ระบบจัดการที่จอดรถอัตโนมัติ

AUTOMATIC CARS PARKING MANAGEMENT

โดย

นายณัฐพล วิละรัตน์

นายวิศวะ สมานวิจิตร

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2546

AUTOMATIC CARS PARKING MANAGEMENT

BY

Mr. NATTAPOL WILARAT

Mr. WISSAWA SAMANWIJIT

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF

THE REQUIREMENT FOR THE DEGREE OF

BACHELOR IN DEPARTMENT OF INFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2003

หัวข้อปริญญานิพนธ์	ระบบจัดการที่จอดรถอัตโนมัติ
ชื่อนักศึกษา	นายณัฐพล วิละรัตน์ รหัสประจำตัว 44015689
	นายวิศวะ สมานวิจิตร รหัสประจำตัว 44015713
อาจารย์ที่ปรึกษา	อาจารย์ดลชัย สุขเจริญผล
	อาจารย์ ไพศาล สิทธิโยภาสกุล
ระดับการศึกษา	ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
	สาขาวิศวกรรมสารสนเทศ
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2546

ปริญญานิพนธ์ฉบับนี้ได้รับการอนุมัติเป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

.....

(อาจารย์ดลชัย สุขเจริญผล)

อาจารย์ผู้ควบคุมปริญญานิพนธ์

.....

(อาจารย์ไพศาล สิทธิโยภาสกุล)

อาจารย์ผู้ควบคุมปริญญานิพนธ์

หัวข้อปริญญานิพนธ์ ระบบจัดการที่จอดรถอัตโนมัติ
ชื่อนักศึกษา นายณัฐพล วิละรัตน์ รหัสนักศึกษา 44015689
นายวิศวะ สมานวิจิตร รหัสนักศึกษา 44015713
อาจารย์ที่ปรึกษา อาจารย์ ดลชัย สุขเจริญผล
อาจารย์ ไพศาล สิทธิโยภาสกุล
ระดับการศึกษา ปริญญาตรี วิศวกรรมศาสตรบัณฑิต
สาขาวิศวกรรมสารสนเทศ
ภาควิชา วิศวกรรมสารสนเทศ
ปีการศึกษา 2546

บทคัดย่อ

โครงงานนี้เป็นการนำไมโครคอนโทรลเลอร์, อุปกรณ์ตรวจจับ และคอมพิวเตอร์ มาประยุกต์ใช้ร่วมกันเพื่อจัดการระบบที่จอดรถให้มีประสิทธิภาพ โดยมีวัตถุประสงค์คือ สามารถบอกจำนวนที่ว่างสำหรับจอดรถที่แน่นอนในแต่ละชั้นว่ามีจำนวนเหลือเท่าใด ตำแหน่งใดว่างลงบ้าง โดยโครงงานนี้จะใช้เซ็นเซอร์เป็นอุปกรณ์ในการตรวจจับการเข้าจอดของรถแล้วทำการประมวลผลโดยใช้ไมโครคอนโทรลเลอร์ MCS-51 จากนั้นจะส่งข้อมูลต่อไปยังคอมพิวเตอร์เพื่อแสดงผลเป็นตัวเลขและรูปภาพทางหน้าจอคอมพิวเตอร์ โดยใช้โปรแกรม Visual Basic ในการเขียน Application โปรแกรมแสดงผล และสามารถจัดเก็บข้อมูลของปริมาณรถยนต์ในแต่ละช่วงเวลาในหนึ่งวัน

Thesis Title	Automatic Cars Parking Management	
Student	Mr. Nattapol Wilarat	ID. 44015689
	Mr. Wissawa Samanwijit	ID. 44015713
Advisor	Mr. Dolachai Sookcharoenphol	
	Mr. Phaisan Sitthiyophassakol	
Graduate Level	Bachelor Degree of Information Engineering	
Department	Information Engineering	
Academic Year	2003	

Abstract

This project apply microcontroller, sensor and computer for more efficiency in parking car. The purpose is to be able to tell the amount of the vacant area for car in each floor. For this project, we will use sensor to verify parking of the car and process by using microcontroller MCS-51 and then send the data to computer for indicating figure and picture in the monitor by using Visual Basic program in writing Application

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้ สำเร็จสมบูรณ์ได้ด้วยความอนุเคราะห์จากบุคคลหลายๆท่าน โดยเฉพาะท่านอาจารย์ ดลชัย สุขเจริญผล ซึ่งเป็นอาจารย์ที่ปรึกษา เป็นผู้ให้คำแนะนำข้อมูลและชี้แนะแนวทางในการดำเนินงาน พร้อมทั้งขอขอบ คุณเจตน์ ออสวัสดิ์ ที่ให้คำแนะนำในทุกๆด้าน รวมทั้งขอขอบคุณอาจารย์ทุกท่านที่ไม่ได้กล่าวถึงในที่นี้ เพื่อนๆทุกคนที่เป็นกำลังใจ ท้ายสุดขอขอบพระคุณ คุณพ่อ คุณแม่ ที่เป็นกำลังใจที่สำคัญในการทำงานนี้ให้สำเร็จลุล่วง

หวังเป็นอย่างยิ่งว่าปริญญานิพนธ์ฉบับนี้จะก่อประโยชน์ และเป็นแนวทางในการดำเนินงานสำหรับรุ่นน้องและผู้สนใจต่อไป

คณะผู้จัดทำ

สารบัญ

สารบัญ (ต่อ)

สารบัญรูป

หน้า

สารบัญรูป (ต่อ)

หน้า

สารบัญตาราง

หน้า

บทที่ 1
บทนำ

1.1 แนวคิดและที่มาของปัญหา

ในปัจจุบันบริเวณที่จอดรถของห้างสรรพสินค้า หรือบริษัทต่างๆ ที่อยู่ในอาคารสูงหลายชั้นที่มีการให้บริการที่จอดรถแก่ลูกค้านั้น ลูกค้ามักพบปัญหาในการหาที่ว่างสำหรับจอดรถ ทำให้ต้องเสียเวลาอย่างมากในการหาที่ว่างในการเลือกจอดรถยนต์ในแต่ละชั้น ซึ่งก่อให้เกิดปัญหาหลายๆอย่างตามมา อาทิเช่น การจัดการจราจรของรถขาเข้าและขาออกทำได้ยาก, การจราจรติดขัด, ยังเป็นการสร้างมลภาวะทางอากาศให้กับบริเวณใกล้เคียง และยังเป็นการใช้เชื้อเพลิงอย่างไม่มีประสิทธิภาพอีกด้วย ดังนั้นการหาวิธีที่จะช่วยลดระยะเวลาในการหาที่ว่าง เพื่อเป็นการอำนวยความสะดวกในการเข้ามาจอดรถ โดยลูกค้าสามารถทราบได้แน่นอนว่าบริเวณที่จอดรถในแต่ละชั้นมีจำนวนเหลือเท่าใด ตำแหน่งใดว่างบ้าง

โครงงานนี้มีชื่อว่า ระบบจัดการที่จอดรถอัตโนมัติโดย เป็นการนำไมโครคอนโทรลเลอร์, อุปกรณ์ตรวจจับ และคอมพิวเตอร์ มาประยุกต์ใช้ร่วมกันเพื่อจัดการระบบที่จอดรถให้มีประสิทธิภาพ โดยมีวัตถุประสงค์คือ สามารถบอกจำนวนที่ว่างสำหรับจอดรถที่แน่นอนในแต่ละชั้นว่ามีจำนวนเหลือเท่าใด ตำแหน่งใดว่างลงบ้าง โดยโครงงานนี้จะใช้เซ็นเซอร์เป็นอุปกรณ์ในการตรวจจับการเข้าจอดของรถแล้วทำการประมวลผลโดยใช้ไมโครคอนโทรลเลอร์ MCS-51 จากนั้นจะส่งข้อมูลต่อไปยังคอมพิวเตอร์เพื่อแสดงผลเป็นตัวเลขและรูปภาพทางหน้าจอคอมพิวเตอร์ และทำการจัดเก็บข้อมูลเพื่อใช้ในการตรวจสอบภายหลัง โดยใช้โปรแกรม Visual Basic ในการเขียน Application โปรแกรมแสดงผล

1.2 วัตถุประสงค์

1.2.1 เพื่อจัดการระบบให้บริการที่จอดรถโดยผ่านการควบคุมจากไมโครคอนโทรลเลอร์

1.2.2 เพื่อใช้ระบุจำนวนและตำแหน่งที่ว่างในการจอดรถที่แน่นอน โดยแสดงผลทางจอคอมพิวเตอร์

1.2.3 เพื่อเพิ่มเติมความรู้แก่ผู้จัดทำในด้านการประยุกต์ใช้งานไมโครคอนโทรลเลอร์ร่วมกับคอมพิวเตอร์

1.2.4 เพื่อให้สามารถนำโครงงานนี้ไปประยุกต์ใช้กับระบบที่มีการคิดค่าบริการที่จอดรถและสามารถรองรับการใช้งานที่เพิ่มขึ้นในอนาคตได้

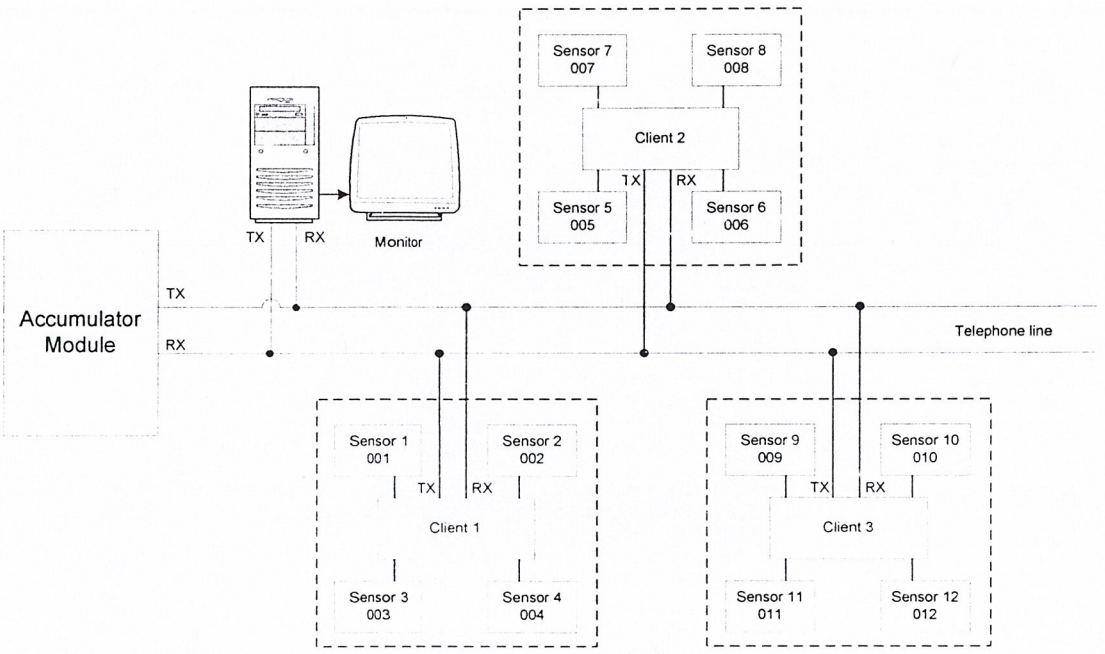
1.3 ขอบเขตของโครงงาน

1.3.1 สร้างวงจรเซ็นเซอร์ในการตรวจจับการเข้าจอดของรถแล้วส่งข้อมูลไปประมวลผลโดยใช้ไมโครคอนโทรลเลอร์ MCS-51

1.3.2 สามารถระบุจำนวนที่จอดรถที่ว่างทั้งหมดได้ และสามารถระบุตำแหน่งของที่จอดรถที่ว่างได้ โดยแสดงผลเป็นตัวเลขและรูปภาพทางหน้าจอคอมพิวเตอร์ โดยใช้โปรแกรม Visual Basic ในการเขียน Application โปรแกรมแสดงผล

1.3.3 สามารถนำไปใช้งานในบริเวณพื้นที่กว้างได้โดยการออกแบบในลักษณะเมตริกซ์

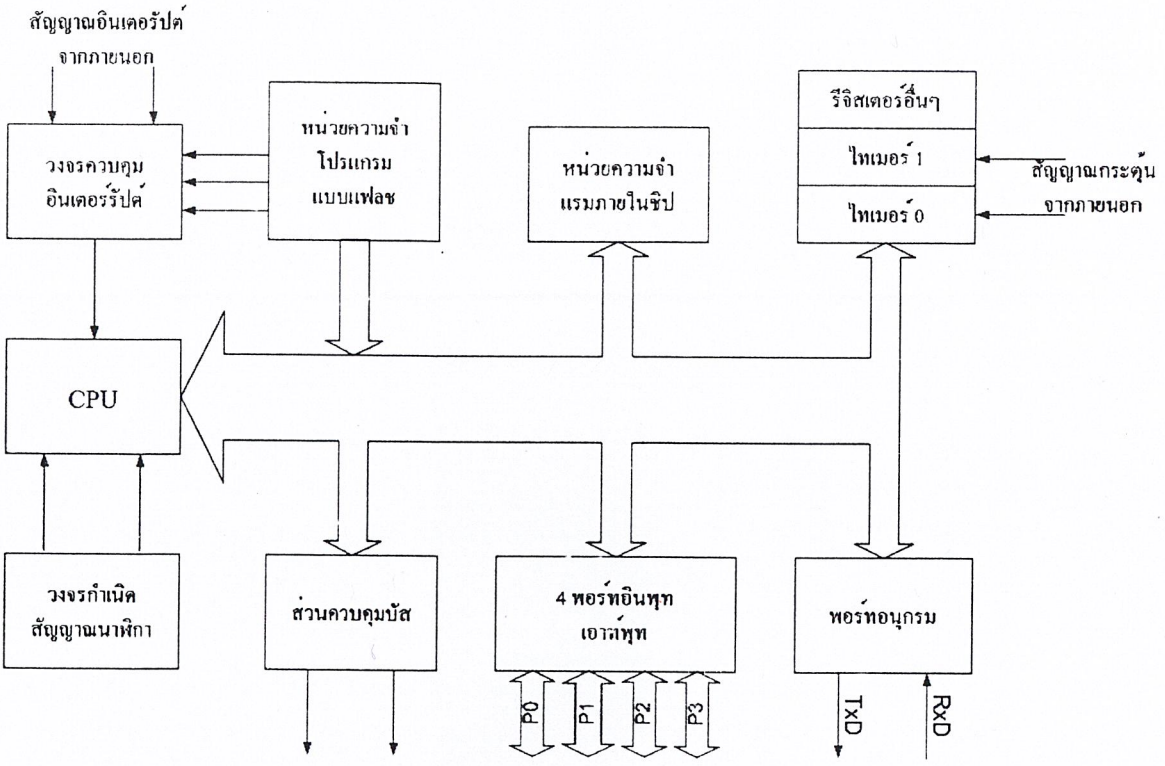
1.4 สถาปัตยกรรมของระบบ



รูปที่ 1.1 แสดงสถาปัตยกรรมของระบบ

บทที่ 2
ทฤษฎีที่เกี่ยวข้อง

2.1 โครงสร้างและสถาปัตยกรรมของไมโครคอนโทรลเลอร์ MCS-51 แบบ Flash



รูปที่ 2.1 โครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์

2.1.1 คุณสมบัติของไมโครคอนโทรลเลอร์ MCS-51

- เป็นไมโครคอนโทรลเลอร์ ที่ใช้ซีพียูขนาด 8 บิต
- ภายในมีหน่วยความจำโปรแกรมเป็นแบบแฟลชสามารถลบและเขียนใหม่ได้นับพันครั้ง
- หน่วยความจำข้อมูลพื้นฐานเป็นหน่วยความจำแบบแรม
- ขาพอร์ตเป็นแบบสองทิศทาง สามารถเป็นได้ทั้งอินพุตและเอาท์พุต
- มีวงจรสื่อสารอนุกรมแบบฟูลดูเพล็กซ์
- ไทเมอร์/เคาน์เตอร์ขนาด 16 บิตอย่างน้อย 2 ตัว
- สามารถรองรับแหล่งกำเนิดอินเตอร์รัปต์ได้ 6 ประเภท

- สามารถขยายหน่วยความจำภายนอกเพิ่มเติมได้สูงสุด 64 กิโลไบต์
- มีวงจรกำเนิดสัญญาณนาฬิกาอยู่ภายในชิป
- มีวงจรสื่อสารอนุกรมแบบ SPI สำหรับในอนุกรม AT89Sxx

2.1.2 การจัดขาของไมโครคอนโทรลเลอร์ MCS-51

- ขา VCC ใช้สำหรับต่อไฟเลี้ยง +5V
- ขา GND เป็นขากราวด์ สำหรับต่อกราวด์ของระบบ
- ขาพอร์ต 0 (P0.0-P0.7) ขาที่ 32-39 แต่ละขาสามารถกำหนดได้เป็นทั้งอินพุตและเอาท์พุต สำหรับใช้งานทั่วไป ถ้าต้องการกำหนดขาพอร์ต 0 ขาใดขาหนึ่งเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์ต่ำของหน่วยความจำภายนอก (A0-A7) และขาข้อมูล (D0-D7) โดยใช้กระบวนการมัลติเพล็กซ์เข้าช่วย เพื่อสลับการทำงานเป็นไปได้ทั้งขาติดต่อแอดเดรสและขาข้อมูล
- ขาพอร์ต 1 (P1.0-P1.7) ขาที่ 1-8 แต่ละขาสามารถกำหนดได้เป็นทั้งอินพุตและเอาท์พุต สำหรับใช้งานทั่วไป ถ้าต้องการกำหนดขาพอร์ตขาใดขาหนึ่งเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นในอนุกรม AT89Sxx จะใช้ขา P1.0 เป็นขาอินพุตสำหรับนับค่าของไทเมอร์ 2 และ P1.1 เป็นขาอินพุตทริกเกอร์ของไทเมอร์ 2 ในขณะที่ขา P1.4 ถึง P1.7 เป็นขาสำหรับเชื่อมต่อแบบ SPI เพื่อทำการโปรแกรมข้อมูลในระบบ
- ขาพอร์ต 2 (P2.0-P2.7) ขาที่ 21-28 แต่ละขาสามารถกำหนดได้เป็นทั้งอินพุตและเอาท์พุต สำหรับใช้งานทั่วไป ถ้าต้องการกำหนดขาพอร์ตขาใดขาหนึ่งเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังถูกใช้งานในการติดต่อกับขาแอดเดรสไบต์สูงของหน่วยความจำภายนอก (A0-A7)
- ขาพอร์ต 3 (P3.0-P3.7) ขาที่ 10-17 แต่ละขาสามารถกำหนดได้เป็นทั้งอินพุตและเอาท์พุต สำหรับใช้งานทั่วไป ถ้าต้องการกำหนดขาพอร์ตขาใดขาหนึ่งเป็นอินพุตสามารถทำได้โดยการเขียนข้อมูล "1" ไปยังแต่ละบิตของพอร์ตที่ต้องการติดต่อด้วย ส่งผลให้ขาพอร์ตนั้นมีสถานะปล่อยลอย (float) จึงมีอินพุตอิมพีแดนซ์สูง สามารถใช้งานเป็นขาพอร์ตอินพุตได้ นอกจากนั้นขาพอร์ตนี้ยังมีหน้าที่การใช้งานพิเศษ ดังนี้

P3.0 ใช้เป็นขาอินพุตสำหรับข้อมูลจากการสื่อสารแบบอนุกรม หรือขา RxD

P3.1 ใช้เป็นขาเอาท์พุตสำหรับข้อมูลจากการสื่อสารแบบอนุกรม หรือขา TxD

P3.2 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอกช่อง 0 หรือขา $\overline{INT0}$

P3.3 ใช้เป็นขาอินพุตรับสัญญาณอินเตอร์รัปต์จากภายนอกช่อง 1 หรือขา $\overline{INT1}$

P3.4 ใช้เป็นขาอินพุตสำหรับรับสัญญาณไทเมอร์จากภายนอกช่อง 0 หรือขา T0

P3.5 ใช้เป็นขาอินพุตสำหรับรับสัญญาณไทเมอร์จากภายนอกช่อง 1 หรือขา T1

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

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

- ขารีเซต (Reset) ขาที่ 9 ใช้ในการรีเซตการทำงานของไมโครคอนโทรลเลอร์โดยการป้อนสัญญาณเพื่อรีเซตสถานะที่ขานี้ต้องอยู่ในระดับรีเซตอย่างน้อย 2 แมชชีนไซเกิลโดยที่วงจรกำหนดสัญญาณนาฬิกายังคงทำงานต่อเนื่องไปอย่างปกติ

- ขา $\overline{ALE}$ / $PROG$ (Address Latch Enable/Program pulse input) ขาที่ 30 เป็นขาที่ใช้ในการควบคุมการแลตช์ของขาพอร์ต 0 เมื่อมีการใช้งานหน่วยความจำภายนอก นอกจากนั้นขานี้ยังใช้เป็นขาสำหรับรับพัลซ์ของการโปรแกรมสำหรับโปรแกรมข้อมูลลงในไมโครคอนโทรลเลอร์ MCS-51 ในรุ่นใหม่ที่มีหน่วยความจำเป็นแบบอีพรอม

- ขา $\overline{PSEN}$ (Program Store Enable) ขาที่ 29 เป็นขาที่ใช้ในการส่งสัญญาณเพื่อร้องขอติดต่อกับหน่วยความจำโปรแกรมภายนอก เมื่อไมโครคอนโทรลเลอร์ต้องการอ่านข้อมูลจากหน่วยความจำโปรแกรมภายนอกตัวไมโครคอนโทรลเลอร์ ตัวไมโครคอนโทรลเลอร์จะส่งสัญญาณออกมาที่ขานี้ 2 ครั้งในแต่ละแมชชีนไซเกิล แต่ถ้าหากติดต่อกับหน่วยความจำข้อมูลภายนอก ขานี้จะไม่มีการส่งสัญญาณใดๆออกมา

- ขา $\overline{EA}$ / V_{pp} (External Access enable/Programming voltage Input) ใช้สำหรับเลือกการติดต่อหน่วยความจำจากภายนอกหรือภายในตัวไมโครคอนโทรลเลอร์ ถ้าหากขานี้เป็น “0” เป็นการเลือกให้ไมโครคอนโทรลเลอร์ติดต่อกับหน่วยความจำโปรแกรมภายนอก แต่ถ้าหากขานี้เป็น “1” เป็นการเลือกให้ไมโครคอนโทรลเลอร์ ติดต่อกับหน่วยความจำภายในไมโครคอนโทรลเลอร์ นอกจากนี้ที่ขานี้ยังใช้เป็นขาอินพุตสำหรับรับแรงดันไฟสูงสำหรับการโปรแกรมหน่วยความจำภายในไมโครคอนโทรลเลอร์ สำหรับไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลช ต้องการแรงดันสำหรับการโปรแกรม +12V

- ขา XTAL 1 และ XTAL 2 เป็นขาสำหรับต่อคริสตอลเพื่อสร้างสัญญาณนาฬิกาในการกำหนดจังหวะการทำงานของไมโครคอนโทรลเลอร์

2.1.3 การจัดการหน่วยความจำของไมโครคอนโทรลเลอร์ MCS-51 แบบแฟลช

ไมโครคอนโทรเลอร์ MCS-51 แบบแฟลชในเบอร์ต่างๆที่นิยมใช้งานอันประกอบด้วยเบอร์ AT89C51 และ AT89C52 จะเห็นได้ว่าทั้งสองเบอร์สามารถติดต่อหน่วยความจำโปรแกรมได้สูงสุด 64 กิโลไบต์ โดยสามารถเลือกใช้หน่วยความจำโปรแกรมภายในอย่างเดียวหรือรวมกับหน่วยความจำโปรแกรมภายนอกอย่างเดียวก็ได้ โดยภายใน AT89C51 ที่เลือกใช้มีหน่วยความจำภายใน 4 กิโลไบต์

ในกรณีที่ใช้หน่วยความจำภายในและภายนอกรวมกัน หากใช้ AT89C51 ก็สามารถติดต่อกับหน่วยความจำภายนอกได้ 60 กิโลไบต์ และถ้าใช้เบอร์ AT89C52 จะสามารถติดต่อกับหน่วยความจำโปรแกรมภายนอกได้ 56 กิโลไบต์

หน่วยความจำโปรแกรม ใช้เก็บข้อมูลของโปรแกรมควบคุมการทำงานของไมโครคอนโทรลเลอร์ หรือที่เรียกว่า โปรแกรมมอนิเตอร์ (Monitor Program) หากใช้หน่วยความจำภายนอกมักจะบรรจุอยู่ในหน่วยความจำชนิดอีพรอม (EPROM : Erasable Programmable Read-Only Memory)

หน่วยความจำโปรแกรมมีแอดเดรสเริ่มต้นที่ 0000H เมื่อซีพียูได้รับการรีเซตให้เริ่มต้นการทำงาน จะต้องมาเริ่มต้นที่แอดเดรส 0000H นี้เสมอ ซึ่งมีการสงวนพื้นที่บางตำแหน่งไว้สำหรับการบริการอินเตอร์รัปต์ 6 ประเภท ประเภทละ 8 ไบต์ ประกอบด้วย

- พื้นที่สำหรับบริการอินเตอร์รัปต์ 0 จากภายนอก กำหนดไว้ที่แอดเดรส 0003H
- พื้นที่สำหรับบริการอินเตอร์รัปต์จากไทเมอร์ 0 กำหนดไว้ที่แอดเดรส 000BH
- พื้นที่สำหรับบริการอินเตอร์รัปต์ 1 จากภายนอก กำหนดไว้ที่แอดเดรส 0013H
- พื้นที่สำหรับบริการอินเตอร์รัปต์จากไทเมอร์ 1 กำหนดไว้ที่แอดเดรส 001BH
- พื้นที่สำหรับบริการอินเตอร์รัปต์ของการสื่อสารอนุกรม กำหนดไว้ที่แอดเดรส 0023H
- พื้นที่สำหรับบริการอินเตอร์รัปต์จากไทเมอร์ 2 กำหนดไว้ที่แอดเดรส 002BH

2.1.4 พอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS-51

ไมโครคอนโทรลเลอร์ตระกูล MCS-51 มีวงจรสื่อสารอนุกรมแบบฟูลดูเพล็กซ์ 1 ชุด (วงจรสื่อสารแบบฟูลดูเพล็กซ์หมายถึงวงจรสื่อสารที่สามารถทำการรับและส่งข้อมูลในลักษณะ 2 ทิศทางได้ในเวลาเดียวกัน)โดยใช้ขาสัญญาณของพอร์ต 3 คือขา P3.0 เป็นขารับข้อมูลเข้าหรือ RxD และขา P3.1 เป็นขาส่งข้อมูลออกหรือ TxD โดยใช้วงจรสื่อสารข้อมูลแบบอนุกรมของไมโครคอนโทรลเลอร์ตระกูลMCS-51 แบบแฟลช เป็นแบบอะซิงโครนัส ปกติแล้วพอร์ตอนุกรมของไมโครคอนโทรลเลอร์ MCS-51 จะใช้ในการติดต่อสื่อสารกับพอร์ตอนุกรมของคอมพิวเตอร์โดยใช้มาตรฐาน RS-232 แต่ในปัจจุบันสามารถติดต่อในมาตรฐาน RS-422 หรือ RS-485 ได้แล้ว

โดยใช้ไอซีพิเศษที่ทำหน้าที่แปลงสัญญาณในการสื่อสารดังกล่าว รีจิสเตอร์ที่เกี่ยวข้องกับการทำงานของพอร์ตอนุกรมใน MCS-51ดังนี้

2.1.4.1 รีจิสเตอร์ควบคุมไทเมอร์

เนื่องจากว่า การใช้พอร์ตอนุกรม นั้นมีสิ่งที่ต้องคำนึงถึงคืออัตราการรับ-ส่งข้อมูล (จังหวะการเลื่อนข้อมูลเข้าออกจาก MCS-51) หรือเรียกว่า อัตราบอด (Baud rate) โดยอัตราบอดนี้สามารถสร้างขึ้นภายในชิพของ MCS-51 ได้จากไทม์เมอร์แชนแนล 1 โดยทำงานในโหมด 2 ซึ่งมีการโหลดค่ากลับอัตโนมัติ ดังนั้นรีจิสเตอร์ที่ต้องการทำการโปรแกรมมีดังนี้

- TMOD ตำแหน่ง 89H ทำหน้าที่เลือกโหมดของไทม์เมอร์
- TCON ตำแหน่ง 88H ทำหน้าที่เริ่มต้นการสร้างอัตราบอด
- TH1 ตำแหน่ง 8CH ทำหน้าที่ใส่ข้อมูลการนับของไทม์เมอร์ 1 เพื่อสร้างอัตราบอด

2.1.4.2 รีจิสเตอร์ควบคุมการลดกำลัง

เนื่องจากว่า การสร้างอัตราบอดนั้นจะต้องนำบิตในรีจิสเตอร์ PCON มาใช้ในการคำนวณข้อมูลของ TH1 ดังนั้นรีจิสเตอร์ที่ใช้ก็คือ PCON ตำแหน่ง 87H ทำหน้าที่ในการคำนวณข้อมูลที่จะใส่ในรีจิสเตอร์ TH1 ดังนี้

7	6	5	4	3	2	1	0
SMOD	-	-	-	GF1	GF0	PD	IDL

บิตที่ 7 (SMOD) คือ บิตที่ใช้สำหรับเปลี่ยนแปลงแก้ไขอัตราการส่งข้อมูลอนุกรม โดยใช้โปรแกรมเซตให้เป็น 1 เป็นการเพิ่มอัตราการส่งขึ้นเท่าตัวเมื่อใช้ไทม์เมอร์ 1 เข้าช่วย ถ้าเลือกใช้การรับส่งข้อมูลในโหมด 1,2 และ 3 แต่ถ้าเคลียร์เป็น 0 จะใช้ไทเมอร์ 1 ตามปกติที่เป็นอยู่ตามการกำหนดเริ่มต้นของโปรแกรม

บิตที่ 6-4 ไม่ใช้งาน

บิตที่ 3 (GF1) คือ แฟลกใช้งานทั่วไป กำหนดโดยผู้ใช้ไม่เกี่ยวข้องกับการควบคุมใดๆ

บิตที่ 2 (GF0) คือ แฟลกใช้งานทั่วไปเช่นกัน

บิตที่ 1 (PD) คือ บิตที่แสดงการลดลงของกำลังไฟ (Power down) เซ็ตเป็น 1 โดยโปรแกรมเพื่อเข้าสู่โหมดการลดของกำลังไฟ ใช้ในไมโครโปรเซสเซอร์ที่เป็นแบบ CHMOS เท่านั้น

บิตที่ 0 (IDL) คือ บิตที่แสดงในโหมดไอเดิล (idle) เซ็ตเป็น 1 โดยโปรแกรมเพื่อเข้าไอเดิลในไมโครโปรเซสเซอร์ที่เป็นแบบ CHMOS เท่านั้น

2.1.4.3 รีจิสะเตอร์ควบคุมการอินเตอร์รัปต์

MCS-51ใช้งานพอร์ตอนุกรมในลักษณะการอินเตอร์รัปต์ได้ จึงมีรีจิสเตอร์ที่เกี่ยวข้องคือ

-IE ตำแหน่ง A8H ทำหน้าที่ยอมให้เกิดการอินเตอร์รัปต์จากพอร์ตอนุกรมได้หรือไม่

-IP ตำแหน่ง B8H ทำหน้าที่จัดลำดับความสำคัญของการอินเตอร์รัปต์

2.1.4.4 รีจิสตอร์ควบคุมพอร์ตอนุกรม

การใช้งานพอร์ตอนุกรมขึ้นอยู่กับรีจิสเตอร์โดยตรงคือ

-SBUF ตำแหน่ง 99H ทำหน้าที่เป็นบัฟเฟอร์การรับหรือส่งข้อมูลของพอร์ตอนุกรม

-SCON ตำแหน่ง 98H ทำหน้าที่ควบคุมและกำหนดโหมดการใช้งานพอร์ตอนุกรมทั้งหมด ซึ่งมีรายละเอียดดังนี้

7	6	5	4	3	2	1	0
SM0	SM1	SM2	REN	TB8	RB8	TI	RI

บิตที่ 7 (SM0) คือ โหมดของพอร์ตอนุกรมบิต 0 ทำการเซ็ตหรือเคลียร์ โดยใช้โปรแกรม

บิตที่ 6 (SM1) คือ โหมดของพอร์ตอนุกรมบิต 1 ทำการเซ็ตหรือเคลียร์ โดยใช้โปรแกรม

SM0	SM1	Mode	Detail
0	0	0	รีจิสเตอร์แบบเลื่อนบิต;อัตราการส่ง = f/12
0	1	1	UART ชนิด 8 บิต;อัตราการส่งเปลี่ยนแปลงได้
1	0	2	UART ชนิด 9 บิต;อัตราการส่ง= f/32 หรือ f/64
1	1	3	UART ชนิด 9 บิต;อัตราการส่งเปลี่ยนแปลงได้

บิตที่ 5 (SM2) ใช้เป็นบิตแสดงการติดต่อแบบไมโครโปรเซสเซอร์หลายตัว (Multi-Processor Communication) เซ็ตและเคลียร์ค่าโดยโปรแกรมควบคุม ในกรณีนี้จะใช้เฉพาะโหมด 2 และ 3 เมื่อบิตนี้ถูกเซ็ตเป็น 1 การอินเทอร์รัพท์จะไม่เกิดขึ้นเมื่อใช้งานในโหมด 1 อินเทอร์รัพท์จะเกิดขึ้นเมื่อได้รับบิตหยุดที่ถูกต้องแล้ว ถ้าโหมด 0 บิตนี้จะถูกเคลียร์เป็น 0

บิตที่ 4 REN เป็นบิตอีนาเบิลการรับ บิตนี้จะถูกเซ็ตเป็น 1 เมื่อต้องการรับสัญญาณอนุกรม และจะถูกเคลียร์เป็น 0 เมื่อไม่ต้องการรับสัญญาณอนุกรม

บิตที่ 3 TB8 ใช้เลือกว่าจะให้ส่งบิต 8 หรือไม่ใช้ สำหรับกรณีรับข้อมูลในโหมด 2 หรือ 3 บิตหยุดในโหมด 1 ส่วนในโหมด 0 จะไม่ใช้งานบิตนี้

บิตที่ 2 RB0 ใช้เลือกว่าจะให้รับบิต 8 หรือไม่ใช้ สำหรับกรณีรับข้อมูลในโหมด 2 หรือ 3 บิตหยุดในโหมด 1 ส่วนในโหมด 0 จะไม่ใช้งานบิตนี้

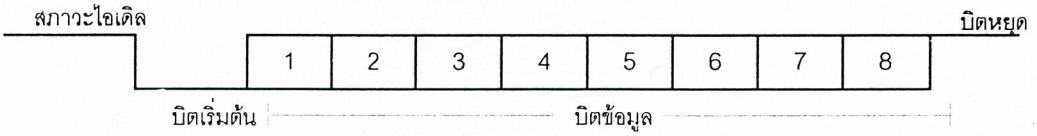
บิตที่ 1 TI แฟลกอินเทอร์รัพท์ เมื่อส่งข้อมูลในโหมด 0 จะถูกเซ็ตเป็น 1 หลังจากส่งบิต 7 ออกไป การเคลียร์บิตนี้จะทำได้โดยใช้โปรแกรม

บิตที่ 0 RI แฟลกอินเทอร์รัพท์ เมื่อรับข้อมูลในโหมด 0 จะถูกเซ็ตเป็น 1 หลังจากรับบิต 7 เข้ามาแล้ว ในโหมดอื่นๆ จะเซ็ตเป็น 1 เมื่อรับบิตหยุดได้ครึ่งหนึ่ง การเคลียร์บิตนี้จะทำได้โดยใช้โปรแกรม

การอ้างอิงแบบแบบบิตแอดเดรสของรีจิสเตอร์นี้คือ SCON.0 – SCON.7

2.1.4.5 การส่งข้อมูลอนุกรมในโหมด 1 (Standard UART)

การส่งข้อมูลในโหมดนี้จะเป็นการส่งข้อมูลแบบอะซิโครนัส 8 บิตข้อมูล 1 บิตเริ่มต้นและ 1 บิตหยุด การทำงานในโหมดนี้จะทำโดยการกำหนดข้อมูลในรีจิสเตอร์ SCON บิต SM0 และบิต SM1 ให้มีค่าเป็น “01” ซึ่งเป็นการกำหนดให้รีจิสเตอร์ SBUF กลายเป็นตัวรับส่งข้อมูลขนาด 10 บิต แบบฟูลดูเพลกซ์ (Full Duplex) ซึ่งสามารถรับ และส่งข้อมูลได้ภายในเวลาเดียวกัน โดยใช้ขา RXD ทำหน้าที่รับสัญญาณอนุกรมที่เข้ามา และขา TXD ทำการส่งข้อมูลแบบอนุกรมออกไปภายนอก



รูปที่ 2.2 รูปแบบการรับส่งข้อมูลในโหมด 1

การส่งข้อมูลจะเริ่มต้นด้วยการส่งบิตเริ่มต้น (Start bit) ออกไปก่อนแล้วตามด้วยบิตข้อมูล (โดยส่ง 0 ออกไปก่อน) จากนั้นจึงเป็นการส่งบิตหยุด (Stop bit) แฟล็ก TI จะเซ็ตเมื่อส่งข้อมูลครบทั้ง 10 บิต

การรับข้อมูลจะเริ่มจากลอจิกในสายสัญญาณเปลี่ยนสภาวะจาก 0 เป็น 1 (ขอบบวกของบิตเริ่มต้น) บิตเริ่มต้นจะถูกข้ามไปไม่สนใจ จะสุ่มเอาข้อมูลอีก 8 บิตที่เหลือเข้ารีจิสเตอร์เลื่อนบิตภายในพอร์ตอนุกรมเมื่อครบทั้ง 8 บิตแล้ว สิ่งต่อไปนี้จะเกิดขึ้น

บิต 9 (บิตหยุด) จะถูกเก็บเข้าไปในบิต RB8 ของ SCON

SBUF จะทำการโหลดข้อมูลทั้ง 8 บิตเข้าตัวเอง

แฟล็ก RI เซ็ต

สิ่งที่กล่าวมาทั้งหมดจะเป็นจริงต่อเมื่อ

RI = 0

SM2 = 1 และบิตหยุดที่รับเข้ามาเป็น 1 หรืออีกกรณีหนึ่งคือ SM2 = 0

2.1.4.5.1 การคำนวณอัตราบอดของโหมด 1

อัตราความเร็วในการรับและส่งข้อมูลของการรับส่งข้อมูลแบบอะซิงโครนัส หรือ อัตราบอดเรต ที่ใช้สำหรับพอร์ตอนุกรม RS-232 มีด้วยกันหลายค่าตั้งแต่ 110 ถึง 19,200 บิตต่อวินาที โดยมีค่าเพิ่มขึ้นตามเทคโนโลยีของคอมพิวเตอร์เนื่องจากอัตราบอดเรตคือค่าของจำนวนบิตที่สามารถส่งได้ใน 1 วินาที

ไทม์เมอร์ 1 จะถูกใช้เป็นตัวสร้างอัตราบอด เมื่อกำหนดการรับส่งข้อมูลแบบอนุกรมในโหมด 1 โดยไทม์เมอร์ 1 จะถูกใช้ให้ทำงานในโหมด 2 (โหลดค่าเข้าไปอัตโนมัติ) การคำนวณจะเป็นดังนี้

$$\text{Fbuad} = \frac{2^{\text{SMOD}} \times \text{Oscillator Frequency}}{32_D \times 12_D \times [256_D - (\text{TH1})]}$$

เมื่อ SMOD เป็นบิตควบคุมอยู่ในรีจิสเตอร์ PCON ซึ่งเป็น 0 หรือ 1 ก็ได้ถ้าเป็น 0 จะเป็นความถี่ปกติถ้าเป็น 1 ความถี่จะเพิ่มขึ้นเป็น 2 เท่าได้ดังที่กล่าวมาแล้วในรีจิสเตอร์ PCON

ถ้าไทม์เมอร์ 1 ไม่ถูกใช้งานในโหมด 2 ค่าอัตราบอดจะเป็น

$$\text{Fbuad} = \frac{2^{\text{SMOD}} \times (\text{timer 1 overflow frequency})}{32_D}$$

2.2 ลักษณะของการสื่อสารตามมาตรฐาน

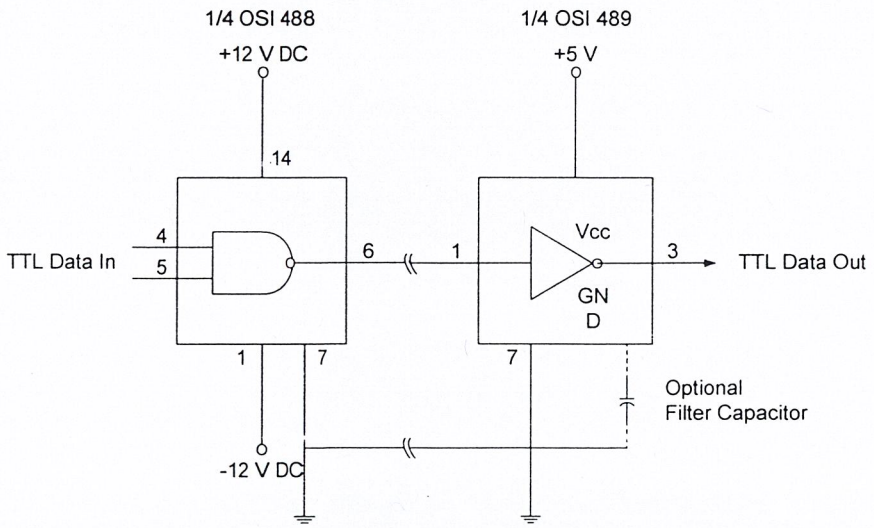
ในปัจจุบันสัญญาณที่ใช้ในการสื่อสารส่วนใหญ่จะใช้สัญญาณดิจิตอล (Digital Signal) เป็นหลักและสัญญาณสื่อสารที่ใช้ส่วนมากจะเป็นสัญญาณดิจิตอลที่มีระดับสัญญาณแบบTTL ซึ่งถ้าทำการติดต่อสื่อสารในลักษณะของระดับสัญญาณแบบ TTL แล้วจะสามารถทำการติดต่อสื่อสารได้โดยตรง แต่ในหลายกรณีที่มีความจำเป็นที่จะต้องสื่อสารด้วยสัญญาณในระดับอื่นที่ไม่ใช่ระดับสัญญาณแบบ TTL หรือไม่ได้เป็นสัญญาณดิจิตอล ซึ่งทำให้เกิดปัญหาในการสื่อสาร แต่ปัญหาดังกล่าวสามารถแก้ไขได้โดยอาศัยการแปลงสัญญาณที่ต้องการสื่อสารให้เป็นสัญญาณตามมาตรฐาน โดยที่แต่ละมาตรฐานจะมีข้อกำหนดที่แตกต่างกันไป ในที่นี้จะกล่าวถึงมาตรฐานสำคัญที่ใช้ในการสื่อสารดังนี้

2.2.1 ลักษณะการเชื่อมต่อตามมาตรฐาน RS-232-C

มาตรฐาน RS-232-C เป็นมาตรฐานที่ได้รับการพัฒนามานานและถูกใช้งานกันอย่างแพร่หลาย และใช้เป็นมาตรฐานในการเชื่อมต่อ DTE(Data Terminal Equipment) เข้ากับ DCE(Data Communication Equipment) เช่น การต่อคอมพิวเตอร์ส่วนบุคคลเข้ากับโมเด็ม

ลักษณะการเชื่อมต่อตามมาตรฐาน RS-232-C นั้นจะใช้สายสัญญาณเพียงคู่เดียวในการส่งสัญญาณ โดยสามารถส่งสัญญาณไปได้ในทิศทางเดียว อัตราเร็วสูงสุดในการส่งข้อมูลมีค่าเท่ากับ 20 kbps (กิโลบิตต่อวินาที) และระยะทางที่ใช้ในการส่งข้อมูลไม่ควรเกิน 50 ฟุต สำหรับการส่งข้อมูลจะใช้ระดับแรงดันแทนค่าทางตรรก (Logic) ของข้อมูลโดยระดับแรงดันที่มีค่าอยู่ระหว่าง –5 โวลต์ ถึง –15โวลต์ จะแทนค่าสถานะ 0 และระดับแรงดันที่มีค่าอยู่ระหว่าง +5โวลต์ ถึง +15 โวลต์ จะแทนค่าสถานะ 1 ส่วนในช่วงระดับแรงดันที่มีค่าอยู่ระหว่าง –5โวลต์ ถึง +5 โวลต์ นั้นจะใช้ในการแบ่งแยกระดับสถานะของสัญญาณระหว่างสถานะ 0 และสถานะ 1

ตัวอย่างของวงจรที่ใช้ในการเชื่อมต่อตามมาตรฐาน RS-232-C จะแสดงให้เห็นได้ดังรูปที่ 2.3 โดยจะมีการแปลงระดับสัญญาณแบบ TTL ไปเป็นระดับแรงดันสัญญาณตามที่กำหนดไว้ในมาตรฐาน RS-232-C แล้วส่งไปตามสายสัญญาณและแปลงกลับจากระดับแรงดันสัญญาณในมาตรฐาน RS-232-C ไปเป็นระดับสัญญาณแบบ TTL

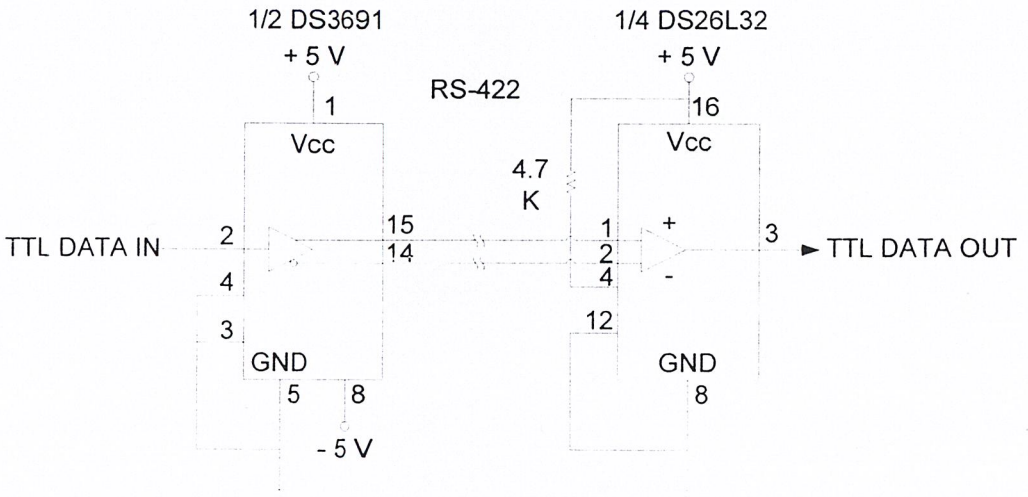


รูปที่ 2.3 แสดงวงจรขับและรับสัญญาณที่ใช้กับมาตรฐาน RS-232-C

2.2.2 ลักษณะการเชื่อมต่อตามมาตรฐาน RS-422

มาตรฐาน RS-422 เป็นมาตรฐานที่ได้รับการพัฒนามาจากมาตรฐาน RS-423 ให้มีประสิทธิภาพในการสื่อสารเพิ่มมากขึ้น โดยมีการพัฒนาให้อัตราเร็วในการส่งข้อมูลมีค่าสูงกว่าในมาตรฐาน RS-423 และระยะทางที่ใช้ในการส่งข้อมูลระหว่างตัวส่งและตัวรับก็มากกว่าในมาตรฐาน RS-423 นอกจากนี้ยังมีความไวต่อสัญญาณมากกว่าในมาตรฐาน RS-423 อีกด้วย

ลักษณะการเชื่อมต่อตามมาตรฐาน RS-422 นั้นจะใช้สายสัญญาณเพียงคู่เดียวในการส่งสัญญาณ โดยสามารถส่งสัญญาณไปได้ทิศทางเดียวในลักษณะของ one-way balanced-line ดังรูปที่ 2.4 อัตราเร็วสูงสุดในการส่งข้อมูลมีค่าเท่ากับ 10 Mbps โดยระยะทางที่ใช้ในการส่งข้อมูลสามารถขยายได้ถึง 4000 ฟุต สำหรับการส่งข้อมูลจะใช้ระดับแรงดันแทนค่าทางตรรก ของข้อมูลโดยระดับแรงดันที่มีค่าอยู่ระหว่าง –2 โวลต์ ถึง –6 โวลต์ จะแทนค่าสถานะ 0 และระดับแรงดันที่มีค่าอยู่ระหว่าง +2 โวลต์ ถึง +6 โวลต์ จะแทนค่าสถานะ 1 ส่วนในช่วงระดับแรงดันที่มีค่าอยู่ระหว่าง–2 โวลต์ ถึง +2 โวลต์ นั้นจะใช้ในการแบ่งแยกระดับสถานะของสัญญาณระหว่างสถานะ 0 และสถานะ 1 นอกจากนี้ตัวรับสัญญาณยังสามารถจับสัญญาณที่มีระดับแรงดันต่ำกว่า 200 mV ได้อีกด้วยทำให้มีความไวต่อสัญญาณเพิ่มมากขึ้น



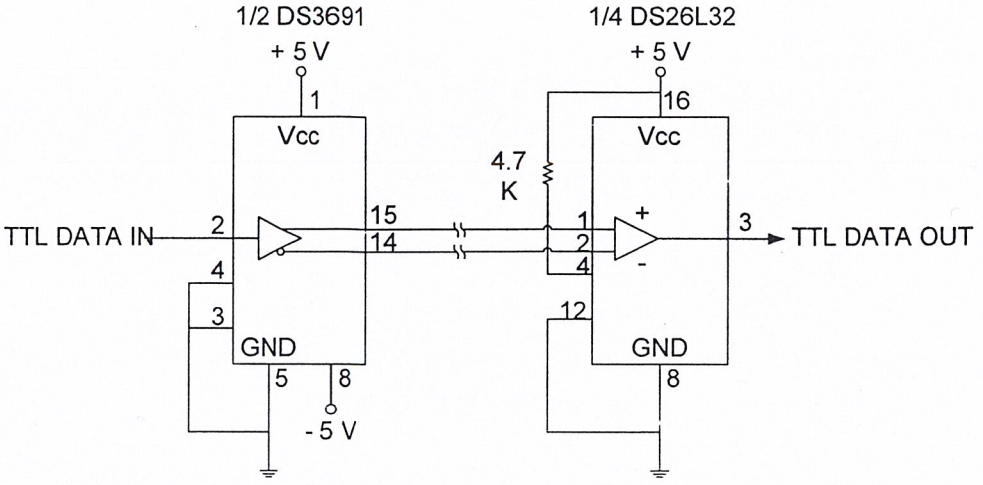
รูปที่ 2.4 แสดงวงจรขับและรับสัญญาณที่ใช้กับมาตรฐาน RS-422

ตัวอย่างของวงจรที่ใช้ในการเชื่อมต่อตามมาตรฐาน RS-422 จะแสดงให้เห็นได้ดังรูปที่ 2.4 โดยจะมีการแปลงระดับสัญญาณแบบ TTL ไปเป็นระดับแรงดันสัญญาณตามที่กำหนดไว้ในมาตรฐาน RS-422 แล้วส่งไปตามสายสัญญาณและแปลงกลับจากระดับแรงดันสัญญาณในมาตรฐาน RS-422 ไปเป็นระดับสัญญาณแบบ TTL

2.2.3 ลักษณะการเชื่อมต่อตามมาตรฐาน RS-485

มาตรฐาน RS-485 เป็นมาตรฐานที่ได้รับการพัฒนามาจากมาตรฐาน RS-422 ให้มีประสิทธิภาพในการสื่อสารเพิ่มขึ้นเป็นอย่างมาก โดยมีการพัฒนาให้วงจรของตัวขับสัญญาณเป็นแบบ 3 สถานะ (Tri State) ทำให้สามารถส่งข้อมูลได้สองทิศทางบนสายคู่เดียวและสามารถต่อเครือข่ายแบบหลายจุด (Multidrop) ซึ่งอุปกรณ์หลายๆตัวสามารถรับและส่งแบบ Half-duplex บนสายสัญญาณคู่เดียวได้

ระบบเครือข่าย RS-485 สามารถติดต่อกันผ่านสาย 2 เส้น (โดยทั่วไปใช้สายคู่ตีเกลียวหรือ twisted pair) สัญญาณข้อมูลที่ส่งจะถูกวัดความแตกต่างของสัญญาณที่ตัวรับ ซึ่งมีข้อดีคือ สามารถส่งข้อมูลในระยะไกลและกำจัดปัญหาสัญญาณรบกวนได้ดี ลักษณะการเชื่อมต่อของ RS-485 สามารถทำได้ 2 ลักษณะคือ 2 สาย และ 4 สาย ในระบบ 2 สาย ตัวส่งและตัวรับของแต่ละอุปกรณ์ต่อเข้ากับสายคู่ตีเกลียว 1 เส้น ระบบ 4 สาย ตัวส่งของพอร์ตแม่ (master port) ต่อเข้ากับตัวรับของตัวลูก(slave port) ทุกตัวในระบบผ่าน สายคู่ตีเกลียวเส้นหนึ่ง และตัวส่งของตัวลูกทุกตัวจะต่อเข้ากับตัวรับของตัวแม่ผ่านสายคู่ตีเกลียวอีกเส้นหนึ่ง การติดต่อกันในระบบอุปกรณ์ทุกตัวจะต้องถูกกำหนดตำแหน่ง(address) ตัวแม่ติดต่อกับตัวลูกทุกตัวโดยตรงแต่ตัวลูกแต่ละตัวจะติดต่อกับตัวแม่ได้เท่านั้นไม่สามารถติดต่อกันเองได้ และในการติดต่อนั้นตัวลูกจะติดต่อกับตัวแม่ได้ทีละตัว เมื่อตัวลูกตัวใดตัวหนึ่งกำลังติดต่อกับตัวแม่อยู่ ตัวลูกที่เหลือจะไม่สามารถติดต่อกับตัวแม่ได้และจะอยู่ในสถานะที่เรียกว่าไฮอิมพีแดนซ์ (High impedance) หรือ 3 สถานะ ซึ่ง Hardware บางผลิตภัณฑ์สามารถจัดการสถานะดังกล่าวได้โดยอัตโนมัติ แต่สำหรับบางกรณีจำเป็นต้องใช้ Software เป็นตัวจัดการ ผลจากการทำงานแบบ tri-state ทำให้ต้องเสียเวลาระยะเวลาหนึ่งระหว่างการสิ้นสุดการส่งข้อมูลและการทำ tri-state ซึ่งช่วงเวลาดังกล่าวเรียกว่า turn-around delay ในช่วงเวลานี้จะไม่มีตัวลูกตัวใดสามารถส่งข้อมูลได้ ซึ่งมีผลกับระบบแบบ 2 สาย แต่ไม่มีผลกับ ระบบแบบ 4 สาย



รูปที่ 2.5 แสดงวงจรขับและรับสัญญาณที่ใช้กับมาตรฐาน RS-485

2.2.3.1 คุณสมบัติในการสื่อสารแบบสองทิศทางคนละเวลา (Half-duplex)

เนื่องจากตัวรับและตัวส่งตามมาตรฐาน RS-485 ถูกออกแบบให้เป็นแบบ 3 สถานะ(Tri-state) จึงสามารถทำการสื่อสารได้สองทิศทางบนสายสัญญาณคู่เดียว (bi-directional) ซึ่งมีข้อดีคือทำให้สะดวกและประหยัดต่อการใช้งาน นอกจากนี้ตามมาตรฐาน RS-485 นั้นยังสามารถที่จะเชื่อมต่อเป็นเครือข่ายได้มากถึง 32 จุด (Unit Load: UL) บนสายสัญญาณคู่เดียว

2.2.3.2 คุณสมบัติทางการเชื่อมต่อเป็นเครือข่าย

ตามมาตรฐาน RS-485 นั้นเครือข่ายสามารถเชื่อมต่อได้หลายจุดบนสายสัญญาณเพียงคู่เดียว (Multiple Transceivers) และการเชื่อมต่อตามมาตรฐาน RS-485 นั้นมีพื้นฐานอยู่บนการเชื่อมต่อแบบบัส (Bus-Type Network) ซึ่งสามารถเชื่อมต่อได้หลายแบบโดยอาศัยการแปลงให้เป็นเครือข่ายเสมือนเพื่อให้เกิดประโยชน์สูงสุดในการทำงาน

2.2.3.3 คุณสมบัติทางด้านอัตราเร็วและระยะทางในการส่งข้อมูล

ในการส่งข้อมูลตามมาตรฐาน RS-485 นั้นสามารถที่จะส่งข้อมูลได้สูงสุดถึง 10 Mbps และส่งข้อมูลได้ไกลที่สุดถึง 4000 ฟุต(1200 เมตร)

2.2.3.4 คุณสมบัติทางด้านสัญญาณรบกวน

สัญญาณรบกวนจะมีผลต่อการสื่อสารตามมาตรฐาน RS-485 น้อยมาก ถ้าเลือกอัตราเร็วและระยะทางในการส่งข้อมูลให้เหมาะสม เนื่องจากตามมาตรฐาน RS-485 นั้นการสื่อสารจะเป็นแบบ Current Loop และใช้ความต่างศักย์ของคู่สายสัญญาณในการส่งข้อมูล ทำให้สามารถทนต่อสัญญาณรบกวนได้ดีโดยเฉพาะสัญญาณรบกวนในลักษณะของ Common-mode noise

2.3 โปรโตคอลการเชื่อมโยงข้อมูล (Data link Protocol)

หน้าที่ทั่วไปของโปรโตคอลนั้นนอกจากจะสร้างกฎให้สถานีตอบสนองต่อการเชื่อมข้อมูลและอักขระอื่นๆ แล้ว ยังมีหน้าที่กำหนดสถานีทุติยภูมิ(สถานีลูก)ที่สถานีปฐมภูมิ(สถานีแม่) หรือสถานีปฐมภูมิเป็นผู้ควบคุมการสื่อสารเมื่อต้องการจะสื่อสารด้วย โดยในระบบการเชื่อมต่อหลายสถานีแบบ Multipoint System ซึ่งสถานีปฐมภูมิ 1 ตัวจะต่อกับสถานีทุติยภูมิหลายตัว สถานีปฐมภูมิสามารถเชื่อมการสื่อสารได้สองวิธี วิธีแรกนั้นสถานีปฐมภูมิสามารถส่งคำถามหรือโพล (Poll) ไปยังสถานีทุติยภูมิตัวใดตัวหนึ่งว่ามีข้อความจะส่งให้สถานีปฐมภูมิหรือไม่ สำหรับวิธีที่สองสถานีปฐมภูมิสามารถส่งการเลือก หรือ Selection ให้สถานีทุติยภูมิตัวใดตัวหนึ่งแจ้งว่าพร้อมที่จะรับข้อความหรือ Traffic หรือไม่ สถานีทุติยภูมิสามารถตอบสนองในโหมดใดโหมดหนึ่งคือ ส่ง (Send), รับ(Receive) หรือ Local โดยแจ้งโหมดดังกล่าวตอบต่อโพล หรือซีเลคชั่นของสถานีทุติยภูมิ การตอบสนองต่อโพลส่วนใหญ่มีดังนี้

- สถานีทุติยภูมิอยู่ในโหมดส่ง และส่งข่าวสารออกไป

- สถานีทุติยภูมิตอบรับ (Acknowledge หรือ ACK) ต่อโพลแต่อยู่ในโหมดรับโดยจะแจ้งสถานีปฐมภูมิว่าไม่มีข่าวสารจะส่ง แต่ตัวเองพร้อมรับกระแสข้อมูล

- สถานีทุติยภูมิปิดสาย (Off-Line) ซึ่งอยู่ในโหมดโลคัล โดยจะแจ้งไปยังสถานีปฐมภูมิว่าอยู่ในโหมดนี้

การตอบสนองต่อซีเลคชั่นส่วนใหญ่จะคล้ายกัน มีแตกต่างเล็กน้อยดังนี้

- สถานีทุติยภูมิอยู่ในโหมดรับจะแจ้งไปว่าพร้อมที่จะรับกระแสข้อมูลจากสถานีปฐมภูมิ

- สถานีทุติยภูมิอยู่ในโหมดส่ง และไม่พร้อมที่จะส่งข้อมูล

- สถานีทุติยภูมิอยู่ในโหมดโลคัล และไม่พร้อมที่จะส่งหรือรับกระแสข้อมูล

แต่ละโปรโตคอลจะรวมถึงการจัดการเกี่ยวกับโพลและซีเลคชั่นและการตอบสนองของสถานีทุติยภูมิ นอกจากนี้ยังกำหนดให้สถานีทุติยภูมิว่าตัวใดควรจะได้โพลหรือซีเลคชั่น สถานีทุติยภูมิสามารถกำหนดโพลให้กับสถานีทุติยภูมิรับครั้งละสถานีเดียวเท่านั้น

การกำหนดซีเลคชั่นมี 3 แบบ คือ

- สถานีทุติยภูมิแต่ละสถานีมีแอดเดรสในการซีเลคชั่นแอดเดรสโดยเฉพาะ (Unique Selection Address) เพื่อที่จะให้สถานีปฐมภูมิเลือกเฉพาะสถานีใดสถานีหนึ่งเท่านั้น

- การให้แอดเดรสเฉพาะกลุ่ม (Group Address) คือ การกำหนดเฉพาะสถานีทุติยภูมิกลุ่มหนึ่งให้สามารถถูกเลือกที่จะรับข่าวสารได้พร้อมกันโดยสถานีอื่นไม่สามารถรับได้

- การให้แอดเดรสทั่วไป (Broadcast Address) โดยทุกๆสถานีทุติยภูมิจะถูกเลือกเมื่อให้แอดเดรสแบบนี้ ดังนั้นสถานีปฐมภูมิสามารถส่งข่าวสารเดียวกันไปยังสถานีทุติยภูมิทุกๆสถานีได้

นอกจากจะสร้างรูปแบบในการโพลและซีเลคชั่น และกำหนดแอดเดรสให้สถานีทุติยภูมิแล้วโปรโตคอลยังสามารถกำหนดรูปแบบข้อความ และการสนองตอบต่ออักขระการเชื่อมข้อมูล ซึ่งอยู่ในรูปแบบรหัสแอสกี้ประกอบด้วยอักขระ 3 ชนิด คือตัวอักษร กราฟฟิค และ การเชื่อมข้อมูลอักขระ อักขระสองชนิดแรกนั้นจะเกี่ยวข้องกับการแสดงผลทางจอภาพหรือการพิมพ์ ส่วนอักขระการเชื่อมข้อมูลจะถูกใช้ในการสร้างซอฟท์แวร์การเชื่อมการสื่อสาร ได้แก่ฟังก์ชั่นต่างๆ เช่น การเริ่มต้นข้อความ การสิ้นสุดข้อความ อักขระเหล่านี้จะได้รับการสนองตอบต่างกันขึ้นอยู่กับแต่ละโปรโตคอล แต่ส่วนใหญ่แล้วจะทำหน้าที่พื้นฐานที่จำเป็นคล้ายๆกัน

ในการเชื่อมโยงเพื่อส่งข้อมูลระหว่างคอมพิวเตอร์กับคอมพิวเตอร์หรืออุปกรณ์ปลายทางรับส่งข้อมูล (Data Terminal Equipment : DTE) กับอุปกรณ์ปลายทางรับส่งข้อมูล หลังจากที่อุปกรณ์ปลายทางรับส่งข้อมูลเชื่อมต่อทางกันทางกายภาพโดยสามารถส่งสัญญาณทางไฟฟ้าถึงกันได้แล้ว ก่อนที่อุปกรณ์ปลายทางรับส่งข้อมูลจะส่งข้อมูลผ่านการเชื่อมโยงทางกายภาพ อุปกรณ์ปลายทางที่ส่งข้อมูลต้องบอกอุปกรณ์ปลายทางที่รับข้อมูลว่าต้องการส่งข้อมูลหรือต้องการติดต่อด้วย โดยการเริ่มต้นสร้างการเชื่อมโยง และในการส่งข้อมูลถ้าหากข้อมูลมีจำนวนมาก ภาคส่งจะต้องทำการแบ่งข้อมูลออกเป็นเฟรม ในการส่งเฟรมผ่านช่องสัญญาณถ้าหากมีการผิดพลาดเกิดขึ้นจะต้องมีการแก้ไขโดยการส่งซ้ำกลับไปใหม่ สุดท้ายเมื่อหมดเฟรมข้อมูลในการติดต่อก็จะต้องมีการบอกยกเลิกการติดต่อ หน้าที่ของโปรโตคอลเชื่อมโยงข้อมูลมีดังนี้

- การควบคุมเฟรม (Frame control) เป็นการกำหนดการเริ่มต้นและสิ้นสุดของเฟรม ในทางอุดมคติเฟรมจะต้องส่งอักษรหรือรูปแบบของบิตใดๆก็ได้

- การควบคุมการผิด (Error control) ในการเชื่อมโยงข้อมูล การรับข้อมูลต้องปราศจากการผิดของบิต ดังนั้นหน้าที่สำคัญของโปรโตคอลเชื่อมโยงข้อมูลคือต้องมีความสามารถในการตรวจจับการผิดที่เกิด ขึ้นในเฟรม และถ้าหากในการรับเฟรมมีการผิดของบิตเกิดขึ้น ภาคส่งจะต้องมีการส่งเฟรมเดิมซ้ำกลับไปใหม่

- การควบคุมการไหลของข้อมูล (Flow control) เป็นการควบคุมให้อุปกรณ์ปลายทางที่รับข้อมูลสามารถรับข้อมูลได้ทันโดยที่อุปกรณ์ปลายทางที่ส่งข้อมูลต้องไม่ส่งเฟรมข้อมูลในอัตราที่เร็วกว่าอุปกรณ์ปลายทางรับที่ข้อมูลจะสามารถรับได้

- การจัดการเชื่อมโยง (Link establish) ในการเชื่อมโยงต้องมีการสร้างการเชื่อมโยง การยกเลิกการเชื่อมโยงเชิงตรรก โดยเฉพาะการเชื่อมโยงของหลายสถานีที่ใช้การเชื่อมต่อทางกายภาพร่วมกัน

2.3.1การควบคุมเฟรม

การควบคุมเฟรมแบ่งออกได้เป็น 2 รูปแบบใหญ่ๆ คือ การควบคุมเฟรมของโปรโตคอลแบบอิงบิต (bit oriented) และการควบคุมเฟรมของโปรตคอลแบบอิงอักษร (Character oriented)

2.3.1.1 โปรโตคอลแบบอิงบิต

- การแทรกบิต (Bit stuff) วิธีนี้แสดงการเริ่มต้นและการหยุดเฟรมด้วยซีเควนของบิตพิเศษจำนวนหนึ่ง เรียกว่าเฟลก(Flag) โดยเฟลกจะประกอบด้วยบิต 01111110 ดังนั้นเมื่อภาครับพบบิต 1 ติดกัน 6 ตัวจะทราบทันทีว่าเป็นการเริ่มต้น หรือหยุดเฟรม และเพื่อให้ส่วนของข้อมูลสามารถส่งไบนารี 1 ติดกันได้เกิน 6 ตัว เมื่อภาคส่งพบบิต 1 ติดกัน 5 ตัวขึ้นไปภาคส่งจะทำการแทรกบิต 0 หลังบิต 1 ที่ติดกัน 5 ตัวทันที ที่ภาครับหลังจากตรวจสอบเฟลกนำแล้วเมื่อพบบิต 0 ที่ตามหลังบิต 1 ที่ติดกัน 5 ตัวภาครับจะตัดบิต 0 ออก

- การใช้ฟิลด์ (Field) บอกความยาวข้อมูล วิธีนี้เฟรมจะถูกแบ่งออกเป็นฟิลด์หรือเขตต่างๆ เริ่มต้นจากปรีแอมเบิล (Preamble) ที่ใช้สำหรับให้สัญญาณนาฬิกาของภาครับซิงโครไนซ์กับภาคส่ง และตามด้วยหัวเฟรมและฟิลด์บอกความยาวของเฟรม ซึ่งภาครับจะทราบขอบเขตของเฟรมโดยอ่านจากฟิลด์บอกความยาวของเฟรม

- การใช้บิตพิเศษ วิธีนี้ใช้การเข้ารหัสของบิตที่ระดับกายภาพ โดยใช้การเข้ารหัสสัญญาณเบสแบนด์เป็นแบบไบเฟสและบอกการเริ่มต้น และสิ้นสุดของเฟรมได้ด้วยการใช้บิตพิเศษที่ไม่มีการเปลี่ยนแปลงที่กลางบิต

2.3.1.2 โปรโตคอลแบบอิงอักษร

โปรโตคอลแบบนี้ใช้กับการส่งข้อมูลที่เป็นอักษรเช่นการส่งไฟล์อักษร (Text file) โดยอักษรในไฟล์จะแทนด้วยรหัส ASCII ซึ่งในรหัสASCII จะมีกลุ่มอักษรที่ใช้ควบคุม การสื่อสาร (Control character) หรือรูปแบบการพิมพ์

อักษร	รหัส	ความหมาย
SOH	01H	Start of Head
STX	02H	Start of Text
ETX	03H	End of Text
EOT	04H	End of Transmission
ENQ	05H	Enquiry
ACK	06H	Acknowledge
DLE	10H	Data Link Escape

อักษร	รหัส	ความหมาย
NAK	15H	Negative Acknowledge
SYN	16H	Synchronous Idle
ETB	17H	End of Text Block

ตารางที่ 2.1 แสดงกลุ่มอักษรที่ใช้ในการควบคุม การสื่อสาร (Control character)

SYN	STX	Data	ETX	Error check

รูปที่ 2.6 แสดงการควบคุมเฟรมโดยใช้อักษรควบคุม

เมื่ออุปกรณ์ปลายทางที่รับข้อมูลรับข้อมูลอักษรเหล่านี้แล้ว จะแปลความหมายของรหัสเพื่อกระทำตามหน้าที่นั้นๆ โดยเฟรมการส่งแบบอิงอักษรจะเริ่มต้นด้วยอักษรซิงโครไนซ์ (SYN) เพื่อให้ทราบว่ามีการเริ่มต้นเฟรม เมื่อภาครับพบอักษร STX หมายถึงการเริ่มต้นของอักษร ภาครับก็จะทราบทันทีว่าอักษรต่อไปจะเป็นข้อมูลและเมื่อพบอักษร ETX แล้วภาครับก็จะทราบได้ว่าเป็นการสิ้นสุดข้อมูล

2.3.2 การควบคุมการผิด

การควบคุมการผิดของการส่งข้อมูลประกอบด้วยการตรวจจับการแก้ไขการผิดและการส่งเฟรมที่ผิดใหม่ (Retransmission) สำหรับรูปแบบของการผิดในการส่งข้อมูลมี2รูปแบบ คือ เฟรมผิดพลาด (Error) เป็นการผิดของบิตในเฟรมเนื่องจากสัญญาณรบกวน และ เฟรมหาย (Loss) ซึ่งเกิดจากการส่งเฟรมผ่านช่องสัญญาณที่มีการจางหาย (Fading) ทำให้เฟรมที่ส่งอาจเกิดการสูญหายก่อนที่จะถึงภาครับ สำหรับวิธีการควบคุมการผิดของการส่งเฟรมกล่าวได้โดยลำดับคือ

- ขบวนการตรวจสอบการผิดของบิต (Error detection) ในการส่งเฟรมในกรณีที่มีการแก้ไขการผิดล่วงหน้า (Forward Error Correction : FEC) ถ้าพบผิดบิตที่ภาครับจะทำการแก้ไขบิตผิดที่เกิดขึ้นโดยถ้าหากไม่สามารถแก้ไขได้ หรือ ระบบรับส่งข้อมูลมีเพียงการตรวจจับการผิดของบิตแล้วเมื่อพบบิตผิด ภาครับจะต้องตรวจจับการผิดให้ได้

- การตอบรับ (Acknowledgement : ACK) ถ้าเฟรมที่รับได้เป็นเฟรมที่ถูกต้อง ภาครับจะตอบรับ (ACK) เพื่อยืนยันให้ภาคส่งทราบว่าเฟรมที่ส่งมาถูกต้อง และตอบปฏิเสธ (Negative Acknowledgement : NACK) เมื่อตรวจพบการผิดของบิต

- การส่งซ้ำ (Retransmit) ภาคส่งจะส่งเฟรมที่ถูกตอบปฏิเสธกลับไปใหม่ รวมทั้งส่งเฟรมซ้ำในกรณีที่ส่งเฟรมไปแล้วไม่ได้รับการตอบรับ ACK หรือการตอบปฏิเสธ NACK และที่ภาคส่งหลังจากส่งเฟรมใดๆ ออกไปแล้ว ภาคส่งจะมีการตั้งเวลาขึ้นหากภาคส่งไม่ได้รับการตอบกลับในระยะเวลาที่ตั้งไว้เนื่องจากเฟรมข้อมูลหรือเฟรมตอบกลับหาย ภาคส่งจะจัดส่งเฟรมเดิมกลับไปใหม่อีกครั้ง

ขั้นตอนทั้งสามที่ได้กล่าวมานี้เราเรียกว่าการร้องขอส่งซ้ำอัตโนมัติ (Automatic Repeat Request: ARQ) โดยรูปแบบของการร้องขอส่งซ้ำอัตโนมัติ แบ่งออกได้เป็น 2 รูปแบบคือ การร้องขอส่งซ้ำอัตโนมัติแบบหยุดรอ และ การร้องขอส่งซ้ำอัตโนมัติแบบต่อเนื่อง ในรูปแบบการร้องขอส่งซ้ำอัตโนมัติแบบหยุดรอนั้น จะใช้กับการสื่อสารแบบกึ่งดูเพลก(Half duplex) เมื่อภาคส่งส่งข้อมูลออกไปแล้วก่อนที่จะส่งเฟรมต่อไป ภาคส่งจะรอการตอบรับหรือตอบปฏิเสธทุกครั้งหากไม่มีการตอบกลับหรือได้รับเฟรมปฏิเสธ ภาคส่งจะส่งเฟรมเดิมซ้ำอีก

2.3.3 การควบคุมการไหล

ในการส่งเฟรมจากภาคส่งไปยังภาครับ เพื่อให้การรับส่งเฟรมในอัตราที่ไม่เร็วเกินไปจนกระทั่งภาครับไม่สามารถรับข้อมูลนั้นได้ทัน หรือส่งเฟรมเกินขนาดของที่พักข้อมูลนั้น จะต้องมีการควบคุมการไหลของข้อมูลโดยการควบคุมการไหลของข้อมูลซึ่งแบ่งออกได้เป็นสองวิธี คือวิธีหยุดและรอ และวิธีการเลื่อนหน้าต่าง

2.3.3.1 วิธีหยุดและรอ

วิธีนี้เป็นวิธีควบคุมการไหลสำหรับการร้องขอส่งซ้ำอัตโนมัติแบบหยุดรอ ซึ่งในการร้องขอส่งซ้ำอัตโนมัติแบบหยุดรอนั้นเมื่อภาคส่ง ส่งข้อมูลไปแล้วจะรอการตอบรับหรือปฏิเสธจากภาครับ ดังนั้นหากภาครับยังอยู่ในสภาวะที่ไม่พร้อมรับข้อมูลภาครับจะไม่ตอบกลับภาคส่ง ซึ่งถือว่าเป็นการควบคุมการไหลของข้อมูลไปด้วย

2.3.3.2 วิธีการเลื่อนหน้าต่าง (Sliding window)

เป็นการควบคุมการไหลสำหรับการร้องขอส่งซ้ำอัตโนมัติแบบต่อเนื่องเพื่อควบคุมไม่ให้ข้อมูลเกินขนาดที่พักของภาครับ

2.3.4 วิธีการเข้าถึงช่องสัญญาณแบบหยั่งสัญญาณ (Polling)

เป็นการควบคุมการส่งข้อมูลในเครือข่ายการสื่อสาร โดยการหยั่งสัญญาณ ซึ่งจะทำการจัดลำดับให้กับสถานีทุติยภูมิซึ่งมีอยู่หลายสถานี แล้วส่งสัญญาณไปสอบถามว่ามีข้อมูลจะส่งมาหรือไม่ทีละสถานีตามลำดับจนครบและวนกลับมาหยั่งสัญญาณสถานีแรกใหม่ การหยั่งสัญญาณหรือการสอบถามจะทำได้ทีละเครื่อง ซึ่งแต่ละสถานีอาจต้องการส่งข่าวสารหรือไม่มีการส่งข่าวสารก็ได้ หากสถานีลูกข่ายที่ได้รับการหยั่งสัญญาณไม่มีข้อมูลที่จะส่ง สถานีลูกข่าย ต้องตอบกลับ

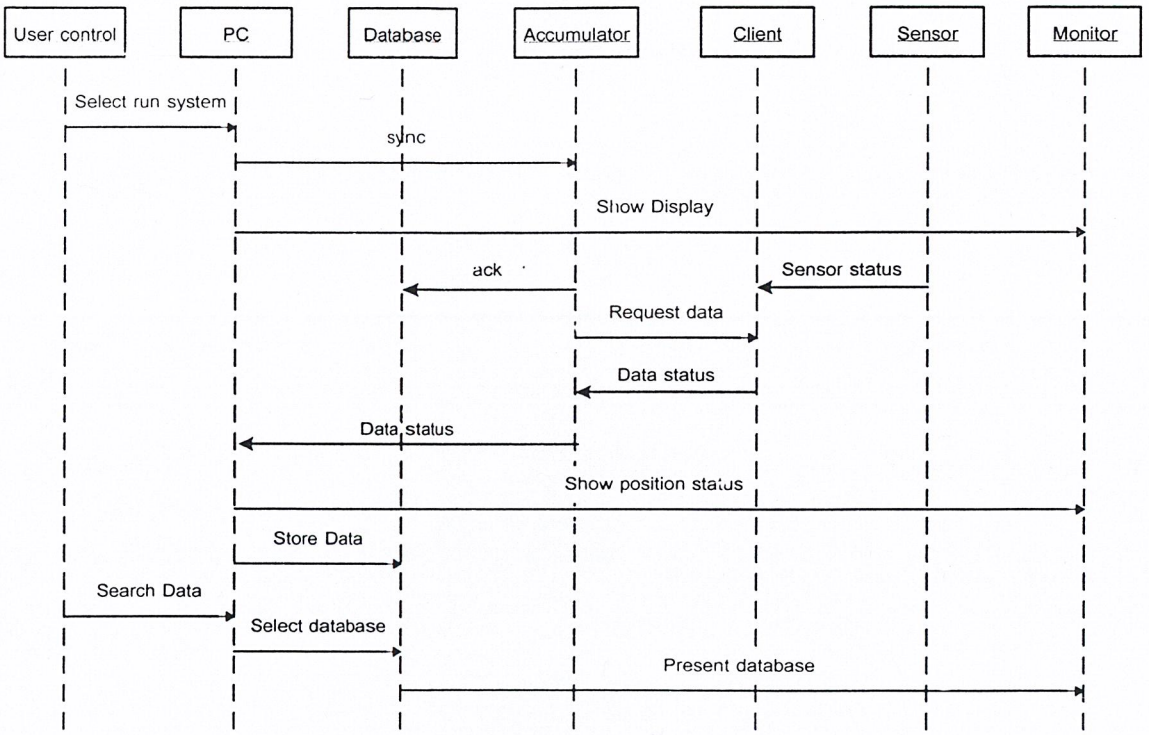
สถานีควบคุมเพื่อแสดงว่าได้รับการหยั่งสัญญาณ โดยเวลาในการหยั่งสัญญาณจะเพิ่มขึ้นตามจำนวนสถานีของลูกข่ายเรียกการหยั่งสัญญาณแบบนี้ว่าการหยั่งสัญญาณแบบทยอยเรียก (Roll and polling)

ในระบบการหยั่งสัญญาณนี้สถานีทุติยภูมิแต่ละตัวต้องมีรหัสประจำเครื่อง เมื่อมีการหยั่งสัญญาณเกิดขึ้น สถานีทุติยภูมิทุกสถานีจะรู้ว่ามีสัญญาณส่งมา แต่จะมีเพียงสถานีเดียวเท่านั้นที่จะรับสัญญาณนี้ได้ คือสถานีที่มีตำแหน่งรหัสตรงกับตำแหน่งที่ส่งมาพร้อมกับสัญญาณหยั่งสัญญาณ

บทที่ 3

การออกแบบ

3.1 ผังลำดับการทำงาน (Sequence diagram) ของระบบ

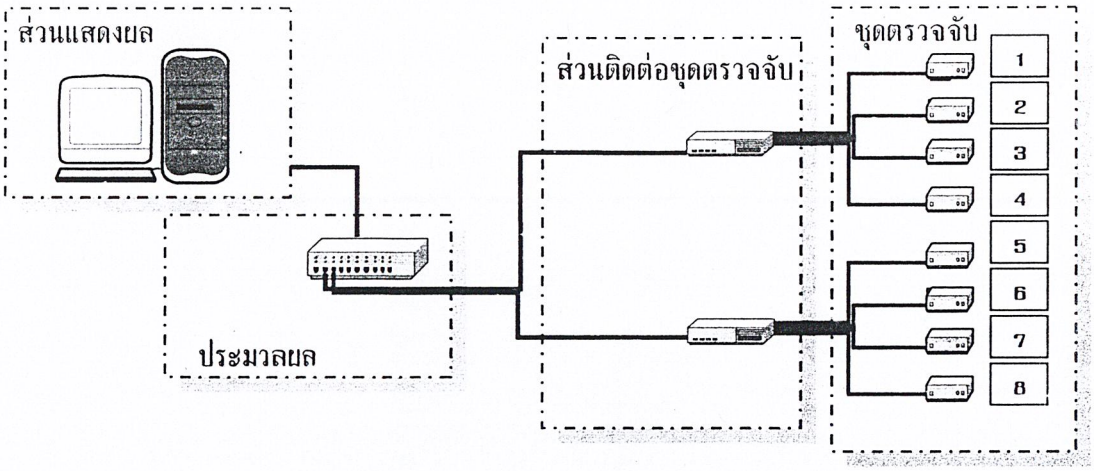


รูปที่ 3.1 แสดงผังลำดับการทำงาน (Sequence diagram) ของระบบ

3.2 การทำงานของระบบ

การทำงานโดยรวมของระบบจัดการที่จอดรถอัตโนมัตินั้นในส่วนของการเริ่มต้นการทำงานจะใช้เครื่องคอมพิวเตอร์เป็นตัวควบคุมโดยกำหนดสัญญาณเริ่มต้นการทำงาน โดยส่ง หัวของเฟรมข้อมูล (Header) เป็น “A” ให้กับส่วนประมวลผลกลางของระบบ เมื่อส่วนประมวลผลกลางได้รับสัญญาณแล้วจะทำการสร้างแอดเดรสเพื่อติดต่อกับส่วนติดต่อชุดตรวจจับ (Client) แต่ละจุดแบบกระจาย (Broadcast) โดยส่วนหัวเฟรมข้อมูลจะเป็นตัวระบุจุดหมายปลายทาง ให้ถูกต้องว่าต้องการติดต่อกับอุปกรณ์ชุดใด ในที่นี้ การติดต่อกับส่วนติดต่อชุดตรวจจับจะระบุเฟรมหัวข้อมูล (Header) เป็น “ I ” ซึ่งส่วนติดต่อชุดตรวจจับแต่ละจุดจะทำการรับค่าเข้ามาและเปรียบเทียบกับแอดเดรสของแต่ละตัวว่า ตรงกันหรือไม่ โดยในการระบุแอดเดรสนั้น จะสามารถทำได้โดยใช้

ดิปสวิทซ์ 8 หลัก เมื่อตรวจสอบแล้วว่าตรงกับที่มีอยู่ จะทำการส่งกับไปให้ ส่วนประมวลผลกลาง โดยระบุส่วนหัวของเฟรมข้อมูล (Header) เป็น "A" และ มี status = "Y", "N" เพื่อนำไปส่งต่อให้ส่วนแสดงผล นำไปแสดงผลให้ถูกต้องกับตำแหน่งของ ส่วนตรวจจับนั้นๆ โดยระบุหัวของเฟรมข้อมูล (Header) เป็น "C" จากนั้นจะทำการเพิ่มค่าแอดเดรสขึ้นหนึ่งค่า และทำการส่งออกไปแบบเดิมจนครบ 256 ตำแหน่ง จึงเริ่ม นับ 1 ใหม่ เมื่อส่วนแสดงผลรับข้อมูลที่มีส่วนหัวของเฟรมข้อมูลเป็น "C" ได้ ก็จะทำการตรวจสอบตำแหน่ง (Address) และสถานะ(Status) ของข้อมูลที่ส่งมาตามเงื่อนไข แล้วทำการแสดงผลพร้อมจัดเก็บข้อมูล



รูปที่ 3.2 แสดงโครงสร้างของระบบ

3.3 รูปแบบของเฟรมข้อมูล

Header	Address	Position	Status

รูปที่ 3.3 แสดงรูปแบบเฟรมข้อมูล

คำอธิบายส่วนประกอบของเฟรมข้อมูล

Header – เป็นส่วนที่ใช้ในการระบุข้อมูลว่าเป็นของอุปกรณ์ใด โดยใช้รหัสแอสกี (ACSII Code) มีขนาด 1 ไบต์ ในที่นี้กำหนดให้

"A" ใช้ในการอ้างอิงถึงส่วนประมวลผลหลัก

"C" ใช้ในการอ้างอิงถึงส่วนแสดงผล

"I" ใช้ในการอ้างอิงส่วนติดต่อชุดตรวจจับ

Address – เป็นส่วนที่ใช้ในการกำหนด Address ของส่วนติดต่อชุดตรวจจับ (Client) ว่าเป็นตำแหน่งใด ใช้ข้อมูลเป็นเลขฐาน 16 ได้สูงสุด 256 ตำแหน่ง

Position – เป็นส่วนที่ใช้ในการระบุว่าเป็นชุดตรวจจับตัวใดของชุดติดต่อชุดตรวจจับ ซึ่งชุดตรวจจับ 1 จุดจะมีชุดตรวจจับได้สูงสุด 4 ตัว

Status – เป็นส่วนที่ใช้ในการระบุสถานะของชุดตรวจจับว่ามีรถจอดหรือไม่โดยกำหนดให้

"Y" หมายถึง ชุดตรวจจับตรวจพบว่ามีรถจอด

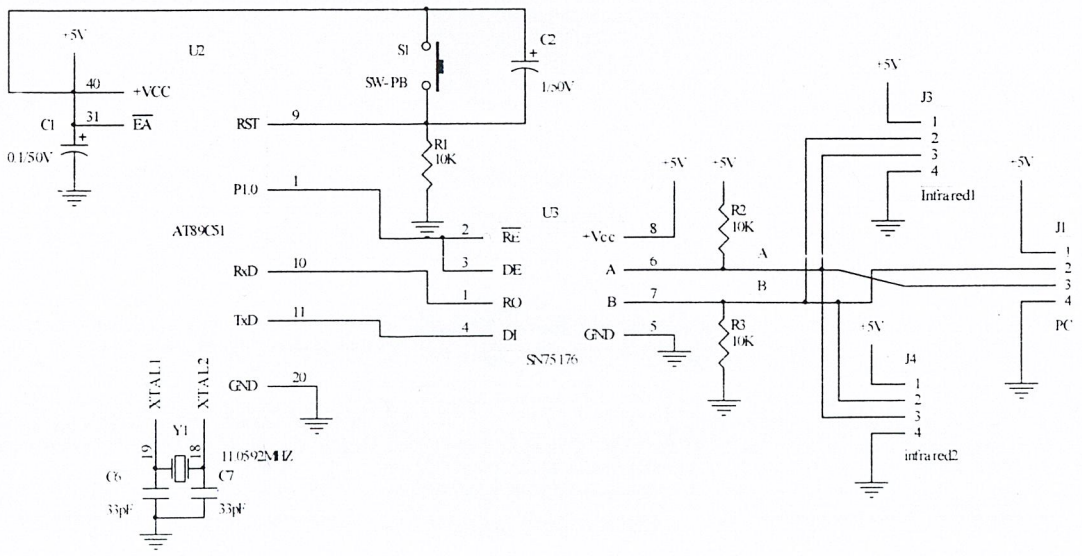
"N" หมายถึง ชุดตรวจจับไม่พบว่ามีรถจอด

3.4 ส่วนประมวลผลหลัก

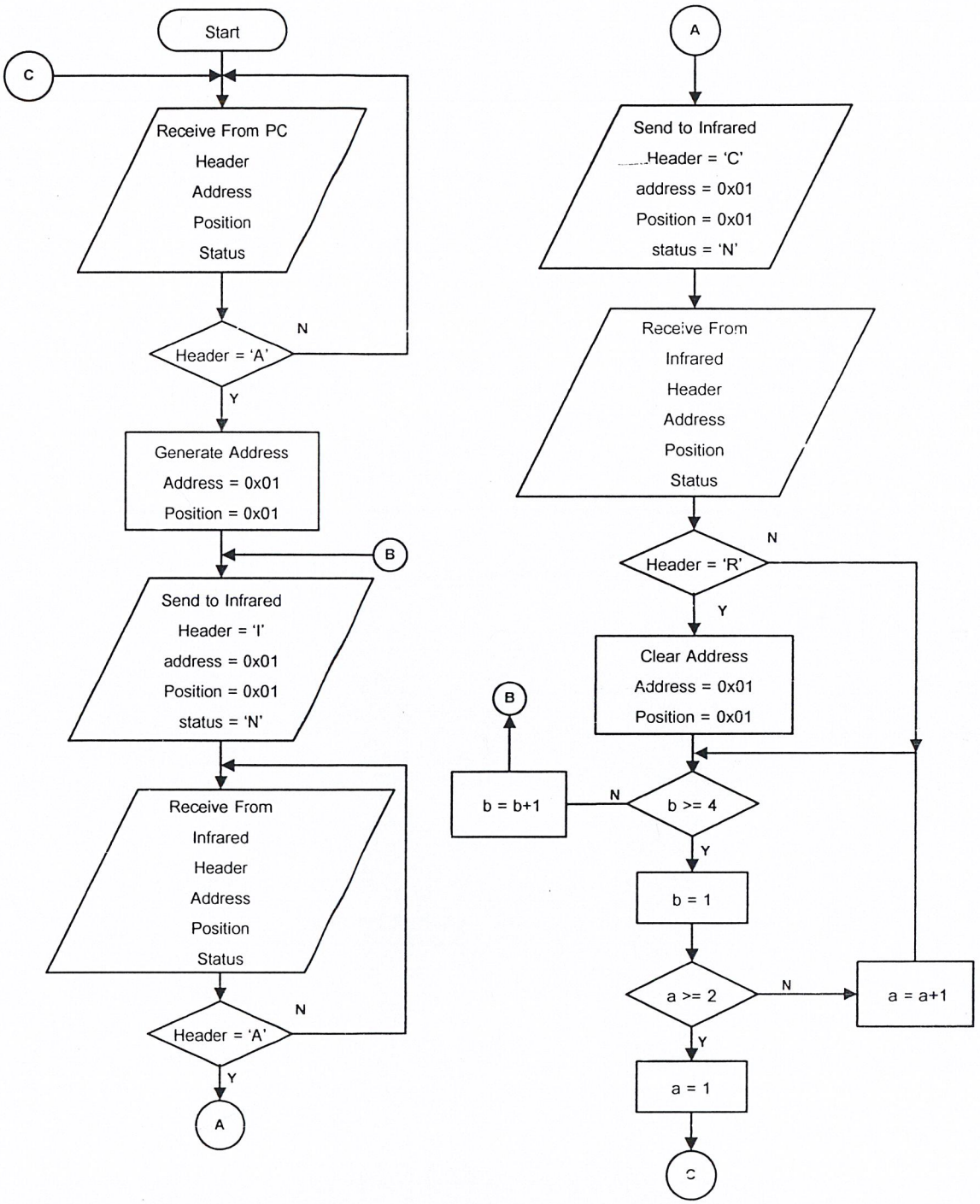
- ทำหน้าที่สร้างแอดเดรสเพื่อติดต่อกับส่วนติดต่อชุดตรวจจับ (Client) แต่ละจุดแบบ Broadcast โดยกำหนดหัวของเฟรมข้อมูล (Header) จะกำหนดค่าเป็น "I" หากตัวใดที่มีชุดตรวจจับที่กำหนดแอดเดรส (Address) ตรงกับที่ส่งมา ก็จะทำการตอบสถานะของชุดตรวจจับนั้นว่ามีค่าเป็น "0" (ไม่พบรถ) หรือ "1" (พบรถ) กลับไปให้ ส่วนประมวลผลหลัก

- ทำหน้าที่เพิ่มค่าแอดเดรสขึ้นหนึ่งค่า และทำการส่งออกไปแบบเดิมจนครบ "256" จึงเริ่มนับที่ "1"ใหม่ และ ระบุตำแหน่งของชุดตรวจจับโดยใช้เฟรมตำแหน่ง (Position) เพื่อเลือกรับข้อมูลว่าต้องการติดต่อจุดไหน โดยจะส่งเฟรมสถานะ (Status) กลับไปยังส่วนประมวลผลกลาง

- ทำหน้าที่นำค่าสถานะที่ได้ส่งไปที่ส่วนแสดงผลโดยใช้วิธีส่งแบบ Broadcast เช่นกันโดยส่วนหัวของเฟรมข้อมูล (Header) จะกำหนดค่าเป็น "C" ซึ่งเมื่อส่วนแสดงผลรับค่ามาแล้วก็จะนำไปจัดการแสดงผล



รูปที่ 3.4 แสดงวงจรส่วนประมวลผลหลัก



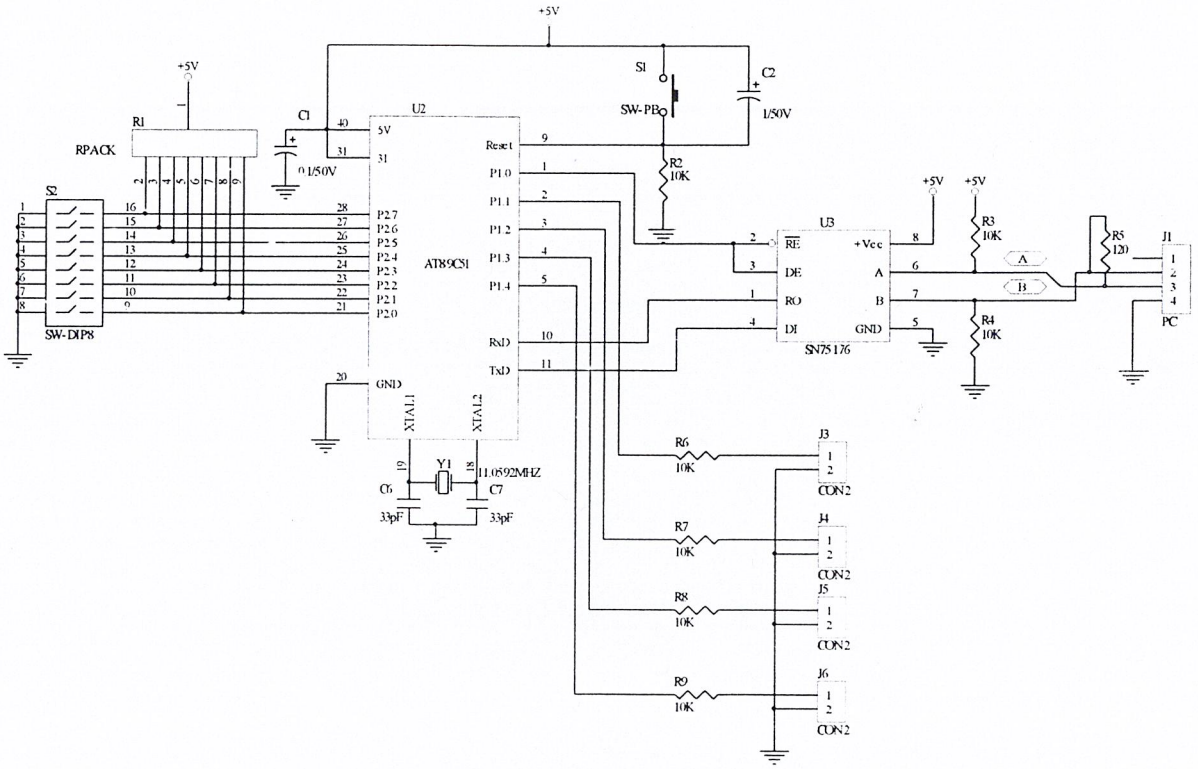
รูปที่ 3.5 แสดงขั้นตอนการทำงานของส่วนประมวลผลหลักของระบบ

3.5 ส่วนติดต่อชุดตรวจจับ (Client)

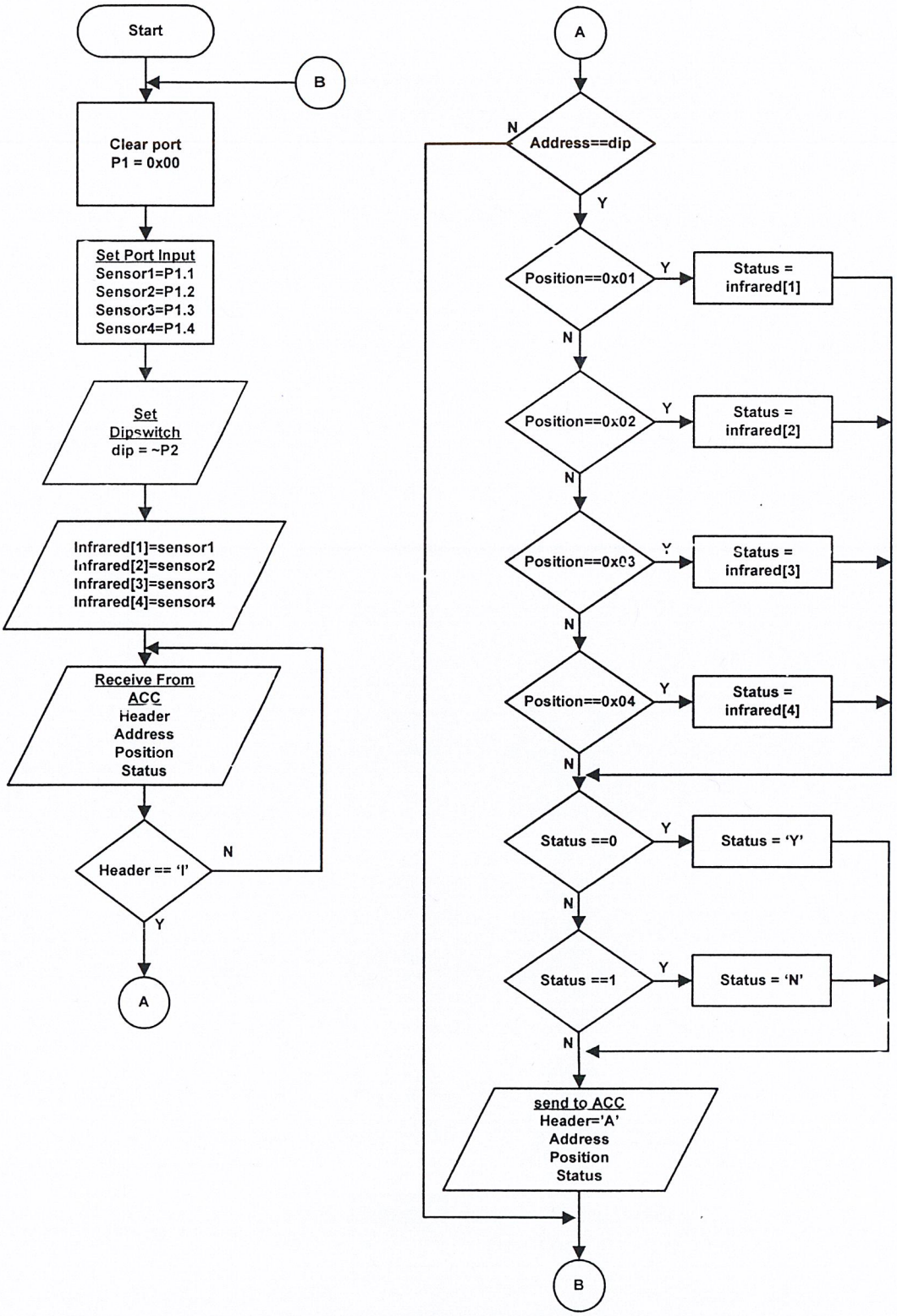
- ทำหน้าที่กำหนดค่าแอดเดรสให้กับชุดตรวจจับแต่ละชุด โดยส่วนติดต่อชุดตรวจจับ 1 ตัว ต่อชุดตรวจจับ 2 ชุด โดยจะใช้พอร์ตขนาน P1.1, P1.2, P1.3, P1.4 ในการติดต่อ

- ทำหน้าที่รอรับค่าที่ส่งมาจากส่วนประมวลผลหลัก ว่าส่วนหัวของข้อมูลเป็น “I” หรือไม่ ถ้าใช่จึงจะตรวจสอบแอดเดรสที่ได้รับ ถ้าค่าตรงกับแอดเดรสของชุดตรวจจับที่มีอยู่ จะส่งสถานะของชุดตรวจจับนั้นกลับไปให้ ส่วนประมวลผลหลักโดยระบุ ส่วนหัวข้อมูลเป็น “A”

- ส่วนที่พิเศษก็คือ เราสามารถใช้ ดิปสวิทซ์ ในการกำหนด Address ให้กับส่วนติดต่อชุดตรวจจับได้ ซึ่งโปรแกรมที่ใช้จะเป็นโปรแกรมชุดเดียวกัน ทำให้สะดวกในการใช้งาน



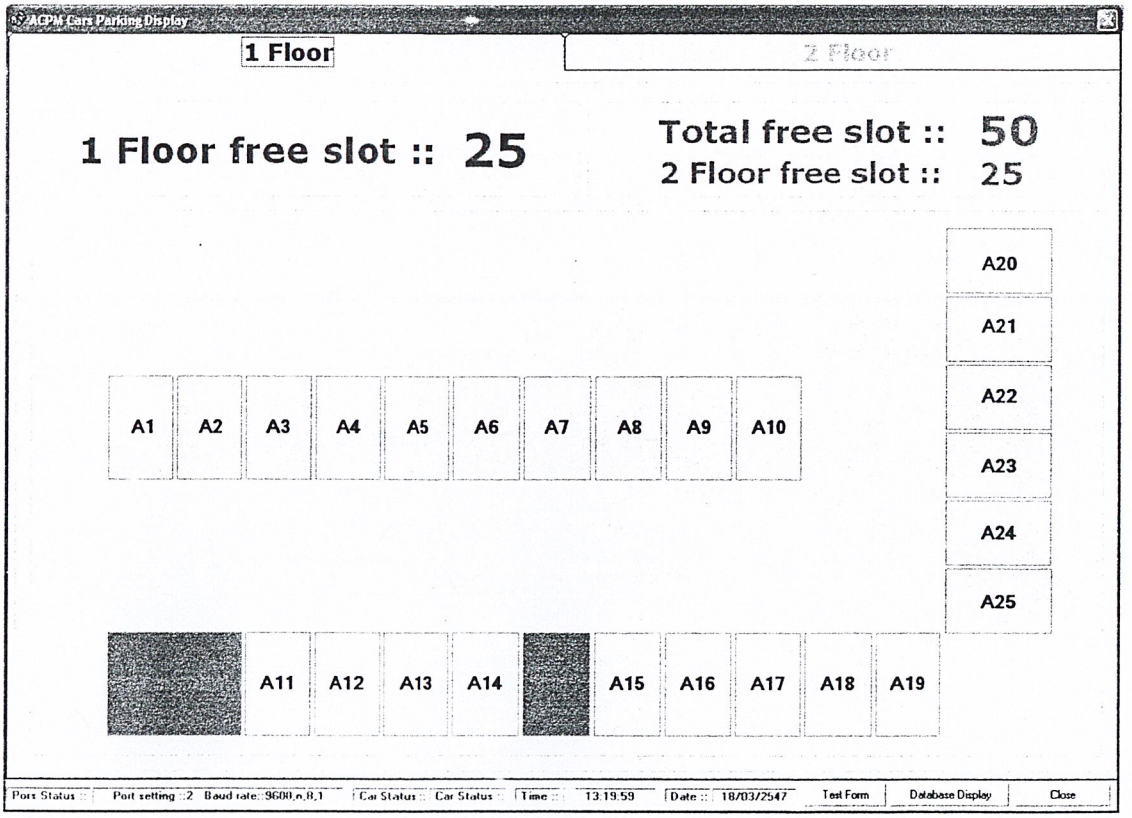
รูปที่ 3.6 แสดงวงจรส่วนติดต่อชุดตรวจจับ



รูปที่ 3.7 แสดงขั้นตอนการทำงานของส่วนติดต่อชุดตรวจจับ

3.6 ส่วนของภาคแสดงผล

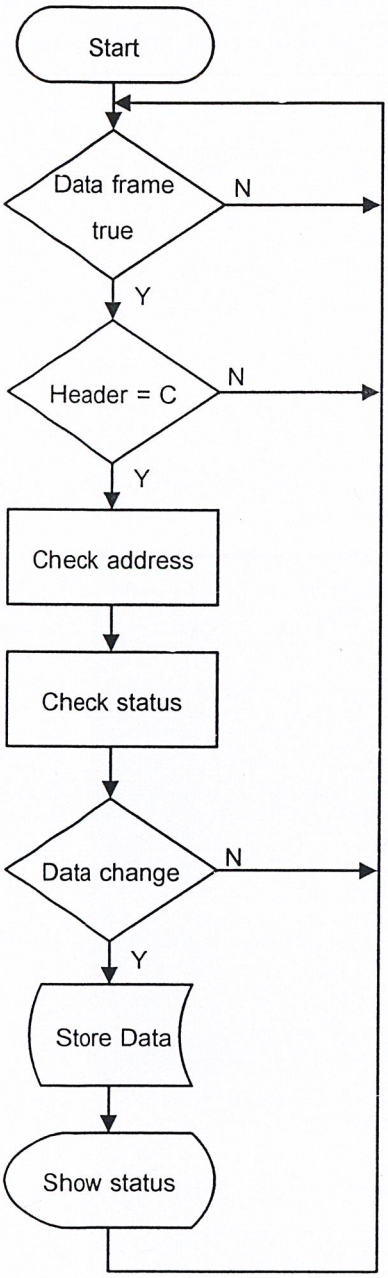
ในส่วนของภาคแสดงผล จะเป็นการแสดงผลผ่านคอมพิวเตอร์ส่วนบุคคล (Personal Computer) โดยใช้โปรแกรม Visual Basic ในการรับข้อมูลมาแสดงผล โดยตัวโปรแกรมจะทำงานโดยติดต่อกับพอร์ตอนุกรม ซึ่งใช้ control ที่สำคัญจากโปรแกรม Visual Basic คือ Mscomm ซึ่งใช้สำหรับการสื่อสารผ่านพอร์ตอนุกรมโดยเฉพาะ โดยในส่วนการแสดงผลนี้จะมีหน้าที่ที่สำคัญคือ



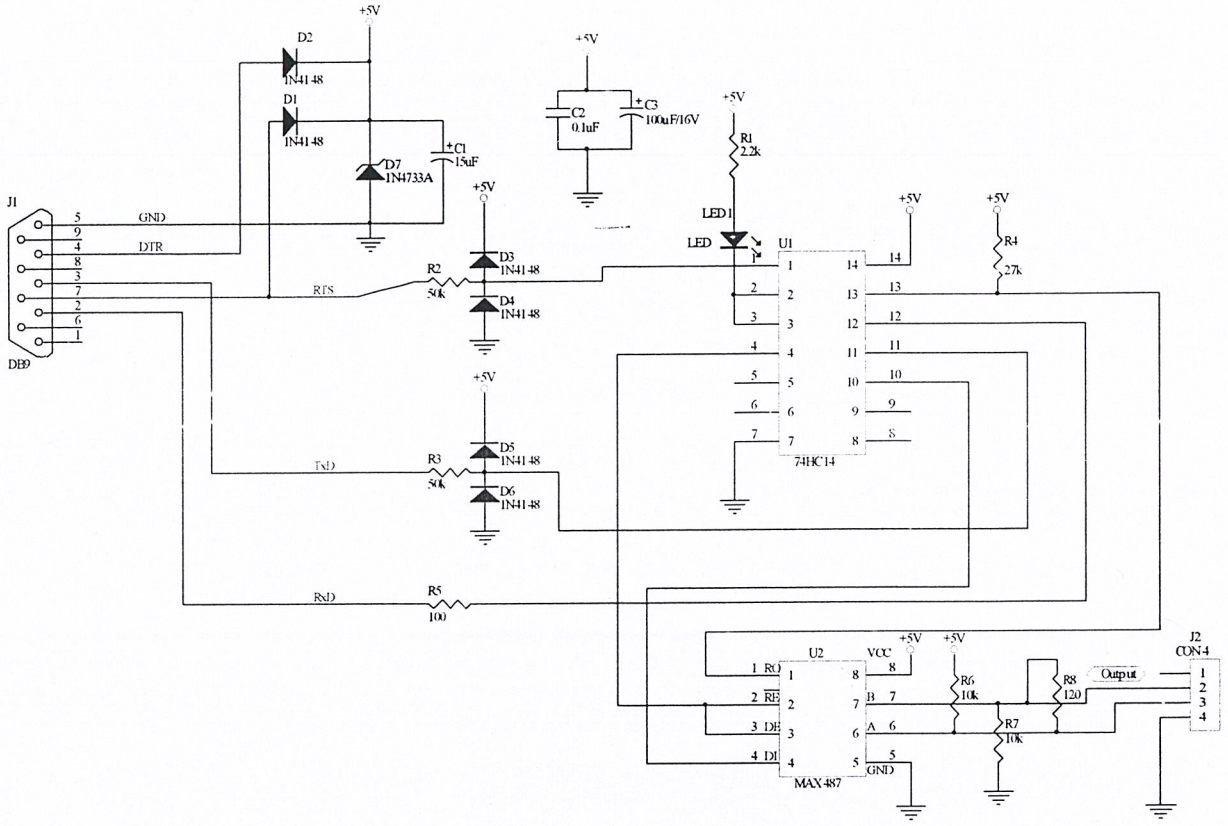
รูปที่ 3.8 แสดงหน้าจอของส่วนการแสดงผล

- ทำหน้าที่กำหนดสัญญาณเริ่มต้นการทำงานโดยหัวของเฟรมข้อมูล (Header) เป็น “A” ให้กับ ส่วนประมวลผลหลัก เพื่อเริ่มการทำงาน หรือ reset สถานะการทำงานใหม่ (Header เป็น “R”)

- ทำหน้าที่รับค่าที่ได้จาก ส่วนประมวลผลหลัก มาทำการตรวจสอบในส่วนหัวของเฟรมข้อมูลว่าเป็น “C” หรือไม่ ถ้าใช่ก็จะนำสถานะที่ได้มาตรวจสอบว่าเป็นข้อมูลที่มีการจัดเก็บแล้วหรือไม่ แล้วทำการแสดงผลที่หน้าจอตามข้อมูล address กับ positionที่ได้รับมาพร้อมจัดเก็บข้อมูล มีผังการทำงานดังนี้



รูปที่ 3.9 แสดงขั้นตอนการทำงานของส่วนภาคแสดงผล



รูปที่ 3.10 แสดงวงจรภาครับของส่วนแสดงผลที่มีการติดต่อแบบอนุกรม RS485

การส่งข้อมูลแบบ RS485 นั้น จะใช้การส่งแบบสมดุล (Balance Transmission) คือ เป็นการส่งข้อมูลที่ใช้ผลต่างของแรงดันของสัญญาณ 2 เส้น (ข้อมูล 1 เส้นจะใช้สายสัญญาณ 2 เส้น) โดยมีช่วงการเปลี่ยนแปลง (Transition Region) ของระดับสัญญาณอยู่ในช่วง $\pm$ 200 mVเท่านั้น ทำให้ข้อมูลทนต่อแรงสัญญาณรบกวนได้ดี สามารถส่งข้อมูลได้ด้วยความเร็วสูงสุดถึง 2.5 Mbps และได้ไกลสุดถึง 1000 เมตร

การใช้สายข้อมูลด้วย RS485 มีอยู่ 2 แบบคือ ใช้สายสัญญาณข้อมูล 4 เส้นและใช้สายสัญญาณข้อมูล 2 เส้น แบบ Full-Duplex และ Half-Duplex ตามลำดับ

การทำงานของวงจรก็แบ่งออกเป็น 2 ส่วนคือ วงจรภาคไฟเลี้ยง และวงจรภาคแปลงสัญญาณ ซึ่งมีรายละเอียดดังนี้

1. วงจรภาคไฟเลี้ยง มีหน้าที่ สร้างไฟเลี้ยงให้แก่วงจรภาคแปลงสัญญาณซึ่งต้องการไฟเลี้ยงเดี่ยวขนาด 5V โดยวงจรที่ใช้คือ การใช้ Diode D1, D2 และ ZD7 ในการ rectifier โดย

D1,D2 ทำหน้าที่ป้องกันไม่ให้ไฟลบประมาณ -12V ผ่านเข้ามาในวงจร ส่วน ZD7 ทำหน้าที่รักษาระดับแรงดันที่จ่ายออกไปให้มีขนาด 5V

2. วงจรภาคแปลงสัญญาณ หน้าที่หลักของวงจรก็คือ แปลงสัญญาณ RS232 ซึ่งเป็นการส่งสัญญาณแบบไม่สมดุล (Unbalanced Transmission) ให้เป็นสัญญาณ RS485 ซึ่งเป็นการส่งสัญญาณแบบสมดุล โดยการทำงานก็คือ

- ขา 2 - RxD นั้นเป็นสัญญาณข้อมูล Inputที่ PC ได้รับจากการที่วงจรอื่นส่งเข้ามา

- ขา 3 - TxD นั้นเป็นสัญญาณข้อมูล Output ที่ PC ต้องการส่งออกไป

- ขา 7 - RTS เป็นสัญญาณควบคุมทิศทางของสัญญาณข้อมูลว่าต้องการรับหรือส่ง โดยถ้า RTS = 12V หรือ มากกว่า 5V จะเป็นการส่งข้อมูล คือการแปลงสัญญาณจาก RS232 เป็น RS485 แต่ถ้าRTS =-12V หรือ น้อยกว่า 0V จะเป็นการรับข้อมูล คือแปลงสัญญาณจาก RS485 เป็น RS232 และยังมีส่วนหนึ่งใช้ในการสร้างไฟเลี้ยงให้วงจรด้วย

- ขา 4 - DTR ใช้ในการสร้างไฟเลี้ยงให้แก่วงจร

การทำงานของวงจรภาครับสัญญาณข้อมูล (มองจากด้าน RS-232) ที่ขา RE ของ IC2 (MAX487) ต้องเป็น 0 ซึ่งจะทำหน้าที่เป็นตัวรับสัญญาณ RS-485 สัญญาณที่รับเข้ามาจะถูกตีเป็น logic 0 หรือ 1 และส่งสัญญาณออกไปที่ขา RO สัญญาณ RO นี้ จะผ่าน IC2 ไปเพื่อให้ระดับสัญญาณแรงดันและ ลอจิกของสัญญาณเข้ากันได้กับ RS-232

ส่วนการทำงานวงจรภาคส่งนั้น ที่ขา DE ของ IC2 ต้องเป็น logic 1 หรือ RTS = 5V ทำให้ระดับสัญญาณที่ ขา A,B ของสัญญาณ ขึ้นกับระดับสัญญาณที่ขา DI ของ IC2 โดย DI นี้ ได้มาจากสัญญาณ TxD ที่ผ่าน IC1 โดยมี Diode ต่อเพื่อช่วยขจัดแรงดันและกระแสของสัญญาณที่เข้ามายัง IC1 Diode ตัวอื่นๆก็เช่นกัน

LED1 ใช้แสดงสถานการณ์ทำงานของวงจร ว่าอยู่ในสถานะเป็นตัวส่ง (Transmitter) ในทางตรงข้ามหาก LED 1 ดับแสดงว่าวงจรทำหน้าที่เป็นตัวรับ (Receiver)

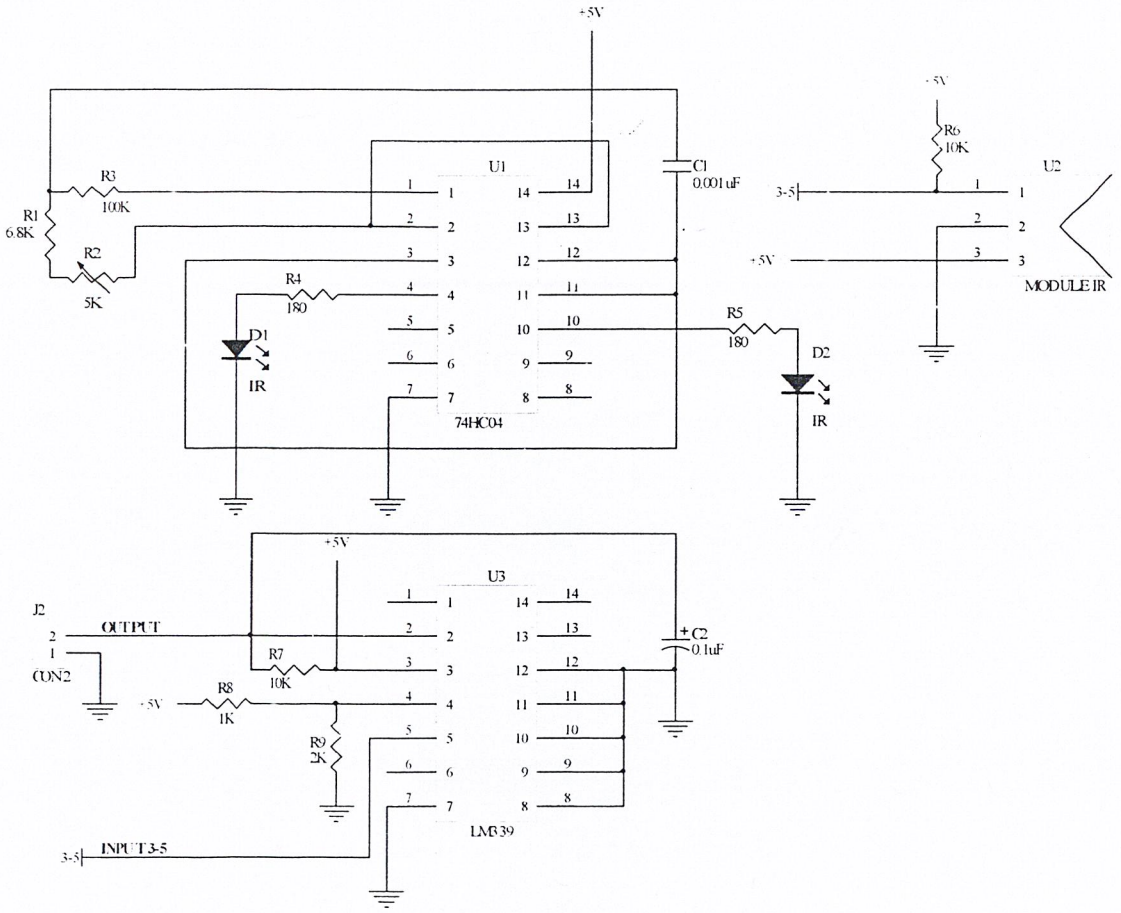
ตัวต้านทาน R6, R7 ที่ต่ออยู่ที่Output ของ IC2 นั้น มีหน้าที่ป้องกันการเกิด Start bit เมื่อทุกวงจรที่ต่ออยู่ในสถานะ เป็นตัวรับหมด ส่วน R8 นั้นเป็นความต้านทานที่ใช้ลดปัญหาเมื่อสายที่ต่อยาวมาก (Match Impedance) ในสายสัญญาณ

3.7 ชุดตรวจจับ (Sensor)

- ทำหน้าที่ในการตรวจจับว่ามีรถจอดหรือไม่ โดยส่วนนี้จะใช้อุปกรณ์อินฟาเรดเป็นตัวตรวจจับซึ่งจะทำการส่งความถี่ที่ 38-40 KHz ออกไป เมื่อกระทบกับวัตถุก็จะสะท้อนความถี่ของแสงอินฟาเรดกลับมายังตัวตรวจจับ(Detector) และ ส่งค่าoutput เป็นแรงดันไฟ ดังนี้

- 0V มีstatus = "Y" หมายถึงชุดตรวจจับตรวจพบว่ามีรถจอด

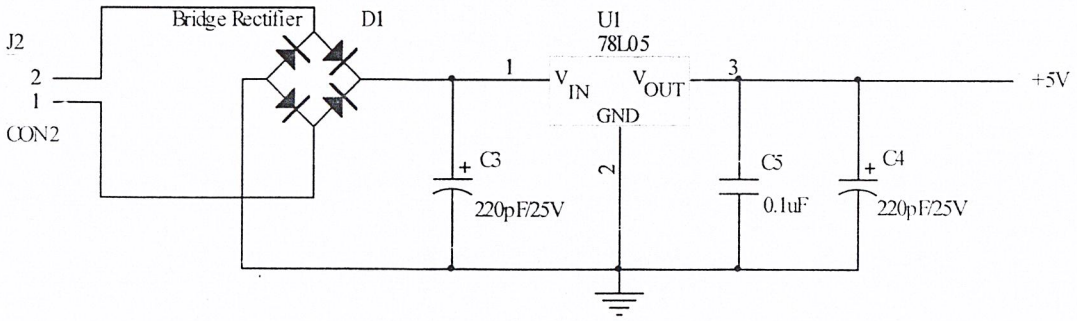
- 5V มีstatus = "N" หมายถึงชุดตรวจจับตรวจไม่พบว่ามีรถจอด



รูปที่ 3.11 แสดงวงจรภาครับของชุดตรวจจับ

3.8 ส่วนของภาคจ่ายไฟเลี้ยงวงจร

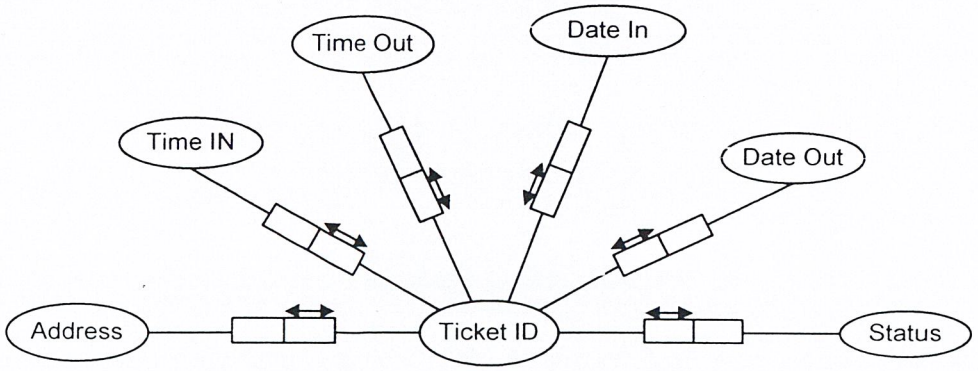
ในส่วนนี้ได้ใช้ IC 78L05 ซึ่งเป็น IC Regulator ทำหน้าที่แปลงไฟเลี้ยงจาก 12Vแรงดันไฟ 5V เพื่อใช้เลี้ยงวงจรต่างๆ ในแต่ละวงจร และ Diode Bridge Rectifier ทำหน้าที่แปลงแรงดันจากกระแสสลับให้เป็นกระแสตรง และ มี Capacitor ทำหน้าที่ ลดการกระเพื่อมของแรงดัน (Ripper)



รูปที่ 3.12 แสดงวงจรแหล่งจ่ายไฟ 5V

3.9 ส่วนของฐานข้อมูล

เป็นส่วนที่ใช้ในการจัดเก็บข้อมูลที่ส่งมาจากส่วนแสดงผลเพื่อใช้ตรวจสอบภายหลัง มีรายละเอียดของฐานข้อมูล คือ Ticket ID, Address, Time IN, Time Out, Date IN, Date Out, Status โดยกำหนดให้ Ticket ID เป็นคีย์หลัก (Primary Key) ซึ่งมีประโยชน์ในการนำไปประยุกต์ใช้กับระบบการคิดค่าบริการที่จอดรถ เพราะสามารถตรวจสอบข้อมูลได้จากเวลารถเข้า และรถออก



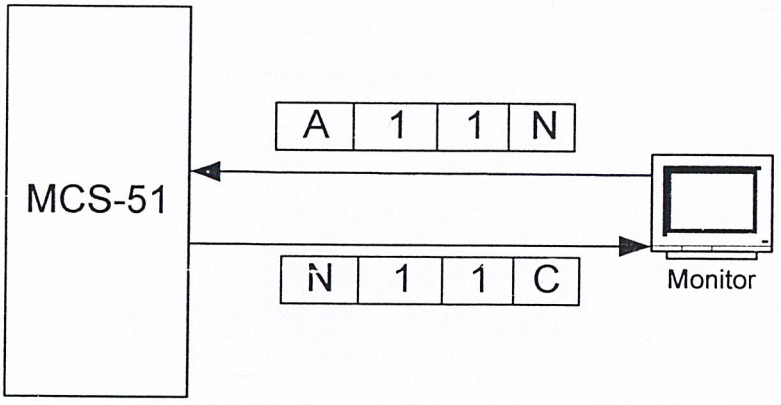
รูปที่ 3.13 แสดงความสัมพันธ์ของข้อมูล

บทที่ 4

ผลการทดลอง

4.1 การทดลองรับ-ส่งเฟรมข้อมูล

เป็นการทดลองการสื่อสารระหว่างคอมพิวเตอร์ และ ไมโครคอนโทรลเลอร์ โดยคอมพิวเตอร์จะส่งข้อมูลไปยังไมโครคอนโทรเลอร์ แล้วรอรับเฟรมข้อมูลจากวงจรไมโครคอนโทรลเลอร์ เพื่อนำไปแสดงผลที่หน้าจอคอมพิวเตอร์ โดยใช้โปรแกรม Hyper Terminal ด้วย Baudrate 9600 ดังรูป โดยมีการระบุการเข้าถึงอุปกรณ์โดยใช้ ส่วนหัวข้อมูล (Header)

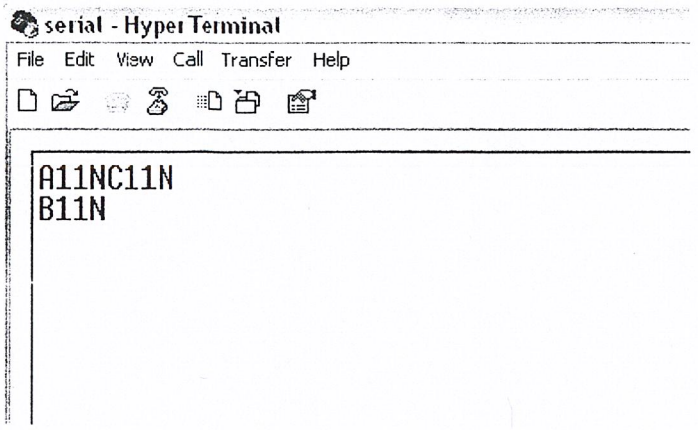


รูปที่ 4.1 แสดงรูปแบบเฟรมข้อมูลที่คาดว่าจะเกิดขึ้นหากรูปแบบการสื่อสารถูกต้อง

4.1.1 ผลการทดลองที่4.1

เมื่อคอมพิวเตอร์ส่งข้อมูลโดยระบุ Header = 'A' ไมโครคอนโทรลเลอร์ จะตอบกลับด้วยเฟรมข้อมูล โดยมี Header = 'C' เพื่ออ้างถึงส่วนแสดงผล ซึ่งจะแสดงที่หน้าจอของโปรแกรม Hyper Terminal

หากส่ง Header เป็นข้อมูลอื่น ผลที่ได้คือไม่มีการเปลี่ยนแปลงเกิดขึ้นที่หน้าจอโปรแกรม Hyper Terminal จนกว่าจะมีการระบุ Header ที่ถูกต้อง



รูปที่ 4.2 แสดงผลการทดลองรับส่งเฟรมข้อมูลโดยใช้โปรแกรม Hyper Terminal

4.1.2 สรุปผลการทดลองที่4.1

ในการสื่อสารแบบอนุกรมนั้น จะต้องมีการระบุ Baudrate ให้กับอุปกรณ์ที่ต้องการสื่อสาร เพื่อให้มีอัตราการส่งข้อมูลที่เท่ากัน หากไม่มีการกำหนดการสื่อสารจะเกิดความผิดพลาดได้ โดยสามารถคำนวณได้ดังนี้

$$\text{Fbuad} = \frac{2^{\text{SMOD}} \times \text{Oscillator Frequency}}{32_D \times 12_D \times [256_D - (\text{TH1})]}$$

เพื่อใช้ในการคำนวณค่าเริ่มต้นให้กับ TH1 เพื่อใช้ในการกำหนด Baud rate ได้ดังนี้

$$\text{TH1} = \frac{2^{\text{SMOD}} \times \text{Oscillator Frequency}}{32_D \times 12_D \times \text{Baudrate}}$$

4.2 การทดลองวงจรตรวจจับแบบอินฟราเรด

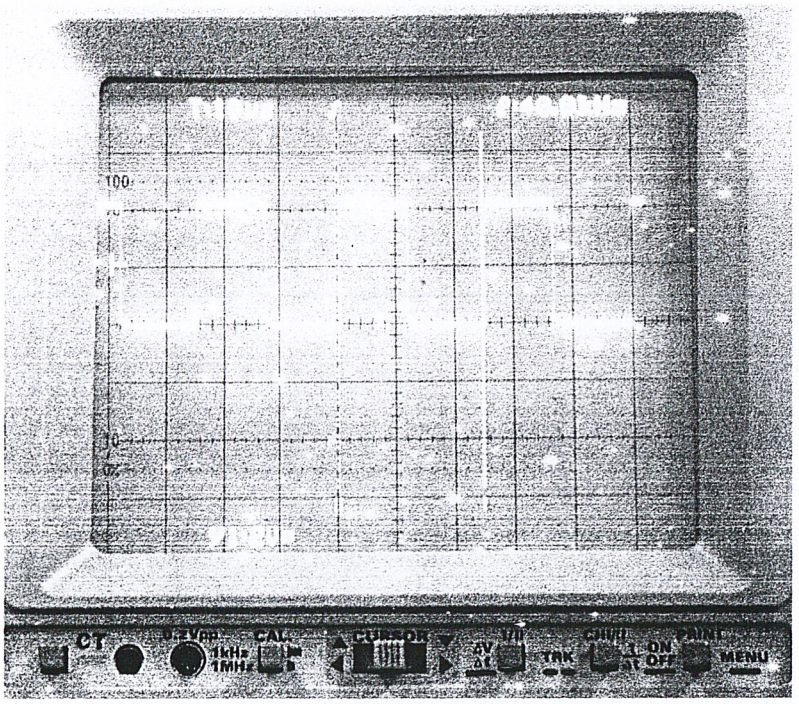
เป็นการทดลองวัดความถี่ที่วงจรตรวจจับส่งออก และทดลองวัดแรงดันที่ได้จาก Output ของวงจรตรวจจับ โดยทำการเปรียบเทียบค่าแรงดันที่ได้ของสองเหตุการณ์ที่น่าจะเกิดขึ้น คือ กรณีวงจรตรวจจับตรวจพบสิ่งกีดขวาง และกรณีวงจรตรวจจับไม่พบสิ่งกีดขวาง

4.2.1 ผลการทดลองที่ 4.2

- กรณีวงจรตรวจจับตรวจพบสิ่งกีดขวาง ผลที่ได้คือ แรงดันจาก Output = 3V
- กรณีวงจรตรวจจับไม่พบสิ่งกีดขวาง ผลที่ได้คือ แรงดันจาก Output = 4.5V

จากการทดลองทำให้ทราบระดับแรงดันที่ได้จากเอาท์พุตของวงจรตรวจจับ แต่เนื่องจากการรับสัญญาณอินพุตของไมโครคอนโทรเลอร์นั้นมีข้อกำหนด คือจะให้ Logic "0" เมื่อแรงดันเท่ากับ 0V และ Logic "1" เมื่อแรงดันเท่ากับ 4.5-5V ดังนั้นจึงต้องมีการต่อวงจร Comparator เพื่อปรับระดับแรงดันจากเอาท์พุทได้วัดได้ 3V ให้เท่ากับ 0 V และ แรงดันที่วัดได้ 4.5 ให้เท่ากับ 5V

การทดลองวัดความถี่ที่วงจรตรวจจับส่งออก สามารถวัดความถี่ได้ที่ 40 KHz



รูปที่ 4.3 ผลการทดลองวัดความถี่ที่วงจรตรวจจับส่งออก

4.2.2 สรุปผลการทดลองที่ 4.2

ชุดวงจรตรวจจับแบบอินฟราเรดนั้น จะให้แรงดัน Output ที่มีระดับแตกต่างกันและ มีการสื่อสารกันระหว่าง ภาครับ และ ภาคส่ง ด้วยความถี่ 38-40 kHz เมื่อมีวัตถุมาขวางอยู่ในระยะ 0-1 เมตร

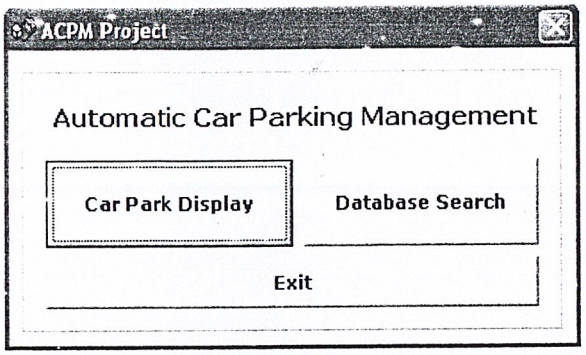
4.3 การทดลองใช้โปรแกรมระบบจัดการที่จอดรถอัตโนมัติ

โปรแกรมระบบจัดการที่จอดรถอัตโนมัติจะสามารถแบ่งส่วนการทำงานออกเป็น

4.3.1 เมนูหลัก

เป็นส่วนที่ให้ผู้ใช้เลือกฟอร์มที่ต้องการใช้งาน ซึ่งจะมีรายการให้เลือก 3 รายการคือ

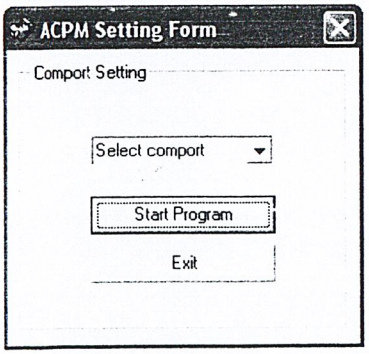
- Cars Parking Display เป็นส่วนของการจัดการแสดงผลตำแหน่งที่จอดรถ
- Database Display เป็นส่วนของการแสดงผลข้อมูลที่ได้ทำการจัดเก็บ
- Exit เป็นการออกจากการใช้งานโปรแกรม



รูปที่ 4.4 แสดงหน้าจอรายการเมนูหลัก

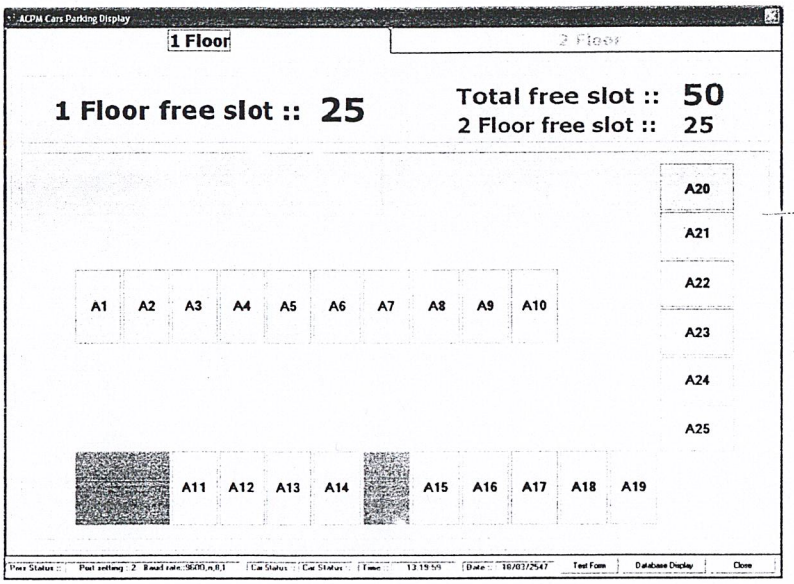
4.3.2 Cars Parking Display

หากผู้ใช้เลือกรายการ Cars Parking Display จะเป็นการเปิดระบบการตรวจสอบและจัดการพื้นที่จอดรถ ดังนั้นผู้ใช้จึงต้องทำการกำหนดค่าในส่วนของพอร์ทที่ใช้ติดต่อกับอุปกรณ์ของระบบจากเมนูย่อยที่ปรากฏขึ้นมา (ACPM Setting Form)



รูปที่ 4.5 แสดงหน้าจอสำหรับเลือกกำหนดพอร์ตที่ใช้ติดต่ออุปกรณ์ระบบ

เมื่อเลือกพอร์ทและเลือกรายการ Start Program แล้ว ระบบจะแสดงหน้าจอดังนี้

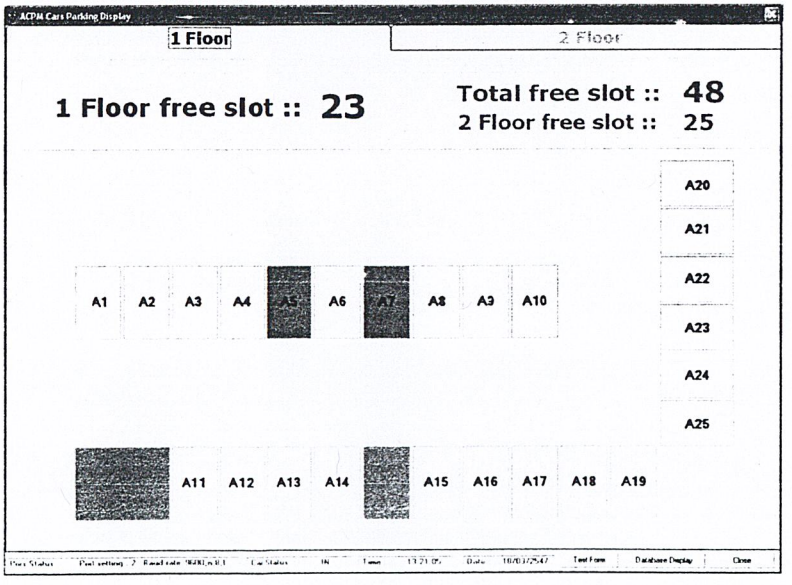


รูปที่ 4.6 แสดงหน้าจอของ ACPM Cars Parking Display

ส่วนของ Cars Parking Display นี้จะมีการเปลี่ยนแปลงเมื่ออุปกรณ์ของระบบตรวจพบรถที่เข้าจอด หรือตรวจพบการเปลี่ยนแปลงของบริเวณที่จอดรถ

กรณีที่มีรถเข้ามาจอด สิ่งที่เปลี่ยนแปลงคือ

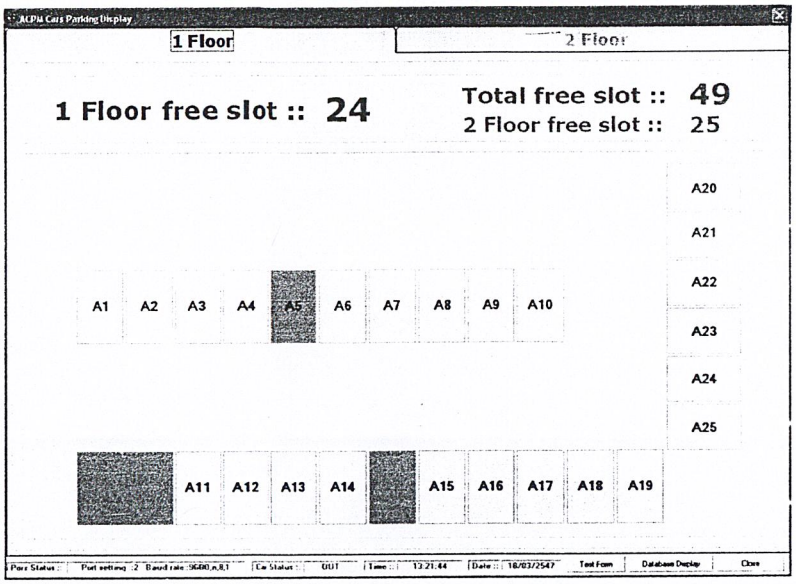
- ตำแหน่งที่ตรวจพบรถ จะเปลี่ยนสีจากสีเทาเป็นสีแดง
- ค่าจำนวนที่จอดรถที่สามารถจอดได้ จะลดค่าลง ดังรูป



รูปที่ 4.7 แสดงหน้าจอของ ACPM Cars Parking Display ในกรณีที่มีรถเข้ามาจอด

กรณีที่รถออกจากที่จอดรถ

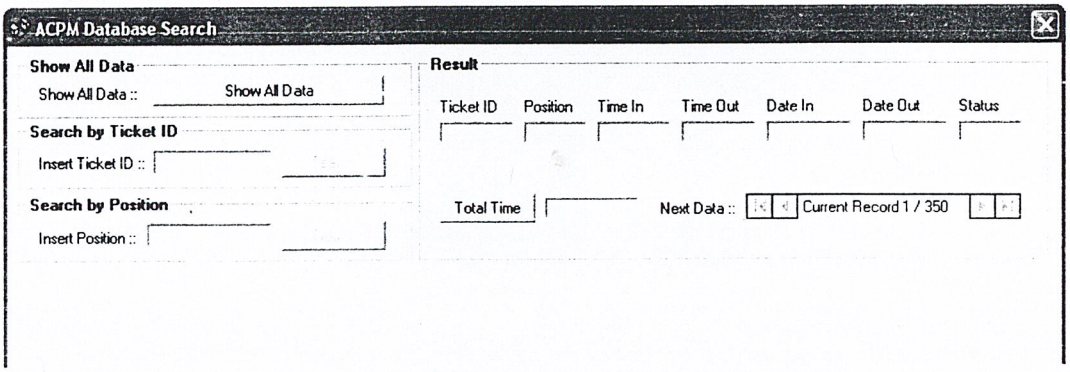
- ตำแหน่งที่รถออกจากที่จอดรถจะเปลี่ยนจากสีแดงเป็นสีเทา
- ค่าจำนวนที่จอดรถที่สามารถจอดได้จะเพิ่มขึ้น ดังรูป



รูปที่ 4.8 แสดงหน้าจอของ ACPM Cars Parking Display ในกรณีที่มีรถออกจากที่จอดรถ

4.3.3 Database search

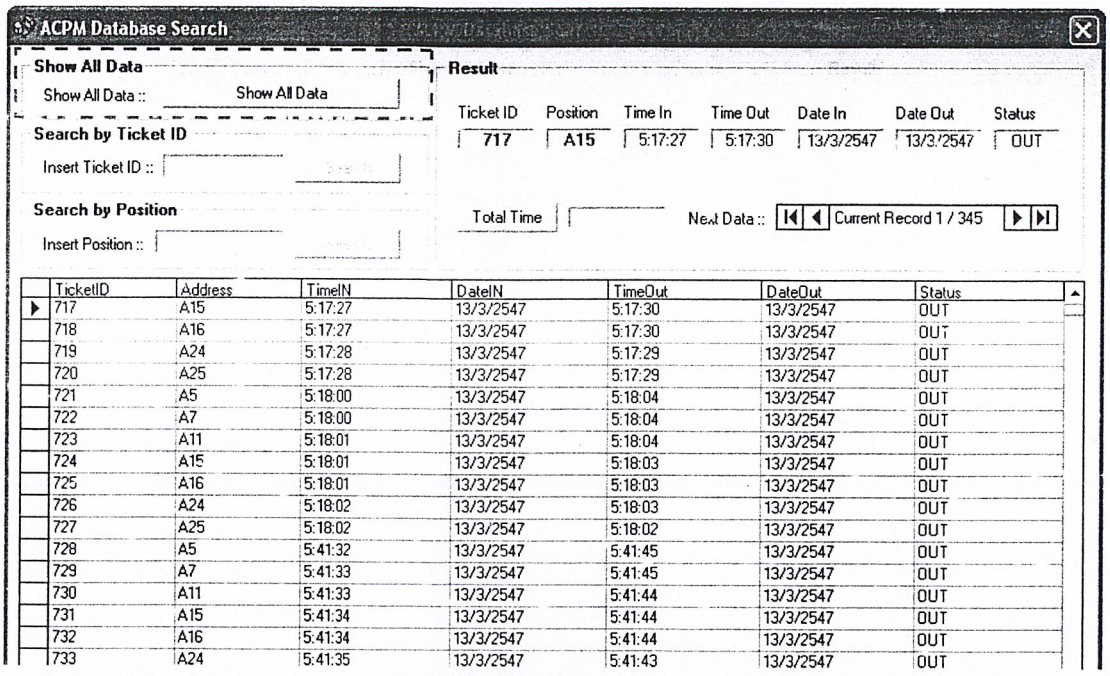
หากผู้ใช้เลือกรายการ Database search จะเป็นการแสดงผลในส่วนของข้อมูลที่ได้ทำการจัดเก็บเมื่อมีการเปลี่ยนแปลงสถานะของบริเวณที่จอดรถ ผู้ใช้สามารถเลือกเพื่อดูข้อมูลได้สามรูปแบบคือ Show All Data, Search by Ticket ID, Search by Position



รูปที่ 4.9 แสดงหน้าจอของ ACPM Database Search

4.3.3.1 Show All Data

เป็นการเลือกดูข้อมูลทั้งหมดที่จัดเก็บ ซึ่งจะแสดงผลดังรูป

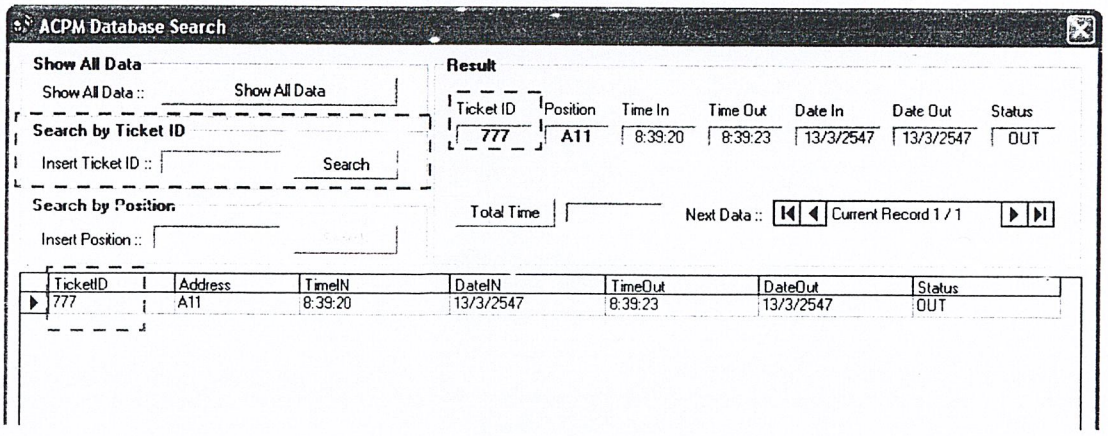


TicketID	Address	TimeIN	DateIN	TimeOut	DateOut	Status
717	A15	5:17:27	13/3/2547	5:17:30	13/3/2547	OUT
718	A16	5:17:27	13/3/2547	5:17:30	13/3/2547	OUT
719	A24	5:17:28	13/3/2547	5:17:29	13/3/2547	OUT
720	A25	5:17:28	13/3/2547	5:17:29	13/3/2547	OUT
721	A5	5:18:00	13/3/2547	5:18:04	13/3/2547	OUT
722	A7	5:18:00	13/3/2547	5:18:04	13/3/2547	OUT
723	A11	5:18:01	13/3/2547	5:18:04	13/3/2547	OUT
724	A15	5:18:01	13/3/2547	5:18:03	13/3/2547	OUT
725	A16	5:18:01	13/3/2547	5:18:03	13/3/2547	OUT
726	A24	5:18:02	13/3/2547	5:18:03	13/3/2547	OUT
727	A25	5:18:02	13/3/2547	5:18:02	13/3/2547	OUT
728	A5	5:41:32	13/3/2547	5:41:45	13/3/2547	OUT
729	A7	5:41:33	13/3/2547	5:41:45	13/3/2547	OUT
730	A11	5:41:33	13/3/2547	5:41:44	13/3/2547	OUT
731	A15	5:41:34	13/3/2547	5:41:44	13/3/2547	OUT
732	A16	5:41:34	13/3/2547	5:41:44	13/3/2547	OUT
733	A24	5:41:35	13/3/2547	5:41:43	13/3/2547	OUT

รูปที่ 4.10 แสดงหน้าจอของ ACPM Database Search เมื่อเลือกรายการ Show All Data

4.3.3.2 Search by Ticket ID

เป็นการเลือกดูข้อมูลตามหมายเลขบัตรจอดรถ ซึ่งจะแสดงผลดังรูป

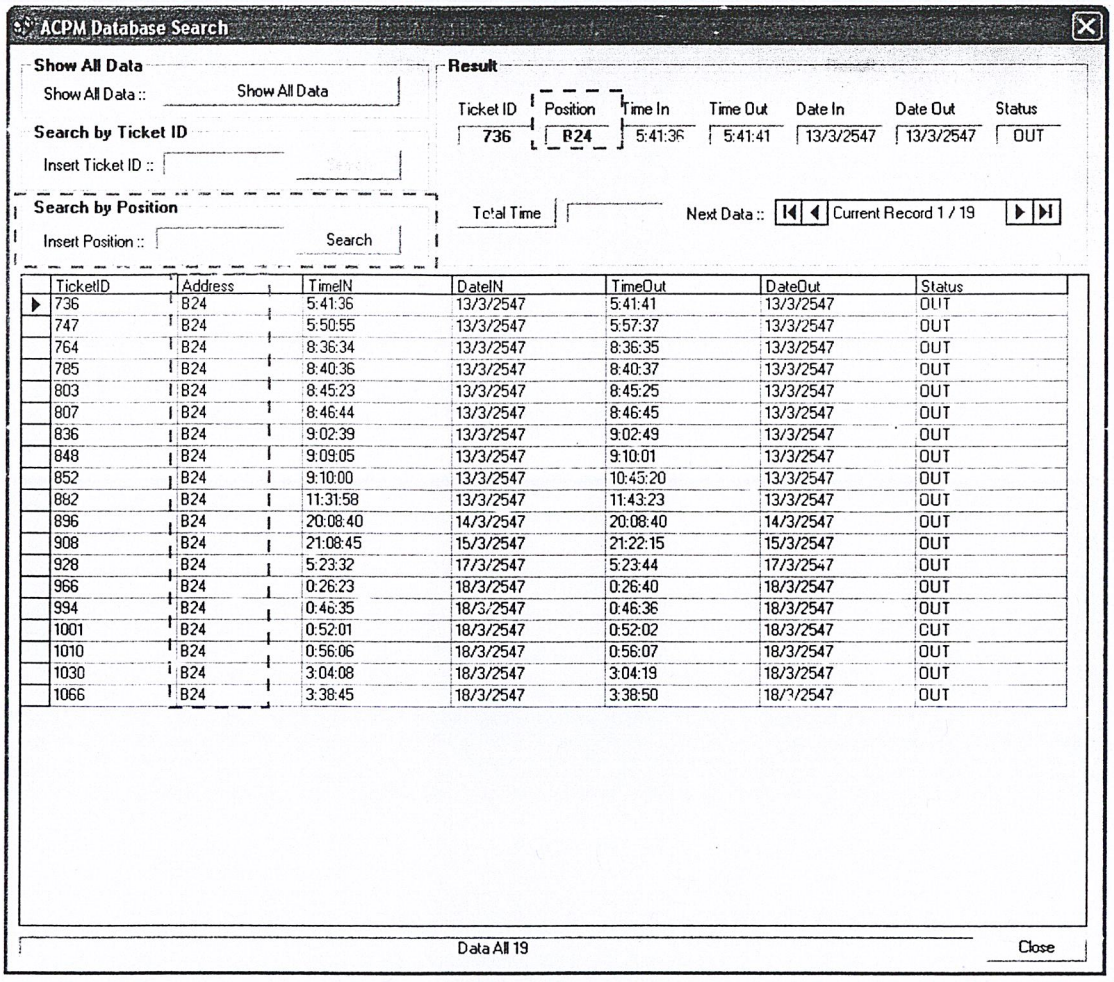


TicketID	Address	TimeIN	DateIN	TimeOut	DateOut	Status
777	A11	8:39:20	13/3/2547	8:39:23	13/3/2547	OUT

รูปที่ 4.11 แสดงหน้าจอของ ACPM Database Search เมื่อเลือกรายการ Search by Ticket ID

4.3.3.3 Search by Position

เป็นการเลือกดูข้อมูลจากตำแหน่งที่จอดรถ



รูปที่ 4.12 แสดงหน้าจอของ ACPM Database Search เมื่อเลือกรายการ Search by Position

บทที่ 5
บทสรุปและวิจารณ์

การใช้คอมพิวเตอร์ติดต่อกับอุปกรณ์ภายนอก เช่น วงจรควบคุมอุปกรณ์ไฟฟ้านั้น จำเป็นต้องใช้การสื่อสารแบบอนุกรมโดยผ่านพอร์ตของคอมพิวเตอร์ในการสื่อสาร โดยผ่านวงจร RS-232 หากเป็นการสื่อสารในระยะทางที่ไกลมากนั้นจำเป็นจะต้องใช้การสื่อสารอนุกรม RS-485 ในการสื่อสารแทน

การส่งข้อมูลจากคอมพิวเตอร์เพื่อติดต่อกับไมโครคอนโทรลเลอร์นั้นจำเป็นต้องมีการออกแบบเฟรมข้อมูลเพื่อใช้ในการสื่อสารให้สามารถสื่อสารได้อย่างถูกต้องแม้จะมีข้อมูลจำนวนมากก็ตาม ในการสื่อสารแบบกระจาย (Broadcast) นั้นจำเป็นจะต้องมีการกำหนดแอดเดรสของอุปกรณ์ลูกข่ายแต่ละตัวไม่ซ้ำกัน การส่งข้อมูลไปยังเครื่องลูกข่ายแบบกระจายนี้จะสามารถป้องกันการชนกันของข้อมูลได้ โดยเมื่อเครื่องลูกข่ายได้รับเฟรมข้อมูลแล้วหากพบว่าเป็นเฟรมข้อมูลของตัวเองจึงจะทำการส่งข้อมูลกลับไปให้หากไม่ใช่เฟรมข้อมูลของตนเองก็จะไม่มีสิทธิในการส่งข้อมูล

ปัญหาที่เกิดขึ้น

- เนื่องจากการเริ่มต้นการทำงานของคอมพิวเตอร์และอุปกรณ์ลูกข่ายที่ติดต่อ เริ่มการทำงานไม่พร้อมกัน ทำให้การสื่อสารข้อมูลเกิดความผิดพลาด

- ในการรับข้อมูลจากตัวประมวลผลหลัก มาทำการแสดงผลนั้นเกิดข้อผิดพลาดเนื่องจากความเร็วในการส่งข้อมูล ทำให้ไม่สามารถแยกข้อมูลออกเป็นเฟรมๆ เพื่อใช้แสดงผลได้

- รูปแบบการส่งข้อมูลแบบอนุกรมมาตรฐาน RS485 นี้ จำเป็นจะต้องออกแบบรูปแบบเฟรมข้อมูล ในการติดต่อกันแบบเครือข่าย เนื่องจาก รูปแบบการกระจายข้อมูลเป็นแบบ Broadcast ดังนั้น อุปกรณ์ทุกตัวที่ต่ออยู่จะได้รับข้อมูลทั้งของตัวเอง และ ของผู้อื่นที่ต่ออยู่ในเครือข่าย ทำให้อาจเกิดการรับส่งข้อมูลที่ผิดพลาดได้

- ในการออกแบบส่วนของวงจรควบคุมชุดตรวจจับ ในการโปรแกรมลงในไอซีนั้น เพื่อความยืดหยุ่นในการใช้งาน จำเป็นที่จะต้องออกแบบให้ใช้โปรแกรมในแต่ละจุดเหมือนกัน เพื่อสะดวกในการเปลี่ยนในกรณีที่ตัวใดตัวหนึ่งใช้งานไม่ได้

- ในการออกแบบวงจรชุดตรวจจับ แบบอินฟราเรด ปัญหาที่เกิดก็คือ ระดับแรงดันที่ได้จาก วงจรนั้นคือ 3 กับ 5 V ดังนั้น ทำให้ไม่สามารถนำ มาใช้เป็นอินพุทของ ไมโครคอนโทรเลอร์ได้ และ ระยะทางที่ได้นั้นใกล้มากทำให้ยากต่อการนำไปใช้

- การออกแบบโปรแกรมส่วนของชุดแสดงผลนั้น โดยใช้คอมพิวเตอร์ เป็นตัวแสดงผล จำเป็นที่จะต้องมีการเก็บข้อมูลเป็นสถิติการใช้งานที่จอดรถในแต่ละจุดด้วย เพื่อนำไปประยุคใช้ด้านธุรกิจต่อไป

แนวทางแก้ไข

- กำหนดให้มีการส่งสัญญาณเริ่มต้น (sync) จากคอมพิวเตอร์ไปตัวประมวลผลหลัก เพื่อกำหนดให้เริ่มต้นการทำงานพร้อมกัน

- ส่วนการแสดงผล ต้องมีการเปลี่ยนรูปแบบวิธีการรับข้อมูล โดยการใช้ การทำงานตามเหตุการณ์ที่เกิดขึ้น(command event) แทนการใช้ ไทม์เมอร์

- ในการสื่อสารในแต่ละอุปกรณ์ จำเป็นที่จะต้องมีการระบุเป้าหมายที่ต้องการ โดยการออกแบบเฟรมข้อมูล ซึ่งประกอบไปด้วย เฟรมส่วนหัวข้อมูล (Headcr) ใช้ในการระบุชนิดของอุปกรณ์ที่ต้องการติดต่อด้วย เฟรมแอดเดรส (Address) ใช้ในการระบุตำแหน่งของชุดควบคุมชุดตรวจจับ แต่ละจุด ซึ่ง ในแต่ละชุดจะประกอบไปด้วย วงจรตรวจจับแบบอินฟราเรด 4 ชุด โดยใช้เฟรมตำแหน่ง (Position) เป็นตัวระบุ และ เฟรมสถานะ(Status) ใช้ระบุว่า ชุดตรวจจับอินฟราเรดแต่ละจุด สามารถตรวจจับ รถที่มาจอดได้หรือไม่

- ได้ออกแบบให้มีการระบุแอดเดรสของ ชุดวงจรควบคุมชุดตรวจจับ โดยการใช้ ดิปสวิทซ์ เพื่อระบุว่าอยู่ที่จุดไหน โดยไม่จำเป็นต้องเขียนโปรแกรมในแต่ละจุดแตกต่างกัน เพื่อสะดวกในการ ซ่อมบำรุง

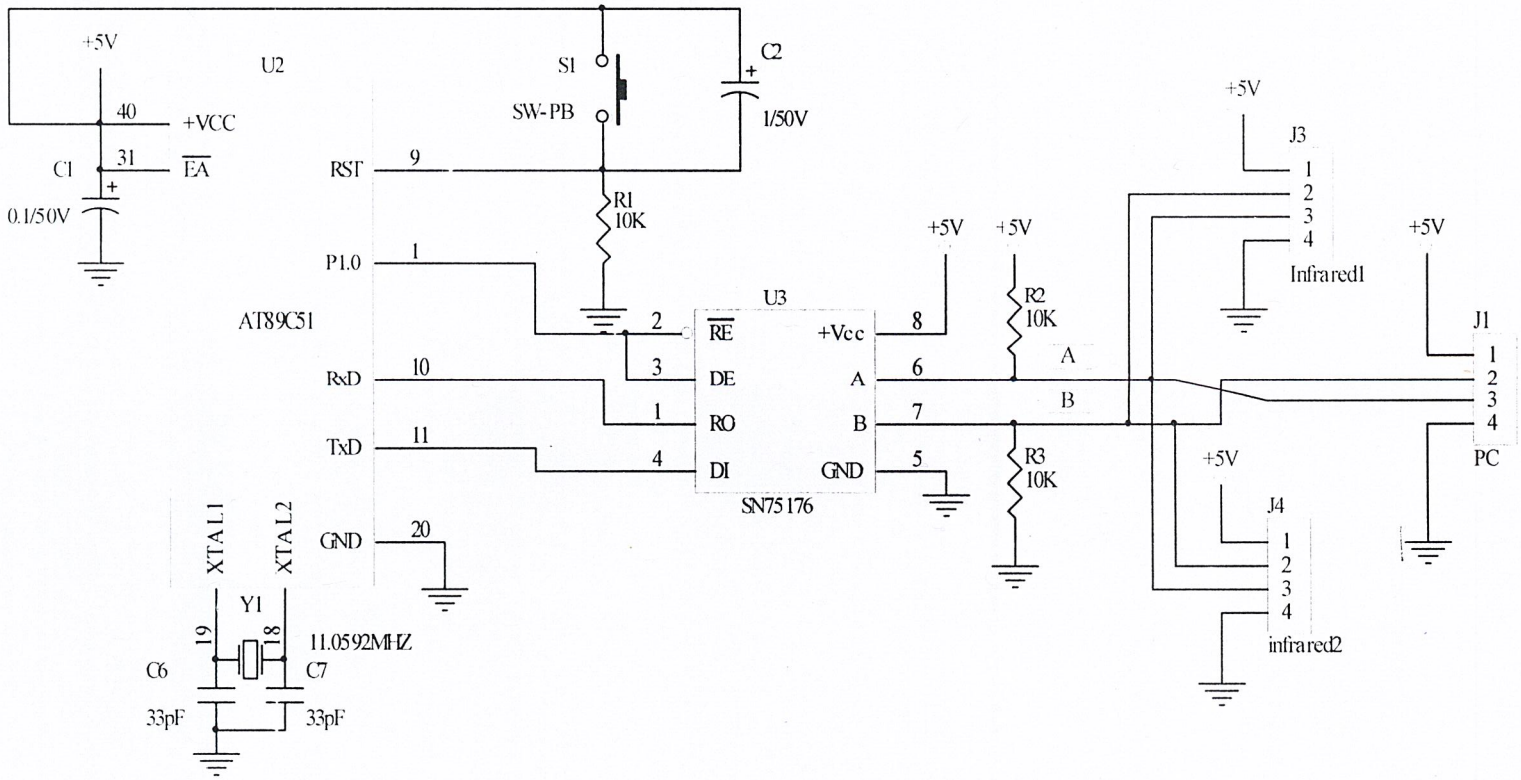
- ในส่วนของวงจรตรวจจับแบบอินฟราเรดนั้น ได้มีการออกแบบวงจรเปรียบเทียบระดับแรงดัน (Comparetor) ให้ ถ้ามีรถจอด จะส่งสถานะ ลอจิก 0 คือ 0V และ ถ้าไม่มีรถจอด จะส่งสถานะ ลอจิก 1 คือ 4-5V ทำให้สามารถส่งเข้าไปในตัวไมโครคอนโทรลเลอร์ได้ ส่วนปัญหาเรื่องระยะทางนั้น ได้แก้โดยออกแบบการจัดวางวงจรชุดตรวจจับไว้ที่กำแพง ทำให้สามารถใช้งานในระยะใกล้ได้

- ได้มีการออกแบบฐานข้อมูลเพื่อจัดเก็บวันที่ เวลา ในแต่ละจุด เพื่อ เก็บสถิติ การใช้งาน และมีการคำนวณอัตราการใช้บริการด้วย

บรรณานุกรม

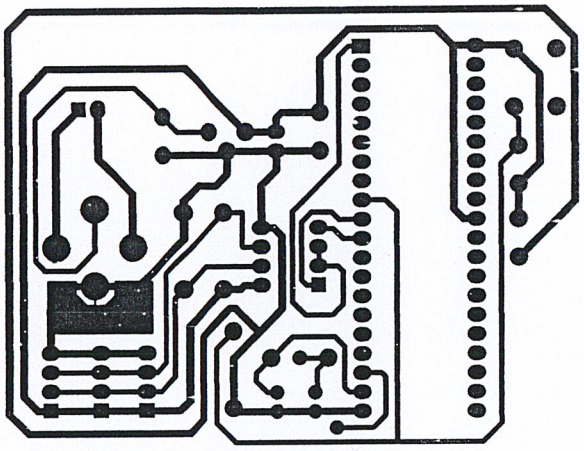
1. Jan Axelson, Serial Port Complete Programming and circuit for RS-232 and RS-485 Link and Networks , Lakeview Research.
2. ธีรวัฒน์ ประกอบผล, การพัฒนาโปรแกรมไมโครคอนโทรลเลอร์ด้วยภาษาซี, สำนักพิมพ์ ส.ส.ท., พิมพ์ครั้งที่ 1 ,2545
3. กฤดากร กล่อมการ, การสื่อสารข้อ,แผนกตำรา คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, พิมพ์ครั้งที่ 1, 2545.
4. สุรชัย เพิ่มสินทวี , ธเนศ สุวรรณผ่อง, การสื่อสารข้อมูลด้วยโมเด็ม , บริษัท ซีเอ็ดยูเคชั่น จำกัด มหาชน.

ภาคผนวก ก
รายละเอียดของวงจร

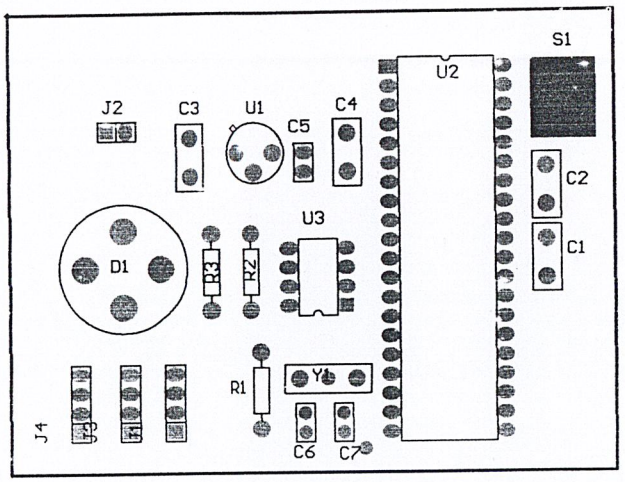


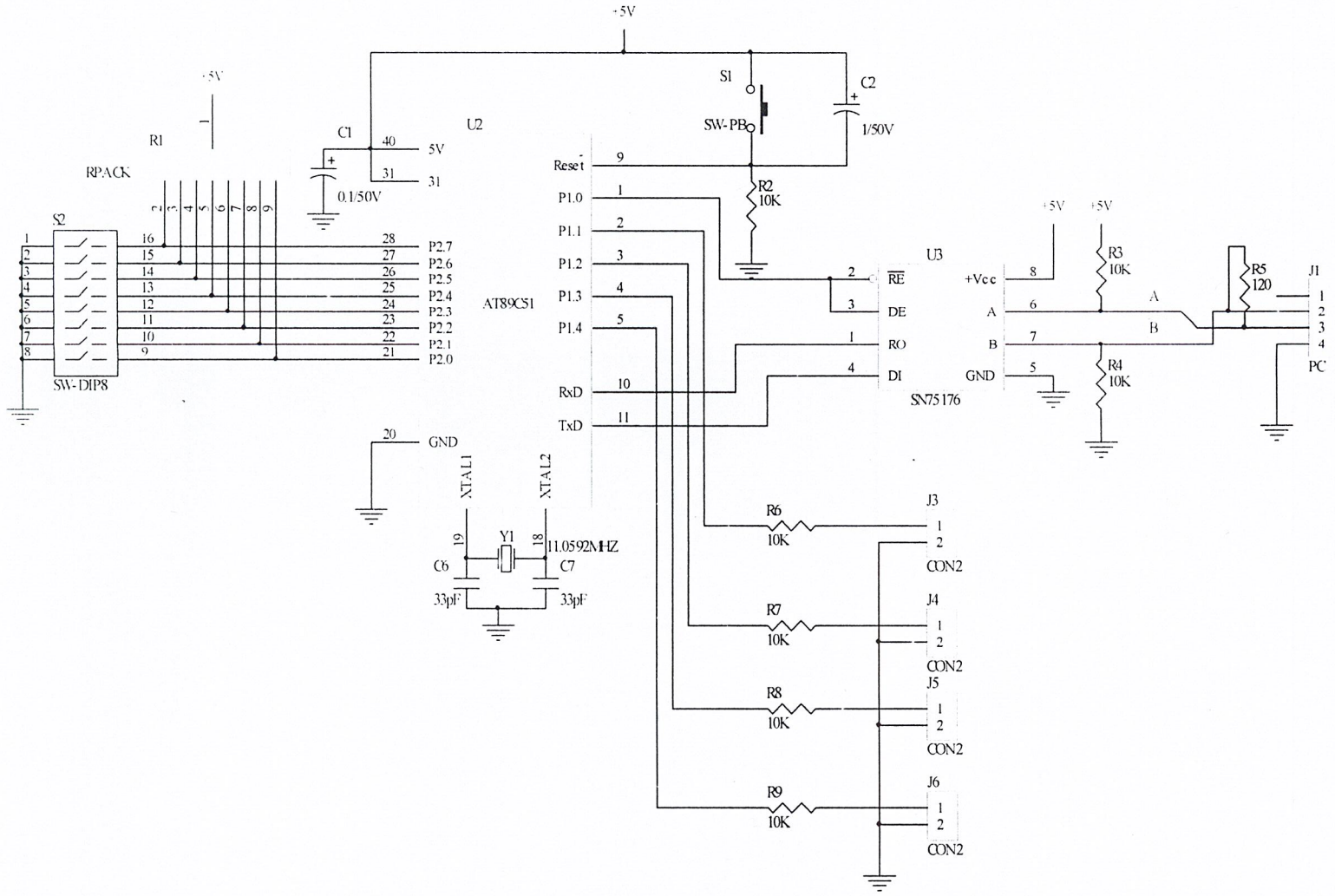
วงจรส่วนประมวลผลหลัก

ลายวงจรส่วนประมวลผลหลัก



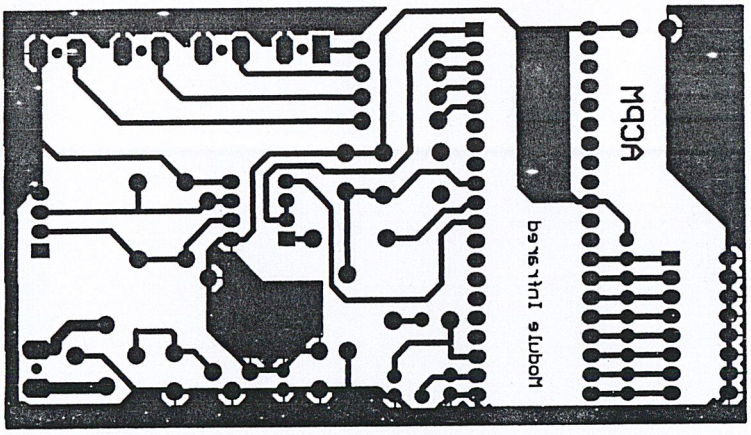
การวางอุปกรณ์วงจรส่วนประมวลผลหลัก



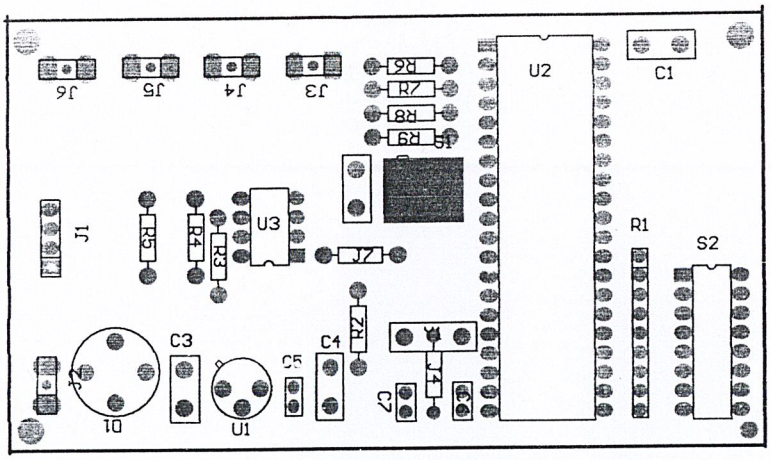


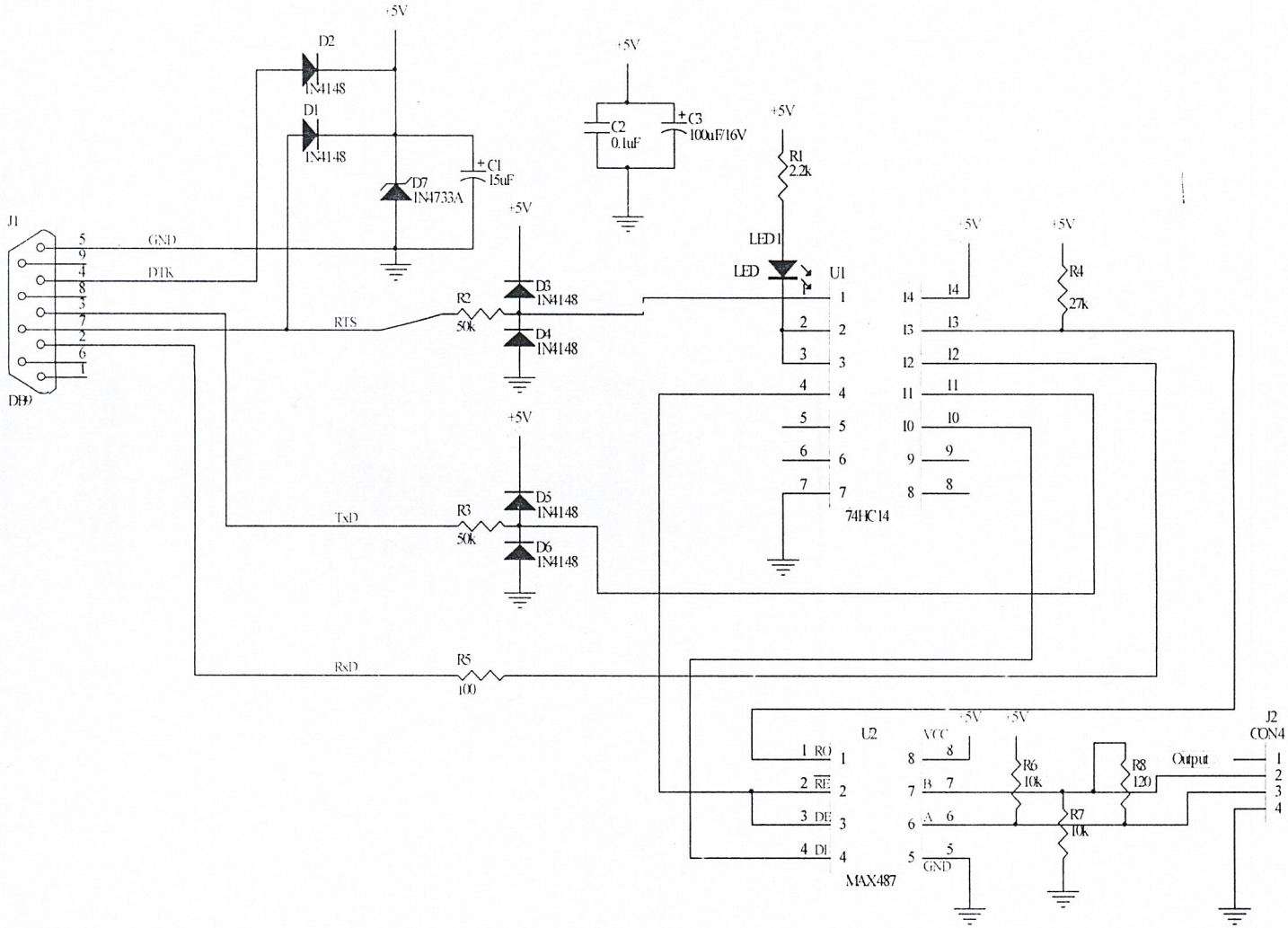
วงจรส่วนติดต่อชุดตรวจจับ

ลายวงจรส่วนติดต่อชุดตรวจจับ



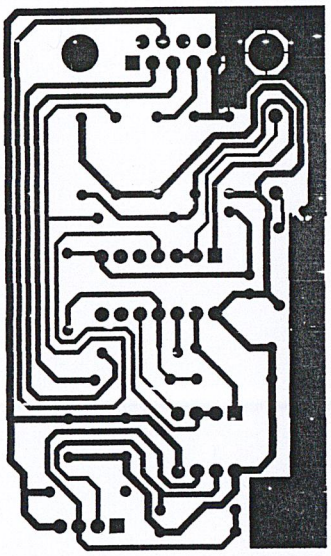
การวางอุปกรณ์วงจรส่วนติดต่อชุดตรวจจับ



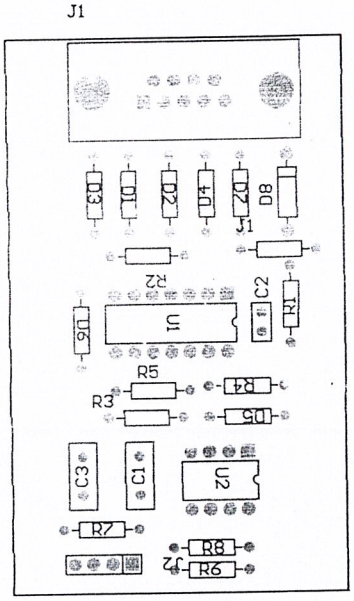


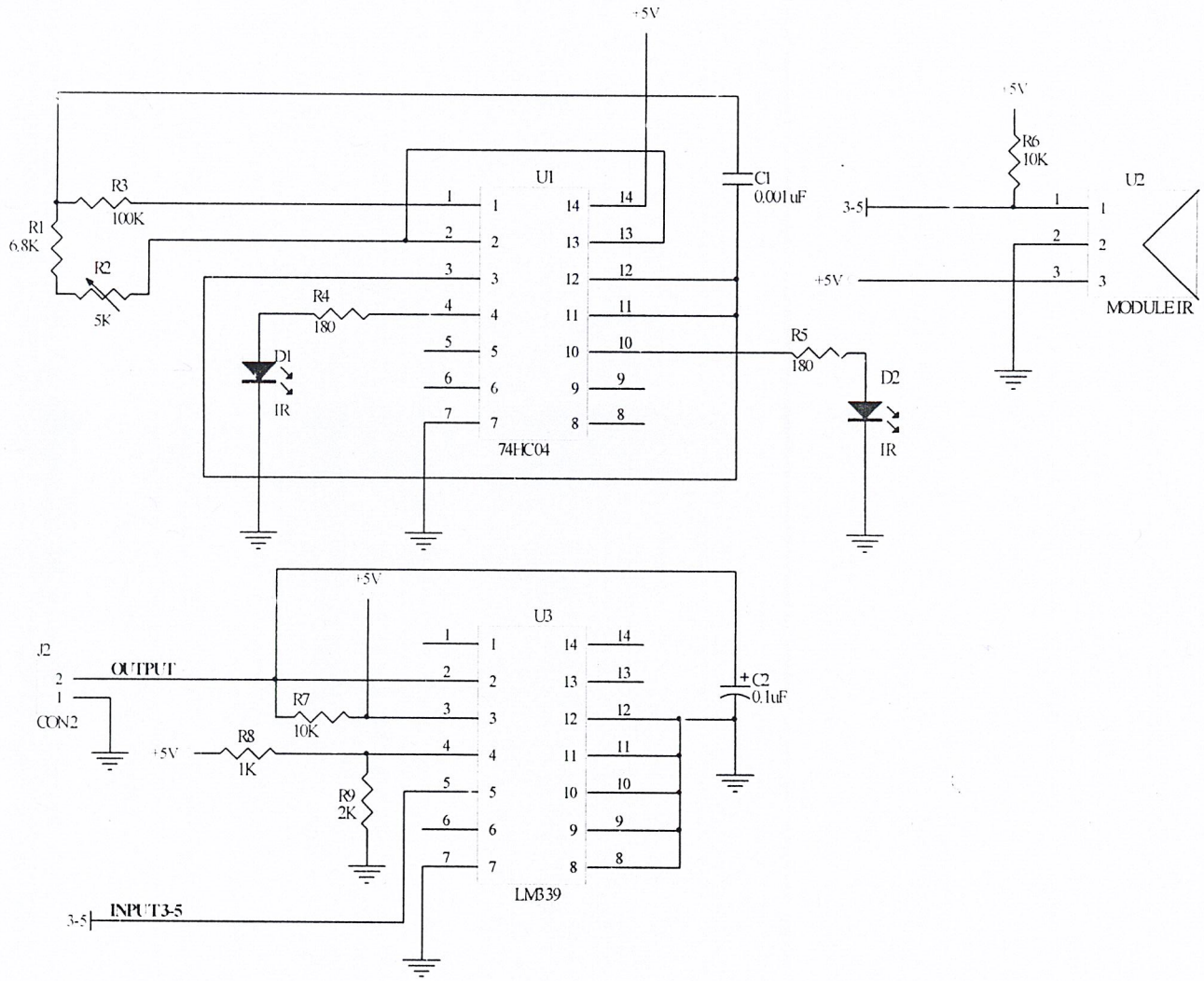
แสดงวงจรภาครับของส่วนแสดงผลที่มีการติดต่อแบบอนุกรม RS485

ลายวงจรภาครับของส่วนแสดงผลที่มีการติดต่อแบบอนุกรม RS485



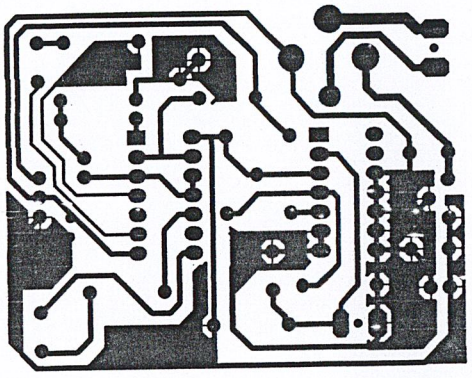
การวางอุปกรณ์วงจรภาครับของส่วนแสดงผลที่มีการติดต่อแบบอนุกรม RS485



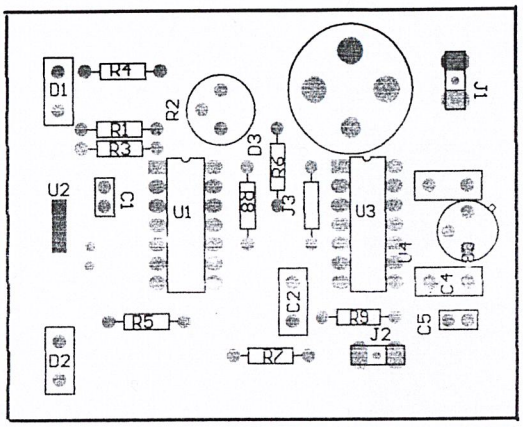


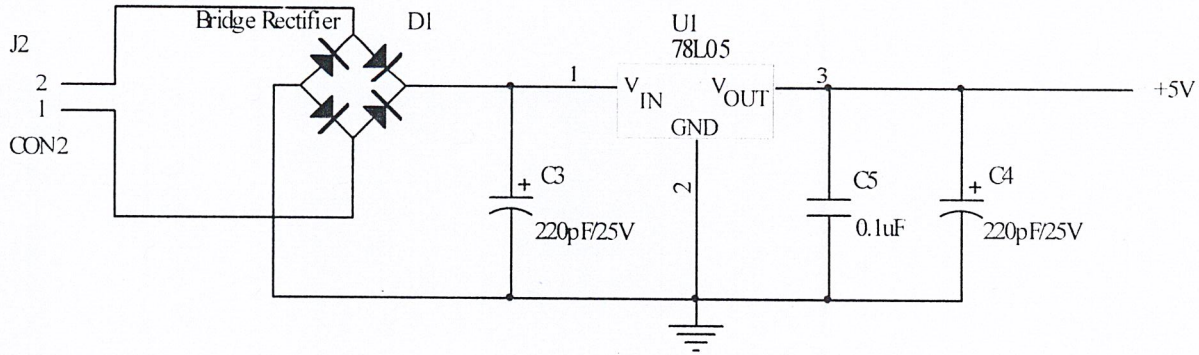
แสดงวงจรชุดตรวจจับ

ลายวงจรชุดตรวจจับ



การวางอุปกรณ์วงจรชุดตรวจจับ

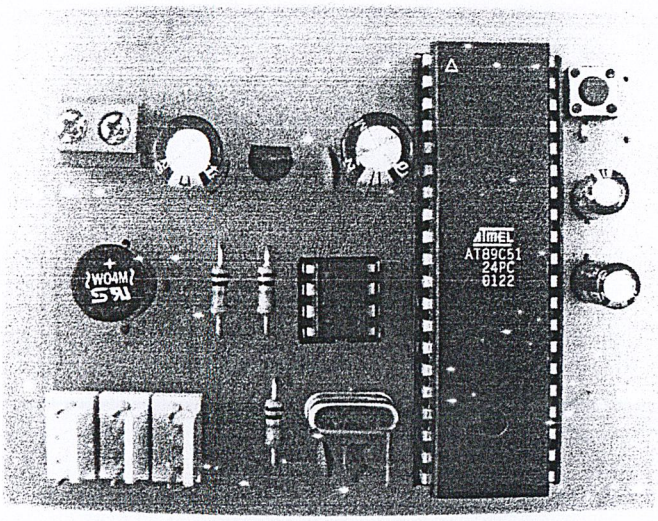




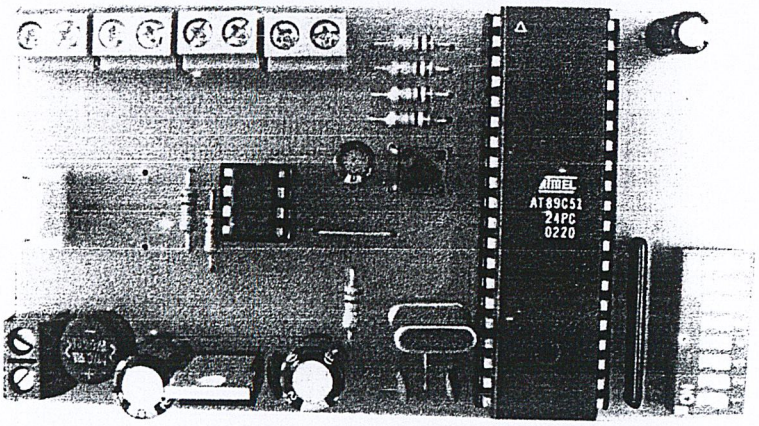
แสดงวงจรภาคจ่ายไฟ

ภาคผนวก ข
ภาพถ่ายโครงงาน

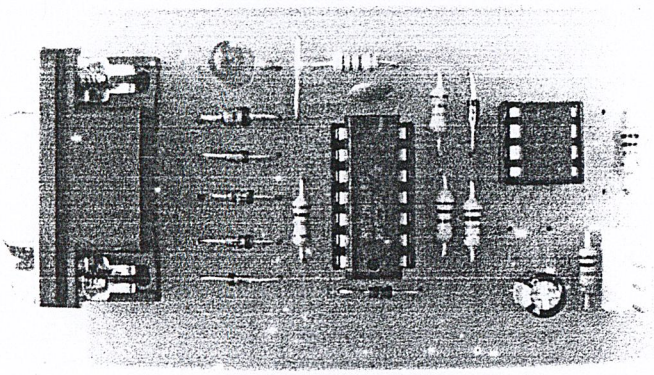
วงจรส่วนประมวลผลหลัก



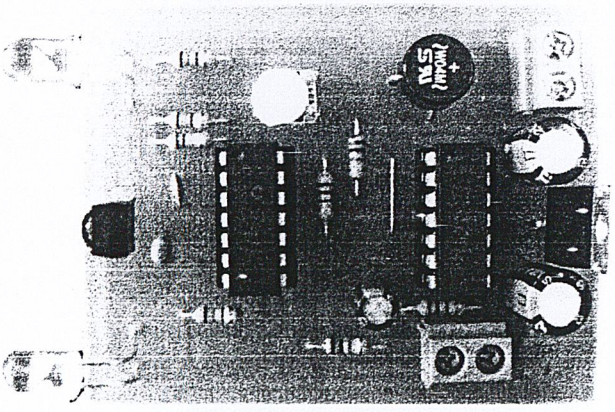
วงจรส่วนติดต่อชุดตรวจจับ



วงจรภาครับของส่วนแสดงผลที่มีการติดต่อแบบอนุกรม RS485



วงจรชุดตรวจจับ



ภาคผนวก ค
รายละเอียดของโปรแกรม

โปรแกรมภาษาซี

Module Accumulator

```
#pragma code
#include<reg51.h>
#define ON 1
#define OFF 0
sbit ENABLE = P1^0;     //enable RS-485 to TxD
void sender(unsigned char btest);
receive(unsigned char btest);
void delay(unsigned int time);
void main()
{       unsigned int counter=1;//delay
        unsigned char address,position,a,b;//address
        unsigned char header; // header A,R
        unsigned char status;// default = 'N'
        do{     ENABLE = OFF;  // Receive enable
                header = receive(0);
                address = receive(0);
                position = receive(0);
                status =  receive(0);
                switch(header)
                { case 'R': { header='A';
                a=0x02;
                b=0x04;
                } break;
                case 'A': header='A';break;
                default : break;
        }
        }while(header!='A');
        while(1)
        {       for(a=0x01;a<=0x02;a++)  // 1-2
                {   for(b=0x01;b<=0x04;b++) //1-4
                {
//==== send to module infrared====
        ENABLE = ON;
        sender('I'); delay(counter);
                sender(a); delay(counter);
                sender(b); delay(counter);
                sender(status); delay(counter);
//==== Receive From module infrared ======
do{     ENABLE = OFF;       // Receive enable
                header = receive(0);
                address = receive(0);
                position = receive(0);
                status =  receive(0);
                }while(header!='A');
//============Send to PC=================
                ENABLE = ON;    // Transmission Enable
                sender('C'); delay(counter);
                sender(address); delay(counter);
                sender(position); delay(counter);
                sender(status); delay(counter);
//======= Data Flow control========
do{     ENABLE = OFF;       // Receive enable
                header = receive(0);
                address = receive(0);
                position = receive(0);
                status =  receive(0);
switch(header)
                { case 'R': { header='A';
                a=0x02;
                b=0x04;
                  } break;
                case 'A': header='A';break;
                default : break;
                }
                }while(header!='A');
                                        }
                                }
                }
}
```

```
//*************** Function Send sender***************
void sender(unsigned char btest)
{       //unsigned char btest = 'k';    //byte test
        SCON = 0x52;      //sender port mode 1
        TMOD = 0x20;
        //Timer 1 mode2(buad rate generation)
        //TH1 = 0xFB;
        //19200 bit/sec,TH1=253D or -3
        TH1 = 0xFD;//9600 * 12 clock
        TR1 = 1;//Start timer 1
        while(!TI);
        TI = 0;
        SBUF = btest;
        return;
}
//*************** Function recieve sender***************
receive(unsigned char btest)
{       //unsigned char btest = 'k';    //byte test
        SCON = 0x52;
        //sender port mode 1
                TMOD = 0x20;
        //Timer 1 mode2(buad rate generation)
        //TH1 = 0xFB;
        //19200 bit/sec,TH1=253D or -3
                TH1 = 0xFD;//9600 * 12 clock
                TR1 = 1;//Start timer 1
                while(!RI);
                RI = 0;
                btest = SBUF;
                return(btest);
}
//*******Call Timer0 10 mSec for delay **************
void delay(unsigned int time) // timer 0 (TF0)
{       unsigned int i;
                for(i=0;i<=time;i++)
                {
                        TH0 = 0x88;
                        TL0 = 0x00;
                        TF0 = 0;
                        TR0 = 1;
                        while(TF0==0);
                        TR0 = 0;
                }
        return;
}
```

Module Client

```
#pragma code
#include<reg51.h>
#define ON 1
#define OFF 0
sbit ENABLE = P1^0;
//enable RS-485 to TxD
sbit sensor1 = P1^1;
sbit sensor2 = P1^2;
sbit sensor3 = P1^3;
sbit sensor4 = P1^4;
void sender(unsigned char btest);
// Func Send serial
receive(unsigned char btest);
// Func Receive serial
void delay(unsigned int time);
void main()
{       unsigned int counter=1;//delay
        unsigned char
header,address,position,status;
        unsigned char
dip;//address[Header,Dip,Number,Status]
        unsigned char infrared[6];
        P1 = 0x00; // clear port
        sensor1 = ON; // set port P1.1 to input port
```

```
        sensor2 = ON; // set port P1.2 to input port
        sensor3 = ON; // set port P1.3 to input port
        sensor4 = ON; // set port P1.4 to input port
        while(1)
        {
//=========== scan dip switch ========
        P2 = 0x0FF;//set input port
        dip = ~P2;//Dip No.
        infrared[1] = sensor1; //input
        infrared[2] = sensor2;
        infrared[3] = sensor3;
        infrared[4] = sensor4;

//=========== receive packet from ACC ======
        do{
        ENABLE = OFF;  // Receive enable
        header = receive(0);
        address = receive(0);
        position = receive(0);
        status =  receive(0);
        }while(header!='I');

//========== compare value Dip switch =======
        if(address==dip)
        {
        switch(position)// compare address of infrared
                {
        case 0x01: {    status = infrared[1];
                        break;    }
        case 0x02: {    status = infrared[2];
                        break,    }
        case 0x03: {    status = infrared[3];
                        break;    }
        case 0x04: {    status = infrared[4];
                        break;    }
                }
        if(status==0)
        {       status = 'Y';
        }
        if(status==1)
        {       status = 'N';
        }

//========= sender packet to ACC========
        ENABLE = ON;
        sender('A'); delay(counter);
        sender(address); delay(counter);
        sender(position); delay(counter);
        sender(status); delay(counter);
                }
        }
}

//*************** Function Send sender**************
void sender(unsigned char btest)
{       //unsigned char btest = 'k';    //byte test
        SCON = 0x52;      //sender port mode 1
        TMOD = 0x20;      //Timer 1 mode2
        //TH1 = 0xFB;
        //19200 bit/sec,TH1=253D or -3
        TH1 = 0xFD;//9600 * 12 clock
        TR1 = 1;//Start timer 1
        while(!TI);
        TI = 0;
        SBUF = btest;
        return;
}
```

```
//*************** Function recieve sender***************
receive(unsigned char btest)
{       //unsigned char btest = 'k';    //byte test
        SCON = 0x52;        //sender port mode 1
        TMOD = 0x20;        //Timer 1 mode2
        //TH1 = 0xFB;
        //19200 bit/sec,TH1=253D or -3
        TH1 = 0xFD;//9600 * 12 clock
        TR1 = 1;//Start timer 1
        while(!RI);
        RI = 0;
        btest = SBUF;
        return(btest);
}
//*******Call Timer0 10 mSec for delay **************
void delay(unsigned int time) // timer 0 (TF0)
{       unsigned int i;
        for(i=0;i<=time;i++)
        {
                TH0 = 0x88;
                TL0 = 0x00;
                TF0 = 0;
                TR0 = 1;
while(TF0==0);
                TR0 = 0;
        }
        return;
}
```

โปรแกรม Visual Basic

Form Main

```
Private Sub Form_Load()
Load Frm_Setting
Load Frm_ParkDP
Load Frm_Database
End Sub

Private Sub CmdDatabast_Click()
   Frm_Database.Show
End Sub

Private Sub CmdParkDP_Click()
   Frm_Setting.Show
End Sub

Private Sub CmdExit_Click()
End
End Sub
```

Form Setting

```
Private Sub CmdStart_Click()
  On Error GoTo Errlabel
   Frm_ParkDP.MSComm1.Settings = "9600,n,8,1"
   Frm_ParkDP.MSComm1.CommPort = Combo1.Text
   Frm_ParkDP.MSComm1.InputMode =
comInputModeText
   Frm_ParkDP.MSComm1.InputLen = 0
   Frm_ParkDP.MSComm1.RThreshold = 1
   Frm_ParkDP.MSComm1.DTREnable = True
   Frm_ParkDP.MSComm1.PortOpen = True
   Frm_ParkDP.Timer1.Enabled = True
   Frm_ParkDP.SSTab1.TabEnabled(1) = False
   counter = 0
   counter1 = 25
   counter2 = 25
   Frm_ParkDP.Show
   Unload Frm_Setting
Errlabel:
   App.Title = "ACPM_PROJECT2 TEST APP"
  If Err.Number = 380 Then MsgBox "Select comport  And
Select Baud rate", vbInformation, "8051 Control I/O"
   If Err.Number = 8002 Then MsgBox "Port this not
support. Please select new port ", vbInformation, "8051
Control I/O"
End Sub

Private Sub CmdClose_Click()
Frm_Main.Show
Unload Frm_Setting
End Sub
```

Form park display

```
Dim Buffer As String
Dim Status As String
Dim pause As Boolean
Dim Id As Variant
Dim Sh As Variant
Dim Temp As Integer
Dim textTemp As String
Dim mode As Boolean

Private Sub Form_Load()
txtConv.Text = ""
txtconv2.Text = ""
txtUse.Text = ""
mode = True
End Sub

Private Sub MSComm1_OnComm()
   Select Case MSComm1.CommEvent
      Case comEvReceive
      Dim Buffer  As String
      Dim mstrData(1 To 50) As String
        Buffer = MSComm1.Input
        TxtDataIN.Text = (TxtDataIN.Text & (Buffer))  '
Output
        If Buffer = "C" Then
          TxtDataIN.Text = "" & Buffer
          Hp3.Delay_ms (250)
          txtConv.Text = TxtDataIN.Text
        End If
     End Select
End Sub

Private Sub Timer3_Timer()
If mode Then
   mode = False
   send_sync
End If
End Sub

Private Sub txtConv_Change()
   Dim intCurrentLetter1 As Integer
   If Len(txtConv.Text) = 4 Then
      txtConv.ForeColor = vbBlack
      For intCurrentLetter1 = 1 To Len(txtConv.Text)
        txtconv2.Text = txtconv2.Text &
Hex(Asc(Mid(txtConv.Text, intCurrentLetter1, 1))) & " "
      Next intCurrentLetter1
   End If
End Sub

Private Sub txtConv2_Change()
  If Len(txtconv2.Text) = 10 Then
  Source2$ = txtconv2.Text
     searchPattern$ = "43"
     foundpos2% = InStr(Source2$, searchPattern$)
     texttemp2 = Mid$(txtconv2.Text, foundpos2%, 9)
  If Mid$(texttemp2, 1, 2) = "43" Then
        Header = "C"
        Address = "0" & Mid$(texttemp2, 4, 1) & "0" &
Mid$(texttemp2, 6, 1)
        If Mid$(texttemp2, 8, 2) = "4E" Then
        Status = "N"
        ElseIf Mid$(texttemp2, 8, 2) = "59" Then
        Status = "Y"
        End If
     txtUse.Text = Header & Address & Status
     TxtTest.Text = txtUse.Text
     txtConv.Text = ""
  End If
End If
End Sub
```

```
Private Sub TxtDataIN_Change()
If mode Then
   Timer3.Interval = 2000
   Timer3.Enabled = False
   Timer3.Enabled = True
ElseIf Not mode Then
   mode = True
   End If
End Sub

Private Sub TxtTest_Change()
Dim intCurrentLetter As Integer
If TxtTest.Text <> "" Then
Call splitstring(TxtTest.Text)
End If
End Sub

Private Sub splitstring(ByVal textinput As String)
Dim textTemp As String
If Len(textinput) = 6 Then
  Source$ = textinput
     searchPattern$ = "C"
     foundpos% = InStr(Source$, searchPattern$)
     textTemp = Mid$(textinput, foundpos%, 6)
     Index = (Mid$(textTemp, 2, 4))
     Status = Mid$(textTemp, 6, 1)
     If Status = "N" Then
     Show_statusN (Index)
        ElseIf Status = "Y" Then
        Show_statusY (Index)
     End If
  End If
End Sub

Private Sub show_label_car_status(ByVal CarStatus As
String, Index)
   If CarStatus = "N" Then
   lblcarstatus.Caption = " OUT "
   TxtStatus.Text = "OUT"
      ElseIf CarStatus = "Y" Then
      lblcarstatus.Caption = " IN "
      TxtStatus.Text = "IN"
   End If
End Sub

Private Sub send_sync()
txtConv.Text = ""
txtconv2.Text = ""
txtUse.Text = ""
If MSComm1.PortOpen = False Then
MSComm1.PortOpen = True
MSComm1.InputLen = 0
MSComm1.DTREnable = True
MSComm1.DTREnable = True
MSComm1.RTSEnable = True
Hp1.Delay_ms (110)
MSComm1.Output = "A"
Hp2.Delay_ms (110)
MSComm1.RTSEnable = False
ElseIf MSComm1.PortOpen = True Then
MSComm1.InputLen = 0
MSComm1.DTREnable = True
MSComm1.RTSEnable = True
Hp1.Delay_ms (110)
MSComm1.Output = "A"
Hp2.Delay_ms (110)
MSComm1.RTSEnable = False
End If
End Sub
```

```
Private Sub Show_statusN(ByVal Index As String)
  If Len(Index) = 3 Then
    TxtID.Text = "0" & Index
  Else: TxtID.Text = Index
End If
    Id = Iut(Mid$(TxtID.Text, 1, 2))
    Sh = Iut(Mid$(TxtID.Text, 3, 2))
    Text2.Text = Id & Sh
    datacount = Text2.Text
        If Sh = 1 Then
        If Shape1(Id).FillColor = &HFF& Then
        Shape1(Id).FillColor = &HE0E0E0
        TxtAddname.Text = Label1(Id).Caption
        TxtAddress.Text = TxtAddname.Text
        show_label_car_status ("N"), (Id)
        check_addressInc (datacount)
        TxtDataIN.Text = ""
        txtConv.Text = ""
        txtconv2.Text = ""
        send_sync
        End If
    ElseIf Sh = 2 Then
        If Shape2(Id).FillColor = &HFF& Then
        Shape2(Id).FillColor = &HE0E0E0
        TxtAddname.Text = Label2(Id).Caption
        TxtAddress.Text = TxtAddname.Text
        show_label_car_status ("N"), (Id)
        check_addressInc (datacount)
        TxtDataIN.Text = ""
        txtConv.Text = ""
        txtconv2.Text = ""
        send_sync
        End If
    ElseIf Sh = 3 Then
        If Shape3(Id).FillColor = &HFF& Then
        Shape3(Id).FillColor = &HE0E0E0
        TxtAddname.Text = Label3(Id).Caption
        TxtAddress.Text = TxtAddname.Text
        show_label_car_status ("N"), (Id)
        check_addressInc (datacount)
        TxtDataIN.Text = ""
        txtConv.Text = ""
        txtconv2.Text = ""
        send_sync
        End If
    ElseIf Sh = 4 Then
        If Shape4(Id).FillColor = &HFF& Then
        Shape4(Id).FillColor = &HE0E0E0
        TxtAddname.Text = Label4(Id).Caption
        TxtAddress.Text = TxtAddname.Text
        show_label_car_status ("N"), (Id)
        check_addressInc (datacount)
        TxtDataIN.Text = ""
        txtConv.Text = ""
        txtconv2.Text = ""
        send_sync
        End If
  End If
  TxtDataIN.Text = ""
  txtConv.Text = ""
  txtconv2.Text = ""
      send_sync
End Sub

Private Sub Show_statusY(ByVal Index As String)
Dim datacount As String
  If Len(Index) = 3 Then
    TxtID.Text = "0" & Index
  Else: TxtID.Text = Index
  End If
  Id = Iut(Mid$(TxtID.Text, 1, 2))
  Sh = Iut(Mid$(TxtID.Text, 3, 2))
```

```
Text2.Text = Id & Sh
datacount = Text2.Text
  If Sh = 1 Then
      If Shape1(Id).FillColor = &HE0E0E0 Then
      Shape1(Id).FillColor = &HFF&
      TxtAddname.Text = Label1(Id).Caption
      TxtAddress.Text = TxtAddname.Text
      show_label_car_status ("Y"), (Id)
      check_addressDec (datacount)
      TxtDataIN.Text = ""
      txtConv.Text = ""
      txtconv2.Text = ""
      send_sync
      End If
  ElseIf Sh = 2 Then
      If Shape2(Id).FillColor = &HE0E0E0 Then
      Shape2(Id).FillColor = &HFF&
      TxtAddname.Text = Label2(Id).Caption
      TxtAddress.Text = TxtAddname.Text
      show_label_car_status ("Y"), (Id)
      check_addressDec (datacount)
      TxtDataIN.Text = ""
      txtConv.Text = ""
      txtconv2.Text = ""
      send_sync
      End If
  ElseIf Sh = 3 Then
      If Shape3(Id).FillColor = &HE0E0E0 Then
      Shape3(Id).FillColor = &HFF&
      TxtAddname.Text = Label3(Id).Caption
      TxtAddress.Text = TxtAddname.Text
      show_label_car_status ("Y"), (Id)
      check_addressDec (datacount)
      TxtDataIN.Text = ""
      txtConv.Text = ""
      txtconv2.Text = ""
      send_sync
      End If
  ElseIf Sh = 4 Then
      If Shape4(Id).FillColor = &HE0E0E0 Then
      Shape4(Id).FillColor = &HFF&
      TxtAddname.Text = Label4(Id).Caption
      TxtAddress.Text = TxtAddname.Text
      show_label_car_status ("Y"), (Id)
      check_addressDec (datacount)
      TxtDataIN.Text = ""
      txtConv.Text = ""
      txtconv2.Text = ""
      send_sync
      End If
  End If
      TxtDataIN.Text = ""
      txtConv.Text = ""
      txtconv2.Text = ""
  send_sync
End Sub

Private Sub check_addressDec(ByVal datacount As
String)
  If datacount <= 71 Then
    counter1_dec
    HpSync.Delay_ms (95)
    checktab (1)
  ElseIf datacount > 71 Then
    counter2_dec
    HpSync.Delay_ms (95)
    checktab (2)
  End If
End Sub
```

```
Private Sub check_addressInc(ByVal datacount As String)
    If datacount <= 71 Then
        counter1_Inc
        HpSync.Delay_ms (95)
        checktab (1)
    ElseIf datacount > 71 Then
        counter2_Inc
        HpSync.Delay_ms (95)
        checktab (2)
    End If
End Sub

Private Sub counter1_Inc()
Dim strQry As String
    strQry = "'" & TxtAddname.Text & "'"
    Data1.Refresh
    Data1.RecordSource = "Select * From ticket where
Address like " & strQry & " and Status like 'IN';"
    Data1.Recordset.Edit
    Data1.Recordset.[TimeOut] = TxtTimeOUT.Text
    Data1.Recordset!Status = TxtStatus2.Text
    Data1.Recordset!DateOut = TxtDate.Text
    Data1.Recordset.Update
    Data1.Refresh
    counter1 = counter1 + 1
    Call counter_1
    Call counter_All
End Sub

Private Sub counter2_Inc()
Dim strQry2 As String
    strQry2 = "'" & TxtAddname.Text & "'"
    Data1.RecordSource = "Select * From ticket where
Address like " & strQry2 & " and Status like 'IN';"
    Data1.Recordset.Edit
    Data1.Recordset.[TimeOut] = TxtTimeOUT.Text
    Data1.Recordset!Status = TxtStatus2.Text
    Data1.Recordset!DateOut = TxtDate.Text
    Data1.Recordset.Update
    Data1.Refresh
    counter2 = counter2 + 1
    Call counter_2
    Call counter_All
End Sub

Private Sub counter1_dec()
    Data1.Recordset.AddNew
    Data1.Recordset!Address = TxtAddname.Text
    Data1.Recordset!TimeIN = TxtTimeIN.Text
    Data1.Recordset!Status = TxtStatus.Text
    Data1.Recordset!DateIN = TxtDate.Text
    Data1.Recordset.Update
    Data1.Refresh
    counter1 = counter1 - 1
    Call counter_1
    Call counter_All
End Sub

Private Sub counter2_dec()
    Data1.Recordset.AddNew
    Data1.Recordset!Address = TxtAddname.Text
    Data1.Recordset!TimeIN = TxtTimeIN.Text
    Data1.Recordset!Status = TxtStatus.Text
    Data1.Recordset!DateIN = TxtDate.Text
    Data1.Recordset.Update
    Data1.Refresh
    counter2 = counter2 - 1
    Call counter_2
    Call counter_All
End Sub
```

```
Private Sub counter_1()
Labelfreeslot1(0) = counter1
Labelfreeslot1(1) = counter1
End Sub

Private Sub counter_2()
Labelfreeslot2(0) = counter2
Labelfreeslot2(1) = counter2
End Sub

Private Sub counter_All()
counter = counter1 + counter2
LabelfreeslotAll(0) = counter
LabelfreeslotAll(1) = counter
End Sub

Private Sub checktab(ByVal key As Integer)
  If key = 1 Then
     SSTab1.TabEnabled(0) = True
     SSTab1.TabVisible(0) = True
     SSTab1.TabVisible(1) = False
     SSTab1.TabVisible(1) = True
  If counter1 = 0 Then
     SSTab1.TabVisible(0) = False
     SSTab1.TabVisible(1) = True
     SSTab1.TabVisible(0) = True
     SSTab1.TabEnabled(0) = True
     End If
  ElseIf key = 2 Then
     SSTab1.TabVisible(0) = False
     SSTab1.TabEnabled(1) = True
     SSTab1.TabVisible(1) = True
     SSTab1.TabVisible(0) = True
  If counter1 = 0 Then
     SSTab1.TabVisible(0) = False
     SSTab1.TabVisible(1) = True
        SSTab1.TabVisible(0) = True
        SSTab1.TabEnabled(0) = True
     End If
  End If
  End Sub

  Private Sub CmdTestFrm_Click()
  Frm_test.Show
  End Sub

  Private Sub Timer1_Timer()
    TxtDate.Text = Format(Date, "DD/MM/YYYY")
    LblDate.Caption = Format(Date, "DD/MM/YYYY")
    lblTime.Caption = Format(Time(), "HH:MM:SS")
    TxtTimeIN.Text = Format(Time(), "HH:MM:SS")
    lblStatus.Caption = "Port setting ::" &
  MSComm1.CommPort & "   Baud rate::" &
  MSComm1.Settings
  End Sub

  Private Sub Timer2_Timer()
  TxtTimeOUT.Text = Format(Time(), "HH:MM:SS")
  End Sub

  Function lut(ip As String) As Byte
  dh = Mid$(ip, 2, 1)
    If dh = "0" Then
    lut = 0
    Elseif dh = "1" Then lut = 1
    Elseif dh = "2" Then lut = 2
    Elseif dh = "3" Then lut = 3
    Elseif dh = "4" Then lut = 4
    Elseif dh = "5" Then lut = 5
    Elseif dh = "6" Then lut = 6
    Elseif dh = "7" Then lut = 7
    Elseif dh = "8" Then lut = 8
```

```
    Elself dh = "9" Then lut = 9
    Elself dh = "A" Then lut = 10
    Elself dh = "B" Then lut = 11
    Elself dh = "C" Then lut = 12
    Elself dh = "D" Then lut = 13
    Elself dh = "E" Then lut = 14
    Elself dh = "F" Then lut = 15
    Else: lut = 0
End If
dl = Mid$(ip, 1, 1)
    If dl = "0" Then
    lut = lut + 0
    Elself dl = "1" Then lut = lut + 16
    Elself dl = "2" Then lut = lut + 32
    Elself dl = "3" Then lut = lut + 48
    Elself dl = "4" Then lut = lut + 64
    Elself dl = "5" Then lut = lut + 80
    Elself dl = "6" Then lut = lut + 96
    Elself dl = "7" Then lut = lut + 112
    Elself dl = "8" Then lut = lut + 128
    Elself dl = "9" Then lut = lut + 144
    Elself dl = "A" Then lut = lut + 160
    Elself dl = "B" Then lut = lut + 176
    Elself dl = "C" Then lut = lut + 192
    Elself dl = "D" Then lut = lut + 208
    Elself dl = "E" Then lut = lut + 224
    Elself dl = "F" Then lut = lut + 240
    Else: lut = 0
End If
End Function

Private Sub CmdExit_Click()
MSComm1.RTSEnable = False
Unload Frm_ParkDP
End Sub
```

Form Database

```
Private Sub Form_Load()
Dim cost As Integer
Dim s1, s2, m1, m2, h1, h2, t, th As Integer
Adodc1.Enabled = False
End Sub

Private Sub CmdShowAll_Click()
Adodc1.RecordSource = "Select * From ticket where
Status like 'OUT';"
    Adodc1.Enabled = True
    DataGrid1.Enabled = True
    Adodc1.Refresh
End Sub

Private Sub Adodc1_MoveComplete(ByVal adReason As
ADODB.EventReasonEnum, ByVal pError As
ADODB.Error, adStatus As ADODB.EventStatusEnum,
ByVal pRecordset As ADODB.Recordset)
Dim Current As Integer
Dim All As Integer
    With Adodc1.Recordset
    Current = .AbsolutePosition
    All = .RecordCount
    End With
    Adodc1.Caption = "  Record " & Current & " / " & All
   lblStatusAll.Caption = "Data All " & All
End Sub

Private Sub lblID1_Change()
    If lblID1.Caption <> "" Then
        If Mid$(lblTimeOut.Caption, 5, 1) = ":" Then
        s1 = Mid$(lblTimeOut.Caption, 6, 1) &
Mid$(lblTimeOut.Caption, 7, 1)
        Else: s1 = Mid$(lblTimeOut.Caption, 7, 2)
    End If
```

```
    If Mid$(lblTimeIn.Caption, 5, 1) = ":" Then
       s2 = Mid$(lblTimeIn.Caption, 6, 1) &
Mid$(lblTimeIn.Caption, 7, 1)
       Else: s2 = Mid$(lblTimeIn.Caption, 7, 2)
    End If
    If Mid$(lblTimeOut.Caption, 2, 1) = ":" Then
       m1 = Mid$(lblTimeOut.Caption, 3, 1) &
Mid$(lblTimeOut.Caption, 4, 1)
       Else: m1 = Mid$(lblTimeOut.Caption, 4, 2)
    End If
    If Mid$(lblTimeIn.Caption, 2, 1) = ":" Then
       m2 = Mid$(lblTimeIn.Caption, 3, 1) &
Mid$(lblTimeIn.Caption, 4, 1)
       Else: m2 = Mid$(lblTimeIn.Caption, 4, 2)
    End If
    If Mid$(lblTimeOut.Caption, 2, 1) = ":" Then
       h1 = "0" & Mid$(lblTimeOut.Caption, 1, 1)
       Else: h1 = Mid$(lblTimeOut.Caption, 1, 2)
    End If
    If Mid$(lblTimeIn.Caption, 2, 1) = ":" Then
       h2 = "0" & Mid$(lblTimeIn.Caption, 1, 1)
    Else: h2 = Mid$(lblTimeIn.Caption, 1, 2)
    End If
    If s1 < s2 Then
       Text8.Text = (s1 + 60) - s2
       Else: Text8.Text = s1 - s2
    End If
    If m1 < m2 Then
       Text9.Text = (m1 + 60) - m2
       Else: Text9.Text = m1 - m2
    End If
    If h1 < h2 Then
       Text10.Text = (h1 + 24) - h2
       Else: Text10.Text = h1 - h2
    End If
    lblTotal.Caption = Text10.Text & ":" & Text9.Text & ":" &
Text8.Text  A = Mid$(lblTotal.Caption, 1, 2)
    Label3.Caption = A
       If Mid$(Label3.Caption, 2, 1) = ":" Then
          Label3.Caption = "0" & Mid$(Label3.Caption, 1, 1)
          Else: Label3.Caption = A
       End If

    Private Sub Txtsearch_KeyPress(KeyAscii As Integer)
       If TxtSearch.Text <> " " Then
          CmdSearch.Enabled = True
          CmdSearch2.Enabled = False
       End If
    End Sub

    Private Sub Txtsearch2_KeyPress(KeyAscii As Integer)
       If TxtSearch2.Text <> " " Then
          CmdSearch2.Enabled = True
          CmdSearch.Enabled = False
       End If
    End Sub

    Private Sub TxtTimeSearch1_KeyPress(KeyAscii As
Integer)
       If TxtSearch.Text <> " " Then
          CmdTimeSh.Enabled = True
          CmdSearch.Enabled = False
       End If
    End Sub

    Private Sub CmdSearch_Click()
    Dim strQry As String
       Adodc1.Enabled = True
       DataGrid1.Enabled = True
       DataGrid1.Visible = True
       strQry = "'" & TxtSearch.Text & "'"
```

```
    Adodc1.RecordSource = "Select * From ticket where        End Sub
Address like " & strQry & " and Status like 'OUT';"
    Adodc1.Refresh
    TxtSearch.Text = ""
    TxtSearch2.SetFocus
End Sub

Private Sub CmdSearch2_Click()
Dim strQry2 As String
    Adodc1.Enabled = True
    DataGrid1.Enabled = True
    DataGrid1.Visible = True
    strQry2 = "'" & TxtSearch2.Text & "'"
    Adodc1.RecordSource = "Select * From ticket where
TicketID like " & strQry2 & " and Status like 'OUT';"
   Adodc1.Refresh
   TxtSearch2.Text = ""
   TxtSearch2.SetFocus
End Sub

Private Sub CmdClose_Click()
    Frm_Main.Show
    Unload Frm_Database
```

ภาคผนวก ง
รายละเอียดของอุปกรณ์

- **Package Options Include Plastic Small-Outline (D), Shrink Small-Outline (DB), Thin Shrink Small-Outline (PW), and Ceramic Flat (W) Packages, Ceramic Chip Carriers (FK), and Standard Plastic (N) and Ceramic (J) 300-mil DIPs**

description

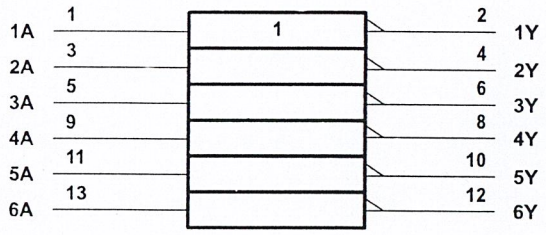
These devices contain six independent inverters. They perform the Boolean function $Y = \overline{A}$ in positive logic.

The SN54HC04 is characterized for operation over the full military temperature range of –55°C to 125°C. The SN74HC04 is characterized for operation from –40°C to 85°C.

FUNCTION TABLE
(each inverter)

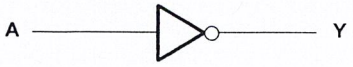
INPUT A	OUTPUT Y
H	L
L	H

logic symbol†

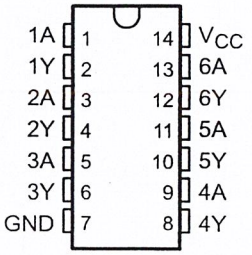


† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for the D, DB, J, N, PW, and W packages.

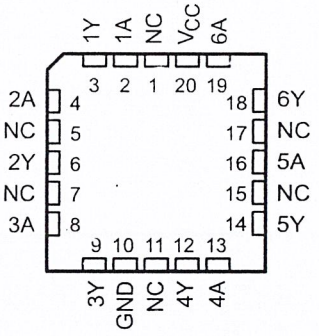
logic diagram (positive logic)



SN54HC04 . . . J OR W PACKAGE
SN74HC04 . . . D, DB, N, OR PW PACKAGE
(TOP VIEW)



SN54HC04 . . . FK PACKAGE
(TOP VIEW)



NC – No internal connection

- **Package Options Include Plastic Small-Outline (D), Thin Shrink Small-Outline (PW), and Ceramic Flat (W) Packages, Ceramic Chip Carriers (FK), and Standard Plastic (N) and Ceramic (J) 300-mil DIPs**

description

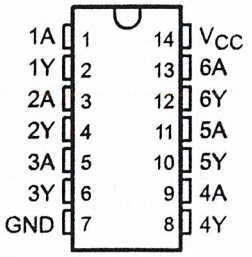
These Schmitt-trigger devices contain six independent inverters. They perform the Boolean function $Y = \overline{A}$ in positive logic.

The SN54HC14 is characterized for operation over the full military temperature range of –55°C to 125°C. The SN74HC14 is characterized for operation from –40°C to 85°C.

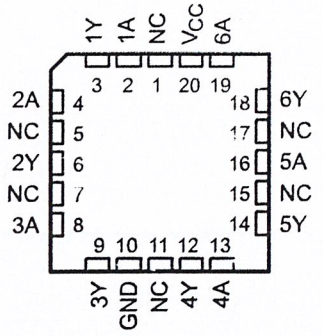
FUNCTION TABLE
(each inverter)

INPUT A	OUTPUT Y
H	L
L	H

SN54HC14 . . . J OR W PACKAGE
SN74HC14 . . . D, N, OR PW PACKAGE
(TOP VIEW)

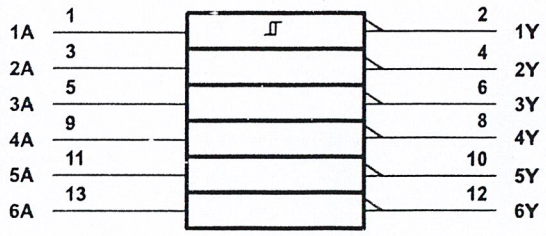


SN54HC14 . . . FK PACKAGE
(TOP VIEW)



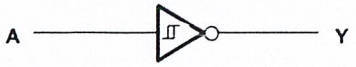
NC – No internal connection

logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.
Pin numbers shown are for the D, J, N, PW, and W packages.

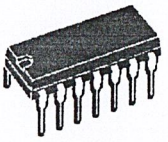
logic diagram (positive logic)



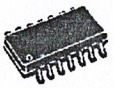
LM139,A
LM239,A-LM339,A

LOW POWER QUAD VOLTAGE COMPARATORS

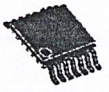
- WIDE SINGLE SUPPLY VOLTAGE RANGE OR DUAL SUPPLIES FOR ALL DEVICES : +2V TO +36V OR ±1V TO ±18V
- VERY LOW SUPPLY CURRENT (1.1mA) INDEPENDENT OF SUPPLY VOLTAGE (1.4mW/comparator at +5V)
- LOW INPUT BIAS CURRENT : 25nA TYP
- LOW INPUT OFFSET CURRENT : ±5nA TYP
- LOW INPUT OFFSET VOLTAGE : ±1mV TYP
- INPUT COMMON-MODE VOLTAGE RANGE INCLUDES GROUND
- LOW OUTPUT SATURATION VOLTAGE : 250mV TYP. (I_O = 4mA)
- DIFFERENTIAL INPUT VOLTAGE RANGE EQUAL TO THE SUPPLY VOLTAGE
- TTL, DTL, ECL, MOS, CMOS COMPATIBLE OUTPUTS



N
DIP14
(Plastic Package)



D
SO14
(Plastic Micropackage)



P
TSSOP14
(Thin Shrink Small Outline Package)

DESCRIPTION

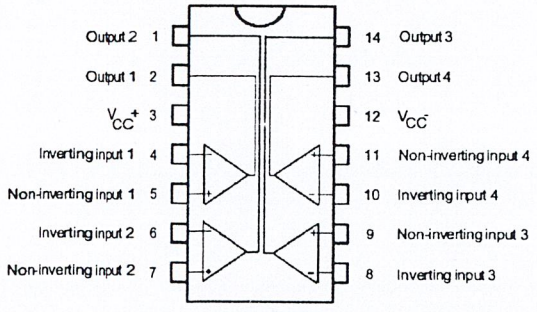
These devices consist of four independent precision voltage comparators with an offset voltage specifications as low as 2mV max for LM339A, LM239A and LM139A. All these comparators were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible.

These comparators also have a unique characteristic in that the input common-mode voltage range includes ground even though operated from a single power supply voltage.

ORDER CODES

Part Number	Temperature Range	Package		
		N	D	P
LM139,A	-55, +125°C	•	•	•
LM239,A	-40, +105°C	•	•	•
LM339,A	0, +70°C	•	•	•
Example : LM139AN				

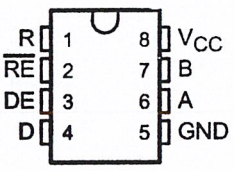
PIN CONNECTIONS (top view)



SLLS101B – JULY 1985 – REVISED JUNE 1999

- **Bidirectional Transceivers**
- **Meet or Exceed the Requirements of ANSI Standards TIA/EIA-422-B and TIA/EIA-485-A and ITU Recommendations V.11 and X.27**
- **Designed for Multipoint Transmission on Long Bus Lines in Noisy Environments**
- **3-State Driver and Receiver Outputs**
- **Individual Driver and Receiver Enables**
- **Wide Positive and Negative Input/Output Bus Voltage Ranges**
- **Driver Output Capability . . . ±60 mA Max**
- **Thermal Shutdown Protection**
- **Driver Positive and Negative Current Limiting**
- **Receiver Input Impedance . . . 12 kΩ Min**
- **Receiver Input Sensitivity . . . ±200 mV**
- **Receiver Input Hysteresis . . . 50 mV Typ**
- **Operate From Single 5-V Supply**

D OR P PACKAGE
(TOP VIEW)



description

The SN65176B and SN75176B differential bus transceivers are monolithic integrated circuits designed for bidirectional data communication on multipoint bus transmission lines. They are designed for balanced transmission lines and meet ANSI Standards TIA/EIA-422-B and TIA/EIA-485-A and ITU Recommendations V.11 and X.27.

The SN65176B and SN75176B combine a 3-state differential line driver and a differential input line receiver, both of which operate from a single 5-V power supply. The driver and receiver have active-high and active-low enables, respectively, that can be connected together externally to function as a direction control. The driver differential outputs and the receiver differential inputs are connected internally to form differential input/output (I/O) bus ports that are designed to offer minimum loading to the bus when the driver is disabled or V_{CC} = 0. These ports feature wide positive and negative common-mode voltage ranges, making the device suitable for party-line applications.

The driver is designed for up to 60 mA of sink or source current. The driver features positive and negative current limiting and thermal shutdown for protection from line-fault conditions. Thermal shutdown is designed to occur at a junction temperature of approximately 150°C. The receiver features a minimum input impedance of 12 kΩ, an input sensitivity of ±200 mV, and a typical input hysteresis of 50 mV.

The SN65176B and SN75176B can be used in transmission-line applications employing the SN75172 and SN75174 quadruple differential line drivers and SN75173 and SN75175 quadruple differential line receivers.

The SN65176B is characterized for operation from –40°C to 105°C and the SN75176B is characterized for operation from 0°C to 70°C.

Pin Description

PIN					NAME	FUNCTION
MAX481/MAX483/ MAX485/MAX487/ MAX1487		MAX488/ MAX490		MAX489/ MAX491		
DIP/SO	µMAX	DIP/SO	µMAX	DIP/SO		
1	3	2	4	2	RO	Receiver Output: If A > B by 200mV, RO will be high; If A < B by 200mV, RO will be low.
2	4	--	—	3	$\overline{RE}$	Receiver Output Enable. RO is enabled when $\overline{RE}$ is low; RO is high impedance when $\overline{RE}$ is high.
3	5	—	—	4	DE	Driver Output Enable. The driver outputs, Y and Z, are enabled by bringing DE high. They are high impedance when DE is low. If the driver outputs are enabled, the parts function as line drivers. While they are high impedance, they function as line receivers if $\overline{RE}$ is low.
4	6	3	5	5	DI	Driver Input. A low on DI forces output Y low and output Z high. Similarly, a high on DI forces output Y high and output Z low.
5	7	4	6	6, 7	GND	Ground
—	—	5	7	9	Y	Noninverting Driver Output
—	—	6	8	10	Z	Inverting Driver Output
6	8	—	—	--	A	Noninverting Receiver Input and Noninverting Driver Output
—	—	8	2	12	A	Noninverting Receiver Input
7	1	—	—	—	B	Inverting Receiver Input and Inverting Driver Output
—	—	7	1	11	B	Inverting Receiver Input
8	2	1	3	14	V_{CC}	Positive Supply: 4.75V ≤ V_{CC} ≤ 5.25V
—	—	—	—	1, 8, 13	N.C.	No Connect—not internally connected

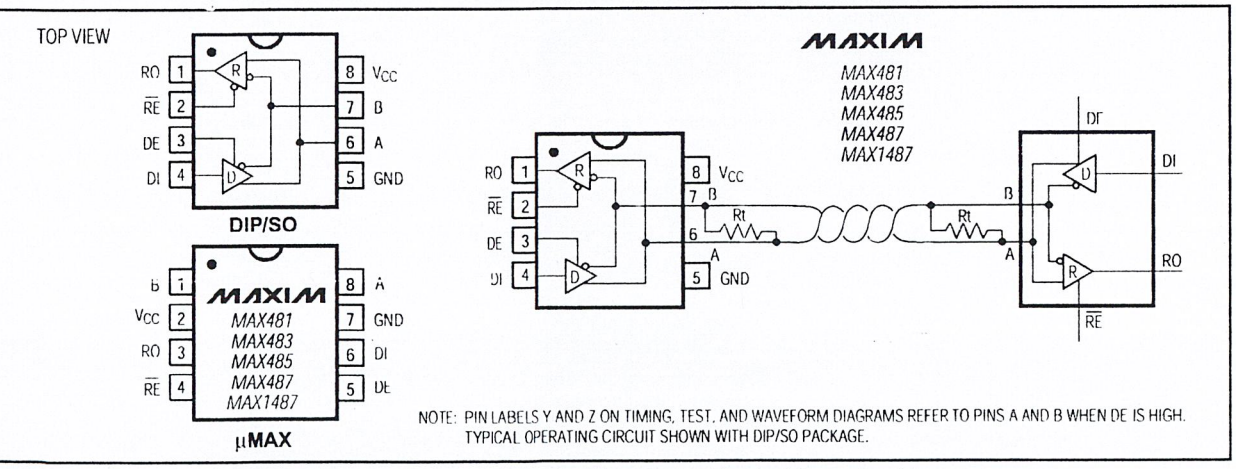


Figure 1. MAX481/MAX483/MAX485/MAX487/MAX1487 Pin Configuration and Typical Operating Circuit

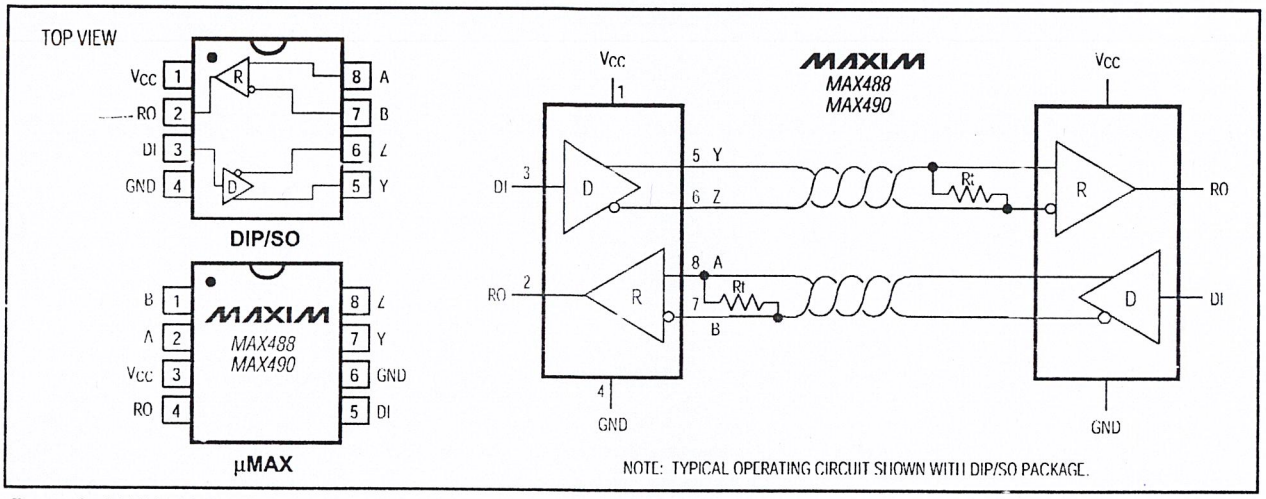


Figure 2. MAX488/MAX490 Pin Configuration and Typical Operating Circuit

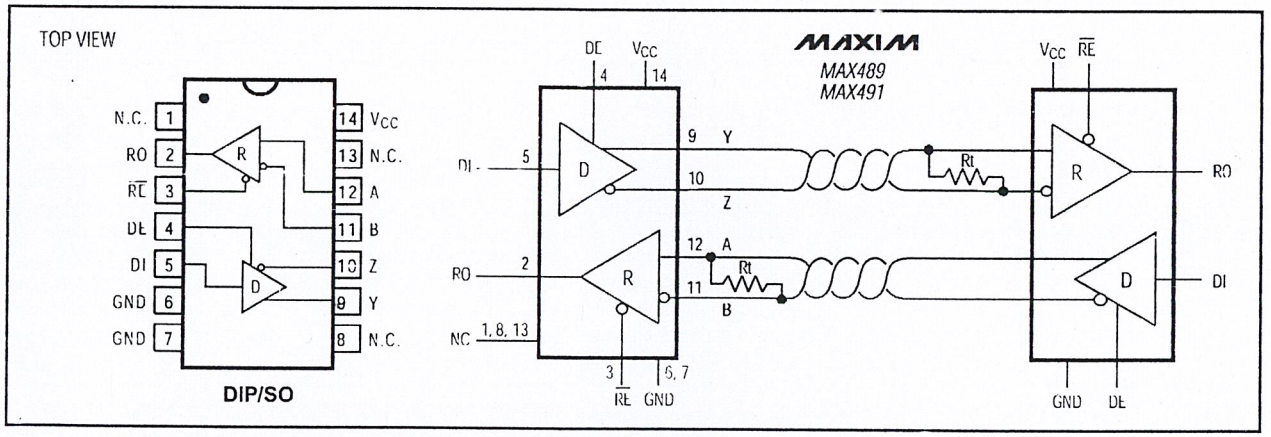


Figure 3. MAX489/MAX491 Pin Configuration and Typical Operating Circuit

Applications Information

The MAX481/MAX483/MAX485/MAX487–MAX491 and MAX1487 are low-power transceivers for RS-485 and RS-422 communications. The MAX481, MAX485, MAX490, MAX491, and MAX1487 can transmit and receive at data rates up to 2.5Mbps, while the MAX483, MAX487, MAX488, and MAX489 are specified for data rates up to 250kbps. The MAX488–MAX491 are full-duplex transceivers while the MAX481, MAX483, MAX485, MAX487, and MAX1487 are half-duplex. In addition, Driver Enable (DE) and Receiver Enable ($\overline{RE}$) pins are included on the MAX481, MAX483, MAX485, MAX487, MAX489, MAX491, and MAX1487. When disabled, the driver and receiver outputs are high impedance.

MAX487/MAX1487: 128 Transceivers on the Bus

The 48kΩ, 1/4-unit-load receiver input impedance of the MAX487 and MAX1487 allows up to 128 transceivers on a bus, compared to the 1-unit load (12kΩ input impedance) of standard RS-485 drivers (32 transceivers maximum). Any combination of MAX487/MAX1487 and other RS-485 transceivers with a total of 32 unit loads or less can be put on the bus. The MAX481/MAX483/MAX485 and MAX488–MAX491 have standard 12kΩ Receiver Input impedance.

Features

- Compatible with MCS-51™ Products
- 4K Bytes of In-System Reprogrammable Flash Memory
  - Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-Level Program Memory Lock
- 128 x 8-Bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-Bit Timer/Counters
- Six Interrupt Sources
- Programmable Serial Channel
- Low Power Idle and Power Down Modes

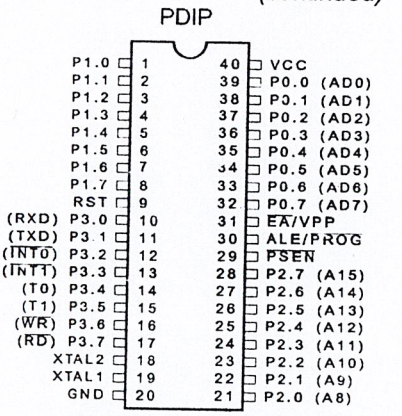
Description

The AT89C51 is a low-power, high-performance CMOS 8-bit microcomputer with 4K Bytes of Flash Programmable and Erasable Read Only Memory (PEROM). The device is manufactured using Atmel's high density nonvolatile memory technology and is compatible with the industry standard MCS-51™ instruction set and pinout. The on-chip Flash allows the program memory to be reprogrammed in-system or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with Flash on a monolithic chip, the Atmel AT89C51 is a powerful microcomputer which provides a highly flexible and cost effective solution to many embedded control applications.

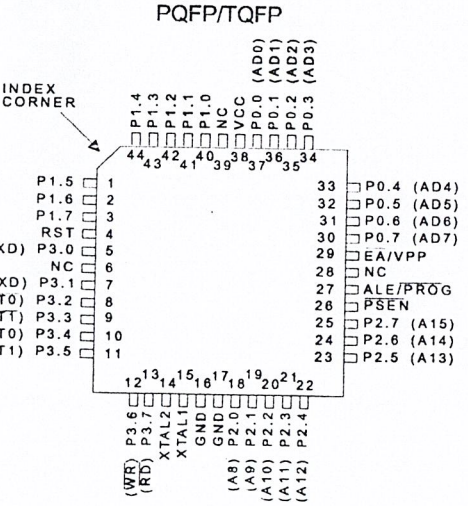
(continued)

Pin Configurations

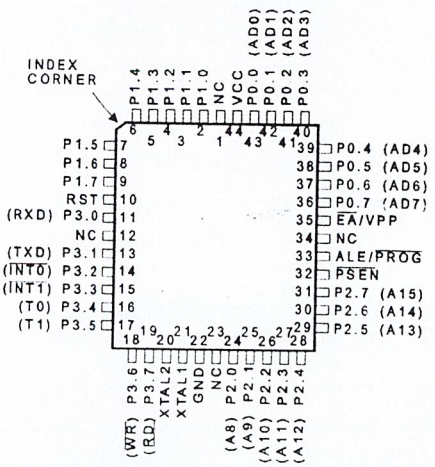
PDIP



PQFP/TQFP



PLCC



8-Bit Microcontroller with 4K Bytes Flash

AT89C51

0265F-A–12/97

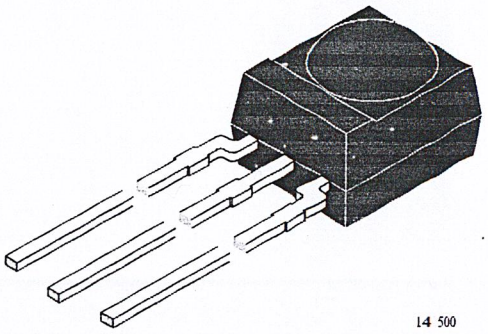
Photo Modules for PCM Remote Control Systems

Available types for different carrier frequencies

Type	fo	Type	fo
TSOP4830	30 kHz	TSOP4833	33 kHz
TSOP4836	36 kHz	TSOP4837	36.7 kHz
TSOP4838	38 kHz	TSOP4840	40 kHz
TSOP4856	56 kHz		

Description

The TSOP48.. – series are miniaturized receivers for infrared remote control systems. PIN diode and preamplifier are assembled on lead frame, the epoxy package is designed as IR filter.
The demodulated output signal can directly be decoded by a microprocessor. TSOP48.. is the standard IR remote control receiver series, supporting all major transmission codes.

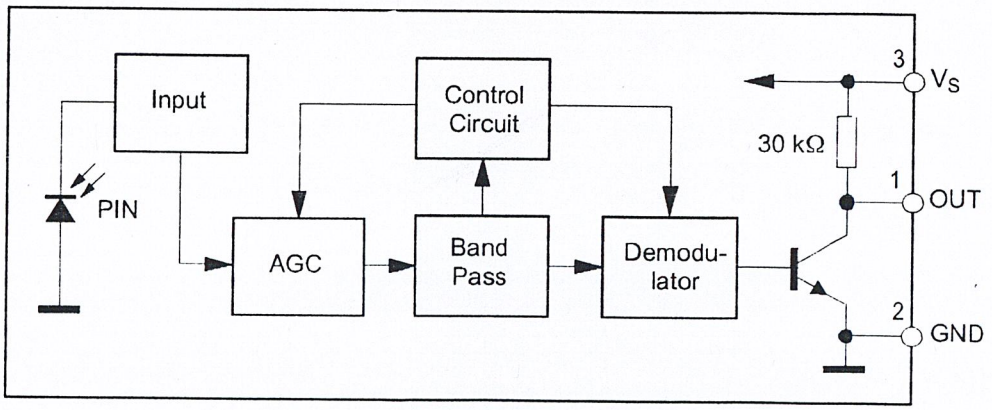


14 500

Features

- Photo detector and preamplifier in one package
- Internal filter for PCM frequency
- Improved shielding against electrical field disturbance
- TTL and CMOS compatibility
- Output active low
- Low power consumption
- High immunity against ambient light
- Continuous data transmission possible (800 bit/s)
- Suitable burst length ≥10 cycles/burst

Block Diagram



TAIWAN OASIS LED DATA SHEET (FOR INFRARED)

PART NO. : TOIR-50b94bCEa

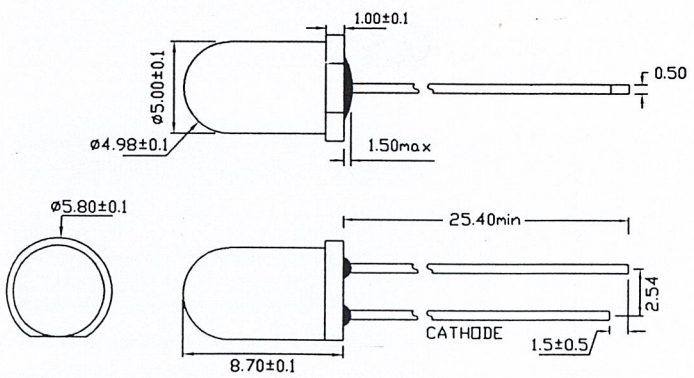
ABSOLUTE MAXIMUM RATINGS AT TA=25°C

PARAMETER	SYMBOL	DATA	UNIT
Forward Current	I_{FM}	100	mA
Peak Forward Current (duty=1:100, f=100KHZ)	I_{FP}	1000	mA
Reverse Voltage	V_R	6	V
Power Dissipation	P_D	150	mW
Operating Temperature Range		−25 to +85	°C
Storage Temperature Range		−30 to +85	°C
Lead Sold Temperature (1/10 Inch Below Seating Plane)		260°C for 3 sec.	

ELECTRICAL/OPTICAL CHARACTERISTICS AT TA=25°C

PARAMETER	SYMBOL	DATA	UNIT	TEST CONDITION
Radiated Output Power	$P_{O(Typ.)}$	12.0	mW	Distance: 10cm I_F=50mA Detector Area: 1cm²
Forward Voltage	V_F	TYP: 1.25 MAX: 1.45	V	I_F=20mA
Wavelength	λP	940	nm	I_F=20mA
Spectrum Width of Half Value	△λ	50	nm	I_F=20mA
Reverse Current	I_R	10	μA	V_R=5V
Full Viewing Angle	$2X\frac{1}{2}\theta$	25	°	I_F=20mA
Lens		Water Clear		
Radiation Material		GaAs/GaAs		

PACKAGE DIMENSIONS &
INTERNAL CIRCUIT DIAGRAM



DATE	01/10/01'	SCALE	2.5:1	TOLERANCE	±0.25 ANGLE ±6°	DRAWN	华明亮	CHECKED	
UNIT	M/M	SHEET NO.	1/2	DRAWING NO.	S-60b94bCEa-A	CUSTOMER		APPROVED	