ระบบตรวจสอบสถานะที่จอดรถผ่านโทรศัพท์เคลื่อนที่

PARKING STATUS CHECKING SYSTEM VIA MOBILE PHONE

โดย

นายวัลนพ หลักแวงมล

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

ระบบตรวจสอบสถานะที่จอดรถผ่านโทรศัพท์เคลื่อนที่

PARKING STATUS CHECKING SYSTEM VIA MOBILE PHONE

โดย

นายวัลนพ หลักแวงมล 45010701

อาจารย์ที่ปรึกษา

ผศ.นภัทร สระเอี่ยม

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2549

ผ่านการตรวจรูปเล่มแล้ว

(ลงชื่อ).....ผู้ตรวจ

ผ่านการตรวจชิ้นงานแล้ว

(ลงชื่อ).....ผู้ตร

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง ระบบตรวจสอบสถานการณ์จอดรถผ่านโทรศัพท์เคลื่อนที่

PARKING STATUS CHECKING SYSTEM VIA MOBILE PHONE

ผู้จัดทำ

นายวัลนพ หลักแวงมล 45010701

..... อาจารย์ที่ปรึกษา

(ผศ.นภัทร สระเอี่ยม)

ระบบตรวจสอบสถานะที่จอดรถผ่านโทรศัพท์เคลื่อนที่
PARKING STATUS CHECKING SYSTEM VIA MOBILE PHONE

โดย นายวัลนพ หลักแวงมล 45010701

อาจารย์ที่ปรึกษา ผศ.นภัทร สระเอี่ยม

บทคัดย่อ

โครงงานนี้เป็นการสร้างระบบการตรวจสอบสถานะการจอดรถผ่านโทรศัพท์เคลื่อนที่ โดยไมโครคอนโทรลเลอร์จะทำการส่งข้อมูลสถานะที่ได้รับจากระบบเซนเซอร์ไปยังคอมพิวเตอร์ผู้ให้บริการ จากนั้นโปรแกรมประยุกต์บนคอมพิวเตอร์ผู้ให้บริการจะทำการเก็บบันทึกข้อมูล และแสดงผลให้กับผู้ใช้บริการทางโทรศัพท์มือถือผ่านเครือข่ายอินเตอร์เน็ต

ABSTRACT

This project invents a parking status via mobile phone. A microcontroller transmits status data received form sensor system to server computer. An application program on sensor computer records status data into database and displays on user's mobile phone via internet.

กิตติกรรมประกาศ

ปริญญานิพนธ์นี้สำเร็จได้ด้วยดี ทางคณะผู้จัดทำจึงขอขอบคุณทุกท่านที่มีส่วนร่วมในโครงงานนี้และขอบพระคุณ ผศ.นภัทร สระเอี่ยม และอาจารย์ทุกท่านในภาควิชาวิศวกรรมศาสตร์โทรคมนาคมที่ได้อดทนให้ความรู้ ให้คำแนะนำปรึกษา คลอดจนกราบขอบพระคุณบิดา มารดา ที่คอยสนับสนุนและเป็นกำลังใจมาโดยตลอด

ผู้จัดทำ
นายมนู วิเศษศิริวรชัย
นางสาวยุพดี ถิ่นชีลอง
นายวัลนพ หลักแวงมล

สารบัญ

หน้า

สารบัญ (ต่อ)

หน้า

หน้า

สารบัญรูปภาพ

หน้า

สารบัญรูปภาพ (ต่อ)

หน้า

สารบัญตาราง

หน้า

บทที่ 1
บทนำ

1.1 ที่มาและความสำคัญของโครงงาน

ในปัจจุบันการตื่นตัวของบรรดายักษ์ใหญ่ในแวดวงข่าวสารข้อมูลและการบันเทิงทั่วโลกต่อธุรกิจระบบเครือข่ายโทรศัพท์เคลื่อนที่รูปแบบหนึ่งของเทคโนโลยีอินเตอร์เน็ต (Internet) เป็นสัญญาณที่บ่งบอกถึงความคึกคักของธุรกิจผลิตข่าวสารข้อมูลได้เป็นอย่างดี การให้ความสนใจของผู้นำตลาดซอฟต์แวร์และการสื่อสารผ่านคอมพิวเตอร์ ไม่ว่าจะเป็นค่ายไมโครซอฟต์ ไปจนถึงค่ายเน็ตสเคป (Netscape) ล้วนเป็นการเริ่มต้นจุดกระแสการดำเนินการทางการตลาดผู้บริโภคเกี่ยวกับการสื่อสารแบบมัลติมีเดียไร้สาย เครื่องลูกข่ายโทรศัพท์ เคลื่อนที่รุ่นใหม่ ๆ ที่มีขีดความสามารถใกล้เคียงกับเครื่องคอมพิวเตอร์ขนาดเล็กจะเริ่มเข้าสู่ท้องตลาดมากขึ้นเรื่อย ๆ การสนับสนุนของรัฐบาลในประเทศต่าง ๆ ที่มีต่ออุตสาหกรรมการผลิตซอฟต์แวร์และการพัฒนาเว็บไซต์นั้นเป็นสิ่งที่กำลังจะเกิดขึ้นในช่วงเวลาอันใกล้นี้

สิ่งที่น่าจับตามองก็คือการปรับเปลี่ยนบทบาทของบรรดาผู้ผลิตอุปกรณ์เครือข่ายและเครื่องลูกข่ายโทรศัพท์เคลื่อนที่รายใหญ่ ซึ่งคาดกันว่าหลังจากเสร็จสิ้นการสร้างเครือข่าย 3G แล้ว ปริมาณความต้องการอุปกรณ์เครือข่ายสื่อสารไร้สาย รวมถึงโครงข่ายสื่อสัญญาณความเร็วสูง(High Speed Transmission Backbone) ของตลาดสื่อสาร โทรคมนาคมทั่วโลกจะลดลง ธุรกิจสื่อสาร โทรคมนาคมจะเริ่มเปลี่ยนเส้นทางไปรวมตัวกับธุรกิจอินเตอร์เน็ตมากขึ้นเรื่อย ๆ ซึ่งข้อเท็จจริงดังกล่าวเริ่มมีปัจจัยบ่งชี้ที่ชัดเจนขึ้นนับตั้งแต่การออกแบบเทคโนโลยีจีพีอาร์เอส (GPRS) ให้สามารถใช้เครือข่ายจีเอสเอ็ม (GSM) ได้ โดยไม่ต้องภารการลงทุนติดตั้งเครือข่ายสถานีฐานขึ้นใหม่ งบประมาณในการลงทุนติดตั้งเครือข่ายโทรศัพท์เคลื่อนที่ใหม่ ๆ เริ่มลดลง ยิ่งไปกว่านั้นผู้ให้บริการโทรศัพท์เคลื่อนที่หลายรายก็สนใจการเปิดให้บริการ โดยใช้เฉพาะเทคโนโลยีจีพีอาร์เอส

อย่างไรก็ตามโดยรวม จะเห็นว่าธุรกิจระบบเครือข่ายโทรศัพท์เคลื่อนที่รูปแบบหนึ่งของเทคโนโลยีอินเตอร์เน็ตเป็นการสร้างคุณค่าเพิ่มเติมให้กับการสื่อสารผ่านเครือข่ายโทรศัพท์เคลื่อนที่โดยเฉพาะจีพีอาร์เอสและ 3G ย่อมเป็นแนวทางที่ชัดเจนสำหรับธุรกิจโทรคมนาคมทั่วโลกในปัจจุบัน

โครงงานนี้จึงจัดทำขึ้นมา เพื่อเป็นต้นแบบการขยายขอบเขตของการใช้บริการโทรศัพท์เคลื่อนที่ไปในภาคการตรวจวัด ข้อมูลจากอุปกรณ์เซนเซอร์ ซึ่งสามารถนำไปประยุกต์ใช้ในงานวัดคุม หรือควบคุมอื่นๆได้ เช่น การตรวจวัดระดับความร้อน แสง เสียง รังสีคลื่นแม่เหล็กไฟฟ้า ฯลฯ โดยชนิดเซนเซอร์ขึ้นกับลักษณะงานที่แตกต่างกันออกไปหรือนำไปประยุกต์ใช้ในการให้บริการ เช่น การจองโต๊ะอาหารในภัตตาคาร สำหรับในโครงงานนี้ได้เลือกประยุกต์ใช้กับการให้บริการตรวจสอบสถานะลานจอดรถผ่านโทรศัพท์เคลื่อนที่

1.2 วัตถุประสงค์ของโครงงาน

1. สร้างระบบต้นแบบในการตรวจรับค่าเซนเซอร์ (Sensor) แสดงผลบนโทรศัพท์เคลื่อนที่ผ่าเครือข่ายอินเตอร์เน็ต ซึ่งสามารถใช้เป็นแนวทางนำไปประยุกต์ใช้งานอื่นๆ ได้ เช่น การจองโต๊ะอาหารในภัตตาคาร
2. พัฒนาแอพพลิเคชัน การให้บริการเว็บเซอร์วิส (Web service)
3. สามารถนำมาประยุกต์ใช้งานกับการให้บริการผ่านทางโทรศัพท์เคลื่อนที่ได้

1.3 ขอบเขตของโครงงาน

1. สามารถนำสัญญาณจากเซนเซอร์เก็บลงฐานข้อมูลเพื่อการให้บริการได้
2. สร้างเว็บแอพพลิเคชันที่สามารถติดต่อฐานข้อมูลได้
3. ระบบสามารถให้บริการกับโทรศัพท์ทั้งรุ่นที่ใช้บราวเซอร์และไมโครบราวเซอร์ได้

1.4 แนวทางในการดำเนินงาน

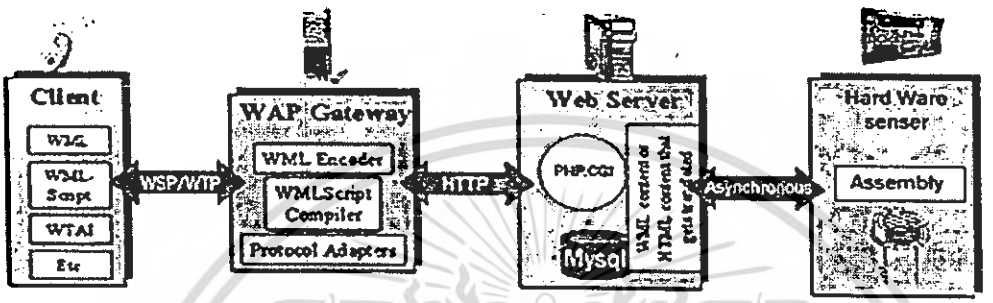
1. ใช้ไมโครคอนโทรลเลอร์ตระกูล 8051 และภาษาแอสแซมบลี จัดทำภาคฮาร์ดแวร์
2. ใช้ภาษา C++ ในการพัฒนาโปรแกรมวิ่งรับสัญญาณจากฮาร์ดแวร์บนเครื่องผู้ให้บริการ
3. ใช้ระบบจัดการฐานข้อมูล MySQL
4. ใช้ภาษา SQL ในการเข้าถึงฐานข้อมูลผ่านทางคอสโหมด (DOS Mode) หรือ PHP MyAdmin
5. ใช้ Apache Server จัดทำเซิร์ฟเวอร์
6. ใช้ภาษา PHP ในการเขียนเว็บแอพพลิเคชัน (Web Application) ติดต่อกับฐานข้อมูล MySQL
7. ใช้ภาษา HTML ในจัดทำหน้าเอกสารเว็บ โดย Macromedia Dreamweaver MX
8. ใช้ภาษา WML ในการจัดทำหน้าเอกสารแวบ โดย Nokia Mobile Internet Toolkit
9. ใช้โปรแกรมไมโครบราวเซอร์ สำหรับโทรศัพท์ยุคที่ 2.5 และใช้ บราวเซอร์ สำหรับสมาร์ทโฟนติดผ่านทางอินเตอร์เน็ตด้วยจีพีอาร์เอส

1.5 ประโยชน์ที่คาดว่าจะได้รับ

1. มีพื้นฐานการใช้ภาษาแอสแซมบลี C++, SQL, PHP, HTML, WML
2. มีความเข้าใจการทำงานของเว็บเซอร์วิส
3. มีพื้นประสบการณ์การโปรแกรมกับฐานข้อมูล
4. ได้รับความรู้ของระบบเครือข่าย การให้บริการ พัฒนาการของโทรศัพท์เคลื่อนที่

บทที่ 2
ทฤษฎีและหลักการ

ระบบตรวจสอบสถานะการจอดรถผ่านโทรศัพท์เคลื่อนที่มีบล็อกไดอะแกรม (Block diagram) การทำงานแสดงในรูปที่2.1



รูปที่ 2.1 บล็อกไดอะแกรมการทำงาน

จากรูปที่ 2.1 แบ่งออกเป็น 4 บล็อก โดยบล็อกแรกจะเป็นส่วนเกี่ยวกับวงจรของเซนเซอร์จะประกอบไปด้วยเซนเซอร์แอลดีอาร์ (Light Dependent Resistor, LDR) ที่ต่อกับวงจรเปรียบเทียบแรงดันไฟฟ้าเพื่อมาตรวจสอบสถานะการจอดรถ โดยนำค่าที่รับมาส่งไปยังวงจรไมโครคอนโทรลเลอร์ (Microcontroller) นำออกพอร์ต (Port) เข้าสู่ฐานข้อมูลที่ได้สร้างขึ้นมา ถัดมาก็จะเป็นบล็อกของเว็บเซิร์ฟเวอร์ที่มีฐานข้อมูลไว้คอยรับค่าเข้ามาบล็อกแวบ (WAP) บล็อกผู้ใช้บริการ

โครงสร้างของโครงงานมี 2 ส่วนหลักๆ ดังนี้

ส่วนฮาร์ดแวร์ (Hardware) ประกอบด้วย

1. ส่วนของวงจรเซนเซอร์และวงจรเปรียบเทียบเทียบแรงดันไฟฟ้า
2. ส่วนของวงจรไมโครคอนโทรลเลอร์

ส่วนซอฟต์แวร์ (Software) ประกอบด้วย

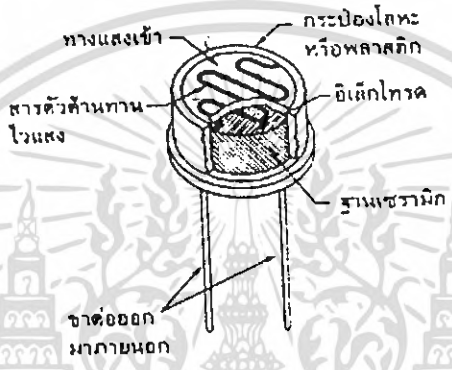
1. โปรแกรมควบคุมการทำงานในไมโครคอนโทรลเลอร์นำสัญญาณจากเซนเซอร์ส่งออกพอร์ต
2. โปรแกรมในเครื่องผู้ให้บริการ (Server)

C++ แอพพลิเคชัน (Application) ทำหน้าที่คอยรับสัญญาณจากพอร์ตอนุกรมที่ถูกส่งมาจากภาคไมโครคอนโทรลเลอร์ เชื่อมต่อฐานข้อมูลเพื่อนำสัญญาณที่รับมาได้จากฮาร์ดแวร์มาปรับปรุงฐานข้อมูล ซึ่งอาจเพิ่มหน้าที่พิเศษเพิ่มเติม เช่น เก็บบันทึกการใช้งานย้อนหลัง ตรวจสอบผู้ใช้บริการ เป็นต้น

- เว็บและแวบเซอร์วิส (Wap service) โปรแกรมให้บริการแวบและเว็บแก่โทรศัพท์เคลื่อนที่ด้วย WML,HTML,PHP

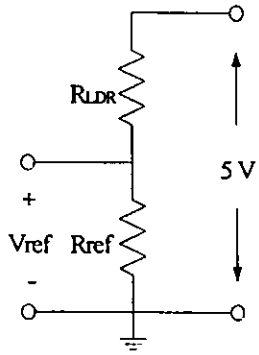
2.1 เซนเซอร์ (Sensors)

โครงสร้างตัวเซนเซอร์แอลดีอาร์ มีเรียกกันอีกหลายชื่อ เช่น โฟโต้คอนดักตีฟเซล (Photoconductive cell) หรือ ตัวต้านทานไวแสง (Light Sensitive Resistor, LSR) ส่วนใหญ่จะทำจากสารกึ่งตัวนำประเภทแคดเมียมซัลไฟด์ (CdS) หรือแคดเมียมซีนิไนด์ (CdSe) ซึ่งนำมาฉาบลงบนแผ่นเซรามิกที่ใช้เป็นฐานรองแล้วต่อขาจากสารกึ่งตัวนำออกมาใช้งาน



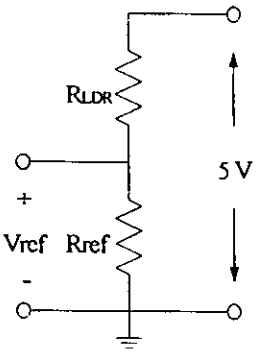
รูปที่ 2.2 โครงสร้างแอลดีอาร์

โดยโครงสร้างแอลดีอาร์แสดงดังรูปที่ 2.2 ประกอบด้วยสารตัวต้านทานไวแสง อิเล็กโทรดทำหน้าที่เป็นตัวนำไฟฟ้า สำหรับต่อขาออกมาใช้งานและฐานเซรามิก ค่าความต้านทานของแอลดีอาร์จะเพิ่มมากขึ้นหรือลดลงนั้นขึ้นอยู่กับค่าความเข้มของแสงที่ตกกระทบผิวหน้าของแอลดีอาร์ กล่าวคือ ความต้านทานของแอลดีอาร์เพิ่มขึ้นเมื่อความเข้มของแสงน้อยและความต้านทานของแอลดีอาร์น้อยลงเมื่อความเข้มของแสงมาก ซึ่งความเร็วในการเปลี่ยนค่าความต้านทานจากมากไปน้อยหรือจากน้อยไปมากนั้นขึ้นอยู่กับชนิดของแอลดีอาร์ จากหลักการดังกล่าวจึงนำแอลดีอาร์มาสร้างเป็นสวิตช์แสง โดยที่แสงมีความเข้มมากจะให้ตรรกะ 1 และเมื่อแสงมีความเข้มน้อยจะให้ตรรกะ 0



ก. ค่าความต้านทานของแอลดีอาร์น้อย แสงมาก

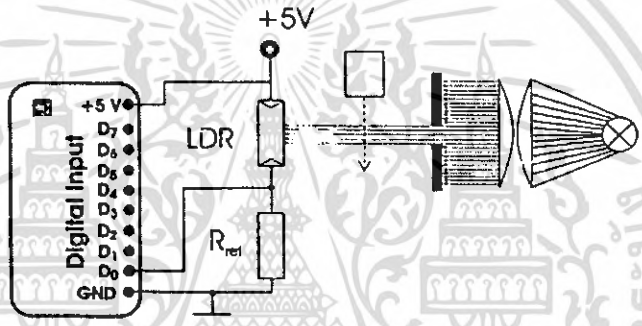
Vref = 5 V



ข. ค่าความต้านทานของแอลดีอาร์มาก แสงน้อย

Vref = 0 V

รูปที่ 2.3 แสดงการทำงานของแอลดีอาร์



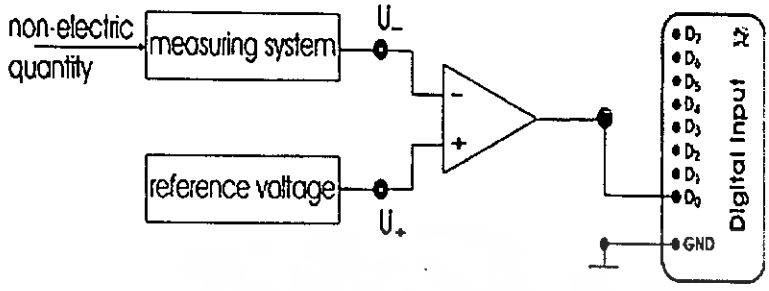
รูปที่ 2.4 ตัวอย่างการใช้งานของแอลดีอาร์

รูปที่ 2.4 แสดงตัวอย่างการใช้แอลดีอาร์ โดยทั่วไปจะใช้การบังแสงเลนส์รวมแสงจะอยู่หน้าแหล่งกำเนิดแสงเพื่อประมาณค่าของลำแสงขนานที่ตกลงบนเลนส์แอลดีอาร์ ถ้ามีวัตถุมาบังแสงทำให้ความสว่างที่แอลดีอาร์ลดลง ค่าความต้านทานเพิ่มขึ้นและค่าความต่างศักย์บนตัวต้านทานอ้างอิง(R_{ref}) ลดลง ซึ่งค่าของตัวต้านทานอ้างอิงขึ้นอยู่กับชนิดของแอลดีอาร์ ตำแหน่งของแอลดีอาร์และตัวต้านทานอ้างอิง

2.2 วงจรเปรียบเทียบแรงดันไฟฟ้า (Voltage comparator)

ในระบบวัดปริมาณทางกายภาพทั่วไป อย่างเช่น ตัวตรวจทางแสง เป็นต้น เอาต์พุตที่ได้เป็นแรงดันไฟฟ้าแบบต่อเนื่อง ซึ่งในบางกระบวน การควบคุมระบบนั้น เราไม่ต้องการค่าสัญญาณอะนาล็อกของตัววัด แต่เราต้องการแค่ตรวจสอบว่ามีหรือไม่มีเท่านั้น ฉะนั้นจะต้องมีวงจรที่มีกระบวนการ ในการเปรียบเทียบแรงดันไฟฟ้าได้ โดยจะต้องมีเกณฑ์ของแรงดันไฟฟ้าอ้างอิงใน การเปรียบเทียบวงจรนี้เรียกว่า วงจรเปรียบเทียบแรงดันไฟฟ้า โดยมี

หลักการการสร้างคือให้ตัววัดต่อเข้ากับขาแรงดันไฟฟ้าอินพุตบวกและแรงดันไฟฟ้าอ้างอิงต่อเข้าที่ขาแรงดันไฟฟ้าอินพุตลบ ดังรูป



รูปที่ 2.5 วงจรเปรียบเทียบแรงดันไฟฟ้า

จากรูปที่ 2.5 แสดงเอาต์พุตออกมาเป็นดิจิตอล ซึ่งจะมีเกณฑ์ในการดูเปรียบเทียบกันดังนี้

ถ้าแรงดันไฟฟ้า $U_+ > U_-$ แล้วเอาต์พุตที่ได้จะมีค่าสูงสุดคือ แรงดันไฟฟ้าบวกที่จ่ายให้ ออปแอมป์ $+U_s$ เทียบได้กับเป็นค่าลอจิก (logic) เป็น 1

ถ้าแรงดันไฟฟ้า $U_+ < U_-$ แล้วเอาต์พุตที่ได้จะมีค่าต่ำสุดคือ แรงดันไฟฟ้าลบที่จ่ายให้ ออปแอมป์ $-U_s$ เทียบได้กับเป็นค่าลอจิกเป็น 0

2.3 ไมโครคอนโทรลเลอร์ตระกูล MCS-51

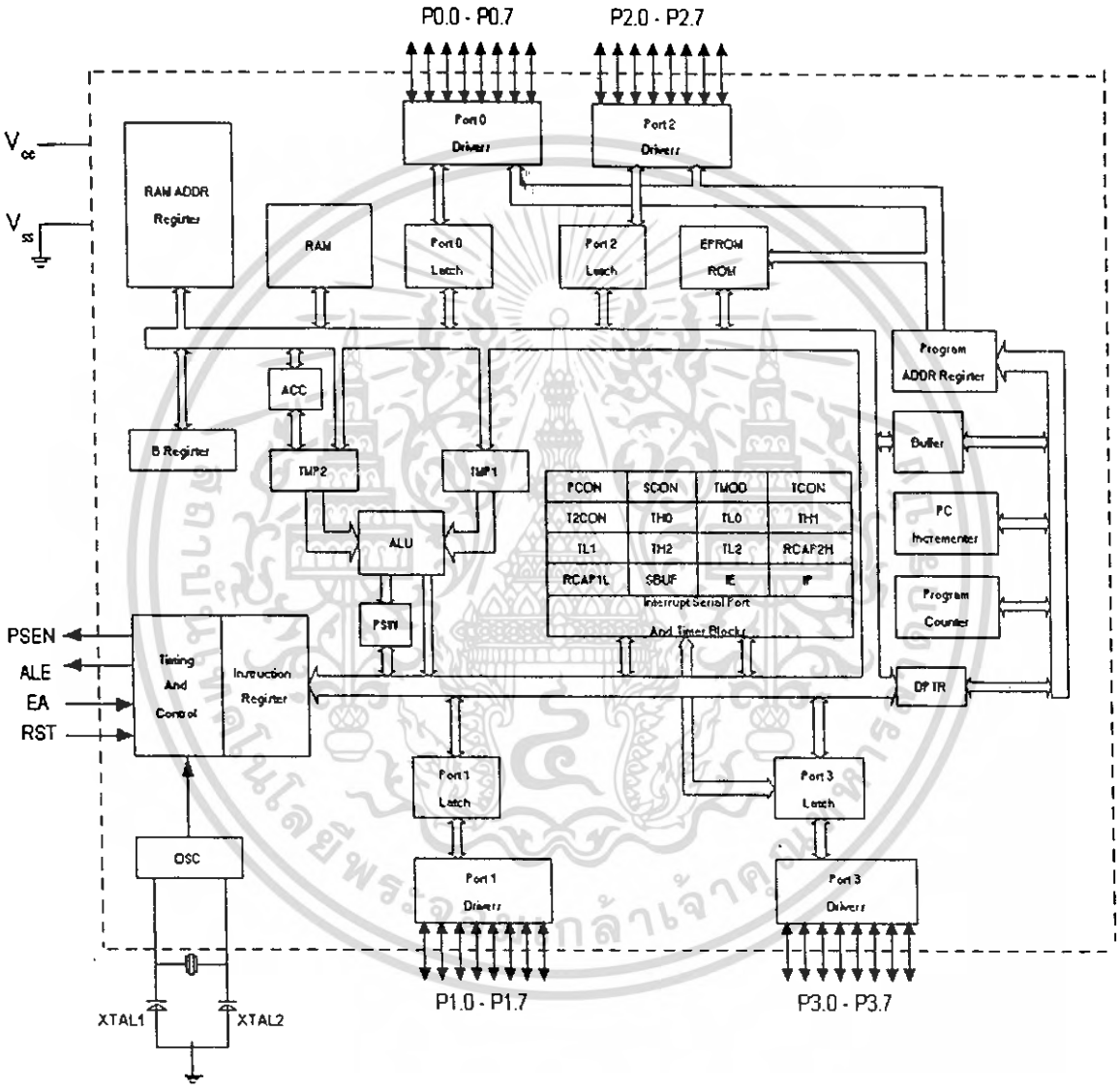
ไมโครคอนโทรลเลอร์ตระกูล MCS-51 เป็นชื่อเรียกของอุปกรณ์อิเล็กทรอนิกส์และมีความสามารถในการประมวลผล หน่วยคำนวณทางคณิตศาสตร์และลอจิก หน่วยความจำ เป็นต้น ซึ่งมีอยู่ด้วยกันหลายเบอร์ และเบอร์ที่เรียกใช้แตกต่างกันออกไปขึ้นอยู่กับโครงสร้างภายใน

2.3.1 คุณสมบัติของไมโครคอนโทรลเลอร์ MCS-51

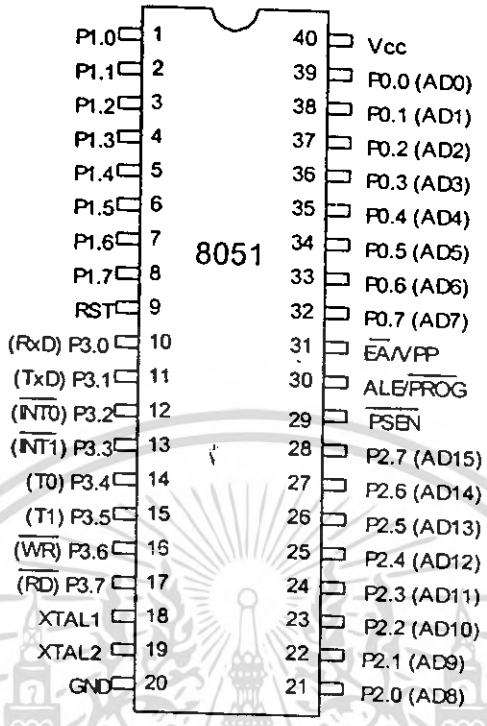
- ต้องการแหล่งจ่ายไฟ +5V
- มีหน่วยความจำโปรแกรมภายใน 4 กิโลไบต์ สำหรับ 8xx1 และ 8 กิโลไบต์ สำหรับ 8xx2 ส่วน 803x จะไม่มีหน่วยความจำส่วนนี้
- มีหน่วยความจำภายในสำหรับเก็บข้อมูล 128 ไบต์ สำหรับ 8xx1 และ 256 ไบต์สำหรับ 8xx2
- มีหน่วยความจำภายนอกสำหรับเก็บโปรแกรมและข้อมูล 64 กิโลไบต์
- คำสั่งที่ใช้เวลาน้อยที่สุดประมาณ 1 μs เมื่อทำงานที่ความถี่ 12 MHz
- มีตัวจับเวลาและตัวนับ 16 บิตซึ่งทำงานได้ 4 โหมด (Mode)
- มีพอร์ตที่สามารถรับส่งข้อมูลได้ทั้ง 2 ทิศทาง จำนวน 4 พอร์ต พอร์ตละ 8 บิต

- มีรีจิสเตอร์ (Register) สำหรับใช้งานเป็นไทม์เมอร์ (Timer) หรือเคาน์เตอร์ (Counter) เพื่อนับจำนวนสัญญาณนาฬิกา ภายในชิปหรือนับเปลี่ยนสถานะของสัญญาณภายนอก 16 บิต จำนวน 2 ตัว เพื่อใช้สำหรับนับพัลส์ วัดความกว้างของพัลส์ หรือใช้วัดช่วงเวลา

2.3.2 โครงสร้างภายในของ MCS-51



รูปที่ 2.6 โครงสร้างภายในของ MCS-51



รูปที่ 2.7 การจัดวางขาของ 8051

2.3.3 การจัดขาต่างๆ ของ 8051 ดังนี้

- V_{cc} ขาที่ 40 ต่อกับ +5V
- V_{ss} ขาที่ 20 ต่อลงกราวด์ (Ground)
- พอร์ต 0 ขาที่ 32 -39 มีทั้งหมด 8 บิต (P0.0 – P0.7) นอกจากจะใช้งานเป็นพอร์ตอินพุต-เอาต์พุตแล้วถูกใช้งานเป็นขาแอดเดรสและขาข้อมูล โดยแต่ละบิต จะมีชื่อ AD0-AD7 ถ้าต้องการให้พอร์ตนี้ทำหน้าที่เป็นอินพุต-เอาต์พุต ต้องป้อนลอจิก 1 ลงไปทุกบิตของพอร์ต 0 จึงจะทำงานได้
- พอร์ต 1 ขาที่ 1-8 มี 8 บิต (P1.0 - P1.7) โครงสร้างภายในจะมีตัวต้านทานต่อแบบพูลอัพ (Pull-up) อยู่ ทำให้สามารถใช้งานเป็นพอร์ตอินพุต-เอาต์พุตได้ทันที ถ้าต้องการให้พอร์ตนี้ทำหน้าที่เป็นอินพุต-เอาต์พุตต้องป้อนลอจิก 1 ลงไปทุกบิต
- พอร์ต 2 ขาที่ 21-28 มี 8 บิต (P2.0 – P2.7) โครงสร้างภายในคล้าย 0 และมีตัวต้านทานต่อแบบพูลอัพ ถ้าต้องการให้พอร์ตนี้ทำหน้าที่เป็น อินพุต-เอาต์พุตต้องป้อนลอจิก 1 ลงไปทุกบิต นอกจากนี้ยังใช้งานเป็นขาแอดเดรสสูงด้วย คือ AD8-AD15
- พอร์ต 3 ขาที่ 10 -17 มี 8 บิต(P3.0 – P3.7) โครงสร้างคล้ายพอร์ต 1 ซึ่ง ในแต่ละบิตได้ออกแบบไว้ให้ใช้งานเป็นพอร์ตควบคุม โดยมีหน้าที่ดังนี้

บิต	ชื่อฟังก์ชัน	ขาที่	หน้าที่
P3.0	RxD	10	ใช้เป็นขารับข้อมูลเข้าแบบอนุกรม
P3.1	TxD	11	ใช้เป็นขาส่งข้อมูลแบบอนุกรม
P3.2	INT0	12	ใช้เป็นขารับสัญญาณขัดจังหวะจากภายนอกหมายเลข 0
P3.3	INT1	13	ใช้เป็นขารับสัญญาณขัดจังหวะจากภายนอกหมายเลข 1
P3.4	T0	14	ใช้รับสัญญาณจากภายนอกของตัวนับหมายเลข 0
P3.5	T1	15	ใช้รับสัญญาณจากภายนอกของตัวนับหมายเลข 1
P3.6	WR	16	เป็นสัญญาณที่ใช้เขียนข้อมูลลงหน่วยความจำภายนอก
P3.7	RD	17	เป็นสัญญาณที่ใช้อ่านข้อมูลจากหน่วยความจำภายนอก

ตารางที่ 2.1 หน้าที่ต่างๆของพอร์ต 3 ในแต่ละบิต

- RST (Reset) ขาที่ 9 เป็นขารีเซต ซึ่งตัวซีพียู (CPU) จะรีเซตเมื่อเราป้อนลอจิก 1 อย่างน้อย 2 แมชชีนไซเคิล (Machine Cycle) ค่ารีจิสเตอร์ต่างๆหลังไมโครคอนโทรลเลอร์ถูกรีเซตจะมีค่า ดังนี้

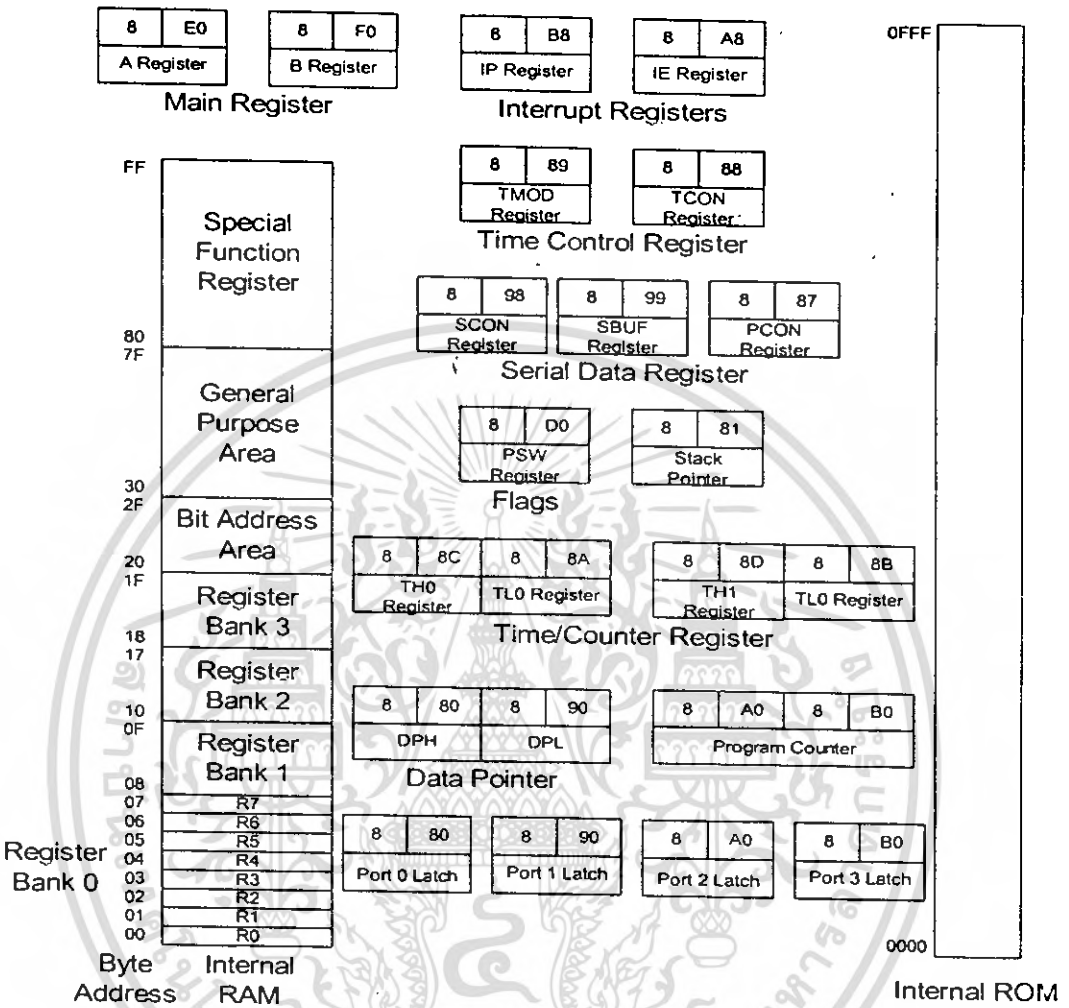
รีจิสเตอร์	ค่าเริ่มต้น
PC	0000
ACC	0000
B	0000
PSW	0000
SP	0007
DPTR	0000

ตารางที่ 2.2 ค่ารีจิสเตอร์ต่างๆ หลังถูกรีเซต

- EA (External Access) ขาที่ 31 ถ้าขานี้มีลอจิกเป็น 1 หมายความว่าให้อ่านโปรแกรมจากหน่วยความจำภายในชิป แต่ถ้าเป็น MCS-51 ที่มีหน่วยความจำอยู่ภายนอก ต้องต่อขาอีเอ ลงกราวด์ เป็นลอจิก 0 เพื่อให้ตัวซีพียูอ่านหน่วยความจำภายนอกชิป
- PSEN (Program Store Enable) ขาที่ 29 ขานี้จะทำงานเมื่อลอจิกเป็น 0 คือ ต้องการอ่านค่าจากหน่วยความจำโปรแกรมภายนอก แต่ถ้าเป็นการอ่านจากหน่วยความจำภายใน ขานี้จะไม่มีสัญญาณออกมา

- ALE (Address Latch Enable, ALE) ขาที่ 30 เป็นขาเอาต์พุตที่ทำงานเมื่อลอจิกเป็น 1 โดยซีพียูจะส่งสัญญาณนี้ออกมาเอง เนื่องจากพอร์ต P0 ทำงานได้ 2 หน้าที่คือ ขาแอดเดรสต่ำ และขาข้อมูล ดัว MCS-51 จะใช้สัญญาณขาเอแอลอีมัลติเพล็กซ์ (Multiplex) กับ P0 ว่าในขณะนั้นจะใช้งานเป็นขาแอดเดรส หรือขาข้อมูล
- XTAL1 ขาที่ 19 ใช้ต่อคริสตอล (Crystal) ภายนอกโดยเป็นอินพุตเข้าสู่วงจรกำเนิดสัญญาณ
- XTAL2 ขาที่20 ใช้ต่อคริสตอลภายนอกโดยเป็นเอาต์พุตของวงจรกำเนิดสัญญาณ

2.3.4 โครงสร้างของหน่วยความจำและรีจิสเตอร์ภายใน 8051

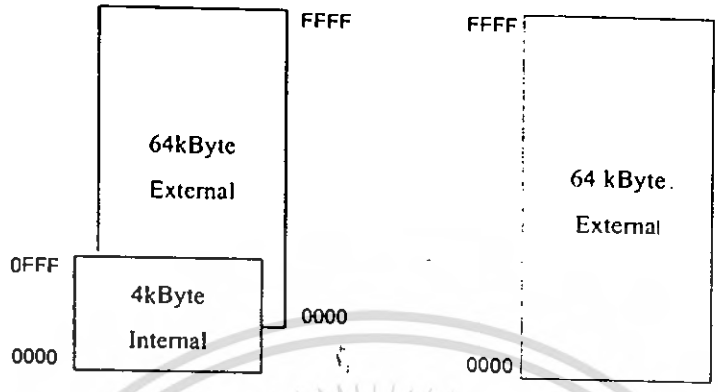


รูปที่ 2.8 ตำแหน่งของหน่วยความจำและรีจิสเตอร์ของ 8051

2.3.5 หน่วยความจำของไมโครคอนโทรลเลอร์ ซึ่งสามารถแบ่งได้เป็น

- หน่วยความจำภายใน ซึ่งจะแบ่งเป็นหน่วยความจำ ซึ่งก็คือ หน่วยความจำแบบรอม (ROM) ถ้าเป็น 8xx1 จะมี 4 กิโลไบต์ (0000 – 0FFF) และ 8xx2 จะมี 8 กิโลไบต์ (0000 – 1FFF) จะเอาไว้เก็บโปรแกรมสั่งงานภายในตัวชิป MCS-51 และอีกส่วนเป็นหน่วยความจำแบบแรม (RAM) ถ้าเป็น 8xx1 จะมี 128 ไบต์ (00 – 7F) ส่วน 8xx2 จะมี 256 ไบต์ (00 – FF)

- หน่วยความจำภายนอก ซึ่งของ MCS-51 มีได้ 64 กิโลไบต์ สามารถแบ่งใช้งานได้ดังนี้



ก.ใช้งานเป็นหน่วยความจำภายในกับภายนอก ข.แบ่งเป็นหน่วยความจำภายนอกอย่างเดียว

รูปที่ 2.9 (ก) และ (ข) การแบ่งใช้งานหน่วยความจำของ 8051

2.3.6 การทำงานของ 8051

ไมโครคอนโทรลเลอร์ 8051 จะทำงานได้ต้องประกอบด้วย 2 ส่วนคือ อุปกรณ์และส่วนโปรแกรม จะนำไอซี 8051 ไปใช้งานได้โดยจะต้องเขียนโปรแกรมคำสั่งและนำไปประมวลเป็นภาษาเครื่องเสียก่อน โดยโปรแกรมที่ประมวลผลแล้วจะมีลักษณะของข้อมูลเป็นเลขฐาน 16 จึงจะนำโปรแกรมนั้นมาใส่ในหน่วยความจำของไอซีเพื่อนำไปใช้งาน เมื่อทำการป้อนไฟเข้ามาเลี้ยงวงจร ไอซี 8051 จะทำการเซตค่าก่อนเริ่มทำงานทุกครั้ง หลังจากการรีเซตค่าของวงจร การทำงานจะเริ่มในส่วนของโปรแกรมเคาน์เตอร์ซึ่งจะทำหน้าที่เป็นวงจรนับและจะทำการส่งค่าตำแหน่งของหน่วยความจำของโปรแกรมไปเก็บไว้ยังโปรแกรมเอดีดีอาร์รีจิสเตอร์ (Program ADDR regisster) โดยในส่วนของเอดีดีอาร์จะทำหน้าที่เป็นวงจรที่ค้างค่าตำแหน่งของแอดเดรสที่รับเข้ามา ในสภาวะเริ่มทำงานเมื่อวงจรทำการรีเซต ค่าที่รับที่ตำแหน่งนี้จะมีค่าเป็น 000H หน่วยความจำโปรแกรมที่ได้จะเป็นค่าของหน่วยความจำภายนอกหรือภายใน จะทราบได้จากการตรวจสอบสภาวะลอจิกที่ขาเอเนเบิ้ลแอดเซส ที่ต่ออยู่กับวงจรในส่วนของไทม์มิ่งแอนด์คอนโทรล (Timing and control) โดยจะทำการสร้างสัญญาณและส่งค่าแอดเดรสไปยังรอมผ่านทางบัสไปพักไว้ที่อินสตรัคชั่นรีจิสเตอร์ (Instruction register) จะทำหน้าที่เอาค่าข้อมูลส่งกลับไปยังไทม์มิ่งแอนด์คอนโทรลเพื่อทำการถอดรหัสและนำไปควบคุมการทำงานในส่วนอื่นๆต่อไป ในกรณีที่ติดต่อกับรอมภายนอกนั้นรอมจะส่งข้อมูลที่อยู่ในตำแหน่งที่ถูกชี้โดยค่าแอดเดรสกลับมาพักไว้ในอินสตรัคชั่นรีจิสเตอร์และส่งกลับไปถอดรหัสที่ไทม์มิ่งแอนด์คอนโทรล การติดต่อรอมภายในก็เช่นเดียวกัน

2.3.7 ไมโครคอนโทรลเลอร์กับการรับส่งข้อมูลอนุกรม

การรับส่งข้อมูลแบบอนุกรมกับไมโครคอนโทรลเลอร์ MCS-51 ภายในไอซีจะมียูเออาร์ที (Universal asynchronous receiver transmitter, UART) เป็นตัวแปลงการรับส่งสัญญาณแบบขนานเป็นอนุกรมและแปลงการ

รับส่งสัญญาณแบบอนุกรมเป็นแบบขนาน การรับส่งข้อมูลอนุกรมของไมโครคอนโทรลเลอร์โครงสร้าง การทำงานในแบบที่เรียกว่าฟูลดูเพล็กซ์(Full duplex) คือการรับและส่งข้อมูลอนุกรมได้ในเวลาเดียวกัน การรับส่งข้อมูลแบบอนุกรม คือ การรับส่งข้อมูลออกมาทีละบิตจากตัวส่งไปตัวรับข้อมูล แล้วมารวมกันเป็นไบต์ซึ่งทางรับต้องคอยตรวจสอบว่าบิตใดเป็นบิตเริ่มต้นหรือบิตสุดท้ายของข้อมูล การตรวจสอบนั้นจะขึ้นอยู่กับรูปแบบของรหัสของบิตข้อมูลที่ใช้ซึ่งในการรับส่งข้อมูลแบบ อนุกรมระหว่างคอมพิวเตอร์กับอุปกรณ์ภายนอกนั้นจำเป็นต้องมีมาตรฐานในการรับส่งข้อมูล ซึ่งมาตรฐานที่นิยมมากที่สุดก็คือ มาตรฐาน RS - 232

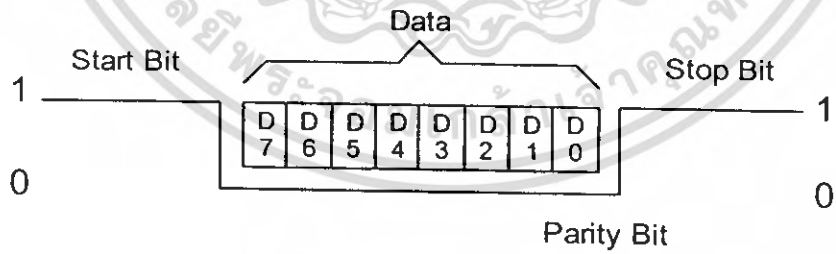
2.3.7.1 การสื่อสารข้อมูลแบบอะซิงโครนัส (Asynchronous)

การสื่อสารข้อมูลแบบอะซิงโครนัส คือการรับส่งข้อมูลโดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วย แต่จะใช้ความเร็วในการรับส่งข้อมูลให้มีค่าเท่ากัน จะใช้การแปลงข้อมูลให้เป็นอนุกรมแล้วเพิ่มเติมบิตบางอย่างร่วมเข้าไปกับการส่งข้อมูลจริงซึ่งได้แก่

1. บิตเริ่มต้น มีหน้าที่สำหรับบอกให้ทราบถึงตำแหน่งจุดเริ่มต้นก่อนบิตข้อมูล ค่าของบิตเริ่มต้นจะเป็นระดับลอจิกต่ำ

2. บิตแสดงสภาวะความเป็นเลขคู่หรือเลขคี่ บิตนี้มีหน้าที่ตรวจสอบความถูกต้องของข้อมูลซึ่งเรียกว่า บิตพาริตี้ซึ่งต่อท้ายบิตข้อมูล ค่าของบิตนี้ขึ้นอยู่กับจำนวนค่าของบิตที่เป็น 1 ซึ่งจะเป็นได้ 2 ลักษณะคือ พาริตี้คู่หรือพาริตี้คี่ เช่น ระบบที่ติดต่อกันโดยระบุว่าใช้พาริตี้คี่ ทางด้านส่งจะนำค่าข้อมูลที่จะส่งมาพิจารณาหาจำนวนบิตที่มีค่า 1 เป็นเลขจำนวนคู่อยู่แล้ว ค่าของพาริตี้จะมีค่าเป็น 0 แต่หากว่าจำนวนของบิตที่มีค่าเป็น 1 มีค่าเป็นเลขจำนวนคี่ ค่าของบิตพาริตี้ก็จะมีค่าเป็นหนึ่ง การพิจารณาทางด้านรับเป็นการตรวจสอบจำนวนบิตที่มีค่าเป็น 1 ของข้อมูลที่ได้รับมาทั้งหมดรวมทั้งบิตพาริตี้ ถ้ามีค่าเป็นเลขจำนวนคู่แสดงว่าข้อมูลที่ได้รับเข้ามานี้ถูกต้อง แต่หากไม่เป็นเลขจำนวนคู่แสดงว่าเกิดการผิดพลาดของข้อมูลขึ้น เป็นต้น

3. บิตสุดท้าย บิตสุดท้ายเป็นบิตที่เพิ่มขึ้นเพื่อระบุถึงขอบเขตการสิ้นสุดของกลุ่มบิตข้อมูล บิตสุดท้ายนี้สามารถโปรแกรมบิตได้คือ 1 บิต 1.5 บิต และ 2 บิต ดังนั้นกรณีของการส่งข้อมูล 8 บิต หากข้อมูลถูกส่งออกไปด้วยอัตราเร็ว 2400 บอกเวลาโดยรวมในการส่งข้อมูลหนึ่งไบต์จะมีค่าเป็น (12 x 416) หรือ 4.99*ms*



รูปที่ 2.10 การส่งข้อมูลอนุกรมแบบอะซิงโครนัส

เนื่องจากการสื่อสารแบบพอร์ตอนุกรมเป็นการรับส่งข้อมูลในลักษณะกลุ่มของบิตข้อมูล ดังนั้นจึงต้องพิจารณาถึงอัตราความเร็วในการส่งบิตข้อมูลเป็นอันดับแรก โดยทั่วไปมักระบุในหน่วยของจำนวนบิต

ข้อมูลภายในเวลาหนึ่งวินาทีเรียกว่าอัตราบอด ค่ามาตรฐานเหล่านี้ได้แก่ 110, 150, 300, 1200, 2400, 4800, 9600 และ 12000 บิตต่อวินาที เป็นต้น เช่น ถ้าข้อมูลทั้ง 8 บิตนี้ หากว่าถูกส่งออกมาด้วยอัตราบอด 2400 จะใช้เวลาในการส่งข้อมูลหนึ่งบิตมีค่าเท่ากับ 1/2400 หรือ 416μs เวลาในการส่งข้อมูลทั้ง 8 บิตมีค่าเท่ากับ (8x416) หรือ 3328μs

2.3.7.2 รีจิสเตอร์ที่เกี่ยวข้องกับการทำงานของพอร์ตอนุกรมใน MCS-51

ในการทำงานของพอร์ตอนุกรมใน MCS-51 มีรีจิสเตอร์ที่เกี่ยวข้อง ดังนี้

1. รีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม (Serial data buffer, SBUF) โดยทางตัวส่งจะมีรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม ทำหน้าที่เก็บข้อมูลที่จะส่งออกไป การใช้คำสั่งเขียนหรือโอนย้ายข้อมูลมายังรีจิสเตอร์นี้จะเป็นการส่งข้อมูลออกไปยังพอร์ตอนุกรมทางขาสัญญาณ TxD (พอร์ต 3.1) โดยอัตโนมัติ ส่วนด้านตัวรับจะมีรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรมเช่นเดียวกัน แต่จะทำหน้าที่เก็บข้อมูลที่นำมาจากส่วนของวงจรเลื่อนบิต (Shift register) ของวงจรรับข้อมูลสัญญาณข้อมูลอนุกรมที่รับเข้ามาจะผ่านทางขา RxD (พอร์ต 3.0)

2. รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม (Serial port control register, SCON) เป็นรีจิสเตอร์ 8 บิต ที่เข้าถึงข้อมูลระดับบิตได้ ใช้ในการกำหนดค่าบิตเริ่มต้น บิตสุดท้าย เลือกโหมดการสื่อสาร โดยแต่ละบิตจะมีความหมายดังนี้

SM0	SM1	SM2	REN	TB8	RB8	TI	RI

รูปที่ 2.11 โครงสร้างของรีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม

โดยที่ SM0-SM1 : เป็นบิตที่ใช้เลือกโหมดการสื่อสารซึ่งมีอยู่ด้วยกัน 4 โหมด

SM2 : เป็นบิตที่ใช้ควบคุมการสื่อสารแบบมัลติโพรเซสเซอร์

REN : receive enable เป็นบิตที่ควบคุม การรับข้อมูลของพอร์ตอนุกรม ถ้าบิตนี้ถูกเซตเป็นลอจิก 1 หมายความว่า จะรับข้อมูลเข้ามาได้

TB8 : transfer bit 8 เป็นบิตที่ใช้เก็บข้อมูลที่ 9 ที่จะส่งออกไปทางพอร์ตอนุกรมเมื่อทำงานในโหมด 2 และ 3

RB8 : receive bit 8 เป็นบิตที่ใช้เก็บข้อมูลที่ 9 เมื่อรับข้อมูลเข้ามาทางพอร์ตอนุกรมเมื่อทำงานในโหมดที่ 2 และ 3

TI : transmit interrupt เมื่อ MCS-51 ส่งข้อมูลหนึ่งไบต์ออกทางพอร์ตอนุกรมเรียบร้อยแล้ว บิตนี้จะมีค่าลอจิกเป็น 1 ซึ่งสามารถนำไปขัดจังหวะไมโครคอนโทรลเลอร์ได้ โดยบิตนี้จะเซตเมื่อมีการส่งบิตหยุดของข้อมูลออกไปและต้องเคลียร์ด้วยซอฟต์แวร์

RI : receive interrupt เมื่อ MCS-51 รับข้อมูลเข้ามาทางพอร์ตอนุกรมเรียบร้อยแล้ว บิตนี้จะมีค่าลอจิกเป็น 1 ซึ่งสามารถนำไปขัดจังหวะไมโครคอนโทรลเลอร์ได้ โดยบิตนี้จะถูกเซต

เมื่อมีการส่งบิตหยุดข้อมูลได้ส่งเข้ามาแล้วและต้องเคลียร์ด้วยซอฟต์แวร์

SM0	SM1	โหมด	ลักษณะการทำงาน
0	0	0	เป็นการขยายพอร์ตอินพุตและเอาต์พุต
0	1	1	ใช้สำหรับการเชื่อมต่ออนุกรมแบบยูเออาร์ทีโดยใช้ข้อมูลเป็น 10 บิต และสามารถเปลี่ยนแปลงความเร็วในการส่งข้อมูลได้
1	0	2	ใช้สำหรับการเชื่อมต่ออนุกรมแบบยูเออาร์ที โดยใช้ข้อมูลเป็น 11 บิต แต่ความเร็วในการส่งข้อมูลจะกำหนดไว้คงที่
1	1	3	ใช้สำหรับการเชื่อมต่ออนุกรมแบบยูเออาร์ที โดยใช้ข้อมูลเป็น 11 บิต และสามารถเปลี่ยนแปลงความเร็วในการส่งข้อมูลได้

ตารางที่ 2.3 การเซตค่า SM0-SM1 ในการเลือกโหมดการทำงาน

พอร์ตอนุกรมโหมด 0

การทำงานพอร์ตอนุกรมโหมด 0 นั้นเป็นการขยายพอร์ตอินพุตหรือเอาต์พุต ให้มีจำนวนมากขึ้น โดยจะทำการสร้างสัญญาณนาฬิกาขึ้นเพื่อให้จังหวะของการทำงานที่พร้อมกัน สำหรับการเลื่อนบิตจากไอซีรีจิสเตอร์ภายนอก เมื่อมีการโอนย้ายข้อมูลมายังรีจิสเตอร์ในแต่ละครั้งก็จะมีผลให้เกิดการส่งข้อมูลทั้ง 8 บิตออกมา แม้ว่าแฟล็กสถานะ TI จะยังมีค่าเป็น 1 อยู่ นอกจากนี้เมื่อใดก็ตามที่ค่าแฟล็กสถานะ RI ค่าเป็น 1 ก็จะย้ายข้อมูลนั้นออกจากรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรมเสียก่อนที่จะได้มีการกำหนดค่าแฟล็ก RI เป็น 0 เพื่อให้รับ ข้อมูลใหม่ต่อไป

การทำงานของพอร์ตอนุกรมโหมด 0 เป็นการรับและส่งข้อมูลอนุกรม 8 บิตโดยใช้เพียงขา RxD เท่านั้น ส่วนขา TxD จะนำไปใช้เพื่อเป็นขาสัญญาณนาฬิกาในการให้จังหวะการเลื่อนข้อมูลกับวงจรเลื่อนบิตภายนอก สำหรับอัตราการเลื่อนบิตจะถูกกำหนดไว้คงที่ที่ค่า 1/2 ของความถี่ออสซิลเลเตอร์

พอร์ตอนุกรมโหมด 1

การทำงานพอร์ตอนุกรมโหมด 1 เป็นการสื่อสารข้อมูลอนุกรม 10 บิต ประกอบด้วยบิตเริ่มต้น 1 บิต บิตข้อมูล 8 บิต และบิตหยุด 1 บิต โดยข้อมูลจะถูกส่งออกทางขา TxD และรับสัญญาณเข้าที่ขา RxD ในส่วนของข้อมูล 8 บิตที่ได้รับหรือส่งออกจะเป็นบิตนัยสำคัญต่ำเป็นอันดับแรก และบิตสุดท้ายของข้อมูลที่รับเข้ามาจะเก็บในบิต RB8 ในรีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม สำหรับอัตราความเร็วในการส่งข้อมูลของโหมด 1 สามารถกำหนดเลือกได้ โดยใช้ไทม์เมอร์ 1 ทำหน้าที่เป็นตัวกำเนิดอัตราการส่งข้อมูล

- กรณีใช้ ไทม์เมอร์ 1 ทำงานในโหมด 2 (8 bit automatic reload)

$$\text{ความถี่อัตราบอด} = \frac{2^{SMOD}}{32} \times \frac{Oscillator\ frequency}{12 \times [256 \times TH1]}$$

- กรณีใช้ ไทม์เมอร์ 1 ทำงานในโหมดอื่นที่ไม่ใช่โหมด 2

$$\text{ความถี่อัตราบอด} = \frac{2^{SMOD}}{32} \times \text{อัตราการ } overflow \text{ ของ Timer 1}$$

พอร์ตอนุกรมโหมด 2 และ 3

พอร์ตอนุกรมโหมด 2 และ 3 จะทำการรับส่งข้อมูลจำนวน 11 บิตเหมือนกัน ซึ่งประกอบด้วย บิตเริ่มต้น บิตข้อมูล บิตข้อมูลบิตที่ 9 และบิตหยุด แต่พอร์ตอนุกรมโหมด 2 และ 3 จะต่างกันตรงที่โหมด 2 นั้นไม่สามารถเปลี่ยนแปลงอัตราการรับส่งข้อมูลได้

$$\text{ความถี่อัตราบอด} = \frac{2^{SMOD}}{32} \times Oscillator\ frequency$$

ในขณะที่โหมด 3 สามารถเปลี่ยนแปลงอัตราการรับส่งข้อมูลได้ ซึ่งมีการคำนวณอัตราบอดเหมือนพอร์ตอนุกรมโหมด 1

การส่งข้อมูลอนุกรมในโหมด 2 และ 3 จะต้องนำค่าข้อมูลนั้นไปเก็บยังรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม สำหรับค่าของบิตที่ 9 ที่เพิ่มขึ้นนั้นนำมาจากค่าของ TB8 ภายในรีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม ซึ่งจะต้องได้รับการกำหนดค่าจากผู้ใช้งาน เมื่อข้อมูลถูกส่งออกไปภายนอกเรียบร้อยแล้ว แฟล็กสถานะ TI จะมีค่าเป็น 1 และผู้ใช้จะต้องทำการเปลี่ยนกลับให้เป็นค่า 0 ตามเดิม สำหรับการรับข้อมูลจะถูกนำมาเก็บไว้ในรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม โดยค่าบิตที่ 9 จะนำไปเก็บไว้ยังบิต RB8 ภายในรีจิสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม

2.4 มาตรฐาน RS - 232

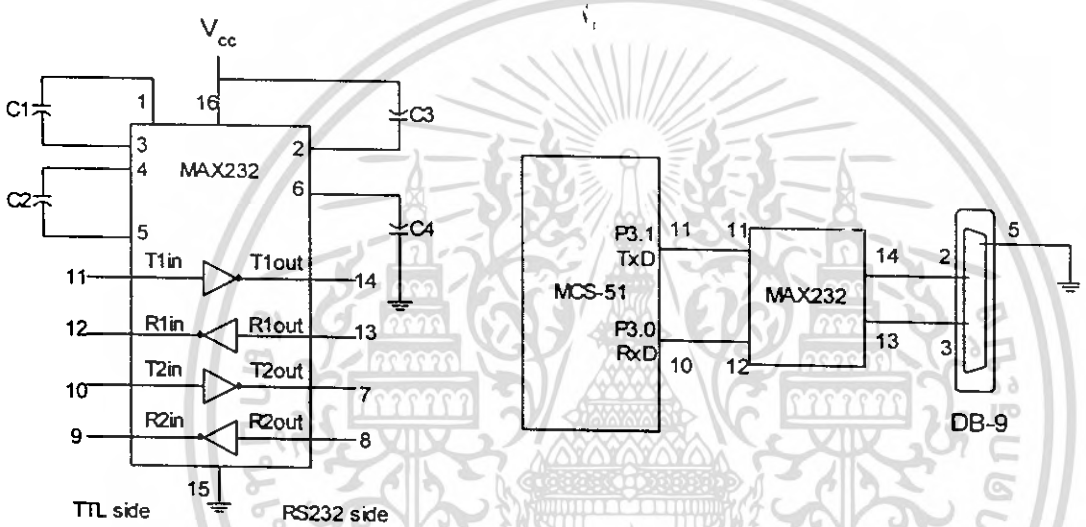
มาตรฐาน RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อการใช้งานรูปแบบของตัวเชื่อมต่อที่สอดคล้องกัน จะได้ลดปัญหาการเข้ากันไม่ได้ระหว่างสัญญาณของอุปกรณ์ที่นำมาเชื่อมต่อกันทั้ง 2 ด้าน โดยได้กำหนดรูปแบบของอุปกรณ์เชื่อมต่อข้อมูล (Data terminal equipment, DTE) กับวงจรข้อมูลปลายทาง (Data communication equipment, DCE) จะได้ว่าอุปกรณ์ดีทีอีจะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัว เช่น

communication equipment, DCE) จะได้ว่าอุปกรณ์ดีทีอีจะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัว เช่น ไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ ส่วนอุปกรณ์ดีซีอีจะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจากทางดีทีอี โดยการใช้สายสัญญาณเพียง 3 เส้นเท่านั้น คือ ใช้สาย TxD สาย RxD และสายกราวด์

2.4.1 การเชื่อมต่อ MCS-51 เข้ากับ RS - 232

เนื่องจากระบบแรงดันไฟฟ้าที่ใช้และการแทนระดับลอจิกตามแต่ละมาตรฐานนี้จะแตกต่างกันไป โดยที่ MCS - 51 ใช้ระดับสัญญาณมาตรฐาน TTL ซึ่งคนละมาตรฐานกับการส่งข้อมูลอนุกรมมาตรฐาน RS-232 โดยสามารถนำ IC MAX 232 มาช่วยปรับแรงดันเพื่อการรับส่งสัญญาณกันได้อย่างถูกต้อง

พอร์ตการรับส่งข้อมูลอนุกรมของ MCS-51 จะส่งออกทางขา 10 RxD และขา 11 TxD โดยต่อเข้ากับ IC MAX 232 ที่ขา 12 และ ขา 11 ตามลำดับ ดังแสดงในรูปที่ 2.12



รูปที่ 2.12 โครงสร้างภายใน MAX 232 และการเชื่อมต่อกับ MCS - 51

2.5 MySQL

วัตถุประสงค์เพื่อสร้างภาษาที่ใช้งานร่วมกับระบบสารสนเทศและฐานข้อมูลได้ ภาษา SQL (Structure query language) จึงถูกพัฒนาขึ้นและเริ่มเข้ามามีบทบาทอย่างกว้างขวางเนื่องจาก SQL เป็นกลไกสำคัญซึ่งควบคุมการทำงานของระบบจัดการฐานข้อมูลในยุคปัจจุบัน

ในช่วงต้นปี ค.ศ. 1990 โปรแกรมเมอร์กลุ่มหนึ่งได้ร่วมกันพัฒนาซอฟต์แวร์และเชื่อมโยงข้อมูลกับฐานข้อมูลขึ้นโดยใช้ชื่อว่า mSQL ภาษานี้ได้รับความนิยมในหมู่ผู้ใช้ซึ่งนิยมโปรแกรมฟรีแวร์ และเมื่อมีการใช้ซอฟต์แวร์นี้มากขึ้น เนื่องจากผู้ใช้ส่วนใหญ่ต้องการให้ mSQL รุ่นใหม่มีการถ่ายทอดข้อมูลที่รวดเร็วและคล่องตัวมากกว่าที่เป็นจึงส่งผลต่อการออกแบบระบบถ่ายทอดข้อมูลใน SQL ที่พัฒนาในรุ่นต่อมาระบบใหม่ที่ออกแบบนั้นมีการทำงานความคล้ายกับระบบเชื่อมโยงข้อมูลแบบ API ใน mSQL

2.5.1 คุณลักษณะของ MySQL

MySQL เป็นโปรแกรมจัดการฐานข้อมูลซึ่งมีลักษณะเป็นฟรีแวร์ พัฒนาขึ้นโดยบริษัท MySQL AB ในประเทศสวีเดน บริษัท MySQL AB ก่อตั้งโดย David Axmark, Allan Lasson และ Michael Monty Widenius โดยมีสมาชิกเป็นนักพัฒนาซอฟต์แวร์จาก 12 ประเทศทั่วโลก ซึ่งติดต่อสื่อสารกันผ่านระบบเครือข่ายและอินเตอร์เน็ต

MySQL เติบโตและพัฒนาอย่างรวดเร็วโดยผู้ใช้กลุ่มต่างๆได้พยายามพัฒนาฟรีแวร์ชนิดนี้เพื่อใช้ภายในกลุ่มและเผยแพร่สู่ผู้ใช้อื่นๆต่อไป เช่นเดียวกับรูปแบบการพัฒนาของระบบปฏิบัติการลีนุกซ์ (Linux) โดยวัตถุประสงค์ของผู้ผลิตซอฟต์แวร์มีดังนี้

- ต้องการสร้างสรรค์และพัฒนาซอฟต์แวร์สำหรับการจัดการฐานข้อมูลที่มีขนาดเล็กแต่ให้มีความสามารถสูง
- เป็นซอฟต์แวร์ที่มีกลุ่มผู้ใช้ทุกมุมโลก
- ใช้งานง่าย
- มีการปรับปรุงอย่างต่อเนื่องและรวดเร็ว
- ปราศจากข้อผิดพลาดจากตัวโปรแกรม
- ไม่มีค่าใช้จ่ายในการจัดหา

MySQL เป็นโปรแกรมสำหรับใช้ในการสร้างฐานข้อมูล โดยมีคุณลักษณะของระบบจัดการฐานข้อมูลที่ไม่ยิ่งหย่อนไปกว่าระบบจัดการฐานข้อมูลชนิดอื่นๆ รวมทั้งสามารถสร้างและจัดการฐานข้อมูลขนาดใหญ่ได้รวดเร็วอีกด้วย โดยที่ MySQL มีระบบสืบค้นข้อมูลที่รวดเร็วและแม่นยำ สามารถใช้งานได้กับคอมพิวเตอร์ระบบ Stand-Alone และเครือข่ายรวมทั้งทำงานร่วมกับแอพพลิเคชันได้หลายชนิด MySQL เป็นระบบจัดการฐานข้อมูลเชิงสัมพันธ์

MySQL มีความสามารถในการเชื่อมโยงข้อมูลระหว่างตาราง จัดเก็บข้อมูลจำนวนมากสะดวก และค้นหาง่ายซึ่งเป็นคุณลักษณะปกติของโปรแกรมที่พัฒนาขึ้นมาจาก SQL แต่การสร้างฐานข้อมูลเชิงสัมพันธ์ของ MySQL ให้ทางเลือกในการออกแบบ และพัฒนาข้อมูลแก่ผู้ใช้มากกว่าโปรแกรมจัดการฐานข้อมูลชนิดอื่น

MySQL เป็นซอฟต์แวร์แบบฟรีแวร์และเป็นโอเพ่นซอร์ส (Open source) หมายถึงผู้ใช้ MySQL สามารถพัฒนาโปรแกรมต่อเนื่องได้อย่างอิสระและทุกคนมีสิทธิ์ที่จะดาวน์โหลด (Download) ระบบจัดการฐานข้อมูลนี้ผ่านทางอินเตอร์เน็ตหรือทำสำเนาได้ แต่โปรแกรม MySQL มีการจดลิขสิทธิ์ ดังนั้นสิทธิบางประการ เช่น การจัดจำหน่ายซอฟต์แวร์ที่มาจาก MySQL หรือการจำหน่ายซอฟต์แวร์เสริมจากการทำงานของ MySQL จะถูกสงวนไว้โดยบริษัทผู้ผลิต ซึ่งคุณสมบัติของโปรแกรมจัดการฐานข้อมูลตัวนี้มีดังนี้

- เป็นระบบจัดการฐานข้อมูลภายในคอมพิวเตอร์ และสามารถเคลื่อนย้ายระบบได้
- พัฒนาจากภาษา C และ C++ ทำให้แปลงข้อมูลได้หลายรูปแบบ
- สามารถทำงานกับระบบปฏิบัติการที่แตกต่างกันและให้พื้นที่ในการจัดเก็บข้อมูลมากกว่า SQL

เช่น ในระบบปฏิบัติการ Solaris 2.6 ระบบจัดการฐานข้อมูล MySQL ให้พื้นที่เก็บข้อมูลสูงสุด2GB

- เลือกรูปแบบการเชื่อมต่อได้หลายระบบ เช่น c, c++, Eiffel, Java, Perl, PHP, Python หรือ Ruby เป็นต้น
- เก็บข้อมูลที่มีขนาดใหญ่ได้ดี
- มีชนิดของข้อมูลให้เลือกมาก โดยแบ่งออกเป็น integer 1, 2, 3, 4 และ 8 bytes, FLOAT, DOUBLE, CHAR, VARCHAR, TEXT, BLOB, DATE, TIME, DATETIME, TIMESTAMP, YEAR, SET หรือ ENUM

โปรแกรมที่ใช้ในการจัดการฐานข้อมูลมีอยู่หลายชนิดที่สนับสนุนการทำงานของ PHP ได้แก่

- ชนิด ORACLE เช่น Oracle (OC17 and OCI8), AdabasD, Ingres, FilPro (read-only) และ Solid เป็นต้น
- ชนิด Access เช่น dBase, InterBase, Ovrimos Empress และ FrontBase เป็นต้น
- ชนิด SQL เช่น MS SQL, PostgreSQL, mSQL และ MySQL เป็นต้น

สำหรับในโครงงานชิ้นนี้เลือก ระบบการจัดการฐานข้อมูล MySQL เนื่องจากถูกพัฒนามาให้มีความเข้ากันได้ดีกับ PHP

2.5.2 การใช้งาน MYSQL

MySQL เป็นระบบจัดการฐานข้อมูลที่ติดต่อกับผู้ใช้ผ่านทางบรรทัดคำสั่งของดอส(DOS) การใช้งาน MySQL นั้นเราจะต้องเปิดวินโดว์ MS-DOS Promp (หรือ Command Prompt ในกรณีของ Windows 2000/XP) ขึ้นมาก่อน จากนั้นให้เข้าไปยังโฟลเดอร์ c:\MySQL\bin (ด้วยคำสั่ง cd\MySQL\bin) แล้วป้อนคำสั่ง MySQL -u root -p

จากนั้นกดเอนเตอร์ (Enter) แล้วให้ป้อนรหัสผ่านลงไปแล้วกดเอนเตอร์ ซึ่งถ้าหากไม่มีอะไรผิดพลาดจะปรากฏ MySQL> ขึ้นมาแทนเครื่องหมายรับคำสั่งของดอส ก็แสดงว่าเราอยู่ในโหมดการทำงานของ MYSQL แล้ว

ในโครงงานนี้เราใช้ฐานข้อมูลเพื่อเก็บข้อมูลคำสั่งของผู้ใช้ในการควบคุมอุปกรณ์เพียงแค่ 8 ชุด ดังนั้นการสร้างตารางขึ้นมา 1 ตารางเพื่อเก็บคำสั่งของผู้ใช้ ซึ่งจะใช้คำสั่ง SQL ในการสร้างขึ้นมาโดยมีคำสั่งดังนี้

การสร้างฐานข้อมูล

- MySQL>create databases ชื่อฐานข้อมูล ;

หลังจากสร้างฐานข้อมูลแล้วให้ทำการเลือกการใช้งานด้วยคำสั่ง

MySQL>use ชื่อฐานข้อมูล ;

2.5.2.1 การค้นหาข้อมูลด้วยอินเด็กซ์ (Index)

อินเด็กซ์เป็นเครื่องมือที่ช่วยในการค้นหาข้อมูลจากฐานข้อมูล เนื่องจากข้อมูลอาจจะเก็บอยู่อย่างกระจัดกระจายเพราะ MySQL เป็นฐานข้อมูลชนิดเชิงสัมพันธ์ ทำให้กระบวนการค้นหาข้อมูลตามปกติหากมีข้อมูลมากจะทำงานได้ช้า การกำหนดหรือสร้างอินเด็กซ์สามารถทำได้ 2 กรณี กรณีแรกคือ กำหนดอินเด็กซ์ด้วยคำสั่ง

CREATE TABLE ตอนสร้างตารางทันที ส่วนอีกกรณีคือ เมื่อมีตารางอยู่แล้วมากำหนดอินเด็กซ์ภายหลังโดยใช้คำสั่ง ALTER TABLE ...ADD อินเด็กซ์ มี 4 ประเภท คือ

1. PRIMARY KEY เป็นอินเด็กซ์สำหรับกำหนดให้แก่คอลัมน์ที่ต้องการเก็บข้อมูลที่ไม่ซ้ำกัน และมีเงื่อนไขบังคับว่า คอลัมน์ที่จะกำหนดเป็นอินเด็กซ์ประเภทนี้ ได้ ต้องไม่เก็บค่านัล (NULL) กล่าวคือ คอลัมน์ที่จะกำหนดอินเด็กซ์ประเภทนี้ได้ต้องเป็นคอลัมน์ที่มีคุณสมบัติไม่นัล (NOT NULL) โดยอาจจะกำหนดคุณสมบัติไม่นัลไว้แล้วค่อยกำหนดอินเด็กซ์ภายหลัง (ด้วยคำสั่ง ALTER TABLE...ADD) หรือกำหนดอินเด็กซ์นี้พร้อมๆ กับกำหนดคุณสมบัติไม่นัลก็ได้ (ด้วยคำสั่ง CREATE TABLE) ถ้าคอลัมน์ใดไม่มีคุณสมบัติไม่นัลอยู่ก่อน จะไม่สามารถสร้างหรือกำหนดอินเด็กซ์ PRIMARY KEY ให้แก่คอลัมน์นั้นได้

2. UNIQUE เหมือนกับ PRIMARY KEY คือ ใช้สำหรับคอลัมน์ที่ต้องการเก็บข้อมูลที่ไม่ซ้ำกัน แต่มีข้อแตกต่างจาก PRIMARY KEY คือ UNIQUE ไม่มีเงื่อนไขบังคับว่า คอลัมน์ที่กำหนดเป็นอินเด็กซ์ประเภทนี้ได้ ต้องมีคุณสมบติไม่นัลกล่าวคือไม่ว่าคอลัมน์นั้นจะกำหนดหรือไม่กำหนดคุณสมบัติไม่นัล สามารถกำหนดอินเด็กซ์ UNIQUE ได้ ซึ่งสามารกำหนดคุณสมบัติไม่นัลให้แก่คอลัมน์นั้นได้ภายหลัง ด้วยคำสั่ง ALTER TABLE...MODIFY เว้นแต่ว่าข้อมูลเดิมในคอลัมน์มีค่านัลอยู่แล้ว จะกำหนดคุณสมบัติไม่นัลไม่ได้หรือต้องลบรายการที่มีค่านัลทิ้งไป

3. INDEX ใช้สำหรับคอลัมน์ที่อนุญาตให้เก็บข้อมูลที่ซ้ำกัน และสามารถเก็บค่านัลลงไปได้ อินเด็กซ์ 3 ประเภทแรก ใช้กับคอลัมน์ที่เก็บข้อมูลประเภทตัวเลข วันที่ เวลา หรือสตริง แต่ในกรณีของสตริงควรจะเป็นคอลัมน์ที่เก็บข้อมูลไม่เกิน 255 ตัวอักษร คือมีชนิดข้อมูลเป็น TINYTEXT ,CHAR, VARCHAR เท่านั้น สำหรับคอลัมน์ที่มีชนิดข้อมูลเป็น TEXT, MEDIUMTEXT, LONGTEXT ซึ่งเก็บข้อมูลได้เกิน 255 ตัวอักษร

4. FULLTEXT มีคุณสมบัติเหมือนอินเด็กซ์ประเภท INDEX คือใช้สำหรับคอลัมน์ ให้เก็บข้อมูลที่ซ้ำกัน และสามารถเก็บค่านัลลงไปได้ เพียงแต่ว่าควรจะเป็นคอลัมน์ที่สามารถเก็บข้อมูลสตริงเป็นจำนวนมากกล่าวคือคอลัมน์ที่กำหนดชนิดข้อมูลเป็น TEXT , MEDIUMTEXT หรือ LONGTEXT ส่วนชนิดข้อมูล TINYTEXT , CHAR, VARCHAR ก็ใช้สร้างอินเด็กซ์ประเภทนี้ได้เช่นกัน แต่ไม่เหมาะและอินเด็กซ์ประเภทนี้ต้องใช้กับ MySQL เวอร์ชัน 3.2 เป็นต้น

2.5.2.2 การแก้ไขโครงสร้างตาราง

1. การเปลี่ยนชื่อและชนิดข้อมูลของคอลัมน์

ในการจัดการฐานข้อมูลจำเป็นต้องมีการปรับเปลี่ยนแก้ไขโครงสร้างของตารางใหม่เช่น เปลี่ยนชื่อ ชนิดข้อมูลเพื่อให้เก็บข้อมูลได้เยอะขึ้นหรือเพิ่มคอลัมน์ลงไปใหม่ สำหรับ การแก้ไขโครงสร้างตารางส่วนใหญ่แล้ว จะใช้คำสั่ง ALTER TABLE, CHANGE หรือ MODIFY โดยที่คำสั่ง CHANGE สามารถเปลี่ยนทั้งชื่อและชนิดข้อมูลของคอลัมน์พร้อมกัน ในขณะที่คำสั่ง MODIFY เพียงแต่ระบุชื่อคอลัมน์ใหม่ตามหลังชื่อคอลัมน์เดิมและระบุชนิดคอลัมน์ใหม่ตามหลังตามหลังชื่อคอลัมน์ใหม่

2. การเพิ่ม– ลบคอลัมน์

สามารถเพิ่มคอลัมน์ใหม่เข้าไปในตารางหรือลบคอลัมน์ที่มีอยู่ทิ้งได้ โดยใช้คำสั่ง DROP ตามด้วยชื่อ คอลัมน์ที่ต้องการลบ เช่น ALTER TABLE alt DROP I ; เป็นคำสั่งให้ตาราง alt ถูกลบคอลัมน์ I ทิ้งไป ALTER TABLE alt ADD i ;

2.5.2.3 แปลงประเภทของตารางตามการใช้งาน

ตารางที่ใช้เก็บข้อมูลในระบบฐานข้อมูล MySQL มีอยู่หลายประเภท ได้แก่

1. ISAM เป็นตารางรุ่นเก่าที่ใช้เก็บข้อมูลใน MySQL เวอร์ชันแรก ๆ ปัจจุบันไม่เป็นที่นิยมใช้กัน และเปลี่ยนไปใช้ตารางประเภท MyISAM แทน

2. MyISAM เป็นประเภทของตารางที่นิยมใช้เก็บข้อมูลกันมากที่สุด เพราะมีประสิทธิภาพและมีความยืดหยุ่นสูงในการใช้งาน ปกติแล้วเมื่อเราสร้างตารางขึ้นมาในระบบฐานข้อมูล MySQL โดยไม่ได้กำหนดประเภท ตารางนั้นจะถูกกำหนดเป็นประเภท MyISAM โดยอัตโนมัติ

3. HEAP ตารางประเภทนี้นิยมใช้เป็นตารางเก็บข้อมูลชั่วคราว เนื่องจากเป็นตารางในหน่วยความจำซึ่งมีความเร็วในการทำงานมากที่สุด ดังนั้น ข้อเสียของตารางประเภทนี้คือ ถ้า MySQL เกิดหยุดการทำงานย่อมทำให้ข้อมูลทั้งหมดในตารางประเภทนี้สูญหายไป

4. MERGE ตารางประเภทนี้เกิดจากการนำเอาตารางหลายๆ ตารางมารวมกันเสมือนเป็นตารางเดียว ทำให้สามารถจัดการกับข้อมูลที่มีอยู่ในหลายๆ ตารางได้สะดวกขึ้น แต่มีเงื่อนไขจำกัดว่าตารางนำมารวมกันจะต้องเป็นตารางประเภท MyISAM เท่านั้น

ตารางทั้ง 4 ประเภทข้างต้นไม่สนับสนุนการทำงานในลักษณะของทรานแซกชัน (Transaction) ซึ่งผิดกับตารางอีก 2 ประเภท คือ InnoDB กับ BDB ที่สนับสนุนการทำงานในลักษณะของทรานแซกชันทั้งคู่ คือ สามารถใช้คำสั่ง COMMIT เพื่อยืนยันการทำงานของคำสั่งก่อนหน้าหรือใช้คำสั่ง ROLLBACK เพื่อยกเลิกการทำงานของคำสั่งและกู้ข้อมูลกลับคืน ในกรณีที่เกิดความผิดพลาดของตารางประเภท InnoDB กับ BDB มีข้อแตกต่างกันคือ ประเภทแรกใช้ engine ของ InnoDB ส่วนประเภทหลังใช้ engine ของ BerkeleyDB บางครั้งอาจจำเป็นต้องเปลี่ยนประเภทของตารางที่ใช้งานไปเป็นอีกประเภทอื่น เช่น ตารางประเภท ISAM ซึ่งเป็นตารางที่ใช้กันใน MySQL ซึ่งมีข้อจำกัดไม่สามารถเก็บค่านัลลงในคอลัมน์ที่กำหนดเป็นอินเด็กซ์ได้ แต่ถ้าต้องการใช้ตารางที่เป็น ISAM และต้องการเก็บค่านัลในคอลัมน์ที่กำหนดอินเด็กซ์ ดังนั้นจึงควรเปลี่ยนมาเป็นตารางประเภท MyISAM

2.5.2.4 การสร้างเลขลำดับ

ข้อมูลแต่ละรายการในฐานข้อมูลควรจะมีการกำหนดรหัสหรือเลขที่ประจำรายการกล่าวคือ การกำหนดเลขลำดับ ซึ่ง MySQL มีความสามารถในการกำหนดให้โดยอัตโนมัติ ในระบบฐานข้อมูลนิยมใช้เลขลำดับ เช่น เก็บเลขลำดับข้อมูลแต่ละรายการใช้สำหรับสร้างเป็นรหัสเฉพาะของข้อมูล เช่น รหัสสินค้า รหัสนักศึกษา รหัสพนักงาน

1. นับเลขอัตโนมัติด้วย AUTO_INCREMENT

สมมติว่าจะทำการสร้างฐานข้อมูลตามคำสั่งดังนี้

CREATE TABLE employee(id INT UNSIGNED NOT NULL AUTO_INCREMENT name VARCHAR(50) NOT NULL PRIMARY KEY(id));

AUTO_INCREMENT เป็นการกำหนดให้สร้างเลขลำดับในคอลัมน์ id โดยอัตโนมัติ โดยเริ่มตั้งแต่เลข 1 และนับไปเรื่อย ๆ ตามจำนวนรายการที่มีอยู่ และเนื่องจากคอลัมน์ใดก็ตามที่กำหนดคุณสมบัติเป็น AUTO_INCREMENT จะไม่สามารถเก็บค่านัล ได้ ดังนั้นจึงกำหนดให้คอลัมน์ id เป็นไม่นัล

UNSIGNED เป็นการกำหนดให้ใช้ตัวเลขเฉพาะค่าบวก สาเหตุที่ต้องกำหนดคุณสมบัตินี้ เพราะข้อมูลชนิด INT ใน MYSQL สามารถเก็บค่าได้ทั้งค่าบวกและค่าลบตั้งแต่ –2,147,483,648 ถึง 2,147,483,647 แต่เลขลำดับจะใช้เฉพาะค่าบวกเท่านั้น และเมื่อกำหนดให้เป็น UNSIGNED มีข้อดีคือ ทำให้คอลัมน์ id เก็บข้อมูลได้มากกว่าเดิมเป็น 2 เท่า เพราะสามารถนับเลขลำดับตั้งแต่ 1 จนถึง 4,294,967,295

2. กำหนดเลขลำดับเองโดยไม่สนใจลำดับที่ถูกต้อง

การเพิ่มรายการใหม่ โดยไม่ต้องการใช้เลขลำดับที่ คุณสมบัติ AUTO_INCREMENT ของตารางเพิ่มให้เช่น ลำดับสูงสุดที่มี คือ 9 แต่ต้องการเพิ่มรายการใหม่ให้มีลำดับเป็น 15 สามารถทำได้ โดยระบุค่าของ id เป็น 15 ในคำสั่ง INSERT เลยแทนที่จะระบุเป็นค่านัล

แต่วิธีการนี้มีข้อเสีย เมื่อทำการเพิ่มรายการใหม่ตารางที่กำหนด AUTO_INCREMENT จะเพิ่มลำดับจากค่าที่มากที่สุดในตารางคือกลายเป็น 16 โดยไม่ย้อนกลับไปใช้ค่าเลขลำดับที่ผ่านมาแต่ยังไม่ได้ใช้ คือ 10-14 สามารถกลับไปใช้เลขเหล่านี้ได้โดยวิธีการเดียวกัน

เลขลำดับที่มีอยู่แล้วในตารางจะไม่สามารถกำหนดซ้ำอีกได้ เพราะคอลัมน์ id ถูกกำหนดให้เป็น PRIMARY KEY ถ้าแทรกเลขลำดับซ้ำจะเกิดการทำงานที่ไม่สมบูรณ์

3. การเพิ่มค่าในการเก็บเลขลำดับ

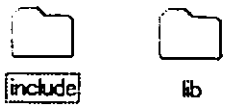
การใช้งานฐานข้อมูลหากมีผู้เข้ามาใช้บริการเป็นจำนวนมาก การกำหนดชนิดข้อมูล และขอบเขตของการรองรับอย่างเพียงพอ

2.5.3 การเชื่อมต่อ MySQL ด้วย VisualC++

การนำข้อมูลเข้าออกจากฐานข้อมูล MySQL นั้นใช้ภาษา SQL ถ้าต้องการเชื่อมต่อกับ MySQL ผ่านทางภาษา C++ โดยใช้ Visual C++ ต้องใช้ MySQL++ ซึ่งเป็น C-API มีฟังก์ชันช่วยให้ติดต่อกับฐานข้อมูลได้ โดยต้องติดตั้งตัว header and library ของ MySQL คือ h กับ .lib และ .dll ลงเครื่องคอมพิวเตอร์

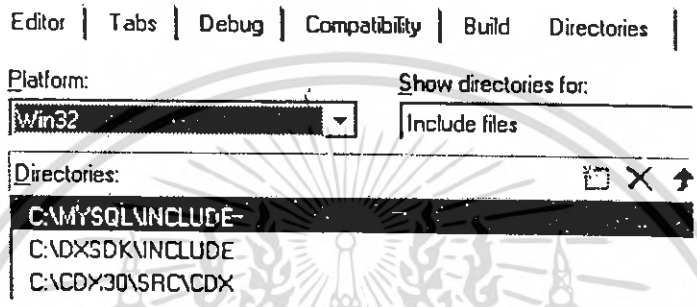
2.5.3.1 การติดตั้ง

ให้ unzip ไฟล์ MySQLlib.zip มาไว้ที่ c:\MySQL



รูปที่ 2.13 แสดงโฟลเดอร์ include และโฟลเดอร์ lib

จากนั้นให้ config ใน Visual C++ ให้รู้จักกับชุดคำสั่งนี้ ไปที่ Tool -> Options ปรับตัวเลือกของ โฟล์เดอร์ ในช่อง include และ library



รูปที่ 2.14 การกำหนดตำแหน่งไฟล์ include

สำหรับ library ก็จะเซตดังนี้

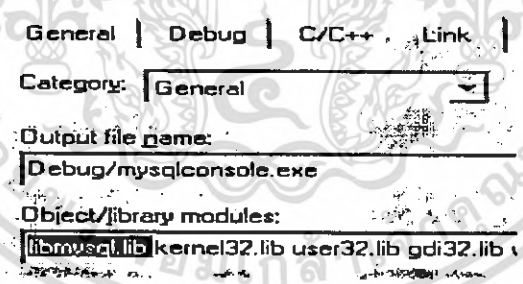
รูปที่ 2.15 การกำหนดตำแหน่งไฟล์ Library

ต่อไปเป็นการทดสอบการเชื่อมต่อ ซึ่งจะเหมือนกับการใช้ PHP กับ MySQL ให้สร้าง project ใน Visual C แบบ Dialog Based หรือแบบ Console ก็ได้ เพื่อทดสอบ ตัวอย่าง

```
#include <stdio.h>
#include <MySQL.h>
void main()
{
```

```
MYSQL *link;
MYSQL_RES *result;
MYSQL_ROW row;
        int numofrows;
        link = MySQL_init(NULL);
        MySQL_real_connect(link,"localhost","","","game",0,NULL,0);
 MySQL_query(link,"SELECT * FROM playerinfo");
 result = MySQL_store_result(link);
        numofrows = MySQL_num_rows(result);
        printf("Num of result = %d\n",numofrows);
        while( row = MySQL_fetch_row(result) )
 {
        printf("%s %s %s %s \n",row[0],row[1],row[2],row[3]);
 }
 MySQL_close(link);
}
```

ก่อนจะสร้างให้เพิ่มชื่อ library เข้าไปในโปรเจ็กต์ก่อนโดยไปที่ Project -> Setting ไปที่แผ่น Link เพิ่มไฟล์ชื่อ libMySQL.lib เข้าไป จากนั้นกดปุ่ม OK



รูปที่ 2.16 การเพิ่มไฟล์ชื่อ libMySQL.lib

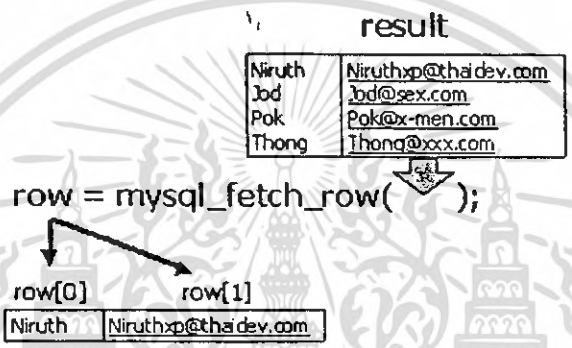
หลังจากสร้างโปรเจ็กต์ให้คัดลอกไฟล์ libMySQL.dll จากโฟลเดอร์ c:\MySQL\lib มาไว้ในโฟลเดอร์เดียวกับโปรเจ็กต์ก่อนที่จะเริ่มการทำงานของโปรแกรม .exe ที่ได้นี้ จะต้องเรียกใช้คำสั่งจาก .dll โดยตัวแปรแต่ละมีหน้าที่ดังนี้

link เป็นตัวแปรพอยเตอร์ ที่จะบอกว่าตอนนี้เปิด MySQLได้หรือไม่

result เป็นตัวแปร pointer เหมือนกันจะเก็บผลของการ Query เอาไว้

row เป็นตัวแปร MYSQL_ROW จะใช้เก็บผลที่ได้ทีละแถวในเวลาใช้งาน จะใช้ทีละอย่าง เริ่มต้น เรียก link=MySQL_init(NULL) และเก็บผลที่ได้ ไว้ใน link จากนั้นทำการ connect ด้วย MySQL_real_connect(link ," ชื่อโฮสต์","username","password","ชื่อ db",0,NULL,0); เสร็จแล้ว เมื่อติดต่อได้ก็ ทำการQuery เช่นต้องการให้ Dump ข้อมูลโดยใช้ "select * from playerinfo" จะแสดงข้อมูลจากตาราง playerinfo ใน db ที่ชื่อว่า game ก็สั่ง MySQL_query(link,"SELECT * FROM playerinfo"); และการได้ผลการ Query ให้ทำการเรียกใช้ฟังก์ชันดังนี้ result = MySQL_store_result(link); MySQL_store_result จะเอาผลที่ได้จาก link มาเก็บไว้ที่ตัวแปร result เมื่อนำไปใช้ต้องนำออกมาทีละแถว โดยใช้คำสั่ง row = MySQL_fetch_row(result);

คำสั่ง fetch row นี้จะถอดทีละแถว ๆ ที่เป็นผลลัพธ์มาไว้ที่ row ซึ่งเป็นอะเรย์ตามลำดับ 0, 1, 2, 3, 4,.... ไป เรื่อย ๆ ดังรูปที่ 2.17



รูปที่ 2.17 แสดงการทำงานของฟังก์ชัน fetch row

ถ้า query มาเป็นตัวแปร result พอ fetch row 1 ครั้ง ก็จะได้ row[0] กับ row[1] ที่เก็บค่าในช่องแรกเอาไว้ แต่ถ้า fetch row อีก ก็จะแสดงข้อมูลใน record ต่อไปของ result ออกมา ดังนั้นจึงต้องใช้ while เข้ามาช่วยในการวนเพื่อให้วนจนครบผลของ result ทั้งหมด

ดังนั้น ถ้าต้องการแสดงผลออกมาทีละ ฟิลด์ สามารถเขียนได้ดังนี้

```
while( row = MySQL_fetch_row(result) )
{
  printf("%s %s %s %s \n",row[0],row[1],row[2],row[3]);
}
```

2.6 PHP (Personal Home Pages)

คือ ภาษาสคริปต์ที่ถูกใช้ในเว็บเพจที่สร้างด้วยภาษา HTML โดยเว็บเพจที่มีสคริปต์ PHP แทรกอยู่นั้นจะทำงานที่เว็บเซิร์ฟเวอร์ PHP มีความสามารถพื้นฐาน เช่น การรับข้อมูลจากแบบฟอร์ม รับส่งเพื่อแลกเปลี่ยนข้อมูลระหว่างผู้ใช้งานกับเซิร์ฟเวอร์ สามารถรองรับฐานข้อมูลได้อย่างกว้างขวาง ทำให้ง่ายต่อการเขียนโฮมเพจ และ

เชื่อมต่อฐานข้อมูลรูปแบบต่างๆได้ง่าย สำหรับฐานข้อมูล PHP และฐานข้อมูลที่สามารถใช้ได้ ได้แก่ Access , dBase , Oracle , MySQL , SQL Server เป็นต้น

2.7 APACHE Web server ให้บริการฐานข้อมูลแก่โทรศัพท์เคลื่อนที่

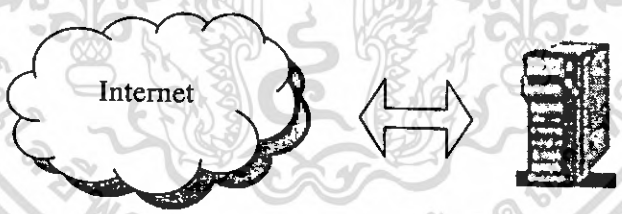
Apache คือโปรแกรมเว็บเซิร์ฟเวอร์ใช้กันมากในลินุกซ์หรือยูนิกซ์ (Unix) และเป็นโอเพ่นซอร์สที่ดีมาก สำหรับข้อมูลการใช้ apache ในวินโดวส์ (Windows) รุ่นต่าง ๆ และ FAQs ทาง apache.org ได้เขียนไว้อย่างละเอียด ส่วนการใช้ apache ใน Windows XP จะต้องติดตั้ง Service Pack 1 และ Service Pack 2 จึงจะใช้ apache ได้ หลังติดตั้งจะทำให้เครื่องเป็นเว็บเซิร์ฟเวอร์สามารถทดสอบเปิด http://localhost หรือ http://127.0.0.1

2.7.1 AppServ คือ โปรแกรมที่รวบรวมโอเพ่นซอร์สซอฟต์แวร์หลาย ๆ อย่างเข้าด้วยกัน สำหรับวินโดวส์ พัฒนาโดย Phanupong Panyadee (apples@chek.com) ในส่วนรายละเอียดเกี่ยวกับโปรแกรมนี้สามารถหาได้ทั่วไปเช่น ใน www.appservnetwork.com ใน AppServ ประกอบด้วย

ซึ่งในโครงงานนี้ได้ใช้ Appserv ในการติดตั้ง AppServ คือโปรแกรมที่รวบรวมโอเพ่นซอร์สซอฟต์แวร์หลาย ๆ อย่างเข้าด้วยกัน สำหรับติดตั้งให้ วินโดวส์พัฒนาโดย Phanupong Panyadee (apples@chek.com)

Appserv ประกอบด้วยโอเพ่นซอร์สหลายตัวเช่น

- Apache : เว็บเซิร์ฟเวอร์(apache.org)
- PHP : สคริปต์ คอมไพเลอร์(php.net)
- MySQL : ระบบจัดการฐานข้อมูล(mysql.com)
- PHP MyAdmin : ระบบช่วยดูแลฐานข้อมูล(phpmyadmin.sourceforge.net)



รูปที่ 2.18 การเชื่อมต่อของเว็บเซิร์ฟเวอร์กับอินเตอร์เน็ต

2.8 โทรศัพท์เคลื่อนที่เซลลูลาร์

ระบบเซลลูลาร์เริ่มพัฒนาขึ้นใช้งานในปี ค.ศ. 1983 ต่อมาในปี ค.ศ. 1990 กลุ่มผู้พัฒนาระบบเซลลูลาร์ได้พัฒนามาตรฐานใหม่โดยให้ชื่อว่า ระบบจีเอสเอ็ม (Global system for mobile communication, GSM) โดยเน้นระบบเชื่อมโยงติดต่อกันได้ทั่วโลก ในสหรัฐอเมริกาก็มีการพัฒนาระบบขึ้นมาใช้ในปี ค.ศ. 1991 โดยให้ชื่อว่า

IS - 54 - Interim Standard - 54 ในปี ค.ศ. 1993 ก็ได้พัฒนาต่อเป็นระบบ IS-95 โดยใช้ระบบ CDMA ที่มีช่องความถี่มากขึ้น ซึ่งเป็นระบบที่ใช้ร่วมกับระบบ AMPS เดิมได้

พัฒนาการของโทรศัพท์แบบเซลลูลาร์แบ่งออกเป็นยุคของการพัฒนาเทคโนโลยีได้ดังนี้

ยุค 1G เป็นยุคแรกของการพัฒนาระบบโทรศัพท์แบบเซลลูลาร์ การรับส่งสัญญาณใช้วิธีการมอดูเลตสัญญาณอะนาล็อกเข้าช่องสื่อสารโดยใช้การแบ่งความถี่ออกมาเป็นช่องเล็ก ๆ ด้วยวิธีการนี้มีข้อจำกัดในเรื่องจำนวนช่องสัญญาณและการใช้ไม่เต็มประสิทธิภาพ จึงติดขัดเรื่องการขยายจำนวนเลขหมาย การขยายแถบความถี่ระบบเครื่องรับส่งสัญญาณวิทยุกำหนดขนาดของเซล และความแรงของสัญญาณเพื่อให้เข้าถึงสถานีเบสได้ ตัวเครื่องโทรศัพท์เซลลูลาร์ยังมีขนาดใหญ่ ใช้กำลังงานไฟฟ้ามาก ในภายหลังจึงเปลี่ยนมาเป็นระบบดิจิตอล และการเข้าช่องสัญญาณแบบแบ่งเวลา โทรศัพท์เคลื่อนที่แบบ 1G จึงใช้เฉพาะในยุคแรกเท่านั้น

ยุค 2G เป็นยุคที่พัฒนาต่อมาโดยการเข้ารหัสสัญญาณเสียง โดยบีบอัดสัญญาณเสียงในรูปแบบดิจิตอลให้มีขนาดจำนวนข้อมูลน้อยลงเหลือเพียงประมาณ 9 กิโลบิตต่อวินาทีต่อช่องสัญญาณ การติดต่อจากสถานีลูก หรือตัวโทรศัพท์เคลื่อนที่กับสถานีเบส ใช้วิธีการ 2 แบบคือ TDMA คือการแบ่งช่องเวลาออกเป็นช่องเล็ก ๆ และแบ่งกันใช้ ทำให้ใช้ช่องสัญญาณความถี่วิทยุได้เพิ่มขึ้นจากเดิมอีกมาก กับอีกแบบหนึ่งเป็นการแบ่งการเข้าถึงตามการเข้ารหัสและการถอดรหัสโดยใส่แอดเดรสเหมือนไอพีเราเรียกวิธีการนี้ว่า CDMA ในยุค 2G จึงเป็นการรับส่งสัญญาณโทรศัพท์แบบดิจิตอลหมดแล้ว

ยุค 3G เป็นยุคแห่งอนาคตอันใกล้ โดยสร้างระบบใหม่ให้รองรับระบบเก่าได้ โดยการเข้าถึงเครือข่ายแบบไร้สายสามารถกระทำได้ด้วยอุปกรณ์หลากหลาย เช่น จากคอมพิวเตอร์ จากเครื่องใช้ไฟฟ้า ระบบยังคงใช้การเข้าช่องสัญญาณเป็นแบบ CDMA ซึ่งสามารถบรรจุช่องสัญญาณเสียงได้มากกว่า แต่ใช้แบบแถบกว้าง (wideband) ในระบบนี้จึงเรียกอีกอย่างหนึ่งว่า WCDMA

นอกจากนี้ยังมีกลุ่มบริษัทได้พัฒนาในรุ่น 3G เป็นแบบ CDMA แต่เรียกว่า CDMA2000 กลุ่มบริษัทนี้พัฒนารากฐานมาจาก IS95 ซึ่งใช้ในสหรัฐอเมริกาและยังขยายรูปแบบเป็นการรับส่งในช่องสัญญาณที่ได้อัตราการรับส่งสูง (High data rate, HDR) การพัฒนาในยุค 3G ต้องการใช้งานร่วมระหว่างเทคโนโลยีทั้งแบบ 1G และ 2G โดยเรียกรูปแบบนี้ว่าจีพีอาร์เอส และได้มีการพัฒนาต่อเป็นระบบอีดีจีอี (Enhanced data rate for GSM evolution, EDGE)

2.9 แวป (Wireless Application Protocol, WAP)

เป็นมาตรฐานสื่อสารซึ่งเป็นผลจากการวางข้อกำหนดระหว่างบริษัทอีริคสัน โนเกีย โมโตโรลาและบริษัทอันไวร์ แพลเน็ต ซึ่งร่วมกันจัดตั้งองค์กรที่มีชื่อเรียกว่า WAP Forum ขึ้นในปี พ.ศ. 2540 ด้วยวัตถุประสงค์ เพื่อพัฒนาแอพพลิเคชันให้สามารถทำงานผ่านการเชื่อมต่อเครือข่ายแบบไร้สาย นอกจากนี้แวบยังมีการประยุกต์ใช้งานเพื่อให้บรรดาผู้ให้บริการระบบเครือข่าย ผู้ผลิตอุปกรณ์สื่อสารและบริษัทที่ให้บริการข่าวสารข้อมูลสามารถสร้างบริการพิเศษให้กับผู้ใช้บริการ โดยมีข้อกำหนดของไมโครบราวเซอร์ (Microbrowser) การเขียนสคริปต์

อิเล็กทรอนิกส์เมล์ บริการแลกเปลี่ยนข่าวสารระหว่างเว็บไซต์กับเครื่องลูกข่ายโทรศัพท์เคลื่อนที่และบริการรับส่งเทเลแฟกซ์ผ่านโทรศัพท์เคลื่อนที่ โดย WAP Forum ได้ พัฒนา Mark-up Language และ Transport Protocol ขึ้นมาใหม่เรียกว่า WML(Wireless Markup Language) ที่ถูกพัฒนามาจาก HTML(Hyper Text Markup Language) ในทำนองเดียวกัน WML Script จะคล้ายกับ Java Script

แวบซึ่งเป็นมาตรฐานเปิดที่ใช้ในการรับส่งข้อมูลบนระบบเครือข่ายโทรศัพท์เคลื่อนที่รูปแบบหนึ่งของเทคโนโลยีอินเตอร์เน็ตในเครือข่ายไร้สาย ซึ่งในการรับส่งข้อมูลในอินเตอร์เน็ตในเครือข่ายไร้สายมีความสามารถจำกัดที่จะประมวลผลข้อมูล ดังนั้นจึงได้ทำการแก้ไขโดย เมื่อต้องการส่งข้อมูลจากเว็บเซิร์ฟเวอร์ไปยังโทรศัพท์มือถือ ข้อมูลนั้นจะถูกส่งผ่านเครือข่ายอินเตอร์เน็ตที่ใช้โพรโตคอล ทีซีพี/ไอพี มาให้แก่ ตัวกลาง ที่เรียกว่าแวบเกทเวย์ ซึ่งทำหน้าที่แปลงข้อมูลไปเป็นอีกรูปแบบหนึ่งก่อนที่จะส่งข้อมูลไปยังโทรศัพท์มือถือ เพื่อให้เหมาะสมกับลักษณะการทำงานและการใช้งานเครือข่ายแบบไร้สาย

2.9.1 องค์ประกอบของการใช้บริการแวบ

บริการแวบจะเกิดขึ้นได้ด้วย 3 ส่วนหลักๆ ดังนี้

1. WAP Phone : หมายถึงเครื่องโทรศัพท์มือถือที่รองรับการใช้งานแวบ คือมีแวบบราวเซอร์ (WAP Browser) ซึ่งใช้สำหรับเปิดดูแวบไซต์ ในปัจจุบันแวบบราวเซอร์ที่ใช้อยู่ในโทรศัพท์มือถือถูกพัฒนาให้สามารถแสดงผลได้ใกล้เคียงหรือเทียบเท่ากับเว็บบราวเซอร์ในเครื่องคอมพิวเตอร์แล้ว

2. WAP Server : เป็นเครื่องคอมพิวเตอร์ประสิทธิภาพสูงเครื่องหนึ่งที่ทำหน้าที่เป็นตัวกลางในการเก็บข้อมูลต่างของแวบไซต์ เช่น เอกสารแวบ เพื่อรอการเรียกดูจากผู้เข้าชม ซึ่งก็สามารถเป็นเครื่องเดียวกันกับเว็บเซิร์ฟเวอร์ ได้

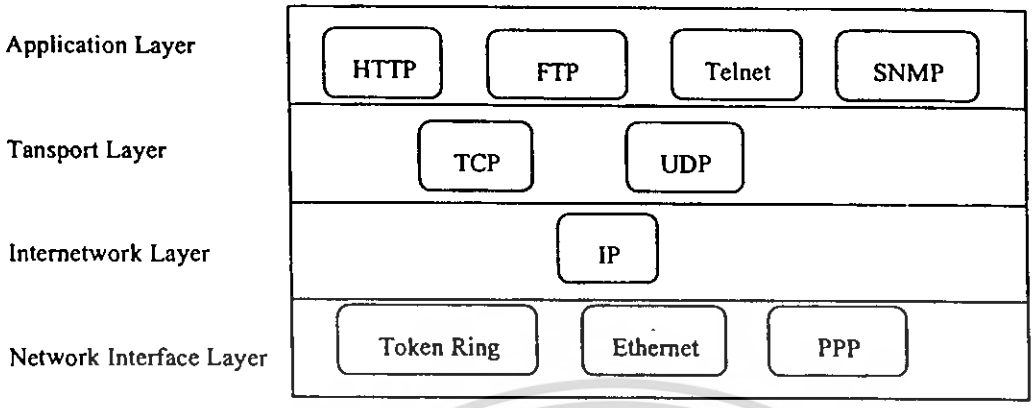
3. แวบเกทเวย์ : ถือเป็นตัวกลางในการเชื่อมต่อของระบบแวบ เนื่องจากแวบเกทเวย์ เป็นเครื่องคอมพิวเตอร์ที่ทำงานเป็นตัวกลาง ระหว่างแวบโฟน (WAP phone) กับแวบเซิร์ฟเวอร์ ซึ่งการทำงานระหว่างโทรศัพท์มือถือ (ทำงานอยู่บนเครือข่ายแบบไร้สาย) และแวบเซิร์ฟเวอร์ (ทำงานอยู่บนเครือข่ายอินเตอร์เน็ต) ดังนั้นจึงต้องอาศัยแวบเกทเวย์เป็นตัวกลาง

2.9.2 WAP บนโครงข่าย GPRS

แวบเป็นโปรโตคอลที่ใช้บนเครือข่ายจีพีอาร์เอส ด้วยเหตุผลบางประการที่ทำให้โปรโตคอลทีซีพี/ไอพีที่ใช้ในโครงข่ายอินเตอร์เน็ตไม่เหมาะที่นำมาใช้บนเครือข่ายของโทรศัพท์มือถือ ซึ่งเป็นข้อจำกัดของโครงข่ายจีพีอาร์เอส ดังนั้นในโครงงานนี้จึงขออธิบายและเปรียบเทียบ เพื่อให้ทราบถึงเหตุผลของการใช้แวบโปรโตคอล

TCP/IP

กฎการสื่อสารแบบทีซีพี/ไอพี ได้กำหนดขั้นตอนในการสื่อสารระหว่างคอมพิวเตอร์ออกเป็น 4 ขั้นตอนหลักๆ โดยแบ่งเป็นชั้นและแต่ละชั้นก็มีโปรโตคอลที่เกี่ยวข้องในขั้นตอนการสื่อสารนั้นๆ เรียกว่า แบบจำลองชั้นสื่อสารทีซีพี/ไอพี



รูปที่ 2.19 แสดงโปรโตคอลในแต่ละชั้น

ในการรับส่งข้อมูลทางระบบอินเตอร์เน็ต โปรโตคอลในแต่ละชั้นจะทำหน้าที่ซึ่งแตกต่างกัน เพื่อให้การรับส่งข้อมูลเป็นไปอย่างสมบูรณ์ ทีซีพีและไอพีเป็นโปรโตคอลคนละตัว แต่เนื่องการใช้งานจะต้องสนับสนุนซึ่งกันและกัน ดังนั้นจึงเรียกรวมกันว่าทีซีพี/ไอพี

ชั้น Application Layer

ในชั้นนี้มีโปรโตคอลที่เกี่ยวข้องกับการใช้งานต่างๆ เช่น

- โปรโตคอล HTTP เป็นข้อกำหนดเกี่ยวกับการรับส่งข้อมูลหรือไฟล์ เพื่อเรียกดูข้อมูลในระบบ WWW โดยเฉพาะอย่างยิ่งระหว่างเว็บบราวเซอร์ (ไคลเอนต์) กับเซิร์ฟเวอร์
- โปรโตคอล FTP เป็นข้อกำหนดเกี่ยวกับการถ่ายโอนข้อมูลจากเว็บเซิร์ฟเวอร์ มายังลูกข่าย (ไคลเอนต์)
- โปรโตคอล Telnet เป็นข้อกำหนดเกี่ยวกับการเข้าใช้งานเซิร์ฟเวอร์ผ่านทางคอมพิวเตอร์ที่เป็นไคลเอนต์ในระยะไกล โดยอาศัยโปรแกรม Telnet

ชั้น Tansport Layer

ในชั้นนี้มีโปรโตคอลที่เกี่ยวข้องอยู่ 2 ตัว คือทีซีพีและยูดีพี (User Datagram Protocol, UDP) ซึ่งทำหน้าที่ควบคุมการรับส่งข้อมูลระหว่างส่งและฝั่งรับ เช่น ควบคุมความเร็วในการรับส่งข้อมูล

- TCP (Transmission Control Protocol) จะควบคุมการส่งข้อมูลระหว่างฝั่งส่งและฝั่งรับ โดยการติดต่อระหว่างทั้ง 2 ฝั่ง จำเป็นต้องสร้างการเชื่อมต่อและให้ฝ่ายรับพร้อมที่จะรับข้อมูลถึงจะเริ่มรับส่งข้อมูลกันได้
- ยูดีพีมีหน้าที่ควบคุมการติดต่อสื่อสารข้อมูล ซึ่งจะไม่มีการสร้างการเชื่อมต่อระหว่างผู้รับและผู้ส่งก่อน ฝ่ายรับไม่จำเป็นต้องรับรู้ว่าจะมีข้อมูลส่งมาและไม่มีการตรวจสอบ ส่งข้อมูลได้เร็วจึงเหมาะกับการส่งข้อมูลจำพวกมัลติมีเดียในลักษณะ streaming เช่น เพลง ภาพแอนิเมชัน ฯลฯ

ชั้น Internetwork Layer

เป็นชั้นสื่อสารที่มีโปรโตคอลหลัก คือ ไอพี(Internet Protocol) ซึ่งทำหน้าที่ในการหาเส้นทางที่เหมาะสมสำหรับการส่งข้อมูลแต่ละหน่วยย่อยๆลักษณะการทำงานก็จะเป็นแบบ Packing-Switching คือ แต่ละแพ็กเก็ตจะเดินทางจากจุดเริ่มต้นผ่านไปยังเครือข่ายต่างๆ ในระบบอินเตอร์เน็ตได้อย่างมีอิสระจนถึงจุดหมายปลายทางโดยไม่จำเป็นต้องไปเส้นทางเดียวกันเสมอและลำดับของแพ็กเก็ตที่ไปถึงปลายทางก็ไม่สำคัญ

ชั้น Network Interface Layer

เป็นชั้นที่รับข้อมูลมาจากชั้น Internetwork Layer โดยปรับแพ็กเก็ตให้อยู่ในรูปที่เหมาะสมกับการส่งไปในเครือข่ายหรือสื่อสาร เช่น ปรับสัญญาณทางไฟฟ้าเพื่อส่งไปในสายสื่อสาร เป็นต้น หากเป็นฝ่ายรับก็จะรับข้อมูลมาจากสายสื่อสารแล้วปรับรูปแบบของข้อมูลเพื่อส่งขึ้นไปยัง ชั้น Internetwork Layer ต่อไป

เทคโนโลยีอินเตอร์เน็ตพัฒนาขึ้นตามความสามารถในการประมวลผลด้วยคอมพิวเตอร์ แต่เมื่อจะเปลี่ยนรูปแบบการใช้งานมาเป็นโทรศัพท์มือถือ ย่อมมีข้อจำกัดกว่าเครื่องคอมพิวเตอร์ซึ่ง ดังนี้

- ซีพียูมีพลังในการประมวลผลด้อยกว่า
- มีหน่วยความจำน้อยกว่า(ทั้งหน่วยความจำแรมและรอม)
- มีข้อจำกัดในด้านแหล่งจ่ายพลังงาน เมื่อเปรียบเทียบกับคอมพิวเตอร์ซึ่งใช้กระแสไฟฟ้าและสามารถทำงานได้ตลอดเวลาที่เสียบปลั๊ก
- หน้าจอแสดงผลมีพื้นที่เล็กกว่า
- การใช้งานด้วยปุ่มกดบนตัวโทรศัพท์ไม่สะดวกเมื่อเทียบกับการใช้แป้นคีย์บอร์ด(key board) หรือเมาส์(mouse)

นอกจากนี้ ในส่วนของเครือข่ายไร้สาย ก็ยังมีข้อจำกัดเมื่อเทียบกับเครือข่ายอินเตอร์เน็ต ดังนี้

- มีแถบความถี่ช่วงความถี่ที่ใช้งานแคบกว่า
- ระยะเวลาในการส่งข้อมูลผ่านเครือข่ายมากกว่า
- สภาพการเชื่อมต่อมีความเสถียรต่ำกว่า

จากข้อจำกัดดังกล่าวสรุปว่า ข้อมูลในอินเตอร์เน็ตไม่สามารถรับส่งได้ดีในเครือข่ายไร้สาย เนื่องมาจากโปรโตคอลทีซีพีทำงานได้ไม่ดีในเครือข่ายไร้สาย ในการแก้ไขเมื่อต้องการส่งข้อมูลจากเว็บเซิร์ฟเวอร์ไปยังมือถือ ข้อมูลนั้นจะถูกส่งผ่านเครือข่ายอินเตอร์เน็ตที่ใช้โปรโตคอลทีซีพี/ไอพีเป็นโปรโตคอลหลักมาให้แก่ตัวกลางซึ่งทำหน้าที่แปลงข้อมูลไปเป็นอีกรูปแบบหนึ่ง ก่อนที่จะส่งต่อไปยังโทรศัพท์มือถือ เพื่อให้ข้อมูลนั้นอยู่ในลักษณะที่เหมาะกับการรับส่งผ่านเครือข่ายไร้สายและเหมาะสมกับความสามารถในการประมวลผลของโทรศัพท์มือถือและตัวกลางที่ทำหน้าที่แปลงข้อมูลมีชื่อ เรียกว่า แวบเกทเวย์

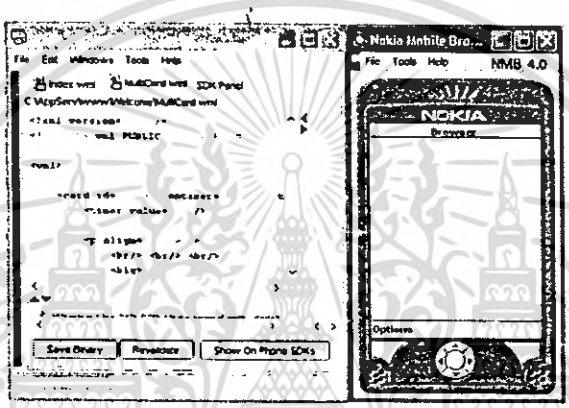
2.9.3 การเลือกเทคโนโลยี

เครื่องมือสำหรับพัฒนา

1. Wap Toolkit

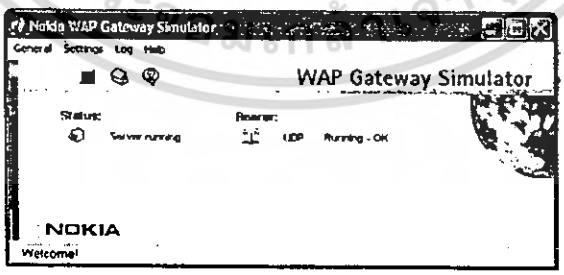
ประกอบด้วยแวบเอดิเตอร์(Wap Editor) สำหรับช่วยเขียนโค้ดให้ง่ายขึ้นและแวบบราวเซอร์ ซึ่งเป็นแวบอีมูเลเตอร์(Wap Emulator) ในตัวคือจำลองการทำงานเอกสารแวบหรือเว็บของอุปกรณ์สื่อสาร ซึ่งมีอยู่หลายค่ายโทรศัพท์ เช่น อีริกสัน , โมโตโรลา สำหรับในโครงงานนี้ใช้ของทูลแวบของโนเกีย(Nokia Wap Tookit) ซึ่งหาดาวน์โหลดได้จากเว็บไซต์ของบริษัทผู้ผลิต

เครื่องมือหลักๆ ที่มีในทูลของโนเกียประกอบด้วย



รูปที่ 2.20 แวบเอดิเตอร์และแวบบราวเซอร์

การใช้แวบเอดิเตอร์ก็ไม่ค่อยมีความจำเป็น เพียงใช้โปรแกรมจำพวกเท็กเอดิเตอร์(Text Editor) เช่น โน้ตแพ็ด (Notepad) ก็เพียงพอแล้ว เพียงแต่เครื่องมือพวกนี้ช่วยในการจัดสคริปต์ให้ดูง่ายขึ้น แต่สำหรับทูลของโนเกียจะรวมเอาการจัดการกับแวบบราวเซอร์มาให้ด้วย



รูปที่ 2.21 แวบเกทเวย์ซิมมูเลเตอร์

สำหรับการดูแวบไซต์ไม่ต้องใช้แวบเกทเวย์ก็ได้ แต่หากเป็นการทดลองแวบแอพพลิเคชันที่มีการใช้ CGI ก็ควรจะทดลองใช้แวบเกทเวย์ด้วย โดยให้กำหนดไอพีแอดเดรสของแวบเกทเวย์แล้วเรียกแวบไซต์ หากไม่มีอะไรมาแสดงที่แวบบราวเซอร์ ก็สันนิษฐานได้ว่าอาจจะระบุไอพีแอดเดรสของแวบเกทเวย์ผิดหรือมิฉะนั้นแวบเกทเวย์ก็อาจมีปัญหาหรือปิดให้บริการแล้ว

รูปที่ 2.22 การจำลองการเชื่อมต่อของแวบบราวเซอร์เข้าสู่เครือข่ายอินเตอร์เน็ต

เมื่อเราทำการติดตั้งเครื่องมือในการพัฒนาแล้วก็เป็นการจำลองการเชื่อมต่อผ่านเครือข่ายอินเตอร์เน็ตของอุปกรณ์สื่อสารในเครื่องพีซี

2.10 WML

แวบใช้ภาษา WML(Wireless Markup Language) ในการกำหนดรูปแบบของข้อมูลในเรื่องของระบบการทำงานของแวบไซต์(เข้าถึงโดยผ่านโทรศัพท์มือถือ) กับเว็บไซต์(เข้าถึงผ่านเครื่องคอมพิวเตอร์)นั้น ค่อนข้างคล้ายกันแต่การสร้างแวบไซต์ขึ้นมานั้นไม่ได้ใช้ภาษา HTML เหมือนกับการสร้างเว็บไซต์ แต่จะใช้ภาษา WML แทน ทั้งนี้เนื่องจากโทรศัพท์มือถือมีข้อจำกัดเรื่องทรัพยากรต่างกับเครื่องคอมพิวเตอร์อย่างมาก เช่น ขนาดของหน่วยความจำภายใน ความรวดเร็วในการประมวลผล ความเร็วในการโหลดข้อมูลและที่สำคัญคือ ขนาดและสีสันของหน้าจอแสดงผลที่ต่างกันอย่างมาก

	แวบไซต์	เว็บไซต์
Page Format	WML	HTML
Dynamic Script	WMLScript	JavaScript
Images&Graphics	WBMP	JPG,GIF
Network Protocal	WSP	HTTP

ตารางที่ 2.4 เปรียบเทียบความแตกต่างระหว่างแวบไซต์และเว็บไซต์

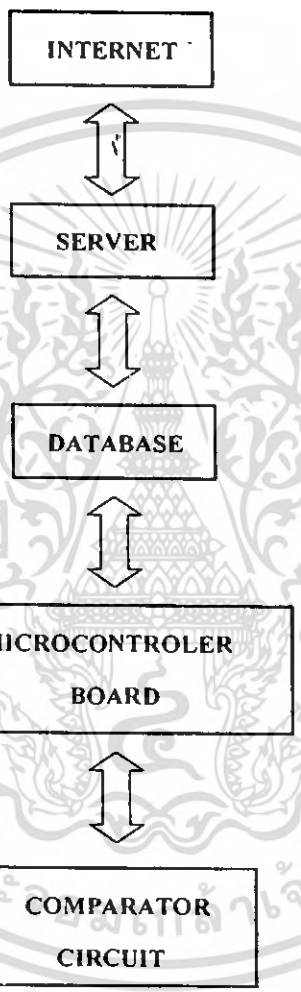
2.11 GPRS (General Packet Radio Service)

จีพีอาร์เอสเป็นโครงข่ายบริการสำหรับโทรศัพท์เคลื่อนที่ที่พัฒนาต่อมาจากยุคที่ 2 ซึ่งแต่เดิมนั้นโทรศัพท์เคลื่อนที่ถูกออกแบบมาเพื่อใช้สำหรับการสื่อสารโดยใช้เสียงเป็นหลักเท่านั้นในระบบ GSM การสื่อสารโดยใช้เสียงนั้นจะต้องทำการเชื่อมต่อกับปลายทางให้ได้ เนื่องจากการพูดโต้ตอบกันนั้นจะเป็นแบบทันทีทันใด เสียงพูดจะไม่มีการขาดหายเป็นช่วงๆและไม่ถูกหน่วงเวลาให้ช้าออกไป แต่สำหรับการสื่อสารเพื่อส่งผ่านข้อมูลนั้น การรับส่งข้อมูลนั้นสามารถแบ่งข้อมูลออกเป็นส่วนๆและส่งข้อมูลออกไปโดยไม่คำนึงถึงช่วงเวลาได้ จึงไม่จำเป็นต้องรับส่งข้อมูลทันทีทันใดตลอดเวลา โดยฝั่งผู้รับสามารถนำข้อมูลมาเรียงต่อกันในภายหลังได้อย่างถูกต้องและไม่จำเป็นต้องมาจองวงจรเชื่อมต่อ การสื่อสารจึงเหมาะสำหรับการสื่อสารที่ใช้ในระบบจีพีอาร์เอส ความเร็วในการรับส่งข้อมูลของจีพีอาร์เอสสูงสุดซึ่งเร็วกว่าการสื่อสารผ่านสายโทรศัพท์ในปัจจุบันถึง 3 เท่า การรับส่งข้อมูลจะทำได้ตลอดเวลาเหมือนคอมพิวเตอร์ที่ต่ออยู่บนแลน (LAN) ทำให้ใช้งานสะดวกทุกที่ทุกเวลาจีพีอาร์เอสใช้โปรโตคอลทีซีพี/ไอพี ในการรับส่งข้อมูลเช่นเดียวกับที่รับส่งกันในอินเตอร์เน็ต โทรศัพท์เคลื่อนที่ที่ใช้บริการจีพีอาร์เอสจะได้รับการกำหนดไอพีแอดเดรส ชั่วคราวจาก GGSN (Gateway GPRS Support Node) เพื่อที่จะสามารถรับส่งข้อมูลกับปลายทางที่ต่างๆบนเครือข่ายไอพี

SGSN (Serving GPRS Support Node) เป็นอุปกรณ์หลักของเครือข่ายโทรศัพท์เคลื่อนที่ในการให้บริการจีพีอาร์เอส โดย SGSN จะทำหน้าที่รับส่งข้อมูลจากเครือข่ายไอพีอื่นๆไปยังเครื่องลูกข่ายจีพีอาร์เอสและยังมีหน้าที่เข้ารหัส ถอดรหัสข้อมูลรวมถึงควบคุมการรับส่งข้อมูลของเครื่องลูกข่ายจีพีอาร์เอสอีกด้วย เมื่อเครื่องลูกข่ายเคลื่อนที่ไปยังเขตบริการจีพีอาร์เอสอื่น SGSN จะโอนย้ายของบริการจีพีอาร์เอสกับอุปกรณ์ชุมสายระบบ GSM ทั้งหมด บริการจีพีอาร์เอสนั้น เมื่อเทียบกับ OSI 7 Layer แล้วจีพีอาร์เอสจะอยู่ Layer 1 ถึง Layer 4 เป็นโครงสร้างพื้นฐานของการรับส่งข้อมูลระหว่างเครื่องโทรศัพท์เคลื่อนที่กับอุปกรณ์ปลายทางอื่นๆบนเครือข่ายไอพีซึ่งโปรแกรมประยุกต์ที่จะนำมาใช้งานร่วมกับจีพีอาร์เอสจะอยู่ใน Layer 5 ถึง Layer 7 ได้แก่ เว็บบราวเซอร์ อีเมล์ แวบ เป็นต้น สิ่งหนึ่งที่ทำให้ Mobile Internet (การใช้งานอินเตอร์เน็ตผ่านโทรศัพท์มือถือ) ในช่วงเริ่มแรกนั้นไม่ได้รับความนิยมจะเป็นเรื่องของความเร็วในการรับส่งข้อมูล ซึ่งเริ่มแรกจะใช้ระบบ CSD (Circuit Switched Data) ที่ให้ความเร็วในการรับส่งข้อมูลสูงสุดเพียง 9.6 Kbps และเสียค่าบริการค่อนข้างสูง ทางผู้ให้บริการจึงได้คิดค้นระบบขึ้นมาให้มีความเร็วเพิ่มขึ้นคือระบบ HSCSD (High-Speed Circuit Switch Data) ที่ให้ความเร็วมากขึ้นเป็น 28.8-57.6 Kbps จนกระทั่งประมาณปลายปี พ.ศ. 2544 มีจีพีอาร์เอสเข้ามาช่วยกระตุ้น Mobile Internet อีกครั้งหนึ่ง ซึ่งจีพีอาร์เอสเป็นเทคโนโลยีประเภทเดียวกันกับ CSD หรือ HSCSD ซึ่งเป็นตัวช่วยในการรับส่งข้อมูลบนเครือข่ายโทรศัพท์มือถือ โดยมีความเร็วที่พัฒนาเพิ่มขึ้นเป็น 40 Kbps (ในทางทฤษฎีอาจจะรับส่งข้อมูลได้เร็วถึง172.2Kbps)

บทที่ 3
การคำนวณและการสร้าง

3.1 หลักการการทำงานของระบบ



รูปที่ 3.1 แสดงบล็อกไดอะแกรมของระบบ

หลักการทำงานของโครงงาน

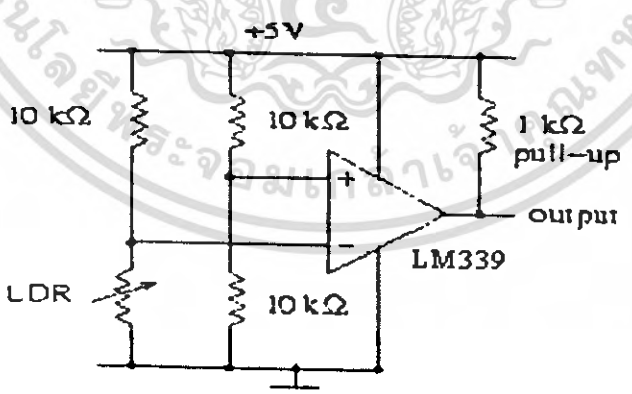
จากบล็อกไดอะแกรมจะเห็นว่าเราสามารถแบ่งการทำงานของโครงงานได้เป็น 6 ส่วนใหญ่ๆโดยที่ทั้ง 6 ส่วนมีการทำงานที่สัมพันธ์กันดังนี้

- บล็อกไดอะแกรมในส่วนของระบบอินเตอร์เน็ต เป็นส่วนที่ให้ผู้ใช้งานสามารถที่จะทราบสถานะลานจอดรถได้โดยผ่านการแสดงผลที่หน้าจอโทรศัพท์มือถือ ซึ่งก่อนที่ผู้ใช้จะเข้าไปตรวจสอบสถานะลานจอดรถจะต้องมีการร้องขอไปยังเครื่องเซิร์ฟเวอร์ จากนั้นเซิร์ฟเวอร์จะทำการแยกแยะผู้ใช้ ว่าควรให้เอกสารเว็บหรือเอกสารแวบมาแสดงยังมือถือของผู้ใช้
- บล็อกไดอะแกรมเครื่องเซิร์ฟเวอร์ มีส่วนที่เป็นแอพพลิเคชันที่คอยดึงข้อมูลจากฐานข้อมูลมาประมวลผลและส่งสัญญานไปบันทึกลงฐานข้อมูล
- บล็อกไดอะแกรมของระบบฐานข้อมูล เป็นส่วนที่ทำหน้าที่นำข้อมูลจากไมโครคอนโทรลเลอร์เพื่อเป็นข้อมูลให้เว็บแอพพลิเคชันทำการดึงข้อมูลไปทำการประมวลผล
- บล็อกไดอะแกรมของไมโครคอนโทรลเลอร์ ทำหน้าที่ในการรับข้อมูลจากวงจรคอมพาราเตอร์โดยรับข้อมูลเข้ามาทางพอร์ต 1 มา 8 บิต (โดยแต่ละบิตจะแสดงสถานะของการจอดรถของรถแต่ละคัน) จากนั้นข้อมูลจะถูกเก็บไว้ในฐานข้อมูล
- บล็อกไดอะแกรมของวงจรเปรียบเทียบแรงดันรับสัญญาณมาจากเซนเซอร์ จากนั้นทำการเปรียบเทียบกับแรงดันไฟฟ้าอ้างอิงเพื่อให้ได้ข้อมูลลอจิก (Logic) ที่แสดงสถานะการจอดรถ

3.2 การออกแบบและการสร้างในส่วนของฮาร์ดแวร์

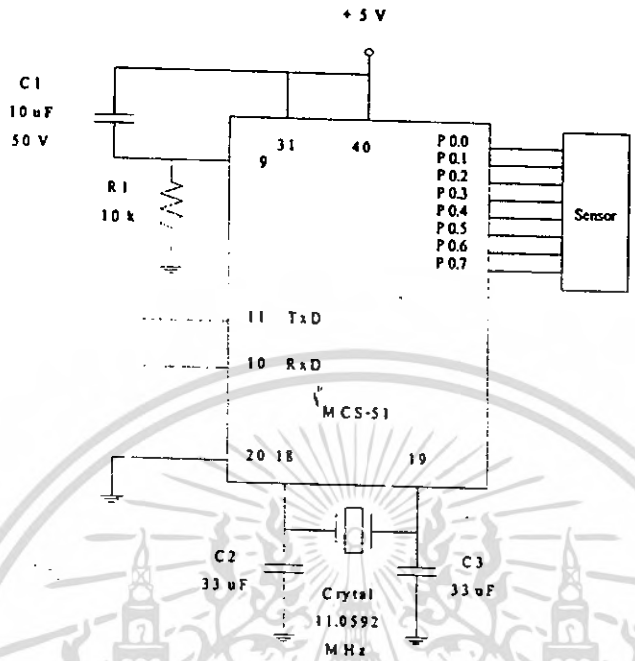
3.2.1 ส่วนของเซนเซอร์แอลดีอาร์และวงจรเปรียบเทียบแรงดันไฟฟ้า

ต่อวงจร ดังรูปที่ 3.2 นำค่าเอาต์พุตที่ได้จากวงจรนี้ส่งไปยังไมโครคอนโทรลเลอร์



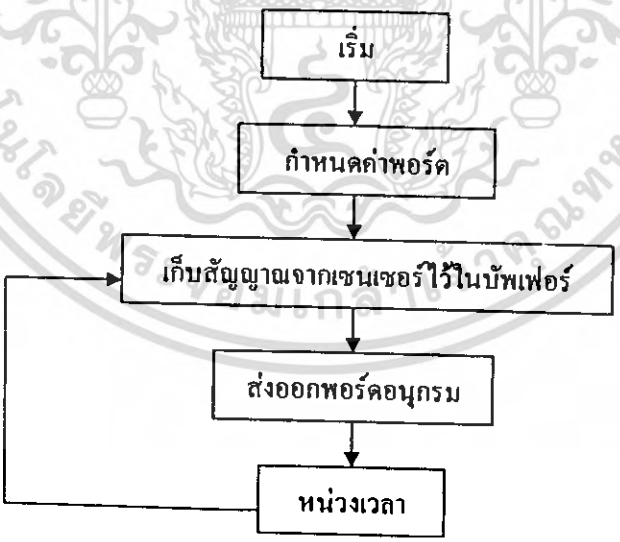
รูปที่ 3.2 วงจรเซนเซอร์แอลดีอาร์และวงจรเปรียบเทียบแรงดันไฟฟ้า

3.2.2 วงจรไมโครคอนโทรลเลอร์



รูปที่ 3.3 วงจรไมโครคอนโทรลเลอร์

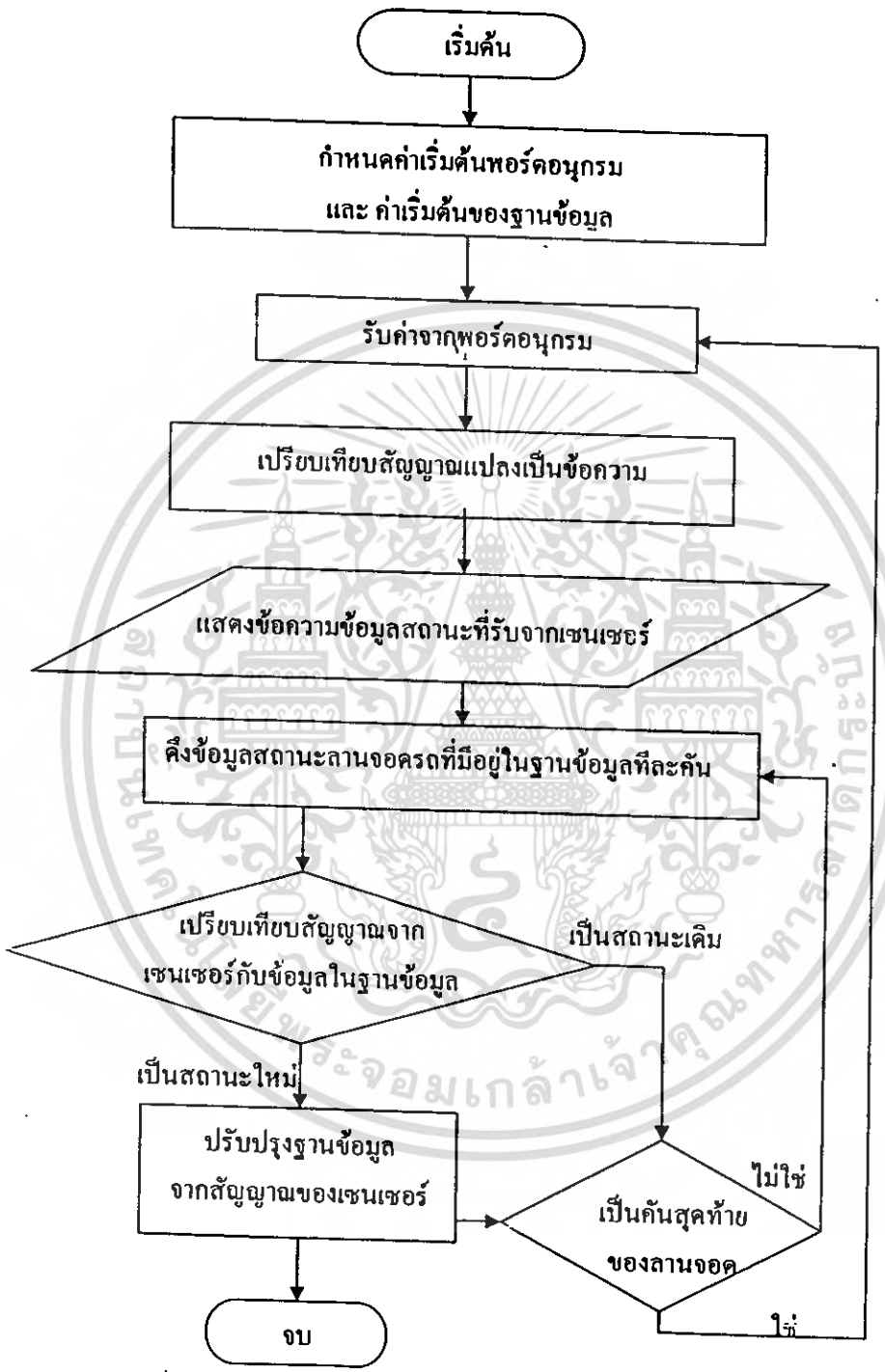
3.2.3 โปรแกรมในไมโครคอนโทรลเลอร์



รูปที่ 3.4 ผังการทำงานโปรแกรมในไมโครคอนโทรลเลอร์

3.3.การออกแบบและการสร้างในส่วนของซอฟต์แวร์

3.3.1 โปรแกรมเชื่อมต่อพอร์ตอนุกรมและฐานข้อมูล



รูปที่ 3.5 ผังการทำงานของโปรแกรมรับสัญญาณจากพอร์ตอนุกรม

ในการออกแบบซอฟต์แวร์เราสามารถใช้ โมเดล Use Case เป็นเครื่องมือที่ใช้ในการตรวจสอบความต้องการของผู้ใช้ถึง คุณสมบัติและฟังก์ชันการทำงานที่ซอฟต์แวร์จะสามารถทำได้ โมเดล Use Case จะเป็นเงื่อนไขหรือข้อกำหนดที่ซอฟต์แวร์จะต้องสามารถทำได้ ดังนั้น โมเดล Use Case จึงเป็นข้อมูลอินพุตที่สำคัญสำหรับขั้นตอนการออกแบบและทดสอบระบบ

3.3.2.บันทึกความต้องการเบื้องต้น

คุณสมบัติ
โปรแกรมส่วนเว็บแอพพลิเคชัน
1. สามารถตรวจสอบเครื่องผู้ใช้ เพื่อแยกการให้บริการแวบ (สำหรับโทรศัพท์รุ่น2G หรือรุ่นที่ใช้ไมโครบราวเซอร์) หรือเว็บ (สำหรับโทรศัพท์รุ่น 3G หรือสมาร์ทโฟนหรือรุ่นที่มีบราวเซอร์) ได้
2. สามารถตรวจสอบชื่อผู้ใช้และรหัสผ่านได้
3. สามารถเพิ่มรายชื่อและข้อมูลอื่นๆ ของผู้ใช้รายใหม่ได้
4. สามารถดูข้อมูลสถานลานจอดรถได้
5. สามารถเปลี่ยนชื่อผู้ใช้บริการและรหัสผ่านได้
6. สามารถจองล่วงหน้าได้(1ชั่วโมง)
โปรแกรมส่วนรับสัญญาณจากฮาร์ดแวร์
1. สามารถติดต่อพอร์ตอนุกรมได้
2. สามารถแสดงผลข้อมูลที่รับเข้ามาจากพอร์ตอนุกรมออกหน้าจอได้
3. สามารถเชื่อมต่อฐานข้อมูลได้
4. สามารถแสดงผลข้อมูลที่เก็บไว้ในฐานข้อมูลออกหน้าจอได้
5. สามารถตรวจสอบสถานะการจองของผู้ใช้เพื่อตัดสินใจการเปลี่ยนแปลงข้อมูลการจอง
6. สามารถตรวจสอบวันเวลาที่มีการเปลี่ยนแปลงสถานะการจอดได้

ตารางที่ 3.1 คุณสมบัติที่ต้องการจากโปรแกรม

ประเภท	ความต้องการ	คำอธิบาย
โปรแกรมส่วนเว็บแอพพลิเคชัน		
เงื่อนไขการพัฒนา	ใช้ภาษา PHP ในการพัฒนา	เนื่องจากเซิร์ฟเวอร์ที่เลือกใช้ เป็น Apache จึงต้องการความเข้ากันได้ และต้องการใช้โอเพ่นซอร์ส
โปรแกรมส่วนรับสัญญาณจากฮาร์ดแวร์		
เงื่อนไขการพัฒนา	ใช้ Visual C++ ในการพัฒนา	เนื่องจากการจับจังหวะสัญญาณที่รับมาจากฮาร์ดแวร์ ใช้เทคนิคการเลื่อนบิตข้อมูล

ตารางที่ 3.2 บันทึกความต้องการเพิ่มเติม

3.3.3. ค้นหา Use Case (Find Actors and Use Case)

ในกิจกรรมของการค้นหาแอคเตอร์และ Use Case นั้นก็เพื่อ

- กำหนดขอบเขตของซอฟต์แวร์
- เป็นการเริ่มต้นค้าหาว่าใครและอะไรที่จะติดต่อกับซอฟต์แวร์และฟังก์ชันอะไรที่จะให้บริการ

สำหรับโครงงานนี้มีวัตถุประสงค์หลักในการสร้างระบบเพื่อให้ผู้ใช้ตรวจดูสภาพลานจอดรถ เพื่อเป็นข้อมูลเบื้องต้นในการตัดสินใจเข้าลานจอดรถ เนื่องจากเป็นระบบสำหรับมีผู้ใช้บริการจำนวนมาก จึงได้เพิ่มในส่วนระบบบริหารสมาชิกเพิ่มเติม ได้แก่ ระบบล็อกอิน ระบบสมัครสมาชิกใหม่ เปลี่ยนชื่อและรหัสผ่าน และเนื่องจากโทรศัพท์เคลื่อนที่ในเมืองไทย ปัจจุบันยังอยู่ในช่วงยุค 2.5 เป็นส่วนมาก กล่าวคือยังมีความจำกัดทั้งทางด้านเครือข่ายและสมรรถนะของตัวเครื่องโทรศัพท์ที่มีความจำกัดในการแสดงผลและประมวลผลจึงได้เพิ่มระบบให้รองรับทั้งแวบและเว็บกล่าวคือ สามารถตรวจสอบ โปรแกรมของเครื่องผู้ใช้เพื่อแยกการบริการให้สอดคล้องตามจริง และในระบบสุดท้ายที่เพิ่มเติมคือ ระบบการจองตำแหน่งที่จอดรถ ซึ่งเป็นส่วนของแนวทางในการนำไปประยุกต์ใช้เพิ่มเติมในงานอื่นๆ เช่น โต๊ะอาหารในภัตตาคารสามารถจัดทำระบบการจองโต๊ะอาหารผ่านอินเตอร์เน็ตบริการให้ลูกค้าได้

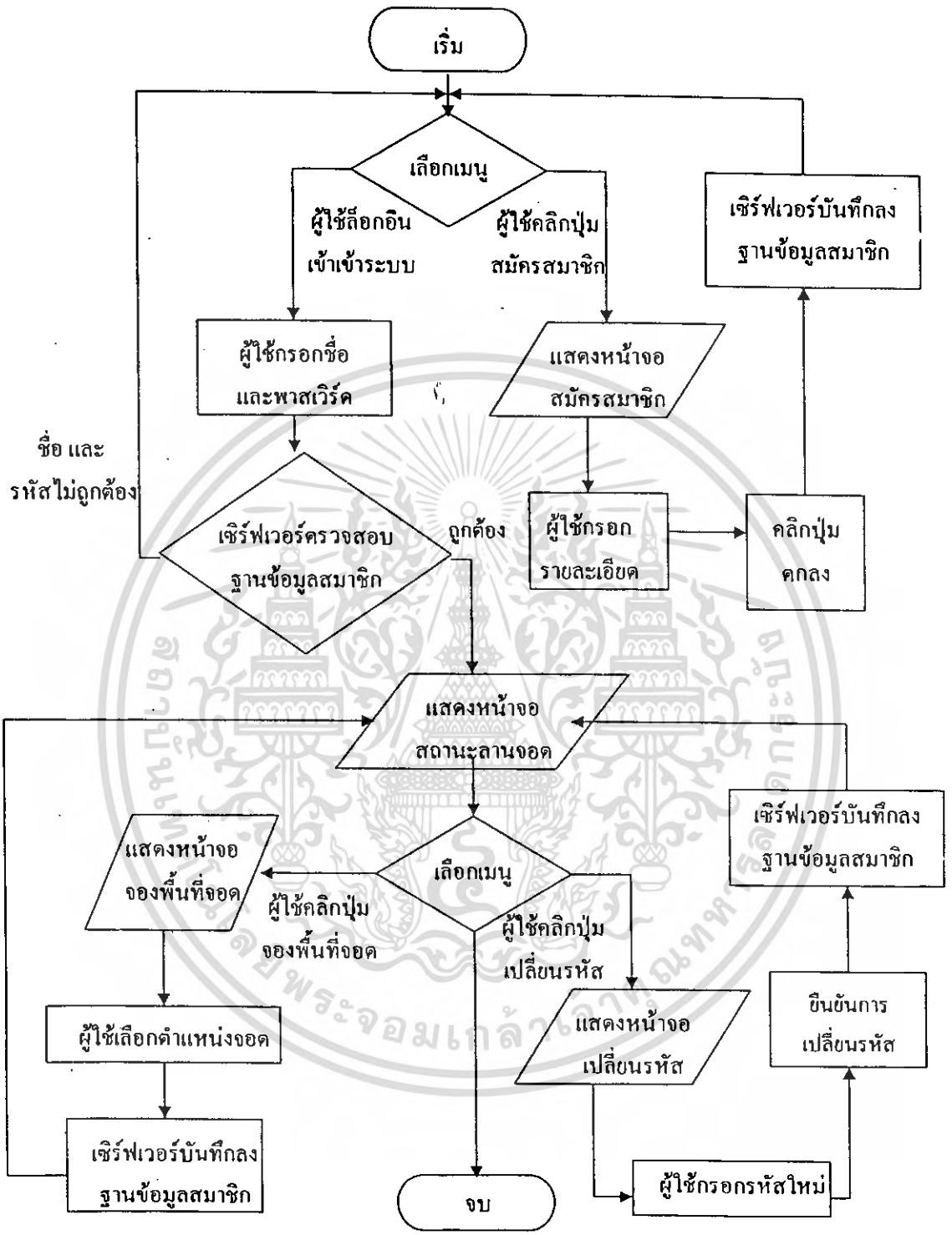
ดังนั้นจึงสรุป ความต้องการของระบบทั้งหมดที่ต้องออกแบบในโครงงานนี้ดังตารางที่ 3.3

Use Case	คำอธิบาย
โปรแกรมส่วน เว็บ และ แวบพลิเคชัน	
ล็อกอิน	ล็อกอินเพื่อเข้าดูรายละเอียด ข้อมูลลานจอดรถ
สมัครสมาชิกใหม่	ลูกค้าสมัครสมาชิกใหม่เพื่อป้อนข้อมูลส่วนตัวลงระบบ
เปลี่ยนชื่อเรียกและรหัสผ่าน	เปลี่ยนชื่อเรียกและรหัสผ่านใหม่ลงฐานข้อมูล
จองที่จอดรถ	ผู้ใช้ระบุตำแหน่งเพื่อส่งบันทึกลงฐานข้อมูล
แยกระบบแวบและเว็บ	ตรวจสอบความสามารถในการใช้งานบราวเซอร์เครื่องฝั่งผู้ใช้เพื่อแยกการให้บริการเอกสารแวบหรือเอกสารเว็บ
ตรวจสอบการกรอกข้อมูล	บังคับให้ผู้ใช้กรอกข้อมูลให้ครบตามที่กำหนด
โปรแกรมส่วน รับข้อมูลจากฮาร์ดแวร์ (C++)	
แสดงการรับสัญญาณ	เพื่อดูการทำงานของพอร์ตอนุกรม
แสดงตารางบันทึกผล	เพื่อดูสถานะลานจอดรถและสถานะการจองของผู้ใช้
แสดงวันเวลาที่ทำการบันทึก	เพื่อดูวันและเวลา ที่มีการเปลี่ยนแปลงข้อมูลล่าสุด
แสดงข้อมูลลานจอดย้อนหลัง	ทำการบันทึกข้อมูลทุกครั้งที่ลานจอดมีการเปลี่ยนแปลง

ตารางที่ 3.3 การค้นหา Use Case และการกำหนดคำอธิบาย

3.4 โปรแกรมให้บริการแวบและเว็บ

เนื่องจากอุปกรณ์สื่อสารเคลื่อนที่ในปัจจุบันได้มีการพัฒนาความสามารถทางด้านฮาร์ดแวร์มากขึ้น ทำให้มีการใช้โปรแกรมประยุกต์ เช่น บราวเซอร์ส่งผลให้สามารถใช้บริการ เอกสารเว็บผ่านอุปกรณ์เหล่านี้ได้ แต่อย่างไรก็ตามยังมีอุปกรณ์ที่มีความจำกัดทางด้านฮาร์ดแวร์ที่ไม่สามารถใช้โปรแกรม บราวเซอร์ได้อย่างเต็มรูปแบบ จึงต้องลดขนาดของตัวบราวเซอร์ลงเพื่อให้ใช้ได้กับอุปกรณ์เหล่านี้เรียกว่าไมโครบราวเซอร์ซึ่งใช้ประมวลผลเอกสารแวบ การโปรแกรมบริการเว็บ(เอกสาร HTML)หรือแวบ(เอกสาร WML) ใช้ PHP และ SQL เพื่อจักการข้อมูลในฐานข้อมูลเหมือนกัน



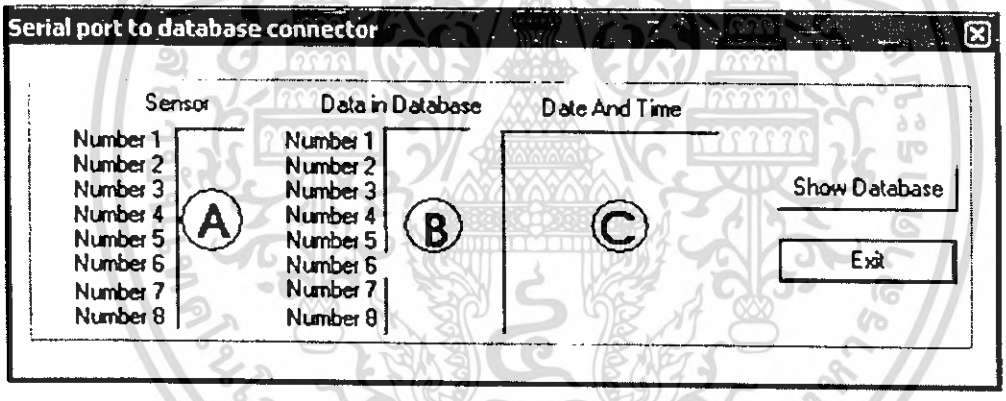
รูปที่ 3.6 ผังการทำงานของเว็บแอพพลิเคชัน

3.4.1 ความต้องการของโปรแกรม

1. สามารถติดต่อพอร์ตอนุกรมเพื่อรับข้อมูลได้
2. สามารถแปลงข้อมูลจากตัวเซนเซอร์ มาแสดงผลให้ผู้ใช้บริการทราบได้อย่างถูกต้อง
3. สามารถติดต่อฐานข้อมูลได้เพื่อบันทึกข้อมูล และแสดงผลสถานะข้อมูลในฐานข้อมูลได้
4. สามารถแสดงเวลาที่สถานะลานจอดรถเปลี่ยนแปลงได้
5. สามารถจับเวลาการจองของผู้ใช้บริการได้
6. สามารถแสดงผลการใช้งานลานจอดรถย้อนหลังได้
7. สามารถดึงข้อมูลจากฐานข้อมูลมาบันทึกเป็นเท็กซ์ไฟล์ได้

3.4.2 การสร้างโปรแกรม

ในการโปรแกรมการติดต่อฐานข้อมูล Mysql และพอร์ตอนุกรมได้ใช้ไลบรารีไฟล์ ซึ่งสามารถหาดาวน์โหลด ได้ในเครือข่ายอินเตอร์เน็ต ซึ่งรายระเอียดอยู่ในบทที่ 2 และได้ใช้คลาสไลบรารีของ MFC ที่มีมากับ Visual C++



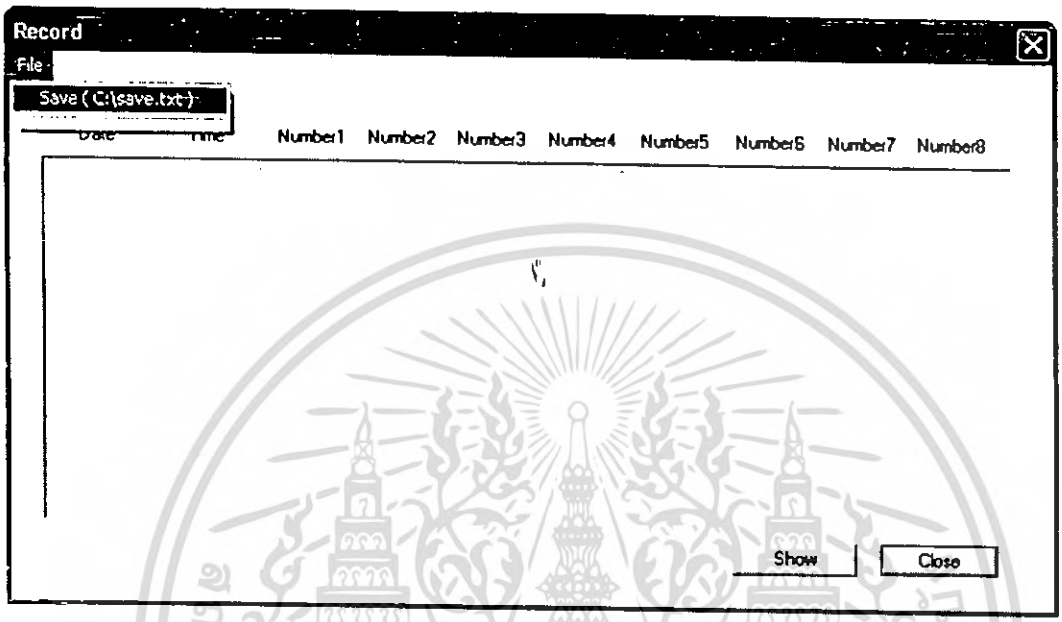
รูปที่ 3.7 หน้าต่างหลักของโปรแกรม

A เป็นส่วนแสดงผลสัญญาณที่รับเข้ามาจากพอร์ตอนุกรม ที่ทำการแปลงสัญญาณเป็นข้อความแล้ว

B เป็นส่วนแสดงผลข้อมูลสถานะลานจอดรถ จากฐานข้อมูล ซึ่งจะมีการเปลี่ยนแปลงก็ต่อเมื่อสัญญาณที่รับเข้ามาของตำแหน่งที่จอดใดๆ เป็นข้อมูลที่ไม่ใช่ค่าเดิมจึงทำการปรับปรุงฐานข้อมูล

C เป็นส่วนแสดงผลเวลาล่าสุดที่ลานจอดรถนะตำแหน่งใดๆ มีการเปลี่ยนแปลงไป ซึ่งจะคงค่าเวลานั้นไว้จนกว่าจะมีการเปลี่ยนแปลงการจอดรถ

หน้าต่างแสดงผลข้อมูลลานจอดรถย้อนหลัง ประกอบด้วย ส่วนแสดงผลข้อมูลย้อนหลังและเมนูสำหรับการบันทึกเก็บเป็นเท็กซ์ไฟล์



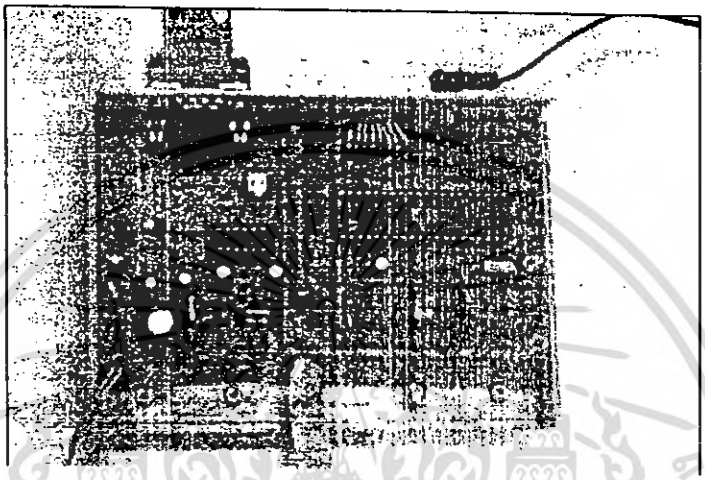
รูปที่ 3.8 หน้าต่างแสดงผลข้อมูลลานจอดรถย้อนหลัง

บทที่4

การทดลองและผลการทดลอง

4.1 ส่วนของภาคฮาร์ดแวร์

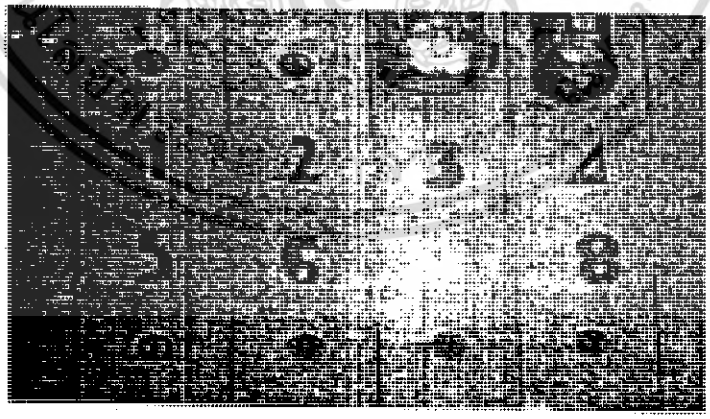
4.1.1 แผงวงจรของระบบตรวจสอบสถานะการจอดรถ



รูปที่ 4.1 แผงวงจรของระบบตรวจสอบสถานะการจอดรถ

4.1.2 แบบจำลองที่จอดรถ

แสดงสถานะลานจอดรถดังรูปที่ 4.2 โดยจะนำค่าที่ตรวจสอบได้ไปเก็บไว้ในฐานข้อมูล MySQL



รูปที่ 4.2 แบบจำลองที่จอดรถ

4.2 ส่วนของฐานข้อมูล MySQL

การสร้างฐานข้อมูลใช้งานส่วนหนึ่งได้ใช้ PHP MyAdmin ช่วยในการสร้างตารางขึ้นมาและใช้ภาษา SQL ปรับแต่ง มี 4 ตาราง คือ

1. ตารางเก็บข้อมูลผู้ใช้

ประกอบด้วย หมายเลขอันดับ ชื่อ นามสกุล ที่อยู่ เบอร์โทรศพท์ อีเมล์และสถานะการจองที่จอดรถ

```
mysql> select*from member\G
*************************** 1. row ***************************
    #ID: 5
  NAME: art
SURNAME: maiuono
ADDRESS: 161 rama4 road klongtoi Bangkok
  PHONE: 07-7127222
  EMAIL: sr@yahoo.com
BOOKING: not
*************************** 2. row ***************************
    #ID: 4
  NAME: Tawal
SURNAME: piyavanishsakul
ADDRESS: 162 soi  90 satorn Bangkok
  PHONE: 02-8487567
  EMAIL: tawal@hotmail.com
BOOKING: not
*************************** 3. row ***************************
    #ID: 3
  NAME: pitak
SURNAME: surakulanan
ADDRESS: 161 soi pailom bangna Bangkok
  PHONE: 02-2841417
  EMAIL: pitak@gmail.com
BOOKING: not
*************************** 4. row ***************************
    #ID: 2
  NAME: manu
SURNAME: visatsiriwarchai
ADDRESS: 100 soi 70 Rama 3 yannava Bangkok
  PHONE: 06-8414236
  EMAIL: hothotmanu@yahoo.com
BOOKING: not
*************************** 5. row ***************************
    #ID: 1
  NAME: Jesus
SURNAME: Christ
ADDRESS: heavenly
  PHONE: 7777777
  EMAIL: jesus@heavenly.com
BOOKING: not
5 rows in set (0.00 sec)

mysql>
```

รูปที่ 4.3 ตารางบันทึกข้อมูลผู้ใช้บริการ

2. ตารางเก็บรายชื่อและรหัสสำหรับล็อกอิน

```
mysql> select*from login;
+------+----------+----------+
| #ID  | USERNAME | PASSWORD |
+------+----------+----------+
|    5 | art      | artoe    |
|    4 | wal      | tawal    |
|    2 | manu     | yumaeku  |
|    3 | tak      | sura     |
|    1 | je       | ch       |
+------+----------+----------+
5 rows in set (0.01 sec)
```

รูปที่ 4.4 แสดงรายชื่อและรหัสผ่านที่เก็บไว้ในระบบฐานข้อมูล

3. ตารางเก็บสถานะของลานจอดรถ

ประกอบด้วย

- ground เก็บตำแหน่งที่จอดรถ
- status เก็บสถานะของที่จอดรถ
- time เก็บวันเวลาตั้งแต่ที่ลานจอดมีการเปลี่ยนแปลง
- timestamp เก็บข้อมูลฟิลด์ time ไว้ในรูปวินาที เพื่อเปรียบเทียบกับฟิลด์ timeout ในการบอกหมดเวลาการจอง(ตั้งไว้ที่ 1 ชั่วโมง)
- customer เก็บรายชื่อผู้ใช้ที่ทำการจอง
- timeout เก็บเวลาสิ้นสุดการจอง

```
mysql> select*from park;
+---------+----------+---------------------+----------------+----------+----------------+
| ground  | status   | time                | timestamp      | customer | timeout        |
+---------+----------+---------------------+----------------+----------+----------------+
| number1 | reserved | 2006-03-14 14:36:09 | 20060316124041 | jesus    | 20060316124041 |
| number2 | YES      | 2006-03-14 14:36:09 | 20060316101445 | not      | 20060314203650 |
| number3 | NO       | 2006-03-14 14:36:09 | 20060316101421 | not      | 20060316042032 |
| number4 | NO       | 2006-03-14 14:36:09 | 20060316101226 | not      | 20060316042508 |
| number5 | YES      | 2006-03-16 03:07:01 | 20060316030701 | not      | 20060300132941 |
| number6 | reserved | 2006-03-14 14:37:50 | 20060314162150 | wal      | 20060314162150 |
| number7 | reserved | 2006-03-13 23:04:35 | 20060314163420 | tak      | 20060314163420 |
| number8 | YES      | 2006-03-14 14:38:12 | 20060316101457 | got      | 20060314165420 |
+---------+----------+---------------------+----------------+----------+----------------+
8 rows in set (0.10 sec)

mysql>
```

รูปที่ 4.5 ตารางบันทึกสถานะลานจอดรถ

4. ตารางเก็บข้อมูลย้อนหลัง

เป็นส่วนเพิ่มเติมของโครงงานนี้ ซึ่งเป็นข้อมูลไว้สำหรับการดูการเปลี่ยนแปลงของลานจอดรถ

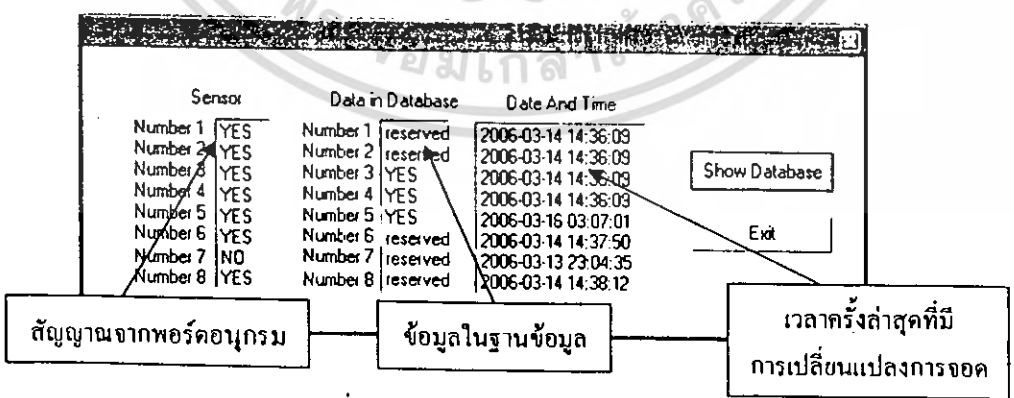
```
mysql> select*from record limit 50,3\G
*************************** 1. row ***************************
   DATE: 2006-03-14
   TIME: 14:36:32
NUMBER1: YES
NUMBER2: YES
NUMBER3: YES
NUMBER4: YES
NUMBER5: YES
NUMBER6: YES
NUMBER7: NO
NUMBER8: YES
*************************** 2. row ***************************
   DATE: 2006-03-14
   TIME: 14:36:33
NUMBER1: YES
NUMBER2: YES
NUMBER3: YES
NUMBER4: YES
NUMBER5: YES
NUMBER6: NO
NUMBER7: NO
NUMBER8: YES
*************************** 3. row ***************************
   DATE: 2006-03-14
   TIME: 14:36:34
NUMBER1: YES
NUMBER2: YES
NUMBER3: YES
NUMBER4: YES
NUMBER5: YES
NUMBER6: YES
NUMBER7: NO
NUMBER8: NO
3 rows in set (0.00 sec)

mysql>
```

รูปที่ 4.6 ตารางเก็บข้อมูลลานจอดรถย้อนหลัง

4.3 โปรแกรมทางด้านเซิร์ฟเวอร์

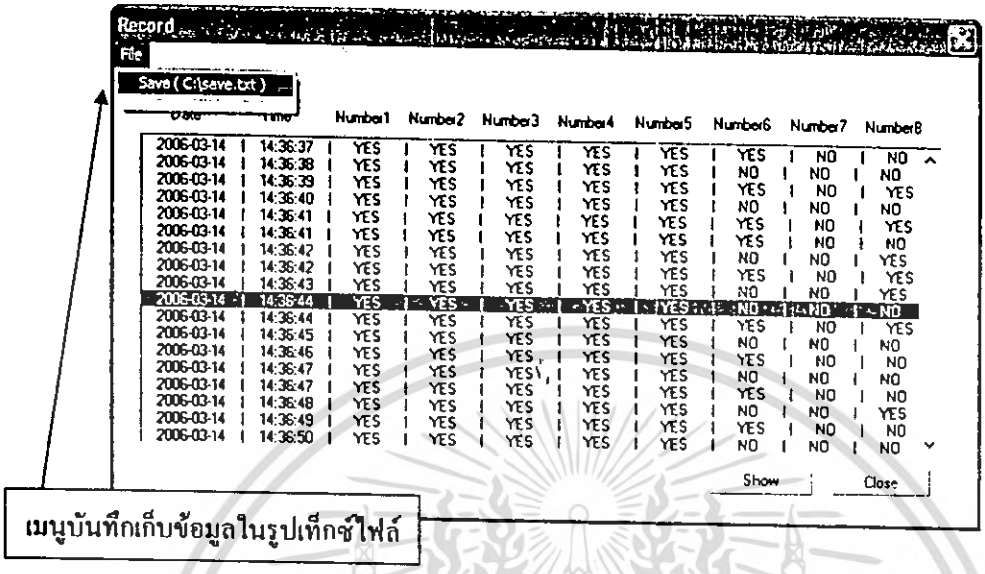
4.3.1 หน้าต่างแสดงสถานะลานจอดรถ

เป็นโปรแกรมที่มีหน้าที่ในการรับสัญญาณจากพอร์ตอนุกรมมาแสดงผล บันทึกลงฐานข้อมูลเก็บ
สถานะการจอดรถทุกครั้งที่มีการเปลี่ยนแปลงและเวลาการจองที่จอดรถของผู้ใช้บริการ

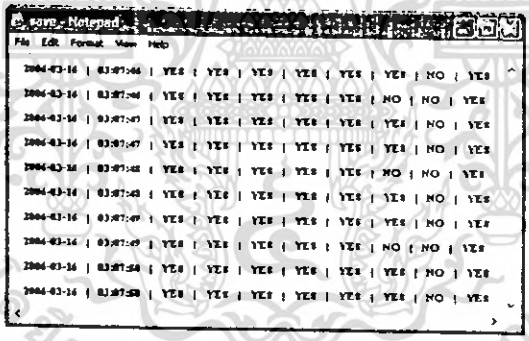


รูปที่ 4.7 หน้าต่างแสดงสถานะลานจอดรถ

4.3.2 หน้าต่างแสดงสถานะลานจอดรถย้อนหลังจากฐานข้อมูล



รูปที่ 4.8 หน้าต่างแสดงสถานะลานจอดรถย้อนหลัง



รูปที่ 4.9 เท็กซ์ไฟล์ที่ได้จากการบันทึก

4.4 ส่วนการแสดงผลให้ผู้ใช้บริการ

4.4.1 ส่วนบริการผ่านเว็บบราวเซอร์

หน้าต่างเว็บเพจหลัก

เมื่อเปิดเข้าสู่เว็บเพจหลักไปที่ก็จะปรากฏหน้าเว็บเพจหลักขึ้นมา http:/127.0.0.1/index.php ซึ่งเป็นส่วนการตรวจสอบโปรแกรมผู้ใช้บริการว่าเป็นบราวเซอร์หรือไมโครบราวเซอร์ ระบบจะทำการแยกการให้บริการโดยอาศัยจากตัวแปรแวดล้อม ที่ โปรแกรมเครื่องผู้ใช้ส่งมาพร้อมกับการร้องขอบริการเอกสาร CGI ซึ่งตัว

เซิร์ฟเวอร์จะรับเข้ามาทำให้ทราบสถานะต่างๆเกี่ยวกับการร้องขอเช่น ชื่อโปรแกรมที่ไคลเอนต์ใช้ รหัสภาษาที่ไคลเอนต์ใช้และเข้าใจได้ ชื่อเซิร์ฟเวอร์ หมายเลขพอร์ต วิธีการเข้ามาร้องขอบริการ เป็นต้น สำหรับโครงงานนี้ต้องการทราบประเภทข้อมูลที่ไคลเอนต์รู้จักเอกสารแวบหรือเว็บซึ่งใช้คำสั่ง HTTP_ACCEPT และต้องการทราบชื่อโปรแกรมที่ไคลเอนต์ใช้ร้องขอมายังเอกสาร CGI โดยใช้คำสั่ง HTTP_USER_AGENT ใช้ ฟังก์ชันของ PHP คือ eregi() ในการค้นหาอักษรหรือข้อความที่แสดงให้ทราบรายชื่อประเภทข้อมูลที่ทางไคลเอนต์รู้จักเป็น text/vnd.wap.wml หรือไม่

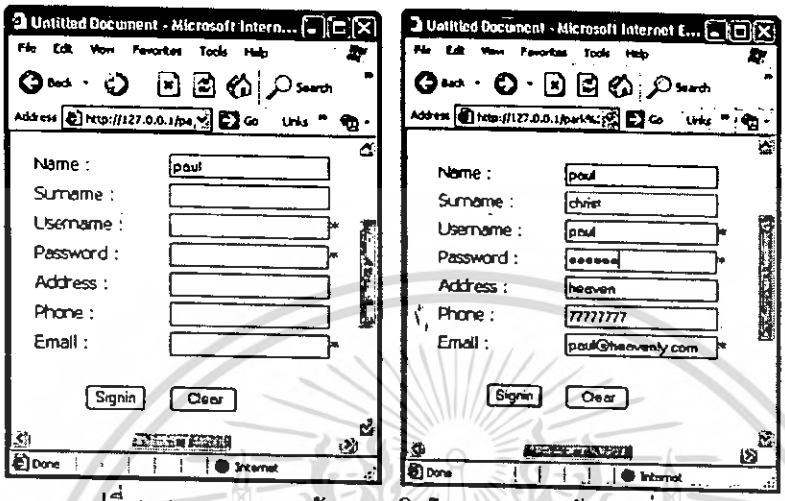
โดยใช้โค้ดดังนี้ if (eregi("text/vnd.wap.wml",$HTTP_ACCEPT)) แล้วจึงสั่งให้เว็บเซิร์ฟเวอร์โหลดไฟล์ ที่เหมาะสม ด้วยคำสั่ง header("Location:index.wml");

หน้าต่างล็อกอิน

กรอกชื่อผู้ใช้บริการและรหัสผ่าน โดยชื่อผู้ใช้และรหัสผ่านจะถูกเก็บไว้ในฐานข้อมูลของระบบอยู่ที่ http:/127.0.0.1/login.php

รูปที่ 4.10 หน้าต่างล็อกอิน

หากไม่พบข้อมูลในระบบฐานข้อมูล ระบบจะแจ้งให้สมัครสมาชิกใช้บริการและทำการกรอกข้อมูลลงไปเก็บไว้ในฐานข้อมูล



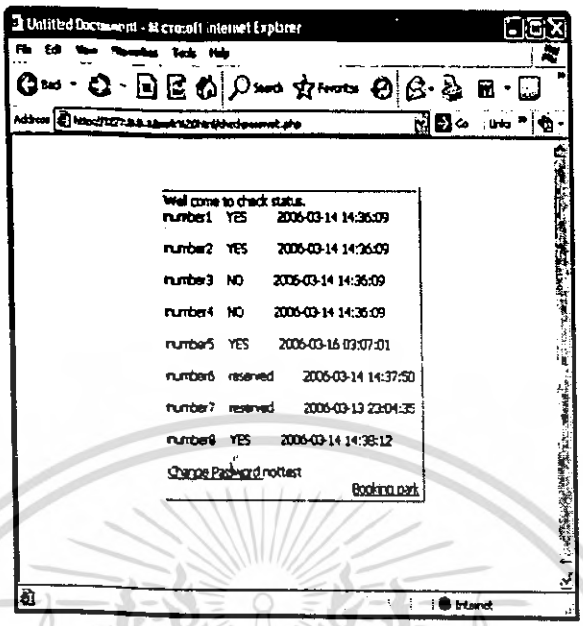
รูปที่ 4.13 แสดงการสมัครสมาชิกโดยการกรอกข้อมูลเพื่อใช้บริการ

เมื่อทำการสมัครเรียบร้อยแล้ว ระบบจะทำการแจ้งข้อมูลกลับมา

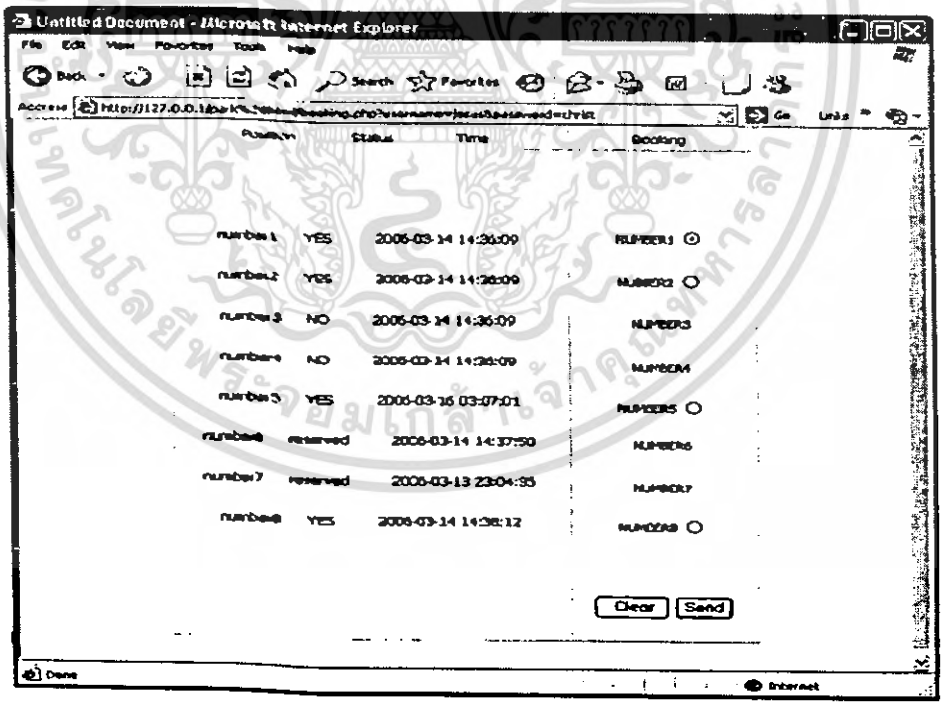
รูปที่ 4.14 ระบบทำการแจ้งข้อมูลที่เก็บไว้ในฐานข้อมูลเมื่อใช้บริการสำเร็จ

หน้าต่างแสดงสถานะลานจอดรถ

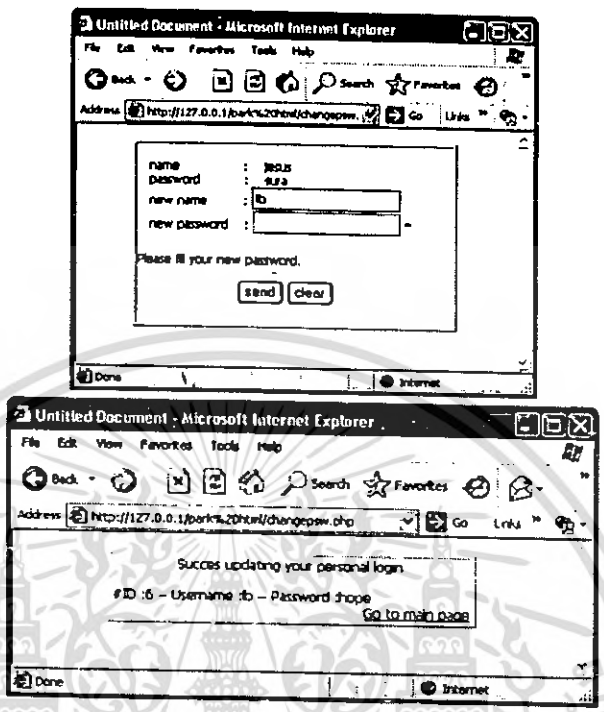
ประกอบด้วยส่วนแสดงสถานะลานจอดรถ ตำแหน่ง และเวลาที่มีการเปลี่ยนแปลงข้อมูล รวมทั้งเมนูเสริม สำหรับการเปลี่ยนชื่อ และรหัสผู้ใช้ และเมนูสำหรับขอจองตำแหน่งที่จอดรถ ซึ่งในส่วนของเมนูการจองนี้ในทางปฏิบัติจะเหมาะกับการไปประชุมกับงานที่ตำแหน่งที่จองมีความสำคัญ และมีการควบคุมที่หน้างาน เช่น การจองโต๊ะอาหารในภัตตาคาร



รูปที่ 4.13 หน้าต่างแสดงสถานะลานจอดรถ

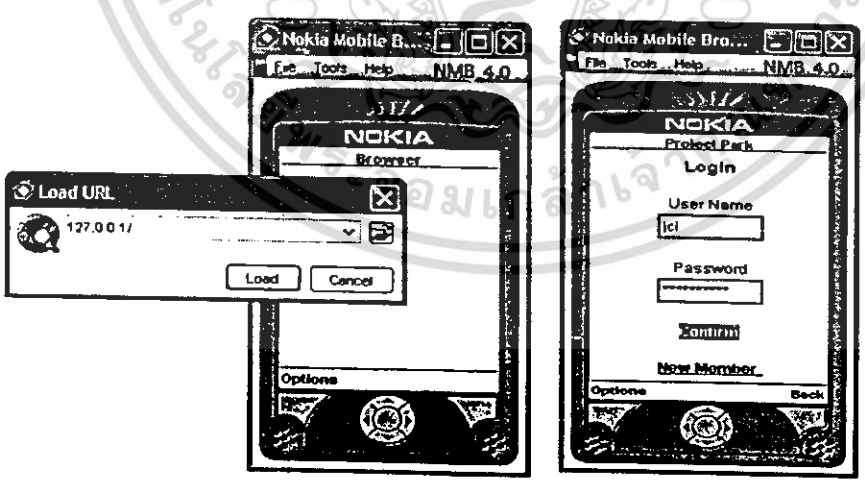


รูปที่ 4.14 ผลลัพธ์ของเมนู จองที่จอดรถ



รูปที่ 4.15 ผลลัพธ์ของเมนู เปลี่ยนรหัส

4.4.2 ส่วนบริการผ่านแวบ

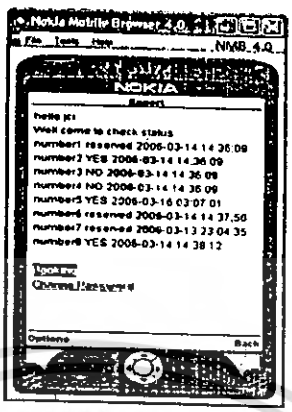


รูปที่ 4.16 หน้าล็อกอิน

ทำการกรอกชื่อผู้ใช้บริการและรหัสผ่าน หากระบบไม่พบในฐานข้อมูลก็จะมีการแจ้งและให้สมัครสมาชิกก่อนใช้บริการ

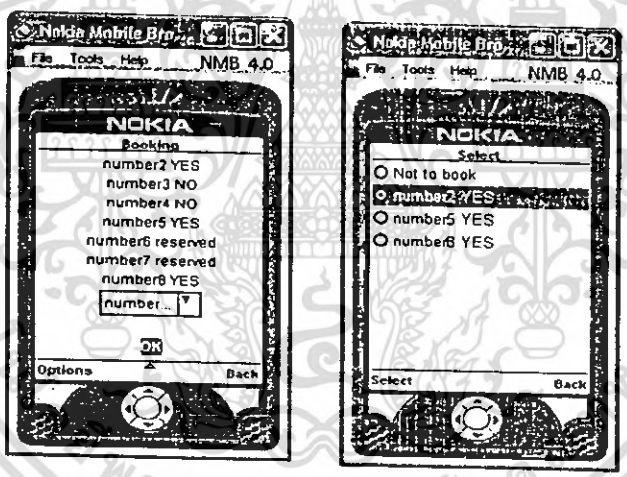
รูปที่ 4.17 แสดงขั้นตอนของระบบสมาชิกใหม่

หน้าต่างแสดงสถานะลานจอดรถ



รูปที่ 4.18 แสดงสถานะลานจอดรถ

เมื่อเข้าสู่หน้าการจองที่จอดรถแล้วก็ทำการเลือกที่จอดรถ โดยผู้ใช้บริการจะสามารถมีสิทธิ์ทำการจองได้เพียง 1 คันเท่านั้น



รูปที่ 4.19 แสดงส่วนการจองที่จอดรถ

บทที่ 5
บทวิจารณ์และบทสรุป

โครงงานนี้ได้ทำการตรวจสอบสถานะการจอดรถผ่านโทรศัพท์เคลื่อนที่ โดยรับค่าสัญญาณจากเซนเซอร์ส่งไปให้ไมโครคอนโทรลเลอร์ซึ่งจะนำข้อมูลที่ได้ส่งไปเก็บยังฐานข้อมูลที่ด้านผู้ให้บริการ เมื่อผู้ใช้บริการเข้ารับบริการผ่านทางโทรศัพท์เคลื่อนที่ก็จะสามารถทำการตรวจสอบสถานะการจอดรถได้

5.1 อุปสรรคที่พบในโครงงาน

- ในส่วนของโปรแกรมพอร์ตอนุกรมติดต่อฐานข้อมูลพบข้อบกพร่องมาก
- ความเสถียรภาพของตัวเซนเซอร์มีผลต่อระบบตรวจสอบสถานการจอดรถ

5.2 แนวทางแก้ไขปัญหา

- ระวังในเรื่องการใช้โปรแกรม ศึกษาฟังก์ชันการทำงาน
- ควรเลือกใช้เซนเซอร์ที่มีคุณภาพ มีความเสถียรต่อระบบ

5.3 แนวทางในการพัฒนา

- ประยุกต์ใช้กับงานอื่นๆ เช่น ระบบการจองโต๊ะอาหารในภัตตาคาร
- เปลี่ยนจากการใช้เทคโนโลยีของเว็บแอพพลิเคชันเป็นแอพพลิเคชัน
- พัฒนาในส่วนอินเตอร์เฟสให้สวยงามมากขึ้น โดยใช้เทคโนโลยีของ Flash

หนังสืออ้างอิง

1. กิตติ ภักดีวัฒนะกุล. **คัมภีร์ PHP**. เคทีพี คอมพ์ แอนด์ คอนซัลท์: กรุงเทพฯ, 2547
2. กิดิภูมิ วรฉัตร. **Php เปลี่ยนวิถีสู่การสร้างโฮมเพจอย่างมือโปร**: บริษัท วิตตี้ กรุ๊ป จำกัด. กรุงเทพฯ, **2543**
3. ธีรวัฒน์ ประกอบผล. **การประยุกต์ใช้งาน ไมโครคอนโทรลเลอร์**: สำนักพิมพ์ สมาคมส่งเสริมเทคโนโลยี (ไทย – ญี่ปุ่น). กรุงเทพฯ,2543
4. นิรันดร ทนงศักดิ์มนตรี. **W@P The World In Your Hand ย่อโลกไว้ในมือคุณ**: บริษัท ซีเอ็ดยูเคชั่น จำกัด (มหาชน). กรุงเทพฯ, 2543
5. นิรุธ อำนวยศิลป์. **Microsoft Visual C++ By Example**: ซัคเซส มีเดีย. กรุงเทพฯ, 2544
6. ผศ.อกินเดร อูนากูล. **Web Application Development Process and Mehodology Using UML**. วิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง. กรุงเทพฯ, 2546
7. ผศ.อุทัย สุขสิงห์. **ไมโครโพรเซสเซอร์และไมโครคอนโทรลเลอร์ MCS-51**: สำนักพิมพ์ ส.ส.ท. กรุงเทพฯ, 2547
8. มณีโชติ สมานไทย. **คู่มือการออกแบบฐานข้อมูลและภาษา SQL**: ด่านสุทธาการพิมพ์ จำกัด.กรุงเทพ,2546
9. ยุทธนา ลีลาศวัฒนกุล. **คู่มือการเขียนโปรแกรมและใช้งาน Visual C++6.0 ฉบับโปรแกรมเมอร์**: อินโฟเพรส.กรุงเทพฯ, 2544
10. เรืองไกร รังสิพล. **เจาะระบบ TCP/IP จุดอ่อนของโปรโตคอลและวิธีป้องกัน**: บริษัท โปรวิชั่น จำกัด. กรุงเทพฯ, 2544
11. สราวุธ อ้อยศรีสกุล. **เปิดมิติ Mobile Intenet ด้วย WAP**: บริษัท วิตตี้ กรุ๊ป จำกัด. กรุงเทพฯ, 2544
12. สมประสงค์ ธิตินิลนิธิ. **เรียนลัด PHP4 ครอบคลุม PHP เวอร์ชั่น4.2**: บริษัท โปรวิชั่น จำกัด. กรุงเทพฯ, 2545
13. สาธิต ชัยวิวัฒน์ตระกูล. **เติมเทคนิค MySQL ให้เต็มประสิทธิภาพ**: บริษัท วิตตี้ กรุ๊ป จำกัด. กรุงเทพฯ, 2547
14. อัมรินทร์ เพ็ชรกุล. **เลือกซื้อเลือกสนุก ไขปัญหามือถือ**: บริษัท ซัคเซส มีเดีย จำกัด. กรุงเทพฯ, 2544

ภาคผนวก

Serial.asm

```
                ORG   0000H
init:           mov   scon, #052h
                mov   tmod, #020h
                mov   th1, #0FDh
                setb  tr1
main:           acall tx
                acall DELAY_1S
                sjmp  main
tx:             mov   a, p1
                jnb   ti, $
                clr   ti
                mov   sbuf, a
                ret
DELAY_10MS:     MOV   R6, #10
DELAY_10MS_1:   MOV   R7, #0E6H
DELAY_10MS_2:   DJNZ  R7, DELAY_10MS_2
                DJNZ  R6, DELAY_10MS_1
                RET

DELAY_1S:       MOV   R5, #100
DELAY_1S_1:     ACALL DELAY_10MS
                DJNZ  R5, DELAY_1S_1
                RET
                end
```

Commtest.cpp

```
/*
**    FILENAME          Commtest.h
**    PURPOSE           This is the application class.
**    CREATION DATE     15-09-1997
**    LAST MODIFICATION 12-11-1997
**    AUTHOR            Remon Spekreijse
*/
#include "stdafx.h"
#include "commtest.h"
#include "commtestDlg.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
BEGIN_MESSAGE_MAP(CCommtestApp, CWinApp)
//{{AFX_MSG_MAP(CCommtestApp)
// NOTE - the ClassWizard will add and remove mapping macros here.
//    DO NOT EDIT what you see in these blocks of generated code!
//}}AFX_MSG
      ON_COMMAND(ID_HELP, CWinApp::OnHelp)
END_MESSAGE_MAP()
CCommtestApp::CCommtestApp(){}
CCommtestApp theApp;
BOOL CCommtestApp::InitInstance(){
// Standard initialization
// If you are not using these features and wish to reduce the size
//  of your final executable, you should remove from the following
//  the specific initialization routines you do not need.
```

```
#ifdef _AFXDLL
      Enable3dControls();// Call this when using MFC in a shared DLL
#else
Enable3dControlsStatic();      // Call this when linking to MFC statically
#endif
CCommtestDlg dlg;
m_pMainWnd = &dlg;
dlg.DoModal();
// Since the dialog has been closed, return FALSE so that we exit the
//  application, rather than start the application's message pump.
      return FALSE;}
```

commtest.Dlg.cpp

```
/*
**    FILENAME                CommTestDlg.cpp
**    PURPOSE                      This is the dialog that shows the comm
activity.
**    CREATION DATE           15-09-1997
**    LAST MODIFICATION 12-11-1997
**    AUTHOR                       Remon Spekreijse
*/
#include "stdafx.h"
#include "commtest.h"
#include "commtestDlg.h"
#include "configdlg.h"
#include "Record.h"
#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif
CCommtestDlg::CCommtestDlg(CWnd* pParent /*=NULL*/)
 CDialog(CCommtestDlg::IDD, pParent){
      //{{AFX_DATA_INIT(CCommtestDlg)
      //}}AFX_DATA_INIT
      // Note that LoadIcon does not require a subsequent DestroyIcon in
Win32
      m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);}
void CCommtestDlg::DoDataExchange(CDataExchange* pDX)
{CDialog::DoDataExchange(pDX);
//{{AFX_DATA_MAP(CCommtestDlg)
DDX_Control(pDX, IDC_LISTTIME, m_listtime);
DDX_Control(pDX, IDC_LISTDB, m_listdb);
//}}AFX_DATA_MAP}
BEGIN_MESSAGE_MAP(CCommtestDlg, CDialog)
//{{AFX_MSG_MAP(CCommtestDlg)
ON_WM_PAINT()
ON_WM_QUERYDRAGICON()
ON_MESSAGE(WM_COMM_RXCHAR, OnCommunication)
ON_MESSAGE(WM_COMM_CTS_DETECTED, OnCTSDetected)
ON_BN_CLICKED(IDC_SHOWDATABASE, OnShowdatabase)
ON_COMMAND(ID_FILE_SAVE, OnFileSave)
      //}}AFX_MSG_MAP
END_MESSAGE_MAP()

BOOL CCommtestDlg::OnInitDialog()
```

```
{CDialog::OnInitDialog();
link = mysql_init(NULL);
// Set the icon for this dialog.  The framework does this automatically
//  when the application's main window is not a dialog
      SetIcon(m_hIcon, TRUE);            // Set big icon
      SetIcon(m_hIcon, FALSE);           // Set small icon
m_ListBox[0].SubclassDlgItem(IDC_LIST1, this);
if (m_Ports[0].InitPort(this, 0 + 1, 0 == 0 ? 9600 : 19200))
m_Ports[0].StartMonitoring();
if (!mysql_real_connect(link,"localhost","root","","project",0,NULL,0)){
      MessageBox("error connect to DB");
      }else{
      MessageBox("connect to DB");}
//.............initial record class.............//
      CRecord::IDC_DATE.AddString("manu love wow");
return TRUE;  // return TRUE  unless you set the focus to a control}
void CCommtestDlg::OnPaint() {
      if (IsIconic())
      {CPaintDC dc(this); // device context for painting
      SendMessage(WM_ICONERASEBKGND, (WPARAM) dc.GetSafeHdc(), 0);
// Center icon in client rectangle
int cxIcon = GetSystemMetrics(SM_CXICON);
int cyIcon = GetSystemMetrics(SM_CYICON);
CRect rect;
      GetClientRect(&rect);
      int x = (rect.Width() - cxIcon + 1) / 2;
      int y = (rect.Height() - cyIcon + 1) / 2;
// Draw the icon
      dc.DrawIcon(x, y, m_hIcon);}
      else{CDialog::OnPaint();}}
HCURSOR CCommtestDlg::OnQueryDragIcon()
{return (HCURSOR) m_hIcon;
}
LONG CCommtestDlg::OnCommunication(WPARAM ch, LPARAM port)
{
      CString show,string,query,pnumber;
      string = (char)ch;
      int i,count=0;
      if (port <= 0 || port > 4)
      return -1;
      char x = 0x01;
      char c = (char) ch;
CString record,sensor[8] = {"0","0","0","0","0","0","0","0"};
for( i=0;i<8;i++,x<<=1)
      {char temp = c;
      if((temp &= x)==0){
      sensor[i]="NO";}
      else{
      sensor[i]="YES";}}
      m_ListBox[0].ResetContent();
//-------Update data & check reserve timeout----
      for( i=0;i<8;i++){
      CString data,number ;
      data.Format("%s",sensor[i]);
      //......convert int to CString.........
      number.Format("%d",i+1);
      m_ListBox[0].AddString(sensor[i]);
```

```
      query.Format("select status,timestamp,timeout from park where
ground='number%s';",number);
      mysql_query(link,query);
      result = mysql_store_result(link);
      row = mysql_fetch_row(result);
      CString timenow,timeout,status;
      status.Format("%s",row[0]);
      timenow.Format("\n%s",row[1]);
      timeout.Format("\n%s",row[2]);
                  if(status != data)//"status" != "data"{count++;
      if(timenow>timeout){
      query.Format("UPDATE member SET booking ='not'where
booking='number%s';",number);
      mysql_query(link,query);
      query.Format("UPDA % park SET time =NOW(),timestamp = NOW(),status
='%s' where ground='number%s';",data,number);
      mysql_query(link,query);      }}
      //.........using park record ..........}
      if(count>0){
      record.Format("insert into
`record`(`date`,`time`,`number1`,`number2`,`number3`,`number4`,`number5`,`
number6`,`number7`,`number8`)
VALUES(CURDATE(),CURTIME(),'%s','%s','%s','%s','%s','%s','%s','%s');",sens
or[0],sensor[1],sensor[2],sensor[3],sensor[4],sensor[5],sensor[6],sensor[7
]);
      mysql_query(link,record);}
      CCommtestDlg::showdb();
//.....................................................
      return 0;}
CCommtestDlg::OnCTSDetected(WPARAM, LPARAM port)
{if (port <= 0 || port > 4)
      return -1;
CString string; string = "Clear To Send";
      m_ListBox[0].AddString(string);
      m_ListBox[0].SetSel(m_ListBox[0].GetCount()-1, TRUE);
return 0;}
void CCommtestDlg::showdb()
{
m_listdb.ResetContent();
m_listtime.ResetContent();
char cmd1[80] = "select status from park";
char cmd2[80] = "select time from park";
mysql_query(link,cmd2);
result = mysql_store_result(link);
while ( row = mysql_fetch_row(result))
      {
            CString str;
            str.Format("\n%s",row[0]);
            m_listtime.AddString(str);
      }
mysql_query(link,cmd1);
result = mysql_store_result(link);
CString test;
test.Format("%d",result);
while ( row = mysql_fetch_row(result))
      {
            CString str;
```

```
            str.Format("\n%s",row[0]);
            m_listdb.AddString(str);
      }}
void CCommtestDlg::OnShowdatabase()
{CRecord record;
record.DoModal();

}void CCommtestDlg::OnFileSave() {
CString str;
CStdioFile s("c:/save.txt",CFile::modeCreate|CFile::modeWrite);
mysql_query(link,"select*From record");
result = mysql_store_result(link);
while(row = mysql_fetch_row(result)) {
str.Format("   %s   |   %s   |   %s   |   %s   |   %s   |   %s   |   %s
|   %s   |   %s   |   %s
\n",row[0],row[1],row[2],row[3],row[4],row[5],row[6],row[7],row[8],row[9])
;
      s.WriteString(str);}}
```

SerialPort.cpp

```
/*
**     FILENAME                CSerialPort.cpp
**     PURPOSE                      This class can read, write and watch
one serial port.
       It sends messages to its owner when something happends on the port
**                                  The class creates a thread for reading
and writing so the main
**                                  program is not blocked.
**     CREATION DATE           15-09-1997
**     LAST MODIFICATION 12-11-1997
**     AUTHOR                       Remon Spekreijse
*/

#include "stdafx.h"
#include "SerialPort.h"
#include <assert.h>
//
// Constructor
//
CSerialPort::CSerialPort()
{m_hComm = NULL;
// initialize overlapped structure members to zero
m_ov.Offset = 0;
m_ov.OffsetHigh = 0;

      // create events
      m_ov.hEvent = NULL;
      m_hWriteEvent = NULL;
      m_hShutdownEvent = NULL;
      m_szWriteBuffer = NULL;
      m_bThreadAlive = FALSE;}
// Delete dynamic memory
CSerialPort::~CSerialPort()
{do{SetEvent(m_hShutdownEvent);
} while (m_bThreadAlive);
TRACE("Thread ended\n");
```

```
delete [] m_szWriteBuffer;}
// Initialize the port. This can be port 1 to 4.
BOOL CSerialPort::InitPort(CWnd* pPortOwner,     // the owner (CWnd) of the
port (receives message)
   UINT  portnr,  // portnumber (1..4)
UINT  baud,        // baudrate
   char  parity,          // parity
UINT  databits,          // databits
 UINT  stopbits,         // stopbits
DWORD dwCommEvents,
// EV_RXCHAR, EV_CTS etc
UINT  writebuffersize)  // size to the writebuffer{
assert(portnr > 0 && portnr < 5);
      assert(pPortOwner != NULL);
      // if the thread is alive: Kill.
      if (m_bThreadAlive)
      {do{
            SetEvent(m_hShutdownEvent);
            } while (m_bThreadAlive);
            TRACE("Thread ended\n");}
      // create events
      if (m_ov.hEvent != NULL)
      ResetEvent(m_ov.hEvent);
      m_ov.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
if (m_hWriteEvent != NULL)
      ResetEvent(m_hWriteEvent);
      m_hWriteEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
      if (m_hShutdownEvent != NULL)
      ResetEvent(m_hShutdownEvent);
      m_hShutdownEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
      // initialize the event objects
      m_hEventArray[0] = m_hShutdownEvent;      // highest priority
      m_hEventArray[1] = m_ov.hEvent;
      m_hEventArray[2] = m_hWriteEvent;
      // initialize critical section
      InitializeCriticalSection(&m_csCommunicationSync);

 // set buffersize for writing and save the owner
      m_pOwner = pPortOwner;
      if (m_szWriteBuffer != NULL)
            delete [] m_szWriteBuffer;
      m_szWriteBuffer = new char[writebuffersize];
      m_nPortNr = portnr;
      m_nWriteBufferSize = writebuffersize;
      m_dwCommEvents = dwCommEvents;
      BOOL bResult = FALSE;
      char *szPort = new char[50];
      char *szBaud = new char[50];
      // now it critical!
      EnterCriticalSection(&m_csCommunicationSync);
      // if the port is already opened: close it
      if (m_hComm != NULL)
      {CloseHandle(m_hComm);
      m_hComm = NULL;
      }
      sprintf(szPort, "COM%d", portnr);
```

```
      sprintf(szBaud, "baud=%d parity=%c data=%d stop=%d", baud, parity,
databits, stopbits);
      // get a handle to the port
      m_hComm = CreateFile(szPort,        // communication port string
(COMX)
                           GENERIC_READ | GENERIC_WRITE,
// read/write types
   0, // comm devices must be opened with exclusive access
   NULL,    // no security attributes
    OPEN_EXISTING,      // comm devices must use OPEN_EXISTING
   FILE_FLAG_OVERLAPPED,           // Async I/O
      0);                          // template must be 0 for comm devices
if (m_hComm == INVALID_HANDLE_VALUE)
      {
      // port not found
      delete [] szPort;
      delete [] szBaud;
      return FALSE;}
// set the timeout values
m_CommTimeouts.ReadIntervalTimeout = 1000;
m_CommTimeouts.ReadTotalTimeoutMultiplier = 1000;
m_CommTimeouts.ReadTotalTimeoutConstant = 1000;
m_CommTimeouts.WriteTotalTimeoutMultiplier = 1000;
m_CommTimeouts.WriteTotalTimeoutConstant = 1000;

      // configure
      if (SetCommTimeouts(m_hComm, &m_CommTimeouts))
      {if (SetCommMask(m_hComm, dwCommEvents))
      {
if (GetCommState(m_hComm, &m_dcb))
      {m_dcb.fRtsControl = RTS_CONTROL_ENABLE;
// set RTS bit high!
      if (BuildCommDCB(szBaud, &m_dcb))
      {     if (SetCommState(m_hComm, &m_dcb)); // normal operation...
continue
      else ProcessErrorMessage("SetCommState()");
      }else
      ProcessErrorMessage("BuildCommDCB()");}elseProcessErrorMessage("GetC
ommState()");}elseProcessErrorMessage("SetCommMask()");}else
ProcessErrorMessage("SetCommTimeouts()");
delete [] szPort;
delete [] szBaud;
      // flush the port
      PurgeComm(m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR | PURGE_RXABORT |
PURGE_TXABORT);
      // release critical section
LeaveCriticalSection(&m_csCommunicationSync);

      TRACE("Initialisation for communicationport %d completed.\nUse
Startmonitor to communicate.\n", portnr);
      return TRUE;
}
//  The CommThread Function.
UINT CSerialPort::CommThread(LPVOID pParam)
{
// Cast the void pointer passed to the thread back to
      // a pointer of CSerialPort class
```

```
      CSerialPort *port = (CSerialPort*)pParam;

// Set the status variable in the dialog class to
// TRUE to indicate the thread is running.
      port->m_bThreadAlive = TRUE;
      // Misc. variables
      DWORD BytesTransfered = 0;
      DWORD Event = 0;
      DWORD CommEvent = 0;
      DWORD dwError = 0;
      COMSTAT comstat;
      BOOL  bResult = TRUE;

      // Clear comm buffers at startup
      if (port->m_hComm)            // check if the port is opened
            PurgeComm(port->m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR |
PURGE_RXABORT | PURGE_TXABORT);

      // begin forever loop.  This loop will run as long as the thread is
alive.
      for (;;) {
            // Make a call to WaitCommEvent().  This call will return
immediatly
bResult = WaitCommEvent(port->m_hComm, &Event, &port->m_ov);
      if (!bResult)  {
// If WaitCommEvent() returns FALSE, process the last error to determin
// the reason..
switch (dwError = GetLastError())
      { case ERROR_IO_PENDING:
      {     // This is a normal return value if there are no bytes
      // to read at the port.
      // Do nothing and continue                  break;     }case 87:{
      // Under Windows NT, this value is returned for some reason.
      // I have not investigated why, but it is also a valid reply
      Also do nothing and continue.
      break;      }default: {
// All other error codes indicate a serious error has// occured.  Process
this error.
port->ProcessErrorMessage("WaitCommEvent()");
      break;      }
                  }
            }
            else{

      bResult = ClearCommError(port->m_hComm, &dwError, &comstat);

      if (comstat.cbInQue == 0)
            continue;}
Event = WaitForMultipleObjects(3, port->m_hEventArray, FALSE, INFINITE);
switch (Event){
      case 0:     {
      // Shutdown event.  This is event zero so it will be
// the higest priority and be serviced first.

port->m_bThreadAlive = FALSE;
// Kill this thread.  break is not needed, but makes me feel better.
      AfxEndThread(100);
```

```
      break;}
      case 1:     // read event{
GetCommMask(port->m_hComm, &CommEvent);
      if (CommEvent & EV_CTS)
                                    ::SendMessage(port->m_pOwner->m_hWnd,
WM_COMM_CTS_DETECTED, (WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RXFLAG)
      ::SendMessage(port->m_pOwner->m_hWnd, WM_COMM_RXFLAG_DETECTED,
(WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_BREAK)
      ::SendMessage(port->m_pOwner->m_hWnd, WM_COMM_BREAK_DETECTED,
(WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_ERR)
      ::SendMessage(port->m_pOwner->m_hWnd, WM_COMM_ERR_DETECTED, (WPARAM)
0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RING)
      ::SendMessage(port->m_pOwner->m_hWnd, WM_COMM_RING_DETECTED,
(WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RXCHAR)
// Receive character event from port.
      ReceiveChar(port, comstat);
      break;}
      case 2: // write event{
      // Write character event from port
      WriteChar(port);
      break; }} // end switch
      } // close forever loop
      return 0;}
//
// start comm watching
//
BOOL CSerialPort::StartMonitoring()
{     if (!(m_Thread = AfxBeginThread(CommThread, this)))
      return FALSE;
      TRACE("Thread started\n");
      return TRUE;      }
//
// Restart the comm thread
//
BOOL CSerialPort::RestartMonitoring()
{
      TRACE("Thread resumed\n");
      m_Thread->ResumeThread();
      return TRUE;
}
//
// Suspend the comm thread
//
BOOL CSerialPort::StopMonitoring()
{     TRACE("Thread suspended\n");
      m_Thread->SuspendThread();
      return TRUE;      }
//
// If there is a error, give the right message
//
void CSerialPort::ProcessErrorMessage(char* ErrorText)
{     char *Temp = new char[200];
```

```
      LPVOID lpMsgBuf;
      FormatMessage(    FORMAT_MESSAGE_ALLOCATE_BUFFER |
FORMAT_MESSAGE_FROM_SYSTEM,
      NULL,
      GetLastError(),
            MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), // Default language
            (LPTSTR) &lpMsgBuf,
            0,
            NULL
      ); sprintf(Temp, "WARNING:  %s Failed with the following error:
\n%s\nPort: %d\n", (char*)ErrorText, lpMsgBuf, m_nPortNr);
      MessageBox(NULL, Temp, "Application Error", MB_ICONSTOP);
LocalFree(lpMsgBuf);
      delete[] Temp;
}
//
// Write a character.
//
void CSerialPort::WriteChar(CSerialPort* port)
{
      BOOL bWrite = TRUE;
      BOOL bResult = TRUE;
      DWORD BytesSent = 0;
      ResetEvent(port->m_hWriteEvent);
      // Gain ownership of the critical section
      EnterCriticalSection(&port->m_csCommunicationSync);
      if (bWrite)
      {
            // Initailize variables
            port->m_ov.Offset = 0;
            port->m_ov.OffsetHigh = 0;

            // Clear buffer
            PurgeComm(port->m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR |
PURGE_RXABORT | PURGE_TXABORT);
      bResult = WriteFile(port->m_hComm,  // Handle to COMM Port
      port->m_szWriteBuffer,        // Pointer to message buffer in
calling finction
                  strlen((char*)port->m_szWriteBuffer),     //
Length of message to send
                        &BytesSent,             // Where to
store the number of bytes sent
      &port->m_ov);                  // Cverlapped structure
      // deal with any error codes
      if (!bResult)
      {DWORD dwError = GetLastError();
      switch (dwError) {
      case ERROR_IO_PENDING:{
      // continue to GetOverlappedResults()
      BytesSent = 0;
      bWrite = FALSE;
      break; }    default:
      {// all other error codes                  port-
>ProcessErrorMessage("WriteFile()");}}}
      else{
      LeaveCriticalSection(&port->m_csCommunicationSync);}
      } // end if(bWrite)
```

```
if (!bWrite){
      bWrite = TRUE;
      bResult = GetOverlappedResult(port->m_hComm,     // Handle to COMM
port
        &port->m_ov,
// Overlapped structure  &BytesSent,
// Stores number of bytes sent
      TRUE);             // Wait flag
LeaveCriticalSection(&port->m_csCommunicationSync);
// deal with the error code
if (!bResult)
      {
port->ProcessErrorMessage("GetOverlappedResults() in WriteFile()");}    }
// end if (!bWrite)
// Verify that the data size send equals what we tried to send
if (BytesSent != strlen((char*)port->m_szWriteBuffer))
      {TRACE("WARNING: WriteFile() error.. Bytes Sent: %d; Message Length:
%d\n", BytesSent, strlen((char*)port->m_szWriteBuffer));}}
//
// Character received. Inform the owner
//
void CSerialPort::ReceiveChar(CSerialPort* port, COMSTAT comstat)
{
      BOOL  bRead = TRUE;
      BOOL  bResult = TRUE;
      DWORD dwError = 0;
      DWORD BytesRead = 0;
      unsigned char RXBuff;
      for (;;)
      {
// Gain ownership of the comm port critical section.
// This process guarantees no other part of this program
      // is using the port object.
      EnterCriticalSection(&port->m_csCommunicationSync);
// ClearCommError() will update the COMSTAT structure and
// clear any other errors.

bResult = ClearCommError(port->m_hComm, &dwError, &comstat);
LeaveCriticalSection(&port->m_csCommunicationSync);

// start forever loop.  I use this type of loop because I
// do not know at runtime how many loops this will have to
// run. My solution is to start a forever loop and to
// break out of it when I have processed all of the
// data available.  Be careful with this approach and
// be sure your loop will exit.
// My reasons for this are not as clear in this sample
// as it is in my production code, but I have found this
// solutiion to be the most efficient way to do this.

if (comstat.cbInQue == 0)
      {
// break out when all bytes have been read
      break;}
EnterCriticalSection(&port->m_csCommunicationSync);
if (bRead)
{
```

```
bResult = ReadFile(port->m_hComm,
// Handle to COMM port &RXBuff,
// RX Buffer Pointer
      // Read one byte
      &BytesRead,
      // Stores number of bytes read
      &port->m_ov);
      // pointer to the m_ov structure
      // deal with the error code
      if (!bResult)  {
      switch (dwError = GetLastError())
      {
      case ERROR_IO_PENDING:
      {
// asynchronous i/o is still in progress
// Proceed on to GetOverlappedResults();
bRead = FALSE;
break; }default:
      {
// Another error has occured.  Process this error.
port->ProcessErrorMessage("ReadFile()");
break; }    }}
      else{
// ReadFile() returned complete. It is not necessary to call
GetOverlappedResults()
      bRead = TRUE;
      }
            }  // close if (bRead)
            if (!bRead)
            {bRead = TRUE;
            bResult = GetOverlappedResult(port->m_hComm,    // Handle to
COMM port   &port->m_ov,           // Overlapped structure
       &BytesRead,
// Stores number of bytes read
  TRUE);    // Wait flag
            // deal with the error code
      if (!bResult)  {
port->ProcessErrorMessage("GetOverlappedResults() in ReadFile()");      }

            }  // close if (!bRead)
LeaveCriticalSection(&port->m_csCommunicationSync);
// notify parent that a byte was received
::SendMessage((port->m_pOwner)->m_hWnd, WM_COMM_RXCHAR, (WPARAM) RXBuff,
(LPARAM) port->m_nPortNr);
      } // end forever loop}
//
// Write a string to the port
//
void CSerialPort::WriteToPort(char* string){    assert(m_hComm != 0);
      memset(m_szWriteBuffer, 0, sizeof(m_szWriteBuffer));
      strcpy(m_szWriteBuffer, string);

      // set event for write
      SetEvent(m_hWriteEvent);}
//
// Return the device control block
//
```

```
DCB CSerialPort::GetDCB()
{return m_dcb;}
//
// Return the communication event masks
//
DWORD CSerialPort::GetCommEvents()
{return m_dwCommEvents;}
//
// Return the output buffer size
//
DWORD CSerialPort::GetWriteBufferSize()
{     return m_nWriteBufferSize;}
Serialport.cpp
/*
**     FILENAME                CSerialPort.cpp
**
**     PURPOSE                      This class can read, write and watch
one serial port.
**                                  It sends messages to its owner when
something happends on the port
**                                  The class creates a thread for reading
and writing so the main
**                                  program is not blocked.
**     CREATION DATE           15-09-1997
**     LAST MODIFICATION 12-11-1997
**     AUTHOR                       Remon Spekreijse
*/

#include "stdafx.h"
#include "SerialPort.h"
#include <assert.h>
//
// Constructor
//
CSerialPort::CSerialPort()
{
      m_hComm = NULL;
// initialize overlapped structure members to zero
      m_ov.Offset = 0;
      m_ov.OffsetHigh = 0;
      // create events
      m_ov.hEvent = NULL;
      m_hWriteEvent = NULL;
      m_hShutdownEvent = NULL;
      m_szWriteBuffer = NULL;
      m_bThreadAlive = FALSE;
}

//
// Delete dynamic memory
//
CSerialPort::~CSerialPort()
{do{
      SetEvent(m_hShutdownEvent);
      } while (m_bThreadAlive);
TRACE("Thread ended\n");
delete [] m_szWriteBuffer;}
```

```
//
// Initialize the port. This can be port 1 to 4.
//
BOOL CSerialPort::InitPort(CWnd* pPortOwner,    // the owner (CWnd) of the
port (receives message)
                                    UINT  portnr,
// portnumber (1..4)                      UINT  baud,
      // baudrate                      char  parity,        // parity
         UINT  databits,            // databits
         UINT  stopbits,            // stopbits
        DWORD dwCommEvents,
// EV_RXCHAR, EV_CTS etc
      UINT  writebuffersize)  // size to the writebuffer
{
      assert(portnr > 0 && portnr < 5);
      assert(pPortOwner != NULL);

      // if the thread is alive: Kill
      if (m_bThreadAlive)
      {do{
      SetEvent(m_hShutdownEvent);
      } while (m_bThreadAlive);
      TRACE("Thread ended\n");}
// create events
      if (m_ov.hEvent != NULL)
      ResetEvent(m_ov.hEvent);
      m_ov.hEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
if (m_hWriteEvent != NULL)
      ResetEvent(m_hWriteEvent);
      m_hWriteEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
      if (m_hShutdownEvent != NULL) ResetEvent(m_hShutdownEvent);
      m_hShutdownEvent = CreateEvent(NULL, TRUE, FALSE, NULL);
      // initialize the event objects
      m_hEventArray[0] = m_hShutdownEvent;      // highest priority
      m_hEventArray[1] = m_ov.hEvent;
      m_hEventArray[2] = m_hWriteEvent;
      // initialize critical section
      InitializeCriticalSection(&m_csCommunicationSync);
// set buffersize for writing and save the owner
      m_pOwner = pPortOwner;

      if (m_szWriteBuffer != NULL)
      delete [] m_szWriteBuffer;
      m_szWriteBuffer = new char[writebuffersize];
      m_nPortNr = portnr;
      m_nWriteBufferSize = writebuffersize;
      m_dwCommEvents = dwCommEvents;
      BOOL bResult = FALSE;
      char *szPort = new char[50];
      char *szBaud = new char[50];
      // now it critical!
      EnterCriticalSection(&m_csCommunicationSync);
      // if the port is already opened: close it
      if (m_hComm != NULL)
      {
            CloseHandle(m_hComm);
            m_hComm = NULL;}
```

```
        // prepare port strings
        sprintf(szPort, "COM%d", portnr);
        sprintf(szBaud, "baud=%d parity=%c data=%d stop=%d", baud, parity,
databits, stopbits);
        // get a handle to the port
        m_hComm = CreateFile(szPort,                    // communication port string
(COMX)
                                        GENERIC_READ | GENERIC_WRITE,
// read/write types
                0,
// comm devices must be opened with exclusive access
                                        NULL,
// no security attributes
                                        OPEN_EXISTING,                          //
comm devices must use OPEN_EXISTING
                                        FILE_FLAG_OVERLAPPED,                   //
Async I/O
                0);                     // template must be 0 for comm devices
if (m_hComm == INVALID_HANDLE_VALUE)
        {// port not found
                delete [] szPort;
                delete [] szBaud;
                return FALSE;}
        // set the timeout values
m_CommTimeouts.ReadIntervalTimeout = 1000;
m_CommTimeouts.ReadTotalTimeoutMultiplier = 1000;
m_CommTimeouts.ReadTotalTimeoutConstant = 1000;
m_CommTimeouts.WriteTotalTimeoutMultiplier = 1000;
m_CommTimeouts.WriteTotalTimeoutConstant = 1000;

        // configure
        if (SetCommTimeouts(m_hComm, &m_CommTimeouts))
        {
        if (SetCommMask(m_hComm, dwCommEvents)){
if (GetCommState(m_hComm, &m_dcb))
{
m_dcb.fRtsControl = RTS_CONTROL_ENABLE;         // set RTS bit high!
        if (BuildCommDCB(szBaud, &m_dcb))
                {
if (SetCommState(m_hComm, &m_dcb))
; // normal operation... continue
        else
        ProcessErrorMessage("SetCommState()");}
else            ProcessErrorMessage("BuildCommDCB()");}
        else
                                ProcessErrorMessage("GetCommState()");}else
        ProcessErrorMessage("SetCommMask()");}
        else
ProcessErrorMessage("SetCommTimeouts()");
delete [] szPort;
delete [] szBaud;

        // flush the port
        PurgeComm(m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR | PURGE_RXABORT |
PURGE_TXABORT);

        // release critical section
```

```
      switch (dwError = GetLastError()) {
      case ERROR_IO_PENDING:  {
// This is a normal return value if there are no bytes
// to read at the port.
// Do nothing and continue
      break;      }
      case 87:
            {     // Under Windows NT, this value is returned for some
reason.     // I have not investigated why, but it is also a valid reply
      // Also do nothing and continue.          break;      }
      default:    {     // All other error codes indicate a serious error
has
      // occured.  Process this error.
port->ProcessErrorMessage("WaitCommEvent()");
break;      }}}
      else{// If WaitCommEvent() returns TRUE, check to be sure there are
      // actually bytes in the buffer to read.
      //
      // If you are reading more than one byte at a time from the buffer
      // (which this program does not do) you will have the situation
occur
      // where the first byte to arrive will cause the
WaitForMultipleObjects()
      // function to stop waiting.  The WaitForMultipleObjects() function
      // resets the event handle in m_OverlappedStruct.hEvent to the non-
signelead state
      // as it returns.
//// If in the time between the reset of this event and the call to
// ReadFile() more bytes arrive, the m_OverlappedStruct.hEvent handle will
be set again
// to the signeled state. When the call to ReadFile() occurs, it will
// read all of the bytes from the buffer, and the program will
// loop back around to WaitCommEvent().
                  //
                  // At this point you will be in the situation where
m_OverlappedStruct.hEvent is set,
// but there are no bytes available to read.  If you proceed and call
// ReadFile(), it will return immediatly due to the async port setup, but
// GetOverlappedResults() will not return until the next character
arrives.
                  //
                  // It is not desirable for the GetOverlappedResults()
function to be in
                  // this state.  The thread shutdown event (event 0) and
the WriteFile()
                  // event (Event2) will not work if the thread is blocked
by GetOverlappedResults().

// The solution to this is to check the buffer with a call to
ClearCommError().
// This call will reset the event handle, and if there are no bytes to
read
// we can loop back through WaitCommEvent() again, then proceed.
// If there are really bytes to read, do nothing and proceed.

                  bResult = ClearCommError(port->m_hComm, &dwError,
&comstat);
```

```
      LeaveCriticalSection(&m_csCommunicationSync);
      TRACE("Initialisation for communicationport %d completed.\nUse
Startmonitor to communicate.\n", portnr);
      return TRUE;}
//
//  The CommThread Function.
//
UINT CSerialPort::CommThread(LPVOID pParam)
{
      // Cast the void pointer passed to the thread back to
      // a pointer of CSerialPort class
      CSerialPort *port = (CSerialPort*)pParam;

// Set the status variable in the dialog class to
// TRUE to indicate the thread is running.
port->m_bThreadAlive = TRUE;

// Misc. variables
DWORD BytesTransfered = 0;
      DWORD Event = 0;
      DWORD CommEvent = 0;
      DWORD dwError = 0;
      COMSTAT comstat;
      BOOL  bResult = TRUE;

      // Clear comm buffers at startup
      if (port->m_hComm)          // check if the port is opened
            PurgeComm(port->m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR |
PURGE_RXABORT | PURGE_TXABORT);

      // begin forever loop.  This loop will run as long as the thread is
alive.
      for (;;)
      {

            // Make a call to WaitCommEvent().  This call will return
immediatly
            // because our port was created as an async port
(FILE_FLAG_OVERLAPPED
            // and an m_OverlappedStructerlapped structure specified).
This call will cause the
            // m_OverlappedStructerlapped element
m_OverlappedStruct.hEvent, which is part of the m_hEventArray to
            // be placed in a non-signeled state if there are no bytes
available to be read,
            // or to a signeled state if there are bytes available.  If
this event handle
            // is set to the non-signeled state, it will be set to
signeled when a
            // character arrives at the port.

            // we do this for each port!

            bResult = WaitCommEvent(port->m_hComm, &Event, &port->m_ov);
if (!bResult)  {
// If WaitCommEvent() returns FALSE, process the last error to determin
      // the reason..
```

```
                  if (comstat.cbInQue == 0)
      continue;
            }     // end if bResult
// Main wait function.  This function will normally block the thread//
until one of nine events occur that require action.
      Event = WaitForMultipleObjects(3, port->m_hEventArray, FALSE,
INFINITE);
switch (Event){case 0:{
      // Shutdown event.  This is event zero so it will be
      // the higest priority and be serviced first.
port->m_bThreadAlive = FALSE;
      // Kill this thread.  break is not needed, but makes me feel better.
      AfxEndThread(100);
      break; }
      case 1:     // read event     {
                        GetCommMask(port->m_hComm, &CommEvent);
      if (CommEvent & EV_CTS)
                              ::SendMessage(port->m_pOwner->m_hWnd,
WM_COMM_CTS_DETECTED, (WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RXFLAG)
                              ::SendMessage(port->m_pOwner->m_hWnd,
WM_COMM_RXFLAG_DETECTED, (WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_BREAK)          ::SendMessage(port->m_pOwner-
>m_hWnd, WM_COMM_BREAK_DETECTED, (WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_ERR)
      ::SendMessage(port->m_pOwner->m_hWnd, WM_COMM_ERR_DETECTED, (WPARAM)
0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RING)           ::SendMessage(port->m_pOwner-
>m_hWnd, WM_COMM_RING_DETECTED, (WPARAM) 0, (LPARAM) port->m_nPortNr);
      if (CommEvent & EV_RXCHAR)
      // Receive character event from port.
                              ReceiveChar(port, comstat);
      break; }
            case 2: // write event
      { // Write character event from port
      WriteChar(port);
                        break;
                  }

            } // end switch

      } // close forever loop
      return 0;}
//
// start comm watching
//
BOOL CSerialPort::StartMonitoring()
{
      if (!(m_Thread = AfxBeginThread(CommThread, this)))
            return FALSE;
      TRACE("Thread started\n");
      return TRUE;
}
//
// Restart the comm thread
//
```

```
BOOL CSerialPort::RestartMonitoring()
{
	TRACE("Thread resumed\n");
	m_Thread->ResumeThread();
	return TRUE;
}
//
// Suspend the comm thread
//
BOOL CSerialPort::StopMonitoring()
{
	TRACE("Thread suspended\n");
	m_Thread->SuspendThread();
	return TRUE;
}


//
// If there is a error, give the right message
//
void CSerialPort::ProcessErrorMessage(char* ErrorText)
{
	char *Temp = new char[200];
	LPVOID lpMsgBuf;
	FormatMessage(
FORMAT_MESSAGE_ALLOCATE_BUFFER | FORMAT_MESSAGE_FROM_SYSTEM,
	NULL,
	GetLastError(),
MAKELANGID(LANG_NEUTRAL, SUBLANG_DEFAULT), // Default language
		(LPTSTR) &lpMsgBuf,
		0,
		NULL );

	sprintf(Temp, "WARNING:  %s Failed with the following error:
\n%s\nPort: %d\n", (char*)ErrorText, lpMsgBuf, m_nPortNr);
	MessageBox(NULL, Temp, "Application Error", MB_ICONSTOP);

	LocalFree(lpMsgBuf);
	delete[] Temp;
}

//
// Write a character.
//
void CSerialPort::WriteChar(CSerialPort* port)
{
	BOOL bWrite = TRUE;
	BOOL bResult = TRUE;

	DWORD BytesSent = 0;

	ResetEvent(port->m_hWriteEvent);

	// Gain ownership of the critical section
	EnterCriticalSection(&port->m_csCommunicationSync);

	if (bWrite)
```

```
      {
            // Initailize variables
            port->m_ov.Offset = 0;
            port->m_ov.OffsetHigh = 0;

            // Clear buffer
            PurgeComm(port->m_hComm, PURGE_RXCLEAR | PURGE_TXCLEAR |
PURGE_RXABORT | PURGE_TXABORT);

            bResult = WriteFile(port->m_hComm,
            // Handle to COMM Port
                                        port->m_szWriteBuffer,
                  // Pointer to message buffer in calling finction
                                        strlen((char*)port-
>m_szWriteBuffer),      // Length of message to send        &BytesSent,
      // Where to store the number of bytes sent
      &port->m_ov);                  // Overlapped structure
// deal with any error codes
      if (!bResult)     {
      DWORD dwError = GetLastError();
      switch (dwError)  {
      case ERROR_IO_PENDING:        {
      // continue to GetOverlappedResults()
                                    BytesSent = 0;
                                    bWrite = FALSE;
                                    break;
                              }
      default:
      {
            // all other error codes
            port->ProcessErrorMessage("WriteFile()");
                              }
            }} else{
LeaveCriticalSection(&port->m_csCommunicationSync);}
      } // end if(bWrite)
      if (!bWrite){
      bWrite = TRUE;
      bResult = GetOverlappedResult(port->m_hComm,     // Handle to COMM
port
&port->m_ov,            // Overlapped structure
                              &BytesSent,         // Stores
number of bytes sent                        TRUE);                  //
Wait flag

      LeaveCriticalSection(&port->m_csCommunicationSync);
// deal with the error code
if (!bResult)
{port->ProcessErrorMessage("GetOverlappedResults() in WriteFile()");
            }     } // end if (!bWrite)
// Verify that the data size send equals what we tried to send
if (BytesSent != strlen((char*)port->m_szWriteBuffer))
      {
      TRACE("WARNING: WriteFile() error.. Bytes Sent: %d; Message Length:
%d\n", BytesSent, strlen((char*)port->m_szWriteBuffer));     }}
//
// Character received. Inform the owner
//
```

```
void CSerialPort::ReceiveChar(CSerialPort* port, COMSTAT comstat)
{
      BOOL  bRead = TRUE;
      BOOL  bResult = TRUE;
      DWORD dwError = 0;
      DWORD BytesRead = 0;
      unsigned char RXBuff;
      for (;;)
      {     // Gain ownership of the comm port critical section.
      // This process guarantees no other part of this program
      // is using the port object.
      EnterCriticalSection(&port->m_csCommunicationSync);
// ClearCommError() will update the COMSTAT structure and
// clear any other errors.
bResult = ClearCommError(port->m_hComm, &dwError, &comstat);

            LeaveCriticalSection(&port->m_csCommunicationSync);

// start forever loop.  I use this type of loop because I
// do not know at runtime how many loops this will have to
// run. My solution is to start a forever loop and to
// break out of it when I have processed all of the
// data available.  Be careful with this approach and
// be sure your loop will exit.
// My reasons for this are not as clear in this sample
// as it is in my production code, but I have found this
// solutiion to be the most efficient way to do this.

      if (comstat.cbInQue == 0)
      {
// break out when all bytes have been read
      break;}
      EnterCriticalSection(&port->m_csCommunicationSync);
            if (bRead){
      bResult = ReadFile(port->m_hComm,         // Handle to COMM port
               &RXBuff,                  // RX Buffer Pointer
               1,                        // Read one byte
         &BytesRead,                      Stores number of bytes read
                                          &port->m_ov);          //
pointer to the m_ov structure
                  // deal with the error code
            if (!bResult)                 {
            switch (dwError = GetLastError()) {
            case ERROR_IO_PENDING:        {
      // asynchronous i/o is still in progress
      // Proceed on to GetOverlappedResults();
      bRead = FALSE;
      break;                        }default:   {                          //
Another error has occured.  Process this error. port-
>ProcessErrorMessage("ReadFile()");
      break;                                }}}else{
// ReadFile() returned complete. It is not necessary to call
GetOverlappedResults()
      bRead = TRUE; }
      }  // close if (bRead)
if (!bRead) {bRead = TRUE;
```

```
      bResult = GetOverlappedResult(port->m_hComm,     // Handle to COMM
port
  &port->m_ov,     // Overlapped structure&BytesRead,
if (!bResult)  {port->ProcessErrorMessage("GetOverlappedResults() in
ReadFile()");}     }  // close if (!bRead)
      LeaveCriticalSection(&port->m_csCommunicationSync);
// notify parent that a byte was received
            ::SendMessage((port->m_pOwner)->m_hWnd, WM_COMM_RXCHAR,
(WPARAM) RXBuff, (LPARAM) port->m_nPortNr);
      } // end forever loop}
//
// Write a string to the port
//
void CSerialPort::WriteToPort(char* string)
{assert(m_hComm != 0);
memset(m_szWriteBuffer, 0, sizeof(m_szWriteBuffer));
      strcpy(m_szWriteBuffer, string);
      // set event for write
      SetEvent(m_hWriteEvent);
}
//
// Return the device control block
//
DCB CSerialPort::GetDCB()
{
      return m_dcb;
}
//
// Return the communication event masks
//
DWORD CSerialPort::GetCommEvents()
{return m_dwCommEvents;
}
//
// Return the output buffer size
//
DWORD CSerialPort::GetWriteBufferSize()
{return m_nWriteBufferSize;
}
gvdlkigc[
```

loingin.html

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-874">
<script language="JavaScript" type="text/JavaScript">
<!--
function MM_reloadPage(init) {  //reloads the window if Nav4 resized
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();
}
```

```
MM_reloadPage(true);
//-->
</script>
</head>

<body>
<p> </p>
<p> </p>
<p> </p>
<table width="75%" border="0" align="center">
  <tr>
    <td colspan="3"><form name="form1" method="post"
action="checkpassweb.php">   <!-- bookingtest.php-->
        <table width="75%" border="0">
          <tr>
            <td height="26">UserName :</td>
            <td><input name="username" type="text" width"150" ></td>
            <td> </td>
            <td><input type="submit" name="submit" value="Submit"></td>
          </tr>
          <tr>
 <td >height="26">Password  : </td>
            <td><input name="password" type="password" width="150" ></td>
            <td> </td>
            <td><input type="reset"   name="clear" value="  Clear "></td>
          </tr>
        </table>
        <tr> </tr>
      </form></td>
  </tr>
  <tr>
    <td width="22%">  </td>
    <td width="37%"> </td>
    <td width="41%"><a href="/park%20html/signin.php"><font size="2"
face="Microsoft Sans Serif, MS Sans Serif, sans-
serif">newmember</font></a></td>
  </tr>
  <tr>
    <td colspan="3"> </td>
  </tr>
</table>
</body>
</html>
```

chekpasswoeb.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-874">

</head>
<body>
<div align="center">
  <?
            echo '<table width="30%" border="1" align="center"><tr><td >';
```

```
            $host="localhost";
            $databasename="project";

            mysql_connect($host) or  die("Can not connect to DB.");
            $result = mysql_db_query($databasename,"select
username,password from login where username='$username' ")or die("can not
call data from table  login");
                                        //    echo "hello","<br/> ";

            if($row = mysql_fetch_array($result))
                  {
            //ตรวจสอบว่ารหัสผ่านถูกต้องหรือไม่ ?
      if($row["password"] == $password)
      {
      echo'Well come to check status.<br/>';
            $sql = "select*from park";
            $result = mysql_query($sql);

while($row = mysql_fetch_array($result))
echo $row[0],"    ",$row[1],"       
",$row[2],"<br/><br/>" ;
      $strg =array( '<a
href="/park%20html/changepsw.php',"?username=$username&password=$password"
,'"> Change Password </a> ');
      echo $strg[0],$strg[1],$strg[2];
result = mysql_db_query($databasename,"SELECT  m.booking FROM member AS m
,login AS l WHERE l.username='$username' and  m.id=l.id ") ; $row =
mysql_fetch_array($result);
if($row[0] != "not"){
echo  '<br/>   ','Booking park <br/>(you have booked the park
',$row[0],'.)';}
else{echo $row[0],"test";
      $strg = array('<a
href="/park%20html/booking.php',"?username=$username&password=$password",'
">Booking park </a>');
            echo '<div
align="right">',$strg[0],$strg[1],$strg[2],'</div>';}}else{
echo"So sorry your password miss match";
echo'<a href=" /park html/login.html ">Back</a><br/>';
//    echo $row[0],"  ",$row[1],"<br/>$username,$password";        }}else{
echo " Sorry  username not found ";}
      mysql_close();
?></td>
  </tr> </table> </div>
</body>
</html>
```

signin.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-874">
<script language="JavaScript" type="text/JavaScript">
<!--function MM_reloadPage(init) {  //reloads the window if Nav4 resized
```

```
  if (init==true) with (navigator) {if
((appName=="Netscape")&&(parseInt(appVersion)==4)) {
    document.MM_pgW=innerWidth; document.MM_pgH=innerHeight;
onresize=MM_reloadPage; }}
  else if (innerWidth!=document.MM_pgW || innerHeight!=document.MM_pgH)
location.reload();}
MM_reloadPage(true);
//-->
</script>
</head>

<body topmargin="0" marginheight="0">
<?php
if($cusname == null  or $username == null  or  $password == null  ||
$email == null){

?>
<table width="950" border="0" align="center" cellpadding="0"
cellspacing="0">
  <!--DWLayoutTable-->
  <tr>
    <td height="55"><p> </p>
      <p> </p>
      <p> </p>
      <p> </p>
      <p> </p></td>
    <td> </td>
  </tr>
  <tr>
    <td width="843" height="55"><form name="form1" method="post"
action="signin.php">
        <table width="75%" border="0" align="center">
          <tr>
            <td width="35%"> </td>
            <td colspan="2"><font color="#3366FF">Name : </font></td>
            <td width="46%"><input name="cusname" type="text" id="name"
value="<? echo "$cusname"?>"><font color="#FF0000" ><?php if($cusname ==
null ) echo "*";?></font></td>
          </tr><tr> <td> </td>
            <td colspan="2"><font color="#3366FF">Surname : </font></td>
            <td><input name="surname" type="text" id="surname" value="<?
echo "$surname"?>"></td>
          </tr><tr> <td> </td>
            <td colspan="2"><font color="#3366FF">Username :</font></td>
            <td><input name="username" type="text" id="username" value="<?
echo"$username"?>"><font color="#FF0000" ><?php if($username == null )
echo "*";?></font></td>
          </tr>  <tr> <td> </td>
            <td colspan="2"><font color="#3366FF">Password : </font></td>
            <td><input name="password" type="password" id="password"
><font color="#FF0000" ><?php if($password == null) echo
"*";?></font></td>
          </tr>
          <tr>
            <td> </td>
            <td colspan="2"><font color="#3366FF">Address :</font></td>
```

```
            <td><input name="address" type="text" id="address" value="<?
echo"$address"?>"></td>
          </tr>
          <tr>
            <td> </td>
            <td colspan="2"><font color="#3366FF">Phone : </font></td>
            <td><input name="phone" type="text" id="phone" value="<?
echo"$phone"?>"></td>
          </tr>
          <tr>
            <td> </td>
            <td colspan="2"><font color="#3366FF">Email :</font></td>
            <td><input name="email" type="text" id="email" value="<? echo
"$email"?>"><font color="#FF0000" ><?php if($email== null)echo
"*";?></font></td>
          </tr>
          <tr>
            <td> </td>
            <td colspan="2"> </td>
            <td> </td>
          </tr>  <tr>
            <td> </td>
            <td width="7%">  </td>
            <td width="12%"><input type="submit" name="Submit"
value="Signin"></td>
            <td><input type="reset" name="Submit2" value="  Clear "></td>
          </tr></table>  </form></td>
    <td width="107"> </td>
  </tr> <tr>
    <td height="222" valign="top"><!--DWLayoutEmptyCell--> </td>
    <td> </td>
  </tr></table><?php
}else{  ?>
<br><br><br><br><br><br>
<?    $host="localhost";                $databasename="project";
      mysql_connect($host) or  die("Can not connect to DB.");
                    //          echo  "<br/>$username $cusname <br/>";
                                $memberdata = array( customername =>
"$cusname", surname =>"$surname", username =>"$username",
password=>"$password",address=>
"$address",phone=>"$phone",email=>"$email");
                                $result =
mysql_db_query($databasename,"INSERT INTO`member`(`ID` , `NAME` ,
`SURNAME` , `ADDRESS` , `PHONE` , `EMAIL` , `BOOKING`  )VALUES('',
'$memberdata[customername]', '$memberdata[surname]',
'$memberdata[address]', '$memberdata[phone]', '$memberdata[email]',
'not')")or die("can not call data from table  login");
                                $result =
mysql_db_query($databasename,"INSERT INTO`login`(`ID` , `USERNAME` ,
`PASSWORD` )VALUES('', '$memberdata[username]',
'$memberdata[password]')")or die("can not call data from table  login");

?>
<table width="52%" align="center">
  <tr>
    <td colspan="2">
      <?
```

```
      $result = mysql_query("select* from member  " );
      if($row = mysql_fetch_array($result))
      {echo'Succes to insert member information  .<br/>';   $sql = "select
name,surname,address,phone,email from member where name='$cusname' limit
1";
      $result = mysql_query($sql);
      while($row = mysql_fetch_array($result))
echo $row[0]," ",$row[1]," ",$row[2]," ",$row[3]," ",$row[4],"
",$row[5],"",$row[6],"<br/>" ;            }else{
      echo " Sorry  can not insert into MEMBER table.";
                              }mysql_close();  ?>
    </td></tr>  <tr>
    <td width="79%">  </td>
    <td width="21%">
      <?
            echo ' <a href="/park%20html/login.html"> login </a>
';?></td></tr>
</table>
<?}?>
</body>
</html>
```

เอกสาร booking.php

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-874">
</head>

<body leftmargin="0" topmargin="0">
<?php
//echo "$booking , , , , $username";
if($send==null){
?>
<table width="52%" border="1" align="center">
  <tr>
    <td colspan="2">
      <?php                          $host="localhost"
      $databasename="project";          mysql_connect($host) or
die("Can not connect to DB.");

      ?>
    </td>
  </tr>
  <tr>
    <td><div align="center">Position
          Status &n
bsp;          
        Time</div></td>
    <td><div align="center">Booking</div></td>
  </tr>
  <tr>
    <td width="68%" height="442" rowspan="2">
      <div align="center"><?php
                                 $result =
mysql_db_query($databasename,"select ground,status,time from  park");
```

```
                                        while($row =
mysql_fetch_array($result)){
                                        echo $row[0],"    
",$row[1],"        
",$row[2],"<br><br><br>";
}?></div></td>
    <td width="32%" height="442"> <form name="form1" method="post"
action="booking.php" >
        <p align="center"><font size="2"> <br>
          <br>
          <br>
          <label></label>
          <?php $result = mysql_db_query($databasename,"select status from
park");
                          $i = 0 ;
                          while($row =mysql_fetch_array($result)){
                          $status [$i] = $row[0];
                          $i++;       }          ?>
          <br>                          (
          <br>
          </font></p>
        <p align="center"><font size="2">
          <label><font size="2">NUMBER1</font></label>
          <?php if(($status[0]  !=  "reserved") and  ($status[0] !=
"NO"))echo '<input type="radio" name="booking" value="number1">'; ?>
          <br>
          <br>
          <br>
          </font></p>
        <p align="center"><font size="2">
          <label></label>
          NUBBER2
          <?php  if($status[1] != "reserved" and $status[1] != NO) echo
'<input type="radio" name="booking" value="number2">'; ?>
          <br>
          <br>
          </font></p>
        <p><font size="2">
          <label></label>
          <br>
          </font><font size="2">
          <label></label>
          </font></p>
        <p align="center"><font size="2"> </font><font size="2">
          <label>NUMBER3</label>
          <?php if($status[2] != "reserved" and $status[2] != NO) echo
'<input type="radio" name="booking" value="number3">'; ?>
          </font><br>
          <br>
          <br>
        </p>
        <p align="center"> <font size="2">
          <label></label>
          <label>NUMBER4</label>
          <?php if($status[3] != "reserved"  and $status[3] != NO ) echo
'<input type="radio" name="booking" value="number4">'; ?>
          <br>
```

```
            <br>
            <br>
            </font></p>
          <p align="center"><font size="2">
            <label></label>
            <label>NUMBER5</label>
            <?php if($status[4] != "reserved" and $status[4] != NO) echo
'<input type="radio" name="booking" value="number5">'; ?>
            <br>
            <br>
            <br>
            </font></p>
          <p align="center"><font size="2">
            <label></label>
            <label>NUMBER6</label>
            <?php if($status[5] != "reserved" and $status[5] != NO) echo
'<input type="radio" name="booking" value="number6">'; ?>
            <br>
            <br>
            <br>
            </font></p>
          <p align="center"><font size="2">
            <label></label>
            <label>NUMBER7</label>
            <?php if(($status[6] != "reserved") and ($status[6] != NO)) echo
'<input type="radio" name="booking" value="number7">'; ?>
            <br>
            <br>
            <br>
            </font></p>
          <p align="center"><font size="2"> <font size="2">
            <label>NUMBER8</label>
            <?php if(($status[7] != "reserved") and ($status[7] != "NO"))
echo '<input type="radio" name="booking" value="number8">'; ?>
            </font> <br>
            </font></p>
          <p align="center"><br>
            <input type="reset" name="Submit2" value="  Clear ">
            <input name="send" type="submit" value="Send">

                <input type="hidden" name="username"value= "<?php echo
$username?>">
              <input type="hidden" name="password"value= "<?php echo
$password?>">
            <br>
        </p>
        </form></td>
  </tr>
</table>

<?php
}
else{                                   $host="localhost";
      $databasename="project";                mysql_connect($host) or
die("Can not connect to DB.");                          echo
'</p><p><br><table width="40%" border="1" align="center"><tr><td
colspan="2"> ';
```

```
                                        echo "Position reserved $booking
<br/>";
                                        $result =
mysql_db_query($databasename,"SELECT m.name FROM member AS m , login AS l
WHERE l.username='$username' AND m.id=l.id ")or die("can not call data
from table login");
while ($row = mysql_fetch_array($result))
          $name = $row[0];
$result = mysql_db_query($databasename,"UPDATE `member` SET `booking` =
'$booking' WHERE `name` = '$name'  ")or die("can not call data from table
login");
$result = mysql_db_query($databasename,"UPDATE `park` SET `status` =
'reserved',`timeout` = NOW('time()+3600'), `customer`='$username'WHERE
`ground` = '$booking'  ")or die("can not call data from table login");

if($result)                    {     echo'<br/><br>Your Information<br>';
     $sql = "SELECT m.name,m.surname,m.address,m.phone,m.email FROM
member  AS  m ,login AS l WHERE l.username='$username' LIMIT 1";
     $result = mysql_query($sql);
while($row = mysql_fetch_array($result))
     echo $row[0]," ",$row[1]," ",$row[2]," ",$row[3]," ",$row[4],"
",$row[5],"<br/>" ;
$strg = array('<a
href="/park%20html/checkpassweb.php',"?username=$username&password=$passwo
rd",'">show status </a>');
     echo '<div align="right">',$strg[0],$strg[1],$strg[2],'</div>';
     }else{
echo " Sorry  can not insert into MEMBER table.";
}
mysql_close();                         echo '</td></tr></table>';
}
?>
</body>
</html>
```

เอกสาร changepsw.php

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=windows-874">
</head>
<body>
<?php
$oldusername  = $username ;
if( $newusername==null  or
$newpassword==null){session_register("$num","$username","$newusername","$p
assword","$newpassword");
?>
<table width="320" border="1" align="center">
<tr><td width="323"><form name="from1" method="post" action="<? echo
$PHP_SELF ?>">
  <br>
<? $host="localhost";                  $databasename="project";
```

```
      mysql_connect($host) or  die("Can not connect to DB.");
                                    $result =
mysql_db_query($databasename,"select  booking from member") ;
                                    if(($row = mysql_fetch_array($result))
!= "not"){}?>
           name
            &n
bsp;  :    <? echo "$oldusername" ?><br>
         
password           
:    <? echo "$password"?><br>
          new name         :
                  <input type="text" name="newusername" value="<?php echo
"$newusername";?>">
        <font color="#FF0000"><?p..p if($newusername==null)echo
"*";?></font>
        <br>
           new password    :
        <input type="password" name="newpassword" value="<?php echo
"$newpassword";?>">
        <font color="#FF0000">
        <?php if($newpassword ==null)echo"*";?>
        </font>
        <p>
          <?  if($send !=null){
                                if($newusername ==null )echo "Please fill
your new name.";

                                    if($newpassword ==null)echo "Please
fill your new password.";
}?>  <br><br>
            &n
bsp;         
          <input type="hidden" name="username" value="<?php echo
"$username";?>">
          <input type="hidden" name="password" value="<?php echo
"$password";?>">
          <input type="submit" name="send" value="send">
          <input type="reset"  value="clear">
        </p>
        </form>
     </td>
  </tr></table>
<?php
      }else{
      echo '<table width="320" border="1" align="center">
      <tr><td width="323">';
      session_start();
      $host="localhost";
      $user="root";
      $databasename="project";
      $link= mysql_connect($host ,$user, "");
      echo '<div align="center">Succes updating your personal login
</div><br>';

      if ($link){
```

```
            $sql="update `login` set
`username`='$newusername',`password`='$newpassword' where
`username`='$username' ";
            $result = mysql_db_query($databasename,$sql);
            $sql="select  * from login where username='$newusername'
limit 1 "; $result = mysql_query($sql);
if ($result ){
while($row = mysql_fetch_array($result)){
echo  "#ID :",$row[0]," -- Username :",$row[1]," -- Password
:",$row[2],"<br>";
      }$strg = array('<a
href="/park%20html/checkpassweb.php","?username=$newusername&password=$new
password",'">Go to main page </a>');echo '<div
align="right">',$strg[0],$strg[1],$strg[2],'</div>';

      }else {echo "can't show table login ";
      }}
      else{
      echo "ไม่สามารถเชื่อมต่อกับmysql server ได้"; }
      session_destroy();
      echo '</td></tr></table>';
      }?>
</body>
</html>
```