รายงานการวิจัย

ระบบรอเรียกตามบัตรคิวแจ้งเตือนด้วย SMS

QUEUING SYSTEM WITH SMS ALERT

ชื่อผู้วิจัย

1. รศ.สมยศ จุณณะปิยะ
2. ผศ.ดร.พิพัฒน์ พรหมมี

ได้รับทุนสนับสนุนงานวิจัยจากเงินรายได้ ประจำปีงบประมาณ 2551

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

รายละเอียดเกี่ยวกับโครงการ

ชื่อโครงการ ระบบบรอเรียกตามบัตรคิวแจ้งเตือนด้วย SMS

QUEUING SYSTEM WITH SMS ALERT

ได้รับทุนอุดหนุนการวิจัยจาก คณะวิศวกรรมศาสตร์

ประจำปี 2551 จำนวนเงิน 100,000 บาท

ระยะเวลาทำวิจัย 1 ปี ตั้งแต่ 1 ตุลาคม 2550 ถึง 30 กันยายน 2551

หน่วยงานและผู้ดำเนินการวิจัยพร้อมหน่วยงานที่สังกัดและเลขหมาย

รศ.สมยศ จุณณะปิยะ

ภาควิชาวิศวกรรมโทรคมนาคม คณะวิศวกรรมศาสตร์

บทคัดย่อ

โครงงานนี้เป็นการนำเสนอระบบบรอเรียกตามบัตรคิวแจ้งเตือนด้วย SMS โดยระบบนี้จะแบ่งออกได้เป็น 2 ส่วน คือ ส่วนของไมโครคอนโทรลเลอร์ และ ส่วนของคอมพิวเตอร์ ในส่วนของไมโครคอนโทรลเลอร์นั้นจะเชื่อมต่อกับปุ่มกดของพนักงานให้บริการ และจอแสดงผลเจ็ดส่วน (7-segment) เพื่อแสดงหมายเลขให้ตรงกับเสียงที่ประกาศตามคิว ในส่วนของคอมพิวเตอร์จะเชื่อมต่อกับเครื่องพิมพ์ เพื่อพิมพ์บัตรคิวและใช้ติดต่อกับผู้ใช้บริการ โดยมีการเล่นเสียงตามหมายเลขที่ให้บริการ ในกรณีที่มีผู้ใช้บริการเป็นจำนวนมาก ระบบนี้สามารถให้บริการในการส่ง SMS เพื่อแจ้งให้ผู้ใช้บริการทราบ เมื่อใกล้ถึงเวลาเข้ารับบริการ

ABSTRACT

This project presents a Queuing System with SMS Alert. This system consists of 2 parts . a microcontroller part and computer part. The microcontroller part uses for interface with their agents and queuing display by 7 - segment. The computer part uses for programming and interfacing with different customers and play a particular sound queue. The SMS alert is able to use in order to a prolong queue for save the valuable time of customers.

สารบัญ

สารบัญรูปภาพ

สารบัญรูปภาพ (ต่อ)

สารบัญตาราง

บทที่ 1
บทนำ

ในปัจจุบัน การใช้บริการต่างๆ มีปัญหาทางด้านการให้บริการด้านการรอรับบริการ เนื่องจากต้องมีการเข้าแถว ทำให้เปลืองพื้นที่ในการใช้สอยไปกับการจัดบริเวณเพื่อเข้าแถว และในช่วงเวลาที่มีจำนวนคนมากๆ จึงเกิดปัญหาทางด้านพื้นที่ สร้างความลำบากในการใช้บริการ สร้างความรำคาญ สูญเสียเวลา และอาจจะก่อให้มีการใช้บริการน้อยลง

ดังนั้นหน่วยงานหรือองค์กรต่างๆ ไม่ว่าจะเป็น ธนาคาร หน่วยงานของรัฐ และหน่วยงานของเอกชน ที่ทำการให้บริการที่เกี่ยวข้องกับบุคคล จึงเล็งเห็นปัญหาที่จำเป็นจะต้องมีเครื่องอำนวยความสะดวกในการรอรับบริการ

โดยระบบรอเรียกตามบัตรคิวแจ้งเตือน SMS เป็นระบบที่ให้ความสะดวก ซึ่งเมื่อผู้ใช้บริการต้องการใช้บริการ จะต้องกดเพื่อรับบัตรคิว และจะมีการเรียกหมายเลขตามลำดับเพื่อเข้ารับบริการ โดยผู้ที่ใช้บริการสามารถนั่งรอ จนกว่าจะมีการเรียก และเมื่อมีการรอเข้าใช้บริการมากๆ ระบบรอเรียกตามบัตรคิวแจ้งเตือน SMS ก็มีการรองรับ โดยมีการให้กรอกหมายเลขโทรศัพท์ เพื่อทำการแจ้งเตือน เมื่อใกล้ถึงการเข้ารับบริการ โดยทำให้ผู้ใช้บริการสามารถที่จะไม่ต้องอยู่ที่จุดรับบริการตลอดเวลาก็ได้ ทำให้สามารถทำภารกิจอื่นๆได้ จนเมื่อได้รับ SMS จึงกลับมารับการให้บริการได้ ทำให้ไม่พลาดการใช้บริการ ซึ่งเป็นการแก้ไขปัญหา คือ เมื่อผู้ใช้บริการไม่ทราบเวลา ทำให้เลยหมายเลขที่ใช้บริการไป ต้องเสียเวลาทำการขอรับบริการใหม่

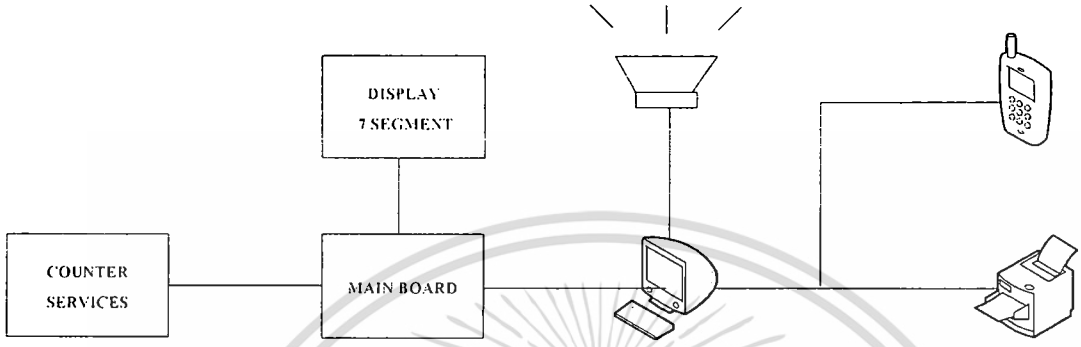
ในชิ้นงานของระบบรอเรียกตามบัตรคิวแจ้งเตือน SMS แบ่งการควบคุมเป็น 2 ส่วนใหญ่ๆ คือ

1.1 คอมพิวเตอร์

1.2 ไมโครคอนโทรลเลอร์

บทที่ 2
หลักการและทฤษฎี

หลักการทำงานของระบบบริหารเรียกตามบัตรคิว เป็นดังต่อไปนี้



รูปที่ 2.1 แสดงการทำงานของระบบบริหารเรียกตามบัตรคิวแจ้งเตือน SMS

2.1 คอมพิวเตอร์

2.1.1 SMS (Short Massage Services)

2.1.1.1 ความหมายของ SMS (Short Massage Services)

SMS หรือ การส่งข้อความสั้น โดยลักษณะของการส่งข้อความสั้นจะมีลักษณะคล้ายกับการส่งข้อความไปยังเพจเจอร์ คือ ผู้ใช้สามารถส่งข้อความไปยังผู้รับ โดยที่ผู้รับสามารถกดอ่านได้จากเครื่องโทรศัพท์มือถือได้ทันที ข้อดีของ SMS ที่ทำให้ต่างกับเพจเจอร์ก็คือ ผู้ใช้หรือผู้ที่ต้องการส่งข้อความสามารถพิมพ์ข้อความได้เองจากโทรศัพท์มือถือ และสามารถส่งไปยังโทรศัพท์มือถือของผู้รับได้ทันที SMS เป็นบริการมาตรฐาน ในการรับส่งข้อความระหว่างโทรศัพท์เคลื่อนที่ และอุปกรณ์อื่นๆสามารถส่งได้ในรูปแบบของตัวเลข, ตัวอักษร และ สัญลักษณ์ต่างๆ SMS ได้ถูกสร้างขึ้นมาครั้งแรกให้ทำงานร่วมกับโทรศัพท์เคลื่อนที่แบบดิจิตอล ระบบ GSM โดยข้อความแรกได้ถูกส่งในเดือนธันวาคม 1992 จากเครื่องคอมพิวเตอร์ส่วนบุคคลไปสู่เครื่องโทรศัพท์บนโครงข่ายระบบ GSM ของ Vodafone ในประเทศอังกฤษ ปัจจุบันบริการ SMS สนับสนุนโครงข่าย GSM , CDMA , และ TDMA สำหรับการส่ง SMS ภาษาไทยจะส่งได้ 70 ตัวอักษร ภาษาอังกฤษส่งได้ 160 ตัวอักษร

เนื่องจาก การรับ-ส่ง SMS เป็นเทคนิคการสื่อสารที่ไม่จำเป็นต้องใช้การสร้างวงจรสนทนา (Call Set-up) จึงทำให้สามารถรับหรือส่งข้อความได้ในขณะที่กำลังสนทนาอยู่ หรือในขณะที่เปิดเครื่องทิ้งไว้เฉยๆ บริการ SMS เป็นบริการที่ได้รับความนิยมมากในปัจจุบัน บริการ SMS มีรายละเอียดดังนี้

ตารางที่ 2.1 รายละเอียดของ SMS

Feature	SMS
Store and Forward (non real time)	Yes
Comfirmation of message delivery	Yes
Communications Type	Person to Person
Media suppored	Text plus binary
Protocols	SMS specific e.g.SMPP
Configuration	Simple telephone number
Platforms	SMS Center
Principle Application	Simple person to person
User behavior	Discrete

บริการ SMS ไม่ใช่บริการแบบ Realtime เนื่องจากการส่งข้อความต้องส่งผ่าน Platform กลาง คือ Short Message Center หรือ SMS-C ซึ่งเป็นอุปกรณ์ที่ผู้ให้บริการเครือข่ายโทรศัพท์เคลื่อนที่ติดตั้งไว้เพื่อให้บริการรับ-ส่งข้อความผ่านทางเครื่องลูกข่ายโทรศัพท์เคลื่อนที่ไปสู่เครื่องลูกข่ายเครื่องอื่นๆ ได้

2.1.1.2 หลักการทำงานของ SMS

SMS เป็นเทคโนโลยี การรับ-ส่งข้อมูลแบบเก็บและส่งต่อ (Store and Forward) ในเครือข่าย GSM เรียกอุปกรณ์ที่ใช้ในการเก็บและส่งต่อข้อมูลว่า Short Message Service Center (SMS-C)

การใช้งาน SMS กระทำได้โดย เมื่อองค์กรต้องการส่งข้อความสั้น (จำนวนมากสุด 160 ตัวอักษร) ก็จะทำการป้อนข้อความ พร้อมทั้งระบุเลขหมายปลายทางที่ต้องการจะส่งไปด้วย แต่เครื่องลูกข่ายที่ต้องการจะส่ง SMS จะต้องระบุเลขหมายของ SMS-C ก่อน จะทำการตรวจสอบเลขหมายปลายทางกับ HLR ว่าเลขหมายปลายทางอยู่ที่ไหนในเครือข่าย เมื่อทราบแล้ว SMS-C ก็จะส่ง SMS ไปยังโทรศัพท์เคลื่อนที่ปลายทาง กรณี SMS ระบุหมายเลขปลายทางเป็นโทรศัพท์เคลื่อนที่นอกเครือข่าย เช่น ส่งจาก DTAC ไป AIS ชุมสายโทรศัพท์เคลื่อนที่ต้นทางในเครือข่าย DTAC (MSC) จะตรวจสอบจากหมายเลขปลายทาง และเมื่อทราบว่าจุดหมายปลายทางเป็นเลขหมายของ AIS ชุมสายโทรศัพท์เคลื่อนที่ (MSC) ของ DTAC จะส่ง SMS ดังกล่าวไปลงที่ SMS-C ของ AIS โดยตรง

2.1.1.3 รูปแบบการให้บริการของผู้ให้บริการเครือข่าย (Operator) ในปัจจุบัน

รูปแบบการให้บริการของ Operator หรือผู้ให้บริการเครือข่ายโทรศัพท์เคลื่อนที่ในปัจจุบัน มี 2 ลักษณะ คือ

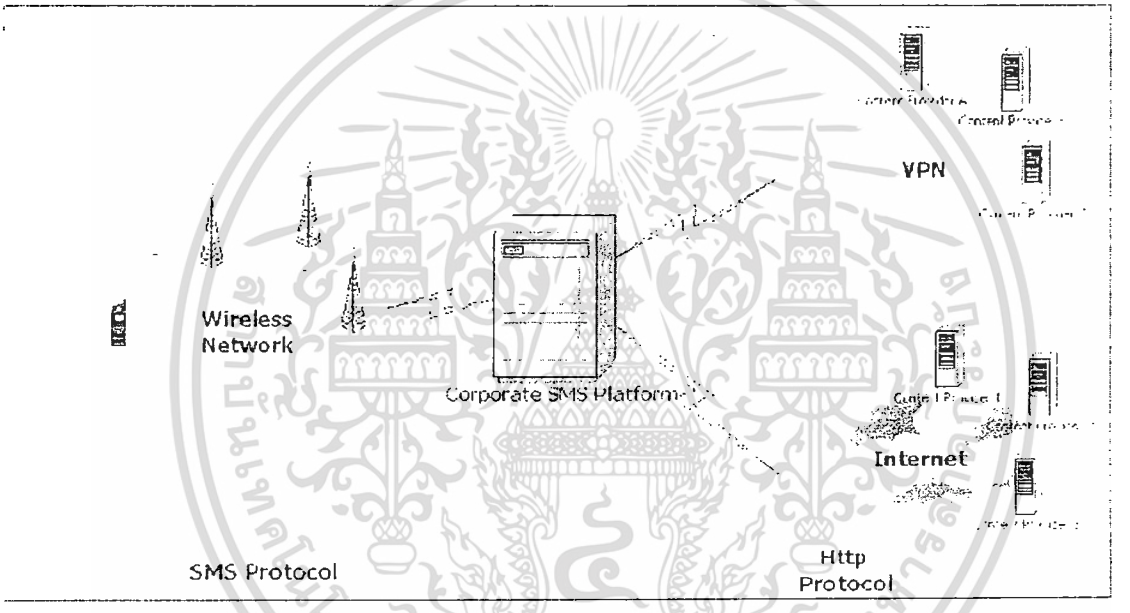
- Bulk SMS

การทำงานผ่าน Corporate SMS Platform เป็นการให้บริการ SMS ในลักษณะองค์กร (Corporate Short Message Service) ลักษณะการใช้งานของ Bulk คือ การส่งจากผู้ส่ง (องค์กร) ไปยัง ผู้รับ (ลูกค้า) ซึ่งมีได้ 2 ลักษณะ คือ One-to-One (ส่งข้อความรายบุคคล) และ One-to-Many (ส่งข้อความจากต้นทางเดียวถึงปลายทางในเวลาเดียวกัน) การเรียกเก็บเงินจะเก็บเงินกับองค์กรที่ใช้บริการ

- CPA SMS

การทำงานผ่าน Content Provider Access Platform (CPA Platform) เป็นการบริการที่ Operator (ผู้ให้บริการเครือข่าย AIS,DTAC,TA,Orange) สร้าง Model Services เอง หรือเปิดโอกาสให้ ตัวแทน (Agents) ที่มีความพร้อมในเรื่องของการสร้าง Model Services หรือที่เรียกว่า Content Provider นำเสนอโครงการให้ Operator พิจารณา เพื่อดำเนินธุรกิจร่วมกัน (Co-pertner) โดย Operator จะเป็นผู้ดำเนินการเก็บค่าบริการให้ ซึ่งการแบ่งส่วนของรายได้ระหว่าง Operator กับ ตัวแทน (Agents) ส่วนใหญ่ในปัจจุบันจะเป็นอัตราส่วนในลักษณะ 50:50 หรือตามที่จะตกลงกัน การเรียกเก็บเงินจะเก็บเงินกับลูกค้าที่ใช้การบริการ

2..1.1.4 โครงสร้างของเครือข่าย (Network Structure)



รูปที่ 2.2 โครงสร้างเครือข่ายการทำงานผ่าน Corporate SMS Platform

Corporate SMS Platform เป็นอุปกรณ์ที่ใช้เป็นตัวกลางในการเชื่อมต่อกับศูนย์กลางการให้บริการการส่งข้อความสั้น (SMSC) โดยมีโปรโตคอล 2 ชนิด คือ

- Http Protocol เป็น Protocol เพื่อใช้ในการเชื่อมต่อกับระบบภายนอก
- SMS Protocol เป็น Protocol เพื่อใช้ในการเชื่อมต่อกับระบบภายในศูนย์กลางการให้บริการการส่งข้อความ

2.1.1.5 ลักษณะของการส่ง SMS

การส่ง SMS หรือ Short Message Service คือ การส่งข้อความสั้นๆ หรือ ข้อมูลสั้นจากเครื่องโทรศัพท์มือถือผู้ส่ง ไปยังเครื่องโทรศัพท์มือถือของผู้รับโดยส่งผ่านเครือข่ายศูนย์บริการ Short Message Service Center (SMSC) โดยการส่งแบบ SMS นี้เราจะสามารถเลือกได้ว่าจะส่งข้อความสั้น หรือ เป็นรูปภาพ โลโก้ หรือเสียงเพลงริงค์โทน ซึ่งจะมีวิธีการส่งที่แตกต่างกัน 2 แบบ คือ โหมดตัวอักษร หรือ

Text-Mode และโหมดพีดียู หรือ PDU (Protocol Data Unit) โดย Text-Mode คือโหมดที่เราสามารถส่งข้อความสั้นๆประมาณ 160 ตัวอักษร ไปยังเครื่องโทรศัพท์มือถือของผู้รับ โดยลักษณะข้อความนั้นจะอยู่ในรูปแบบรหัส ASCII ส่วน PDU-Mode คือโหมดที่สามารถส่งได้ทั้งข้อความสั้นๆ,ส่งรูปภาพ และเพลงริงค์โทนได้ ซึ่ง PDU-Mode จะมารูปแบบการวางข้อมูลที่จะส่งแตกต่างกับ Text-Mode คือ PDU-Mode จะมีการเข้ารหัสที่จะแปลงข้อความเป็นรูปแบบของเลขฐานสิบหก และต้องมีการส่งหัวข้อของชุดข้อมูล (Heading) แต่ใน Text-Mode จะเป็นการส่งแบบรหัส ASCII และไม่จำเป็นต้องส่งหัวข้อของชุดข้อมูล

2.1.1.6 การส่ง SMS แบบ PDU-Mode

ในโครงงานนี้เราจะใช้การส่ง SMS แบบ PDU-Mode ซึ่งรูปแบบการจัดรูปแบบนั้นจะซับซ้อนกว่าแบบ Text-Mode มาก แต่การส่งแบบ PDU-Mode นี้เราสามารถใช้ได้กับโทรศัพท์มือถือได้ทุกรุ่น โดยการส่ง SMS แบบ PDU-Mode มีรายละเอียดดังนี้คือ ใน PDU-Mode นี้จะต้องมีการสร้างหัวข้อของชุดข้อมูลสำหรับส่ง (Heading) ซึ่งประกอบด้วยส่วนของศูนย์บริการ SMSC กับส่วนของชุดข้อความหรือ Transfer Protocol Unit : TPDU โดยทั้งสองส่วนจะมีลักษณะเป็นเลขฐานสิบหกซึ่งจะวางลำดับตามนี้

ตารางที่ 2.2 แสดงส่วนประกอบของชุดข้อมูลในการส่ง SMS แบบ PDU-Mode

Header (Cr)	ส่วนของ SMSC	ส่วนของ TPDU	Stop bit (Crtl-Z)

ในส่วนของ TPDU ก็จะประกอบด้วยส่วนย่อยๆ ซึ่งจะเป็นตัวกำหนดรูปแบบของ SMS ที่จะส่ง โดยถ้าเราต้องการที่จะส่งเป็นข้อความจะต้องจัดรูปแบบเรียงตามนี้

1. โปรโตคอลพารามิเตอร์ คือ พารามิเตอร์ที่บอกว่าโปรโตคอล (Protocol) ที่ใช้ส่งเป็นแบบใด กรณีส่งแบบ TPDU = 0x01
2. ตัวเลขอ้างข้อความ ในกรณีที่มีข้อความหลายๆข้อความ เราสามารถจัดลำดับข้อความโดยใช้ตัวเลขอ้างข้อความได้ (มีค่าปกติ = 0x00)
3. ความยาวของเบอร์โทรศัพท์มือถือของหมายเลขปลายทาง
4. รูปแบบของเบอร์โทรศัพท์มือถือของหมายเลขปลายทาง ซึ่งจะเป็นตัวบอกลักษณะของเบอร์โทรศัพท์มือถือที่เราต้องการส่งข้อความไปให้โดย ส่งแบบสากลจะใช้ค่า = 0x91
5. หมายเลขโทรศัพท์มือถือของหมายเลขปลายทางที่ต้องการจะส่ง โดยหมายเลขโทรศัพท์นี้จะมีการเข้ารหัสแบบสลับ (nibble swapped)
6. ตัวแสดงรูปแบบชุดข้อมูล
7. ลักษณะการเข้ารหัสของข้อมูล คือพารามิเตอร์ที่บอกว่าเราจะส่งเป็นภาษาใด (มาตรฐานคือระบบ GSM)
8. ความยาวของข้อความที่ต้องการส่ง (ก่อนเข้ารหัส)
9. ข้อความที่ต้องการส่ง (หลังเข้ารหัส)

จะเห็นว่าการส่งแบบ PDU-Mode มีการเข้ารหัสที่ซับซ้อน เช่นการเข้ารหัสสลับ (nibble swapped) และการเข้ารหัสของชุดข้อความที่จะส่ง โดยการเข้ารหัสแบบสลับ มีลักษณะดังนี้โดยจะทำการสลับเบอร์โทรศัพท์ที่ติดกันเป็นคู่ๆ และถ้าเหลือเศษจะเติมค่า F เข้าไปก่อนรหัสตัวสุดท้าย เช่น เบอร์โทรศัพท์ คือ 123456789 เมื่อเข้ารหัสสลับแล้วจะกลายเป็น 21436587F9 ส่วนการเข้ารหัสของชุดข้อความจะต้องทำการแปลงข้อความที่เป็น ASCII มาเป็นเลขฐานสองหลักจากนั้นก็ทำการเข้ารหัส

ส่วนของ SMSC จะเป็นส่วนที่กำหนดเครือข่ายการใช้บริการว่า จะใช้บริการผ่านศูนย์บริการ SMSC ใดๆ โดยจะประกอบด้วยส่วนย่อยๆ ดังนี้

1. ความยาวของเบอร์ศูนย์บริการ SMSC
2. รูปแบบของเบอร์ศูนย์บริการ SMSC (ส่งแบบสากลจะใช้ค่า = 0x91)
3. เบอร์ศูนย์บริการ SMSC โดยจะมีการเข้ารหัสแบบสลับ (nibble swapped)

เมื่อผู้รับได้รับ SMS ที่มีการส่งแบบ PDU-Mode รูปแบบของข้อความก็จะอยู่ในลักษณะของ PDU-Mode เราจำเป็นต้องศึกษาถึงรูปแบบของข้อความที่ได้รับดังนี้ คือข้อความที่ได้รับนี้จะประกอบด้วยส่วนสำคัญสองส่วนคือ ส่วนศูนย์บริการ SMSC กับส่วนของชุดข้อความหรือ Transfer Protocol Data Unit (TPDU) โดยทั้งสองส่วนจะมีลักษณะเป็นเลขฐานสิบหกซึ่งจะเหมือนกันการส่ง แต่ชุดข้อมูลบางชุดเพิ่มเติมเข้ามาคือ เวลา วันเดือนปี ที่ได้รับข้อความ และเบอร์โทรศัพท์ของผู้ส่ง ดังนี้

ส่วนของ SMSC เป็นส่วนที่กำหนดเครือข่ายการใช้บริการว่าจะใช้บริการผ่านศูนย์บริการ SMSC ใด รูปแบบลักษณะในส่วนนี้จะคล้ายกับการส่ง โดยจะประกอบด้วยส่วนย่อยๆดังนี้ คือข้อความที่ได้รับนี้จะประกอบด้วยส่วนสำคัญสองส่วนคือ ส่วนศูนย์บริการ SMSC กับส่วนของชุดข้อความหรือ Transfer Protocol Data Unit (TPDU) โดยทั้งสองส่วนจะมีลักษณะเป็นเลขฐานสิบหก ซึ่งจะเหมือนกันการส่ง แต่ละชุดข้อมูลบางชุดเพิ่มเติมเข้ามาคือ เวลา วันเดือนปี ที่ได้รับข้อความ และเบอร์โทรศัพท์ของผู้ส่ง ดังนี้

1. ความยาวของเบอร์ศูนย์บริการ SMSC
2. รูปแบบของเบอร์ศูนย์บริการ SMSC (ส่งแบบสากลจะใช้ค่า = 0x91)
3. เบอร์ศูนย์บริการ SMSC โดยจะมาการเข้ารหัสแบบสลับ (nibble swapped)

ในส่วนของ TPDU ก็จะประกอบด้วยส่วนย่อยๆ ซึ่งจะเป็นตัวกำหนดรูปแบบของ SMS ที่จะส่ง โดยในส่วนนี้จะมีส่วนที่แตกต่างจากการส่งคือเพิ่มเวลา วันเดือนปีที่ได้รับข้อความและเปลี่ยนจากเบอร์ที่ต้องการส่งเป็นเบอร์ที่ส่งมา โดยจัดรูปแบบเรียงตามนี้

1. โปรโตคอลพารามิเตอร์ คือ พารามิเตอร์ที่บอกว่าโปรโตคอล (Protocol) ที่ใช้ส่งเป็นแบบใด กรณีส่งแบบ TPDU = 0x01
2. ตัวเลขอ้างข้อความ ในกรณีที่มีข้อความหลายๆข้อความเราสามารถจัดลำดับข้อความโดยใช้ตัวเลขอ้างข้อความได้ (มีค่าปกติ = 0x00)
3. ความยาวของเบอร์โทรศัพท์มือถือของหมายเลขต้นทาง
4. รูปแบบของเบอร์โทรศัพท์มือถือของหมายเลขต้น ซึ่งจะเป็นตัวบอกลักษณะของเบอร์โทรศัพท์มือถือที่เราต้องการส่งข้อความไปให้โดย ส่งแบบสากลจะใช้ค่า = 0x91

5. หมายเลขโทรศัพท์มือถือของหมายเลขต้นทางที่ต้องการจะส่ง โดยหมายเลขโทรศัพท์นี้จะมีการเข้ารหัสแบบสลับ (nibble swapped)
6. ตัวแสดงรูปแบบชุดข้อมูล
7. ลักษณะการเข้ารหัสของข้อมูล คือพารามิเตอร์ที่บอกว่าเราจะส่งเป็นภาษาใด (มาตรฐานคือระบบ GSM)
8. เวลาและวันเดือนปีที่ได้รับข้อความ (nibble swapped) เช่น 0x99 0x20 0x21 0x50 0x75 0x03 0x21 จะหมายถึง 12. Feb 1999 05:57:30 GMT + 3
9. ความยาวของข้อความที่ต้องการส่ง (ก่อนเข้ารหัส)
10. ข้อความที่ต้องการส่ง (หลังเข้ารหัส)

โดยปกติการส่ง SMS นี้เราสามารถกดส่งจากเครื่องโทรศัพท์มือถือของเราได้ โดยเริ่มจากการเขียนข้อความ เมื่อข้อความเสร็จแล้วจะมีให้เลือกที่จะส่งไปเบอร์โทรศัพท์หมายเลขใด นอกจากที่เราต้องกดส่งจากโทรศัพท์มือถือแล้วเรายังสามารถเลือกที่จะส่ง SMS ได้อีกแบบคือ ในเครื่องโทรศัพท์บางรุ่นที่มีอยู่ในปัจจุบันจะมีพอร์ตอนุกรม ซึ่งเราสามารถใช้พอร์ตอนุกรมนี้เป็นตัวเชื่อมต่อระหว่างคอมพิวเตอร์หรือไมโครคอนโทรเลอร์ได้พอร์ตอนุกรมที่มีนี้ ทำให้เราง่ายในการส่ง SMS อย่างมาก คือเราไม่จำเป็นต้องกดปุ่มที่เครื่องโทรศัพท์มือถือเพียงแค่เราส่งชุดคำสั่งเป็นรหัส ASCII เข้าไปทางพอร์ตอนุกรมนี้เราก็สามารถสั่งงานให้เครื่องโทรศัพท์มือถือส่ง SMS

ตัวอย่างการส่งข้อความ SMS แบบ PDU-Mode

โดยจะทำการส่งข้อความ SMS "hellohello" โดยใช้ PDU-Mode ไปยังหมายเลข "+66 092056208 "

AT+CMGF=0 // เพื่อเลือกโหมดพีดียู

AT+CSMS=0 // เช็คว่ามือถือสนับสนุนการส่ง SMS หรือไม่

AT+CMGS=22 // ต้องการส่งทั้งหมด 22 bytes (ไม่รวมตัวเลข 00 ที่อยู่ข้างหน้าสุด)

>0011000A916629502680000AA0AE8329BFD4697D9EC37 // เมื่อพิมพ์ข้อความครบแล้วกด Ctrl+z ส่วนประกอบของข้อมูลที่ส่งอธิบายในตารางที่ 2.3

ตารางที่ 2.3 ส่วนประกอบของข้อมูลที่ส่ง

กลุ่มตัวเลข 8 บิต (Octet)	รายละเอียด
00	ความยาวของ SMSC Information 00 หมายถึง ให้ใช้ SMSC Information ที่เก็บอยู่ภายในเครื่อง (ปกติเครื่องที่สามารถส่ง SMS ได้จะมีข้อมูล SMSC ภายในเครื่องอยู่แล้ว)
11	First octet of the SMS-SUBMIT message
00	TP-Message-Reference "00" คือให้เครื่องตั้งหมายเลขอ้างอิงข้อความขึ้นเอง
0A	Address-Length ความยาวของเลขหมายผู้รับ (0A hex = 10)

กลุ่มตัวเลข 8 บิต (Octet)	รายละเอียด
91	Type-of-Address (91indicates international format of the phone number)
66 29 50 26 80	เลขหมายผู้รับ (แบบ decimal semi-octets)เป็นเลขฐาน 10 สลับ nibble หมายเลขที่แท้จริง คือ +66 092056208
00	TP-PID (Protocol identifier) เป็น 00
00	TP-DCS (Data coding scheme) เป็น 00
AA	TP-Validity-Period "AA" หมายถึง ช่วงเวลาหมดอายุของข้อความ 4 วัน ถ้าภายในช่วงเวลานี้ ยังส่งไม่ถึงปลายทางข้อความจะถูกยกเลิกโดยอัตโนมัติ
0A	TP-User-Data-Length จำนวนตัวอักษรของข้อความที่ส่ง (10 ตัว)
E8329BFD4697D9EC37	TP-UD ข้อความ "hellohello" ที่เข้ารหัสแล้วจากตัวอักษรแบบ 7 bits เป็นข้อมูล byte ขนาด 8 bits

ตัวอย่างการรับข้อความ SMS แบบ PDU-Mode

ถ้าหากเราเชื่อมต่อกับมือถือแล้วทำการอ่านข้อความ SMS ที่อยู่ใน Inbox โดยใช้คำสั่ง AT+CMGR ข้อมูลที่ได้รับจะอยู่ในรูปของสตริงที่ประกอบไปด้วยข้อมูลของผู้ส่ง,ข้อมูล SMS Service Center (SMSC),Time Stamp และอื่นๆ ที่จำเป็นและตามด้วยส่วนของข้อความซึ่งจะอยู่ท้ายสุดของสตริง ตัวอย่างสตริงต่อไปนี้ข้อความที่ส่งมาคือ "hellohello" จากมือถืออีกเครื่องหนึ่งข้อมูลสตริงนี้จะอยู่ในรูปของตัวเลขฐาน 16 และฐาน 10 (ในบางส่วน) โดยจะเรียกตัวเลขแต่ละคู่ว่า Octet ซึ่งมีรายละเอียดดังตารางที่ 2.4 0691668111808804 0A916629502680000040302121943482 0AE8329BFD4697D9EC37

ตารางที่ 2.4 ส่วนประกอบของสตริงการรับข้อความ SMS

กลุ่มตัวเลข 8 บิต (Octet)	รายละเอียด
06	ความยาวของ SMSC Information 6 Octets (bytes)
91	รูปแบบของเลขหมาย SMSC 91 หมายถึงเลขหมายแบบสากล (international format)
66 81 11 80 88	เลขหมาย SMSC (แบบ decimal semi-octets) ซึ่งจะเป็นเลขฐาน 10 สลับ nibble ในกรณีนี้เลขหมายจริงของ Service Center คือ +6618110888
04	First octet of this SMS-DELIVER message
0A	ความยาวของเลขหมายผู้ส่ง (0A hex = 10)
91	รูปแบบของเลขหมายผู้ส่ง 91 หมายถึง เลขหมายแบบสากล (international format)

กลุ่มตัวเลข 8 บิต (Octet)	รายละเอียด
66 29 50 26 80	เลขหมายผู้ส่ง (แบบ decimal semi-octets)เป็นเลขฐาน 10 สลับ nibble หมายเลขผู้ส่งที่แท้จริง คือ +6692056208
00	TP-PID (Protocol identifier) ในกรณีนี้คือ 00
00	TP-DCS (Data coding scheme) 00 คือเข้ารหัสข้อความแบบ 7 bits Default Alphabet
40 30 21 21 94 34 82	TP-SCTS ข้อมูล Time stamp (แบบ decimal semi-octets) สลับ nibble
0A	TP-UDL User data length จำนวนตัวอักษรของข้อความที่ส่งในที่นี้คือ 10 ตัว
E8329BFD4697D9EC37	TP-UD ข้อความ "hellohello" ที่เข้ารหัสแล้วจากตัวอักษรแบบ 7 bits เป็นข้อมูล byte ขนาด 8 bits

ข้อมูลทั้งหมดในตารางเป็นเลขฐาน 16 ขนาด 8 บิต ยกเว้นหมายเลข Service Center , เลขหมายผู้ส่ง,Timestamp จะเป็นเลขฐาน 10 ขนาด 8 บิตสลับหลักเป็นคู่ๆ (สลับ Nibble) ในส่วนของข้อมูลที่เป็นข้อความนั้นเป็นเลขฐาน 16 ขนาด 8 บิต เช่นกัน โดยข้อมูลนี้จะใช้แสดงข้อความที่ประกอบไปด้วยตัวอักษรขนาด 7 บิต ซึ่งผ่านการแปลง (เข้ารหัส) ข้อมูลจากตัวอักษรขนาด 7 บิต ให้เป็นเลขฐาน 16 ขนาด 8 บิต มาแล้ว ส่วนวิธีการแปลงจะกล่าวในภายหลัง

ในส่วนของข้อมูลที่เป็นเลขฐาน 10 เช่น เลขหมายผู้ส่งตัวเลขในแต่ละคู่ (1 byte) จะถูกสลับหลักกัน เช่น เลขหมายจริง "+ 66 092056208" จะถูกสลับในแต่ละคู่เป็น "66 29 50 26 80" (66 คือ รหัสประเทศส่วนเลขหมวดของหมายเลขมือถือจะถูกตัดเลข 0 ออก เช่น 09 จะเหลือแค่ 9 เป็นต้น แล้วจึงนำตัวเลขทั้งหมดมาต่อกันแล้วสลับคู่) เช่นเดียวกันกับ Time Stamp ข้อมูล "40 30 21 21 94 34 82" ซึ่งมีรูปแบบเป็น "YY/MM/DD HH:MM:SS:ss"หมายถึง ข้อความนี้ส่งเมื่อ "04/03/12 12:49:43:28"

การแปลงตัวอักษรชนิด 7 บิต เป็นข้อมูล 8 บิต (Octet) โดยจากตารางที่ 2.2 ในส่วนของ TPDU จะเป็นส่วนที่เราสามารถใส่รหัสของข้อความที่ต้องการส่ง แต่เนื่องจากเราไม่สามารถนำรหัสของตัวอักษรแบบ 7 บิตใส่ไปได้ โดยตรงจำเป็นต้องผ่านการแปลงให้เป็นรหัสข้อมูลแบบ 8 บิตก่อน โดยตัวอย่างต่อไปนี้เป็นการแปลงข้อความ "hellohello" ยาว 10 ตัวอักษรซึ่งแต่ละตัวเป็นอักษรเป็นชนิด 7 บิต ให้เป็นข้อมูล 8 บิต สำหรับใช้ในการส่ง SMS การแปลงเริ่มจากการนำรหัส 7 บิตของตัวอักษรตัวแรก (h) มาเติมข้างหน้าด้วย 1 บิต ท้ายสุดของรหัส 7 บิต ของอักษรตัวที่ 2 (e) จะได้ผลลัพธ์ 8 บิต (1 byte) เป็น "E8" ขั้นตอนต่อมาให้เอา 6 บิตที่เหลือของอักษรตัวที่ 2 มาเติมข้างหน้าด้วย 2 บิตท้ายของรหัส 7 บิต ของอักษรตัวที่ 3 (l) จะได้ผลลัพธ์ 8 บิต เป็น "32" และทำเช่นนี้เรื่อยไปโดยจำนวนบิตที่นำมากระทำจะเพิ่มขึ้นเป็น 3 บิต 4 บิต จนกระทั่งถึง 7 บิต แล้วเริ่มกระบวนการใหม่จนกระทั่งหมดชุดตัวอักษร หลังจากการแปลงข้อความ "hellohello" จะได้ข้อมูลเป็นเลขฐาน 16 จำนวน 9 ไบต์ เป็น E8 32 9B FD 46 97 D9 EC 37 โดยมีวิธีการแปลงแสดงดังตารางที่ 2.5 โดยที่ตัวอักษรชนิด 7 บิต ถูกกำหนดโดยคู่มือ GSM 03.38 ดังตารางที่ 2.6

ตารางที่ 2.5 แสดงวิธีการแปลงตัวอักษรชนิด 7 บิต เป็นข้อมูล 8 บิตข้อความ "hellohello"

h	e	l	l	o	h	e	l	l	o
104	101	108	108	111	104	101	108	108	111
1101000	1100101	1101100	1101100	1101111	1101000	1100101	1101100	1101100	1101111
1101000	1100101	1101100	1101100	1101111	1101000	1100101	1101100	1101100	1101111
11101000	00110010	10011011	11111101	01000110	10010111	11011001	11101100	110111	
E8	32	9B	FD	46	97	D9	EC	37	

ตารางที่ 2.6 The GSM 03.38 Default Character Set

Dec		0	16	32	48	64	80	96	112
	Hex	0	10	20	30	40	50	60	70
0	0	@	Δ	SP	0	¡	P	¿	p
1	1	£	_	!	1	A	Q	a	q
2	2	$	Φ	"	2	B	R	b	r
3	3	¥	Γ	#	3	C	S	c	s
4	4	è	Λ	¤	4	D	T	d	t
5	5	é	Ω	%	5	E	U	e	u
6	6	ù	Π	&	6	F	V	f	v
7	7	ì	Ψ	'	7	G	W	g	w
8	8	ò	Σ	(	8	H	X	h	x
9	9	Ç	Θ	)	9	I	Y	i	y
10	A	LF	Ξ	*	:	J	Z	j	z
11	B	Ø	<ESC>	+	;	K	Ä	k	ä
12	C	ø	Æ	,	<	L	Ö	l	ö
13	D	CR	æ	-	=	M	Ñ	m	ñ
14	E	Å		.	>	N	Ü	n	ü
15	F	å	É	/	?	O	§	o	à

2.1.2 คำสั่ง AT Command กับมือถือ

การสื่อสารกับอุปกรณ์สื่อสารต่างๆ เช่น โมเด็มหรืออุปกรณ์ DTE (Data Terminal Equipment) นั้นสามารถใช้ชุดคำสั่งที่เป็นมาตรฐานที่เรียกว่า AT Command ในการติดต่อเพื่อโต้ตอบ ตั้งค่า หรือสั่งอุปกรณ์เหล่านั้น ให้ทำงานตามที่ต้องการ โดยชุดคำสั่งพื้นฐานจะถูกกำหนดไว้ใน Hayes AT Command ซึ่งบริษัท Hayes เป็นผู้คิดค้นชุดคำสั่งนี้ เพื่อใช้กับโมเด็มของตนและต่อมาได้กลายเป็นมาตรฐานสำหรับ

ผู้ผลิตโมเด็มรายอื่นๆ โดยอาจจะมีชุดคำสั่งขยาย (Extended AT Command) เพื่อใช้เป็นการเฉพาะสำหรับผู้ผลิตรายนั้นๆ

การติดต่อกับมือถือก็เช่นกันเราสามารถใช้ชุดคำสั่งที่กำหนดไว้ใน GSM AT Command ซึ่งมีคำสั่งเพิ่มเติมที่เหมาะสมสำหรับการใช้งานและควบคุมมือถือและเนื่องจากมีรายระเอียดที่ค่อนข้างมาก จึงจะพูดถึงเฉพาะคำสั่งที่จำเป็นสำหรับโครงงานนี้เท่านั้น การเชื่อมต่อคอมพิวเตอร์กับมือถือนั้น จะทำผ่านสาย Data Link ซึ่งเป็นการเชื่อมต่อแบบอนุกรมโดยใช้โปรแกรมเทอร์มินอลต่างๆ เช่น Hyper Terminal ของ Windows ส่วนความเร็วในการสื่อสารมักจะใช้ 19200 bps

คำสั่ง AT-COMMAND สำหรับ SMS จาก GSM 07.05

AT+CNMI	เป็นคำสั่งเลือกสัญญาณข้อความ SMS ใหม่
AT+CSCB	เป็นคำสั่งในการเลือกข้อความ Cell Broadcast
AT+CMGF	เป็นคำสั่งในการเลือกโหมดของ Message ที่จะส่ง
AT+CSCA	เป็นคำสั่งในการดูค่าของ SMS Service Center
AT+CMGL	เป็นคำสั่งเรียกอ่าน Message โดยให้แสดงตามชนิดที่ต้องการเรียกดู
AT+CMGR	เป็นคำสั่งที่ใช้เรียกอ่าน Message ทีละอันเฉพาะอันที่ต้องการอ่าน
AT+CMGS	เป็นคำสั่งที่ใช้ในการส่ง Message ไปยัง Address ที่เลือกไว้
AT+CMSS	เป็นคำสั่งที่ใช้ส่ง Message จาก SIM Card ไปยัง Address ที่เลือกไว้
AT+CMGW	เป็นคำสั่งสำหรับเขียน Message เก็บไว้ใน SIM Card
AT+CMGD	เป็นคำสั่งสำหรับลบ Message ที่เก็บไว้ใน SIM Card
AT+CSMS	เป็นคำสั่งที่ใช้เลือกบริการข้อความ
AT+CPMS	เป็นคำสั่งที่ใช้เลือกหน่วยความจำของ SMS
AT+CMGC	เป็นคำสั่งที่ใช้ส่งคำสั่ง SMS

ตารางที่ 2.7 ลักษณะชุดคำสั่งของ AT+CNMI

คำสั่ง	ค่าตอบสนอง
AT+CNMI=?	+CNMI:(list of supported <mode>s),(list of supported <mt>s),(list of supported <bm>s),(list of supported <ds>s),(list of supported <bfr>s)
AT+CNMI?	+CNMI:<mode>,<mt>,<bm>,<ds>,<bfr>
AT+CNMI=[<mode>][,<mt>][,<bm>][,<ds>][,<bfr>]	OK /ERROR /+CMS ERROR

<mode> 0 ถ้าบัฟเฟอร์เต็มแล้วมีสัญญาณเข้ามาใหม่จะเข้ามาแทนอันที่เก่าที่สุด
1 ไม่รับ Unsolicited result code ใหม่เมื่อ TA-TE link ถูกจองไว้หรือ ไม่เช่นนั้นก็ Forward ไปยัง TE โดยตรง
2 เก็บ Unsolicited result code ในบัฟเฟอร์ของ TA เมื่อ TA-TE link ถูกจองไว้แล้วถูกส่งไปยังTE เมื่อสิ้นสุดการจอง
3 ทำการ Forward ค่า Unsolicited result code ไปที่ TE โดยตรง <mt> กฎสำหรับการเก็บ SMS ที่รับเข้ามาขึ้นอยู่กับวิธีการเข้ารหัสของข้อมูล,การตั้งค่า Memory Format และค่านี้

<bm> กฎสำหรับการเก็บ CBMs ที่รับเข้ามาขึ้นอยู่กับวิธีการเข้ารหัสของข้อมูล. การเลือกรูปแบบของ CBM และค่านี้

<ds> 0 ไม่มี SMS-STATUS-REPORT ส่งไปยัง TE1 SMS-STATUS-REPORT ส่งไปยัง TE โดยใช้ Unsolicited resultcode +CDS:<length><CR><LF><PDU> (PDU Mode Enable)02 ถ้า SMS-STATUS-REPORT ส่งไปใน ME/TA สัญญาณของ location ของหน่วยความจำถูกส่งไป TE โดยใช้ Unsolicited result code

<bfr> 1 TAบัฟเฟอร์ของ Unsolicited result code จะถูกจำกัดความในคำสั่งนี้

ตารางที่ 2.8 ลักษณะชุดคำสั่งของ AT+CSCB

คำสั่ง	ค่าตอบสนอง
AT+CSCB=?	+CSCB:(list of supported <mode>s)
AT+CSCB?	+CSCB:<mode>,<mids>,<dcss>
AT+CSCB=[<mode>[,<mids>[,<dcss>]]]	OK/ERROR

<mode> 0 รับข้อความ
1 ไม่รับข้อความ

<mids> ค่า CBM message IDs: รูปแบบสตริง

<dcss> ค่า CBM data coding schemes : รูปแบบสตริง

ตารางที่ 2.9 ลักษณะชุดคำสั่งของ AT+CMGF

คำสั่ง	ค่าตอบสนอง
AT+CMGF=?	+CMGF: (list of supported <mode>s)
AT+CMGF?	+CMGF:<mode>
AT+CMGF=[<mode>]	OK/ERROR

<mode> 0 เป็น PDU-Mode

1 เป็น Text-Mode

ตารางที่ 2.10 ลักษณะชุดคำสั่งของ AT+CSCA

คำสั่ง	ค่าตอบสนอง
AT+CSCA=?	OK
AT+CSCA?	+CSCA:<sca>,<tosca>
AT+CSCA=<sca>[,<tosca>]	OK/ERROR

<sca> Service-center address in string format

<tosca> Service-center address format

ตารางที่ 2.11 ลักษณะชุดคำสั่งของ AT+CMGL

คำสั่ง	ค่าตอบสนอง
AT+CMGL=?	+CMGL: (list of supported <stat>s)
AT+CMGL=[<stat>]	If PDU mode (+CMGF=0) +CMGL:<index>,<stat>,[<alpha>],<length><CR> <LF><pdu><CR><LF>

<stat> ตัวบอกสถานะของข้อความที่อยู่ใน SIM Card

0 Messages ที่ได้รับมาแล้วยังไม่ได้อ่าน

1 Messages ที่ได้รับมาแล้วอ่านแล้ว

2 Messages ที่เก็บไว้สำหรับส่งแต่ยังไม่ได้ส่ง

3 Messages ที่ส่งไปแล้ว

4 Messages ทุกชนิด

<index> ตัวบอกตำแหน่งที่เราต้องการเลือกว่าเป็นข้อความที่เท่าไรใน SIM Card

<length> ความยาวของ TPDU โดยจะนับแบบ octets

<pdu> ข้อความที่เป็นส่วนของ SMSC รวมกับ TPDU

ตารางที่ 2.12 ลักษณะชุดคำสั่งของ AT+CMGR

คำสั่ง	ค่าตอบสนอง
AT+CMGR=?	OK
AT+CMGR=<index>	If PDU mode (+CMGF=0) +CMGR:<stat>,[<alpha>],<length><CR><LF><pdu>

<stat> ตัวบอกสถานะของข้อความที่อยู่ใน SIM Card

0 Messages ที่ได้รับมาแล้วยังไม่ได้อ่าน

1 Messages ที่ได้รับมาแล้วอ่านแล้ว

2 Messages ที่เก็บไว้สำหรับส่งแต่ยังไม่ได้ส่ง

3 Messages ที่ส่งไปแล้ว

4 Messages ทุกชนิด

<index> ตัวบอกตำแหน่งที่เราต้องการเลือกว่าเป็นข้อความที่เท่าไรใน SIM Card

<length> ความยาวของ TPDU โดยจะนับแบบ octets

<pdu> ข้อความที่เป็นส่วนของ SMSC รวมกับ TPDU

ตารางที่ 2.13 ลักษณะชุดคำสั่งของ AT+CMGS

คำสั่ง	ค่าตอบสนอง
AT+CMGS=?	OK
If PDU mode (+CMGF=0) AT+CMGS=<length><CR>PDU is given <ctrl-Z/ESC>	If sending is successful: +CMGS:<mr> If sending is not successful: +CMS ERROR:<err>

<length> ความยาวของ TPDU โดยจะนับแบบ octets

<pdu> ข้อความที่เป็นส่วนของ SMSC รวมกับ TPDU

<mr> จำนวนครั้งที่เราส่ง SMS หรือตัวอ้างอิงข้อความ

ตารางที่ 2.14 ลักษณะชุดคำสั่งของ AT+CMSS

คำสั่ง	ค่าตอบสนอง
AT+CMSS=?	OK
AT+CMSS=<index>[,<da>[,<toda>]]	If sending is successful: +CMSS:<mr> If sending is not successful: +CMS ERROR:<err>

<index> ตัวบอกตำแหน่งที่เราต้องการเลือกว่าเป็นข้อความที่เท่าไรใน SIM Card

<da> เป็นหมายเลขโทรศัพท์ที่เราต้องการส่ง SMS โดยจะอยู่ในรูป "หมายเลข" ซึ่งอยู่ในรหัส ASCII

<toda> เป็นหมายเลขโทรศัพท์ที่เราต้องการส่ง SMS โดยจะอยู่ในรูป "+รหัสประเทศตามด้วยหมายเลข" ซึ่งอยู่ในรูปตัวเลข

<mr> จำนวนครั้งที่เราส่ง SMS หรือตัวอ้างอิงข้อความ

ตารางที่ 2.15 ลักษณะชุดคำสั่งของ AT+CMGW

คำสั่ง	ค่าตอบสนอง
AT+CMGW=?	OK
If PDU mode (+CMGF=0) AT+CMGW=<length>[,<stat>]<CR>PDU is given <ctrl-Z/ESC>	- +CMGW:<index> +CMS ERROR:<err>

<stat> ตัวบอกสถานะของข้อความที่อยู่ใน SIM Card

0 Messages ที่ได้รับมาแล้วยังไม่ได้อ่าน

1 Messages ที่ได้รับมาแล้วอ่านแล้ว

2 Messages ที่เก็บไว้สำหรับส่งแต่ยังไม่ได้ส่ง

3 Messages ที่ส่งไปแล้ว

4 Messages ทุกชนิด

<index> ตัวบอกตำแหน่งที่เราต้องการเลือกว่าเป็นข้อความที่เท่าไรใน SIM Card

<length> ความยาวของ TPDU โดยจะนับแบบ octets

<pdu> ข้อความที่เป็นส่วนของ SMSC รวมกับ TPDU

ตารางที่ 2.16 ลักษณะชุดคำสั่งของ AT+CMGD

คำสั่ง	ค่าตอบสนอง
AT+CMGD=?	OK
AT+CMGD=<index>	OK/+ERROR/+CMS ERROR

<index> ตัวบอกตำแหน่งที่เราต้องการเลือกว่าเป็นข้อความที่เท่าไรใน SIM Card

ตารางที่ 2.17 ลักษณะชุดคำสั่งของ AT+CSMS

คำสั่ง	ค่าตอบสนอง
AT+CSMS=?	+CSMS: (list of supported <service>s)
AT+CSMS?	+CSMS:<service>,<mt>,<mo>,<bm>,
AT+CSMS=[<service>]	+CSMS:<mt>,<mo>,<bm> OK/ERROR/+CMS ERROR

<service> 0 GSM 3.40 และ 3.41

<mt> 1 รองรับรูปแบบ Mobile Terminate Massage

0 ไม่รองรับรูปแบบ Mobile Terminate Massage

<mo> 1 รองรับรูปแบบ Mobile Originatee Massage

0 ไม่รองรับรูปแบบ Mobile Originatee Massage

<bm> 1 รองรับรูปแบบ Broadcast Type Massage

0 ไม่รองรับรูปแบบ Broadcast Type Massage

ตารางที่ 2.18 ลักษณะชุดคำสั่งของ AT+CPMS

คำสั่ง	ค่าตอบสนอง
AT+CPMS=?	+CPMS: (list of supported <mem1>s),(list of supported <mem2>s),(list of supported <mem3>s)
AT+CPMS?	+CPMS:<mem1>,<use1>,<total1>,<mem2>,<use2>,<total2>,<mem3>,<use3>,<total3>,
AT+CPMS=<mem1>[,<mem2>,[<mem3>]]	+CPMS:<use1>,<total1>,<use2>,<total2>,<use3>,<total3> OK/ERROR/+CMS ERROR

<mem1> ส่วนความจำสำหรับอ่านและลบข้อความ

<mem2> ส่วนความจำสำหรับเขียนและส่งข้อความ

<mem3> ส่วนความจำสำหรับข้อความที่รับมาเก็บไว้

<memx> ส่วนความจำสำหรับอ่านและลบข้อความ

<usex> จำนวนข้อความที่เก็บอยู่ใน <memx>

<totalx> จำนวนข้อความทั้งหมดที่สามารถเก็บได้ใน <memx>

ตารางที่ 2.19 ลักษณะชุดคำสั่งของ AT+CMGC

คำสั่ง	ค่าตอบสนอง
AT+CMGC=?	OK
If PDU mode (+CMGF=0) AT+CMGC=<length><CR>PDU is given <ctrl-Z/ESC>	If sending is successful: +CMGC:<mr> If sending is not successful: +CMS ERROR:<err>

<length> ความยาวของ TPDU โดยจะนับแบบ octets

<pdu> ข้อความที่เป็นส่วนของ SMSC รวมกับ TPDU

<mr> จำนวนครั้งที่เราส่ง SMS หรือตัวอ้างอิงข้อความ

2.1.3 Delphi 7

Delphi 7คือซอฟต์แวร์ที่เรานำมาใช้ในการเขียนโปรแกรมเพื่อสร้างแอพพลิเคชัน หรือซอฟต์แวร์อีกที โดยมันจะประกอบไปด้วยเครื่องมือชนิดต่างๆที่ใช้ให้การเขียนโปรแกรมทำได้อย่างสะดวก

Delphi 7 จัดเป็นเครื่องมือเขียนโปรแกรมชนิด Visual Programming เช่นเดียวกับ Visual Basic หรือ Visual C++ โดยมีข้อดีคือ สามารถเขียนโปรแกรมได้ง่าย และให้ผลงานออกมาอย่างรวดเร็ว ซึ่งจะแตกต่างจากเครื่องมือเขียนโปรแกรมรุ่นเดิมๆ เช่น Turbo Pascal หรือ Borland C ที่มีความยุ่งยากในการใช้งานและการเรียนรู้ในการเขียนโปรแกรม ดังนั้นจึงจัดให้ Delphi 7 เป็นซอฟต์แวร์ประเภท RAD หรือ Rapid Application Development ซึ่งแปลว่าสามารถสร้างแอพพลิเคชันได้อย่างรวดเร็ว

Delphi 7 นั้นผ่านการพัฒนามาเกือบ 10 ปี ตั้งแต่เวอร์ชัน 1.0 ที่ทำงานบน Windows 3.1X โดยมีจุดเด่นมากๆ ตั้งแต่สมัยนั้นคือ โปรแกรมที่ได้จากการเขียนโปรแกรมมีขนาดเล็ก ทำงานได้รวดเร็ว ซึ่งมักจะถูกนำไปเปรียบเทียบกับ Visual Basic 3.0 ในสมัยนั้น อีกประการหนึ่ง Delphi ใช้ภาษาออบเจ็กต์ปาสคาล จึงเคยถูกเปรียบว่าเป็น Visual Pascal มาแล้ว

เวอร์ชันปัจจุบันของ Delphi นั้นได้รับการพัฒนาให้สามารถสร้างแอพพลิเคชันที่ทำงานบน Windows ได้ดีเหมือนเดิม โดยมีการปรับปรุงให้สามารถพัฒนาแอพพลิเคชันตามแนวความคิดของ .NET ซึ่งจะช่วยให้สามารถเขียนโปรแกรมครั้งเดียว แล้วนำไปใช้งานบนอุปกรณ์ต่างๆไม่ว่าจะเป็น PDA,โทรศัพท์มือถือและบนเว็บได้

2.1.3.1 ความสามารถของ Delphi 7

Delphi 7 นั้นมีความสามารถมากมาย ได้รับการต้อนรับเป็นอย่างดีจากนักพัฒนาแอพพลิเคชันทั่วโลก รวมทั้งเมืองไทยด้วย ซึ่งจะเห็นได้จากการนำ Delphi ไปประกอบการเรียนการสอน การฝึกอบรม ตลอดจนการนำไปสร้างเป็นซอฟต์แวร์เชิงพาณิชย์จำนวนมาก

- สร้างแอพพลิเคชันสำหรับ Windows

Delphi ได้ชื่อว่าเป็นเครื่องมือสำหรับพัฒนาแอพพลิเคชันที่ทำงานบน Windows ที่ใช้งานกันแพร่หลายมาก สามารถสร้างงานได้อย่างรวดเร็ว นิยมนำไปสร้างแอพพลิเคชันทั้งเพื่อการศึกษา และแอพพลิเคชันที่ใช้ในระบบงานจริงๆ ในโลกธุรกิจ

Delphi สามารถสร้างแอพพลิเคชันบน Windows ได้หลายเวอร์ชัน ซึ่งแอพพลิเคชันที่สร้างจาก Delphi ได้รับการยกย่องเป็นอย่างมาก ในด้านขนาดโปรแกรมที่เล็กกะทัดรัด และทำงานได้อย่างรวดเร็ว เมื่อเทียบกับแอพพลิเคชันที่สร้างจากเครื่องมือตัวอื่นๆ ทำให้ Delphi แต่ละเวอร์ชันได้รับรางวัลเกียรติยศจากสถาบันต่างๆทั่วโลกมาโดยตลอด

- สร้างระบบงานด้านฐานข้อมูล

ถ้าจะถามว่า Delphi ถูกนำไปสร้างผลงานอะไรมากที่สุดสำหรับเมืองไทยแล้ว หนีไม่พ้นเรื่องของการพัฒนาระบบงานด้านฐานข้อมูล ซึ่งถูกนำไปใช้งานอย่างแพร่หลายตั้งแต่องค์กรขนาดเล็กไปจนถึงองค์กรระดับประเทศ โดย Delphi มีจุดเด่นในการสร้างแอพพลิเคชันที่ติดต่อกับฐานข้อมูลได้หลายรูปแบบ มีวิธีการและเครื่องมือในการสร้างระบบงาน ระบบบริหารงานบุคคล ระบบคลังสินค้า หรือแม้แต่ระบบจองตั๋วในโรงภาพยนตร์ชั้นนำ

แอพพลิเคชันที่เกี่ยวกับฐานข้อมูลที่สร้างจาก Delphi สามารถนำไปใช้งานกับระบบฐานข้อมูลชั้นนำแทบทุกชนิดทั่วโลก นับตั้งแต่ระบบฐานข้อมูลส่วนบุคคลทั้ง Access, Paradox, Foxpro ไปจนถึงระบบฐานข้อมูลขนาดใหญ่ทั้ง Oracle, Sybase, SQL Server รวมถึงระบบไฟล์ชนิดต่างๆได้ด้วย

สร้างแอพพลิเคชันรองรับ .NET Web Service

แนวโน้มของการพัฒนาแอพพลิเคชันในยุคหน้าที่เราต้องเตรียมตัวให้พร้อมก็คือ การพัฒนาแอพพลิเคชันด้วยเทคโนโลยี .NET ซึ่งจะช่วยให้แอพพลิเคชันชนิดต่างๆ ไม่ว่าจะทำงานบนพีซี โน้ตบุ๊ค PDA หรือแม้แต่โทรศัพท์มือถือ สามารถเชื่อมโยงข้อมูล และคุยกันได้

ใน Delphi 7 นั้นรองรับการพัฒนาแอพพลิเคชันที่เรียกว่า .NET Web Service แล้ว โดยสามารถใช้เครื่องมือชนิดต่างๆพัฒนาแอพพลิเคชันตามแนวความคิดของ .NET เป็นอย่างดี

- ขั้นตอนการเขียนโปรแกรมเพื่อสร้างแอพพลิเคชันกับ Delphi7

ขั้นตอนในการสร้างแอพพลิเคชันด้วย Delphi 7 มีขั้นตอนดังต่อไปนี้

ขั้นที่ 1 : ออกแบบโปรแกรม

นับเป็นขั้นตอนแรกของการเขียนโปรแกรมที่ควรต้องทำ โดยเราจะต้องออกแบบเสียก่อนว่าจะให้โปรแกรมมีหน้าตาอย่างไร มีขั้นตอนการทำงานเป็นอย่างไร ซึ่งบ่อยครั้งที่ผู้เริ่มต้นเขียนโปรแกรมหลายรายละเลย ซึ่งจะส่งผลเสียในอนาคตคือ ถ้าออกแบบโปรแกรมไม่ดี (ขาดการทำงานที่ดี , ขาดความ

ยืดหยุ่นในการปรับแต่ง,ขาดความสวยงามน่าใช้งาน) ก็จะทำให้เกิดความยุ่งยากในการปรับปรุงโปรแกรมนี้ในอนาคต

ในการพัฒนาระบบงานขนาดใหญ่นั้น งานในการออกแบบโปรแกรมกับการเขียนโปรแกรมจะแยกจากกัน อาจจะใช้ทีมงานคนละทีมงานกันทำให้เราต้องมีเครื่องมือที่จะใช้สื่อสาร และบอกรายละเอียดของการออกแบบให้กับผู้เขียนโปรแกรมได้ทราบ ซึ่งเราจะใช้สิ่งที่เรียกว่า โฟลว์ชาร์ต (Flow Chart) เป็นเครื่องมือในการสื่อสารนั้น

โฟลว์ชาร์ตนั้นจะบอกถึงขั้นตอนการทำงาน การตัดสินใจ และข้อมูลที่จะใช้ให้เราทราบอย่างครบถ้วนเพราะฉะนั้นนอกจากจะใช้สื่อสารระหว่างกัน ยังใช้เป็นเครื่องมือสำคัญในการควบคุมการทำงานของผู้เขียนโปรแกรมได้ด้วย

ขั้นที่ 2 : วางคอมโพเนนต์ต่างๆลงใน Form Designer

เราจะนำแบบที่คิดไว้จากขั้นที่ 1 มาสร้างเป็นแอพพลิเคชันไว้ใช้งาน

ขั้นที่ 3 : กำหนดค่าพร็อพเพอร์ตี้ให้กับคอมโพเนนต์ต่างๆ

ในขั้นตอนนี้เราจะกำหนดค่าพร็อพเพอร์ตี้ หรือคุณสมบัติให้กับคอมโพเนนต์ต่างๆที่วางอยู่ใน Form Designer

ขั้นที่ 4 : เขียนคำสั่งกำกับการทำงานให้คอมโพเนนต์

หลังจากกำหนดค่าพร็อพเพอร์ตี้ ซึ่งก็เป็นการกำหนดหน้าตา ลักษณะการทำงานต่างๆแล้ว ต่อไปเราจะต้องเขียนคำสั่งเพื่อกำกับการทำงานของคอมโพเนนต์โดยเขียนคำสั่งผ่าน Code Editor

ขั้นที่ 5 : คอมไพล์โปรแกรมที่เขียนขึ้น

เมื่อเขียนคำสั่งเพื่อดักการทำงานในรูปแบบต่างๆจนครบแล้ว ต่อไปเราก็จะคอมไพล์โปรแกรมที่ได้เขียนขึ้น ซึ่งการคอมไพล์ (Compile) ก็คือการแปรจากคำสั่งที่เราเขียนทั้งภาษาไทย ภาษาอังกฤษที่เราอ่านรู้เรื่องนำมาแปลเป็นภาษาที่คอมพิวเตอร์เข้าใจ และทำงานตามที่เราตั้งใจไว้ได้

ในการคอมไพล์โปรแกรมที่เขียนขึ้นโดย Delphi 7 นั้นให้เราคลิกเมนู Run > Run หรือง่ายกว่านั้นให้กดปุ่ม <F9> ทันที

Delphi 7 จะทำการคอมไพล์คำสั่งที่ได้เขียนขึ้นทั้งหมด แล้วเปลี่ยนสภาวะ (Mode) การทำงานจากสภาวะการออกแบบเขียนโปรแกรม (Design Mode) ไปสู่สภาวะการรันโปรแกรม (Run Mode)

ขั้นที่ 6 : ทดสอบการทำงานของแอพพลิเคชัน

เมื่อคอมไพล์ผ่านแล้ว เราจะทดสอบการทำงานว่าสิ่งที่เราสร้างนั้นทำงานได้ถูกต้องหรือไม่ เพราะบางครั้งเราอาจจะเขียนคำสั่งได้ถูกต้องตามหลักภาษาโปรแกรมทุกอย่าง แต่อาจจะทำงานผิดพลาดก็ได้ จึงต้องตรวจสอบการทำงานให้แน่ใจเสียก่อน

ขั้นที่ 7 : บันทึกผลการทำงาน

เมื่อทดสอบจนพอใจแล้ว เราก็สามารถบันทึกสิ่งที่ได้สร้างขึ้นทั้งหมดเอาไว้ ซึ่งจะสามารถเรียกขึ้นมาแก้ไขปรับปรุงภายหลังได้ โดยเราจะบันทึกไว้ 2 ส่วนคือ ไฟล์ยูนิต (.pas) และโปรเจ็กต์ (.dpr)

2.1.3.2 ภาษาออบเจ็กต์ปาสคาล

ภาษาออบเจ็กต์ปาสคาลในการเขียนโปรแกรมกับ Delphi 7 ซึ่งเราจะเก็บซอร์สโค้ดที่ได้เขียนขึ้นไว้ในไฟล์ที่เรียกว่า ยูนิต (Unit) ซึ่งจะถูกบันทึกไว้ในไฟล์ .pas และเมื่อไฟล์ยูนิตถูกคอมไพล์ก็จะสร้างไฟล์ที่เป็นผลจากการคอมไพล์มีนามสกุลเป็น .dcu

ปกติเมื่อเริ่มสร้างแอพพลิเคชันใน Delphi 7 (ก็คือการสร้างโปรเจ็กต์ใหม่ขึ้นมา) จะมีการสร้างไฟล์ชื่อ Unit1.pas ขึ้นมาให้เราเขียนโค้ดเข้าไป เมื่อเราตรวจสอบจาก Code Editor จะพบว่า Delphi ได้เขียนโค้ดบางส่วนไว้ให้เราอยู่แล้ว

ไฟล์ยูนิตหนึ่งๆจะใช้กับการเขียนโค้ดสำหรับฟอร์ม 1 ฟอร์ม ดังนั้นถ้าโปรเจ็กต์ของเราประกอบด้วย ฟอร์มมากกว่าหนึ่งฟอร์มสามารถจะมีไฟล์ยูนิตที่มีนามสกุล .pas ได้มากกว่า 1 ไฟล์ได้

เมื่อเรามีการนำคอมโพเนนต์เข้ายังฟอร์มก็จะพบว่า Delphi ได้เขียนโค้ดให้เราในตอนต้นของไฟล์ยูนิตแล้ว และเมื่อเราเขียนโปรแกรมเพื่อเลือกจัดการกับอีเวนต์ที่สนใจ Delphi ก็จะเตรียมโค้ดที่เป็นโครงเอาไว้ให้เราเขียนเพิ่มเติม

2.1.3.3 โครงสร้างของยูนิต

เมื่อกลับมามองไฟล์ Unit1.pas ที่ Delphi สร้างไว้ให้เราสามารถแบ่งโค้ดออกเป็นส่วนต่างๆ ดังนี้

ส่วน interface ส่วนนี้เป็นส่วนที่ใช้ประกาศชนิดข้อมูล,ตัวแปร,ค่าคงที่,ออบเจ็กต์,โพรซีเจอร์ และฟังก์ชัน ซึ่งทุกสิ่งที่บรรจุไว้ในส่วนนี้สามารถเข้าถึง และใช้งานได้จากยูนิตอื่นๆ ต้องไม่ลืมว่าโปรเจ็กต์หนึ่งๆ สามารถมีไฟล์ยูนิตได้หลายตัว ซึ่งพื้นที่ส่วน interface นี้จะเริ่มจากคำว่า interface ไปจนถึง implementation

ส่วน implementation ส่วนนี้จะทำหน้าที่เหมือนกับส่วน Interface ต่างกันตรงที่ขอบเขตการเข้าถึงข้อมูลคือ จะเข้าถึงข้อมูล,ตัวแปร,ค่าคงที่,ออบเจ็กต์,โพรซีเจอร์ และฟังก์ชัน ได้จากสิ่งที่อยู่ขอบเขตเฉพาะภายในยูนิตนี้เท่านั้น ยูนิตอื่นๆไม่มีสิทธิ์เข้าถึง ซึ่งพื้นที่ส่วน implementation ไปจนถึง initialization

ส่วน initialization ส่วนนี้จะใช้เก็บคำสั่งที่ถูกเรียกใช้งานก่อนการทำงานของแอพพลิเคชัน โดยปกติจะทำงานก่อนที่จะมีการสร้างออบเจ็กต์ หรือฟอร์มขึ้นมา ดังนั้นเราจึงมักใช้พื้นที่ส่วนนี้กำหนดค่าให้กับตัวแปรบางตัว ซึ่งพื้นที่ของส่วน initialization นี้จะเริ่มต้นจากคำว่า initialization ไปจนถึง finalization (ถ้าไม่มีคำว่า finalization ก็จะถึงคำว่า end.) สำหรับส่วนนี้จะมีก็ได้ ไม่มีก็ได้แล้วแต่โปรแกรม

ส่วน finalization ส่วนนี้จะทำหน้าที่ตรงกันข้ามกับ initialization ซึ่งพื้นที่ส่วน finalization นี้จะเริ่มตั้งแต่คำว่า finalization ไปจนถึง end.

2.1.3.4 ไฟล์โปรเจ็กต์

ปกติเราจะเขียนโปรแกรมที่ไฟล์ยูนิตเท่านั้น แต่ในการทำงานจริงๆของแอพพลิเคชันนั้นจะต้องเริ่มจากการทำงานของไฟล์โปรเจ็กต์ก่อน ซึ่งรายละเอียดการทำงานของไฟล์โปรเจ็กต์นี้เราไม่ต้องจัดการเอง แต่สามารถดูการทำงานภายในได้โดยเลือกเมนู Project > View Source ซึ่งใน Code Editor จะเพิ่มแท็บที่เป็นรายละเอียดของไฟล์โปรเจ็กต์ดังนี้

สำหรับโค้ดของไฟล์โปรเจ็กต์นั้นประกอบไปด้วยส่วนสำคัญ 3 ส่วนคือ

- Program heading
- Uses clause
- การทำงานของโปรแกรม

สำหรับส่วนที่เป็น program heading นั้นจะบอกถึงชื่อโปรแกรม ในที่นี้ก็คือชื่อที่เราตั้งให้ในตอนบันทึกชื่อไฟล์โปรเจ็กต์นั่นเอง และชื่อนี้จะถูกสร้างเป็นไฟล์ .EXE ด้วย

ส่วน uses clause จะเป็นส่วนที่เราใช้ประกาศยูนิตที่ใช้งาน ถ้ามีมากกว่าหนึ่งยูนิตในนี้ก็จะเก็บรายการของยูนิตทั้งหมดที่ใช้เอาไว้

ส่วนสุดท้ายจะเก็บรายละเอียดของคำสั่งในการทำงานของแอพพลิเคชัน ซึ่งทุกๆแอพพลิเคชันที่เป็นชนิดเดียวกันจะมีส่วนนี้เหมือนกันหมดคือ มีส่วนที่กำหนดค่าเริ่มต้น สร้างฟอร์มขึ้นมา แล้วเริ่มการทำงานให้กับแอพพลิเคชัน (ซึ่งถ้าเป็นแอพพลิเคชันทั่วไป ฟอร์มที่ถูกสร้างขึ้นมาก็จะรอการโต้ตอบจากผู้ใช้งานนั่นเอง)

2.1.3.5 การเขียนโปรแกรมแบบ OOP

การเขียนโปรแกรม OOP เป็นการเลียนแบบแนวคิดของสิ่งที่เราพบเห็นในสิ่งต่างๆที่อยู่รอบตัวทั้งนั้น

- คลาสกับออบเจ็กต์

แนวคิดของ OOP นั้นจะกำหนดให้สิ่งต่างๆเป็นวัตถุ หรือออบเจ็กต์ (Object) ซึ่งออบเจ็กต์แต่ละตัว จะถูกสร้างขึ้นมาจากแม่พิมพ์ที่เรียกว่า คลาส (Class) ดังนั้นออบเจ็กต์ที่สร้างจากคลาสจะถือว่าเป็นคนละตัวกัน แต่มีชนิดเดียวกัน เพราะสร้างจากคลาสเดียวกัน

เมื่อมามองถึงออบเจ็กต์ตัวหนึ่งๆ จะเห็นว่ามันจะต้องมีข้อมูลที่ให้กำหนดลักษณะเฉพาะของออบเจ็กต์ซึ่งเราเรียนกว่า พร็อพเพอร์ตี้ (Property)

เมื่อมีพร็อพเพอร์ตี้แล้วสิ่งหนึ่งที่มักจะขาดไม่ได้ในออบเจ็กต์ก็คือ ความสามารถในการทำงานของออบเจ็กต์ซึ่งเราเรียกว่า เมธอด (Method)

เมื่อเราสร้างออบเจ็กต์หนึ่งๆ ขึ้นมาจากคลาส ในภาษาของการเขียนโปรแกรมจะกล่าวได้ว่าเป็น "ออบเจ็กต์ ก็คือ อินสแตนซ์ (Instance) ของคลาส"

2.1.3.6 การเข้าถึงข้อมูลที่เก็บภายในคลาส

ในคลาสจะประกอบด้วยข้อมูลต่างๆนั่นคือ พร็อพเพอร์ตี้ และเมธอด ซึ่งศัพท์ของนักเขียนโปรแกรม จะเรียกทั้งสองอย่างรวมๆว่า เมมเบอร์ (Member) ของคลาส

จุดดีอย่างหนึ่งของคลาสคือ การที่ไม่อนุญาตให้เข้าถึงข้อมูลภายในของคลาสได้โดยตรง จะต้องกระทำผ่านเมมเบอร์ของคลาสซึ่งก็สอดคล้องกับชีวิต ที่เราใช้งานรถยนต์ได้จากสิ่งที่เตรียมไว้ให้ สำหรับการเข้าถึงข้อมูลภายในคลาส Delphi นั้นแบ่งได้เป็น 3 ระดับ ได้แก่

- Private เป็นระดับการเข้าถึงข้อมูลที่เข้าถึงได้เฉพาะข้อมูลที่อยู่ภายในตัวคลาส หรืออินสแตนซ์ของคลาสเท่านั้น ไม่อนุญาตให้ผู้ใช้ หรือคลาสอื่นเข้าถึงได้

- Public เป็นระดับการเข้าถึงข้อมูลที่เข้าถึงได้จากผู้ใช้คลาสทุกคน หรือคลาสอื่นๆ โดยไม่จำกัด
- Protected เป็นระดับการเข้าถึงข้อมูลเฉพาะคลาสที่ถูกสืบทอดมาจากคลาสนี้เท่านั้น

2.1.3.7 Constructor และ Destructor

นอกเหนือจากเมธอดที่เรากำหนดให้คลาสแล้ว ในการเขียนโปรแกรมแบบ OOP นั้นยังต้องมีเมธอดที่จำเป็นอยู่อีก 2 ตัว ซึ่งจะต้องทำหน้าที่ทุกครั้งเมื่อออบเจ็กต์นั้นเริ่มสร้างขึ้นมา หรือ กำลังจะเลิกใช้งานไป

Constructor เป็นเมธอดที่ทำงานตอนที่เราเริ่มสร้างออบเจ็กต์ขึ้นมาจากคลาส โดยจะทำหน้าที่กำหนดค่าเริ่มต้นที่จำเป็นให้กับพร็อพเพอร์ตี้ต่างๆ จับจองทรัพยากรของระบบที่จำเป็นต่อการทำงานของออบเจ็กต์

ส่วน Destructor เป็นเมธอดที่ทำงานในลักษณะตรงกันข้ามคือทำหน้าที่ ตอนที่ออบเจ็กต์นั้นเลิกใช้งาน โดยจะคืนทรัพยากรให้แก่ระบบปฏิบัติการ

2.2 ไมโครคอนโทรลเลอร์

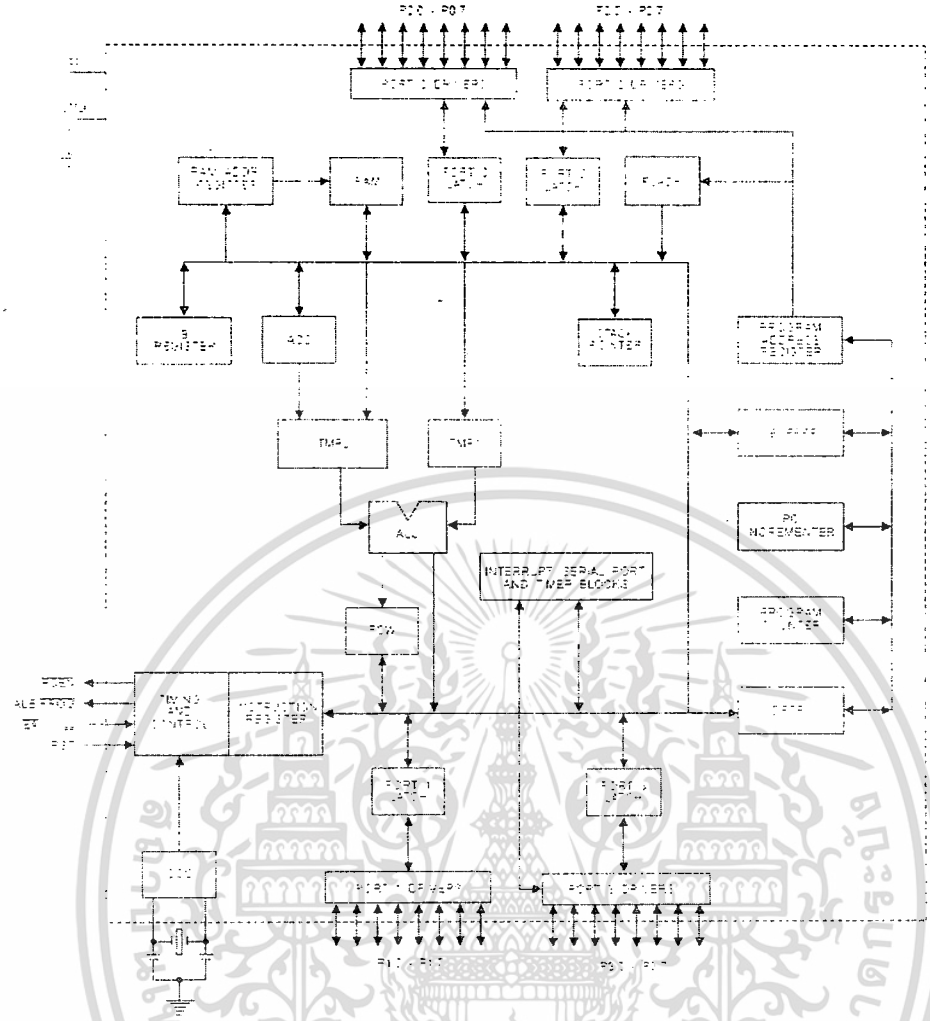
2.2.1 คุณลักษณะพื้นฐานของ MCS-51

คุณสมบัติของไมโครคอนโทรลเลอร์ ตระกูล MCS-51 อนุกรม AT89xxx

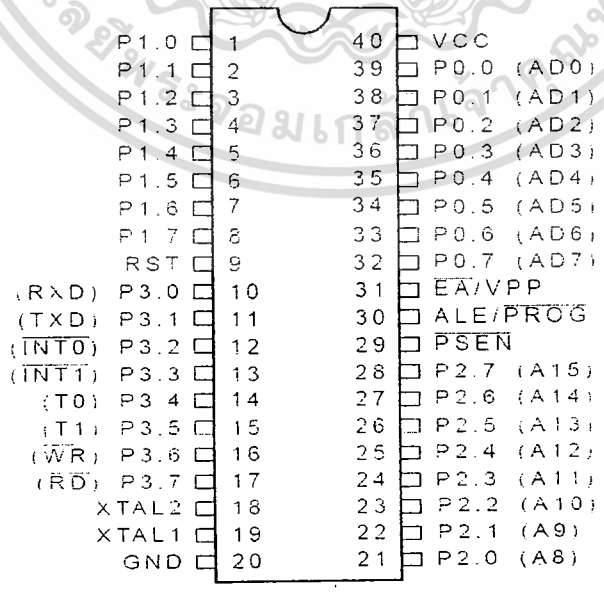
1. เป็นไมโครคอนโทรลเลอร์ที่ใช้ซีพียูขนาด 8 บิต
2. ภายในมีหน่วยความจำโปรแกรมเป็นแบบแฟลช สามารถลบและเขียนใหม่ได้พันครั้ง
3. หน่วยความจำข้อมูลพื้นฐานเป็นหน่วยความจำแบบแรม ในบางเบอร์จะมีหน่วยความจำแบบอีพรอมเพิ่มเติม
4. ขาพอร์ตเป็นแบบสองทิศทาง สามารถใช้งานเป็นได้ทั้งอินพุตและเอาต์พุต
5. มีวงจรสื่อสารอนุกรมแบบฟูลดูเพล็กซ์
6. ไทเมอร์/เคาน์เตอร์ขนาด 16 บิต อย่างน้อย 2 ตัว
7. สามารถรองรับแหล่งกำเนิดอินเตอร์รัพต์ได้ 6 ประเภท
8. สามารถขยายหน่วยความจำภายนอกเพิ่มเติมได้สูงสุด 64 กิโลไบต์
9. มีวงจรกำเนิดสัญญาณนาฬิกาอยู่ภายในชิพ

2.2.2 ลักษณะการจัดขาของ MCS-51

ไมโครคอนโทรลเลอร์ MCS-51 ทุกเบอร์จะมีสถาปัตยกรรมและขาใช้งานพื้นฐานเหมือนกัน ดังแสดงในรูปที่ 2.3 และ 2.4 โดยมีรายละเอียดขั้นต้นดังนี้



รูปที่ 2.3 แสดงรายละเอียดโครงสร้างหลักของไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลช ของ ATMEL



รูปที่ 2.4 การจัดขามาตรฐานของไมโครคอนโทรลเลอร์ MCS-51 ในอนุกรม AT89C5x

ขา Vcc ใช้สำหรับต่อไฟเลี้ยง +5V

ขา GND เป็นขากราวด์ สำหรับต่อกับกราวด์ของระบบ

ขาพอร์ต 0 (P0.0-P0.7) มีขา 8 ขา สามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุต สำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ต 0 ขาใดขาหนึ่งเป็นอินพุต สามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์ต่ำของหน่วยความจำภายนอก (A0-A7) และขาข้อมูล (D0-D7) โดยใช้กระบวนการมัลติเพล็กซ์เข้าช่วย เพื่อสลับการทำงานเป็นได้ทั้งขาติดต่อแอดเดรส และขาข้อมูล

ขาพอร์ต 1 (P1.0-P1.7) มีขา 8 ขา แต่ละขาสามารถกำหนดให้เป็นทั้งอินพุตและเอาต์พุตสำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ตใดเป็นอินพุต สามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย

ขาพอร์ต 2 (P2.0-P2.7) มีขา 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุตสำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ตใดเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์สูงของหน่วยความจำภายนอก (A8-A15)

ขาพอร์ต 3 (P3.0-P3.7) มี 8 ขา แต่ละขาสามารถกำหนดให้เป็นได้ทั้งอินพุตและเอาต์พุตสำหรับใช้งานทั่วไป ถ้าหากต้องการกำหนดให้ขาพอร์ตใดเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ต 3 ยังเป็นขาที่มีหน้าที่การใช้งานพิเศษ ดังมีรายละเอียดขั้นตอนต่อไปนี้

P3.0 ใช้เป็นขาอินพุตสำหรับรับข้อมูลจากการสื่อสารแบบอนุกรม หรือขา RxD

P3.1 ใช้เป็นขาอินพุตสำหรับรับข้อมูลจากการสื่อสารแบบอนุกรม หรือขา TxD

P3.2 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัพต์จากภายนอกช่อง 0 หรือขา $\overline{\text{INT0}}$

P3.3 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัพต์จากภายนอกช่อง 1 หรือขา $\overline{\text{INT1}}$

P3.4 ใช้เป็นขาอินพุตสำหรับรับสัญญาณไทเมอร์จากภายนอกช่อง 0 หรือขา T0

P3.5 ใช้เป็นขาอินพุตสำหรับรับสัญญาณไทเมอร์จากภายนอกช่อง 1 หรือขา T1

P3.6 ใช้เป็นขาสัญญาณ $\overline{\text{WR}}$ ในกรณีที่ใช้เชื่อมต่อกับหน่วยความจำภายนอก

P3.7 ใช้เป็นขาสัญญาณ $\overline{\text{RD}}$ ในกรณีที่ใช้เชื่อมต่อกับหน่วยความจำภายนอก

ขารีเซต (Reset) ใช้ในการรีเซ็ทการทำงานของไมโครคอนโทรลเลอร์ โดยใช้การป้อนสัญญาณเพื่อรีเซ็ทสถานะที่ขานี้ต้องอยู่ในระดับรีเซ็ทอย่างน้อย 2 แมชชีนไซเคิล โดยที่วงจรกำเนิดสัญญาณนาฬิกายังคงทำงานต่อเนื่องไปอย่างเป็นปกติ

ขา $\overline{\text{ALE}}$ /PROG (Address Latch Enable/Program pulse input) เป็นขาที่ใช้ในการควบคุมการแลตช์ของขาพอร์ต 0 เมื่อมีการใช้งานหน่วยความจำภายนอก นอกจากนั้นขานี้ยังใช้เป็นขาสำหรับรับพัลส์

ของการโปรแกรมสำหรับโปรแกรมข้อมูลลงในไมโครคอนโทรลเลอร์ MCS-51 ในรุ่นที่มีหน่วยความจำโปรแกรมเป็นแบบอีพรอม

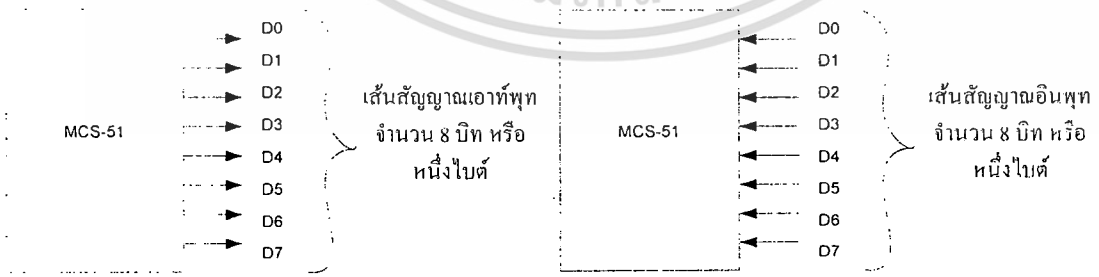
ขา **$\overline{\text{PSEN}}$** (Program Store Enable) ขานี้ใช้ในการส่งสัญญาณเพื่อร้องขอติดต่อกับหน่วยความจำโปรแกรมภายนอก เมื่อไมโครคอนโทรลเลอร์ต้องการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอกตัวไมโครคอนโทรลเลอร์ต้องการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอกตัวไมโครคอนโทรลเลอร์จะส่งสัญญาณออกมาที่ขานี้ 2 ครั้ง ในแต่ละแมชชีนไซเคิล แต่ถ้าหากติดต่อกับหน่วยความจำข้อมูลภายนอกขานี้จะไม่มีการส่งสัญญาณใดๆออกมา

ขา **$\overline{\text{EA}}$** /Vpp (External Access enable/Programming voltage input) ใช้สำหรับเลือกการติดต่อหน่วยความจำโปรแกรมจากภายนอกหรือภายในตัวไมโครคอนโทรลเลอร์ ถ้าหาขานี้เป็น "0" เป็นการเลือกให้ไมโครคอนโทรลเลอร์ติดต่อกับหน่วยความจำโปรแกรมภายนอก แต่ถ้าหากขานี้เป็น "1" เป็นการเลือกให้ไมโครคอนโทรลเลอร์ติดต่อกับหน่วยความจำภายในตัวไมโครคอนโทรลเลอร์ นอกจากนี้ที่ขานี้ยังใช้เป็นขาอินพุตสำหรับรับแรงดันไฟสูงสำหรับการโปรแกรมหน่วยความจำภายในไมโครคอนโทรลเลอร์ สำหรับไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลชต้องการแรงดันสำหรับการโปรแกรม +5V

ขา XTAL1 และ XTAL2 เป็นขาสำหรับต่อคริสตัลเพื่อสร้างสัญญาณนาฬิกาในการกำหนดจังหวะการทำงานของไมโครคอนโทรลเลอร์

2.2.3 พอร์ตอินพุตและพอร์ตเอาต์พุต

พอร์ต คือ แอดเดรสหนึ่งที่ได้รับการกำหนดไว้ เพื่อการโอนย้ายข้อมูลระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอก การกำหนดประเภทของการติดต่อขึ้นอยู่กับทิศทางการไหลของข้อมูล เมื่อพิจารณาจากไมโครคอนโทรลเลอร์เป็นหลัก จากรูปที่ 2.5 (a) เป็นการใช้ไมโครคอนโทรลเลอร์เป็นเอาต์พุตพอร์ต และจากรูป 2.5 (b) เป็นการใช้ไมโครคอนโทรลเลอร์เป็นอินพุตพอร์ต



รูปที่ 2.5 การใช้ไมโครคอนโทรลเลอร์เป็นอินพุตและเอาต์พุตพอร์ต

2.2.3.1 การใช้งานพอร์ตเป็นอินพุต

การใช้งานพอร์ตเป็นการอินพุตข้อมูลจะต้องเริ่มต้นด้วยการส่งข้อมูลที่มีค่าเป็น 1 ออกมาทางบิตของพอร์ตสั้นก่อนเป็นอันดับแรก เพื่อหยุดการทำงานของทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุตของบิตนั้น ทำให้ขาสัญญาณของบิตถูกต่อเข้ากับตัวต้านทานซึ่งทำหน้าที่ Pull-up ภายในซึ่งมีผลทำให้บิตนั้นของพอร์ต 1, 2 และ 3 เป็นสภาวะลอจิกสูง ตัวต้านทานนี้มีค่าประมาณ 50 k ohm ซึ่งเป็นค่าที่สูงมาก และทำให้อุปกรณ์ภายนอกสามารถขับสัญญาณของพอร์ตเหล่านี้เป็นลอจิกต่ำได้ง่าย สำหรับบิตของพอร์ต 0 นั้นแม้ว่าจะมีหลักการทำงานที่คล้ายคลึงกันกับบิตของพอร์ตอื่นๆ แต่เนื่องจากไม่มีตัวต้านทานซึ่งทำหน้าที่ Pull-up ภายในไว้ ทำให้เมื่อทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุตนั้นหยุดการทำงาน ก็จะเป็นผลให้สัญญาณนี้อยู่ในสภาวะอิมพีแดนซ์สูงแทน

2.2.3.2 การใช้งานพอร์ตเป็นเอาต์พุต

เมื่อมีการส่งข้อมูลที่มีค่าเป็น 0 ให้กับแต่ละบิตของพอร์ตทุกพอร์ต ข้อมูลนี้จะถูกส่งให้กับฟลิปฟลอปซึ่งจะค้างค่านี้ไว้ และมีผลทำให้ทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุตนั้นทำงาน ดังนั้นขาสัญญาณก็จะมีสภาวะลอจิกเป็นลอจิกต่ำด้วย

ส่วนการส่งข้อมูลที่มีค่าเป็น 1 ออกมานั้น ในกรณีที่เป็นการทำงานในแต่ละบิตของพอร์ต 1, 2 หรือ 3 จะทำให้ทรานซิสเตอร์ที่ทำหน้าที่ขับสัญญาณเอาต์พุตนั้นหยุดทำงาน มีผลทำให้ขาของสัญญาณเป็นลอจิกสูงด้วยตัวต้านทานที่ Pull-up อยู่ภายในนั้น แต่สำหรับการใช้งานในแต่ละบิตทางพอร์ต 0 นั้นจะมีผลแตกต่างออกไป โดยขาสัญญาณจะมีสภาวะอิมพีแดนซ์สูงแทน เนื่องจากไม่มีตัวต้านทานภายในเชื่อมต่ออยู่นั้นเอง ดังนั้นการใช้งานพอร์ต 0 เป็นการนำข้อมูลออกทางเอาต์พุต จึงจำเป็นต้องใช้ตัวต้านทานภายนอก Pull-up สัญญาณไว้กับลอจิกสูงแทน

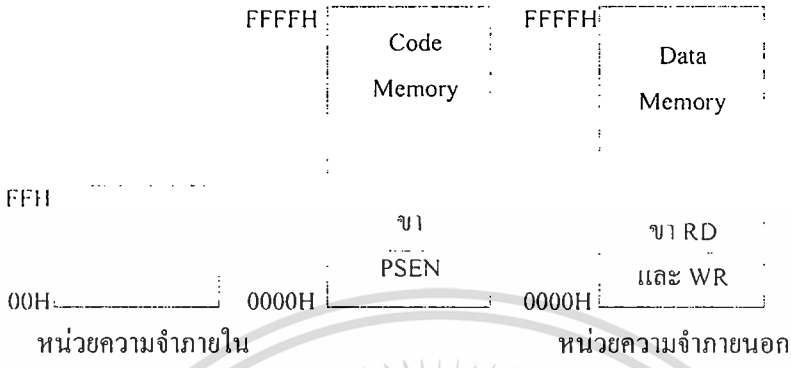
2.2.4 หน่วยความจำโปรแกรมของ MCS-51

หน่วยความจำโปรแกรม (Program Memory) มีไว้เพื่อบรรจุคำสั่งหรือโปรแกรมที่ผู้ใช้พัฒนาขึ้นจัดเก็บไว้ในหน่วยความจำ โดยอาจจะประกอบอยู่ภายในตัวของไอซี 89C51 เอง หรือเป็นไอซีหน่วยความจำ EPROM หรือ ROM แยกออกต่างหากได้ ในกรณีหลังจำเป็นต้องมีการใช้พอร์ตอินพุตเอาต์พุต ทำหน้าที่เป็นบัสแอดเดรส และบัสข้อมูลเพื่อให้สามารถติดต่อกับหน่วยความจำมาตรฐานทั่วไปได้

หน่วยความจำโปรแกรมของ 89C51 เป็นบริเวณหน่วยความจำสำหรับเก็บข้อมูลและคำสั่งใช้งานต่างๆ ซึ่งแม้ว่าจะไม่มีการจ่ายกระแสไฟฟ้าให้ระบบข้อมูลเหล่านี้ก็ยังไม่สูญหาย โครงสร้างของหน่วยความจำโปรแกรมมีลักษณะเช่นเดียวกับหน่วยความจำที่บรรจุอยู่ในไอซีหน่วยความจำประเภทต่างๆเช่น หน่วยความจำแบบ ROM (Read Only Memory) หรือ EPROM (Erasable Programmable Read Only Memory)

การจัดพื้นที่ของหน่วยความจำโปรแกรมของไมโครคอนโทรลเลอร์ 89C51 จะมีการจัดพื้นที่ดังรูปที่ 2.6 โดยที่ไมโครคอนโทรลเลอร์ 89C51 สามารถอ่านข้อมูลหน่วยความจำโปรแกรมสูงสุดได้ไม่เกิน 64 กิโลไบต์ และแยกประเภทของหน่วยความจำโปรแกรมเป็น 2 ลักษณะตามตำแหน่งของ

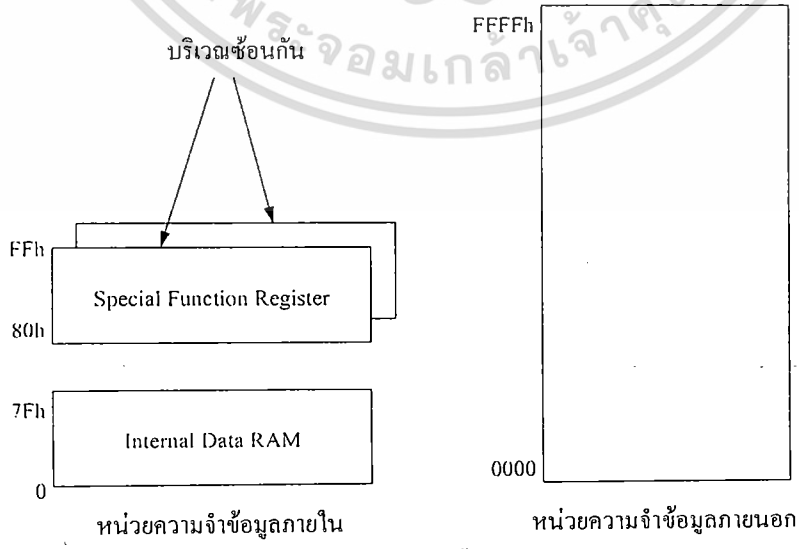
หน่วยความจำนั้น คือ หน่วยความจำโปรแกรมภายใน (Internal Program Memory) ซึ่งเป็นหน่วยความจำ ROM หรือ EPROM ที่อยู่ภายในตัวไอซี ไมโครคอนโทรลเลอร์เองและหน่วยความจำภายนอก (External Program Memory) ซึ่งเป็นการใช้ไอซีหน่วยความจำมาทำหน้าที่เป็นหน่วยความจำโปรแกรมของระบบ



รูปที่ 2.6 แสดงการจัดพื้นที่ของหน่วยความจำโปรแกรมของไมโครคอนโทรลเลอร์ MCS-51

2.2.5 หน่วยความจำข้อมูลของ MCS-51

หน่วยความจำข้อมูลมีหน้าที่สำหรับเก็บข้อมูลหรือตัวแปรที่เกิดขึ้นในขณะที่กำลังประมวลผลโปรแกรมไว้เป็นการชั่วคราว โดยที่หน่วยความจำข้อมูลจะมีลักษณะเป็นหน่วยความจำ RAM แบบสแตติก (Static) ดังนั้นเมื่อไม่มีการจ่ายไฟให้กับระบบ ก็จะมีผลทำให้ข้อมูลที่จัดเก็บไว้ในหน่วยความจำนี้สูญหายไป พื้นที่ของหน่วยความจำข้อมูลของ 89C51 สามารถมีได้ไม่เกิน 64 กิโลไบต์ และแยกประเภทออกเป็นสองลักษณะตำแหน่งที่ตั้งของหน่วยความจำนั้นดังลักษณะในรูปที่ 2.7 คือ หน่วยความจำโปรแกรมภายใน (Internal Date Memory) หรือ RAM ที่อยู่ภายในตัวไมโครคอนโทรลเลอร์เอง และหน่วยความจำข้อมูลภายนอก (External Data Memory) ซึ่งเป็นการใช้ไอซีหน่วยความจำ RAM มาเพิ่มเติมเข้าไปในวงจร ลักษณะเดียวกับการนำไอซี EPROM มาใช้งานเป็นหน่วยความจำโปรแกรมนั่นเอง



รูปที่ 2.7 แสดงลักษณะตำแหน่งที่ตั้งของหน่วยความจำข้อมูล

2.2.5.1 หน่วยความจำข้อมูลภายใน

หน่วยความจำข้อมูลภายใน 89C51 มีจำนวนทั้งหมด 256 ไบต์ โดยจำแนกออกได้เป็นสองลักษณะ คือ พื้นที่เฉพาะสำหรับตัวประมวลผลกลางใช้งานเท่านั้นซึ่งเรียกว่ารีจีสเตอร์ และพื้นที่ใช้งานทั่วไปสำหรับโปรแกรมใช้งานที่ผู้ใช้สร้างขึ้นมา จากรูปที่ 2.8 แสดงถึงการจัดพื้นที่ของหน่วยความจำข้อมูลภายในของ 89C51 ซึ่งแบ่งเป็นสองส่วน คือ หน่วยความจำขนาด 128 ไบต์แรก และหน่วยความจำขนาด 128 ไบต์ถัดไป

2.2.5.2 หน่วยความจำขนาด 128ไบต์แรก

บริเวณนี้จะมีตำแหน่งแอดเดรสอยู่ในช่วง 00H ซึ่งแบ่งได้เป็นอีกสามส่วนดังนี้

- บริเวณแอดเดรส 00H-1FH จำนวน 32 ไบต์ จำแนกออกเป็นกลุ่มหรือแบงก์ข้อมูลจำนวน 8 ไบต์ รวมทั้งหมดสี่กลุ่ม พื้นที่ข้อมูลในแต่ละกลุ่มจะถูกใช้งานในฐานะของรีจีสเตอร์ทั่วไป เรียกว่า รีจีสเตอร์ R0-R7
- บริเวณแอดเดรส 20H-2FH จำนวน 16 ไบต์ จะเป็นพื้นที่ส่วนสำหรับผู้ใช้ที่สามารถอ้างถึงข้อมูลได้ทั้งแบบไบต์และแบบบิต
- บริเวณแอดเดรส 30H-7FH เป็นบริเวณที่สามารถนำไปใช้งานได้อย่างอิสระ โดยสามารถอ้างอิงได้เฉพาะในลักษณะของไบต์ข้อมูลตามปกติเท่านั้น

2.2.5.3 หน่วยความจำขนาด 128 ไบต์ถัดไป

พื้นที่ตั้งแต่แอดเดรส 80H-FFH เป็นหน่วยความจำที่นำมาใช้งานเป็นรีจีสเตอร์หน้าที่พิเศษ แต่ก็ยังมีบริเวณของหน่วยความจำที่อยู่บริเวณนี้ที่ผู้ใช้สามารถเก็บข้อมูลได้ แต่การเรียกใช้งานจะต้องมีการเข้าถึงข้อมูลแบบโดยอ้อมเท่านั้น

2.2.6 รีจีสเตอร์หน้าที่พิเศษ

เป็นรีจีสเตอร์ที่ทำหน้าที่ควบคุมการทำงานของอุปกรณ์หรือพอร์ตของ 89C51 ทั้งหมด โดยมีตำแหน่งอยู่ในบริเวณแอดเดรส 80H-FFH การใช้งานรีจีสเตอร์หน้าที่พิเศษเหล่านี้สามารถทำได้ทั้งการระบุถึงชื่อรีจีสเตอร์หรือตำแหน่งแอดเดรสที่เป็นของรีจีสเตอร์นั้นก็ได้

2.2.6.1 แอกคิวมูเลเตอร์ (Accumulator)

เป็นรีจีสเตอร์ขนาด 8 บิต ทำหน้าที่เก็บข้อมูลที่ส่งให้หน่วยทำงานในซีพียูและเก็บผลลัพธ์ที่ได้จากการทำงานนั้น การใช้งานในโปรแกรมจะเรียกว่า รีจีสเตอร์ A

2.2.6.2 รีจีสเตอร์ B

เป็นรีจีสเตอร์ที่ใช้สำหรับการทำคำสั่งการคูณหารตัวเลข ในกรณีที่ไม่ใช้การคำนวณทางด้านคณิตศาสตร์ ก็สามารถนำไปใช้งานเช่นเดียวกับรีจีสเตอร์ทั่วไปได้

2.2.6.3 โปรแกรมเคาน์เตอร์(Program Counter)

เป็นรีจีสเตอร์ที่ใช้สำหรับการชี้ตำแหน่งแอดเดรสของหน่วยความจำโปรแกรม ซึ่งจะต้องไปทำงานในลำดับถัดไป การใช้งานในโปรแกรมจะเรียกว่า รีจีสเตอร์ PC

2.2.6.4 สแตคพอยน์เตอร์(Stack Pointer)

เป็นรีจีสเตอร์ขนาด 8 บิต ทำหน้าที่เก็บตำแหน่งของตัวชี้หรือพอยน์เตอร์ของบริเวณสแตค สำหรับเก็บข้อมูลแอคคิวมูเลเตอร์ รีจีสเตอร์ต่างๆ รวมทั้งข้อมูลจากโปรแกรม ค่าเริ่มต้นของสแตคจะอยู่ที่ตำแหน่ง 07H การใช้งานในโปรแกรมจะเรียกว่ารีจีสเตอร์ SP

2.2.6.5 ตัวชี้ข้อมูลหรือดาต้าพอยน์เตอร์ (Data Pointer)

เป็นรีจีสเตอร์ขนาด 16 บิต ซึ่งเรียกว่า รีจีสเตอร์ DPTR และสามารถใช้งานแยกออกเป็นรีจีสเตอร์ขนาด 8 บิต สองตัว คือ รีจีสเตอร์ DPH และ DPL เพื่อเก็บค่าแอดเดรสของหน่วยความจำที่จะต้องใช้งานภายในโปรแกรม หรืออาจเป็นแอดเดรสของอุปกรณ์ภายนอก

2.2.6.6 โปรแกรมสเตตัสเวิร์ด (PSW)

รีจีสเตอร์นี้ทำหน้าที่บอกถึงแฟลกซ์สภาวะการทำงานต่างๆ รวมทั้งบิตสำหรับการกำหนดเลือกแบงก์(Bank) ของรีจิสเตอร์ที่ใช้งานด้วย

2.2.6.7 รีจีสเตอร์ที่เกี่ยวข้องกับพอร์ต (Port Register)

รีจีสเตอร์เหล่านี้จะมีความเกี่ยวข้องกับการทำงานของพอร์ตอินพุตเอาต์พุตโดยตรงซึ่งจะเป็นรีจีสเตอร์ขนาด 8 บิต สามารถใช้งานได้ทั้งในลักษณะการอินพุตหรือการเอาต์พุตข้อมูลได้

2.2.6.8 รีจีสเตอร์ SBUF

เป็นบัฟเฟอร์ขนาด 8 บิต สำหรับการสื่อสารข้อมูลแบบอนุกรมทั้งการรับและการส่งข้อมูล

2.2.6.9 รีจีสเตอร์ PCON

เป็นรีจีสเตอร์ควบคุมการทำงานในสามลักษณะ ซึ่งได้แก่ การควบคุมการทำงานของโปรแกรมเซ็นเซอร์ การกำหนดอัตราทวีคูณของอัตราเร็วในการสื่อสารข้อมูลอนุกรมและแฟลกซ์ สภาวะการทำงานทั่วไป

2.2.6.10 รีจีสเตอร์ IP, IE, TMOD, SCON

เป็นกลุ่มรีจีสเตอร์ที่ทำหน้าที่ควบคุมการทำงานของอินเทอร์รัพต์ต่างๆ

2.2.7 การใช้งานพอร์ตสื่อสารอนุกรมแบบ Single Processor

2.2.7.1 พอร์ตสื่อสารอนุกรม

พอร์ตสื่อสารอนุกรมมีโครงสร้างการทำงานในแบบที่เรียกว่า ฟูลดูเพล็กซ์(Full Duplex) สามารถรับและส่งข้อมูลอนุกรมได้ในเวลาเดียวกัน

ทางด้านส่งใช้ขา TxD (พอร์ต 3.1)

ทางด้านรับใช้ขา RxD (พอร์ต 3.0)

Serial Port Buffer (SBUF) ใช้เป็นบัฟเฟอร์สำหรับรับและส่งข้อมูลอนุกรม โดยมีอยู่ 2 ตัว

การส่งข้อมูล ข้อมูลที่จะส่งให้ใส่ใน SBUF โดยใช้คำสั่ง MOV SBUF, A โดยเตรียมข้อมูลที่จะส่งเข้าที่ A ก่อน

การรับข้อมูล ข้อมูลที่รับได้จะอยู่ใน SBUF การถ่ายข้อมูลออกมาใช้คำสั่ง MOV A, SBUF แล้วจึงนำข้อมูลใน A ไปใช้

พอร์ตสื่อสารอนุกรมสามารถโปรแกรมการทำงานได้หลายโหมดด้วยกัน โดยเลือกที่บิต SM1 และ SM0 ซึ่งอยู่ในรีจีสเตอร์ควบคุม SCON การทำงานทั้ง 4 โหมดของพอร์ตสื่อสารอนุกรม มีดังนี้

SM0, Sm1 บิตเลือกโหมดการทำงาน

ตารางที่ 2.20 การเลือกโหมดการทำงานในการรับส่งข้อมูล

SM0	SM1	โหมด	การทำงาน
0	0	0	Shift register ความเร็วในการรับหรือส่งข้อมูลเท่ากับ (1/12) ของ CPU OSC
0	1	1	8 Bit UART ความเร็วในการรับหรือส่งข้อมูลกำหนดได้จาก Time 1,2
1	0	2	9 Bit UART ความเร็วในการรับหรือส่งข้อมูล = (1/32) หรือ (1/64) เท่าของ CPU OSC โดยขึ้นกับบิต SMOD ใน PCON
1	1	3	9 Bit UART ความเร็วในการรับหรือส่งข้อมูลกำหนดที่ Time 1,2

REN (Receive Enable) บิตควบคุมให้รับหรือไม่รับข้อมูล

1 : ให้รับข้อมูลได้

0 : ห้ามรับข้อมูล

หมายเหตุ (การรับข้อมูลสามารถห้ามได้ แต่การส่งข้อมูลห้ามไม่ได้)

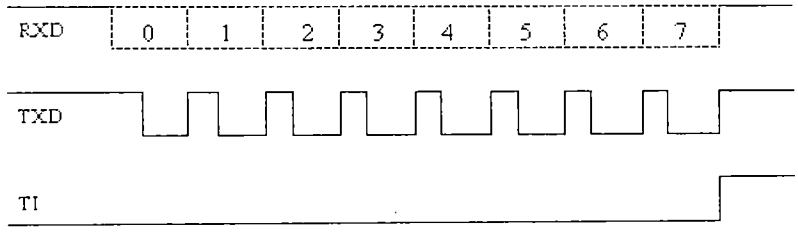
TI แฟลกซ์ TI จะเป็น 1 เมื่อสิ้นสุดการส่งข้อมูล 1 ไบต์

RI แฟลกซ์ RI จะเป็น 1 เมื่อรับข้อมูลเสร็จ 1 ไบต์

การเขียนโปรแกรมควบคุมการรับและส่งข้อมูลทำได้ 2 วิธี

- การตรวจสอบบิต TI หรือ RI โดยใช้คำสั่งตรวจสอบบิต เช่นใช้คำสั่ง WAIT: JNB TI,WAIT
  คำสั่งนี้หมายความว่า ถ้า TI = 0 ให้วนไปยังแอดเดรสชื่อ WAIT
  ถ้า TI = 1 ถือว่าส่งข้อมูลเสร็จแล้วให้ทำคำสั่งถัดไป
- การใช้อินเตอร์รัพต์ควบคุม

โหมด 0 : พอร์ตสื่อสารอนุกรม 8 บิต โดยการส่งข้อมูลจะเลื่อนออกทีละบิตโดยส่งบิต D0 ออกไป ก่อน ทางขา RxD เนื่องจากไม่มีการส่ง Start bit แต่จะส่ง shift clock ทางขา TxD [ความเร็ว (1/12) เท่าของ CPU Clock]

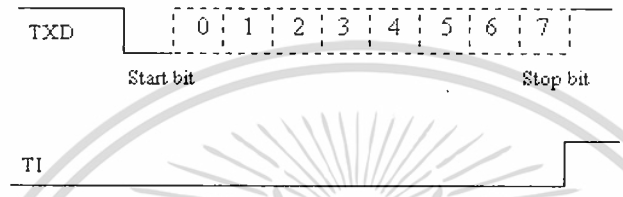


รูปที่ 2.8 แสดง Timing Diagram การส่งข้อมูลโหมด 0

โหมด 1 : พอร์ตสื่อสารอนุกรม 10 บิต ข้อมูล 8 บิต 1 start bit และ 1 stop bit และสามารถเปลี่ยนแปลงความเร็วในการส่งข้อมูลได้ โดยขึ้นกับบิต SMOD ใน PCON และอัตราโอเวอร์โฟลว์ของ Timer 1,2

$$\text{Baud Rate Mode 1,3} = \frac{2^{SMOD} \times \text{CPU OSC}}{32 \times 12 \times [256 - (TH1)]} \quad \text{โดยใช้ Timer 1}$$

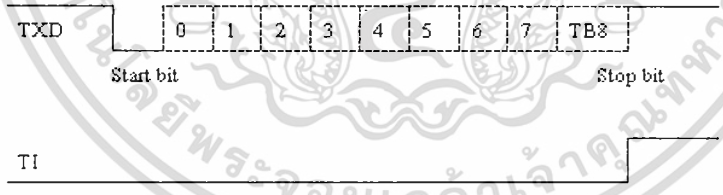
$$\text{Baud Rate Mode 1,3} = \frac{\text{CPU OSC}}{32 \times [65536 - (RCAP2H,RCAP2L)]} \quad \text{โดยใช้ Timer 2}$$



รูปที่ 2.9 แสดง Timing Diagram การส่งข้อมูลโหมด 1

โหมด 2 : พอร์ตสื่อสารอนุกรม 11 บิต ข้อมูล 9 บิต 1 start bit และ 1 stop bit (TB8 นิยมนำมาใช้ส่ง Parity bit) ความเร็วในการรับส่งข้อมูลเท่ากับ (1/32) หรือ (1/64) เท่าของ CPU OSC โดยขึ้นกับบิต SMOD ใน PCON

- Baud Rate (Mode 2) = (1/32) CPU OSC เมื่อ SMOD = 1
- Baud Rate (Mode 2) = (1/64) CPU OSC เมื่อ SMOD = 0

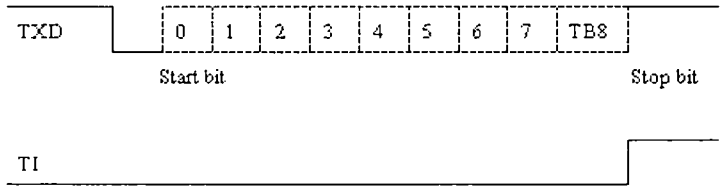


รูปที่ 2.10 แสดง Timing Diagram การส่งข้อมูลโหมด 2

โหมด 3 : พอร์ตสื่อสารอนุกรม 11 bit UART โดย DATA 8 bit, 1 start bit และ 1 stop bit เหมือนโหมด 2 ยกเว้นอัตราความเร็วจะขึ้นกับบิต SMOD ใน PCON และอัตราโอเวอร์โฟลว์ของ Timer 1 สำหรับ 8051 หรือ อัตราโอเวอร์โฟลว์ของ Timer 2 (สำหรับ 80C154D)

$$\text{Baud Rate Mode 3} = \frac{2^{SMOD} \times \text{CPU OSC}}{32 \times 12 \times [256 - (TH1)]} \quad \text{โดยใช้ Timer 1}$$

$$\text{Baud Rate Mode 3} = \frac{\text{CPU OSC}}{32 \times [65536 - (RCAP2H,RCAP2L)]} \quad \text{โดยใช้ Timer 2}$$



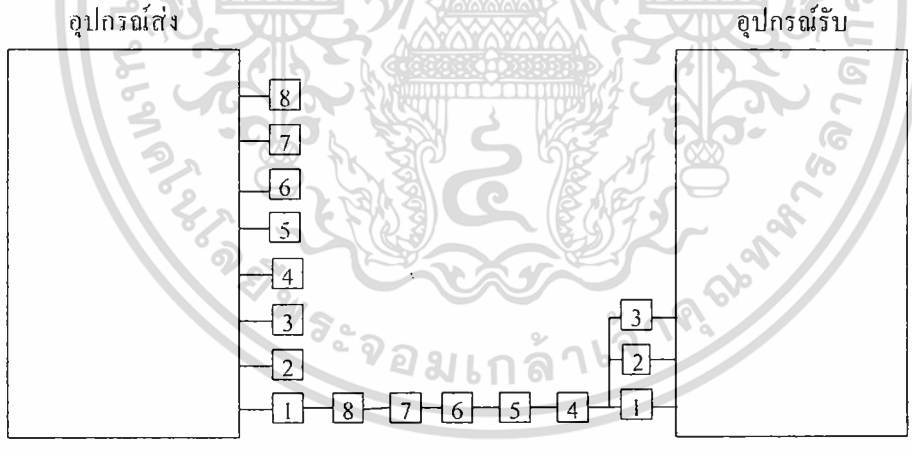
รูปที่ 2.11 แสดง Timing Diagram การส่งข้อมูลโหมด 3

2.2.8 Serial Port

การเชื่อมต่อระบบไมโครคอมพิวเตอร์กับอุปกรณ์ภายนอกส่วนใหญ่จะใช้การเชื่อมต่อแบบขนาน (parallel transmission) กับแบบอนุกรม(serial transmission) สำหรับการเชื่อมต่อแบบอนุกรมนิยมใช้กันมาก เช่น การเคลื่อนย้ายกันระหว่างไมโครคอมพิวเตอร์ หรือ อุปกรณ์เสริมต่างๆ เช่น mouse เป็นต้น

2.2.8.1 วิธีการถ่ายโอนข้อมูล

ในการถ่ายโอนข้อมูลแบบอนุกรมนั้น ข้อมูลจะได้รับการส่งออกมาครั้งละ 1 บิตระหว่างจุดรับและจุดส่ง จะเห็นว่ามีการส่งข้อมูลแบบอนุกรมนี้จะช้ากว่าการส่งข้อมูลแบบขนาน แต่ยังคงใช้อยู่ก็เพราะ ตัวกลางการสื่อสารต้องการช่องเดียวหรือมีสายเพียงคู่เดียวซึ่งจะประหยัดค่าใช้จ่าย ในการใช้ตัวกลางมากกว่าแบบขนานซึ่งถ้าเป็นระยะทางไกลจะดีเพราะเรามีระบบการสื่อสารทางโทรศัพท์อยู่แล้วจึงสามารถนำมาใช้ในการส่งข้อมูลแบบอนุกรมนี้ได้



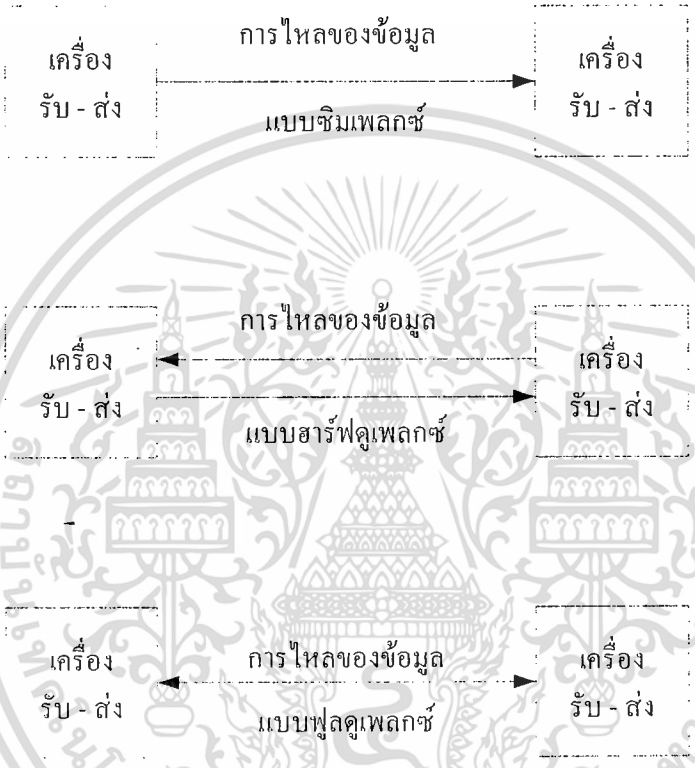
รูปที่ 2.12 แสดงการรับส่งข้อมูล

การส่งข้อมูลแบบอนุกรม ข้อมูลจะถูกเปลี่ยนให้เป็นแบบอนุกรมเสียก่อนแล้วค่อยทยอยส่งครั้งละ 1 บิต ไปยังที่จะรับ ณ จุดรับจะต้องมีกลไกในการเปลี่ยนข้อมูลที่ส่งมาครั้งละบิตให้เป็นสัญญาณแบบขนาน ซึ่งลงตัวพอดี นั่นคือ บิตที่ 1 ลงที่บัสข้อมูลเส้นที่1พอดีการที่จะทำให้การแปลงสัญญาณจากแบบอนุกรม ครั้งละบิตให้ลงพอดีนั้น จำเป็นต้องมีกลไกที่เหมาะสมเพื่อป้องกันการผิดพลาดจากการรับ กลไกที่ว่าแบ่ง ออกเป็น 2 แบบ คือ แบบซิงโครนัส (synchronous) และแบบอะซิงโครนัส (asynchronous)

2.2.8.2 รูปแบบของการติดต่อสื่อสารแบบอนุกรม

การติดต่อแบบอนุกรมอาจจะแบ่งตามรูปลักษณะการส่งข้อมูลได้ 3 แบบคือ

1. แบบซิมเพลกซ์ (simplex) เป็นการส่งข้อมูลได้ทางเดียวเท่านั้น
2. แบบฮาร์ฟดูเพลกซ์ (half duplex) เป็นการส่งข้อมูลได้สองทาง แต่ไม่สามารถส่งในเวลาเดียวกันได้
3. แบบฟูลดูเพลกซ์ (full duplex) ทั้ง 2 สถานีสามารถรับ และส่งได้ในเวลาเดียวกัน



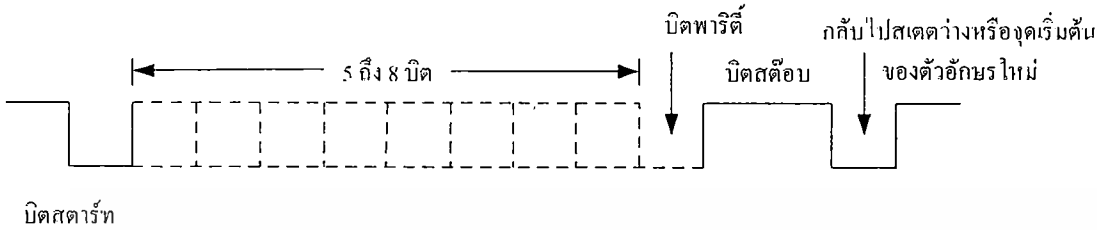
รูปที่ 2.13 แสดงการไหลของข้อมูล

จากรูป 2.13 แสดงให้เห็นว่าการส่งทั้งสามแบบไม่ขึ้นกับจำนวนสายในการติดต่อ เพราะบางครั้งอาจจะใช้วิธีการแยก ความถี่ที่แตกต่างกันระหว่างสัญญาณข้อมูลของฝ่ายส่งกับฝ่ายรับ

ความเร็วของการถ่ายโอนข้อมูลแบบอนุกรมมีหน่วยวัดเป็น บิตต่อวินาที หรือที่เรียกว่า บีพีเอส (bps) แต่เรายังมีหน่วยที่นิยมใช้กันมากคือ บอร์ดเรต หรือ อัตราบอร์ด (baud rate) ซึ่งหมายถึง การเปลี่ยนแปลงของสัญญาณใน 1 วินาที หลายคนยังเข้าใจสับสนระหว่างหน่วยบีพีเอส กับอัตราบอร์ด กล่าวคือ การเปลี่ยนแปลงของสัญญาณ 1 ครั้งอาจจะแสดงถึง การส่งข้อมูลแบบอนุกรมมากกว่า 1 บิต อัตราการส่งข้อมูลเป็นจำนวนบิตจึงเท่ากับ อัตราบอร์ด คูณกับจำนวนบิตใน 1 บอร์ด

2.2.8.3 การสื่อสารแบบอะซิงโครนัส

การสื่อสารแบบนี้ประกอบด้วยบิตเริ่มต้นหรือบิตสตาร์ท (start bit) และบิตสิ้นสุดหรือบิตสต็อป (stop bit)



รูปที่ 2.14 แสดงการสื่อสารแบบอะซิงโครนัส

ขณะที่สถานะของการส่งเป็นแบบว่าง หรือ ไอเดิล (idle) คือยังไม่มีสัญญาณที่ส่งออกมาแต่จะมีสัญญาณ หรือมีแรงดันตลอดเวลาเพื่อความแน่ใจว่าฝ่ายรับยังติดต่ออยู่กับฝ่ายส่งฝ่ายส่งจะเริ่มส่งข้อมูลบอกจุดเริ่มต้น สัญญาณของอะซิงโครนัสจะเป็น "0" หนึ่งช่วงสัญญาณนาฬิกา บิตนี้เรียกว่าบิตสตาร์ท ข้อมูล 1 ตัวอักษรที่ตามหลังบิตสตาร์ตจะมีขนาดตั้งแต่ 5 บิต จนถึง8 บิต โดยอักขระนี้ส่วนมากจะนิยมใช้รหัสแอสกี (ASCII CODE)

แรกเริ่มที่เดียวของการส่งข้อมูล จะส่งข้อมูลจะส่งรหัสบอร์ดดอด (Baudot code) ซึ่งใช้ 5 บิตในการแทนอักขระ 1 ตัว ส่วนที่ตามหลังข้อมูลก็จะเป็นบิตพาริตี้ ซึ่งอาจจะใช้หรือไม่ใช้ก็ได้ บิตพาริตี้จะทำหน้าที่เป็นตัวตรวจสอบ ความถูกต้องของสัญญาณที่ได้รับ บิตพาริตี้อาจจะเป็นแบบคู่ (even) หรือแบบคี่ (odd) ก็ได้ หมายความว่า ถ้าหากเป็นพาริตี้คู่ จำนวนบิตที่เป็น "1" ในช่วงบิตข้อมูลกับบิตพาริตี้รวมกันแล้วต้องเป็นเลขคู่ผู้ส่งข้อมูล จะทำหน้าที่ตรวจสอบข้อมูลแล้วใส่บิตพาริตี้เอง

ฝ่ายรับ เมื่อรับสัญญาณแล้วก็จะต้องตรวจสอบดูว่าเป็นจริงดังสถานการณ์ที่ตั้งไว้หรือไม่หากผิดพลาด ก็หมายความว่า สัญญาณที่รับนั้นผิดพลาดไปจากสถานีที่ส่งออกมา ทั้งนี้ทั้งนั้นจะต้องผิดเป็นจำนวนคี่ เท่านั้นคือ ผิดไป 1 บิต 3 บิต หรือ 5 บิตพร้อมกัน จึงจะตรวจสอบได้ว่าผิดเป็นจำนวนคู่ผลรวมของ จำนวนบิตที่เป็น "1" ก็ยังเป็นคู่อยู่ดี

ทั้งนี้ทั้งนั้นไม่ได้หมายความว่า พาริตี้จะตรวจสอบการผิดพลาดเป็นจำนวนคู่ได้ความจริงแล้วสามารถตรวจสอบความผิดพลาด ได้เหมือนพาริตี้คู่แต่แทนที่จะตรวจสอบดูว่าสัญญาณ ที่รับเข้ามามีจำนวนคู่ ก็ตรวจสอบดูว่ามีจำนวนคี่หรือเปล่า อย่างไรก็ตาม โอกาสที่จะผิดพลาดเป็น 2, 4, 6 หรือ 8 บิตพร้อมกันมีน้อยมาก

ย้อนกลับมาดูสัญญาณอะซิงโครนัสใหม่ หลังจากบิตพาริตี้แล้วจะต้องมีบิตสต็อปซึ่งเป็น "1" ความกว้าง ของบิตสต็อปอาจจะเป็น 1, 1.5 หรือ 2 พัลส์ของสัญญาณนาฬิกา ซึ่งแล้วแต่ผู้รับและผู้ส่งจะตกลงใช้กันเอง

การเริ่มใช้พอร์ตอนุกรมจึงจำเป็นต้องตั้งค่าต่างๆสำหรับการสื่อสาร ซึ่งมีดังต่อไปนี้ คือ

1. ความเร็วของการส่ง
2. ความยาวของรหัส 1 อักขระ
3. บิตตรวจสอบ
4. จำนวนบิตสต็อป

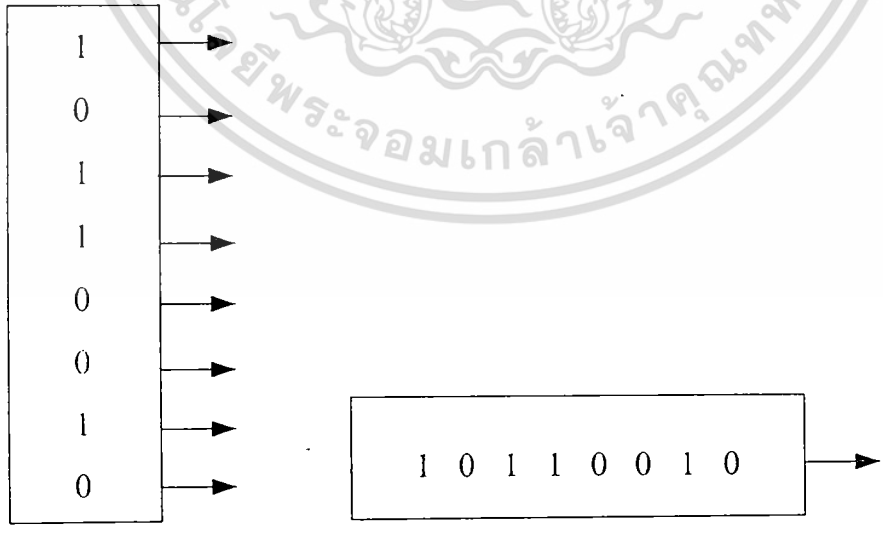
2.2.8.4 การสื่อสารแบบซิงโครนัส

ข้อแตกต่างระหว่างวงจรการส่งข้อมูลอนุกรมแบบซิงโครนัสและแบบอะซิงโครนัสก็คือ ความต่อเนื่อง ของข้อมูลที่ส่ง ในแบบซิงโครนัส ข้อมูลที่ส่งออกมาเป็นแบบต่อเนื่อง ไม่มีบิตสตาร์ทหรือบิตสต็อป หรือแม้กระทั่งบิตพาริตี้รูปแบบที่ใช้ในการส่งข้อมูลแบบซิงโครนัสจึงแตกต่างไปจากการส่งข้อมูล แบบอะซิงโครนัส เช่น รูปแบบของบริษัทไอบีเอ็ม ใช้รูปแบบไบซิงค์ (binary synchronous transmission)

การซิงโครไนซ์จะทำในระดับอักขระซึ่งหมายความว่าอักขระแต่ละตัวมีขอบเขตที่แน่นอนแต่ละอักขระ ไม่มีบิตสตาร์ท หรือบิตสต็อปเหมือนอะซิงโครนัส การซิงโครนัสจะกระทำที่จุดเริ่มต้นของการส่งข้อมูล สถานีส่งจะส่งสัญญาณที่เรียกว่า ตัวอักษรนำ (leading pad character)ไปยังสถานีรับก่อนที่จะเริ่มส่งข้อมูล ตัวอักษรนำจะประกอบด้วย "0" และ "1" สลับกัน เพื่อให้สถานีรับจัดสัญญาณนาฬิกาให้ตรงกันก่อนส่ง ข้อมูลก็จะมีการส่งอักขระที่เรียกว่า syn ตามหลังตัวอักษรนำออกมาสถานีส่งจำเป็นจะบอกความยาว ของข้อมูลมาในกลุ่มนี้ และต้องบอกเครื่องหมายที่เป็นตัวบอกจุดเริ่มต้นของข้อมูลด้วย

2.2.8.5 ระยะเวลาและอัตราการส่งข้อมูล

ตัวอย่างของการส่งข้อมูลที่มีขนาด 8บิตจากระบบไมโครโปรเซสเซอร์ส่งออกที่ช่องสื่อสาร แบบอนุกรม แสดงได้ดังรูป



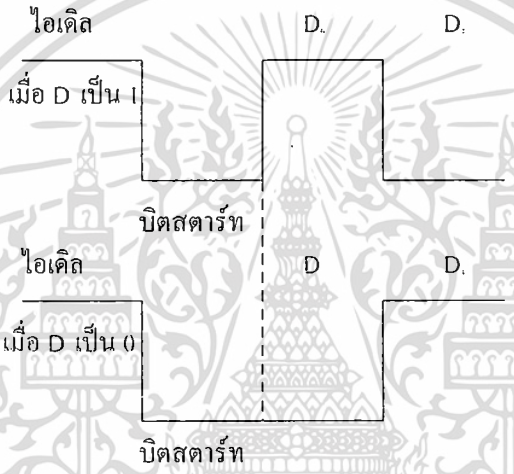
รูปที่ 2.15 ตัวอย่างของการส่งข้อมูลที่มีขนาด 8 บิตจากระบบไมโครโปรเซสเซอร์ส่งออกที่ช่องสื่อสาร แบบอนุกรม

ในการส่งข้อมูลแบบอนุกรมมีสิ่งที่สำคัญที่จะต้องพิจารณาคือ ความเร็วของข้อมูลในการส่งซึ่งเราเรียกว่า อัตราบิต (bit rate) ตามที่กล่าวมา และกรณีที่ให้อัตราการเปลี่ยนแปลงของสัญญาณ 1 ครั้งต่อข้อมูล 1 บิต จะได้อัตราบิตเท่ากับอัตราบอร์ด

อัตราบอร์ดที่ใช้ในการส่งข้อมูลทั่วไปคือ 110,150,300,1200,2400,4800 และ 9600 สมมติว่า ถ้าต้องการส่งข้อมูลด้วยอัตราบอร์ด 2400 บอร์ด ข้อมูลจะได้รับการส่งออกไป

2.2.8.6 บิตสตาร์ทและบิตสต็อป

การรับส่งข้อมูลแบบอะซิงโครนัสจะต้องมีการบอกจุดเริ่มต้นและจุดสิ้นสุดของเฟรม (frame) ข้อมูลเสมอ โดยปกติจะให้สภาวะไอเดิลเหมือนเช่นบิตสต็อป ดังนั้นส่วนของบิตสตาร์ตจะตรงข้ามกับไอเดิล โดยทั่วไปของการส่งข้อมูลจะใช้ 1 บิตเป็นตัวบอกสตาร์ท และใช้ลอจิก "0" เป็นตัวบอกบิตสตาร์ท ส่วนบิตสต็อปจะยาวกว่าที่กำหนดก็ได้ก่อนที่จะเริ่มต้นเฟรมใหม่



รูปที่ 2.16 แสดงบิตสตาร์ท

2.2.8.7 RS-232, RS-232-C

ย่อมาจาก Recommended Standard-232, และคำว่า C แทนรุ่นที่ 3 ของมาตรฐาน คำนี้ไม่ใช่หมายเลขส่วนประกอบของวิทยุสื่อสารสมัครเล่น แต่เป็นวิธีการรับส่งข้อมูลมาตรฐานผ่านสายอนุกรม และใช้กับโมเด็ม เครื่องพิมพ์ และอุปกรณ์อนุกรมอื่น ๆ นอกจากนี้ยังมีเรื่องราวทางเทคนิคที่เกี่ยวข้องกับมาตรฐานอีกมากมาย ดังนี้

RS-232 เป็นมาตรฐานของ Electrical Industries Association ในเรื่องราวสื่อสารแบบไม่ประสานจังหวะ มาตรฐานนี้ได้กำหนดสายต่อ ช่วงเวลาและการส่งสัญญาณที่ใช้ระหว่างคอมพิวเตอร์กับอุปกรณ์ต่อพ่วง โดยใช้สายเชื่อมต่อ DB แบบ 25 และ 9 เข็ม

RS-232- ใช้กับการสื่อสารแบบไม่ประสานจังหวะระหว่างคอมพิวเตอร์กับอุปกรณ์ต่อพ่วง ความยาวของสายมากที่สุดคือ 50 ฟุต แต่จะขยายให้เกินกว่านี้ได้โดยใช้สายที่มีคุณภาพดีมาก ไลน์ไดรเวอร์เพื่อกระตุ้นสัญญาณเหรือโมเด็มแบบ short-haul

RS-232-C คำว่า C แทนรุ่นที่ 3 ของมาตรฐาน ซึ่ง RS-232-C นี้ทำงานได้เหมือนกับมาตรฐาน CCITT V.24

บทที่ 3
การสร้างและการออกแบบ

ระบบคิวในโครงงานนี้ มีหลักการทำงานคือ เริ่มจากเมื่อผู้มาใช้บริการกดพิมพ์บัตรคิว ระบบจะทำการพิมพ์บัตรคิวตามลำดับของคิวที่มีอยู่ในระบบนั้น โดยที่บัตรคิวนี้จะแสดงเวลาที่ออกบัตร และมีการประมาณค่าเวลาที่จะเข้ารับบริการได้ โดยได้มีการจำลองเหตุการณ์ว่า เมื่อจำนวนคิวที่เหลือมีมากกว่า 10 คิว ระบบจะสอบถามไปยังผู้ใช้บริการว่าต้องการให้ระบบแจ้งเตือนเมื่อใกล้จะถึงคิวนั้นหรือไม่ โดยที่การแจ้งเตือนนั้นจะทำโดยส่ง SMS ไปยังเลขหมายที่ผู้ใช้บริการได้ระบุไว้ และระบบจะทำการส่ง SMS ขณะที่มีจำนวนคิวเหลืออยู่ 4 คิวก่อนจะถึงคิวที่มีการระบุเลขหมายนั้น

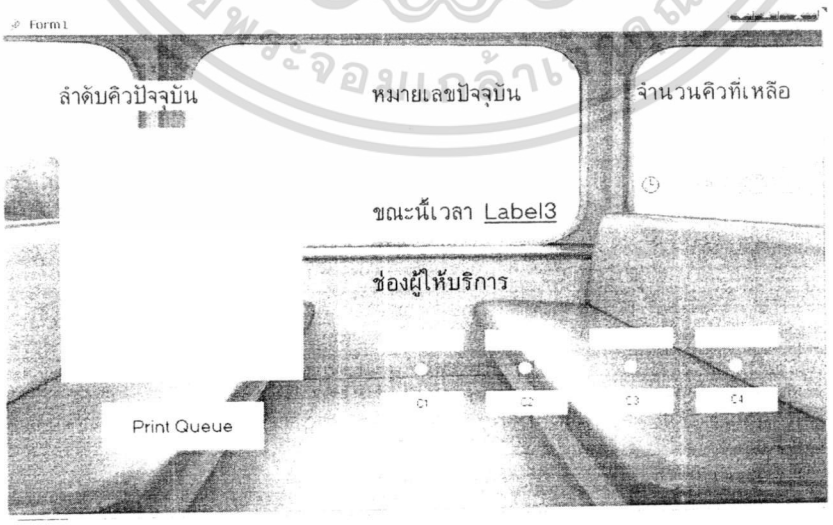
การสร้างและการออกแบบของระบบนี้ จะแบ่งออกเป็น 2 ส่วน คือ ส่วนของคอมพิวเตอร์ และส่วนของไมโครคอนโทรลเลอร์ โดยส่วนโปรแกรมในคอมพิวเตอร์จะแยกออกเป็น

3.1 ส่วนของคอมพิวเตอร์

3.1.1 ส่วนของหน้าจอที่ใช้ในการติดต่อกับผู้มาใช้บริการ

ในส่วนนี้ได้ทำการออกแบบหน้าจอที่ใช้ในการติดต่อกับผู้มาใช้บริการ เป็นหน้าจอที่ผู้ใช้บริการต้องกด เพื่อรับบัตรคิวพร้อมทั้งแสดงรายละเอียดของคิว และช่องให้บริการต่างๆ โดยมีส่วนประกอบดังนี้

- ส่วนของหมายเลขปัจจุบัน และจำนวนคิวที่เหลือ
- ส่วนแสดงเวลา
- ส่วนแสดงช่องให้บริการ (C1 – C4)
- ส่วนปุ่มกด Print Queue
- ส่วนแสดงลำดับคิวปัจจุบัน

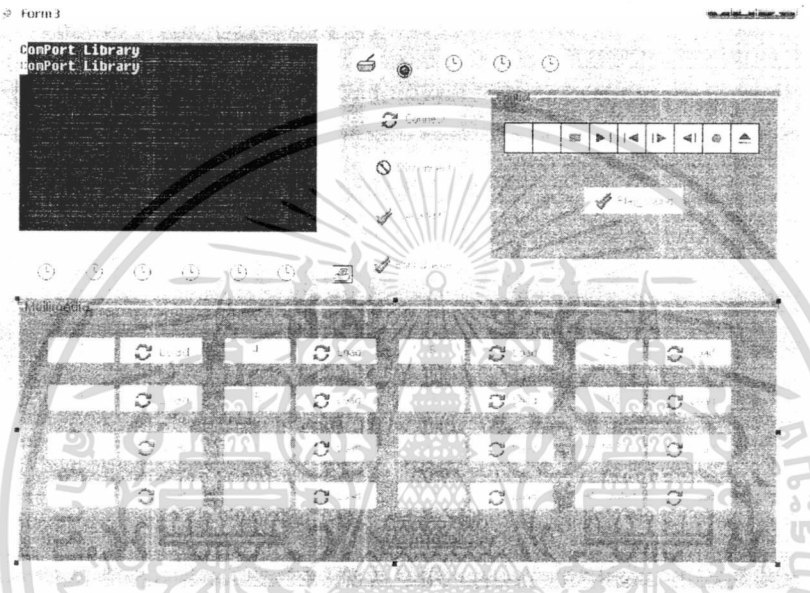


รูปที่ 3.1 แสดงหน้าจอที่ใช้ในการติดต่อกับผู้มาใช้บริการ

3.1.2 ส่วนของหน้าจอที่ใช้ในการเล่นเสียงและส่ง SMS

ในส่วนนี้ได้ทำการออกแบบหน้าจอที่ใช้ในการเล่นเสียงว่าหมายเลขใดเข้ารับบริการที่ช่องบริการใด และจะส่ง SMS เมื่ออีก 4 คิวจะถึงคิวที่ต้องการให้แจ้งเตือน SMS โดยมีส่วนประกอบดังนี้

- ส่วนของการส่ง SMS
- ส่วนการเล่นเสียง
- ส่วนไฟล์เสียงที่บันทึกไว้



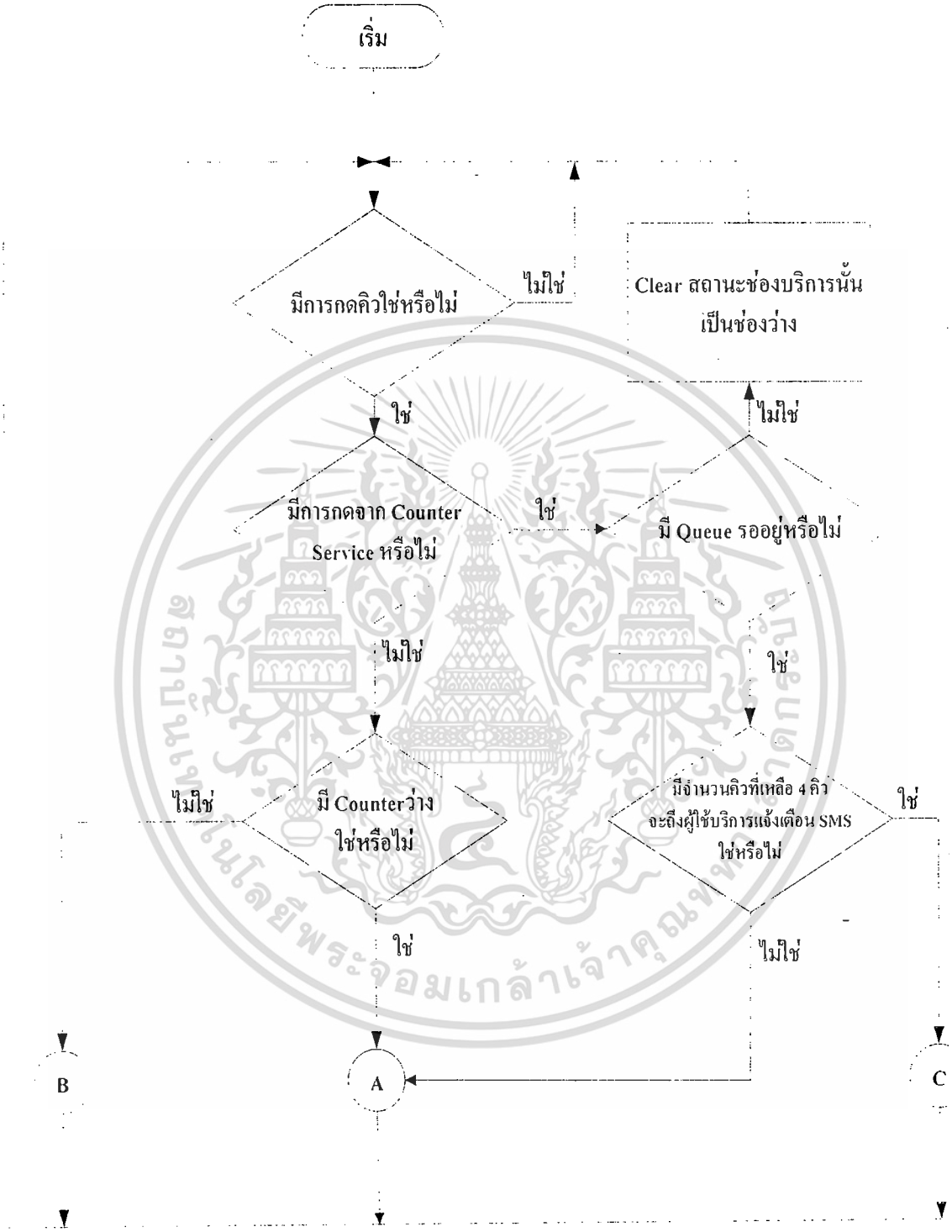
รูปที่ 3.2 แสดงหน้าจอการเล่นเสียงและส่ง SMS

3.1.3 ส่วนหน้าจอที่ใช้ในการเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

ในส่วนนี้ได้ทำการออกแบบหน้าจอที่ใช้ในการเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์ เป็นการส่งผ่านข้อมูลไป – กลับระหว่างคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

รูปที่ 3.3 แสดงหน้าจอที่ใช้ในการเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

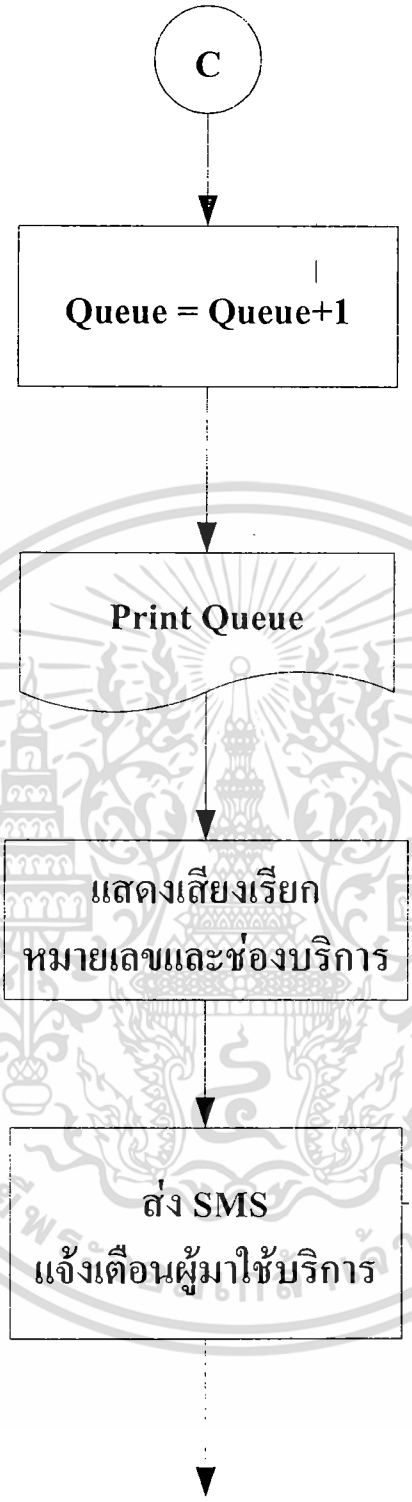
โดยทำการเขียนโปรแกรมควบคุมการทำงานของระบบ ซึ่งมีผังการทำงาน (Flow Chart) ดังนี้



รูปที่ 3.4 แสดงผังการทำงานของโปรแกรม

รูปที่ 3.5 แสดงกรณีเข้ารับบริการและเรียกใช้บริการ

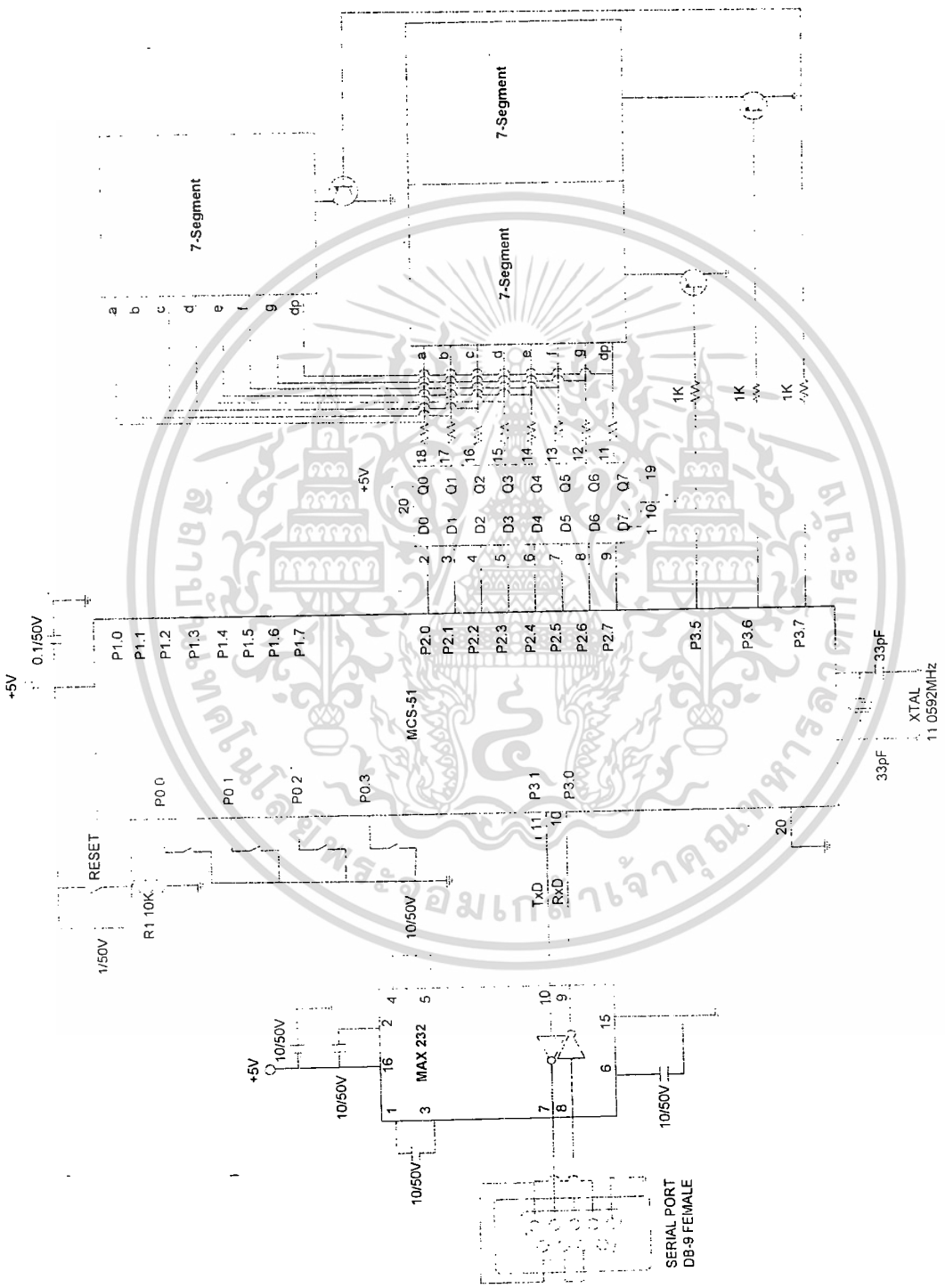
รูปที่ 3.6 แสดงกรณีรอรับบริการยังไม่มีการเรียก



รูปที่ 3.7 แสดงกรณีเข้ารับบริการและแจ้งเตือน SMS

3.2 ส่วนของไมโครคอนโทรลเลอร์

การทำงานในส่วนนี้ไมโครคอนโทรลเลอร์จะคอยรับอินพุตจากการกดสวิตช์ที่ต่ออยู่กับขาพอร์ต P0.0 – P0.3 และนำมาประมวลผล จากนั้นจึงนำค่าที่ได้มาแสดงบนจอแสดงผล 7 (7-segment) ส่วนโดยวิธีมัลติเพล็กซ์ และจะส่งผลไปยังคอมพิวเตอร์ โดยผ่านพอร์ตอนุกรม



รูปที่ 3.8 แสดงการต่อไมโครคอนโทรลเลอร์เข้ากับจอแสดงผลและพอร์ตอนุกรม

การเขียนโปรแกรมควบคุมการทำงานของระบบ มีผังการทำงาน (Flow Chart) ดังนี้

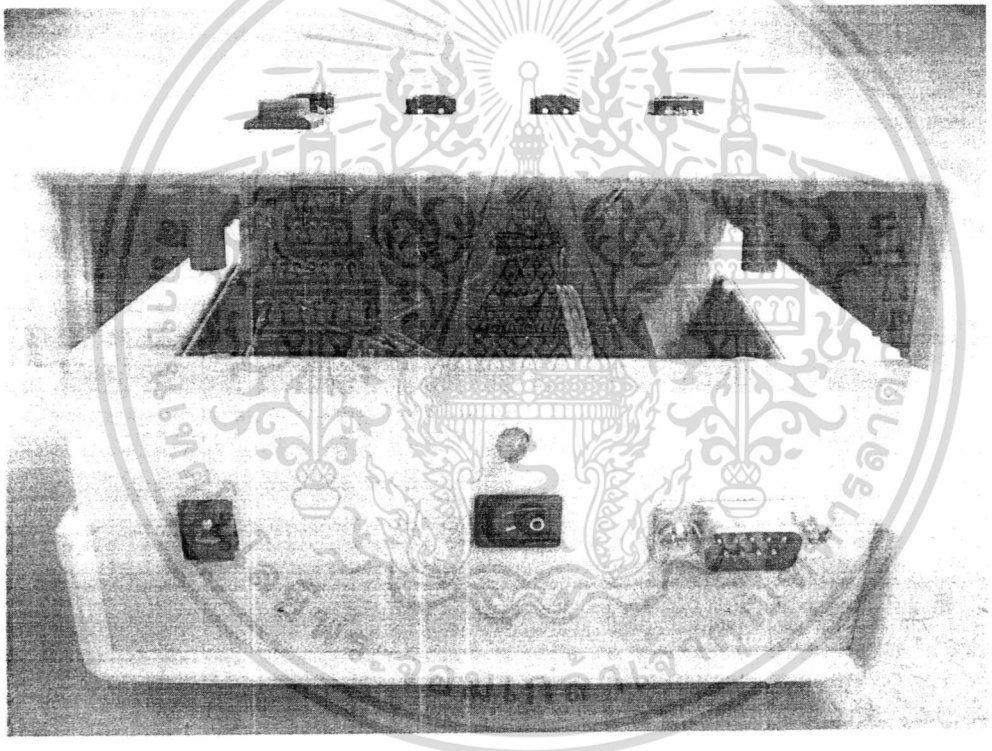
รูปที่ 3.9 แสดงผังการทำงานเมื่อมีการกดเรียกเข้ารับบริการ

บทที่ 4
การทดลองและผลการทดลอง

จากการออกแบบระบบในบทที่ 3 และเขียนโปรแกรมควบคุมการทำงานของแต่ละส่วน แล้วนำมาทดสอบการทำงานร่วมกัน จะได้ผลของส่วนต่างๆ ดังนี้

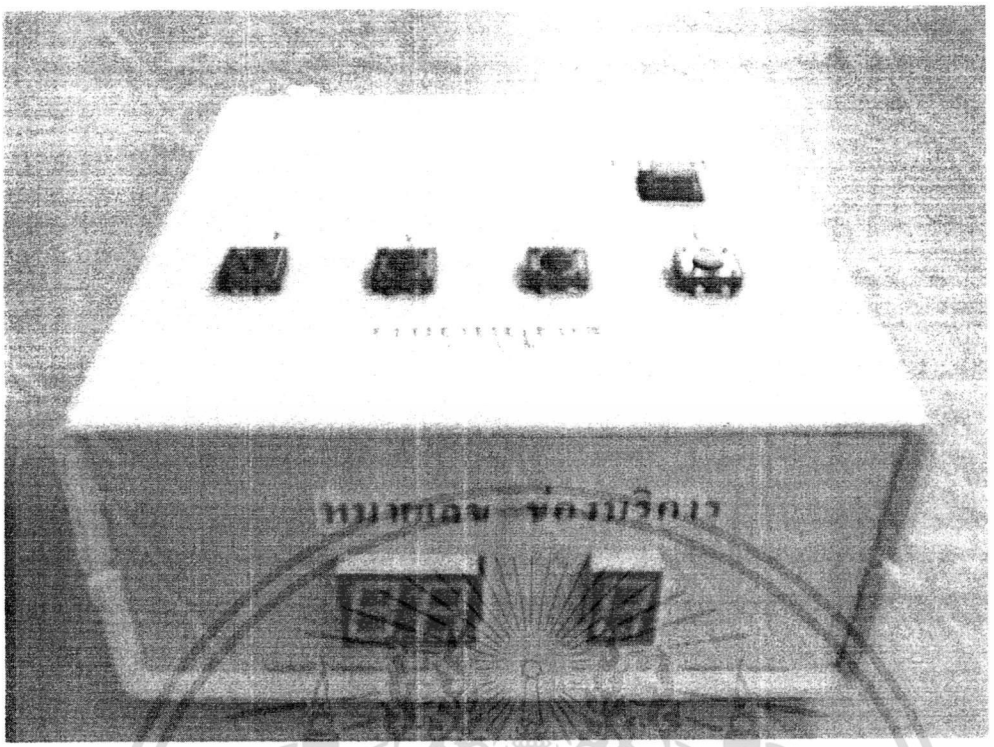
4.1 ส่วนของไมโครคอนโทรลเลอร์

การทำงานในส่วนของไมโครคอนโทรลเลอร์จะเชื่อมต่อเข้ากับจอแสดงผล 7 ส่วน (7-segment) เพื่อแสดงหมายเลขคิวและหมายเลขช่องให้บริการ โดยมีการรับส่งข้อมูลกับคอมพิวเตอร์ เพื่อตรวจสอบสถานะของลำดับคิว

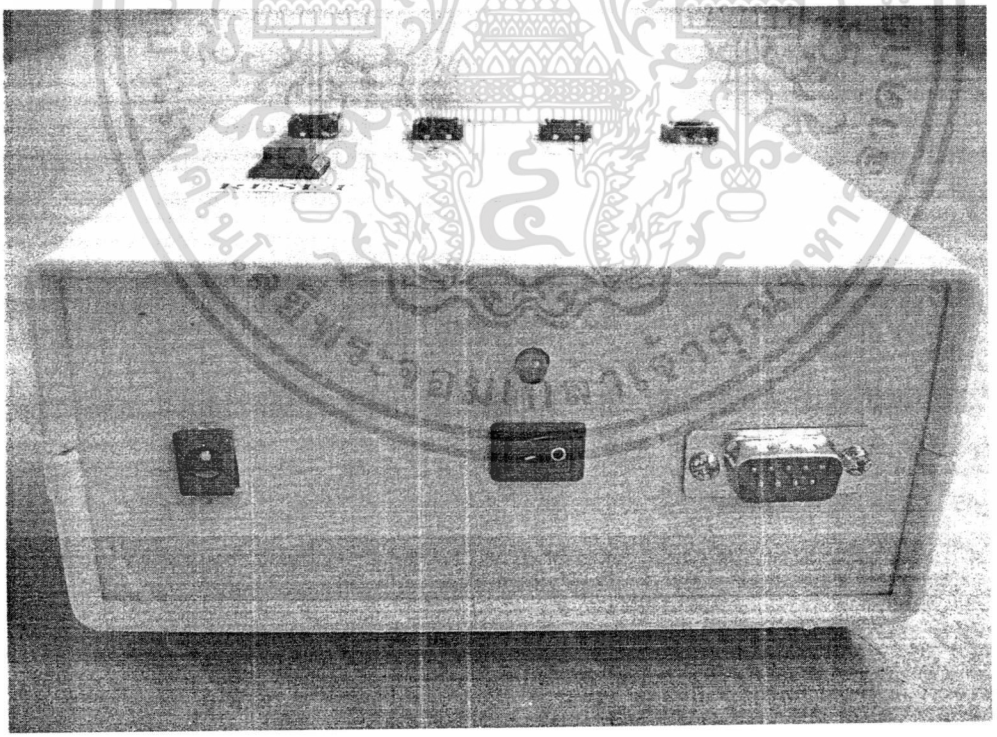


รูปที่ 4.1 แสดงวงจรภายในของตัวอุปกรณ์

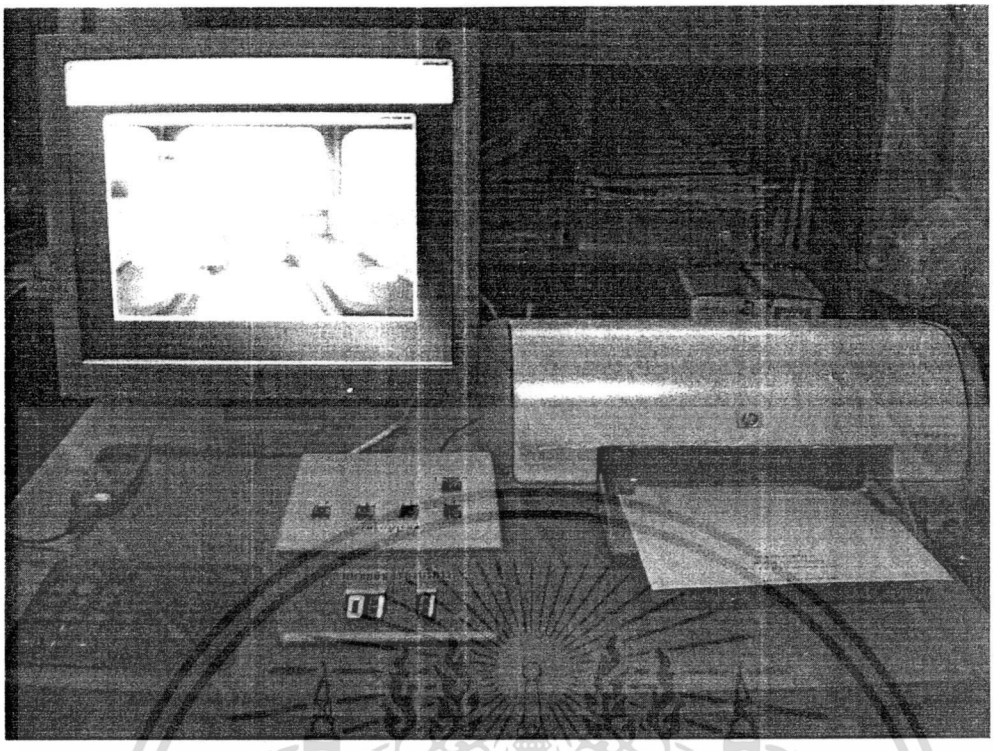
โดยจากรูปที่ 4.2 จะเห็นได้ว่าส่วนของไมโครคอนโทรลเลอร์ด้านหน้าจะเป็นจอแสดงผล 7 ส่วน แสดงหมายเลขช่องให้บริการ และแสดงหมายเลขผู้รับบริการ ส่วนด้านบนจะเป็นปุ่มของผู้ให้บริการที่ใช้ในการกดให้บริการ และปุ่ม Reset ระบบ ส่วนด้านหลัง ตามรูปที่ 4.3 จะมีส่วนของไฟเลี้ยง สวิทช์ และพอร์ตอนุกรม



รูปที่ 4.2 แสดงด้านหน้าและด้านบนของตัวอุปกรณ์



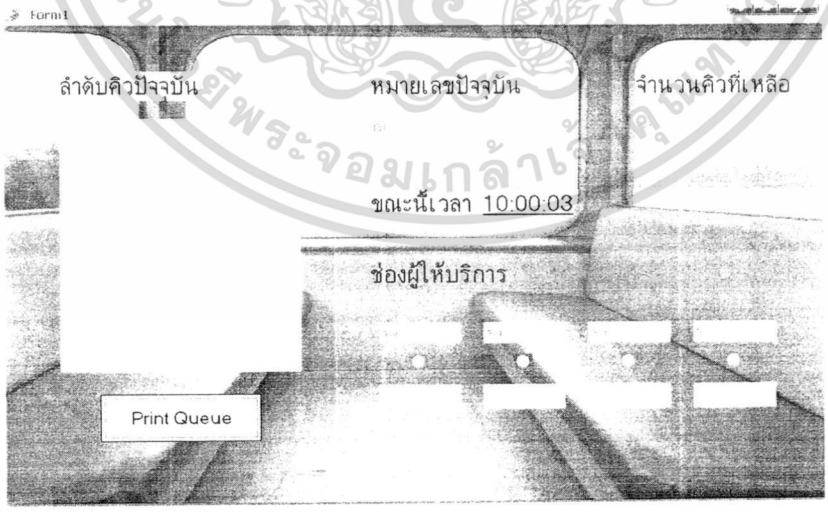
รูปที่ 4.3 แสดงด้านหลังของตัวอุปกรณ์



รูปที่ 4.4 แสดงโครงสร้างการทำงานของระบบ

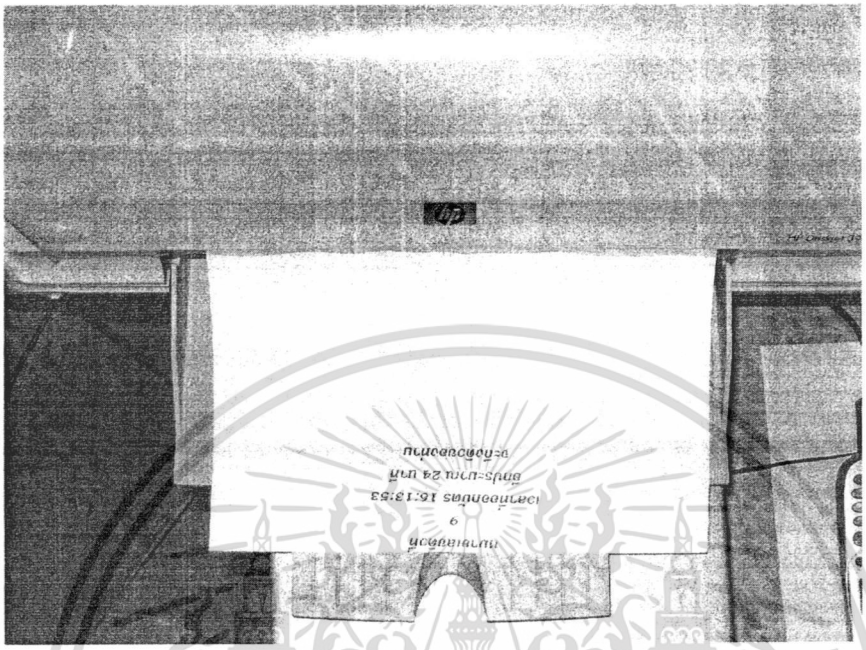
4.2 การรับข้อมูลของผู้มาใช้บริการ

เมื่อทำการรันโปรแกรมแล้วจะได้หน้าจอที่อยู่ในสถานะเตรียมพร้อมสำหรับการให้บริการที่มีลักษณะดังนี้



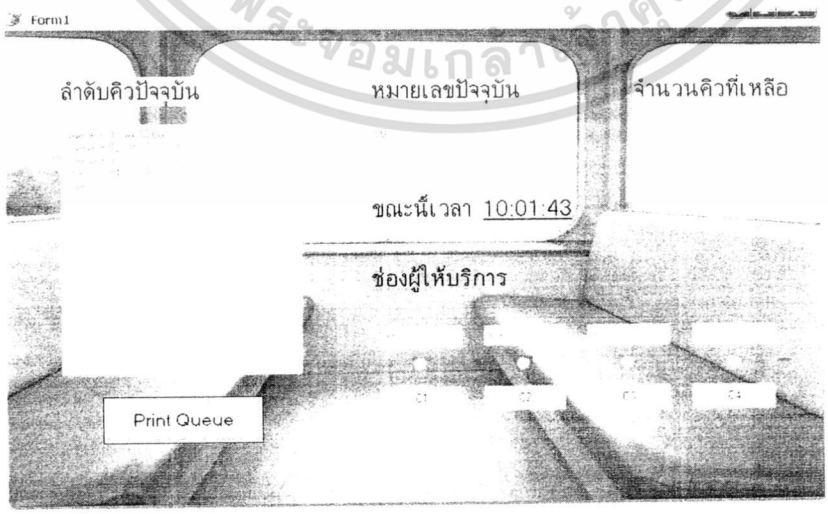
รูปที่ 4.5 แสดงหน้าจอเมื่อรันโปรแกรมในสถานะเตรียมรอรับการใช้บริการ

เมื่อผู้ใช้บริการกดปุ่ม Print Queue โปรแกรมในส่วนที่ควบคุมเครื่องพิมพ์จะสั่งให้เครื่องพิมพ์พิมพ์บัตรคิวออกมาตามหมายเลขคิวที่ได้รับ

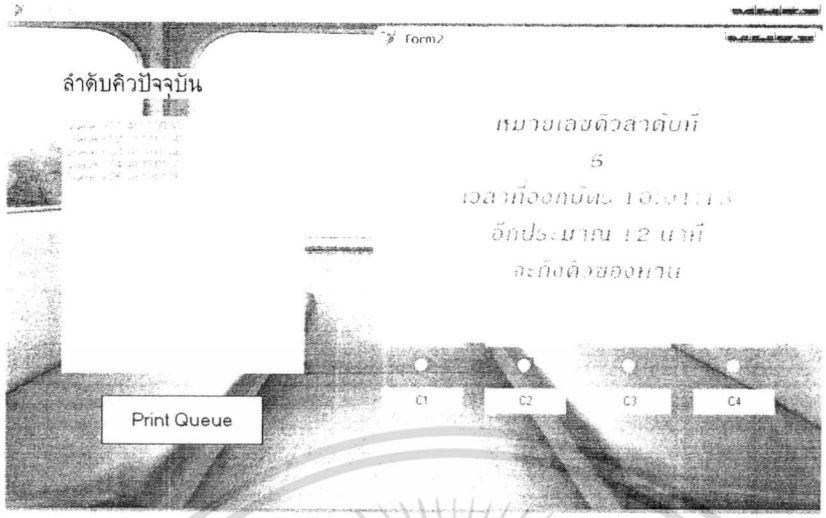


รูปที่ 4.6 แสดงการพิมพ์บัตรคิว

ในส่วนของเคาน์เตอร์ให้บริการ หลังจากมีการกดปุ่ม Print Queue โปรแกรมจะทำการตรวจสอบค่าที่รับมาพร้อมทั้งปรับจำนวนคิวที่เหลือตามจำนวนครั้งที่กดเข้ามาใช้บริการ และแสดงจำนวนบัตรคิวที่ถูกพิมพ์ออกไป ดังรูปที่ 4.7 เมื่อกดปุ่ม Print Queue แล้วจะแสดงบัตรคิวจำลองขึ้น โดยมีรายละเอียด คือ ลำดับคิว เวลาที่ออกบัตร และเวลาโดยประมาณก่อนจะถึงคิว ดังรูปที่ 4.8



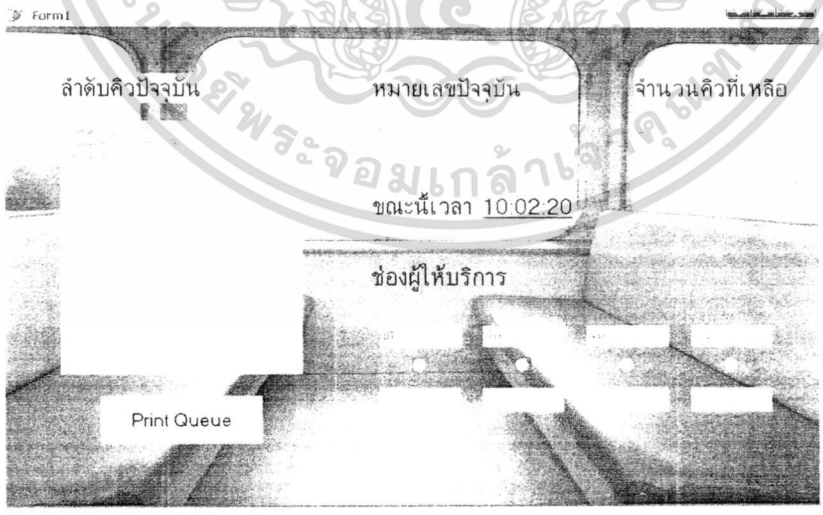
รูปที่ 4.7 แสดงหน้าจอเมื่อมีการกด Print Queue ขอรับบริการ



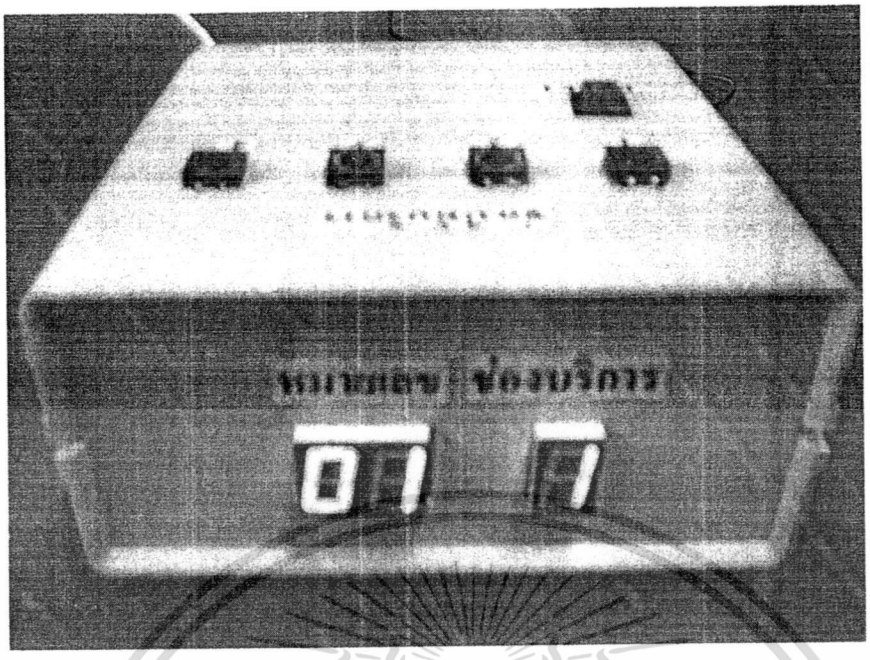
รูปที่ 4.8 แสดงหน้าจอเมื่อมีการกดขอรับบริการจนถึงคิวที่ 5 และแสดงบัตรคิวที่ 5

4.3 การให้บริการของช่องบริการ

เมื่อมีผู้มาใช้บริการทางช่องบริการจะทำการเรียกหมายเลขที่จะใช้บริการตามลำดับ โดยได้ทำการจำลองให้มีช่องให้บริการทั้งหมด 4 ช่อง และมีผู้มาใช้บริการเป็นจำนวน 5 คิว เมื่อช่องบริการที่ 1 จะทำการเรียกผู้รอรับบริการหมายเลข 1 มารับบริการ ซึ่งจะมีเสียงพูดหมายเลขคิวที่รับ และจอแสดงผล 7 ส่วนจะแสดงหมายเลขที่ใช้บริการปัจจุบันว่าคือ หมายเลข 1 และช่องให้บริการว่า คือช่องที่ 1 ดังรูปที่ 4.9 และ 4.10 พร้อมทั้งจะลดจำนวนคิวที่เหลือลงเป็น 4 คิว

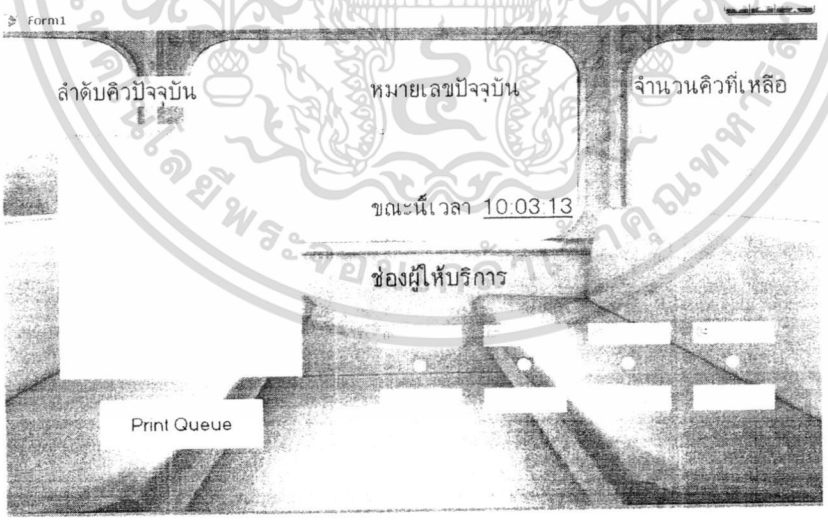


รูปที่ 4.9 แสดงหน้าจอเมื่อช่องให้บริการที่ 1 ให้บริการ

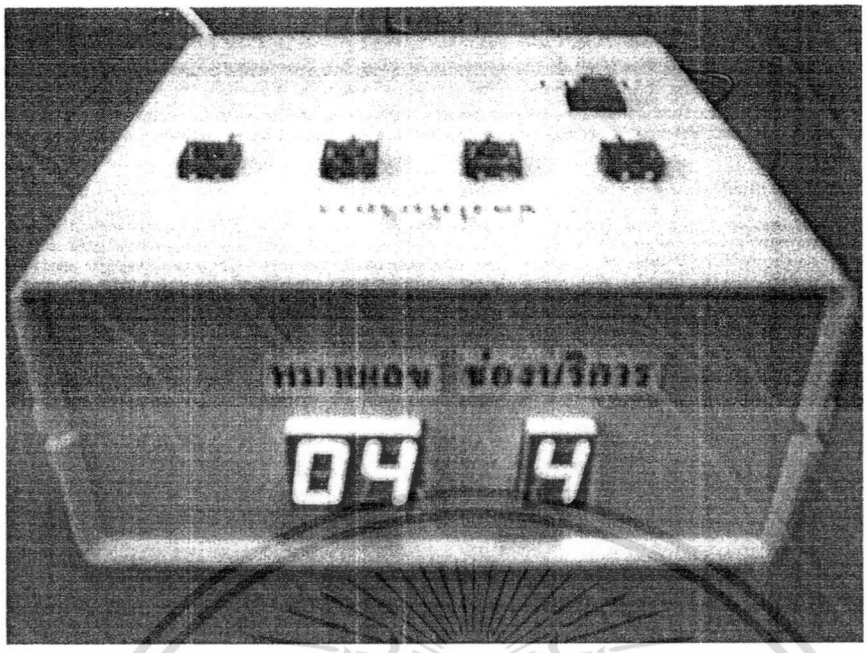


รูปที่ 4.10 แสดงส่วนของจอแสดงผล 7 ส่วน เมื่อช่องบริการที่ 1 ให้บริการคิวที่ 1

เมื่อช่องให้บริการมีผู้ใช้บริการครบทั้ง 4 ช่อง จะทำให้หมายเลขปัจจุบันที่ใช้บริการเปลี่ยนเป็นหมายเลข 4 รวมทั้งจำนวนคิวที่เหลือจะกลายเป็น 1 คิว และจอแสดงผล 7 ส่วนจะแสดงช่องที่ให้บริการล่าสุด คือ ช่องบริการที่ 4 หมายเลขคิวที่ 4 ดังรูปที่ 4.11 และ 4.12



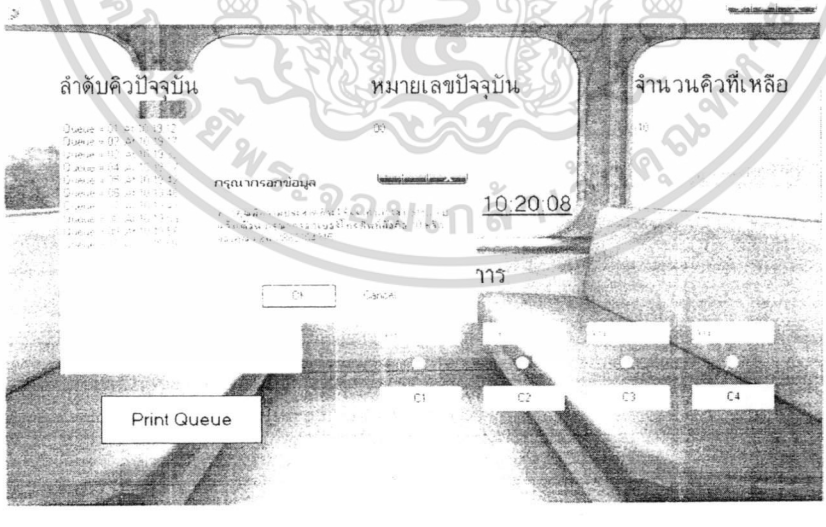
รูปที่ 4.11 แสดงหน้าจอเมื่อช่องให้บริการมีผู้ใช้บริการครบทั้ง 4 ช่อง



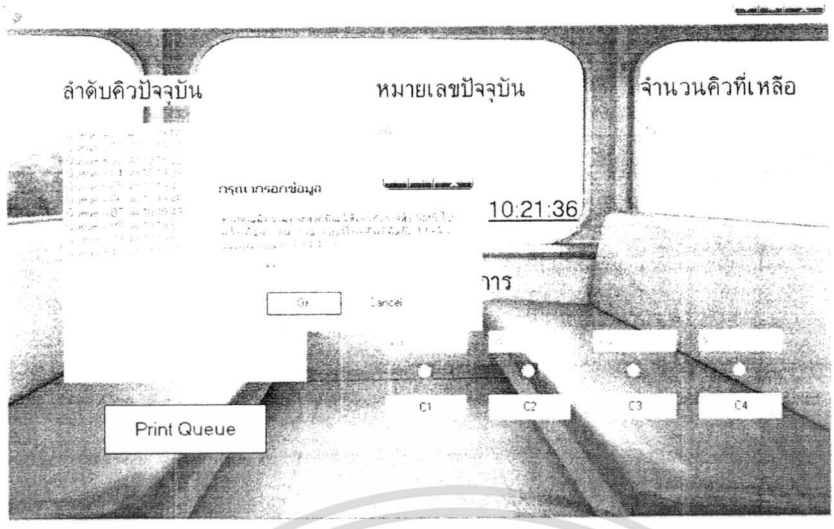
รูปที่ 4.12 แสดงส่วนของจอแสดงผล 7 ส่วน เมื่อช่องบริการที่ 4 ให้บริการคิวที่ 4

4.4 การให้บริการแจ้งเตือนผ่านระบบ SMS

เมื่อมีจำนวนคิวที่เหลือมากกว่า 10 คิว จะมีหน้าต่างปรากฏขึ้นมาถามว่ามีความประสงค์จะให้แจ้งเตือนผ่านทางระบบ SMS หรือไม่ หากผู้ใช้บริการต้องการ ให้ทำการกรอกหมายเลขโทรศัพท์เคลื่อนที่ 10 หลักที่จะให้แจ้งเตือน โดยจะมีตัวอย่างการกรอกหมายเลขที่ถูกต้องอยู่ด้วย ดังรูปที่ 4.13 และรูปที่ 4.14



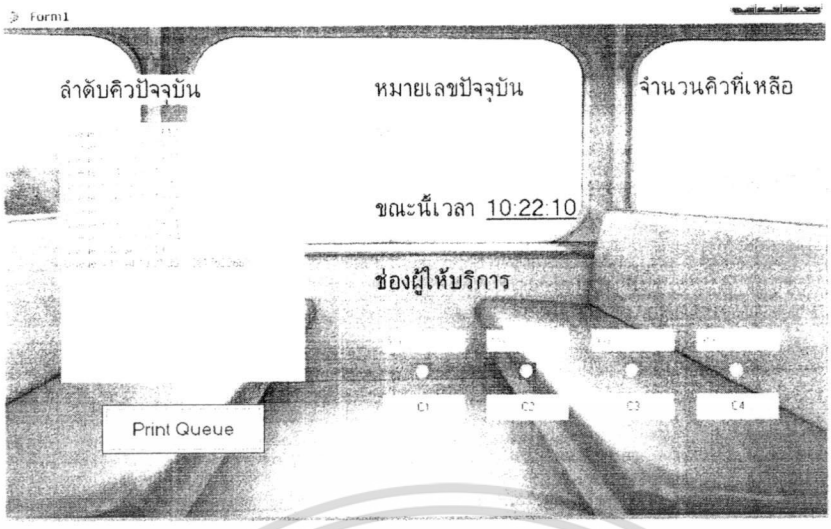
รูปที่ 4.13 แสดงหน้าจอเมื่อมีจำนวนคิวที่เหลือมากกว่า 10 คิว และหน้าต่างให้กรอกหมายเลขโทรศัพท์เคลื่อนที่



รูปที่ 4.14 แสดงหน้าจอเมื่อมีจำนวนคิวที่เหลือมากกว่า 10 คิว และหน้าต่างเมื่อกรอกหมายเลขโทรศัพท์เคลื่อนที่แล้ว

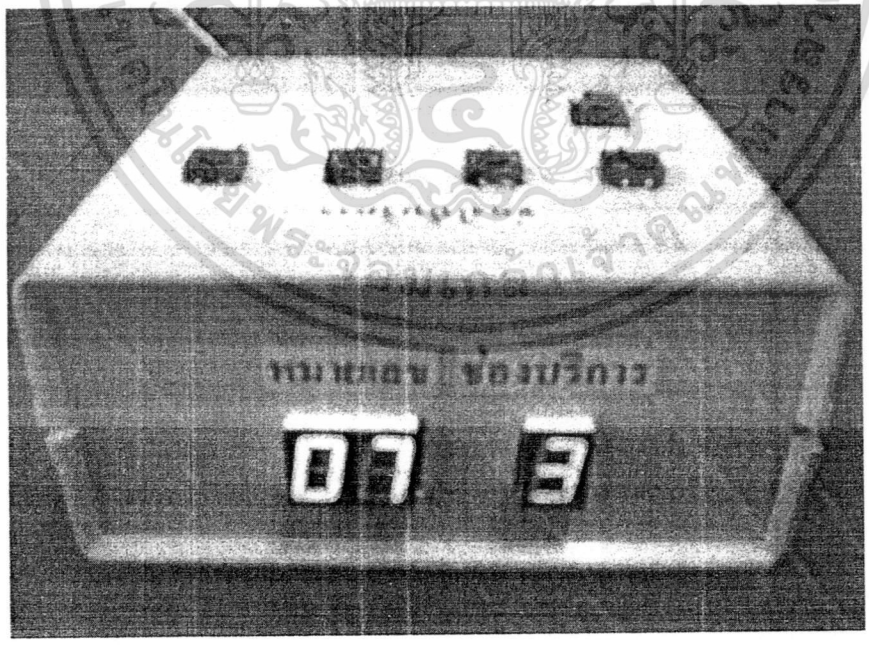
หลังจากกรอกหมายเลขโทรศัพท์เคลื่อนที่แล้ว ที่หน้าจอจะแสดงหมายเลขบัตรคิวที่ได้รับ และเมื่อสังเกตรูปที่ 4.16 จะเห็นว่าจำนวนคิวที่เหลือเป็น 11 คิว และในช่องลำดับคิวปัจจุบัน จะแสดงคิวที่รอรับบริการ เวลาที่เข้ามาใช้บริการ และในคิวที่ 11 จะแสดงหมายเลขโทรศัพท์เคลื่อนที่ที่ต้องการให้แจ้งเตือนโดยผ่านทางระบบ SMS

รูปที่ 4.15 แสดงหน้าจอของบัตรคิวที่ 11

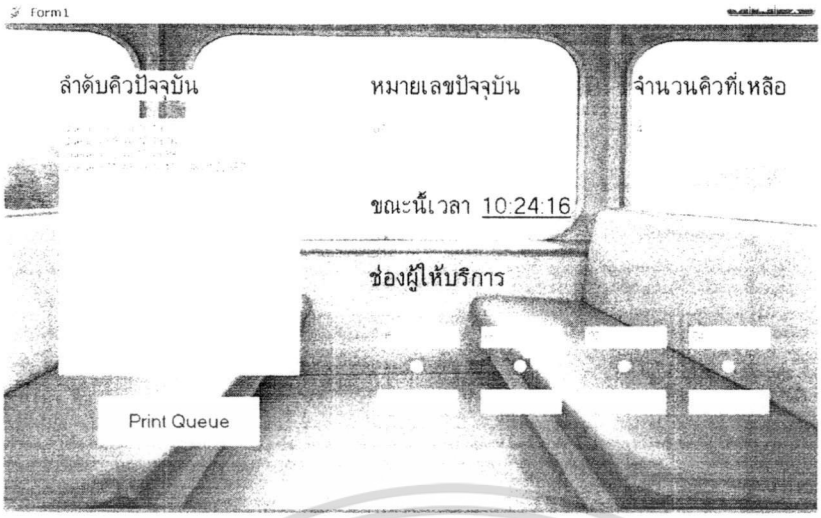


รูปที่ 4.16 แสดงหน้าจอเมื่อทำการกรอกหมายเลขโทรศัพท์เคลื่อนที่เสร็จแล้ว จะปรากฏเลขหมายโทรศัพท์ในคิวลำดับที่ 11

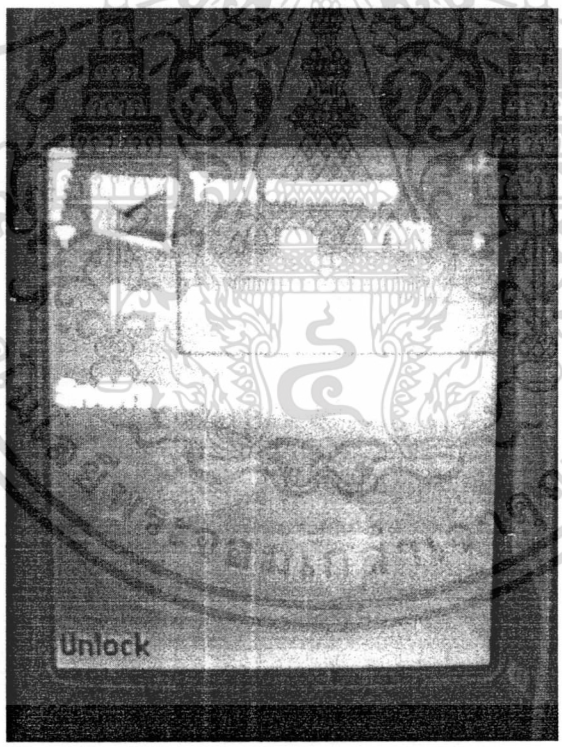
เมื่อมีการให้บริการจนคิวที่เหลือมีจำนวน 4 คิว ก่อนที่จะถึงคิวที่มีการร้องขอให้แจ้งเตือน ตามกรณีที่ช่องให้บริการที่ 3 ให้บริการคิวที่ 7 โปรแกรมจะทำการตรวจสอบหมายเลขโทรศัพท์เคลื่อนที่ที่มีการร้องขอให้แจ้งเตือน และทำการส่ง SMS แจ้งเตือนไปยังเลขหมายดังกล่าว ตามรูปที่ 4.17 และรูปที่ 4.18 โดยโทรศัพท์เคลื่อนที่ของผู้ใช้บริการจะได้รับข้อความ ตามรูปที่ 4.19 ว่า "อีก 4 คิวจะถึงคิวท่าน"



รูปที่ 4.17 แสดงส่วนของจอแสดงผล 7 ส่วน เมื่อช่องบริการที่ 3 ให้บริการคิวที่ 7



รูปที่ 4.18 แสดงหน้าจอเมื่อมีการให้บริการจนจำนวนคิวที่เหลือมีเพียง 4 คิวก่อนที่จะถึงคิวที่มีการร้องขอให้แจ้งเตือน

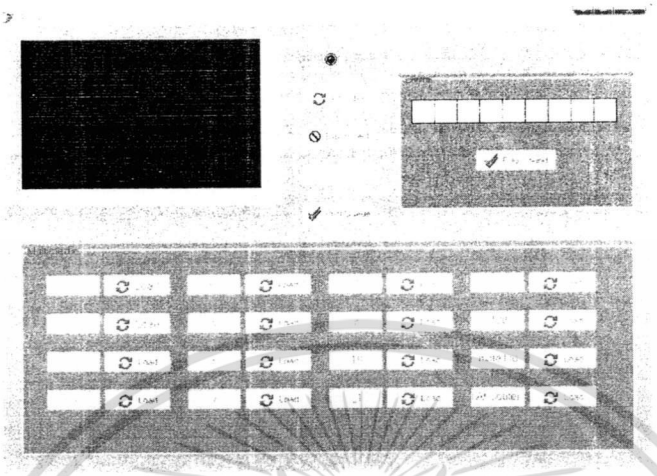


รูปที่ 4.19 แสดงหน้าจอโทรศัพท์เคลื่อนที่เมื่อได้รับ SMS แล้ว

4.5 ระบบเสียงพูด

ในส่วนนี้จะใช้การบันทึกเสียงลงที่เครื่องคอมพิวเตอร์ แล้วเก็บไว้ในรูปแบบของไฟล์เสียง การบันทึกเสียงจะบันทึกทีละคำดังนี้ หนึ่ง สอง สาม สี่ ห้า หก เจ็ด แปด เก้า ศูนย์ สิบ ยี่ ร้อย เชิญหมายเลข ที่ช่องบริการ เมื่อโปรแกรมได้รับค่าจากช่องให้บริการแล้ว โปรแกรมจะทำการตรวจสอบค่าที่รับได้ว่าเป็น

หมายเลขคิวที่เท่าไรและหมายเลขช่องให้บริการใด หลังจากนั้นโปรแกรมจึงจะทำการผสมคำและทำการเล่นเสียงที่เก็บไว้ให้ถูกต้องต่อไป



รูปที่ 4.20 แสดงหน้าจอฟอร์มการให้บริการทางเสียง

4.6 ระบบเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

ในส่วนนี้เป็นส่วนที่ใช้ในการเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์ โดยการเชื่อมต่อผ่านพอร์ตอนุกรม

รูปที่ 4.21 แสดงหน้าจอฟอร์มการเชื่อมต่อคอมพิวเตอร์กับไมโครคอนโทรลเลอร์

บทที่ 5
สรุปและวิจารณ์

จากผลการทดลองพบว่า สามารถทำงานได้ตามวัตถุประสงค์ที่ได้ตั้งไว้ การทำโครงงานชิ้นนี้จะพบว่ามีปัญหาในเรื่องการส่ง SMS ขณะที่ส่งอาจจะเกิดปัญหากับเครือข่ายของระบบโทรศัพท์เคลื่อนที่ ทำให้การส่ง SMS ไม่ถึงผู้ใช้บริการ หรือช้ากว่ากำหนด

ซึ่งในการใช้งานจริง ระบบนี้สามารถที่จะใช้เครื่องคอมพิวเตอร์เพียงเครื่องเดียว ควบคุมการทำงานของทั้งระบบได้ ทำให้เป็นการลดค่าใช้จ่าย และทางคณะผู้จัดทำได้มีการนำความรู้ในเรื่องของไมโครคอนโทรลเลอร์มาประยุกต์ใช้โดยมีการแสดงผลผ่านทางจอแสดงผลเจ็ดส่วน เพื่อให้ง่ายต่อการสังเกตของผู้ที่มาใช้บริการ

บรรณานุกรม

รศ.สมยศ จุณณปิยะ . **การประยุกต์ใช้งาน ไมโครคอนโทรลเลอร์ตระกูล MCS-51** . กรุงเทพฯ : คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง , 2546.

สัจจะ จรัสรุ่งรวีวร และจักรพงษ์ สุขประเสริฐ . **เริ่มต้นอย่างมืออาชีพด้วย Delphi 7 ฉบับสมบูรณ์** . กรุงเทพฯ : บริษัท เอช เอ็น กรุ๊ป จำกัด , 2546.

สัญจกร วุฒิสิทธิกุลกิจ . **หลักการระบบโทรศัพท์เคลื่อนที่** . กรุงเทพฯ : สำนักพิมพ์แก่งจุฬาลงกรณ์มหาวิทยาลัย, 2546.

http://www.dreamfabric.com/sms

http://home.student.utwente.nl/s.p.ekkebus/portfolio/resource/sms_pdu.html

http://www.gsm-modem.de/sms-pdu-mode.html