมือนกัน เนื่องจากว่าทุกๆ คอมโพเนนต์สืบทอดมาจากออบเจกต์บรรพบุรุษซึ่งก็คือ Tcomponent ซึ่ง Tcomponent จะมีคุณสมบัติที่น้อยมากซึ่งเป็นคุณสมบัติที่ทุกคอมโพเนนต์ในเดลไฟจะต้องมีด้วยเช่นกัน

สามสิ่งสำคัญที่เกี่ยวกับคอมโพเนนต์

State information ซึ่งจะแสดงสภาพของคอมโพเนนต์ซึ่งก็คือพรอพเพอร์ตี้มันเอง เช่น ความสูงหรือสีของคอมโพเนนต์นั้นๆ

Action information โดยทั่วไปคอมโพเนนต์จะมี Action สำหรับรองรับการตอบสนองซึ่งจะเรียกว่าเป็น เมธอด(Method) ซึ่งเมธอดก็คือฟังก์ชันและโพรซีเยอร์มันเองโดยภายในคอมโพเนนต์จะต้องมีเมธอดต่างๆ รองรับความต้องการของผู้ใช้โปรแกรม ยกตัวอย่างเช่น การ Hide หรือ Refresh

1. คอมโพเนนต์ที่มองเห็นไม่ได้(Nonvisual component)
2. คอมโพเนนต์ที่มองเห็นได้ เรียกว่า Control Component

คอมโพเนนต์ที่ไม่สามารถมองเห็นได้ซึ่งจะเป็นส่วนหนึ่งภายในแอปพลิเคชันซึ่งผู้ใช้โปรแกรมไม่สามารถติดต่อได้โดยตรง เช่น System timer และคอมโพเนนต์ที่ช่วยในขณะที่ออกแบบโปรแกรมซึ่งสามารถเซตพารามิเตอร์และเขียนโค้ดลงในอีเวนทของนั้นวิซวลคอมโพเนนต์นั้นได้ แต่ก็มีบางนั้นวิซวลคอมโพเนนต์ที่สามารถมองเห็นได้ เช่น main menu bar หรือ ไอคอนต่างๆ

Control Component

เป็นคอมโพเนนต์ให้ผู้ใช้โปรแกรมสามารถติดต่อได้ในขณะที่รันโปรแกรมโดยทั่วไปแล้วจะมีลักษณะเหมือนกันกับคอนโทรลที่ออกแบบโปรแกรม control คอมโพเนนต์จะแบ่งออกเป็นกลุ่มคอมโพเนนต์ย่อยได้อีก คือ Windows Control เป็นคอมโพเนนต์ที่ผู้ใช้โปรแกรมสามารถโฟกัสมันได้ (การที่ออบเจกต์นั้นๆ Active อยู่) Graphical Control เป็นคอมโพเนนต์ที่ไม่สามารถเซตโฟกัสได้ การเลือกคอมโพเนนต์ให้เหมาะกับการใช้งาน

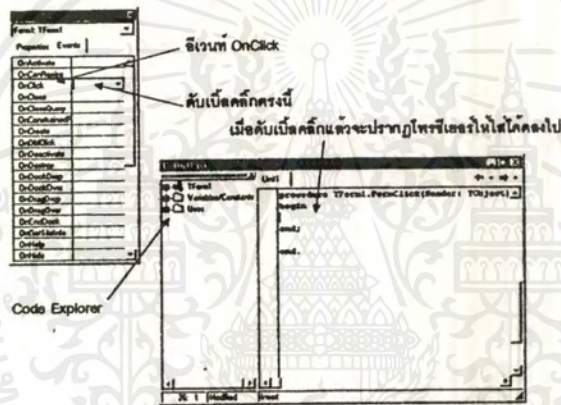
การเลือกคอมโพเนนต์ที่เหมาะสมกับการใช้งานจะต้องคำนึงถึงพารามิเตอร์ และ อีเวนทของแต่ละคอมโพเนนต์ซึ่งบางครั้งคุณซึ่งเป็นผู้ซึ่งเป็นผู้ที่เพิ่งเริ่มเรียนรู้ สามารถดูการแบ่งกลุ่มของคอมโพเนนต์สำหรับการใช้งานได้ดังนี้

- คอมโพเนนต์สำหรับการป้อนข้อมูลเป็นข้อความหรือบันทึกมีดังนี้ Edit box, Memo, Mask-edit, Rich edit, DBText, Db\Bedit, DBMemmo, DBRich edit
- คอมโพเนนต์สำหรับการป้อนข้อมูลแบบพิเศษ เช่น Scroll bar, Track bar, Up-down, hot-key, Splitter
- คอมโพเนนต์สำหรับการควบคุม เช่น Button, Check box, Radio button, Bitmap button, Speed button, DBCheck box, DB radio Button, Tab set tlo bar, Cool bar, Main menu, Pup-up menu
- คอมโพเนนต์แสดงข้อมูลออกมาเป็นลิสต์ของข้อมูลเช่น List box, Tree view, List view, DB-list box, DB checkbox, DB lookup , Combo box, Checklist box, Date/time picker, DBCombo Box
- คอมโพเนนต์สำหรับการจัดกลุ่มเช่น Group box, Radio group, Panel, Scroll box, Tab control, Page control, Header control, DBRadio group
- คอมโพเนนต์แบบ Visual Feedback เช่น label, progress bar, status bar, panel, static text
- คอมโพเนนต์แบบ Tabular display เช่น String grid, Draw grid, DB grid, DB control grid

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



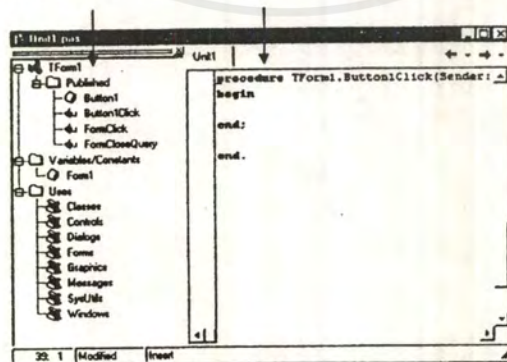
รูปที่ 5.9 แสดงพรอพเพอร์ตี้



รูปที่ 5.10 แสดงการใส่โค้ดลงในอีเวนต์เพจ

5.1.11 โค้ดอิดิเตอร์(Code Editor)

เป็นวินโดวที่ให้โปรแกรมเมอร์เขียนโค้ดลงไป โดยบางส่วนของเซลล์ไฟจะสร้างขึ้นมาให้ เพราะฉะนั้นก่อนลบโค้ดออกให้สังเกตด้วยว่าเป็นโค้ดที่เซลล์ไฟสร้างขึ้นมา หรือเป็นโค้ดที่เขียนเพิ่ม



รูปที่ 5.11 โค้ดอิดิเตอร์

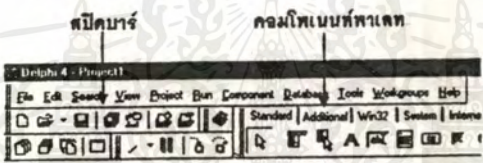
ลงไปซึ่งถ้าลบโดนที่เคลฟสร้างขึ้นมาให้เรา จะทำให้เกิดความผิดพลาดของโค้ดได้ หน้าที่ของเรา ก็คือเขียนโค้ดลงไปให้อีเวนท์แต่ละอีเวนท์เท่านั้น (สำหรับผู้ที่ยังไม่สามารถสร้างโปรซีเยอร์หรือ ฟังก์ชันขึ้นมาเอง)

ส่วนประกอบภายในโค้ดเอ็กโพเรเตอร์

- Published เป็นออบเจคและโปรซีเยอร์หรือฟังก์ชันที่ได้ประกาศเอาไว้
- Variable/Constants เป็นตัวแปรและค่าคงที่
- Uses เป็นไลบรารีที่ต้องใช้ชนิดดังกล่าว

5.1.12 คอมโพเนนท์พาเลท(Component Pallete)

เป็นกลุ่มของคอมโพเนนท์ซึ่งจะแบ่งตามฟังก์ชันในการใช้งาน



รูปที่ 5.12 คอมโพเนนท์พาเลท

สปีดบาร์ (Speed Bar)

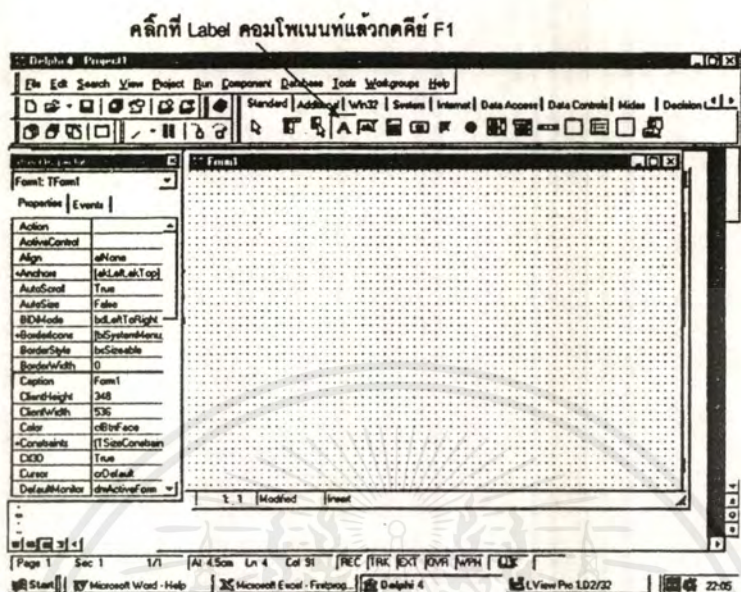
เป็นปุ่มของเมนูที่ต้องใช้เป็นประจำและต้องใช้บ่อยในการเขียนโปรแกรมปุ่มต่างๆจะมีชื่อ บอกเมื่อเลื่อนเมาส์ผ่านซึ่งชื่อตรงกับในเมนู

5.1.13 การใช้ Help

การใช้ทูลต่างๆของโปรแกรมในทุกวันนี้จะมี Online Help ให้มาด้วยเพราะฉะนั้นเมื่อเกิด ความสงสัยในคำหรือในทูลที่ให้มา ก็สามารถอ่านรายละเอียดการใช้งานจาก Help ได้ เคลฟก็เช่น เดียวกันสามารถใช้ Help ได้ตลอดเวลา โดยการคลิกที่เมนู Help หรือกดคีย์ F1 เพื่ออ่านรายละเอียด ของพรอพเพอร์ตี้และคอมโพเนนท์ต่างๆ

ซึ่งจะปรากฏวินโดว์แสดงรายละเอียดของ Label คอมโพเนนท์ดังรูป ซึ่งจะมีเมนูดังนี้

Hierarchy	ซึ่งจะแสดงการถ่ายทอดออบเจคของ TLabel
Properties	พรอพเพอร์ตี้ของ TLabel
Methods	เมธอดที่สามารถเกิดขึ้นได้กับออบเจค
Events	อีเวนท์ที่สามารถเกิดขึ้นได้กับออบเจค
See also	เนื้อหาใกล้เคียง
Example	ตัวอย่างการใช้งาน



รูปที่ 5.13 แสดงการใช้ Help

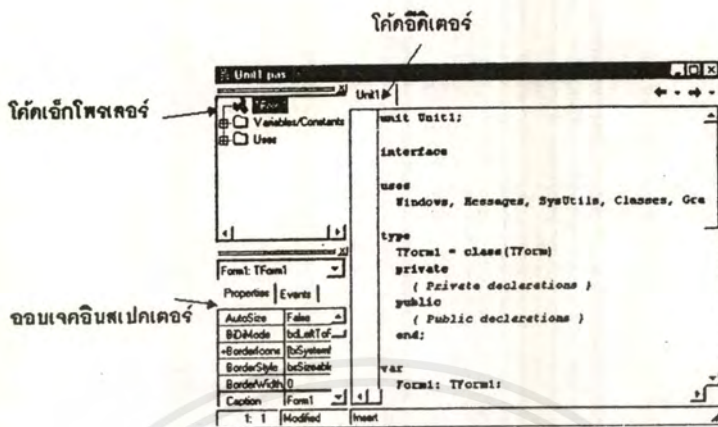
หรือต้องการทราบรายละเอียดของพอร์เตอร์ต์และอีเวนที่ปรากฏในออบเจกอินสเปคเตอร์ ก็ทำได้เช่นเดียวกัน โดยให้คลิกแล้วกดคีย์ F1



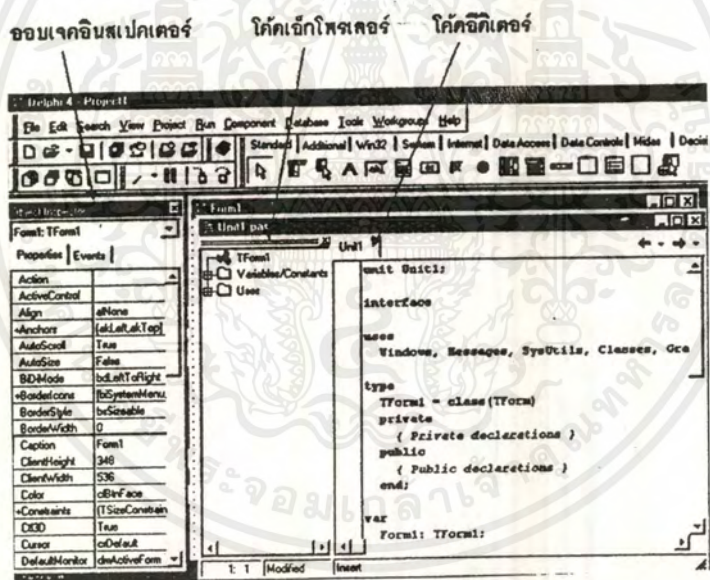
รูปที่ 5.14 แสดงการใช้ Help

5.1.14 การนำวินโดว์มาวางไว้ในกรอบเดียวกัน(Dockable)

ความสามารถอีกอย่างหนึ่งของเคลไฟ IDE คือการนำวินโดว์ซึ่งเป็นส่วนประกอบของเคลไฟมารวมกันให้อยู่ในกรอบเดียวกัน ซึ่งเรียกว่า Dockable สังเกตเคสที่ 1 ของเคลไฟจะพบว่าหน้าต่างของ Code Explorer อยู่ในกรอบเดียวกับ โค้ดอีดีเตอร์หรือทั้งสามวินโดว์จะให้อยู่ในกรอบเดียวกันก็ได้โดยการแดร็กและครีอ



รูปที่ 5.15 แสดงการนำวินโดว์มาวางไว้ในกรอบเดียวกัน



รูปที่ 5.15 (ต่อ) แสดงการนำวินโดว์มาวางไว้ในกรอบเดียวกัน

5.1.15 ดาตาเบสเอ็กโพลเลอร์ (Database Explorer)

เป็นตัวตรวจสอบฐานข้อมูลสร้างฐานข้อมูลและทดลองใช้ SQL สเตทเมนต์ สร้างเอเลียส- (AliasName)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



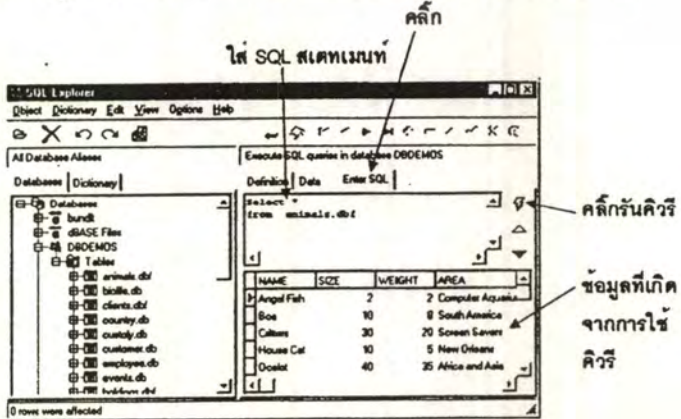
รูปที่ 5.16 แสดง Database Explorer

ภายในฐานข้อมูลนั้นจะประกอบไปด้วยเทเบิลหลายๆเทเบิล ให้คลิกที่เครื่องหมายบวกหน้าฐานข้อมูลนั้นๆจะปรากฏเทเบิลทุกๆเทเบิลที่อยู่ภายในฐานข้อมูลนั้น



รูปที่ 5.17 แสดงภายในฐานข้อมูล

สำหรับ Enter SQL เพจ เป็นส่วนที่ให้ทดลองใช้ SQL สเตทเม้นท์กับเทเบิลภายในฐานข้อมูลนั้น เพื่อสอบถามข้อมูล สำหรับในส่วนของ Data สามารถป้อนข้อมูลหรือลบข้อมูลได้

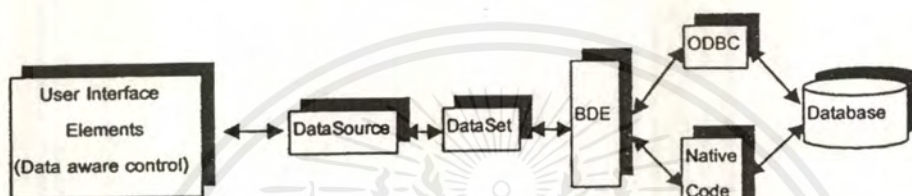


รูปที่ 5.18 แสดงภายใน SQL เพจ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.1.16 การสร้างเอเลียส(AliasName)

เอเลียสคือชื่อของฐานข้อมูลนั่นเองเช่น DBDEMOS, mssqlserver รูปไอคอนจะแตกต่างกันเล็กน้อยสำหรับเอเลียสที่ติดต่อด้วย native code และติดต่อด้าน ODBC ซึ่งการเลือกติดต่อกับฐานข้อมูลนั้นจะเลือกแบบไหนก็ได้ขึ้นอยู่กับความสะดวก ฐานข้อมูลที่ปรากฏใน ODBC ไดรเวอร์จะปรากฏในคาต้าเบสเอ็กโพเรเตอร์โดยอัตโนมัติซึ่งไอคอนจะมีรูปโซ่อยู่ข้างบน



รูปที่ 5.19 เอเลียส

Note Native Code เป็นการติดต่อโดยตรงไม่ต้องอาศัยไครฟ์เวอร์อื่นๆเข้ามาช่วย

คอนฟิกที่คองกำหนดก่อนการติดต่อด้านข้อมูล



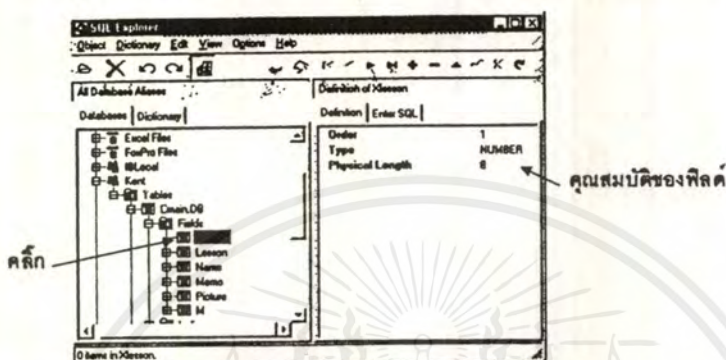
รูปที่ 5.20 แสดงส่วนที่เอเลียสติดต่อด

ขั้นตอนในการสร้างเอเลียส

1. กำหนดชื่อของเอเลียส
2. กำหนดชนิดของฐานข้อมูลนั้น
3. เซตพารามิเตอร์สำหรับการติดต่อ (เซต Path Directory กรณีที่เป็น โกลบอลคาตาเบส)

โดยกำหนดที่ definition เพจในคาตาเบสเอ็กโพเรเตอร์

ขณะนี้เราได้เอเลียสที่ชื่อ Kent และเก็บเทเบิลไว้ที่พาธ C:\database เราสามารถนำเอเลียสดังกล่าวไปใช้เป็น Database พรอพเพอร์ตี้ของคาค้าเซตได้ในคาค้าเบสเอ็กโพรเลอร์ก็สามารถปรับปรุงแก้ไขได้เลย การดูรายโครงสร้างของเทเบิลทำได้โดยการคลิกขวานไปเรื่อยๆตามรูป

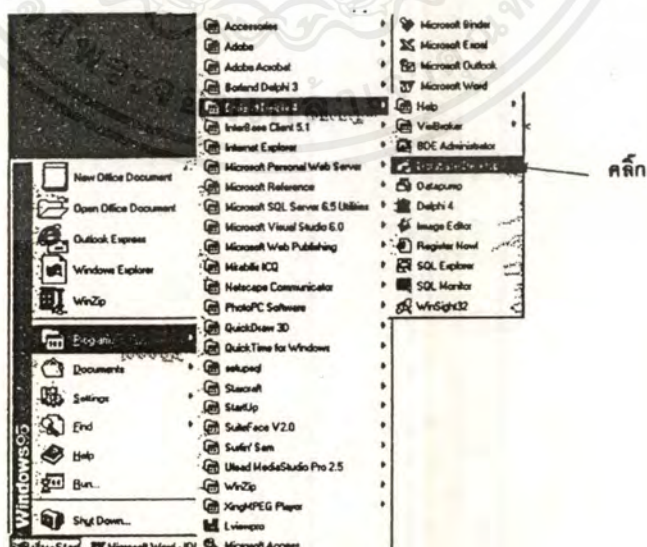


รูปที่ 5.22 แสดงการดูโครงสร้างของเทเบิล

5.1.17 คาค้าเบสเดสก์ท็อป(Database Desktop)

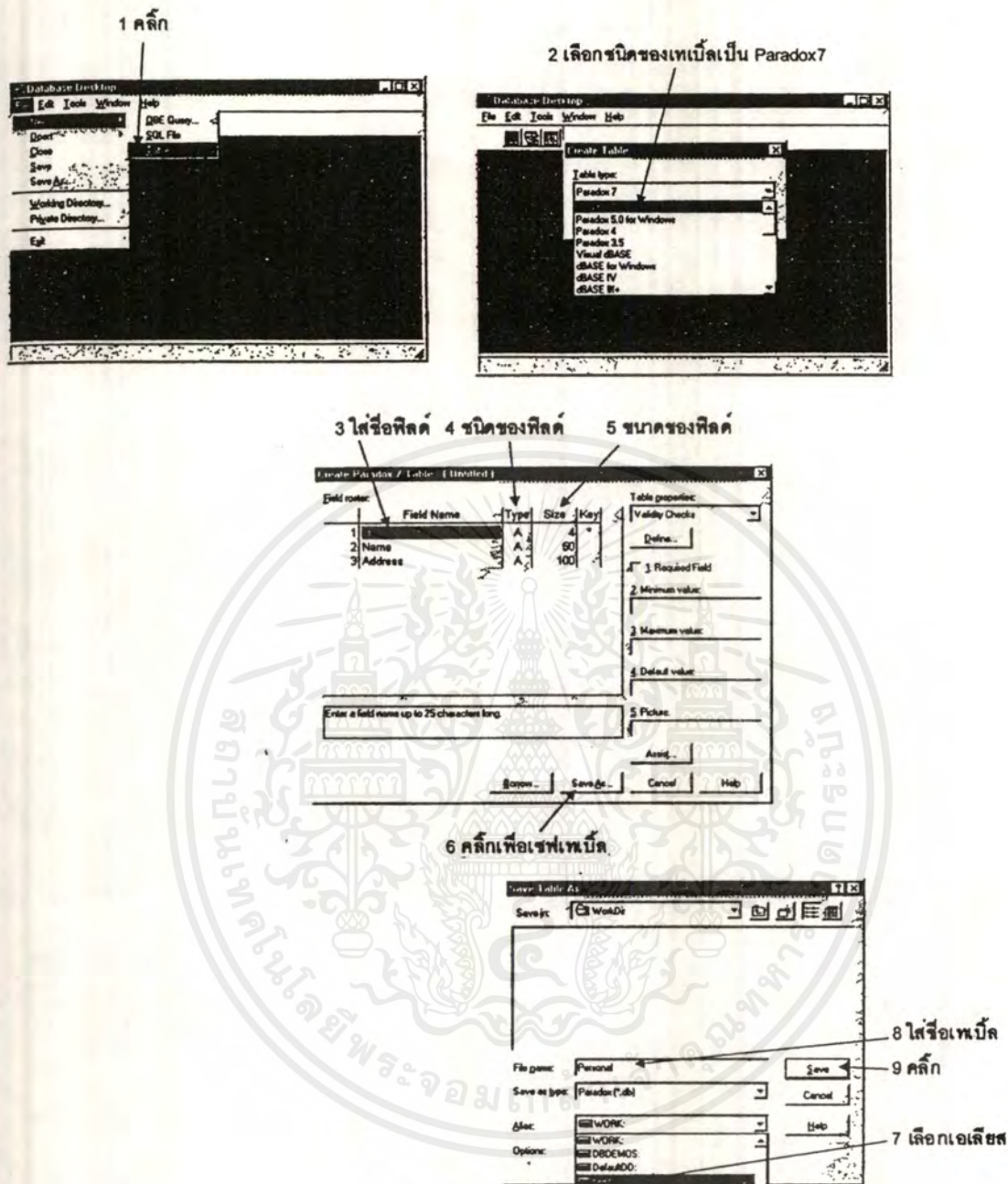
สำหรับผู้ที่ต้องการพัฒนาฐานข้อมูลภายในเครื่องนั้นจะต้องใช้คาค้าเบสเดสก์ท็อป เพื่อสร้างเทเบิลของฐานข้อมูลขึ้นมา เมื่อสร้างเสร็จแล้วเทเบิลดังกล่าวจะถูกเก็บเป็นไฟล์เอาไว้ที่เอเลียสที่ได้สร้างขึ้นมา(ต้องเก็บไว้ที่เอเลียสเท่านั้น)

ตัวอย่างการสร้างเทเบิลสำหรับโลคอลคาค้าเบส เมื่อทำตามขั้นตอนเสร็จเรียบร้อยแล้ว ก็จะมีเทเบิลชื่อ Personal เป็นเทเบิลชนิดพาราคอกซ์เก็บไว้ในเอเลียสKent ซึ่งสามารถนำไปสร้างแอปพลิเคชันขึ้นมาได้



รูปที่ 5.23 ตัวอย่างการสร้างเทเบิล

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

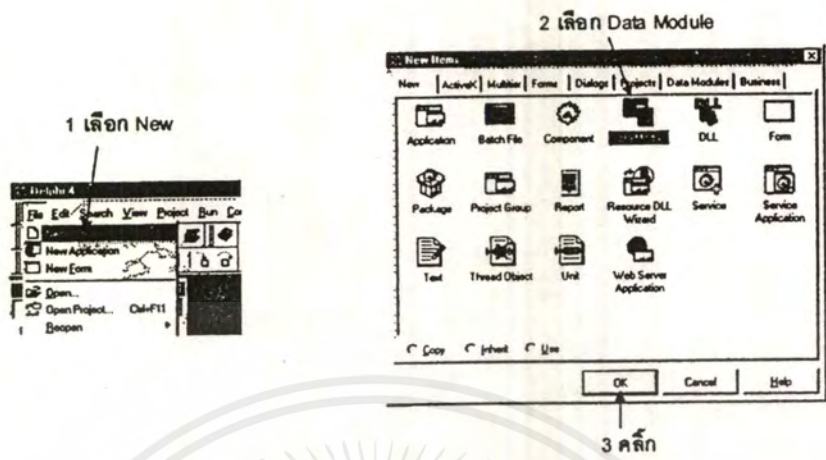


รูปที่ 5.23 (ต่อ) ตัวอย่างการสร้างเทเบิล

5.1.18 การใช้ DataModule

บางครั้งการเขียนโปรแกรมที่มีขนาดใหญ่ก็ทำให้เกิดความยุ่งยากได้ในกรณีพัฒนาแอปพลิเคชันคล้ายๆกัน แตกต่างในเรื่องของรายละเอียดเพียงเล็กน้อยเท่านั้น แดฟจึงสร้าง DataModule ขึ้นมาเพื่อช่วยลดปัญหาดังกล่าว โดยสร้างแอปพลิเคชันบางส่วนที่สามารถนำไปใช้กับแอปพลิเคชันอื่นเก็บไว้ก่อน โดยเก็บไว้ในรูปของไลบรารีซึ่งเป็นยูนิคไฟล์ และเมื่อต้องการนำกลับมาใช้ใหม่ก็เพิ่มยูนิคดังกล่าวลงใน Uses clause ซึ่งคอมไพเลอร์ที่ควรนำไปเก็บไว้ใน DataModule คือ คอมไพเลอร์ใน Data Access ซึ่งเป็นคอมไพเลอร์ที่ไม่สามารถมองเห็นได้ในขณะรันโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

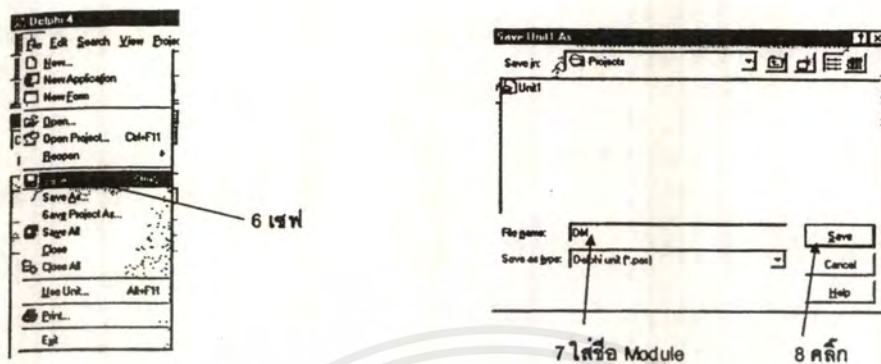


รูปที่ 5.24 การใช้ DataModule

5. เซตพารามิเตอร์ของตารางหรือและตาราง

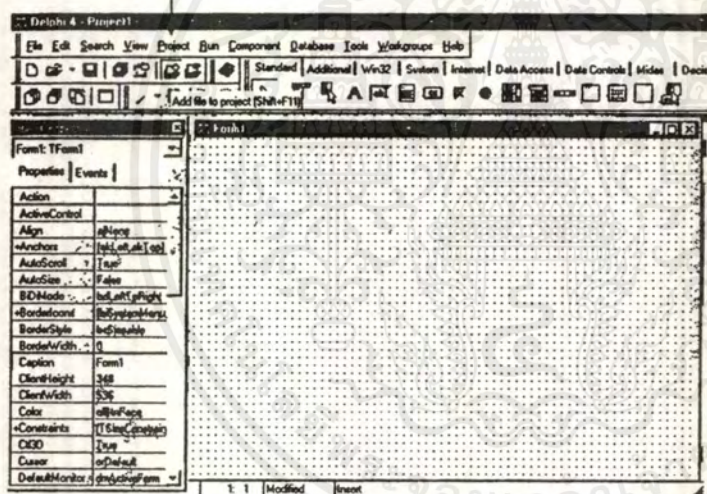
ขอบเขต	พารามิเตอร์	ค่า
Table 1	DatabaseName	DBDEMOS
Table 1	TableName	animals.dbf
Table 1	Active	True
Data Source 1	DataSet	Table1

DataModule ที่ชื่อDM เป็นไลบรารีอันหนึ่งที่สามารถนำกลับมาใช้ใหม่ได้ การนำ DataModule มาใช้ เลือกเมนู File/New Application เพื่อสร้างแอปพลิเคชันใหม่

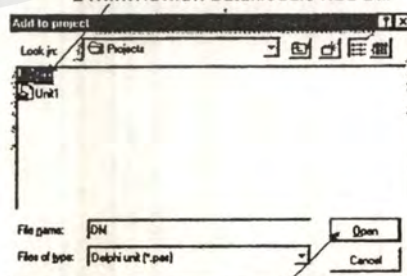


รูปที่ 5.25 การเซฟพรอพเพอร์ตี้ของดาตาซอร์ทและดาตาเซต

1. คลิกปุ่ม Addfile

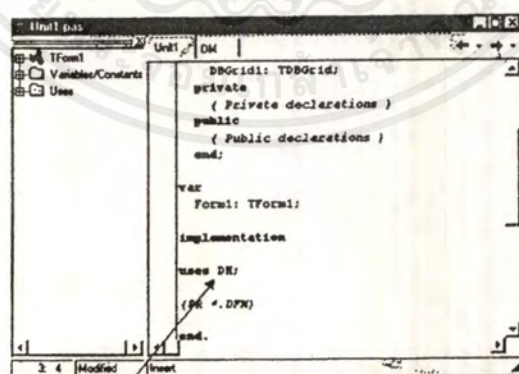
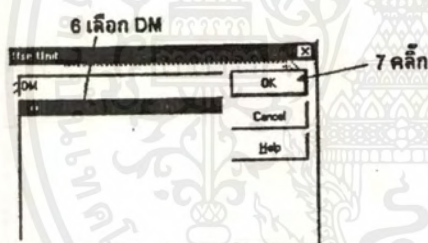
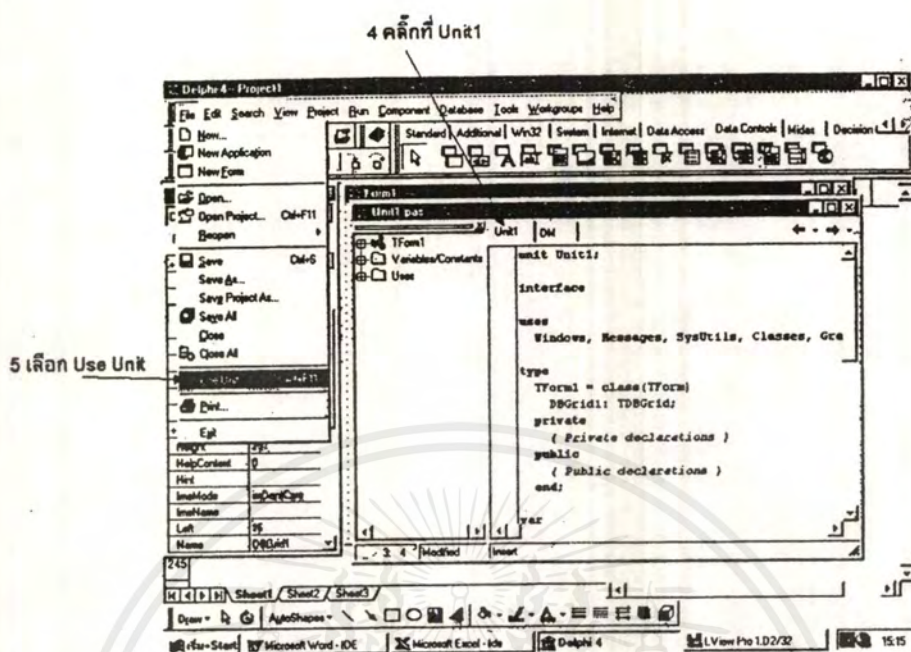


2. คลิกเพื่อเลือก DataModule ที่ชื่อ DM



รูปที่ 5.26 การนำ DataModule มาใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ปรากฏ DM ใน uses clause หลัง Implementation ให้อัตโนมัติ

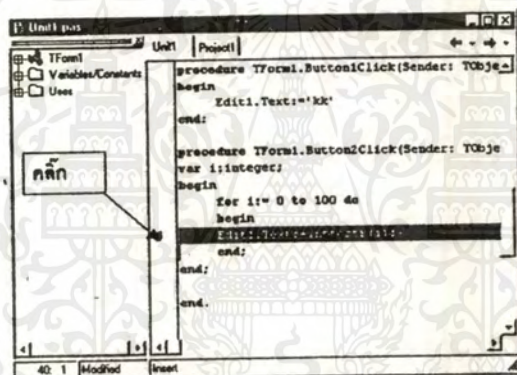
รูปที่ 5.26 (ต่อ) การนำ DataModule มาใช้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.1.19 การดีบั๊กโปรแกรม

ในขณะที่ออกแบบโปรแกรมเราสามารถตรวจสอบการทำงานของโปรแกรมหรือตรวจสอบค่าตัวแปรต่างๆได้ เพื่อลดความผิดพลาดเมื่อนำไปใช้งานจริง

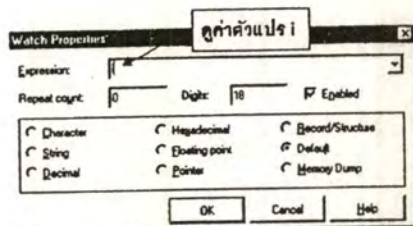
- Step Over หรือกดคีย์ F8 คือการรันโปรแกรมครั้งละ 1 คำสั่ง โดยจะข้ามรูทีนที่มีการวนลูปไป
- Trace into หรือกดคีย์ F7 คือการรันโปรแกรมครั้งละ 1 คำสั่ง โดยจะไม่ข้ามรูทีนที่มีการวนลูปไป
- Run to Cursor โปรแกรมจะรันไปจนกระทั่งถึงบรรทัดที่ได้มาร์คเอาไว้แล้วจะหยุดรัน(การมาร์คทำได้โดยการคลิกที่หน้าของบรรทัดนั้น)



รูปที่ 5.27 การดีบั๊กโปรแกรม

Add Watch

ตรวจสอบค่าภายในตัวแปร ซึ่งจะต้องกำหนดก่อนรันโปรแกรมว่าต้องการดูค่าในตัวแปรใด



รูปที่ 5.28 Add Watch

- Program Reset หยุดการทำงานของโปรแกรมเพื่อกลับไปอยู่ในสถานะของการออกแบบโปรแกรม
- Program Pause หยุดการทำงานของโปรแกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.2 ออบเจกต์ปาสคาล

การเขียนภาษาออบเจกต์ปาสคาลได้

คุณสามารถศึกษารายละเอียดของภาษาออบเจกต์ปาสคาลได้ใน Delphi's Online Help หรืออาจจะต้องซื้อหนังสืออ่านประกอบเช่น หนังสือ Turbo Pascal และ Borland Pascal ภาษาออบเจกต์ปาสคาลเป็นภาษาคอมพิวเตอร์ที่ยืดหยุ่นมาก ในเรื่องของการลดข้อผิดพลาดที่เกิดจากการเขียนโปรแกรม ซึ่งคอมไพเลอร์จะทำหน้าที่เสมือนเป็นผู้ช่วยในการเขียนโปรแกรมคอมพิวเตอร์ในตัว ยกตัวอย่างการเขียนโค้ดดังตัวอย่าง

PROCEDURE

TForm1.

```
Button1Click(Sender: TObject);begin
```

```
Edit1.Color := CLRED;END;
```

และดูที่โค้ดต่อไปนี้ด้วย

```
procedure TForm1.Button1Click(Sender: TObject);
```

```
begin
```

```
Edit1.Color := CLRED;
```

```
END;
```

การเขียนโค้ดทั้งสองรูปมีความหมายเหมือนกันทุกประการแต่อย่างที่สองจะทำให้คุณหรือผู้อื่นอ่านเข้าใจได้ง่ายกว่า

สำหรับการเขียนหมายเหตุลงในแต่ละสเตทเมนต์สำหรับการเขียน

การใช้เครื่องหมายที่แสดงถึงข้อความดังกล่าวเป็นหมายเหตุซึ่งให้คอมไพเลอร์ข้ามการแปลสตริงดังกล่าวออกไปเราจะใช้เครื่องหมายปีกกา {} และใช้เครื่องหมาย // ดับเบิ้ลสแลช เช่น {โพสิชันนี้สั่งให้เปลี่ยนสีของ ๆ Color ออบเจกต์ }

```
procedure TForm1.Button 1 Click(Sender:Tobject); {Change the color of the edit box
```

```
begin
```

```
Edit1. Color := clRed;
```

```
end;
```

หรือ

```
procedure TForm1.Button 1. Click (Sender:Tobject); // เปลี่ยนสีของอิดิตบ็อกซ์
```

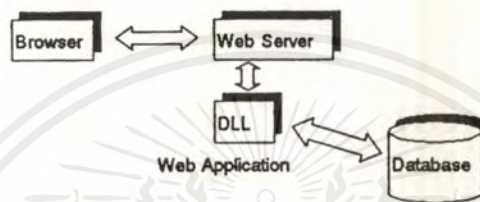
```
begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.3 เว็บแอปพลิเคชันและ Active X ฟอรัม

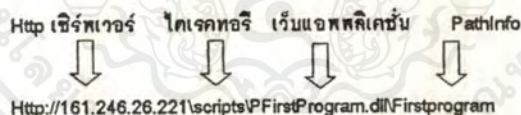
5.3.1 เว็บแอปพลิเคชัน

เคลฟสามารถสร้างเว็บแอปพลิเคชันเพื่อให้เว็บเซิร์ฟเวอร์ติดต่อกับฐานข้อมูลได้โดยการสร้าง DLL ขึ้นมาให้โปรแกรมเว็บเซิร์ฟเวอร์เรียกใช้โดยเฉพาะอย่างยิ่งกับ Internet Information Server (IIS) ซึ่งเป็นเว็บเซิร์ฟเวอร์ที่รันบนเครื่องวินโดวส์เอ็นทีเซิร์ฟเวอร์



รูปที่ 5.29 Web Application

คอมโพเนนต์ในกลุ่ม Internet คอมโพเนนต์จะช่วยให้การแปลงฐานข้อมูลให้เป็นไฟล์สำหรับเว็บเซิร์ฟเวอร์ เพื่อส่งให้กับเบราว์เซอร์ตามการร้องขอเว็บแอปพลิเคชันที่สร้างขึ้นมาจะแบ่งการตอบสนองต่อการร้องขอจากเบราว์เซอร์เป็นกลุ่มที่เรียกว่า Action โดยจะเรียกแต่ละ Action ว่า ActionItem โดย ActionItem แต่ละตัวจะมี PathInfo เป็น /Search เมื่อเรียกจากตัวเบราว์เซอร์ก็ต้องใส่พารามิเตอร์ Search ตามหลัง URL นั้นด้วย



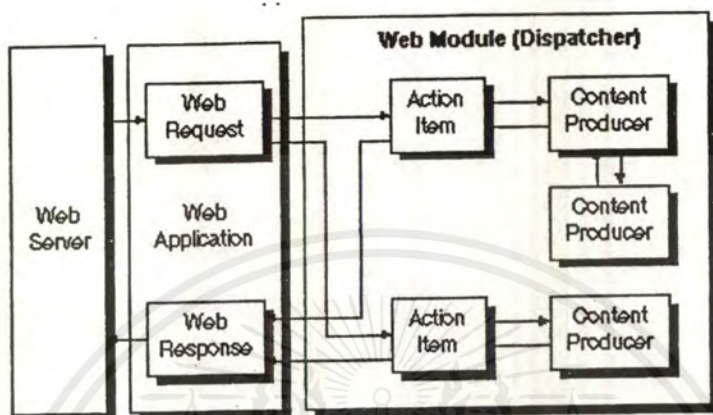
รูปที่ 5.30 ไดเรกทอรี Scripts

ภายใน URL จะต้องบอก http เซิร์ฟเวอร์ที่เก็บแอปพลิเคชันตามด้วยไฟล์ DLL ของเว็บแอปพลิเคชันและท้ายสุดคือ PathInfo ที่ได้กำหนดไว้ในแต่ละ ActionItem การสร้างเว็บแอปพลิเคชันจะพัฒนาบนเอ็นทีเซิร์ฟเวอร์โดยสร้างโปรเจกต์ไฟล์ไว้ที่ ไดเรกทอรี Scripts ซึ่งเป็น ไดเรกทอรีที่กำหนดให้เป็น public สามารถเรียกจากเบราว์เซอร์เข้ามาได้

5.3.2 การทำงานของ ActionItem

เมื่อเว็บแอปพลิเคชันได้รับ HTTP เมสเสจแล้วมันจะสร้าง TWebRequest ออบเจกต์เพื่อตอบสนองต่อการร้องขอดังกล่าว ซึ่ง TWebRequest จะร้องขอไปยัง ActionItem ซึ่งอยู่ภายใน WebModule โดยจะแบ่งการตอบสนองต่อการร้องขอเป็นกลุ่มๆที่เรียกว่า ActionItem เพราะฉะนั้นการสร้างเว็บแอปพลิเคชันจะต้องกำหนด ActionItem ขึ้นมาและเขียนโค้ดลงไป ใน OnActionItem

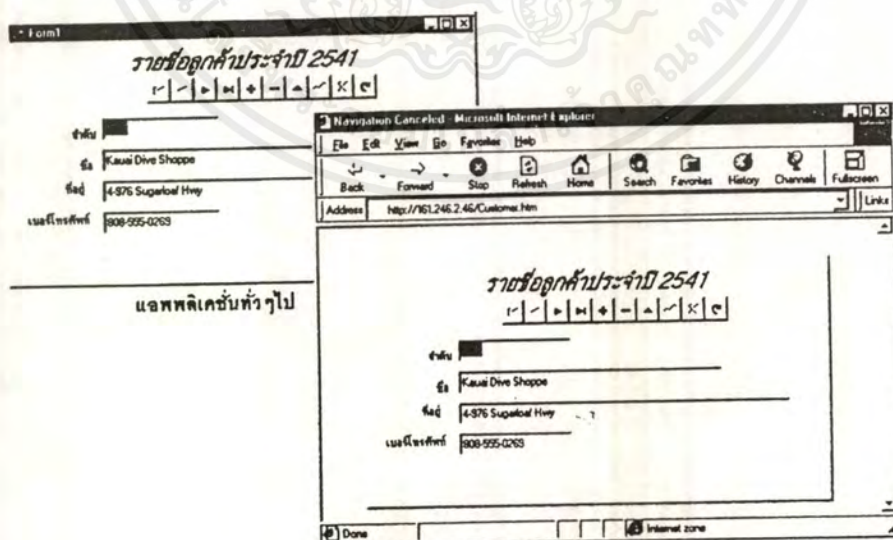
นั้นภายในเว็บโมดูล เมื่อ ActionItem ได้รับแมสเสจแล้วก็จะสร้างโค้ดที่เป็น HTML ไฟล์ส่งไปให้กับเว็บเซิร์ฟเวอร์



รูปที่ 5.31 การทำงานของ ActionItem

5.3.3 การสร้าง ActiveX ฟอรัม

ActiveX ฟอรัมคือฟอรัมที่ทำหน้าที่เป็นแอปพลิเคชันที่ไม่สามารถรันด้วยตัวของมันเองได้แต่จะรันจากตัวบราวน์เซอร์ที่สนับสนุน ActiveX เทคโนโลยีเท่านั้นเช่น Internet Explorer โดยฟอรัมดังกล่าวจะเก็บไว้ที่เว็บเซิร์ฟเวอร์เมื่อตัวบราวเซอร์จากไคลเอ็นท์ร้องขอ เว็บเซิร์ฟเวอร์ก็จะทำหน้าที่ส่งฟอรัมไปให้กับ ตัวบราวเซอร์จากนั้นตัวบราวเซอร์ จะติดตั้งฟอรัมที่ได้ดาวน์โหลดมาลงไปแล้วก็จะรันขึ้นมา

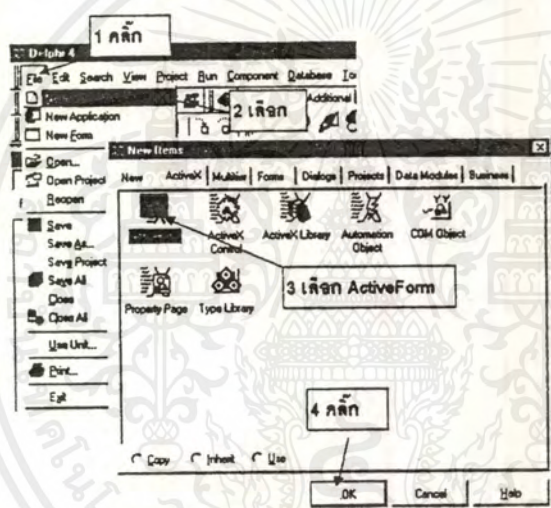


รูปที่ 5.32 ActiveX

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การสร้าง ActiveX ฟอรัมก็คือการสร้าง HTML ไฟล์ขึ้นมาจากนั้นก็เก็บไว้ที่ในไคลเอนต์ที่
ที่สามารถเรียกใช้จากบราวเซอร์ได้เช่น C:\Webshare\wwwroot(กรณีที่ใช้เว็บเซิร์ฟเวอร์ดังกล่าวเป็น
Personal Web Server) ตัวอย่างต่อไปนี้จะสร้าง ActiveX ฟอรัมเพื่อให้เห็นเวลาขึ้นมาโดยจะตั้ง
ชื่อโปรเจกต์ไฟล์เป็น Timer และยูนิตไฟล์เป็น Timer และยูนิตไฟล์เป็น uTimer และเก็บไว้ที่ไคลเ-
นตร์ C:\Webshare\wwwroot เมื่อรีจิสเตอร์เซิร์ฟเวอร์เรียบร้อยแล้ว แล้วรันโปรแกรม เผลไปจะแปลง
ActiveX ดังกล่าวให้สามารถเรียกใช้จากบราวเซอร์ได้ดังนี้ Http:\\161.246.2.46\Timer.htm ซึ่งชื่อ
โปรเจกต์ไฟล์จะเป็นชื่อเดียวกับ Html File

1. เริ่มสร้าง โปรแกรมโดยเลือกเมนู File|Close All เพื่อปิดแอปพลิเคชันที่ Default อยู่ลงไปก่อน



รูปที่ 5.33 การสร้าง ActiveX (ขั้นตอนที่ 1)

2. จากนั้นเลือก Label และ Timer แล้วคอมไพล์บนฟอรัม พร้อมทั้งเซตฟอนต์พรอพเพอร์ตี้ของ label ให้มีขนาดใหญ่ดังรูป



รูปที่ 5.34 การสร้าง ActiveX (ขั้นตอนที่ 2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3. จากนั้นใส่โค้ดลงในอีเวนต์ On Timer ดังนี้

Procedure TactiveFromX.Timer1 Timer(Sender:TObject);

Begin

If Label1.Visible then

Label1.Visible := false

Else

Begin

Label1.Visible:= true;

Label1.Caption:=Datetime to str(now);

End;

End;

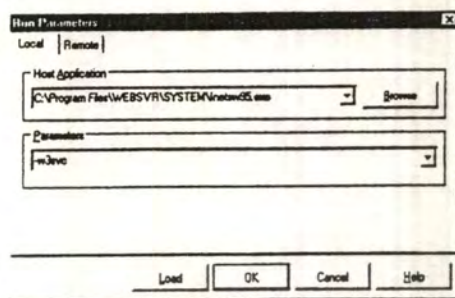
4. จากนั้น เซฟ ยูนิคไฟล์ และ โปรเจคไฟล์ใน ไดเรกทอรี C:\Webshade\wwwroot\ ในชื่อ uTimer.pas และ Timer.Dpr ตามลำดับ

5. จากนั้นเลือกเมนู Project | Web Deployment Options กำหนด Target Dir, Target URL , HTML dir ดังรูป แล้วคลิก OK



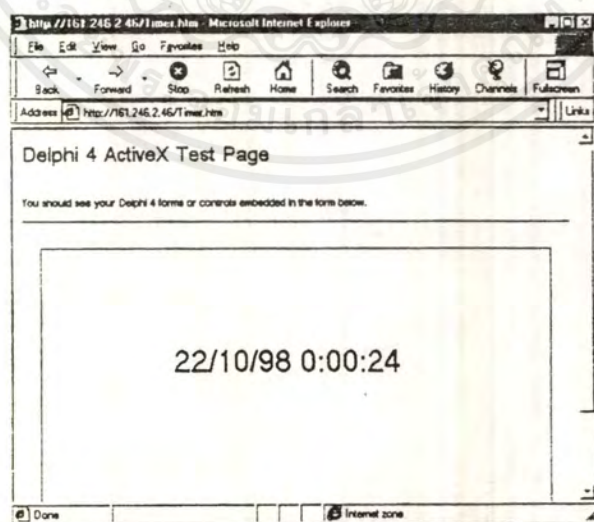
รูปที่ 5.35 การสร้าง ActiveX (ขั้นตอนที่ 5)

6. เลือกเมนู Project | Web deploy



รูปที่ 5.36 การสร้าง ActiveX (ขั้นตอนที่ 6)

7. เลือกเมนู Run | Parameter ใส่ Text ลงไปตามรูป (กรณีที่ใช้ Personal Web Server) Host Application: C:\Program Files\ websvr\system\inetw95.exe
Run Parameters: -w3svc
ซึ่งได้ติดตั้งแล้วจะรัน Windows95 Personal web Server และ ช่วยคีย์ DLL ด้วยสำหรับ IIS
Host Application: C:\Winnt\system32\inetrv\inetinfo.exe
Run Parameters: -ew3svc
8. เลือกเมนู Run | Register ActiveX Server
9. เลือกเมนู Run | Run
10. ใช้บราวเซอร์เรียก URL [Http://161.246.246/Timer.htm](http://161.246.246/Timer.htm)



รูปที่ 5.37 การสร้าง ActiveX (ขั้นตอนที่ 10)

การสร้างแอปพลิเคชันอื่นๆ ก็เช่นเดียวกัน ทำได้คล้ายกับการสร้างแอปพลิเคชันทั่วไป

เพียงแค่เปลี่ยนจากฟอร์มทั่วไป เป็น ActiveX ฟอร์ม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้拿去ใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 6

การทำงานของระบบ

6.3 รายละเอียดการทำงานของระบบ

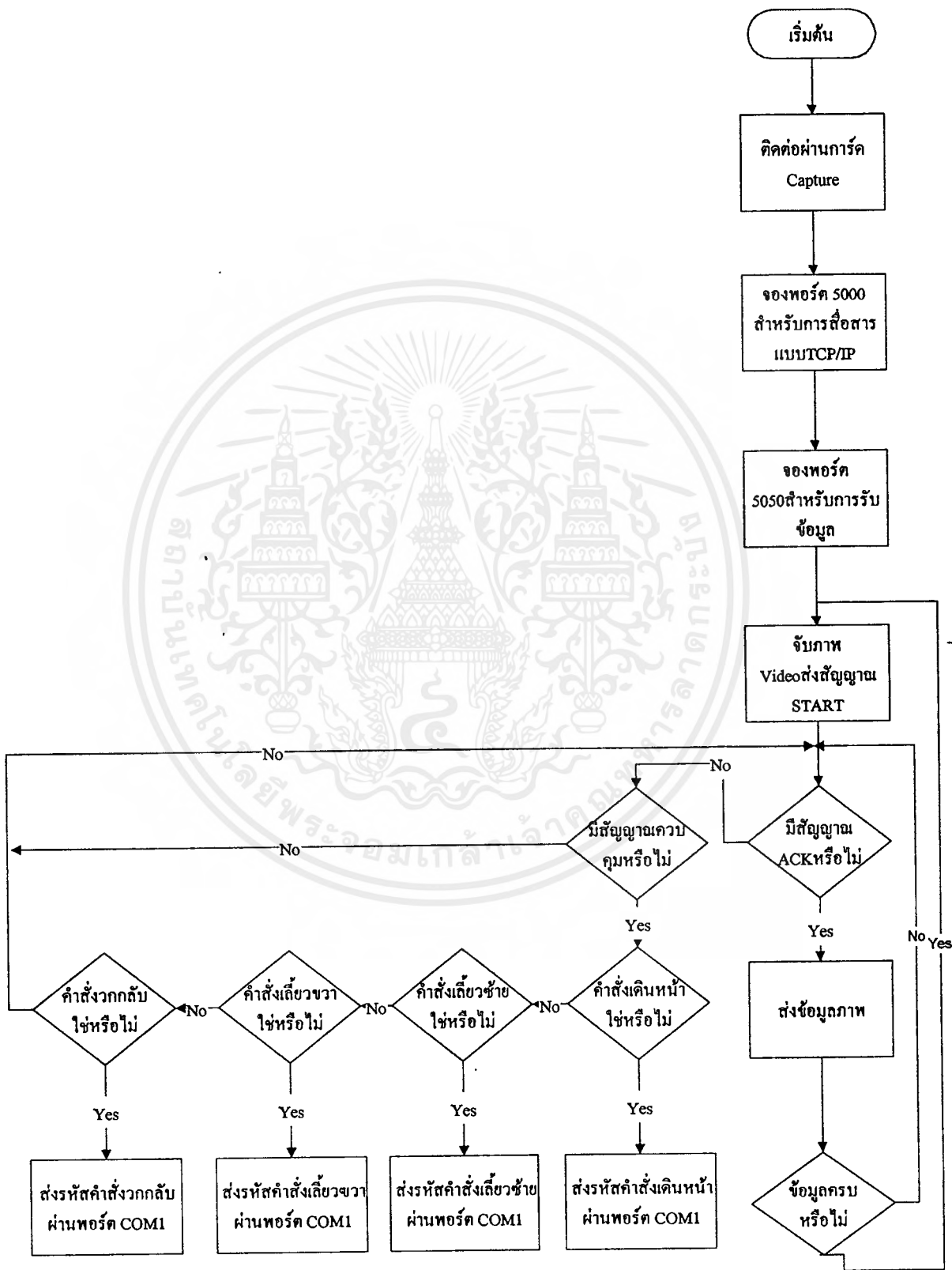
การทำงานของระบบประกอบด้วย

- การควบคุมตัวรถ
- ส่วนการติดต่อกับผู้ใช้
- ส่วนของตัวกลางการประมวลผล (Server)

ส่วนของการควบคุมตัวรถ จะทำหน้าที่ควบคุมการทำงานของรถสำรวจโดยรับคำสั่งผ่าน serial port ซึ่งคำสั่งที่ใช้จะเป็นคำสั่ง 8 bit (1 byte) ซึ่งแบ่งเป็นการควบคุมรถ 4 ทิศทาง ได้แก่

■ เดินหน้า	ใช้รหัสควบคุมเป็น	08 H
■ เลี้ยวซ้าย	ใช้รหัสควบคุมเป็น	04 H
■ เลี้ยวขวา	ใช้รหัสควบคุมเป็น	06 H
■ วกกลับ	ใช้รหัสควบคุมเป็น	05 H

อัตราเร็วในการรับส่งข้อมูลที่ใช้จะถูกกำหนดไว้เป็น 9600 bit ต่อ วินาที ซึ่งทั้งด้านส่วนกลาง-การประมวลผล กับด้านส่วนการควบคุมตัวรถสำรวจ จะต้องถูกกำหนดให้มีค่าให้การรับส่งด้วยความเร็วเดียวกัน ดังนั้นโปรแกรมในส่วนนี้จะเริ่มต้นด้วยการกำหนดค่าอัตราเร็วในการรับส่งข้อมูลเป็นค่าดังกล่าว จากนั้นก็จะทำการตรวจสอบการรับข้อมูลของพอร์ตอนุกรมว่ามีข้อมูลเข้ามาหรือไม่ เมื่อได้รับข้อมูลเข้าระบบจะทำการตรวจสอบข้อมูลนั้นว่าตรงตามรหัสที่กำหนดไว้หรือไม่ ถ้าตรงตามรหัสใดก็จะทำตามคำสั่งของรหัสนั้น เช่น ถ้าข้อมูลที่ได้รับเข้ามาเป็น 06 H ซึ่งจะตรงกับรหัสของคำสั่งเลี้ยวขวา จากนั้นระบบจะทำการควบคุมให้รถเลี้ยวขวา ตัวรถสำรวจจะสามารถเคลื่อนที่ได้ตามที่กำหนดเนื่องจากรถได้รับรหัสที่ได้ทำการกำหนดสัญญาณต่างๆของ IC-LMD18201



6.2 ส่วนตัวกลางการประมวลผล

ส่วนตัวกลางการประมวลผล เป็นส่วนที่ใช้ในการติดต่อระหว่างส่วนการติดต่อกับผู้ใช้ และส่วนการควบคุมตัวรถ โดยที่ระบบจะเริ่มต้นด้วยการติดต่อกับการ์ดแคปเจอร์ ซึ่งใช้จับสัญญาณภาพจากกล้องแล้วนำมาแสดงผลบนหน้าจอของเครื่องที่เป็นตัวให้บริการจากนั้นก็ทำการจองช่องการสื่อสาร (Communication Port) สำหรับโปรโตคอล TCP / IP โดยในที่นี้พอร์ตสำหรับการส่งข้อมูลภาพที่ได้จากกล้อง 1 พอร์ต (Port 5000) และพอร์ตสำหรับการรับรหัสในการควบคุมรถจากทางส่วนการติดต่อกับผู้ใช้ อีก 1 พอร์ต จากนั้นส่วนตัวกลางการประมวลผลก็จะเข้าสู่สภาพการรอ (Listen) เมื่อมีการติดต่อจากส่วนการติดต่อผู้ใช้เข้ามาในระบบจะทำการต้อนรับ และส่งข้อมูลภาพที่ได้จากกล้องไปยังส่วนการติดต่อผู้ใช้ เมื่อแสดงภาพพร้อมกันนั้นก็พร้อมที่จะรอสัญญาณควบคุมจากส่วนการติดต่อผู้ใช้ เมื่อส่วนการติดต่อผู้ใช้ ส่งรหัสคำสั่งสำหรับการควบคุมมา ส่วนตัวกลางการประมวลผล ก็จะตีความเป็นรหัสเพื่อใช้ในการควบคุมการเคลื่อนที่ของรถ แล้วส่งผ่าน Port อนุกรมไปยังส่วนการควบคุมรถนำไปตีความต่อไป

สัญญาณที่ใช้ในการส่งภาพของส่วนกลางการประมวลผลจะเริ่มด้วยส่วนตัวกลางการประมวลผลทำการจับภาพที่ได้จากกล้องครั้งละ 1 เฟรม แล้วทำการส่งผ่านทางพอร์ตของ TCP / IP โดยจะแบ่งไฟล์เป็นส่วนย่อยในที่นี้จะแบ่งเป็นส่วนละ 8Kb เมื่อทำการส่งไปยังส่วนการติดต่อผู้ใช้ และรอจนกว่าจะได้รับสัญญาณตอบรับจากส่วนการติดต่อผู้ใช้ แล้วจึงจะทำการส่งภาพส่วนต่อไป และเป็นเช่นนี้ต่อไปจนกว่าจะครบ

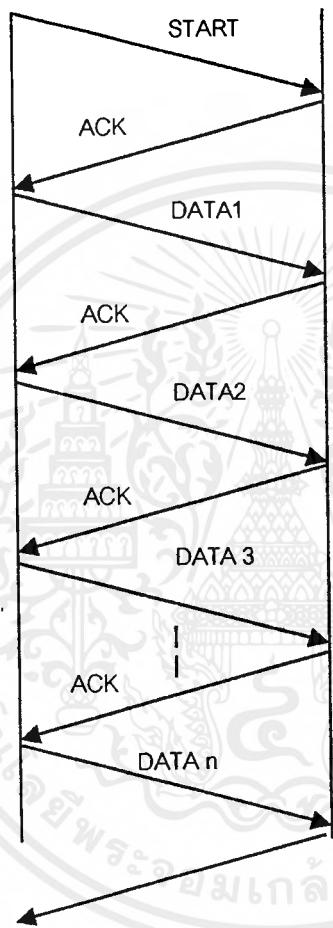
ภาพแสดงเป็น Timing Diagram ได้ดังนี้

ส่วนกลางการประมวลผล

ส่วนการติดต่อผู้ใช้

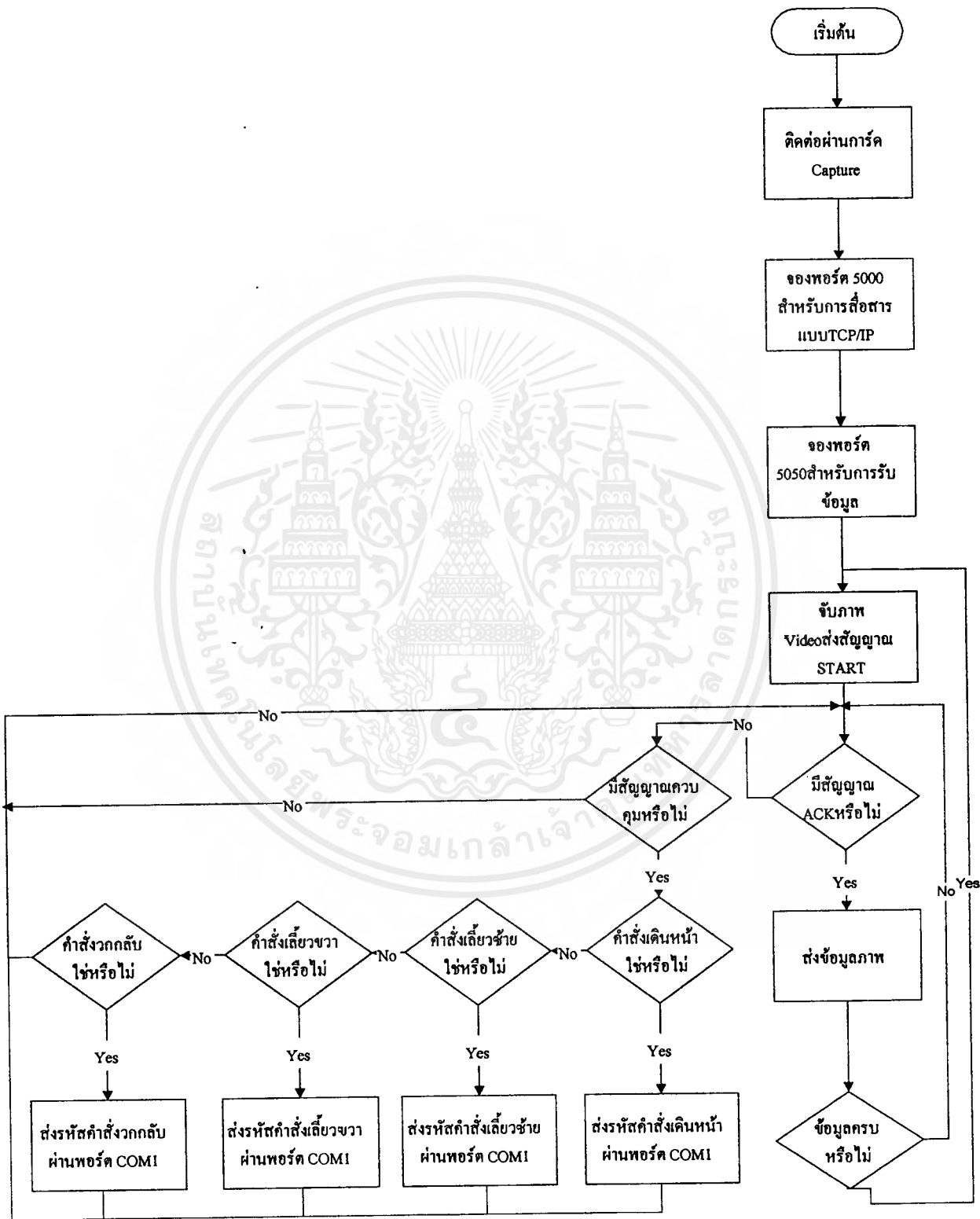
(Server)

(Client)



รูปที่ 6.1 แสดง Timing Diagram

จากไดอะแกรมจะเห็นว่าส่วนตัวกลางการประมวลผลจะทำการส่งสัญญาณ Start ไปยัง ส่วนการติดต่อผู้ใช้เพื่อเริ่มต้นการส่งข้อมูลเมื่อส่วนตัวประมวลผลได้รับสัญญาณ ACK ซึ่งเป็น สัญญาณตอบรับจากส่วนของการติดต่อผู้ใช้ก็จะทำการส่งข้อมูลชุดแรก และรอสัญญาณตอบรับ เพื่อนำการส่งข้อมูลชุดถัดไป ระบบจะสิ้นสุดการส่งข้อมูลเมื่อข้อมูลชุดล่าสุดที่ส่งไปเป็นข้อมูล ชุดสุดท้ายซึ่งจะทราบได้โดยการดูขนาดของข้อมูลจะมีขนาดน้อยกว่า 8 Kb และถือเป็นการสิ้นสุด การทำงาน



6.3 ส่วนการติดต่อผู้ใช้

ส่วนการติดต่อผู้ใช้จะทำงานบนระบบอินเทอร์เน็ตผ่านทางเว็บเพจใช้โปรโตคอล HTTP และติดต่อกับส่วนกลางการประมวลผลโดยใช้โปรโตคอล TCP/IP เพราะฉะนั้นโปรแกรมที่สร้างขึ้นสำหรับส่วนนี้สามารถทำได้ 3 วิธีคือ

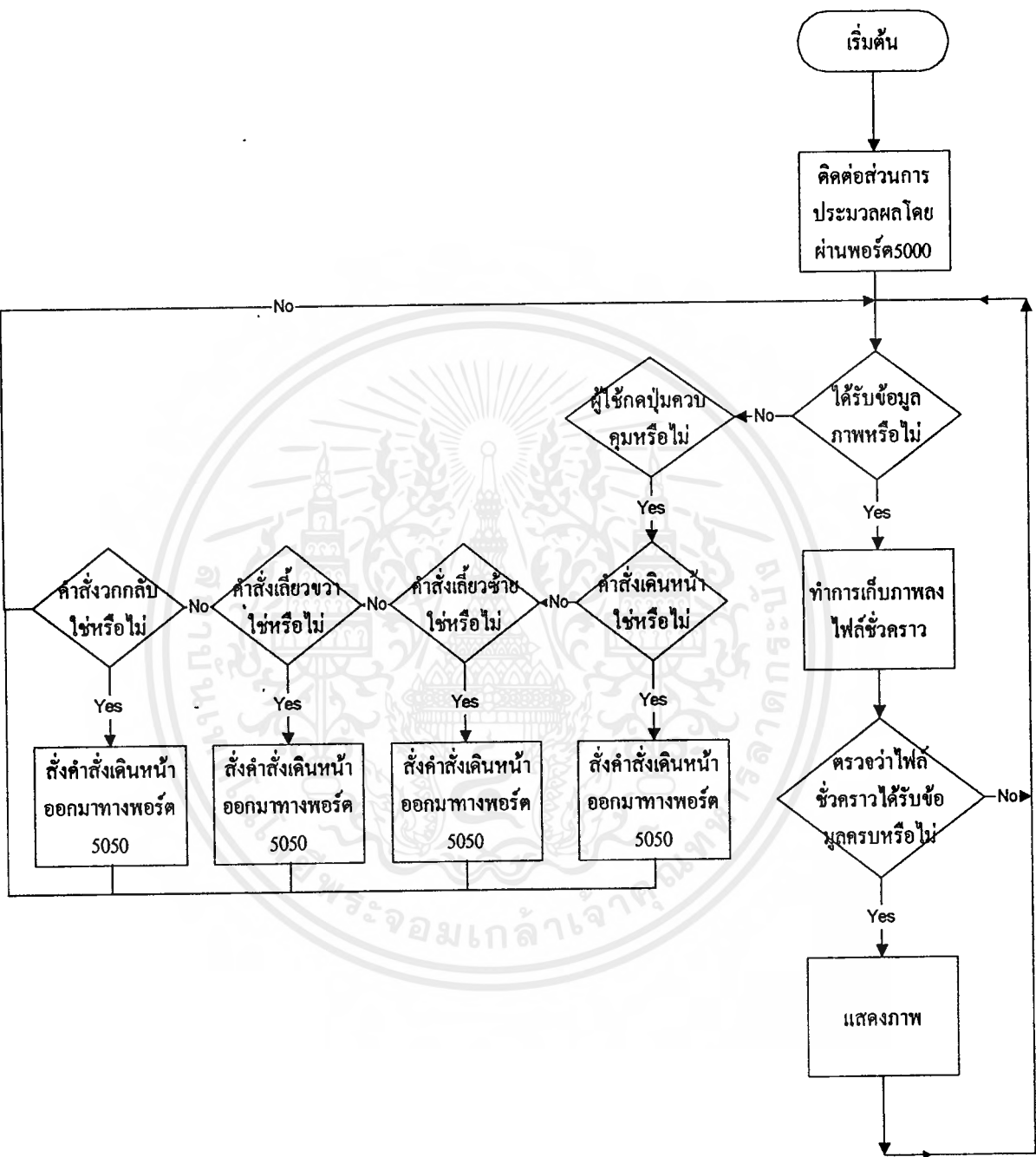
1. CGI
2. Java Language
3. ActiveX

สำหรับ CGI ซึ่งโปรแกรมที่ทำงานบนส่วนกลางการประมวลผลโดยใช้การติดต่อผ่านทางเว็บเพจดังนั้นจะสามารถอ่านข้อมูลจากตัวกล้องและควบคุมการทำงานของตัวรถได้แต่การทำงานของ CGI มีจำกัดเพราะ CGI จะทำงานเฉพาะเมื่อได้รับการติดต่อจากเว็บเพจที่อ้างอิงถึงตัว CGI นั้น และส่งผลที่ได้จากการรันโปรแกรมกลับคืนผ่านทางเว็บเพจครั้งละ 1 เพจ ในกรณีนี้จะทำให้เราสามารถส่งคำสั่งเพื่อควบคุมการทำงานของรถได้แต่ในการรับค่าที่ได้จากตัวกล้องจะไม่ต่อเนื่องเนื่องจากต้องรอการติดต่อกับเว็บเพจครั้งต่อไป

การแก้ปัญหานี้ใช้การส่งข้อมูลแบบ Streaming Video แต่ว่าอาจจะต้องมีโปรแกรมในส่วนนี้ สำหรับ Java Language เป็นภาษาที่ใช้เขียนโปรแกรมบนเว็บที่มีโครงสร้างของภาษาคัดลอกกับภาษา C++ ซึ่งเราสามารถเขียนโปรแกรมส่วนประมวลผลกลางได้โดยตรง

สำหรับ ActiveX เป็นการสร้างโปรแกรมเพื่อให้สามารถรันบนเว็บเพจได้ ทำให้สามารถเขียนโปรแกรมติดต่อกับส่วนกลางการประมวลผลได้เช่นกัน

สรุป ในที่นี้ได้เลือกใช้ (ActiveX) เพราะว่าเนื่องจาก ActiveX สามารถสร้างได้จาก PROGRAMMING TOOL ได้หลายชนิดทำให้เราสามารถเลือก Tool ที่จะใช้เองได้ การทำงานส่วนนี้จะเริ่มต้นด้วยการส่งสัญญาณผ่านทางโปรโตคอล TCP/IP เพื่อติดต่อการส่งกับส่วนกลางการประมวลผลที่พอร์ต 5000 สำหรับรับข้อมูลภาพ และที่พอร์ต 5050 สำหรับการส่งคำสั่งควบคุมรถสำรวจจากผู้ใช้ไปยังส่วนกลางการประมวลผลเมื่อส่วนกลางการประมวลผลส่งภาพมายังส่วนการติดต่อผู้ใช้เมื่อส่วนการติดต่อผู้ใช้ได้รับข้อมูลก็จะส่งสัญญาณ ACK เป็นการตอบรับให้ส่วนกลางการประมวลผลส่งข้อมูลชุดต่อไปเมื่อได้รับข้อมูลครบในแต่ละเฟรมแล้วก็จะแสดงผลบางหน้าจอและในขณะที่เดียวกันถ้าผู้ใช้กดปุ่มเพื่อควบคุมรถระบบก็จะส่งคำสั่งในการควบคุมนั้นไปที่ส่วนกลางการประมวลผลแล้วส่วนกลางการประมวลผลจะนำไปทำงานต่อไป



ภาคผนวก ก.

จะประกอบด้วย

1. โปรแกรมควบคุมการเคลื่อนที่ของหุ่นยนต์
2. โปรแกรม WebPage โดยใช้ HTML
3. โปรแกรม Delphi
4. รูปวงจรอิเล็กทรอนิกส์ และ รายละเอียดของอุปกรณ์ (Data Sheets)



1. ส่วนโปรแกรมควบคุมการเคลื่อนที่ของหุ่นยนต์



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

ORG      0000H

bootup:  djnz   r0, $
         djnz   r1, bootup

INIT:    MOV     SCON,#52H
         MOV     PCON,#00H
         SETB    REN
         MOV     TMOD,#20H
         MOV     TH1,#0FDH
         SETB    TR1
         MOV     P2, #00h

```

RECEIVE: MOV A,SBUF

MOV P2,A

CTRL: SETB BRAKE1

SETB BRAKE2

SETB DIR1

SETB DIR2

SETB PWM1

SETB PWM2

KEY8: CJNE A,#08H,KEY4

CLR BRAKE1

CLR BRAKE2

CALL DELAY1

SETB BRAKE1

SETB BRAKE2

JMP CHECKINPUT

KEY4: CJNE A,#04H,KEY6

CLR BRAKE2

CALL DELAY2

SETB BRAKE2

JMP CHECKINPUT

KEY6: CJNE A,#06H,KEY5

CLR BRAKE1

CALL DELAY2

SETB BRAKE1

JMP CHECKINPUT

KEY5: CJNE A,#05H,CHECKINPUT

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
CLR DIR2
CLR BRAKE1
CLR BRAKE2
CALL DELAY3
SETB BRAKE1
SETB BRAKE2
JMP CHECKINPUT
```

```
DELAY_1: MOV R2,#00H
DL1: MOV R3,DELAY1
DJNZ R3,$
DJNZ R2,DL1
RET
```

```
DELAY_2: MOV R2,#00H
DL2: MOV R3,DELAY2
DJNZ R3,$
DJNZ R2,DL2
RET
```

```
DELAY_3: MOV R2,#00H
DL3: MOV R3,DELAY3
DJNZ R3,$
DJNZ R2,DL3
RET
```

```
END
```

2.ส่วนโปรแกรม WebPage



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<HTML>
<HEAD>
<TITLE> MY PROJECT </TITLE>
<META NAME =keywords  CONTENT = "robot,control">
</HEAD>
<FRAMESET Rows ="17%,*">
  <FRAMESET Cols ="9%,82%,*">
    <FRAME SRC ="c:\jesada\shows1.html"Name="Shows" Frameborder ="No">
    <FRAME SRC ="c:\jesada\shows2.html" Name="Shows1" Frameborder ="No" >
    <FRAME SRC ="c:\jesada\shows1.html"Name="Shows" Frameborder ="No">
  </FRAMESET>

  <FRAMESET Cols ="25%,*">
    <FRAME SRC ="c:\jesada\Content.html" Name="Contents">
    <FRAME SRC ="c:\jesada\J1.html" Name="Main">
  </FRAMESET>
</FRAMESET>
</HTML>
```



```
<HTML>
<HEAD>
<TITLE>Shows</TITLE>
</HEAD>
<BODY BgColor =blue>
<IMG SRC = C:\JESADA\COPYRIGHT4.GIF>
</BODY>
</HTML>
```



```
<HTML>
<HEAD>
<TITLE> Shows Frame </TITLE>
</HEAD>
<BODY BgColor = Aqua>
<FONT Color = FUCHSIA>
<blink><MARQUEE Width = 100% Height=10% Behavior = Alternate Align = Bottom>
<H1><B>INTERNET CONTROLLED ROBOT<B></H1></blink></MARQUEE>
</FONT>
</BODY>
</HTML>
```



```
<HTML>
<HEAD>
<TITLE>CONTENT</TITLE>
<FRAMESET Rows ="60%,*" FRAMEBORDER ="No" FRAMESPACING =0>
  <FRAME SRC ="c:\jesada\contents1.html">
  <FRAME SRC ="c:\jesada\contents2.html">
</FRAMESET>
</HTML>
```



```

<HTML>
<HEAD>
<TITLE>      Contents Frame </TITLE>
</HEAD>
<BODY BgColor = Silver>
  <H1 ALIGN = "CENTER"> Contents </H1>
  <HR>
  <UL TYPE =SQUARE>
    <LI><A HREF ="c:\jesada\aboutus.html" Target = "Main">ABOUT US </LI>
    <LI><A HREF ="http://www.tonn.com/bestproj2.htm" Target = "Main">CONTRO
LLING</A></LI>
  </UL>
</BODY>
</HTML>

```



<HTML>
<HEAD>
<TITLE>PICTURE SHOWS</TITLE>
<BODY BgColor = silver>

</BODY>
</HTML>



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
<HTML>
<HEAD>
<TITLE>About us</TITLE>
</HEAD>
<BODY Text = Blue>
  <IMG SRC = "C:\JESADA\ABOUTUS1.JPG">
  <B><U><H1>WELCOME TO MY PROJECT</H1></U></B>
  <H3> We designed and built a robot that you can control
  via the INTERNET and view the visual output from this site</H3>
  <U><H3>INTRODUCTION : </H3></U>
  <|><H3>SIGN FOR CONTROLLING</H3></|>
  <IMG SRC ="c:\jesada\4.gif" ALIGN ="LEFT"> -FORWARD<BR>
  <IMG SRC ="c:\jesada\2.gif" ALIGN ="LEFT"> -TURN LEFT<BR>
  <IMG SRC ="c:\jesada\3.gif" ALIGN ="LEFT"> -TURN RIGHT<BR>
  <IMG SRC ="c:\jesada\1.gif" ALIGN ="LEFT"> -TURN AROUND<P>
  <IMG SRC ="c:\jesada\grn_psh9.gif" ALIGN ="BOTTOM"> -START CONTROLLING
<P>
  <IMG SRC ="c:\jesada\led6.gif" ALIGN ="BOTTOM"> -STOP CONTROLLING<BR>
  CONTROL : You can control the movements of the robot by using these butto
ms.
  The stated direction will order it to move accordingly
</BODY>
</HTML>
```

3.ส่วนโปรแกรม Delphi



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

library bestProj1;

uses

ComServ,

bestProj1_TLB in 'bestProj1_TLB.pas',

bestImpl1 in 'bestImpl1.pas' {best: TActiveForm} {best: CoClass},

CtrlSignal in '..\CtrlSignal.pas';

{SE ocx}

exports

DllGetClassObject,

DllCanUnloadNow,

DllRegisterServer,

DllUnregisterServer;

{SR *.TLB}

{SR *.RES}

begin

end.

unit bestImpl1;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
ActiveX, AxCtrls, bestProj1_TLB, CtrlSignal, ScktComp, StdCtrls, Buttons,
ExtCtrls;

type

Tbest = class(TActiveForm, Ibest)

RecieverSocket: TClientSocket;

TransmitterSocket: TClientSocket;

Panel1: TPanel;

Image1: TImage;

ForwardButton: TSpeedButton;

ConnectButton: TSpeedButton;

DisconnectButton: TSpeedButton;

TurnAroundButton: TSpeedButton;

TurnRightButton: TSpeedButton;

TurnLeftButton: TSpeedButton;

procedure BitBtn1Click(Sender: TObject);

procedure BitBtn3Click(Sender: TObject);

procedure BitBtn2Click(Sender: TObject);

procedure BitBtn4Click(Sender: TObject);

procedure RecieverSocketRead(Sender: TObject; Socket: TCustomWinSocket);

procedure FormDestroy(Sender: TObject);

procedure ConnectButtonClick(Sender: TObject);

procedure ForwardButtonClick(Sender: TObject);

procedure TurnLeftButtonClick(Sender: TObject);

procedure TurnRightButtonClick(Sender: TObject);

procedure TurnAroundButtonClick(Sender: TObject);

procedure DisconnectButtonClick(Sender: TObject);

private

{ Private declarations }

fp: file;

WaitForData: Boolean;

FEvents: IbestEvents;

procedure ActivateEvent(Sender: TObject);

procedure ClickEvent(Sender: TObject);

procedure CreateEvent(Sender: TObject);

procedure DblClickEvent(Sender: TObject);

procedure DeactivateEvent(Sender: TObject);

procedure DestroyEvent(Sender: TObject);

procedure KeyPressEvent(Sender: TObject; var Key: Char);

procedure PaintEvent(Sender: TObject);

protected

{ Protected declarations }

procedure DefinePropertyPages(DefinePropertyPage: TDefinePropertyPage); override;

procedure EventSinkChanged(const EventSink: IUnknown); override;

function Get_Active: WordBool; safecall;

function Get_AutoScroll: WordBool; safecall;

function Get_AutoSize: WordBool; safecall;

function Get_AxBorderStyle: TxActiveFormBorderStyle; safecall;

function Get_BiDiMode: TxBiDiMode; safecall;

function Get_Caption: WideString; safecall;

function Get_Color: OLE_COLOR; safecall;

function Get_Cursor: Smallint; safecall;

function Get_DoubleBuffered: WordBool; safecall;

function Get_DropTarget: WordBool; safecall;

function Get_Enabled: WordBool; safecall;

function Get_Font: IFontDisp; safecall;

function Get_HelpFile: WideString; safecall;

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

function Get_KeyPreview: WordBool; safecall;
function Get_PixelsPerInch: Integer; safecall;
function Get_PrintScale: TxPrintScale; safecall;
function Get_Scaled: WordBool; safecall;
function Get_Visible: WordBool; safecall;
procedure _Set_Font(const Value: IFontDisp); safecall;
procedure Set_AutoScroll(Value: WordBool); safecall;
procedure Set_AutoSize(Value: WordBool); safecall;
procedure Set_AxBorderStyle(Value: TxActiveFormBorderStyle); safecall;
procedure Set_BiDiMode(Value: TxBiDiMode); safecall;
procedure Set_Caption(const Value: WideString); safecall;
procedure Set_Color(Value: OLE_COLOR); safecall;
procedure Set_Cursor(Value: Smallint); safecall;
procedure Set_DoubleBuffered(Value: WordBool); safecall;
procedure Set_DropTarget(Value: WordBool); safecall;
procedure Set_Enabled(Value: WordBool); safecall;
procedure Set_Font(const Value: IFontDisp); safecall;
procedure Set_HelpFile(const Value: WideString); safecall;
procedure Set_KeyPreview(Value: WordBool); safecall;
procedure Set_PixelsPerInch(Value: Integer); safecall;
procedure Set_PrintScale(Value: TxPrintScale); safecall;
procedure Set_Scaled(Value: WordBool); safecall;
procedure Set_Visible(Value: WordBool); safecall;
public
    { Public declarations }
    procedure Initialize; override;
end;
implementation
uses ComObj, ComServ;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

{ \$R *.DFM }

{ Tbest }

procedure Tbest.DefinePropertyPages(DefinePropertyPage: TDefinePropertyPage);

begin

{ Define property pages here. Property pages are defined by calling

DefinePropertyPage with the class id of the page. For example,

DefinePropertyPage(Class_bestPage); }

end;

procedure Tbest.EventSinkChanged(const EventSink: IUnknown);

begin

FEvents := EventSink as IbestEvents;

end;

procedure Tbest.Initialize;

begin

inherited Initialize;

OnActivate := ActivateEvent;

OnClick := ClickEvent;

OnCreate := CreateEvent;

OnDbClick := DbClickEvent;

OnDeactivate := DeactivateEvent;

OnDestroy := DestroyEvent;

OnKeyPress := KeyPressEvent;

OnPaint := PaintEvent;

end;

function Tbest.Get_Active: WordBool;

begin

Result := Active;

end;

```
function Tbest.Get_AutoScroll: WordBool;
```

```
begin
```

```
    Result := AutoScroll;
```

```
end;
```

```
function Tbest.Get_AutoSize: WordBool;
```

```
begin
```

```
    Result := AutoSize;
```

```
end;
```

```
function Tbest.Get_AxBorderStyle: TxActiveFormBorderStyle;
```

```
begin
```

```
    Result := Ord(AxBorderStyle);
```

```
end;
```

```
function Tbest.Get_BiDiMode: TxBiDiMode;
```

```
begin
```

```
    Result := Ord(BiDiMode);
```

```
end;
```

```
function Tbest.Get_Caption: WideString;
```

```
begin
```

```
    Result := WideString(Caption);
```

```
end;
```

```
function Tbest.Get_Color: OLE_COLOR;
```

```
begin
```

```
    Result := OLE_COLOR(Color);
```

```
end;
```

```
function Tbest.Get_Cursor: Smallint;
```

```
begin
```

```
    Result := Smallint(Cursor);
```

```
end;
```

```
function Tbest.Get_DoubleBuffered: WordBool;
```

```
begin
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    Result := DoubleBuffered;

end;

function Tbest.Get_DropTarget: WordBool;
begin
    Result := DropTarget;
end;

function Tbest.Get_Enabled: WordBool;
begin
    Result := Enabled;
end;

function Tbest.Get_Font: IFontDisp;
begin
    GetOleFont(Font, Result);
end;

function Tbest.Get_HelpFile: WideString;
begin
    Result := WideString(HelpFile);
end;

function Tbest.Get_KeyPreview: WordBool;
begin
    Result := KeyPreview;
end;

function Tbest.Get_PixelsPerInch: Integer;
begin
    Result := PixelsPerInch;
end;

function Tbest.Get_PrintScale: TxPrintScale;
begin
    Result := Ord(PrintScale);
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

function Tbest.Get_Scaled: WordBool;
begin
    Result := Scaled;
end;

function Tbest.Get_Visible: WordBool;
begin
    Result := Visible;
end;

procedure Tbest._Set_Font(const Value: IFontDisp);
begin
    SetOleFont(Font, Value);
end;

procedure Tbest.Set_AutoScroll(Value: WordBool);
begin
    AutoScroll := Value;
end;

procedure Tbest.Set_AutoSize(Value: WordBool);
begin
    AutoSize := Value;
end;

procedure Tbest.Set_AxBorderStyle(Value: TxActiveFormBorderStyle);
begin
    AxBorderStyle := TActiveFormBorderStyle(Value);
end;

procedure Tbest.Set_BiDiMode(Value: TxBiDiMode);
begin
    BiDiMode := TBiDiMode(Value);
end;

procedure Tbest.Set_Caption(const Value: WideString);
begin

```

```

Caption := TCaption(Value);
end;
procedure Tbest.Set_Color(Value: OLE_COLOR);
begin
    Color := TColor(Value);
end;
procedure Tbest.Set_Cursor(Value: Smallint);
begin
    Cursor := TCursor(Value);
end;
procedure Tbest.Set_DoubleBuffered(Value: WordBool);
begin
    DoubleBuffered := Value;
end;
procedure Tbest.Set_DropTarget(Value: WordBool);
begin
    DropTarget := Value;
end;
procedure Tbest.Set_Enabled(Value: WordBool);
begin
    Enabled := Value;
end;
procedure Tbest.Set_Font(const Value: IFontDisp);
begin
    SetOleFont(Font, Value);
end;
procedure Tbest.Set_HelpFile(const Value: WideString);
begin
    HelpFile := String(Value);
end;

```

```

procedure Tbest.Set_KeyPreview(Value: WordBool);
begin
    KeyPreview := Value;
end;
procedure Tbest.Set_PixelsPerInch(Value: Integer);
begin
    PixelsPerInch := Value;
end;
procedure Tbest.Set_PrintScale(Value: TxPrintScale);
begin
    PrintScale := TPrintScale(Value);
end;
procedure Tbest.Set_Scaled(Value: WordBool);
begin
    Scaled := Value;
end;
procedure Tbest.Set_Visible(Value: WordBool);
begin
    Visible := Value;
end;
procedure Tbest.ActivateEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnActivate;
end;
procedure Tbest.ClickEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnClick;
end;
procedure Tbest.CreateEvent(Sender: TObject);
begin

```

```

if FEvents <> nil then FEvents.OnCreate;
end;
procedure Tbest.DblClickEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnDblClick;
end;
procedure Tbest.DeactivateEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnDeactivate;
end;
procedure Tbest.DestroyEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnDestroy;
end;
procedure Tbest.KeyPressEvent(Sender: TObject; var Key: Char);
var
    TempKey: Smallint;
begin
    TempKey := Smallint(Key);
    if FEvents <> nil then FEvents.OnKeyPress(TempKey);
    Key := Char(TempKey);
end;
procedure Tbest.PaintEvent(Sender: TObject);
begin
    if FEvents <> nil then FEvents.OnPaint;
end;
procedure Tbest.BitBtn1Click(Sender: TObject);
begin
    TransmitterSocket.Socket.SendText('เดินหน้า');
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

procedure Tbest.BitBtn3Click(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('เลี้ยวขวา');
end;

procedure Tbest.BitBtn2Click(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('เลี้ยวซ้าย');
end;

procedure Tbest.BitBtn4Click(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('วกกลับ');
end;

procedure Tbest.RecieverSocketRead(Sender: TObject;
Socket: TCustomWinSocket);
Var
Buffer: Array[1..MAX_BUF_SIZE] of Byte;
BufLen: Integer;
begin
//FillChar(Buffer, SizeOf(Buffer), #00);
BufLen := Socket.ReceiveBuf(Buffer, MAX_BUF_SIZE);
if WaitForData then
begin
If BufLen > 0 Then
begin
BlockWrite(fp, Buffer, BufLen);
end;
If BufLen < MAX_BUF_SIZE then
begin
CloseFile(fp);

Image1.Picture.LoadFromFile('ton.bmp');

```

```

//Image1.Invalidate;
WaitForData := False;
end
else
    Socket.SendText(CS_ACK);
end
else
begin
    if (BufLen = 1) and (Buffer[1] = Byte(CS_START)) then
    begin
        AssignFile(fp,'ton.bmp');
        Rewrite(fp, 1);
        WaitForData := True;
        Socket.SendText(CS_ACK);
    end;
end;
end;
procedure Tbest.FormDestroy(Sender: TObject);
begin
    DisconnectButton.Click;
end;
procedure Tbest.ConnectButtonClick(Sender: TObject);
begin
    ConnectButton.Enabled:=False;
    DisconnectButton.Enabled:=True;
    TransmitterSocket.Active :=True;
    RecieverSocket.Active:=True;
    Image1.Show;
    ForwardButton.Enabled:=True;
    TurnRightButton.Enabled:=True;

```

```

TurnLeftButton.Enabled:=True;
TurnAroundButton.Enabled:=True;
end;
procedure Tbest.ForwardButtonClick(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('เดินหน้า');
end;
procedure Tbest.TurnLeftButtonClick(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('เลี้ยวซ้าย');
end;
procedure Tbest.TurnRightButtonClick(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('เลี้ยวขวา');
end;
procedure Tbest.TurnAroundButtonClick(Sender: TObject);
begin
TransmitterSocket.Socket.SendText('วกกลับ');
end;
procedure Tbest.DisconnectButtonClick(Sender: TObject);
begin
ConnectButton.Enabled:=True;
DisconnectButton.Enabled:=False;
TransmitterSocket.Active :=False;
RecieverSocket.Active:=False;
Image1.Hide;
ForwardButton.Enabled:=False;
TurnRightButton.Enabled:=False;
TurnLeftButton.Enabled:=False;
TurnAroundButton.Enabled:=False;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

end;

initialization

TActiveFormFactory.Create(

ComServer,

TActiveFormControl,

Tbest,

Class_best,

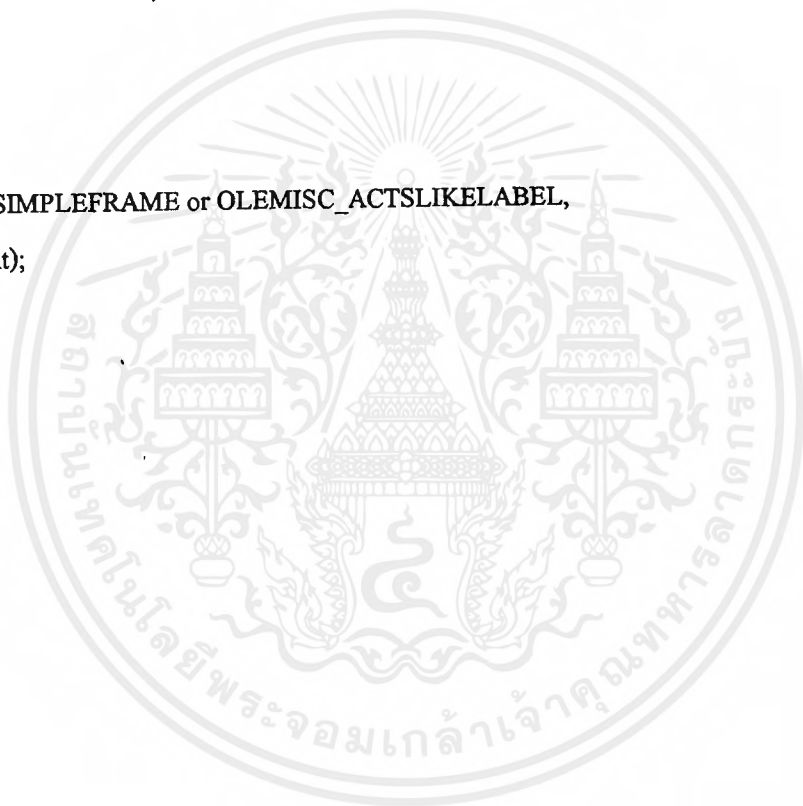
1,

",

OLEMISC_SIMPLEFRAME or OLEMISC_ACTSLIKELABEL,

tmApartment);

end.



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

unit bestProj1_TLB;

// ***** //

// WARNING //

// ----- //

// The types declared in this file were generated from data read from a //
// Type Library. If this type library is explicitly or indirectly (via //
// another type library referring to this type library) re-imported, or the //
// 'Refresh' command of the Type Library Editor activated while editing the //
// Type Library, the contents of this file will be regenerated and all //
// manual modifications will be lost. //

// ***** //

// PASTLWTR : $Revision: 1.11.1.62 $

// File generated on 15/3/99 21:02:44 from Type Library described below.

// ***** //

// Type Lib: D:\programming\Delphi4\Projects\IntObCar\Client\bestProj1.tlb
// IID\LCID: {70A49CE0-DB18-11D2-B0BE-0020185371EC}\0
// Helpfile:
// HelpString: bestProj1 Library
// Version: 1.0

// ***** //

interface

uses Windows, ActiveX, Classes, Graphics, OleCtrls, StdVCL;

// *****//

// GUIDS declared in the TypeLibrary. Following prefixes are used: //

// Type Libraries : LIBID_xxxx //

// CoClasses : CLASS_xxxx //

// DISPInterfaces : DIID_xxxx //

// Non-DISP interfaces: IID_xxxx //

// *****//

const

```

```

LIBID_bestProj1: TGUID = '{70A49CE0-DB18-11D2-B0BE-0020185371EC}';
IID_Ibest: TGUID = '{70A49CE1-DB18-11D2-B0BE-0020185371EC}';
IID_IbestEvents: TGUID = '{70A49CE3-DB18-11D2-B0BE-0020185371EC}';
CLASS_best: TGUID = '{70A49CE5-DB18-11D2-B0BE-0020185371EC}';

// *****//

// Declaration of Enumerations defined in Type Library      //
// *****//

// TxActiveFormBorderStyle constants
type
    TxActiveFormBorderStyle = TOleEnum;
const
    afbNone = $00000000;
    afbSingle = $00000001;
    afbSunken = $00000002;
    afbRaised = $00000003;
// TxPrintScale constants
type
    TxPrintScale = TOleEnum;
const
    poNone = $00000000;
    poProportional = $00000001;
    poPrintToFit = $00000002;
// TxMouseButton constants
type
    TxMouseButton = TOleEnum;
const
    mbLeft = $00000000;
    mbRight = $00000001;
    mbMiddle = $00000002;

// 'TxBiDiMode constants

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

type

TxBiDiMode = TOleEnum;

const

bdLeftToRight = \$00000000;

bdRightToLeft = \$00000001;

bdRightToLeftNoAlign = \$00000002;

bdRightToLeftReadingOnly = \$00000003;

type

// *****//

// Forward declaration of interfaces defined in Type Library //

// *****//

Ibest = interface;

IbestDisp = dispinterface;

IbestEvents = dispinterface;

// *****//

// Declaration of CoClasses defined in Type Library //

// (NOTE: Here we map each CoClass to its Default Interface) //

// *****//

best = Ibest;

// *****//

// Interface: Ibest

// Flags: (4416) Dual OleAutomation Dispatchable

// GUID: {70A49CE1-DB18-11D2-B0BE-0020185371EC}

// *****//

Ibest = interface(IDispatch)

['{70A49CE1-DB18-11D2-B0BE-0020185371EC}']

function Get_Visible: WordBool; safecall;

procedure Set_Visible(Value: WordBool); safecall;

function Get_AutoScroll: WordBool; safecall;

procedure Set_AutoScroll(Value: WordBool); safecall;

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

function Get_AutoSize: WordBool; safecall;
procedure Set_AutoSize(Value: WordBool); safecall;
function Get_AxBorderStyle: TxActiveFormBorderStyle; safecall;
procedure Set_AxBorderStyle(Value: TxActiveFormBorderStyle); safecall;
function Get_Caption: WideString; safecall;
procedure Set_Caption(const Value: WideString); safecall;
function Get_Color: OLE_COLOR; safecall;
procedure Set_Color(Value: OLE_COLOR); safecall;
function Get_Font: IFontDisp; safecall;
procedure _Set_Font(const Value: IFontDisp); safecall;
procedure Set_Font(const Value: IFontDisp); safecall;
function Get_KeyPreview: WordBool; safecall;
procedure Set_KeyPreview(Value: WordBool); safecall;
function Get_PixelsPerInch: Integer; safecall;
procedure Set_PixelsPerInch(Value: Integer); safecall;
function Get_PrintScale: TxPrintScale; safecall;
procedure Set_PrintScale(Value: TxPrintScale); safecall;
function Get_Scaled: WordBool; safecall;
procedure Set_Scaled(Value: WordBool); safecall;
function Get_Active: WordBool; safecall;
function Get_DropTarget: WordBool; safecall;
procedure Set_DropTarget(Value: WordBool); safecall;
function Get_HelpFile: WideString; safecall;
procedure Set_HelpFile(const Value: WideString); safecall;
function Get_DoubleBuffered: WordBool; safecall;
procedure Set_DoubleBuffered(Value: WordBool); safecall;
function Get_Enabled: WordBool; safecall;
procedure Set_Enabled(Value: WordBool); safecall;
function Get_BiDiMode: TxBiDiMode; safecall;
procedure Set_BiDiMode(Value: TxBiDiMode); safecall;

```

function Get_Cursor: Smallint; safecall;

procedure Set_Cursor(Value: Smallint); safecall;

property Visible: WordBool read Get_Visible write Set_Visible;

property AutoScroll: WordBool read Get_AutoScroll write Set_AutoScroll;

property AutoSize: WordBool read Get_AutoSize write Set_AutoSize;

property AxBorderStyle: TxActiveFormBorderStyle read Get_AxBorderStyle write
Set_AxBorderStyle;

property Caption: WideString read Get_Caption write Set_Caption;

property Color: OLE_COLOR read Get_Color write Set_Color;

property Font: IFontDisp read Get_Font write _Set_Font;

property KeyPreview: WordBool read Get_KeyPreview write Set_KeyPreview;

property PixelsPerInch: Integer read Get_PixelsPerInch write Set_PixelsPerInch;

property PrintScale: TxPrintScale read Get_PrintScale write Set_PrintScale;

property Scaled: WordBool read Get_Scaled write Set_Scaled;

property Active: WordBool read Get_Active;

property DropTarget: WordBool read Get_DropTarget write Set_DropTarget;

property HelpFile: WideString read Get_HelpFile write Set_HelpFile;

property DoubleBuffered: WordBool read Get_DoubleBuffered write Set_DoubleBuffered;

property Enabled: WordBool read Get_Enabled write Set_Enabled;

property BiDiMode: TxBiDiMode read Get_BiDiMode write Set_BiDiMode;

property Cursor: Smallint read Get_Cursor write Set_Cursor;

end;

// *****//

// DispIntf: IbestDisp

// Flags: (4416) Dual OleAutomation Dispatchable

// GUID: {70A49CE1-DB18-11D2-B0BE-0020185371EC}

// *****//

IbestDisp = dispinterface

['{70A49CE1-DB18-11D2-B0BE-0020185371EC}']

property Visible: WordBool dispid 1;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
property AutoScroll: WordBool dispid 2;
property AutoSize: WordBool dispid 3;
property AxBorderStyle: TxActiveFormBorderStyle dispid 4;
property Caption: WideString dispid -518;
property Color: OLE_COLOR dispid -501;
property Font: IFontDisp dispid -512;
property KeyPreview: WordBool dispid 5;
property PixelsPerInch: Integer dispid 6;
property PrintScale: TxPrintScale dispid 7;
property Scaled: WordBool dispid 8;
property Active: WordBool readonly dispid 9;
property DropTarget: WordBool dispid 10;
property HelpFile: WideString dispid 11;
property DoubleBuffered: WordBool dispid 12;
property Enabled: WordBool dispid -514;
property BiDiMode: TxBiDiMode dispid 13;
property Cursor: Smallint dispid 14;
end;
```

```
// *****//
```

```
// DispIntf: IbestEvents
```

```
// Flags: (0)
```

```
// GUID: {70A49CE3-DB18-11D2-B0BE-0020185371EC}
```

```
// *****//
```

```
IbestEvents = dispinterface
```

```
['{70A49CE3-DB18-11D2-B0BE-0020185371EC}']
```

```
procedure OnActivate; dispid 1;
```

```
procedure OnClick; dispid 2;
```

```
procedure OnCreate; dispid 3;
```

```
procedure OnDblClick; dispid 4;
```

```
procedure OnDestroy; dispid 5;
```

```

procedure OnDeactivate; dispid 6;

procedure OnKeyPress(var Key: Smallint); dispid 10;

procedure OnPaint; dispid 15;

end;

// *****//

// OLE Control Proxy class declaration

// Control Name    : Tbest

// Help String     : best Control

// Default Interface: Ibest

// Def. Intf. DISP? : No

// Event Interface: IbestEvents

// TypeFlags       : (34) CanCreate Control

// *****//

TbestOnKeyPress = procedure(Sender: TObject; var Key: Smallint) of object;

Tbest = class(TOleControl)

private

    FOnActivate: TNotifyEvent;

    FOnClick: TNotifyEvent;

    FOnCreate: TNotifyEvent;

    FOnDblClick: TNotifyEvent;

    FOnDestroy: TNotifyEvent;

    FOnDeactivate: TNotifyEvent;

    FOnKeyPress: TbestOnKeyPress;

    FOnPaint: TNotifyEvent;

    FIntf: Ibest;

    function GetControlInterface: Ibest;

protected

    procedure CreateControl;

    procedure InitControlData; override;

public

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

property ControlInterface: Ibest read GetControlInterface;

property Visible: WordBool index 1 read GetWordBoolProp write SetWordBoolProp;

property Active: WordBool index 9 read GetWordBoolProp;

property DropTarget: WordBool index 10 read GetWordBoolProp write SetWordBoolProp;

property HelpFile: WideString index 11 read GetWideStringProp write SetWideStringProp;

property DoubleBuffered: WordBool index 12 read GetWordBoolProp write
SetWordBoolProp;

property Enabled: WordBool index -514 read GetWordBoolProp write SetWordBoolProp;

property BiDiMode: TOleEnum index 13 read GetTOleEnumProp write SetTOleEnumProp;

published

property AutoScroll: WordBool index 2 read GetWordBoolProp write SetWordBoolProp
stored False;

property AutoSize: WordBool index 3 read GetWordBoolProp write SetWordBoolProp stored
False;

property AxBorderStyle: TOleEnum index 4 read GetTOleEnumProp write SetTOleEnumProp
stored False;

property Caption: WideString index -518 read GetWideStringProp write SetWideStringProp
stored False;

property Color: TColor index -501 read GetTColorProp write SetTColorProp stored False;

property Font: TFont index -512 read GetTFontProp write SetTFontProp stored False;

property KeyPreview: WordBool index 5 read GetWordBoolProp write SetWordBoolProp
stored False;

property PixelsPerInch: Integer index 6 read GetIntegerProp write SetIntegerProp stored False;

property PrintScale: TOleEnum index 7 read GetTOleEnumProp write SetTOleEnumProp
stored False;

property Scaled: WordBool index 8 read GetWordBoolProp write SetWordBoolProp stored
False;

property Cursor: Smallint index 14 read GetSmallintProp write SetSmallintProp stored False;

property OnActivate: TNotifyEvent read FOnActivate write FOnActivate;

property OnClick: TNotifyEvent read FOnClick write FOnClick;

```
property OnCreate: TNotifyEvent read FOnCreate write FOnCreate;
property OnDbClick: TNotifyEvent read FOnDbClick write FOnDbClick;
property OnDestroy: TNotifyEvent read FOnDestroy write FOnDestroy;
property OnDeactivate: TNotifyEvent read FOnDeactivate write FOnDeactivate;
property OnKeyPress: TKeyPress read FOnKeyPress write FOnKeyPress;
property OnPaint: TNotifyEvent read FOnPaint write FOnPaint;
end;
```

```
procedure Register;
```

```
implementation
```

```
uses ComObj;
```

```
procedure Tbest.InitControlData;
```

```
const
```

```
CEventDispIDs: array [0..7] of DWORD = (
    $00000001, $00000002, $00000003, $00000004, $00000005, $00000006,
    $0000000A, $0000000F);
```

```
CTFontIDs: array [0..0] of DWORD = (
    $FFFFFFE0);
```

```
CControlData: TControlData = (
    ClassID: '{70A49CE5-DB18-11D2-B0BE-0020185371EC}';
    EventIID: '{70A49CE3-DB18-11D2-B0BE-0020185371EC}';
```

```
    EventCount: 8;
```

```
    EventDispIDs: @CEventDispIDs;
```

```
    LicenseKey: nil;
```

```
    Flags: $0000001D;
```

```
    Version: 300;
```

```
    FontCount: 1;
```

```
    FontIDs: @CTFontIDs);
```

```
begin
```

```
    ControlData := @CControlData;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

end;

procedure Tbest.CreateControl;

  procedure DoCreate;

  begin
    FIntf := IUnknown(OleObject) as Ibest;
  end;

begin
  if FIntf = nil then DoCreate;
end;

function Tbest.GetControlInterface: Ibest;
begin
  CreateControl;
  Result := FIntf;
end;

procedure Register;
begin
  RegisterComponents('ActiveX',[Tbest]);
end;

end.

```

```
unit CtrlSignal;
interface
const
    CS_START = 'a';
    CS_ACK   = 'b';
    CS_END   = 'c';
    MAX_BUF_SIZE = 1024*8;
implementation
end.
```



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

unit SeverUnit;

interface

uses

Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, Buttons, ScktComp, ComCtrls, SmallPort, ExtCtrls, VideoCap,
CtrlSignal;

const

TURN_LEFT: Word = 4;
TURN_AROUND: Word = 5;
TURN_RIGHT: Word = 6;
GO_FORWARD: Word = 8;

type

TSeverForm = class(TForm)

TransmitterSocket: TServerSocket;

Btn_Start: TBitBtn;

StatusBar: TStatusBar;

SmallPort: TSmallPort;

VideoSourceBtn: TBitBtn;

VideoLabel: TLabel;

VideoPanel: TPanel;

VideoArea: TPanel;

Timer: TTimer;

RecieverSocket: TServerSocket;

procedure TransmitterSocketAccept(Sender: TObject;

Socket: TCustomWinSocket);

procedure TransmitterSocketClientDisconnect(Sender: TObject;

Socket: TCustomWinSocket);

procedure Btn_StartClick(Sender: TObject);

procedure TransmitterSocketClientRead(Sender: TObject;

Socket: TCustomWinSocket);

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

procedure VideoSourceBtnClick(Sender: TObject);
procedure TimerTimer(Sender: TObject);
procedure TransmitterSocketClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);
procedure FormCreate(Sender: TObject);
procedure RecieverSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
procedure RecieverSocketClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
procedure RecieverSocketClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
  { Private declarations }
  MyPort: Word;
  FirstTimeOpened: Boolean;
  FileIsOpened: Boolean ;
  fp: file;
  procedure CapStatus(Sender: TObject);
public
  { Public declarations }
end;
var
  SeverForm: TSeverForm;
implementation
  {$R *.DFM}
  procedure TSeverForm.CAPStatus(Sender: TObject);
  begin
    // Panel1.Color := clRed;

    // Panel1.Refresh;

```

```

//StopVideoBtn.Enabled := FALSE;
//StartVideoBtn.Enabled := TRUE;
end;
procedure TSeverForm.TransmitterSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar.Panels[0].Text := 'Connected to: ' + Socket.RemoteAddress;
end;
procedure TSeverForm.TransmitterSocketClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  Statusbar.Panels[0].Text := 'Client Disconnected : ';
  Timer.Enabled := FALSE;
  if FileIsOpened Then CloseFile(fp);
  FileIsOpened:=False;
end;
procedure TSeverForm.Btn_StartClick(Sender: TObject);
var
  MyCapStatusProc: TCapStatusProc;
begin
  MyPort := $3f8;
  TransmitterSocket.Active := True;
  RecieverSocket.Active:=True;
  SmallPort.OpenDriver;
  CapSetVideoArea( VideoArea );
  CapSetInfoLabel( VideoLabel );
  MyCapStatusProc := CAPStatus;
  CapSetStatusProcedure( MyCapStatusProc );
  if CapOpenDriver then
    begin

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

CapSetCapSec( 15 * 3 );
CapShow;
VideoPanel.Color := $00FC3934;
if CapHasDlgVSource then VideoSourceBtn.Enabled := TRUE;
end;
Btn_Start.Enabled := False;
end;
procedure TSeverForm.TransmitterSocketClientRead(Sender: TObject;
Socket: TCustomWinSocket);
Var
Data: string;
Buffer: Array[1..MAX_BUF_SIZE] of Char;
BufLen: Integer;
begin
try
Data:=Socket.ReceiveText;
if Data = CS_ACK then
begin
if FirstTimeOpened then
begin
AssignFile(fp, 'stillimage.tmp');
Reset(fp, 1);
FirstTimeOpened := False;
FileIsOpened:=True;
end;
//FillChar(Buffer, SizeOf(Buffer), #00);
BlockRead(fp, Buffer, SizeOf(Buffer), BufLen);
If BufLen > 0 then
Begin
if TransmitterSocket.Socket.Connected then

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

TransmitterSocket.Socket.Connections[0].SendBuf(Buffer,BufLen);
end;
If BufLen < MAX_BUF_SIZE then
begin
    CloseFile(fp);
    Timer.Enabled := True;
    FileIsOpened:=False;
end;
Data := 'Sending...';
end;
StatusBar.Panels[1].Text := Data;
except
end;
end;
procedure TSeverForm.VideoSourceBtnClick(Sender: TObject);
begin
    CapDlgVSource;
end;
procedure TSeverForm.TimerTimer(Sender: TObject);
var
    SingleImageFileName: string;
begin
    // Save Video as Bitmap to file in TEMP-Path
    SingleImageFileName := 'stillimage.tmp';
    CapSetSingleImageFileName(SingleImageFileName);
    CapGrabSingleFrame;
    CapSetVideoLive;
    if TransmitterSocket.Socket.Connected then
        TransmitterSocket.Socket.Connections[0].SendText(CS_START);
    Timer.Enabled := False;

```

```

FirstTimeOpened := True;
end;
procedure TSeverForm.TransmitterSocketClientConnect(Sender: TObject;
Socket: TCustomWinSocket);
begin
Timer.Enabled := TRUE;
end;
procedure TSeverForm.FormCreate(Sender: TObject);
begin
FirstTimeOpened := True;
FileIsOpened:=False;
end;
procedure TSeverForm.RecieverSocketAccept(Sender: TObject;
Socket: TCustomWinSocket);
begin
StatusBar.Panels[0].Text := 'Connected to: ' + Socket.RemoteAddress;
end;
procedure TSeverForm.RecieverSocketClientDisconnect(Sender: TObject;
Socket: TCustomWinSocket);
begin
StatusBar.Panels[0].Text := 'Client DisConnected : ';
end;
procedure TSeverForm.RecieverSocketClientRead(Sender: TObject;
Socket: TCustomWinSocket);
Var
Data: string;
begin
Data:=Socket.ReceiveText;
if Data = 'เดินหน้า' then
SmallPort.Port[MyPort] := GO_FORWARD

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

else
if Data = 'เลี้ยวซ้าย' then
    SmallPort.Port[MyPort].:= TURN_LEFT
else
if Data = 'เลี้ยวขวา' then
    SmallPort.Port[MyPort] := TURN_RIGHT
else
if Data = 'วกกลับ' then
    SmallPort.Port[MyPort] := TURN_AROUND;
    StatusBar.Panels[1].Text := Data;
end;
procedure TSeverForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    CapCloseDriver;
    TransmitterSocket.Active := False;
    RecieverSocket.Active:=False;
end;
end. Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms, Dialogs,
StdCtrls, Buttons, ScktComp, ComCtrls, SmallPort, ExtCtrls, VideoCap,
CtrlSignal;
const
    TURN_LEFT: Word = 4;
    TURN_AROUND: Word = 5;
    TURN_RIGHT: Word = 6;
    GO_FORWARD: Word = 8;
type
    TSeverForm = class(TForm)
        TransmitterSocket: TServerSocket;
        Btn_Start: TBitBtn;
        StatusBar: TStatusBar;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

SmallPort: TSmallPort;
VideoSourceBtn: TBitBtn;
VideoLabel: TLabel;
VideoPanel: TPanel;
VideoArea: TPanel;
Timer: TTimer;
RecieverSocket: TServerSocket;
procedure TransmitterSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
procedure TransmitterSocketClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
procedure Btn_StartClick(Sender: TObject);
procedure TransmitterSocketClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
procedure VideoSourceBtnClick(Sender: TObject);
procedure TimerTimer(Sender: TObject);
procedure TransmitterSocketClientConnect(Sender: TObject;
  Socket: TCustomWinSocket);
procedure FormCreate(Sender: TObject);
procedure RecieverSocketAccept(Sender: TObject;
  Socket: TCustomWinSocket);
procedure RecieverSocketClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
procedure RecieverSocketClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
procedure FormClose(Sender: TObject; var Action: TCloseAction);
private
  { Private declarations }
  MyPort: Word;
  FirstTimeOpened: Boolean;

```

```

FileIsOpened: Boolean ;

fp: file;

procedure CapStatus(Sender: TObject);

public
{ Public declarations }

end;

var
SeverForm: TSeverForm;

implementation
{$R *.DFM}

procedure TSeverForm.CAPStatus(Sender: TObject);
begin
// Panel1.Color := clRed;
// Panel1.Refresh;
//StopVideoBtn.Enabled := FALSE;
//StartVideoBtn.Enabled := TRUE;
end;

procedure TSeverForm.TransmitterSocketAccept(Sender: TObject;
Socket: TCustomWinSocket);
begin
Statusbar.Panels[0].Text := 'Connected to: ' + Socket.RemoteAddress;
end;

procedure TSeverForm.TransmitterSocketClientDisconnect(Sender: TObject;
Socket: TCustomWinSocket);
begin
Statusbar.Panels[0].Text := 'Client Disconnected : ';
Timer.Enabled := FALSE;
if FileIsOpened Then CloseFile(fp);
FileIsOpened:=False;
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

procedure TSeverForm.Btn_StartClick(Sender: TObject);

var
    MyCapStatusProc: TCapStatusProc;
begin
    MyPort := $3f8;
    TransmitterSocket.Active := True;
    RecieverSocket.Active:=True;
    SmallPort.OpenDriver;
    CapSetVideoArea( VideoArea );
    CapSetInfoLabel( VideoLabel );
    MyCapStatusProc := CAPStatus;
    CapSetStatusProcedure( MyCapStatusProc );
    if CapOpenDriver then
    begin
        CapSetCapSec( 15 * 3 );
        CapShow;
        VideoPanel.Color := $00FC3934;
        if CapHasDlgVSource then VideoSourceBtn.Enabled := TRUE;
    end;
    Btn_Start.Enabled := False;
end;

procedure TSeverForm.TransmitterSocketClientRead(Sender: TObject;
    Socket: TCustomWinSocket);

Var
    Data: string;
    Buffer: Array[1..MAX_BUF_SIZE] of Char;
    BufLen: Integer;
begin
    try
        Data:=Socket.ReceiveText;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

if Data = CS_ACK then
begin
  if FirstTimeOpened then
  begin
    AssignFile(fp, 'stillimage.tmp');
    Reset(fp, 1);
    FirstTimeOpened := False;
    FileIsOpened:=True;
  end;
  //FillChar(Buffer, SizeOf(Buffer), #00);
  BlockRead(fp, Buffer, SizeOf(Buffer), BufLen);
  If BufLen > 0 then
  Begin
    if TransmitterSocket.Socket.Connected then
      TransmitterSocket.Socket.Connections[0].SendBuf(Buffer,BufLen);
  end;
  If BufLen < MAX_BUF_SIZE then
  begin
    CloseFile(fp);
    Timer.Enabled := True;
    FileIsOpened:=False;
  end;
  Data := 'Sending...'
end;
StatusBar.Panels[1].Text := Data;
except
end;
end;
procedure TSeverForm.VideoSourceBtnClick(Sender: TObject);
begin

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

CapDlgVSource;

end;

procedure TSeverForm.TimerTimer(Sender: TObject);

var

    SingleImageFileName: string;

begin
    // Save Video as Bitmap to file in TEMP-Path
    SingleImageFileName := 'stillimage.tmp';
    CapSetSingleImageFileName(SingleImageFileName);
    CapGrabSingleFrame;
    CapSetVideoLive;
    if TransmitterSocket.Socket.Connected then
        TransmitterSocket.Socket.Connections[0].SendText(CS_START);
    Timer.Enabled := False;
    FirstTimeOpened := True;
end;

procedure TSeverForm.TransmitterSocketClientConnect(Sender: TObject;
    Socket: TCustomWinSocket);

begin
    Timer.Enabled := TRUE;
end;

procedure TSeverForm.FormCreate(Sender: TObject);

begin
    FirstTimeOpened := True;
    FileIsOpened := False;
end;

procedure TSeverForm.RecieverSocketAccept(Sender: TObject;
    Socket: TCustomWinSocket);

begin
    StatusBar.Panels[0].Text := 'Connected to: ' + Socket.RemoteAddress;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

end;

procedure TSeverForm.RecieverSocketClientDisconnect(Sender: TObject;
  Socket: TCustomWinSocket);
begin
  StatusBar.Panels[0].Text := 'Client DisConnected : ';
end;

procedure TSeverForm.RecieverSocketClientRead(Sender: TObject;
  Socket: TCustomWinSocket);
Var
  Data: string;
begin
  Data:=Socket.ReceiveText;
  if Data = 'เดินหน้า' then
    SmallPort.Port[MyPort] := GO_FORWARD
  else
    if Data = 'เลี้ยวซ้าย' then
      SmallPort.Port[MyPort] := TURN_LEFT
    else
      if Data = 'เลี้ยวขวา' then
        SmallPort.Port[MyPort] := TURN_RIGHT
      else
        if Data = 'วกกลับ' then
          SmallPort.Port[MyPort] := TURN_AROUND;
          StatusBar.Panels[1].Text := Data;
        end;
      end;
    end;
  end;

procedure TSeverForm.FormClose(Sender: TObject; var Action: TCloseAction);
begin
  CapCloseDriver;
  TransmitterSocket.Active := False;
  RecieverSocket.Active:=False;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

end;
end.



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

unit VideoCap;

interface

uses Windows, Dialogs, Controls, SysUtils, StdCtrls, MMSystem, AVICap;

const

MAXVIDDRIVERS = 10;

MS_FOR_15FPS = 66;

MS_FOR_20FPS = 50;

MS_FOR_30FPS = 33;

MS_FOR_25FPS = 40; // rate in msec

type

TCapStatusProc = procedure(Sender: TObject) of object;

var

ghCapWnd : THandle;

gCapVideoArea : TWinControl;

gCapVideoDriverName : string;

gdwCapNofMaxVideoFrame : DWord;

gCapVideoFileName : string;

gCapSingleImageFileName : string;

gCapVideoInfoLabel : TLabel;

gCapStatusProcedure : TCapStatusProc;

procedure CapSetVideoArea(Container: TWinControl);

procedure CapSetVideoFileName(FileName : string);

procedure CapSetSingleImageFileName(FileName : string);

procedure CapSetInfoLabel(InfoLabel : TLabel);

procedure CapSetStatusProcedure(StatusProc : TCapStatusProc);

function CapOpenDriver : Boolean;

function CapInitDriver(Index : Integer): Boolean;

procedure CapCloseDriver;

procedure CapShow;

procedure CapSetCapSec(NofMaxVideoFrame : Integer);

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

procedure CapStart;
procedure CapStop;
function CapHasDlgVFormat : Boolean;
function CapHasDlgVDisplay : Boolean;
function CapHasDlgVSource : Boolean;
procedure CapDlgVFormat;
procedure CapDlgVDisplay;
procedure CapDlgVSource;
procedure CapSetVideoOverlay;
procedure CapSetVideoLive;
procedure CapGrabSingleFrame;
implementation
(*-----*)
(*--CAP-VIDEO DRIVER --*)
(*-----*)
(*-----*)
procedure CapSetVideoArea( Container: TWinControl );
begin
    gCapVideoArea := Container;
end;
(*-----*)
procedure CapSetVideoFileName( FileName : string );
begin
    gCapVideoFileName := FileName;
end;
(*-----*)
procedure CapSetSingleImageFileName( FileName : string );
begin
    gCapSingleImageFileName := FileName;
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

(*-----*)
procedure CapSetInfoLabel( InfoLabel : TLabel );
begin
    gCapVideoInfoLabel := InfoLabel;
end;
(*-----*)
procedure CapSetStatusProcedure( StatusProc : TCapStatusProc );
begin
    gCapStatusProcedure := StatusProc;
end;
(*-----*)
(* -- Video For Windows' Status Callback Function --- *)
(*-----*)
function StatusCallbackProc(hWnd : HWND; nID : Integer; lpsz : LongInt): LongInt; stdcall;
var
    TmpStr    : string;
    dwVideoNum : Integer;
begin
    // hWnd:      Application main window handle
    // nID:        Status code for the current status
    // lpStatusText: Status text string for the current status
    TmpStr := StrPas(PChar(lpsz));
    gCapVideoInfoLabel.Caption := TmpStr;
    gCapVideoInfoLabel.Refresh;
    if nID = IDS_CAP_STAT_VIDEOCURRENT then
    begin
        dwVideoNum := StrToInt( Copy(TmpStr, 0, Pos(' ', TmpStr)-1));
        if dwVideoNum >= gdwCapNofMaxVideoFrame then
            begin

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        capCaptureAbort(ghCapWnd);
        if @gCapStatusProcedure <> nil then gCapStatusProcedure(nil);
    end;
end;
Result := 1;
end;
(*-----*)
function CapOpenDriver : Boolean;
var
    Retc      : LongInt;
    DriverIndex : Integer;
    DriverStarted : boolean;
    achDeviceName : array [0..80] of Char;
    achDeviceVersion : array [0..100] of Char;
    achFileName : array [0..255] of Char;
begin
    Result := FALSE;
    if gCapVideoArea = nil then exit;
    Result := TRUE;
    // Create the Video Capture Window
    ghCapWnd := capCreateCaptureWindow( PChar('KruwoSoft'),
    WS_CHILD or WS_VISIBLE, 0, 0,
    gCapVideoArea.Width, gCapVideoArea.Height,
    gCapVideoArea.Handle, 0);
    if ghCapWnd <> 0 then
    begin
        // Install Status-Callback-Function
        retc := capSetCallbackOnStatus(ghCapWnd, LongInt(0));
        if retc <> 0 then
            begin

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

    retc := capSetCallbackOnStatus(ghCapWnd, LongInt(@StatusCallbackProc));

    if retc <> 0 then
    begin
        // Open Installed Video Driver

        DriverIndex := 0;

        repeat
            DriverStarted := CapInitDriver( DriverIndex );

            if NOT DriverStarted then DriverIndex := DriverIndex + 1;

            until (DriverStarted = TRUE) OR (DriverIndex >=
MAXVIDDRIVERS);

            // Keep Name of Video Driver
            if capGetDriverDescription( DriverIndex,
achDeviceName, 80,
achDeviceVersion, 100 ) then
            begin
                gCapVideoDriverName := string(achDeviceName);

                end;

                // Set Capture FileName
                StrPCopy(achFileName, gCapVideoFileName);

                retc := capFileSetCaptureFile(ghCapWnd, LongInt
(@achFileName));

                if retc = 0 then

                begin
                    showmessage(gCapVideoDriverName+' Error in capFileSetCaptureFile');

                    end;

                    exit;

                end;

            end;

            Result := FALSE;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้าไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

        CapCloseDriver;

        ghCapWnd := 0;
end;
(*-----*)
function CapInitDriver( Index : Integer ): Boolean;
var
    Retc      : LongInt;
    CapParms   : TCAPTUREPARMS;
begin
    Result := FALSE;
    if ghCapWnd = 0 then exit;
        // Connect to Video Capture Driver
    if capDriverConnect(ghCapWnd, Index) <> 0 then
        begin
            retc := capCaptureGetSetup(ghCapWnd, LongInt(@CapParms), sizeof(TCAPTUREPARMS));
            if retc <> 0 then
                begin
                    //
                    CapParms.dwRequestMicroSecPerFrame := 40000; // 25 FPS
                    Requested capture rate
                    //   CapParms.dwRequestMicroSecPerFrame := 100000; // 10 FPS Requested capture rate
                    CapParms.dwRequestMicroSecPerFrame := 66667; // 15 FPS
                    Requested capture rate

                    CapParms.fLimitEnabled := FALSE;

                    CapParms.fCaptureAudio := FALSE;           // NO Audio
                    CapParms.fMCIControl := FALSE;

                    CapParms.fYield := TRUE;

                    CapParms.vKeyAbort := VK_ESCAPE;

                    CapParms.fAbortLeftMouse := FALSE;

                    CapParms.fAbortRightMouse := FALSE;

```

```
retc := capCaptureSetSetup(ghCapWnd, LongInt(@CapParms), sizeof
(TCAPTUREPARMS));
```

```
if retc = 0 then exit;
```

```
end;
```

```
Result := TRUE;
```

```
end;
```

```
end;
```

```
(*-----*)
```

```
procedure CapCloseDriver;
```

```
begin
```

```
if ghCapWnd <> 0 then
```

```
begin
```

```
capSetCallbackOnStatus(ghCapWnd, LongInt(0));
```

```
capDriverDisconnect( ghCapWnd );
```

```
DestroyWindow( ghCapWnd );
```

```
ghCapWnd := 0;
```

```
end;
```

```
end;
```

```
(*-----*)
```

```
procedure CapShow;
```

```
begin
```

```
if ghCapWnd = 0 then exit;
```

```
// Start Video overlay by default
```

```
capPreviewScale(ghCapWnd, 1);
```

```
capPreviewRate(ghCapWnd, MS_FOR_25FPS);
```

```
capOverlay(ghCapWnd, 0);
```

```
capPreview(ghCapWnd, 1);
```

```
end;
```

```
(*-----*)
```

```
procedure CapSetCapSec( NofMaxVideoFrame : Integer );
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```

begin
    gdwCapNofMaxVideoFrame := DWord( NofMaxVideoFrame );
end;
(*-----*)
procedure CapStart;
begin
    if ghCapWnd = 0 then exit;
        // Start video capture to file
    capCaptureSequence( ghCapWnd );
end;
(*-----*)
procedure CapStop;
begin
    if ghCapWnd = 0 then exit;
        // Stop video capture to file
    capCaptureAbort(ghCapWnd);
end;
(*-----*)
function CapHasDlgVFormat : Boolean;
var
    CDrvCaps : TCapDriverCaps;
begin
    Result := TRUE;
    if ghCapWnd = 0 then exit;
        capDriverGetCaps(ghCapWnd, LongInt(@CDrvCaps), sizeof(TCapDriverCaps));
    Result := CDrvCaps.fHasDlgVideoFormat;
end;
(*-----*)
function CapHasDlgVDisplay : Boolean;

```

```

CDrvCaps : TCapDriverCaps;

begin
    Result := TRUE;
    if ghCapWnd = 0 then exit;

    capDriverGetCaps(ghCapWnd, LongInt(@CDrvCaps), sizeof(TCapDriverCaps));
    Result := CDrvCaps.fHasDlgVideoDisplay;
end;
(*-----*)
function CapHasDlgVSource : Boolean;
var
    CDrvCaps : TCapDriverCaps;
begin
    Result := TRUE;
    if ghCapWnd = 0 then exit;
    capDriverGetCaps(ghCapWnd, LongInt(@CDrvCaps), sizeof(TCapDriverCaps));
    Result := CDrvCaps.fHasDlgVideoSource;
end;
(*-----*)
procedure CapDlgVFormat;
begin
    if ghCapWnd = 0 then exit;
    capDlgVideoFormat(ghCapWnd);
end;
(*-----*)
procedure CapDlgVDisplay;
begin
    if ghCapWnd = 0 then exit;
    capDlgVideoDisplay(ghCapWnd);
end;

```

(*-----*)

procedure CapDlgVSource;

begin

if ghCapWnd = 0 then exit;

capDlgVideoSource(ghCapWnd);

end;

(*-----*)

procedure CapSetVideoOverlay;

begin

if ghCapWnd = 0 then exit;

capPreview(ghCapWnd, 0);

capOverlay(ghCapWnd, 1);

end;

(*-----*)

procedure CapSetVideoLive;

begin

if ghCapWnd = 0 then exit;

capOverlay(ghCapWnd, 0);

capPreviewScale(ghCapWnd, 1);

capPreviewRate(ghCapWnd, MS_FOR_25FPS);

capPreview(ghCapWnd, 1);

end;

(*-----*)

procedure CapGrabSingleFrame;

var

achSingleFileName : array [0..255] of Char;

begin

if ghCapWnd = 0 then exit;

capGrabFrame(ghCapWnd);

StrPCopy(achSingleFileName, gCapSingleImageFileName);

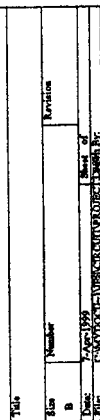
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
capFileSaveDIB(ghCapWnd, LongInt(@achSingleFileName));  
  
end;  
  
initialization  
    ghCapWnd      := 0;  
    gCapVideoArea  := nil;  
    gCapVideoDriverName := 'No Driver';  
    gdwCapNofMaxVideoFrame := 0;  
    gCapVideoFileName := 'Video.avi';  
    gCapSingleImageFileName := 'Image.bmp';  
    gCapVideoInfoLabel := nil;  
    gCapStatusProcedure := nil;  
end.
```

4.รูปวงจรรอิเล็กทรอนิกส์ และ รายละเอียดของอุปกรณ์ (Data Sheets)



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้





+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

General Description

The MAX220-MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, particularly applications where $\pm 12\text{V}$ is not available.

These parts are especially useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than $5\mu\text{W}$. The MAX225, MAX233, MAX235, and MAX245/MAX246/MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

- Portable Computers
- Low-Power Modems
- Interface Translation
- Battery-Powered RS-232 Systems
- Multi-Drop RS-232 Networks

Features

- Superior to Bipolar**
- Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)**
- Low-Power Receive Mode In Shutdown (MAX223/MAX242)**
- Meet All EIA/TIA-232E and V.28 Specifications**
- Multiple Drivers and Receivers**
- 3-State Driver and Receiver Outputs**
- Open-Line Detection (MAX243)**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.
*Contact factory for dice specifications.

Selection Table

Part Number	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (μF)	SHDN & Three-State	Rx Active In SHDN	Data Rate (kbps)	Features
MAX220	+5	2/2	4	4.7/10	No	—	120	Ultra-low-power, industry-standard pinout
MAX222	+5	2/2	4	0.1	Yes	—	200	Low-power shutdown
MAX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	—	120	MAX241 and receivers active in shutdown
MAX225	+5	5/5	0	—	Yes	—	120	Available in SO
MAX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	—	120	5 drivers with shutdown
MAX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; same functions as MAX232
MAX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No	—	120 (64)	Industry standard
MAX232A	+5	2/2	4	0.1	No	—	200	Higher slew rate, small caps
MAX233 (MAX203)	+5	2/2	0	—	No	—	120	No external caps
MAX233A	+5	2/2	0	—	No	—	200	No external caps, high slew rate
MAX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	—	120	Replaces 1488
MAX235 (MAX205)	+5	5/5	0	—	Yes	—	120	No external caps
MAX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	—	120	Shutdown, three state
MAX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	—	120	Complements IBM PC serial port
MAX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	—	120	Replaces 1488 and 1489
MAX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	—	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
MAX240	+5	5/5	4	1.0	Yes	—	120	DIP or flatpack package
MAX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	—	120	Complete IBM PC serial port
MAX242	+5	2/2	4	0.1	Yes	—	200	Separate shutdown and enable
MAX243	+5	2/2	4	0.1	No	—	200	Open-line detection simplifies cabling
MAX244	+5	8/10	4	1.0	No	—	120	High slew rate
MAX245	+5	8/10	0	—	Yes	—	120	High slew rate, int. caps, two shutdown modes
MAX246	+5	8/10	0	—	Yes	—	120	High slew rate, int. caps, three shutdown modes
MAX247	+5	8/9	0	—	Yes	—	120	High slew rate, int. caps, nine operating modes
MAX248	+5	8/8	4	1.0	Yes	—	120	High slew rate, selective half-chip enables
MAX249	+5	6/10	4	1.0	Yes	—	120	Available in quad flatpack package



For free samples & the latest literature: <http://www.maxim-ic.com>, or phone 1-800-998-8800.
For small orders, phone 408-737-7600 ext. 3468.

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งยังมีให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

Supply Voltage (VCC)	-0.3V to +6V
Input Voltages	
VIN	-0.3V to (VCC - 0.3V)
RIN	±30V
TOUT (Note 1)	±15V
Output Voltages	
TOUT	±15V
ROUT	-0.3V to (VCC + 0.3V)
Driver/Receiver Output Short Circuited to GND	Continuous
Continuous Power Dissipation (TA = +70°C)	
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C)	842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C)	889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C)	440mW

16-Pin Narrow SO (derate 8.70mW/°C above +70°C)	696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C)	762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C)	762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C)	800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C)	640mW
16-Pin Cerdip (derate 10.00mW/°C above +70°C)	800mW
18-Pin Cerdip (derate 10.53mW/°C above +70°C)	842mW
Operating Temperature Ranges	
MAX2__AC__, MAX2__C__	0°C to +70°C
MAX2__AE__, MAX2__E__	-40°C to +85°C
MAX2__AM__, MAX2__M__	-55°C to +125°C
Storage Temperature Range	-65°C to +160°C
Lead Temperature (soldering, 10sec)	+300°C

Note 1: Input voltage measured with TOUT in high-impedance state, SHDN or VCC = 0V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

(VCC = +5V ±10%, C1-C4 = 0.1μF, TA = TMIN to TMAX, unless otherwise noted.)

PARAMETER		CONDITIONS	MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	Normal operation			5	40	μA
	SHDN = 0V, MAX222/242, shutdown			±0.01	±1	
Output Leakage Current	VCC = 5.5V, SHDN = 0V, VOUT = ±15V, MAX222/242			±0.01	±10	μA
	VCC = SHDN = 0V, VOUT = ±15V			±0.01	±10	
Data Rate	All except MAX220, normal operation			200	116	kbits/sec
	MAX220			22	20	
Transmitter Output Resistance	VCC = V+ = V- = 0V, VOUT = ±2V		300	10M		Ω
Output Short-Circuit Current	VOUT = 0V		±7	±22		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	VCC = 5V	All except MAX243 R2IN	0.8	1.3		V
		MAX243 R2IN (Note 2)	-3			
RS-232 Input Threshold High	VCC = 5V	All except MAX243 R2IN		1.8	2.4	V
		MAX243 R2IN (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	All except MAX243, VCC = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	IOUT = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	IOUT = -1.0mA		3.5	VCC - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing VOUT = GND		-2	-10		mA
	Sinking VOUT = VCC		10	30		
TTL/CMOS Output Leakage Current	SHDN = VCC or EN = VCC (SHDN = 0V for MAX222), 0V ≤ VOUT ≤ VCC			±0.05	±10	μA

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued) (VCC = +5V ± 10%, C1–C4 = 0.1µF, TA = TMIN to TMAX, unless otherwise noted.)

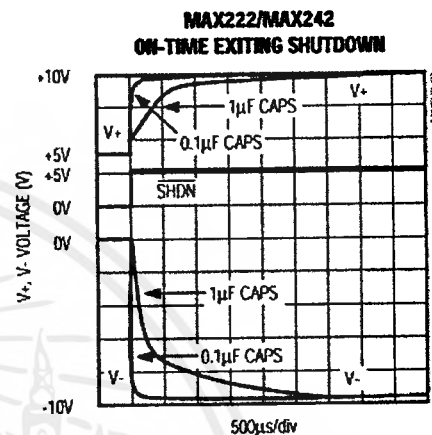
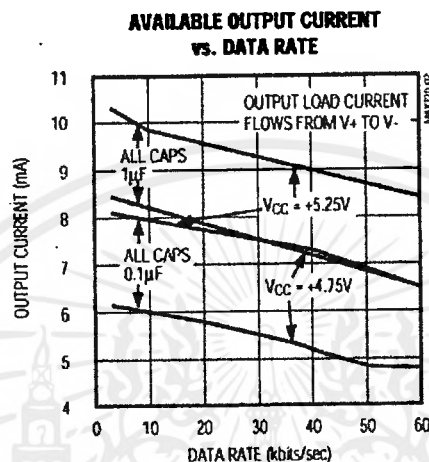
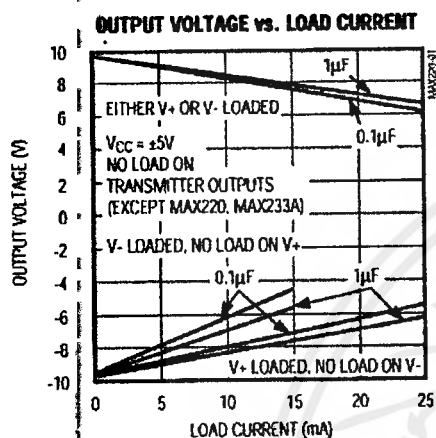
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
EN Input Threshold Low	MAX242			1.4	0.8	V
EN Input Threshold High	MAX242		2.0	1.4		V
Operating Supply Voltage			4.5		5.5	V
VCC Supply Current (SHDN = VCC). Figures 5, 6, 11, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	TA = +25°C		0.1	10	µA
		TA = 0°C to +70°C		2	50	
		TA = -40°C to +85°C		2	50	
		TA = -55°C to +125°C		35	100	
SHDN Input Leakage Current	MAX222/242				±1	µA
SHDN Threshold Low	MAX222/242			1.4	0.8	V
SHDN Threshold High	MAX222/242		2.0	1.4		V
Transition Slew Rate	CL = 50pF to 2500pF, RL = 3kΩ to 7kΩ, VCC = 5V, TA = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/µs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TLL to RS-232 (normal operation). Figure 1	tPHLT	MAX222/232A/233A/242/243		1.3	3.5	µs
		MAX220		4	10	
	tPLHT	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TLL (normal operation). Figure 2	tPHLR	MAX222/232A/233A/242/243		0.5	1	µs
		MAX220		0.6	3	
	tPLHR	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TLL (shutdown). Figure 2	tPHLS	MAX242		0.5	10	µs
	tPLHS	MAX242		2.5	10	
Receiver Output Enable Time, Figure 3	tER	MAX242		125	500	ns
Receiver Output Disable Time, Figure 3	tDR	MAX242		160	500	ns
Transmitter Output Enable Time (SHDN goes high), Figure 4	tET	MAX222/242, 0.1µF caps (includes charge-pump start-up)		250		µs
Transmitter Output Disable Time (SHDN goes low), Figure 4	tDT	MAX222/242, 0.1µF caps		600		ns
Transmitter + to - Propagation Delay Difference (normal operation)	tPHLT - tPLHT	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (normal operation)	tPHLR - tPLHR	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

Note 2: MAX243 R2OUT is guaranteed to be low when R2IN is 0V or is floating.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243



+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ABSOLUTE MAXIMUM RATINGS—MAX223/MAX230—MAX241

V _{CC}	-0.3V to +6V	
V ₊	(V _{CC} - 0.3V) to +14V	
V ₋	+0.3V to -14V	
Input Voltages		
T _{IN}	-0.3V to (V _{CC} + 0.3V)	
R _{IN}	±30V	
Output Voltages		
T _{OUT}	(V ₊ + 0.3V) to (V ₋ - 0.3V)	
R _{OUT}	-0.3V to (V _{CC} + 0.3V)	
Short-Circuit Duration, T _{OUT}		Continuous
Continuous Power Dissipation (T _A = +70°C)		
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C).....800mW		
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C).....842mW		
20-Pin Plastic DIP (derate 11.11mW/°C above +70°C).....889mW		
24-Pin Narrow Plastic DIP		
(derate 13.33mW/°C above +70°C).....1.07W		
24-Pin Plastic DIP (derate 9.09mW/°C above +70°C).....500mW		
16-Pin Wide SO (derate 9.52mW/°C above +70°C).....762mW		

20-Pin Wide SO (derate 10.00mW/°C above +70°C).....	800mW
24-Pin Wide SO (derate 11.76mW/°C above +70°C).....	941mW
28-Pin Wide SO (derate 12.50mW/°C above +70°C).....	1W
44-Pin Plastic FP (derate 11.11mW/°C above +70°C).....	889mW
14-Pin CerdIP (derate 9.09mW/°C above +70°C).....	727mW
16-Pin CerdIP (derate 10.00mW/°C above +70°C).....	800mW
20-Pin CerdIP (derate 11.11mW/°C above +70°C).....	889mW
24-Pin Narrow CerdIP	
(derate 12.50mW/°C above +70°C).....	1W
24-Pin Sidebraz (derate 20.0mW/°C above +70°C).....	1.6W
28-Pin SSOP (derate 9.52mW/°C above +70°C).....	762mW
Operating Temperature Ranges	
MAX2 __ C __	0°C to +70°C
MAX2 __ E __	-40°C to +85°C
MAX2 __ M __	-55°C to +125°C
Storage Temperature Range	-65°C to +160°C
Lead Temperature (soldering, 10sec)	+300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX223/MAX230—MAX241

(MAX223/230/232/234/236/237/238/240/241, V_{CC} = +5V ±10%; MAX233/MAX235, V_{CC} = 5V ±5%, C1-C4 = 1.0μF; MAX231/MAX239, V_{CC} = 5V ±10%; V₊ = 7.5V to 13.2V; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to ground	±5.0	±7.3		V
V _{CC} Power-Supply Current	No load, T _A = +25°C	MAX232/233		5	10
		MAX223/230/234-238/240/241		7	15
		MAX231/239		0.4	1
V ₊ Power-Supply Current		MAX231		1.8	5
		MAX239		5	15
Shutdown Supply Current	T _A = +25°C	MAX223		15	50
		MAX230/235/236/240/241		1	10
Input Logic Threshold Low	T _{IN} : EN, SHDN (MAX233); EN, SHDN (MAX230/235-241)			0.8	V
Input Logic Threshold High	T _{IN}	2.0			V
	EN, SHDN (MAX233); EN, SHDN (MAX230/235/236/240/241)	2.4			
Logic Pull-Up Current	T _{IN} = 0V		1.5	200	μA
Receiver Input Voltage Operating Range		-30		30	V

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ELECTRICAL CHARACTERISTICS—MAX223/MAX230—MAX241 (continued)

(MAX223/230/232/234/236/237/238/240/241, $V_{CC} = +5V \pm 10\%$; MAX233/MAX235, $V_{CC} = 5V \pm 5\%$, C_1 – $C_4 = 1.0\mu F$; MAX231/MAX239, $V_{CC} = 5V \pm 10\%$; $V_+ = 7.5V$ to $13.2V$; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 Input Threshold Low	$T_A = +25^\circ C$, $V_{CC} = 5V$	Normal operation SHDN = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)	0.8	1.2		V
		Shutdown (MAX223) SHDN = 0V, EN = 5V (R_{4IN} , R_{5IN})	0.6	1.5		
RS-232 Input Threshold High	$T_A = +25^\circ C$, $V_{CC} = 5V$	Normal operation SHDN = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)		1.7	2.4	V
		Shutdown (MAX223) SHDN = 0V, EN = 5V (R_{4IN} , R_{5IN})		1.5	2.4	
RS-232 Input Hysteresis	$V_{CC} = 5V$, no hysteresis in shutdown		0.2	0.5	1.0	V
RS-232 Input Resistance	$T_A = +25^\circ C$, $V_{CC} = 5V$		3	5	7	k Ω
TTL/CMOS Output Voltage Low	$I_{OUT} = 1.6mA$ (MAX231/232/233, $I_{OUT} = 3.2mA$)				0.4	V
TTL/CMOS Output Voltage High	$I_{OUT} = -1mA$		3.5	$V_{CC} - 0.4$		V
TTL/CMOS Output Leakage Current	$0V \leq R_{OUT} \leq V_{CC}$; EN = 0V (MAX223); EN = V_{CC} (MAX235–241)			0.05	± 10	μA
Receiver Output Enable Time	Normal operation	MAX223		600		ns
		MAX235/236/239/240/241		400		
Receiver Output Disable Time	Normal operation	MAX223		900		ns
		MAX235/236/239/240/241		250		
Propagation Delay	RS-232 IN to TTL/CMOS OUT, $C_L = 150pF$	Normal operation		0.5	10	μs
		SHDN = 0V (MAX223)	t_{PHLS}	4	40	
			t_{PLHS}	6	40	
Transition Region Slew Rate	MAX223/MAX230/MAX234–241, $T_A = +25^\circ C$, $V_{CC} = 5V$, $R_L = 3k\Omega$ to $7k\Omega$, $C_L = 50pF$ to $2500pF$, measured from $+3V$ to $-3V$ or $-3V$ to $+3V$		3	5.1	30	V/ μs
	MAX231/MAX232/MAX233, $T_A = +25^\circ C$, $V_{CC} = 5V$, $R_L = 3k\Omega$ to $7k\Omega$, $C_L = 50pF$ to $2500pF$, measured from $+3V$ to $-3V$ or $-3V$ to $+3V$			4	30	
Transmitter Output Resistance	$V_{CC} = V_+ = V_- = 0V$, $V_{OUT} = \pm 2V$		300			Ω
Transmitter Output Short-Circuit Current				± 10		mA

MAXIM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

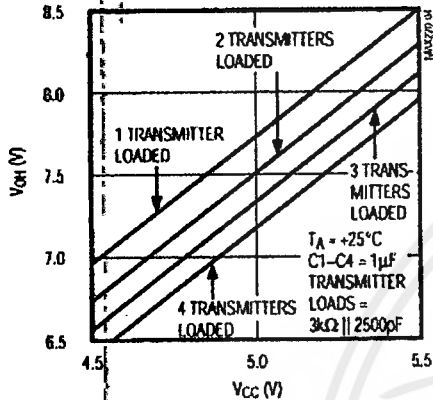
+5V-Powered, Multichannel RS-232 Drivers/Receivers

Typical Operating Characteristics

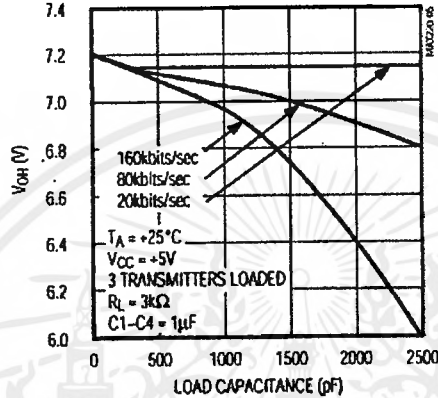
MAX223/MAX230-MAX241

MAX220-MAX249

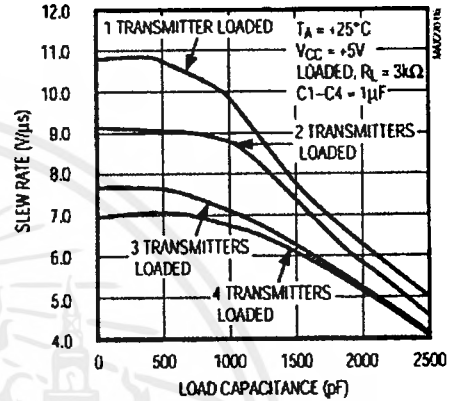
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. V_{CC}



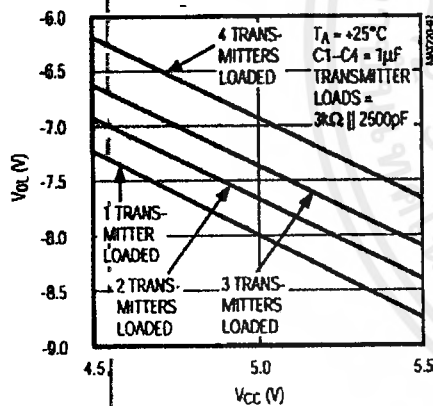
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



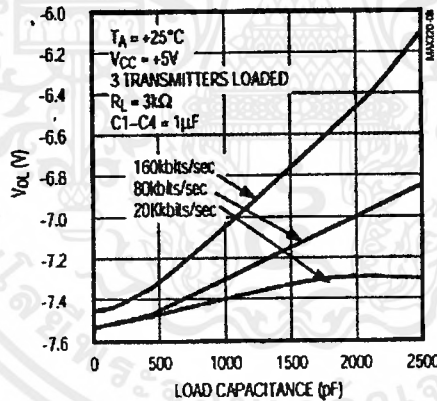
TRANSMITTER SLEW RATE vs. LOAD CAPACITANCE



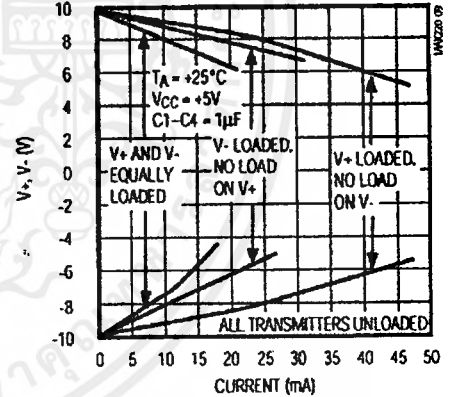
TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. V_{CC}



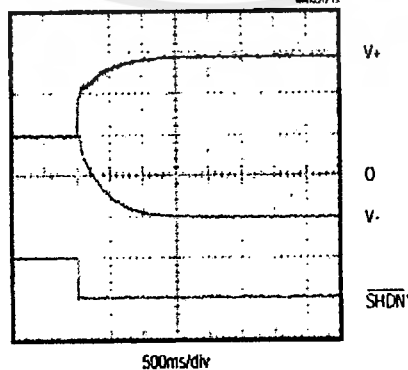
TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



TRANSMITTER OUTPUT VOLTAGE (V_+ , V_-) vs. LOAD CURRENT



V_+ , V_- WHEN EXITING SHUTDOWN ($1\mu F$ CAPACITORS)



*SHUTDOWN POLARITY IS REVERSED FOR NON MAX241 PARTS

MAXIM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX225/MAX244—MAX249

Supply Voltage (V_{CC})	-0.3V to +6V
Input Voltages	
T_{IN} , EN_A , EN_B , EN_R , ENT , EN_{RA} , EN_{RB} , EN_{TA} , EN_{TB}	-0.3V to ($V_{CC} + 0.3V$)
R_{IN}	$\pm 25V$
T_{OUT} (Note 3)	$\pm 15V$
R_{OUT}	-0.3V to ($V_{CC} + 0.3V$)
Short Circuit (one output at a time)	
T_{OUT} to GND	Continuous
R_{OUT} to GND	Continuous

Continuous Power Dissipation ($T_A = +70^\circ C$)

28-Pin Wide SO (derate 12.50mW/ $^\circ C$ above $+70^\circ C$) 1W

40-Pin Plastic DIP (derate 11.11mW/ $^\circ C$ above $+70^\circ C$) ...611mW

44-Pin PLCC (derate 13.33mW/ $^\circ C$ above $+70^\circ C$) 1.07W

Operating Temperature Ranges

MAX225C, MAX24C $0^\circ C$ to $+70^\circ C$

MAX225E, MAX24E $-40^\circ C$ to $+85^\circ C$

Storage Temperature Range $-65^\circ C$ to $+160^\circ C$

Lead Temperature (soldering, 10sec) $+300^\circ C$

Note 3: Input voltage measured with transmitter output in a high-impedance state, shutdown, or $V_{CC} = 0V$.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX225/MAX244—MAX249

(MAX225, $V_{CC} = 5.0V \pm 5\%$; MAX244-MAX249, $V_{CC} = +5.0V \pm 10\%$, external capacitors C1-C4 = $1\mu F$; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

PARAMETER	CONDITIONS	MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS					
Input Logic Threshold Low			1.4	0.8	V
Input Logic Threshold High		2	1.4		V
Logic Pull-Up/Input Current	Tables 1a-1d	Normal operation		10	50
		Shutdown		± 0.01	± 1
Data Rate	Tables 1a-1d, normal operation		120	64	kbits/sec
Output Voltage Swing	All transmitter outputs loaded with $3k\Omega$ to GND	± 5	± 7.5		V
Output Leakage Current (shutdown)	Tables 1a-1d	EN_A , EN_B , ENT , EN_{TA} , $EN_{TB} = V_{CC}$, $V_{OUT} = \pm 15V$		± 0.01	± 25
		$V_{CC} = 0V$, $V_{OUT} = \pm 15V$		± 0.01	± 25
Transmitter Output Resistance	$V_{CC} = V_+ = V_- = 0V$, $V_{OUT} = \pm 2V$ (Note 4)	300	10M		Ω
Output Short-Circuit Current	$V_{OUT} = 0V$	± 7	± 30		mA
RS-232 RECEIVERS					
RS-232 Input Voltage Operating Range				± 25	V
RS-232 Input Threshold Low	$V_{CC} = 5V$	0.8	1.3		V
RS-232 Input Threshold High	$V_{CC} = 5V$		1.8	2.4	V
RS-232 Input Hysteresis	$V_{CC} = 5V$	0.2	0.5	1.0	V
RS-232 Input Resistance		3	5	7	$k\Omega$
TTL/CMOS Output Voltage Low	$I_{OUT} = 3.2mA$		0.2	0.4	V
TTL/CMOS Output Voltage High	$I_{OUT} = -1.0mA$	3.5	$V_{CC} - 0.2$		V
TTL/CMOS Output Short-Circuit Current	Sourcing $V_{OUT} = GND$	-2	-10		mA
	Sinking $V_{OUT} = V_{CC}$	10	30		mA
TTL/CMOS Output Leakage Current	Normal operation, outputs disabled, Tables 1a-1d, $0V \leq V_{OUT} \leq V_{CC}$, $EN_R = V_{CC}$		± 0.05	± 0.10	μA

MAXIM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

ELECTRICAL CHARACTERISTICS—MAX225/MAX244-MAX249 (continued)

(MAX225, $V_{CC} = 5.0V \pm 5\%$; MAX244-MAX249, $V_{CC} = +5.0V \pm 10\%$, external capacitors C1-C4 = $1\mu F$; $T_A = T_{MIN}$ to T_{MAX} ; unless otherwise noted.)

PARAMETER		CONDITIONS	MIN	TYP	MAX	UNITS
POWER SUPPLY AND CONTROL LOGIC						
Operating Supply Voltage		MAX225	4.75		5.25	V
		MAX244–MAX249	4.5		5.5	
V _{CC} Supply Current (normal operation)	No load	MAX225		10	20	mA
		MAX244–MAX249		11	30	
	3kΩ loads on all outputs	MAX225		40		
		MAX244–MAX249		57		
Shutdown Supply Current	T _A = +25°C			8	25	μA
	T _A = T _{MIN} to T _{MAX}				50	
Control Input	Leakage current				±1	μA
	Threshold low			1.4	0.8	V
	Threshold high		2.4	1.4		
AC CHARACTERISTICS						
Transition Slew Rate	C _L = 50pF to 2500pF, R _L = 3kΩ to 7kΩ, V _{CC} = 5V, T _A = +25°C, measured from +3V to -3V or -3V to +3V		5	10	30	V/μs
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t _{PHLT}			1.3	3.5	μs
	t _{PLHT}			1.5	3.5	
Receiver Propagation Delay TLL to RS-232 (normal operation), Figure 2	t _{PHLR}			0.6	1.5	μs
	t _{PLHR}			0.6	1.5	
Receiver Propagation Delay TLL to RS-232 (low-power mode), Figure 2	t _{PHLS}			0.6	10	μs
	t _{PLHS}			3.0	10	
Transmitter + to - Propagation Delay Difference (normal operation)	t _{PHLT} - t _{PLHT}			350		ns
Receiver + to - Propagation Delay Difference (normal operation)	t _{PHLR} - t _{PLHR}			350		ns
Receiver-Output Enable Time, Figure 3	t _{ER}			100	500	ns
Receiver-Output Disable Time, Figure 3	t _{DR}			100	500	ns
Transmitter Enable Time	t _{ET}	MAX246–MAX249 (excludes charge-pump start-up)		5		μs
		MAX225/MAX245–MAX249 (includes charge-pump start-up)		10		ms
Transmitter Disable Time, Figure 4	t _{DT}			100		ns

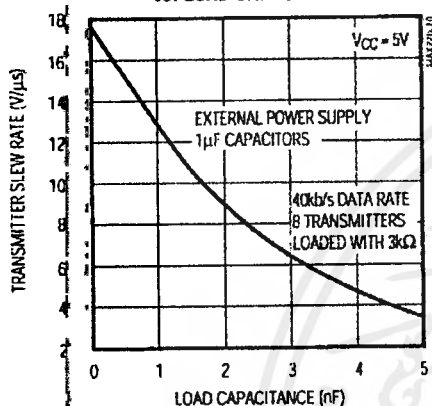
Note 4: The 300 Ω minimum specification complies with EIA/TIA-232E, but the actual resistance when in shutdown mode or $V_{CC} = 0V$ is 10M Ω as is implied by the leakage specification.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

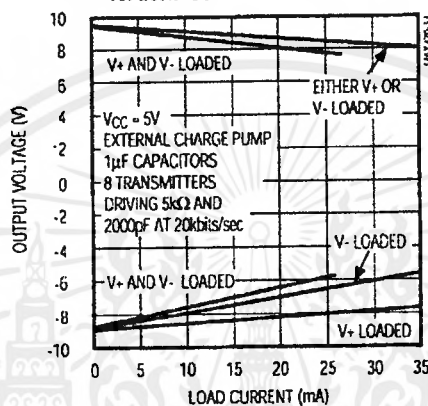
Typical Operating Characteristics

MAX225/MAX244-MAX249

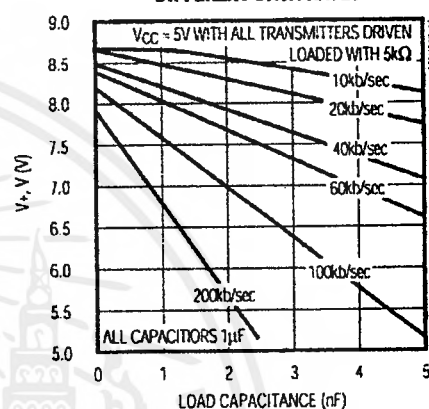
TRANSMITTER SLEW RATE
vs. LOAD CAPACITANCE



OUTPUT VOLTAGE
vs. LOAD CURRENT FOR V+ AND V-



TRANSMITTER OUTPUT VOLTAGE (V+, V-)
vs. LOAD CAPACITANCE AT
DIFFERENT DATA RATES



+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

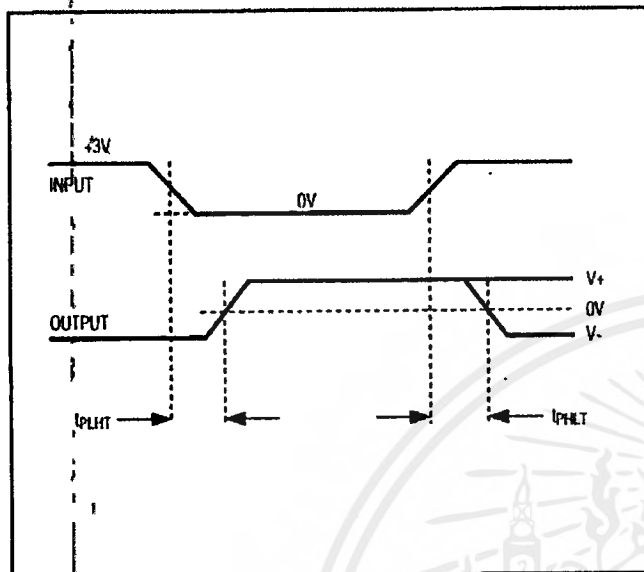


Figure 1. Transmitter Propagation-Delay Timing

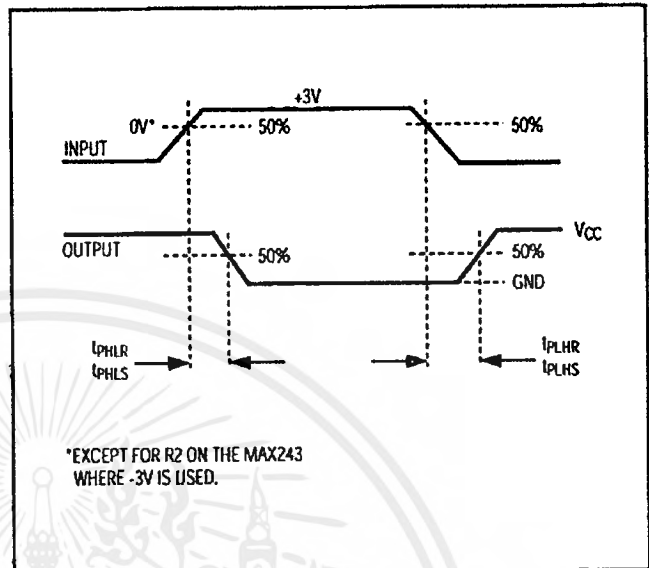


Figure 2. Receiver Propagation-Delay Timing

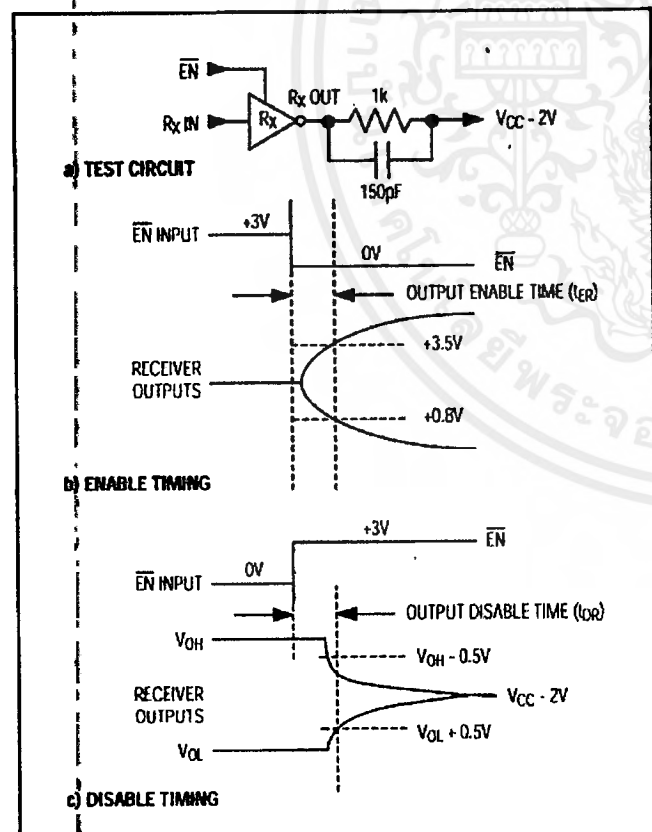


Figure 3. Receiver-Output Enable and Disable Timing

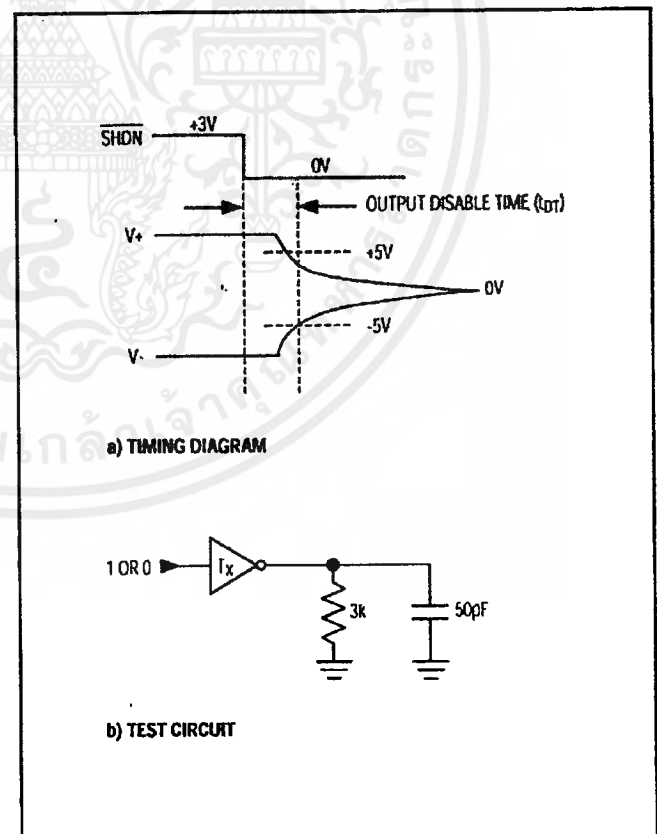


Figure 4. Transmitter-Output Disable Timing

MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Table 1a. MAX245 Control Pin Configurations

ENT	ENR	OPERATION STATUS	TRANSMITTERS	RECEIVERS
0	0	Normal Operation	All Active	All Active
0	1	Normal Operation	All Active	All 3-State
1	0	Shutdown	All 3-State	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State

Table 1b. MAX245 Control Pin Configurations

ENT	ENR	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All Active	RA1-RA4 3-State, RA5 Active	RB1-RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All 3-State	All Low-Power Receive Mode	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State, RA5 Low-Power Receive Mode	RB1-RB4 3-State, RB5 Low-Power Receive Mode

Table 1c. MAX246 Control Pin Configurations

ENA	ENB	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1-TA4	TB1-TB4	RA1-RA5	RB1-RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All 3-State	All Active	RB1-RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All Active	RA1-RA4 3-State, RA5 Active	All Active
1	1	Shutdown	All 3-State	All 3-State	RA1-RA4 3-State, RA5 Low-Power Receive Mode	RB1-RB4 3-State, RA5 Low-Power Receive Mode

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

Table 1d. MAX247/MAX248/MAX249 Control Pin Configurations

ENTA	ENTB	ENRA	ENRB	OPERATION STATUS	TRANSMITTERS			RECEIVERS	
					MAX247	TA1-TA4	TB1-TB4	RA1-RA4	RB1-RB5
					MAX248	TA1-TA4	TB1-TB4	RA1-RA4	RB1-RB4
					MAX249	TA1-TA3	TB1-TB3	RA1-RA5	RB1-RB5
0	0	0	0	Normal Operation		All Active	All Active	All Active	All Active
0	0	0	1	Normal Operation		All Active	All Active	All Active	All 3-State, except RB5 stays active on MAX247
0	0	1	0	Normal Operation		All Active	All Active	All 3-State	All Active
0	0	1	1	Normal Operation		All Active	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
0	1	0	0	Normal Operation		All Active	All 3-State	All Active	All Active
0	1	0	1	Normal Operation		All Active	All 3-State	All Active	All 3-State, except RB5 stays active on MAX247
0	1	1	0	Normal Operation		All Active	All 3-State	All 3-State	All Active
0	1	1	1	Normal Operation		All Active	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247
1	0	0	0	Normal Operation		All 3-State	All Active	All Active	All Active
1	0	0	1	Normal Operation		All 3-State	All Active	All Active	All 3-State, except RB5 stays active on MAX247
1	0	1	0	Normal Operation		All 3-State	All Active	All 3-State	All Active
1	0	1	1	Normal Operation		All 3-State	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
1	1	0	0	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	Low-Power Receive Mode
1	1	0	1	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	All 3-State, except RB5 stays active on MAX247
1	1	1	0	Shutdown		All 3-State	All 3-State	All 3-State	Low-Power Receive Mode
1	1	1	1	Shutdown		All 3-State	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Detailed Description

The MAX220-MAX249 contain four sections: dual charge-pump DC-DC voltage converters, RS-232 drivers, RS-232 receivers, and receiver and transmitter enable control inputs.

Dual Charge-Pump Voltage Converter

The MAX220-MAX249 have two internal charge-pumps that convert +5V to $\pm 10V$ (unloaded) for RS-232 driver operation. The first converter uses capacitor C1 to double the +5V input to +10V on C3 at the V+ output. The second converter uses capacitor C2 to invert +10V to -10V on C4 at the V- output.

A small amount of power may be drawn from the +10V (V+) and -10V (V-) outputs to power external circuitry (see the *Typical Operating Characteristics* section), except on the MAX225 and MAX245-MAX247, where these pins are not available. V+ and V- are not regulated, so the output voltage drops with increasing load current. Do not load V+ and V- to a point that violates the minimum $\pm 5V$ EIA/TIA-232E driver output voltage when sourcing current from V+ and V- to external circuitry.

When using the shutdown feature in the MAX222, MAX225, MAX230, MAX235, MAX236, MAX240, MAX241, and MAX245-MAX249, avoid using V+ and V- to power external circuitry. When these parts are shut down, V- falls to 0V, and V+ falls to +5V. For applications where a +10V external supply is applied to the V+ pin (instead of using the internal charge pump to generate +10V), the C1 capacitor must not be installed and the SHDN pin must be tied to VCC. This is because V+ is internally connected to VCC in shutdown mode.

RS-232 Drivers

The typical driver output voltage swing is $\pm 8V$ when loaded with a nominal 5k Ω RS-232 receiver and VCC = +5V. Output swing is guaranteed to meet the EIA/TIA-232E and V.28 specification, which calls for $\pm 5V$ minimum driver output levels under worst-case conditions. These include a minimum 3k Ω load, VCC = +4.5V, and maximum operating temperature. Unloaded driver output voltage ranges from (V+ -1.3V) to (V- +0.5V).

Input thresholds are both TTL and CMOS compatible. The inputs of unused drivers can be left unconnected since 400k Ω input pull-up resistors to VCC are built in. The pull-up resistors force the outputs of unused drivers low because all drivers invert. The internal input pull-up resistors typically source 12 μA , except in shutdown mode where the pull-ups are disabled. Driver outputs turn off and enter a high-impedance state—where leakage current is typically microamperes (maximum 25 μA)—when in shutdown mode, in three-state mode, or

when device power is removed. Outputs can be driven to $\pm 15V$. The power-supply current typically drops to 8 μA in shutdown mode.

The MAX239 has a receiver three-state control line, and the MAX223, MAX225, MAX235, MAX236, MAX240, and MAX241 have both a receiver three-state control line and a low-power shutdown control. Table 2 shows the effects of the shutdown control and receiver three-state control on the receiver outputs.

The receiver TTL/CMOS outputs are in a high-impedance, three-state mode whenever the three-state enable line is high (for the MAX225/MAX235/MAX236/MAX239-MAX241), and are also high-impedance whenever the shutdown control line is high.

When in low-power shutdown mode, the driver outputs are turned off and their leakage current is less than 1 μA with the driver output pulled to ground. The driver output leakage remains less than 1 μA , even if the transmitter output is backdriven between 0V and (VCC + 6V). Below -0.5V, the transmitter is diode clamped to ground with 1k Ω series impedance. The transmitter is also zener clamped to approximately VCC + 6V, with a series impedance of 1k Ω .

The driver output slew rate is limited to less than 30V/ μs as required by the EIA/TIA-232E and V.28 specifications. Typical slew rates are 24V/ μs unloaded and 10V/ μs loaded with 3 Ω and 2500pF.

RS-232 Receivers

EIA/TIA-232E and V.28 specifications define a voltage level greater than 3V as a logic 0, so all receivers invert. Input thresholds are set at 0.8V and 2.4V, so receivers respond to TTL level inputs as well as EIA/TIA-232E and V.28 levels.

The receiver inputs withstand an input overvoltage up to $\pm 25V$ and provide input terminating resistors with nominal 5k Ω values. The receivers implement Type 1 interpretation of the fault conditions of V.28 and EIA/TIA-232E.

Table 2. Three-State Control of Receivers

PART	SHDN	SHDN	EN	EN(R)	RECEIVERS
MAX223	—	Low High High	X Low High	—	High Impedance Active High Impedance
MAX225	—	—	—	Low High	High Impedance Active
MAX235 MAX236 MAX240	Low Low High	—	—	Low High X	High Impedance Active High Impedance

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249

The receiver input hysteresis is typically 0.5V with a guaranteed minimum of 0.2V. This produces clear output transitions with slow-moving input signals, even with moderate amounts of noise and ringing. The receiver propagation delay is typically 600ns and is independent of input swing direction.

Low-Power Receive Mode

The low-power receive-mode feature of the MAX223, MAX242, and MAX245-MAX249 puts the IC into shutdown mode but still allows it to receive information. This is important for applications where systems are periodically awakened to look for activity. Using low-power receive mode, the system can still receive a signal that will activate it on command and prepare it for communication at faster data rates. This operation conserves system power.

Negative Threshold—MAX243

The MAX243 is pin compatible with the MAX232A, differing only in that RS-232 cable fault protection is removed on one of the two receiver inputs. This means that control lines such as CTS and RTS can either be driven or left floating without interrupting communication. Different cables are not needed to interface with different pieces of equipment.

The input threshold of the receiver without cable fault protection is -0.8V rather than +1.4V. Its output goes positive only if the input is connected to a control line that is actively driven negative. If not driven, it defaults to the 0 or "OK to send" state. Normally, the MAX243's other receiver (+1.4V threshold) is used for the data line (TD or RD), while the negative threshold receiver is connected to the control line (DTR, DTS, CTS, RTS, etc.).

Other members of the RS-232 family implement the optional cable fault protection as specified by EIA/TIA-232E specifications. This means a receiver output goes high whenever its input is driven negative, left floating, or shorted to ground. The high output tells the serial communications IC to stop sending data. To avoid this, the control lines must either be driven or connected with jumpers to an appropriate positive voltage level.

Shutdown—MAX222-MAX242

On the MAX222, MAX235, MAX236, MAX240, and MAX241, all receivers are disabled during shutdown. On the MAX223 and MAX242, two receivers continue to operate in a reduced power mode when the chip is in shutdown. Under these conditions, the propagation delay increases to about 2.5 μ s for a high-to-low input transition. When in shutdown, the receiver acts as a CMOS inverter with no hysteresis. The MAX223 and MAX242 also have a receiver output enable input (EN for the MAX242 and EN for the MAX223) that allows receiver output control independent of $\overline{\text{SHDN}}$ ($\overline{\text{SHDN}}$ for MAX241). With all other devices, $\overline{\text{SHDN}}$ ($\overline{\text{SHDN}}$ for MAX241) also disables the receiver outputs.

The MAX225 provides five transmitters and five receivers, while the MAX245 provides ten receivers and eight transmitters. Both devices have separate receiver and transmitter-enable controls. The charge pumps turn off and the devices shut down when a logic high is applied to the ENT input. In this state, the supply current drops to less than 25 μ A and the receivers continue to operate in a low-power receive mode. Driver outputs enter a high-impedance state (three-state mode). On the MAX225, all five receivers are controlled by the $\overline{\text{ENR}}$ input. On the MAX245, eight of the receiver outputs are controlled by the $\overline{\text{ENR}}$ input, while the remaining two receivers (RA5 and RB5) are always active. RA1-RA4 and RB1-RB4 are put in a three-state mode when $\overline{\text{ENR}}$ is a logic high.

Receiver and Transmitter Enable Control Inputs

The MAX225 and MAX245-MAX249 feature transmitter and receiver enable controls.

The receivers have three modes of operation: full-speed receive (normal active), three-state (disabled), and low-power receive (enabled receivers continue to function at lower data rates). The receiver enable inputs control the full-speed receive and three-state modes. The transmitters have two modes of operation: full-speed transmit (normal active) and three-state (disabled). The transmitter enable inputs also control the shutdown mode. The device enters shutdown mode when all transmitters are disabled. Enabled receivers function in the low-power receive mode when in shutdown.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Tables 1a-1d define the control states. The MAX244 has no control pins and is not included in these tables.

The MAX246 has ten receivers and eight drivers with two control pins, each controlling one side of the device. A logic high at the A-side control input ($\overline{\text{ENA}}$) causes the four A-side receivers and drivers to go into a three-state mode. Similarly, the B-side control input ($\overline{\text{ENB}}$) causes the four B-side drivers and receivers to go into a three-state mode. As in the MAX245, one A-side and one B-side receiver (RA5 and RB5) remain active at all times. The entire device is put into shutdown mode when both the A and B sides are disabled ($\overline{\text{ENA}} = \overline{\text{ENB}} = +5\text{V}$).

The MAX247 provides nine receivers and eight drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control four receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs each control four drivers. The ninth receiver (RB5) is always active. The device enters shutdown mode with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$.

The MAX248 provides eight receivers and eight drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control four receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs control four drivers each. This part does not have an always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$.

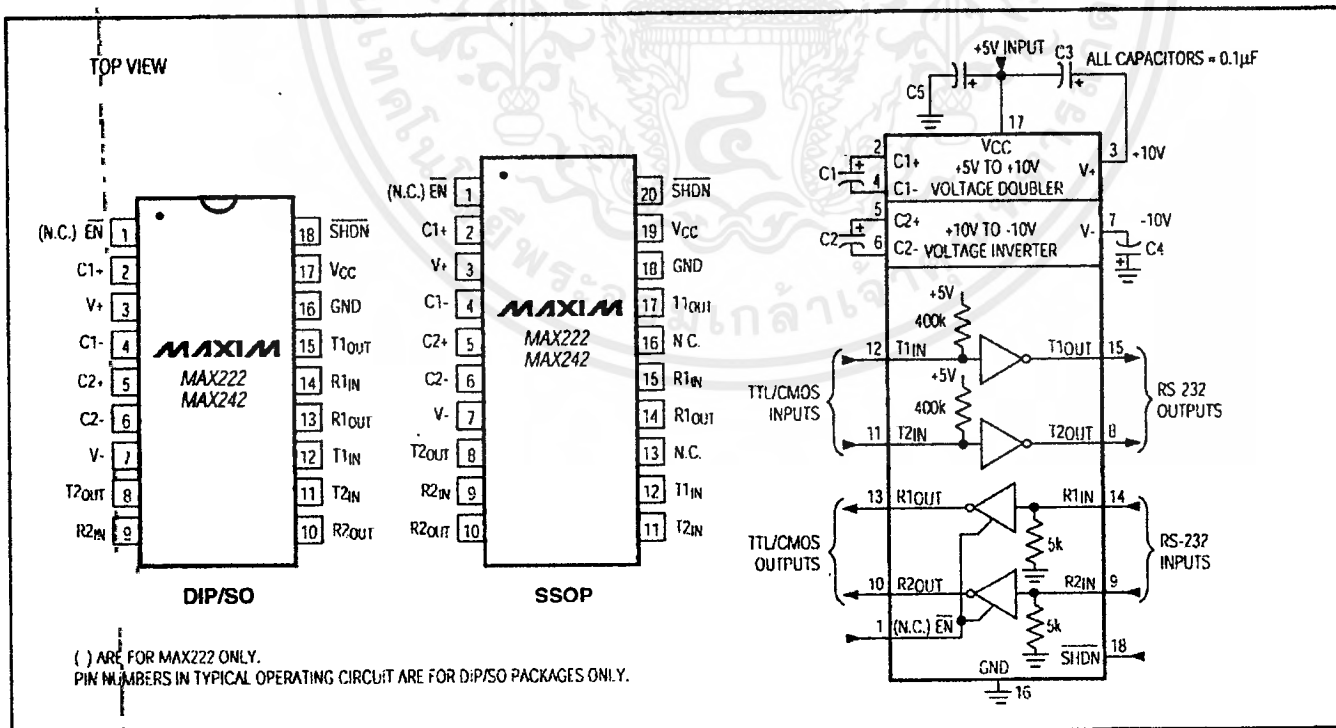
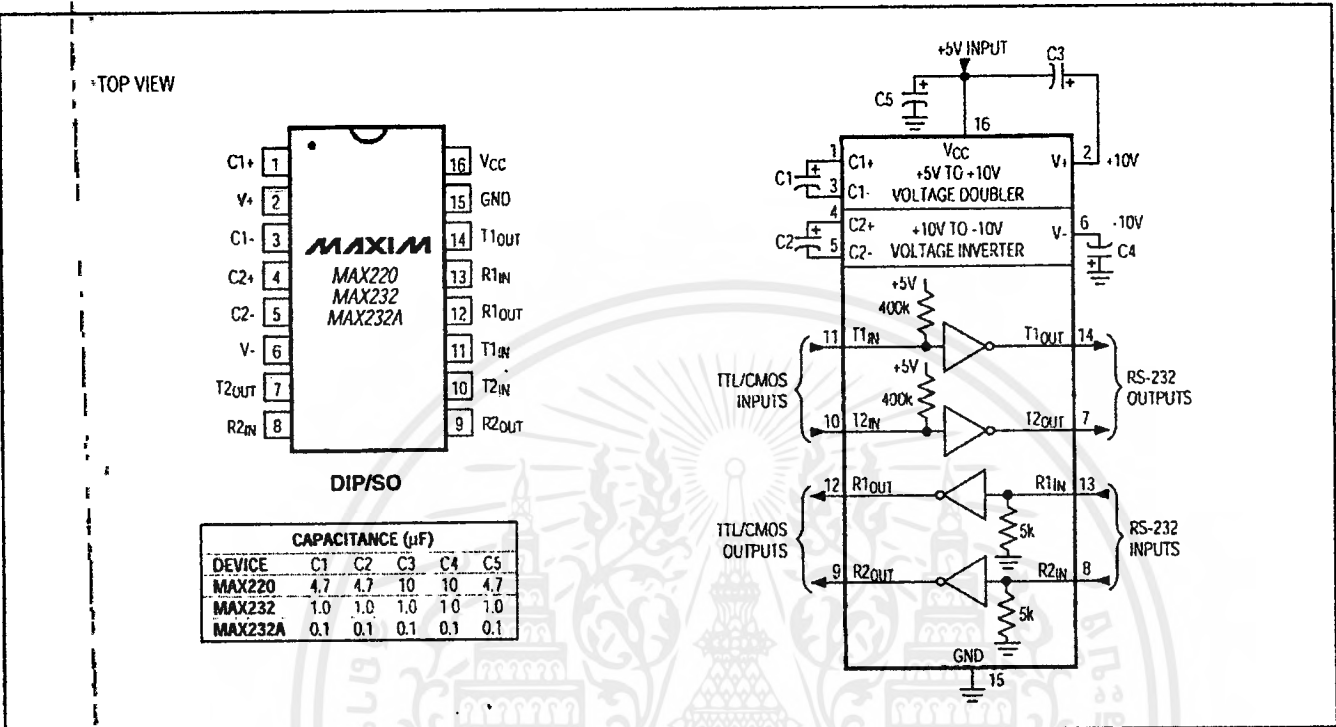
The MAX249 provides ten receivers and six drivers with four control pins. The $\overline{\text{ENRA}}$ and $\overline{\text{ENRB}}$ receiver enable inputs each control five receiver outputs. The $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$ transmitter enable inputs control three drivers each. There is no always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode, with a logic high on both $\overline{\text{ENTA}}$ and $\overline{\text{ENTB}}$. In shutdown mode, active receivers operate in a low-power receive mode at data rates up to 20kbits/sec.

Applications Information

Figures 5 through 25 show pin configurations and typical operating circuits. In applications that are sensitive to power-supply noise, VCC should be decoupled to ground with a capacitor of the same value as C1 and C2 connected as close as possible to the device.

+5V-Powered, Multichannel RS-232 Drivers/Receivers

MAX220-MAX249





National Semiconductor

LMD18201 3A, 55V H-Bridge

General Description

The LMD18201 is a 3A H-Bridge designed for motion control applications. The device is built using a multi-technology process which combines bipolar and CMOS control circuitry with DMOS power devices on the same monolithic structure. The H-Bridge configuration is ideal for driving DC and stepper motors. The LMD18201 accommodates peak output currents up to 6A. Current sensing can be achieved via a small sense resistor connected in series with the power ground lead. For current sensing without disturbing the path of current to the load, the LMD18201 is recommended.

Features

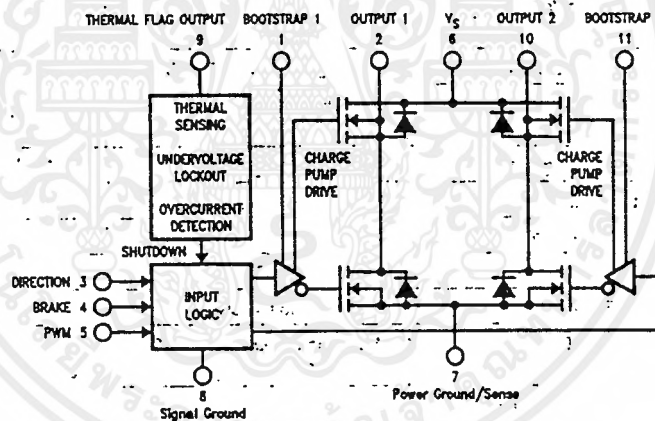
- Delivers up to 3A continuous output
- Operates at supply voltages up to 55V
- Low $R_{DS(ON)}$ typically 0.33 Ω per switch

- TTL and CMOS compatible inputs
- No "shoot-through" current
- Thermal warning flag output at 145°C
- Thermal shutdown (outputs off) at 170°C
- Internal clamp diodes
- Shorted load protection
- Internal charge pump with external bootstrap capability

Applications

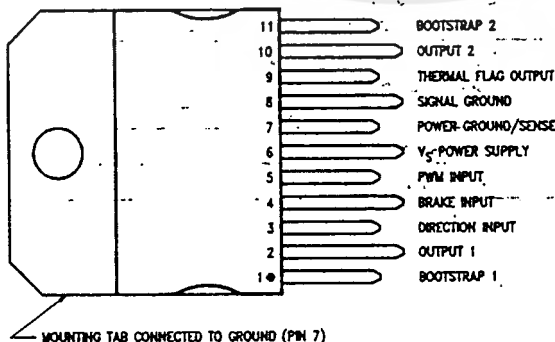
- DC and stepper motor drives
- Position and velocity servomechanisms
- Factory automation robots
- Numerically controlled machinery
- Computer printers and plotters

Functional Diagram



TL/H/10793-1

Connection Diagram and Ordering Information



Top View

TL/H/10793-2

Order Number LMD18201T
See NS Package Number TA11B

LMD18201

4

Absolute Maximum Ratings (Note 1)

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Total Supply Voltage (V_S , Pin 6)	60V
Voltage at Pins 3, 4, 5 and 9	12V
Voltage at Bootstrap Pins (Pins 1 and 11)	$V_{OUT} + 16V$
Peak Output Current (200 ms)	6A
Continuous Output Current (Note 2)	3A
Power Dissipation (Note 3)	25W
Sense Voltage (Pin 7 to Pin 8)	+0.5V to +1.0V

Power Dissipation ($T_A = 25^\circ\text{C}$, Free Air)	3W
Junction Temperature, $T_{J(\text{max})}$	150°C
ESD Susceptibility (Note 4)	1500V
Storage Temperature, T_{STG}	-65°C to +150°C
Lead Temperature (Soldering, 10 sec.)	300°C

Operating Ratings (Note 1)

Junction Temperature, T_J	-40°C to +125°C
V_S Supply Voltage	+12V to +55V

Electrical Characteristics

The following specifications apply for $V_S = 42V$, unless otherwise specified. Boldface limits apply over the entire operating temperature range, $-40^\circ\text{C} \leq T_J \leq +125^\circ\text{C}$, all other limits are for $T_A = T_J = 25^\circ\text{C}$. (Note 5)

Symbol	Parameter	Conditions	Typ	Limit	Units
$R_{OS(ON)}$	Switch ON Resistance	Output Current = 3A (Note 6)	0.33	0.4 / 0.6	Ω (max)
$R_{OS(ON)}$	Switch ON Resistance	Output Current = 6A (Note 6)	0.33	0.4 / 0.6	Ω (max)
V_{CLAMP}	Clamp Diode Forward Drop	Clamp Current = 3A (Note 6)	1.2	1.5	V (max)
V_{IL}	Logic Low Input Voltage	Pins 3, 4, 5		-0.1 / 0.8	V (min) V (max)
I_{IL}	Logic Low Input Current	$V_{IN} = -0.1V$, Pins = 3, 4, 5		-10	μA (max)
V_{IH}	Logic High Input Voltage	Pins 3, 4, 5		2 / 12	V (min) V (max)
I_{IH}	Logic High Input Current	$V_{IN} = 12V$, Pins = 3, 4, 5		10	μA (max)
	Undervoltage Lockout	Outputs Turn OFF		9 / 11	V (min) V (max)
T_{JW}	Warning Flag Temperature	Pin 9 $\leq 0.8V$, $I_L = 2\text{mA}$	145		$^\circ\text{C}$
$V_{F(ON)}$	Flag Output Saturation Voltage	$T_J = T_{JW}$, $I_L = 2\text{mA}$	0.15		V
$I_{F(OFF)}$	Flag Output Leakage	$V_F = 12V$	0.2	10	μA (max)
T_{JSD}	Shutdown Temperature	Outputs Turn OFF	170		$^\circ\text{C}$
I_S	Quiescent Supply Current	All Logic Inputs Low	13	25	mA (max)
$t_{D(ON)}$	Output Turn-On Delay Time	Sourcing Outputs, $I_{OUT} = 3A$ Sinking Outputs, $I_{OUT} = 3A$	300 / 300		ns ns
t_{ON}	Output Turn-On Switching Time	Bootstrap Capacitor = 10 nF Sourcing Outputs, $I_{OUT} = 3A$ Sinking Outputs, $I_{OUT} = 3A$	100 / 80		ns ns
$t_{D(OFF)}$	Output Turn-Off Delay Times	Sourcing Outputs, $I_{OUT} = 3A$ Sinking Outputs, $I_{OUT} = 3A$	200 / 200		ns ns
t_{OFF}	Output Turn-Off Switching Times	Bootstrap Capacitor = 10 nF Sourcing Outputs, $I_{OUT} = 3A$ Sinking Outputs, $I_{OUT} = 3A$	75 / 70		ns ns
t_{PW}	Minimum Input Pulse Width	Pins 3, 4 and 5	1		μs
t_{CPH}	Charge Pump Rise Time	No Bootstrap Capacitor	20		μs

Electrical Characteristics (Continued)

Note 1: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions.

Note 2: See Application Information for details regarding current limiting.

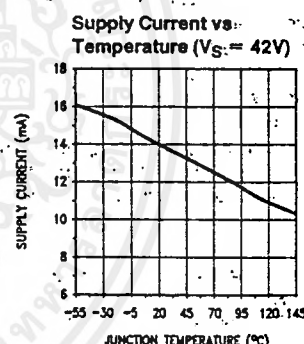
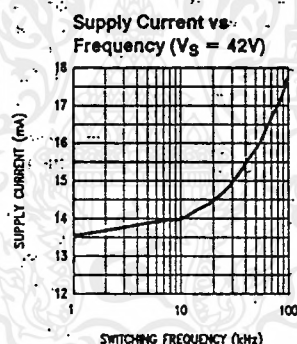
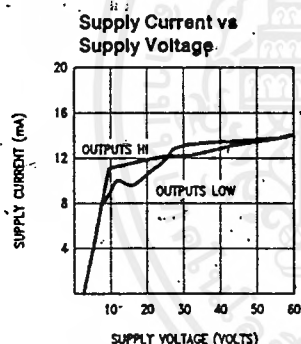
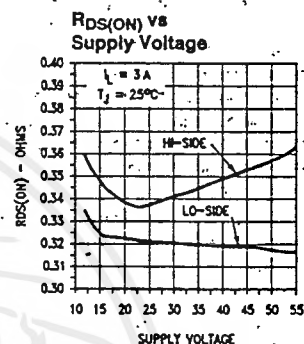
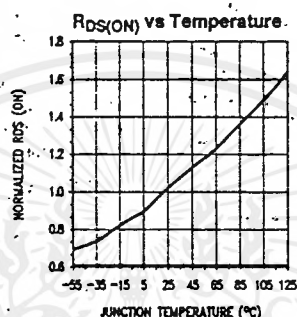
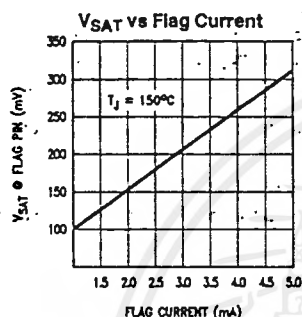
Note 3: The maximum power dissipation must be derated at elevated temperatures and is a function of $T_{J(max)}$, θ_{JA} , and T_A . The maximum allowable power dissipation at any temperature is $P_{D(max)} = (T_{J(max)} - T_A)/\theta_{JA}$, or the number given in the Absolute Ratings, whichever is lower. The typical thermal resistance from junction to case (θ_{JC}) is 1.0°C/W and from junction to ambient (θ_{JA}) is 30°C/W . For guaranteed operation $T_{J(max)} = 125^\circ\text{C}$.

Note 4: Human-body model, 100 pF discharged through a 1.5 k Ω resistor. Except Bootstrap pins (pins 1 and 11) which are protected to 1000V of ESD.

Note 5: All limits are 100% production tested at 25°C . Temperature extreme limits are guaranteed via correlation using accepted SQC (Statistical Quality Control) methods. All limits are used to calculate AOQL (Average Outgoing Quality Level).

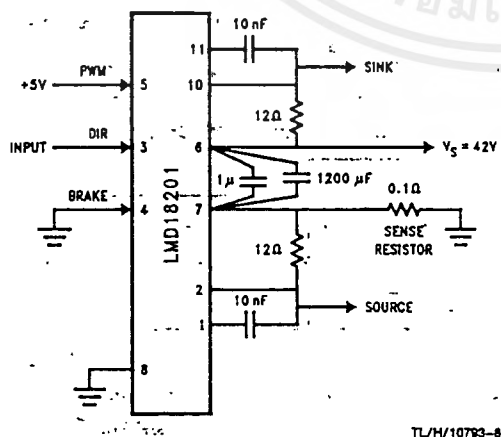
Note 6: Output currents are pulsed ($t_W < 2$ ms, Duty Cycle $< 5\%$).

Typical Performance Characteristics



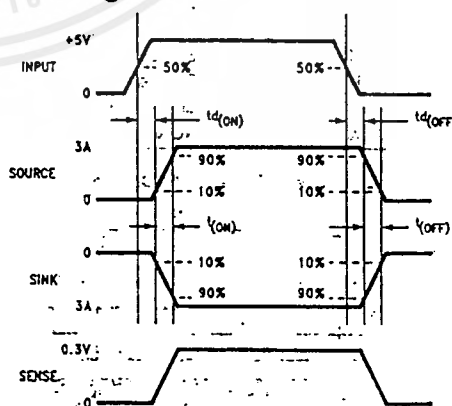
TL/H/10793-3

Test Circuit



TL/H/10793-8

Switching Time Definitions



TL/H/10793-9

Pinout Description (See Connection Diagram)

Pin 1, BOOTSTRAP 1 Input: Bootstrap capacitor pin for half H-Bridge number 1. The recommended capacitor (10 nF) is connected between pins 1 and 2.

Pin 2, OUTPUT 1: Half H-Bridge number 1 output.

Pin 3, DIRECTION Input: See Table I. This input controls the direction of current flow between OUTPUT 1 and OUTPUT 2 (pins 2 and 10) and, therefore, the direction of rotation of a motor load.

Pin 4, BRAKE Input: See Table I. This input is used to brake a motor by effectively shorting its terminals. When braking is desired, this input is taken to a logic high level and it is also necessary to apply logic high to PWM input, pin 5. The drivers that short the motor are determined by the logic level at the DIRECTION input (Pin 3): with Pin 3 logic high, both current sourcing output transistors are ON; with Pin 3 logic low, both current sinking output transistors are ON. All output transistors can be turned OFF by applying a logic high to Pin 4 and a logic low to PWM input Pin 5; in this case only a small bias current (approximately -1.5 mA) exists at each output pin.

Pin 5, PWM Input: See Table I. How this input (and DIRECTION input, Pin 3) is used is determined by the format of the PWM Signal.

Pin 6, V_S Power Supply

Pin 7, POWER GROUND/SENSE Connection: This pin is the ground return for the power DMOS transistors of the H-Bridge. The current through the H-Bridge can be sensed by adding a small, 0.1 Ω , sense resistor from this pin to the power supply ground.

Pin 8, SIGNAL GROUND: This is the ground return for the internal logic circuitry used to control the PWM switching of the H-Bridge.

Pin 9, THERMAL FLAG Output: This pin provides the thermal warning flag output signal. Pin 9 becomes active-low at 145°C (junction temperature). However the chip will not shut itself down until 170°C is reached at the junction.

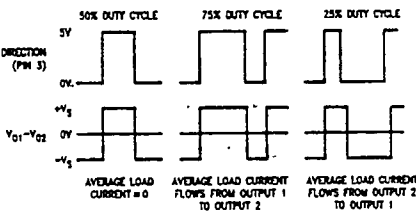
Pin 10, OUTPUT 2: Half H-Bridge number 2 output.

Pin 11, BOOTSTRAP 2 Input: Bootstrap capacitor pin for half H-Bridge number 2. The recommended capacitor (10 nF) is connected between pins 10 and 11.

TABLE I. Logic Truth Table

PWM	Dir	Brake	Active Output Drivers
H	H	L	Source 1, Sink 2
H	L	L	Sink 1, Source 2
L	X	L	Source 1, Source 2
H	H	H	Source 1, Source 2
H	L	H	Sink 1, Sink 2
L	X	H	NONE

Locked Anti-Phase PWM Control



TL/H/10793-4

Application Information

TYPES OF PWM SIGNALS.

The LMD18201 readily interfaces with different forms of PWM signals. Use of the part with two of the more popular forms of PWM is described in the following paragraphs:

Simple, locked anti-phase PWM consists of a single, variable duty-cycle signal in which is encoded both direction and amplitude information. A 50% duty-cycle PWM signal represents zero drive, since the net value of voltage (integrated over one period) delivered to the load is zero. For the LMD18201, the PWM signal drives the direction input (pin 3) and the PWM input (pin 5) is tied to logic high.

Sign/magnitude PWM consists of separate direction (sign) and amplitude (magnitude) signals. The (absolute) magnitude signal is duty-cycle modulated, and the absence of a pulse signal (a continuous logic low level) represents zero drive. Current delivered to the load is proportional to pulse width. For the LMD18201, the DIRECTION input (pin 3) is driven by the sign signal and the PWM input (pin 5) is driven by the magnitude signal.

USING THE THERMAL WARNING FLAG.

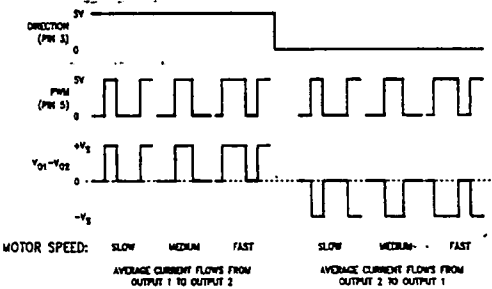
The THERMAL FLAG output (pin 9) is an open collector transistor. This permits a wired OR connection of thermal warning flag outputs from multiple LMD18201's, and allows the user to set the logic high level of the output signal swing to match system requirements. This output typically drives the interrupt input of a system controller. The interrupt service routine would then be designed to take appropriate steps, such as reducing load currents or initiating an orderly system shutdown. The maximum voltage compliance on the flag pin is 12V.

SUPPLY BYPASSING

During switching transitions the levels of fast current changes experienced may cause troublesome voltage transients across system stray inductances.

It is normally necessary to bypass the supply rail with a high quality capacitor(s) connected as close as possible to the V_S Power Supply (Pin 6) and POWER GROUND (Pin 7). A 1 μ F high-frequency ceramic capacitor is recommended. Care should be taken to limit the transients on the supply pin below the Absolute Maximum Rating of the device. When operating the chip at supply voltages above 40V a voltage suppressor (transorb) such as P6KE62A is recommended from supply to ground. Typically the ceramic capacitor can be eliminated in the presence of the voltage suppressor. Note that when driving high load currents a greater amount of supply bypass capacitance (in general at least 100 μ F per Amp of load current) is required to absorb the recirculating currents of the inductive loads.

Sign/Magnitude PWM Control



TL/H/10793-5

Application Information (Continued)

CURRENT LIMITING

Current limiting protection circuitry has been incorporated into the design of the LMD18201. With any power device it is important to consider the effects of the substantial surge currents through the device that may occur as a result of shorted loads. The protection circuitry monitors the current through the upper transistors and shuts off the power device as quickly as possible in the event of an overload condition (the threshold is set to approximately 10A). In a typical motor driving application the most common overload faults are caused by shorted motor windings and locked rotors. Under these conditions the inductance of the motor (as well as any series inductance in the V_{CC} supply line) serves to reduce the magnitude of a current surge to a safe level for the LMD18201. Once the device is shut down, the control circuitry will periodically try to turn the power device back on. This feature allows the immediate return to normal operation once the fault condition has been removed. While the fault remains however, the device will cycle in and out of thermal shutdown. This can create voltage transients on the V_{CC} supply line and therefore proper supply bypassing techniques are required.

The most severe condition for any power device is a direct, hard-wired ("screwdriver") long term short from an output to ground. This condition can generate a surge of current through the power device on the order of 15 Amps and require the die and package to dissipate up to 500W. of power for the short time required for the protection circuitry to shut off the power device. This energy can be destructive, particularly at higher operating voltages (>30V) so some precautions are in order. Proper heat sink design is essential and it is normally necessary to heat sink the V_{CC} supply pin (pin 6) with 1 square inch of copper on the PC board.

INTERNAL CHARGE PUMP AND USE OF EXTERNAL BOOTSTRAP CAPACITORS

To turn on the high-side (sourcing) DMOS power devices, the gate of each device must be driven approximately 8V more positive than the supply voltage. To achieve this, an internal charge pump is used to provide the gate drive voltage. As shown in Figure 1, an internal capacitor is alternately switched to ground and charged to about 14V, then switched to V_S thereby providing a gate drive voltage greater than V_S . This switching action is controlled by a continuously running internal 300 kHz oscillator. The rise time of this drive voltage is typically 20 μ s which is suitable for operating frequencies up to 1 kHz.

For higher switching frequencies, the LMD18201 provides for the use of external bootstrap capacitors. The bootstrap principle is, in essence a second charge pump whereby a large value capacitor is used which has enough energy to quickly charge the parasitic gate input capacitance of the power device resulting in much faster rise times. The switching action is accomplished by the power switches themselves (Figure 2). External 10 nF capacitors, connected from the outputs to the bootstrap pins of each high-side switch provide typically less than 100 ns rise times allowing switching frequencies up to 500 kHz.

INTERNAL PROTECTION DIODES

A major consideration when switching current through inductive loads is protection of the switching power devices from the large voltage transients that occur. Each of the four switches in the LMD18201 have a built-in protection diode to clamp transient voltages exceeding the positive supply or ground to a safe diode voltage drop across the switch.

The reverse recovery characteristics of these diodes, once the transient has subsided, is important. These diodes must come out of conduction quickly and the power switches must be able to conduct the additional reverse recovery current of the diodes. The reverse recovery time of the diodes protecting the sourcing power devices is typically only 70 ns with a reverse recovery current of 1A when tested with a full 3A of forward current through the diode. For the sinking devices the recovery time is typically 100 ns with 4A of reverse current under the same conditions.

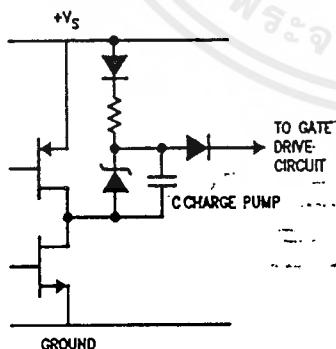


FIGURE 1. Internal Charge Pump Circuitry

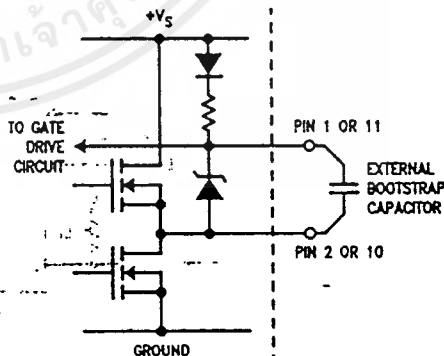


FIGURE 2. Bootstrap Circuitry

Typical Applications

BASIC MOTOR DRIVER

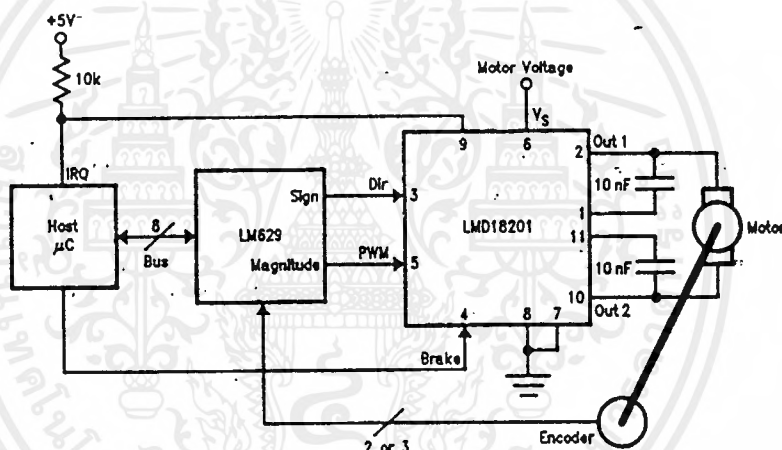
The LMD18201 can directly interface to any Sign/Magnitude PWM controller. The LM629 is a motion control processor that outputs a Sign/Magnitude PWM signal to coordinate either positional or velocity control of DC motors. The LMD18201 provides fully protected motor driver stage.

CURRENT SENSING

In many motor control applications it is desirable to sense and control the current through the motor. For these types of applications a companion product, the LMD18200, is also available. The LMD18200 is identical to the LMD18201 but has current sensing transistors that output a current directly proportional to the current conducted by the two upper DMOS power devices to a separate current sense pin. This technique does not require a low valued, power sense resistor and does not subtract from the available voltage drive to the motor.

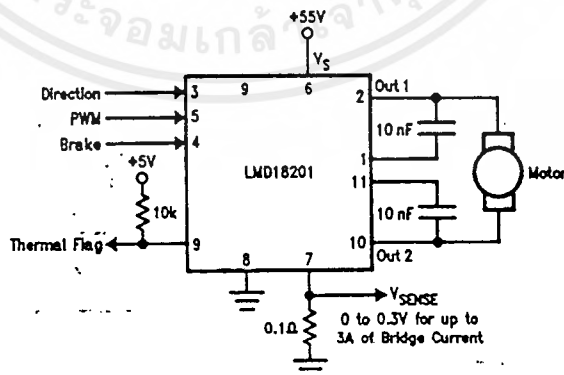
To sense the bridge current through the LMD18201 requires the addition of a small sense resistor between the power ground/sense pin (Pin 7) and the actual circuit ground. This resistor should have a value of 0.1Ω or less to stay within the allowable voltage compliance of the sense pin, particularly at higher operating current levels. The voltage between power ground/sense (Pin 7) and the signal ground (Pin 8) must stay within the range of $-1V$ to $+0.5V$. Internally there is approximately 25Ω between pins 7 and 8 and this resistance will slightly reduce the value of the external sense resistor. Approximately 70% of the quiescent supply current (10 mA) flows out of pin 7. This will cause a slight offset to the voltage across the sense resistor when the bridge is not conducting. During reverse recovery of the internal protection diodes the voltage compliance between pins 7 and 8 may be exceeded. The duration of these spikes however are only approximately 100 ns and do not have enough time or energy to disrupt the operation of the LMD18201.

Basic Motor Driver



TL/H/10793-10

Current Sensing



TL/H/10793-11

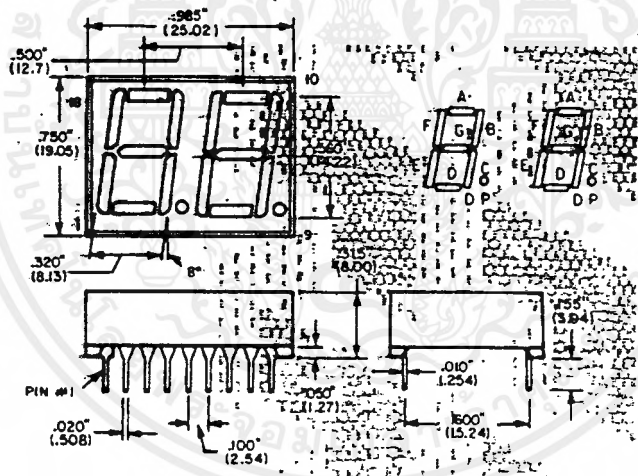
Fig. P37

ECG3074 - Red

1. E - Cathode - 1
2. D - Cathode - 1
3. C - Cathode - 1
4. Dec P₁ Cathode - 1
5. E - Cathode - 2
6. D - Cathode - 2
7. G - Cathode - 2
8. C₂ Cathode - 2
9. Dec P₁ Cathode - 2
10. B - Cathode - 2
11. A - Cathode - 2
12. F - Cathode - 2
13. Common Anode - 2
14. Common Anode - 1
15. B - Cathode - 1
16. A - Cathode - 1
17. G - Cathode - 1
18. F - Cathode - 1

ECG3075 - Red

1. E - Anode - 1
2. D - Anode - 1
3. C - Anode - 1
4. Dec P₁ Anode - 1
5. E - Anode - 2
6. D - Anode - 2
7. G - Anode - 2
8. C - Anode - 2
9. Dec P₁ Anode - 2
10. B - Anode - 2
11. A - Anode - 2
12. F - Anode - 2
13. Common Cathode - 2
14. Common Cathode - 1
15. B - Anode - 1
16. A - Anode - 1
17. G - Anode - 1
18. F - Anode - 1



ภาคผนวก ข.

แนวทางการพัฒนา

- 1.นำระบบการตรวจจับต่างๆมาติดตั้งบนตัวรถ เพื่อที่จะได้เพิ่มความสามารถของรถตำรวจ เช่น ตัวตรวจจับกลิ่น, ตัวตรวจจับความร้อน, ตัวตรวจจับเสียง
- 2.ใช้ CPU เบอร์ที่มีความสามารถสูงติดตั้งบนตัวรถ เพื่อที่ตัวรถจะสามารถเคลื่อนที่ไปได้เอง โดยไม่ต้องมีคนคอยควบคุมอยู่บนหน้าจอ และการเคลื่อนที่ของตัวรถสามารถทำได้หากคอมพิวเตอร์หรือระบบการสื่อสารด้วยคลื่นวิทยุ
- 3.ทำระบบการควบคุมโดยใช้ระบบไร้สาย เช่น ไร้รีโมตคอนโทรล ,ระบบการสื่อสารด้วยคลื่นวิทยุเพื่อเป็นการลดข้อจำกัดของการเคลื่อนที่ของรถที่ต้องต่อด้วยสาย

เอกสารอ้างอิง

1. “ National Power IC’s Databook ”, National Semiconductor
2. “ www.maxim.com “
3. “ การออกแบบกราฟิกด้วย HTML 4.0 ฉบับสมบูรณ์ ”, สวัสดิ์ ไกรคุณ
สำนักพิมพ์ เดอะ ไลบรารี พับลิชชิง
4. “ สร้าง Web Page แบบมืออาชีพด้วย HTML ”, กิตติ ภักดีวัฒนะกุล
5. “ การโปรแกรมภาษาแอสเซมบลี ของ ไมโครคอนโทรลเลอร์ตระกูล 8051 ”
สุนทร วิฑูรพจน์, บริษัทซีเอ็ด ยูเคชั่น จำกัด (มหาชน)
6. “ ไมโครคอนโทรลเลอร์ 8051 ”, รัชชัย อินทุโส, ไครภพ อินทุโส
หจก. สำนักพิมพ์ ฟิสสิกส์เซ็นเตอร์
7. “ คู่มือการเขียนโปรแกรม Delphi 4 ”, กนก กุศลมาลย์บุญกุล, ไกรวุฒิ มั่นเสถียรสิน
จัดพิมพ์โดย บริษัท รักษ์เซล มีเดีย จำกัด
8. “ คู่มือการใช้งาน Borland Delphi version 4 ”, กิตติ เปรมพิณี
9. “ กลเม็ดเคล็ดลับสไลด์ Borland Delphi 4 Client/Server Suite ”, วิมุตติวสะหลาย
บริษัท คอมกราฟ เพรส จำกัด
10. “ Borland Delphi 4 in 21 days ” ของ Kent Reisdorph
11. “ คัมภีร์การใช้งาน การสื่อสารอนุกรมบน PC ”, โดย Peder W.Gofton
12. “ การเขียนแอปพลิเคชันในวินโดวส์ด้วยบอร์แลนด์ ปาสคาล ”, บุญกุล กระจาย
บริษัทซีเอ็ด ยูเคชั่น จำกัด (มหาชน)
13. “ คู่มือและการประยุกต์ใช้งาน ไมโครคอนโทรลเลอร์ MCS-51 ”, ประเมษฐ์ ประณยานันท์,
ปิยพงศ์ เผ่าวนิช, บริษัทซีเอ็ด ยูเคชั่น จำกัด (มหาชน)