สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

เครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ

AUTOMATIC SOFT-DRINK CHECKING AND BILLING SYSTEM

โดย

นายชัยยศ โชคคณาพิทักษ์

นายณัฐวุฒิ ทนุผล

เลขหมู่.....
เลขทะเบียน 42216
วัน, เดือน, ปี 15 พ.ค. 2545

.b.....
.i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

เครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ

AUTOMATIC SOFT-DRINK CHECKING AND BILLING SYSTEM

โดย

นายชัยยศ โชคคณาพิทักษ์ 41013048

นายณัฐวุฒิ ทนุผล 41013053

อาจารย์ที่ปรึกษา

รศ. สมยศ จุณณะปิยะ

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **เครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ**

AUTOMATIC SOFT-DRINK CHECKING AND BILLING SYSTEM

ผู้จัดทำ

1. นายชัยยศ โชคคณาพิทักษ์ **41013048**
2. นายณัฐวุฒิ ทนุผล **41013053**

อาจารย์ที่ปรึกษา

(รศ. สมยศ จุณณะปิยะ)

เครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ

AUTOMATIC SOFT-DRINK CHECKING AND BILLING SYSTEM

โดย นายชัยยศ โชคคณาพิทักษ์ 41013048
นายณัฐวุฒิ ทนุผล 41013053

อาจารย์ที่ปรึกษา รศ. สมยศ จุณณะปิยะ

บทคัดย่อ

Automatic Soft-drink Checking and Billing System เป็นเครื่องเช็คยอดและคิดราคาเครื่องดื่มที่ใส่ไว้ในตู้เย็นของห้องพักแต่ละห้องของโรงแรม ซึ่งจะเช็คชนิดและจำนวนของเครื่องดื่มที่ผู้เข้าพักได้นำออกไปดื่มในระหว่างที่เข้าพักในห้องนั้น ซึ่งใช้ MCS-51 เป็นตัวเก็บข้อมูลและควบคุม โดยใช้การทำงานแบบ Multi-processor โดยตัวแม่หรือ Master จะติดตั้งอยู่ที่เคาน์เตอร์ของโรงแรมและเชื่อมต่อเข้ากับคอมพิวเตอร์แสดงผลและคำนวณจำนวนเงิน และตัวลูกหรือ Slave จะติดตั้งอยู่ที่ตู้เย็นภายในห้องพักแต่ละห้องเพื่อเก็บข้อมูลของแต่ละห้องไว้ และจะส่งข้อมูลต่าง ๆ ไปยังตัวแม่เมื่อมีการเช็คเอาท์หรือต้องการที่จะดูจำนวนเครื่องดื่มที่ได้ใช้ไปแล้วภายในห้องนั้น สำหรับการติดต่อระหว่างตัวแม่กับตัวลูกที่ห้องต่าง ๆ นั้นจะใช้มาตรฐาน RS-485 เป็นตัวเชื่อมต่อ และระหว่างตัวแม่กับคอมพิวเตอร์จะใช้ RS-232

ABSTRACT

Automatic Soft-drink Checking and Billing System is the machine for checking number of Soft-drink in refrigerator of the room in the hotel. It checks type and the number of Soft-drink that tenant drink during stay in the hotel. MCS-51 is the controller and the accessories for saving data in multi-processor system. The master installed at the counter of the hotel and link with computer for showing the data and calculate the money. The slaves are installed in refrigerator at the room to save data and send data to master when tenant checkout or check toe number of Soft-drink. The master and slaves are communicated under RS-485 standard and RS-232 is the standard for the master and computer.

สารบัญ

เรื่อง	หน้า
บทที่ 1 บทนำ	1
บทที่ 2 ทฤษฎีหรือหลักการ	2
2.1 ทำไมต้องใช้ไมโครคอนโทรลเลอร์ MCS-51 ในการสร้างระบบควบคุม	2
2.2 การสื่อสารอนุกรมแบบอะซิงโครนัส	2
2.3 พอร์ตสื่อสารอนุกรมของไมโครคอนโทรลเลอร์	3
2.4 การอินเตอร์เฟส	3
2.4.1 มาตรฐาน RS-232	4
2.4.2 มาตรฐาน RS-485	6
2.5 การสื่อสารอนุกรมแบบยูเออาร์ที (UART)	9
2.5.1 ชนิดของยูเออาร์ที	10
2.6 โครงสร้างของไอซี MAX232	11
2.7 โครงสร้างของไอซี DS75176	11
2.8 การใช้งานพอร์ตอนุกรมของคอมพิวเตอร์ด้วยโปรแกรม Visual Basic	13
บทที่ 3 การออกแบบและการสร้าง	19
3.1 ส่วนของตัวแม่	19
3.2 ส่วนของตัวลูก	20
3.3 หลักการทำงานและวงจร	21
3.3.1 หลักการทำงานของตัวแม่	21
3.3.2 หลักการทำงานของตัวลูก	22
3.3.3 ส่วนวงจรการขับสัญญาณแบบ RS-232	26
3.3.4 ส่วนวงจรการขับสัญญาณแบบ RS-485	27
3.3.5 ส่วนวงจรรวมทั้งหมด	27
3.4 ส่วนตรวจสอบจำนวนเครื่องดื่ม	28
3.5 ส่วนของโปรแกรมที่ใช้ในการควบคุมการทำงานของเครื่อง	35
บทที่ 4 การทดลองและผลการทดลอง	37
4.1 การทดลองการส่งสัญญาณดิจิตอลแบบ UART	37
4.1.1 การทดลองส่งสัญญาณแบบ RS-232	37
4.1.2 การทดลองส่งสัญญาณแบบ RS-485	38
4.2 การทดลองการทำงานของเครื่องที่ออกแบบ	41
บทที่ 5 บทวิจารณ์และสรุป	44

สารบัญรูปภาพ

รูปภาพ	หน้า
รูปที่ 2.1 แสดงรูปแบบของข้อมูลอนุกรมอะซิงโครนัส	3
รูปที่ 2.2 แสดงการจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232	4
รูปที่ 2.3 แสดงการต่ออุปกรณ์ภายนอกกับพอร์ตอนุกรมของคอมพิวเตอร์	5
รูปที่ 2.4 แสดงลักษณะการต่อ RS-485 แบบขนาน	7
รูปที่ 2.5 แสดงตัวขับแบบ RS-485	8
รูปที่ 2.6 แสดงระดับโวลเตทของ RS-485	8
รูปที่ 2.7 บล็อกไดอะแกรมของยูเออาร์ที	9
รูปที่ 2.8 แสดงตัวไอซีและโครงสร้างภายในของ MAX232	11
รูปที่ 2.9 แสดงโครงสร้างภายในและตำแหน่งขาของตัวไอซี DS75176	12
รูปที่ 2.10 แสดงการเชื่อมต่อแบบหลายจุดของ RS-485	12
รูปที่ 3.1 แสดงบล็อกไดอะแกรมของระบบเครื่องเช็คจำนวนเครื่องดื่มอัตโนมัติ	19
รูปที่ 3.2 แสดงบล็อกไดอะแกรมของตัวแม่	20
รูปที่ 3.3 แสดงบล็อกไดอะแกรมของตัวลูก	21
รูปที่ 3.4 แสดงโฟลว์ชาร์ทการทำงานของตัวแม่	23
รูปที่ 3.5 แสดงโฟลว์ชาร์ทการทำงานของตัวลูก	24
รูปที่ 3.6 แสดงโฟลว์ชาร์ทการทำงานของคอมพิวเตอร์	25
รูปที่ 3.7 แสดงช่วงเวลาในการรับส่งข้อมูลทั้งระบบ	26
รูปที่ 3.8 แสดงส่วนของวงจรขับสัญญาณแบบ RS-232	26
รูปที่ 3.9 แสดงส่วนของวงจรขับสัญญาณแบบ RS-485	27
รูปที่ 3.10 แสดงช่องใส่เครื่องดื่มจำนวน 1 แถว (4 ช่อง) ที่ออกแบบขึ้น	28
รูปที่ 3.11 แสดงช่องใส่เครื่องดื่มจำนวน 24 ช่องที่ได้ออกแบบขึ้น	29
รูปที่ 3.12 แสดงช่องใส่เครื่องดื่มที่ทำเสร็จแล้ว	30
รูปที่ 3.13 แสดงช่องใส่เมื่อใส่กระป๋องเครื่องดื่มแล้ว	30
รูปที่ 3.14 แสดงภาพของตัวแม่เมื่อประกอบเสร็จ	31
รูปที่ 3.15 แสดงภาพของตัวลูกที่ประกอบเสร็จ	32
รูปที่ 3.16 แสดงวงจรรวมของตัวแม่	32
รูปที่ 3.17 แสดงวงจรส่วนควบคุมของตัวลูก	33
รูปที่ 3.18 แสดงวงจรส่วนของช่องใส่เครื่องดื่มของตัวลูก	34
รูปที่ 3.19 แสดงฟอร์มหลักของโปรแกรมเครื่องเช็คจำนวนเครื่องดื่มอัตโนมัติ	35
รูปที่ 4.1 แสดงระดับสัญญาณข้อมูลอนุกรมแบบ RS-232 ที่ทดลองส่ง	36
รูปที่ 4.2 แสดงระดับสัญญาณข้อมูลอนุกรมแบบ RS-232 ที่ทดลองส่ง	37

สารบัญรูปภาพ (ต่อ)

รูปภาพ	หน้า
รูปที่ 4.3 แสดงระดับแรงดันที่ขั้ว A กับกราวด์และ B กับกราวด์ ของ RS-485 (ข้อมูล 11001100)	39
รูปที่ 4.4 แสดงระดับแรงดันระหว่างขั้ว A กับ B (V_{AB})	39
รูปที่ 4.5 แสดงระดับแรงดันที่ขั้ว A กับกราวด์และ B กับกราวด์ ของ RS-485 (ข้อมูล 11110000)	40
รูปที่ 4.6 แสดงระดับแรงดันระหว่างขั้ว A กับ B (V_{AB})	40
รูปที่ 4.7 แสดงผลการทดลองการนับจำนวนและคิดเงิน	42

สารบัญตาราง

ตาราง หน้า

ตารางที่ 2.1 แสดงตำแหน่งและชื่อขาของ DB-9 และ DB-25 5

บทที่ 1
บทนำ

ความสะดวกสบาย และความรวดเร็วในการจัดการ ดูเสมือนว่าเป็นสิ่งที่มนุษย์ทุกคนต้องการทำให้เกิดเทคโนโลยีต่างๆมากมายขึ้นมา เพื่อตอบสนองความต้องการของมนุษย์ที่มีอยู่อย่างไม่จำกัดในชีวิตประจำวันจะเห็นได้ว่าจะมีเครื่องอำนวยความสะดวกมากมายที่ทำงานได้เองอัตโนมัติ เช่นเครื่องลงเวลา เครื่องวัดอุณหภูมิ เป็นต้น เราจะพบว่าเครื่องอัตโนมัติต่างๆ มีให้เห็นมากมายในเมืองที่เจริญ สำหรับเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ ซึ่งนำมาทำเป็นโครงงานนี้ จะมีให้เห็นตามโรงแรมต่างๆ เพื่อที่จะได้อำนวยความสะดวกสบายให้กับผู้มาใช้บริการ และง่ายกับการจัดการของเจ้าของกิจการ

เครื่องเช็คเครื่องดื่มอัตโนมัตินี้ ได้นำไมโครคอนโทรลเลอร์ตระกูล MCS-51 มาควบคุมการเช็คจำนวน โดยการใช้งานพอร์ตสื่อสารอนุกรมแบบมัลติโปรเซสเซอร์ซึ่งเป็นไมโครคอนโทรลเลอร์แบบ 8 บิต สามารถอ้างตำแหน่งได้ 256 ตำแหน่ง นั่นคือมีตัวแม่ (Master) 1 ตัว จะมีตัวลูก (Slave) ได้ 256 ตัว โดยที่ตัวลูกจะติดตั้งอยู่ที่ห้องพักในแต่ละห้องของโรงแรมเพื่อทำการตรวจสอบจำนวนของเครื่องดื่มที่มีอยู่ภายในตู้แช่แล้วส่งข้อมูลไปยังตัวแม่ที่ติดตั้งอยู่กับคอมพิวเตอร์ที่เคาน์เตอร์ของโรงแรม ซึ่งการสื่อสารข้อมูลระหว่างตัวไมโครคอนโทรลเลอร์กับคอมพิวเตอร์โดยใช้มาตรฐาน RS-232 ในการเชื่อมต่อกับคอมพิวเตอร์เพื่อแสดงผลที่จอภาพและเก็บข้อมูล ส่วนการติดต่อสื่อสารระหว่างตัวลูกกับตัวแม่ใช้มาตรฐาน RS-485 ซึ่งจะสามารถทราบจำนวนเครื่องดื่มที่ใช้ไป และสามารถคำนวณเงินได้

บทที่ 2
ทฤษฎีหรือหลักการ

2.1 การใช้ไมโครคอนโทรลเลอร์ MCS-51 ในการสร้างระบบควบคุม

ไมโครคอนโทรลเลอร์เป็นอุปกรณ์อิเล็กทรอนิกส์แบบหนึ่ง ซึ่งในตัวจะบรรจุหน่วยประมวลผล หน่วยคำนวณทางคณิตศาสตร์และลอจิก วงจรรับสัญญาณอินพุต วงจรขับสัญญาณแบบออกทางเอาท์พุต หน่วยความจำ วงจรกำเนิดสัญญาณนาฬิกา ซึ่งจะมีความแตกต่างกับไมโครโปรเซสเซอร์ เนื่องจากโครงสร้างของไมโครโปรเซสเซอร์จะประกอบด้วย หน่วยประมวลผลกลาง หน่วยคำนวณทางคณิตศาสตร์และลอจิก บัสข้อมูลและแอดเดรสสำหรับติดต่อกับหน่วยความจำภายนอก และวงจรกำเนิดสัญญาณนาฬิกา นั่นหมายความว่าการใช้งานไมโครโปรเซสเซอร์จะต้องเชื่อมต่อกับหน่วยความจำโปรแกรมภายนอก และถ้าหากมีความต้องการเชื่อมต่อกับอุปกรณ์อินพุตและเอาท์พุต จะต้องอาศัยอุปกรณ์ที่เรียกว่าไอซีขยายพอร์ต ทำให้การสร้างระบบควบคุมด้วยไมโครโปรเซสเซอร์จำเป็นต้องมีอุปกรณ์จำนวนมาก ทำให้ขนาดของระบบใหญ่พอสมควร ซึ่งถ้าใช้ไมโครคอนโทรลเลอร์ในการสร้างระบบควบคุม จะทำให้ขนาดของระบบรวมทั้งราคาก็จะมีขนาดลดลง เนื่องจากได้รวมส่วนของอินพุตและเอาท์พุตไว้ เพียงแต่เขียนโปรแกรมควบคุมลงในตัวไมโครคอนโทรลเลอร์หรืออีพรอม แล้วต่ออุปกรณ์ที่ใช้ในการสร้างวงจรกำเนิดสัญญาณนาฬิกา สุดท้ายเชื่อมต่ออุปกรณ์อินพุตเอาท์พุตที่ต้องการเข้ากับพอร์ตของไมโครคอนโทรลเลอร์เพียงเท่านั้นก็สามารถใช้งานในไมโครคอนโทรลเลอร์ได้แล้ว

เพราะฉะนั้นไมโครคอนโทรลเลอร์จึงเป็นอุปกรณ์ที่ใช้กันอย่างแพร่หลายในปัจจุบันสำหรับการสร้างระบบควบคุม โครงงานนี้จึงได้นำไมโครคอนโทรลเลอร์มาใช้ในการสร้างระบบควบคุมในการเช็คเครื่องดื่มอัตโนมัติ

2.2 การสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส

การสื่อสารข้อมูลแบบอะซิงโครนัสคือการรับและส่งข้อมูลไปในสายโดยไม่จำเป็นต้องมีสัญญาณนาฬิการ่วมด้วยเหมือนกับการรับส่งข้อมูลแบบซิงโครนัส แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาครับและภาคส่งให้มีค่าเท่ากัน ซึ่งเรียกสัญญาณนาฬิกาที่ใช้ในการกำหนดค่าให้ภาครับและภาคส่งนี้ว่าบอดเรต (Baud Rate) มีหน่วยเป็นบิตต่อวินาที

รูปแบบของข้อมูลที่ใช้ในการรับส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกันคือ

1. บิตเริ่มต้น ซึ่งจะมีขนาด 1 บิต
2. บิตข้อมูลอนุกรมจะมีขนาด 5, 6, 7 หรือ 8 บิต
3. บิตตรวจสอบพาริตี้จะมีขนาด 1 บิตหรือไม่มี
4. บิตปิดท้ายจะมีขนาด 1, 1.5 หรือ 2 บิต

รูปที่ 2.1 รูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส

จากรูปที่ 2.1 แสดงรูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส ซึ่งเมื่อไม่มีข้อมูลที่จะส่ง ขาข้อมูลจะมีสถานะลอจิก "1" ซึ่งจะเรียกสถานะนี้ว่าสถานะหยุดรอ การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขาข้อมูลมีลอจิก "0" ด้วยช่วงระยะเวลา 1 บิต ซึ่งจะเรียกบิตนี้ว่าบิตเริ่มต้นจากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มต้นจากบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อน ซึ่งข้อมูลในไบต์ที่จะส่งอาจจะมีจำนวนบิต 5, 6, 7 หรือ 8 บิตก็ได้ จากนั้นตามด้วยบิตพาริตี้ ซึ่งใช้เพื่อตรวจสอบความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล บิตสุดท้ายที่จะส่งคือบิตสุดท้าย ซึ่งจะให้ขาข้อมูลมีสถานะลอจิก "1" อีกครั้งด้วยระยะเวลาอย่างน้อย 1 บิต 1.5 บิต หรือ 2 บิตเพื่อเป็นการแสดงว่าสิ้นสุดข้อมูลแล้ว

2.3 พอร์ตสื่อสารข้อมูลอนุกรมของไมโครคอนโทรลเลอร์

พอร์ตสื่อสารข้อมูลอนุกรมในไมโครคอนโทรลเลอร์ MCS-51 มีการทำงานแบบฟูลดูเพล็กซ์ (Full Duplex) คือสามารถรับและส่งข้อมูลได้ในเวลาเดียวกัน และมีรีจีสเตอร์ที่เกี่ยวข้องกับการทำงานอยู่ด้วยกัน 2 ตัว คือ รีจีสเตอร์บัฟเฟอร์ของพอร์ตอนุกรม (Serial Data Buffer Register : SBUF) เมื่อมีการเขียนข้อมูลมายังรีจีสเตอร์ ข้อมูลนั้นจะถูกส่งต่อไปยังบัฟเฟอร์สำหรับส่งข้อมูล เพื่อส่งออกจากไมโครคอนโทรลเลอร์ผ่านทางขา TxD หรือขา P3.1 และในกรณีที่มีการอ่านข้อมูลจากรีจีสเตอร์ ข้อมูลจะถูกส่งผ่านไปยังรีจีสเตอร์บัฟเฟอร์สำหรับรับข้อมูลเพื่อส่งต่อไปยังไมโครคอนโทรลเลอร์ สำหรับการรับข้อมูลจากภายนอกนั้นจะผ่านมาทางขา RxD หรือ P3.0 และรีจีสเตอร์อีกตัวคือรีจีสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม (Serial Port Control Register : SCON) เป็นรีจีสเตอร์ขนาด 8 บิตสามารถเข้าถึงได้ในระดับบิต ใช้ในการเลือกโหมดการทำงานของพอร์ตอนุกรม โดยจะเลือกโหมดการทำงานได้ 4 โหมด

2.4 การอินเตอร์เฟส (Interface)

การอินเตอร์เฟสระหว่างคอมพิวเตอร์กับตัวไมโครคอนโทรลเลอร์จะใช้มาตรฐาน RS-232 ซึ่งจะกำหนดจำนวนสายของสัญญาณ รูปแบบของสัญญาณไฟฟ้า ระดับแรงดันไฟฟ้าที่ส่งระหว่างคอมพิวเตอร์กับตัวไมโครคอนโทรลเลอร์ ซึ่งจะเรียกพอร์ตของคอมพิวเตอร์ว่า DTE (Data Terminal Equipment) และเรียกตัวไมโครคอนโทรลเลอร์ว่า DCE (Data Circuit-terminating Equipment) และการอินเตอร์เฟสระหว่างตัวแม่กับตัวลูก จะใช้มาตรฐาน RS-485

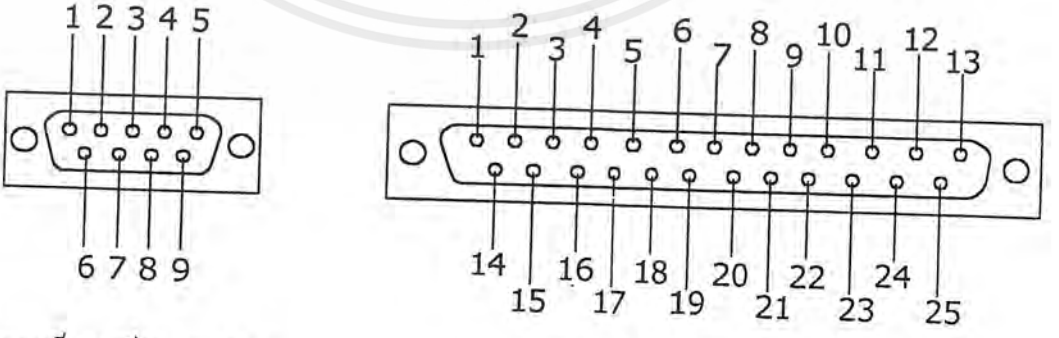
2.4.1 มาตรฐาน RS-232

มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้ในการส่งข้อมูลอนุกรมแบบอะซิงโครนัส 2 ทิศทาง โดยมาตรฐาน RS-232 ในอดีตนั้นถูกออกแบบมาเพื่อการส่งผ่านข้อมูลจากคอมพิวเตอร์ไปยังโมเด็มเพียงอย่างเดียว เพื่อที่จะนำข้อมูลจากโมเด็มนี้สื่อสารผ่านสายโทรศัพท์ไปยังคอมพิวเตอร์อีกชุดซึ่งอยู่ห่างไกลกัน โดยคณะกรรมการที่เรียกว่า สมาคมอุตสาหกรรมอิเล็กทรอนิกส์ (Electronic Industries Association : EIA) ได้วางมาตรฐานที่มีชื่อเรียกกันว่า EIA RS-232 มาตรฐานนี้ในช่วงแรกจะใช้คอนเน็กเตอร์เป็นแบบ DB-25 โดยกำหนดความยาวของสายสัญญาณไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ -3 ถึง -15 โวลท์ แสดงว่ามีข้อมูล (Mark) และ +3 ถึง +15 แสดงว่าเป็นช่องว่าง (Space)

มาตรฐาน RS-232 ได้กำหนดรูปแบบของอุปกรณ์เชื่อมต่อข้อมูลไว้ว่าอุปกรณ์ DTE จะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ ซึ่งมีความสามารถในการสร้างบิตข้อมูลอนุกรมได้ ส่วนอุปกรณ์ DCE จะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจาก DTE เท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS-232 ข้อแตกต่างของอุปกรณ์ DTE และอุปกรณ์ DCE อย่างหนึ่งที่เห็นได้ชัด คือคอนเน็กเตอร์ของ DTE จะเป็นตัวผู้ ส่วนคอนเน็กเตอร์ของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นแบบ DTE ส่วนคอนเน็กเตอร์ที่อยู่ที่โมเด็มจะเป็นแบบ DCE

สำหรับการใช้งานบนคอมพิวเตอร์ พอร์ตอนุกรม RS-232 มักถูกใช้เชื่อมต่อกับโมเด็มหรือเมาส์ โดยสามารถรับส่งข้อมูลได้ที่ความยาวของสายสัญญาณสูงสุดถึง 20 เมตร

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



ก) คอนเน็กเตอร์อนุกรม 9 ขา หรือแบบ DB-9

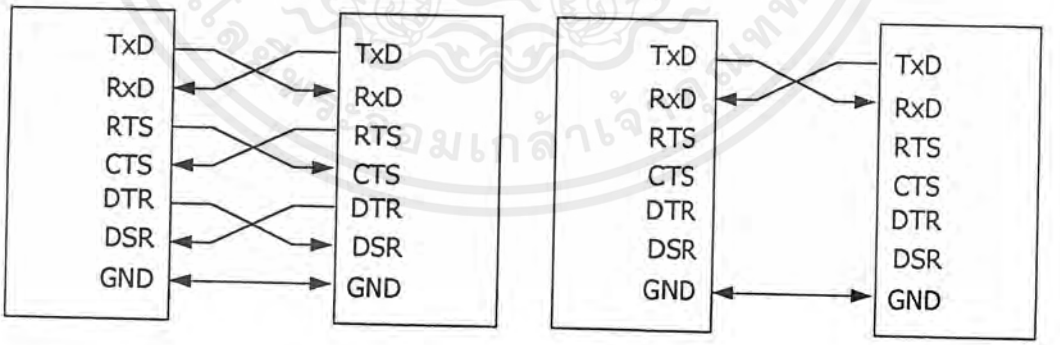
ข) คอนเน็กเตอร์อนุกรม 25 ขา หรือแบบ DB-25

รูปที่ 2.2 แสดงการจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232 แบบ DB-9 และ DB-25

ตารางที่ 2.1 แสดงตำแหน่งและชื่อขาของ DB-9 และ DB-25

คอนเนกเตอร์ DB-9	คอนเนกเตอร์ DB-25	ชื่อของสายสัญญาณ	ชนิดของสายสัญญาณ
1	8	DCD	อินพุต
2	3	RxD	อินพุต
3	2	TxD	เอาท์พุต
4	20	DTR	เอาท์พุต
5	7	GND	-
6	6	DSR	อินพุต
7	4	RTS	เอาท์พุต
8	5	CTS	อินพุต
9	22	RI	อินพุต

สำหรับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกแสดงในรูปที่ 2.4(ก) เป็นการเชื่อมต่อแบบนัลโมเด็ม (Null modem) หรือการเชื่อมต่อโดยตรงโดยไม่ต้องผ่านโมเด็ม โดยมีการตรวจสอบหรือแฮนด์เช็ก (Hand Check) เต็มรูปแบบ ส่วนในรูปที่ 2.4 (ข) เป็นการเชื่อมต่อแบบนัลโมเด็มในลักษณะที่ใช้สายเพียง 3 เส้น โดยเส้นสำหรับส่งข้อมูล อีกเส้นสำหรับรับข้อมูล และเส้นสุดท้ายเป็นกราวด์ สำหรับรายละเอียดหน้าที่การทำงานในแต่ละขาของพอร์ตสื่อสารอนุกรม RS-232 ที่ใช้งานในโครงงานนี้มีดังนี้



ก) การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ Null modem

ข) การต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ RS-232 โดยใช้สายสัญญาณเพียง 3 เส้น

รูปที่ 2.3 แสดงการต่ออุปกรณ์ภายนอกกับพอร์ตอนุกรมของคอมพิวเตอร์

- ขา TxD (Transmission Data) ขานี้ใช้เพื่อส่งข้อมูลออกจากคอมพิวเตอร์โดยนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูลออกไป

- ขา RxD (Receive Data) ขานี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์โดยนำข้อมูลที่อ่านได้เก็บไว้ในรีจีสเตอร์บัฟเฟอร์
- ขา RTS (Request To Send) ขานี้ใช้สำหรับส่งสัญญาณร้องขอให้ทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบนัลโมเด็ม 3 สายจะต้องเชื่อมขา RTS และ CTS ของตัวมันเองด้วยกัน เพื่อจะให้การรับและส่งข้อมูลสามารถเกิดขึ้นได้ตลอดเวลา
- ขา CTS (Clear To Send) ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ข้อมูลที่ขา TxD จะส่งออกไป ดังนั้นขานี้จึงถูกใช้เพื่อตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลหรือไม่
- ขา GND ขากราวด์ของระบบ

เนื่องจากเครื่องคอมพิวเตอร์ส่วนมากมักมีแหล่งจ่ายไฟ ซึ่งมีขนาด ± 5 โวลท์ และ ± 12 โวลท์ ซึ่งใช้สำหรับเลี้ยงวงจรและอุปกรณ์อื่นๆในระบบ ดังนั้นวงจรขับ (Line Driver) ของระบบ RS-232 ที่มีใช้อยู่ในคอมพิวเตอร์จึงมักกำหนดไว้เป็น ± 12 โวลท์ ด้วยเสมอ โดยสัญญาณที่เป็นลอจิก "0" นั้นวงจรภาคขับจะเปลี่ยนเป็น +12 โวลท์ ส่วนสัญญาณลอจิก "1" นั้นขับจะเปลี่ยนเป็น –12 โวลท์ ด้วยดังนั้นอุปกรณ์ขับแบบมาตรฐาน RS-232 นั้นในด้านภาคส่งต้องสามารถเปลี่ยนสัญญาณลอจิกให้เป็นระดับแรงดันตามค่าที่กำหนดไว้ในตารางได้สำหรับในส่วนของวงจรภาครับเองก็ต้อง สามารถตรวจจับระดับแรงดันที่รับเข้ามาแล้วเปลี่ยนกลับให้เป็นสัญญาณลอจิกได้อย่างถูกต้องด้วยเช่นกัน

สำหรับวงจรที่ใช้ในโครงงานนี้ เลือกใช้ไอซีขับ RS-232 สำเร็จรูปเบอร์ MAX232 ซึ่งทำหน้าที่แปลงระดับสัญญาณ TTL ลอจิกให้เป็นระดับสัญญาณในตามข้อกำหนดของมาตรฐาน RS-232 และยังทำหน้าที่แปลงระดับสัญญาณของ RS-232 ให้กลับมาเป็นระดับสัญญาณ TTL ได้ในตัวเดียวกันด้วย

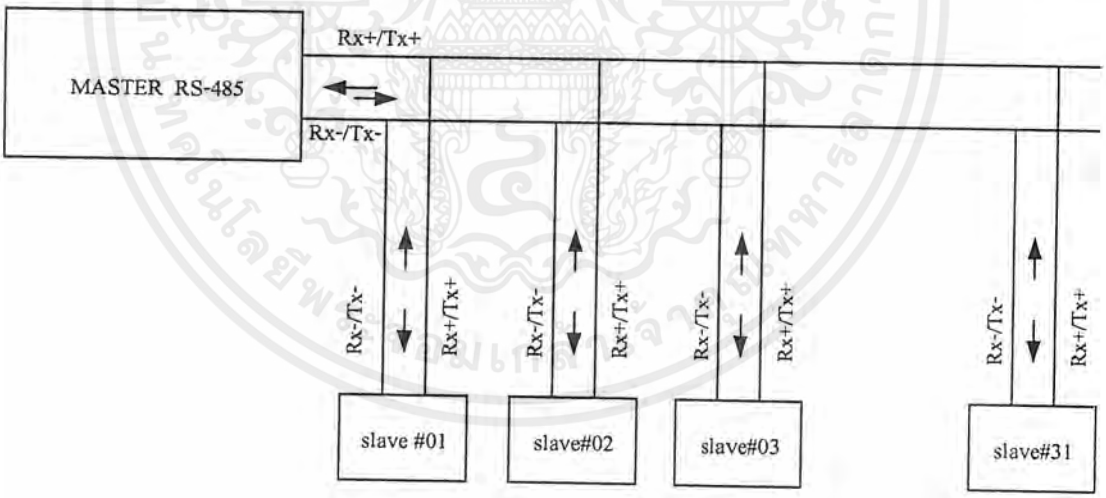
2.4.2 มาตรฐาน RS-485

การรับส่งข้อมูลแบบนี้จะมีลักษณะคล้ายกับแบบ RS-422 ซึ่งข้อกำหนดและคุณสมบัติของสัญญาณต่างๆจะเหมือนกับ RS-422 ทั้งหมดแต่การรับส่งแบบนี้จะมีข้อดีพิเศษกว่าแบบ RS-422 ก็คือใช้สัญญาณในการรับส่งเพียงคู่เดียวเท่านั้นแต่ RS-422 จะใช้สายส่งทั้งหมด 4 เส้น หรือ 2 คู่ คือคู่หนึ่งใช้สำหรับรับและอีกคู่หนึ่งใช้สำหรับส่งซึ่งจะเป็นแบบฟูลดูเพล็กซ์ ส่วนการรับส่งแบบ RS-485 นี้ใช้การรับส่งแบบฮาล์ฟดูเพล็ก (Half Duplex) สามารถจะทำหน้าที่ได้ทั้งรับข้อมูลและส่งข้อมูลโดยใช้สัญญาณคู่เดิมเพียงคู่เดียว แต่ในการรับหรือส่งข้อมูลนั้นต้องทำกันแบบผลัดกันรับผลัดกันส่ง ไม่สามารถจะทำหน้าที่พร้อมกันสองอย่างในช่วงเวลาเดียวกันได้

เนื่องจากการสื่อสารแบบนี้ จะใช้สัญญาณเพียง 1 คู่ในการทำหน้าที่ทั้งรับและส่งและยังสามารถต่ออุปกรณ์ร่วมกันได้มากถึง 32 ตัว พร้อมกันในเวลาเดียวกัน ดังนั้นจึงมีการจัดลำดับการรับส่งข้อมูลที่ดีจึงจะสามารถรับส่งกันได้อย่างมีประสิทธิภาพตามต้องการได้ จึงส่งผลทำให้การรับส่งแบบนี้มีความสลับซับซ้อนมากขึ้น เพราะต้องมีการเขียนโปรแกรมเพื่อคอยกำหนดทิศทางรวบทั้งจัดแบ่งการควบคุม ลำดับการสื่อสารภายในสายส่งให้กับอุปกรณ์แต่ละตัว ที่ต่ออยู่ว่าจะให้อุปกรณ์ตัวใดทำหน้าที่เป็นตัวส่งและให้

อุปกรณ์ตัวใดทำหน้าที่เป็นตัวรับ ซึ่งจะต้องเขียนโปรแกรมเพื่อควบคุมให้มีการส่งข้อมูลออกมาในสัญญาณได้เพียงครั้งละ 1 ตัว เท่านั้น เพราะถ้าหากมีการส่งข้อมูลออกมาในสายสัญญาณพร้อมกันในเวลาเดียวกันมากว่า 1 ตัวแล้วก็ จะทำให้ฝ่ายรับไม่สามารถรับข้อมูลที่ถูกต้องได้และเกิดความผิดพลาดขึ้นได้ ดังนั้นการใช้ระบบ RS-485 นี้ จะต้องมีการออกแบบและวางแผนไว้ล่วงหน้าทั้งทางด้านของอุปกรณ์ฮาร์ดแวร์ และโปรแกรมที่ใช้ในการควบคุมการทำงานของระบบ

สำหรับวิธีการเชื่อมต่อระบบ RS-485 นั้นทำได้หลายแบบ สำหรับโครงงานนี้ใช้การเชื่อมต่อแบบขนานทั้งระบบ จะมีอุปกรณ์ที่ทำหน้าที่เป็นตัวแม่ 1 ตัว สำหรับคอยจัดการ และควบคุมจัดลำดับการรับส่งของอุปกรณ์ตัวอื่น ๆ ในระบบ โดยตัวแม่จะทำหน้าที่เป็นตัวกลางในการรับส่งข้อมูลเองทั้งหมดหรือเป็นเพียงการจัดลำดับการรับส่งให้อุปกรณ์ตัวอื่น ๆ เพียงอย่างเดียวก็ได้ ซึ่งตัวลูกทุก ๆ ตัวนั้นในครั้งแรกต้องกำหนดทิศทางการรับส่งของตัวเองให้เป็น การรับข้อมูลไว้ตลอดเวลา แต่สำหรับตัวแม่นั้นจะคอยทำหน้าที่ติดต่อตัวลูกทีละตัว เพื่อคอยตรวจสอบว่ามีข้อมูลที่ต้องการจะส่งหรือไม่ ถ้ามีแล้วต้องการส่งถึงอุปกรณ์ตัวใด แล้วจึงสั่งให้อุปกรณ์ที่ระบุนั้นคอยรับข้อมูลโดยตรงเอง แล้วตัวแม่ก็จะเปลี่ยนทิศทางมาเป็นรอรับข้อมูลเพื่อ ปล่อยสัญญาณไว้ให้ว่างเพื่อที่อุปกรณ์ทั้งสองตัวที่ได้รับอนุญาตนั้นจะได้รับส่งข้อมูลกันต่อไป ซึ่งเมื่อทั้งคู่ทำการรับส่งข้อมูลกันเสร็จแล้วตัวลูกนั้นก็ต้องส่งคำสั่งมาบอกตัวแม่ให้ทราบว่าเสร็จแล้ว เพื่อที่ตัวแม่จะได้ทำหน้าที่ติดต่อตัวลูกอื่น ๆ ต่อไปเรื่อย ๆ

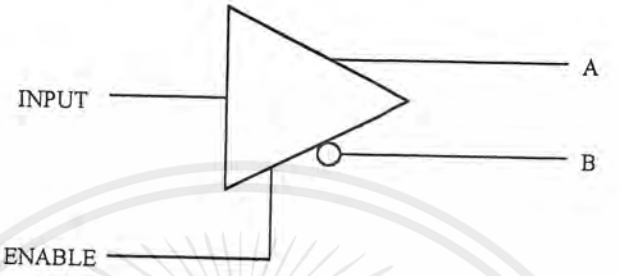


รูปที่ 2.4 ลักษณะการต่อ RS-485 แบบขนาน

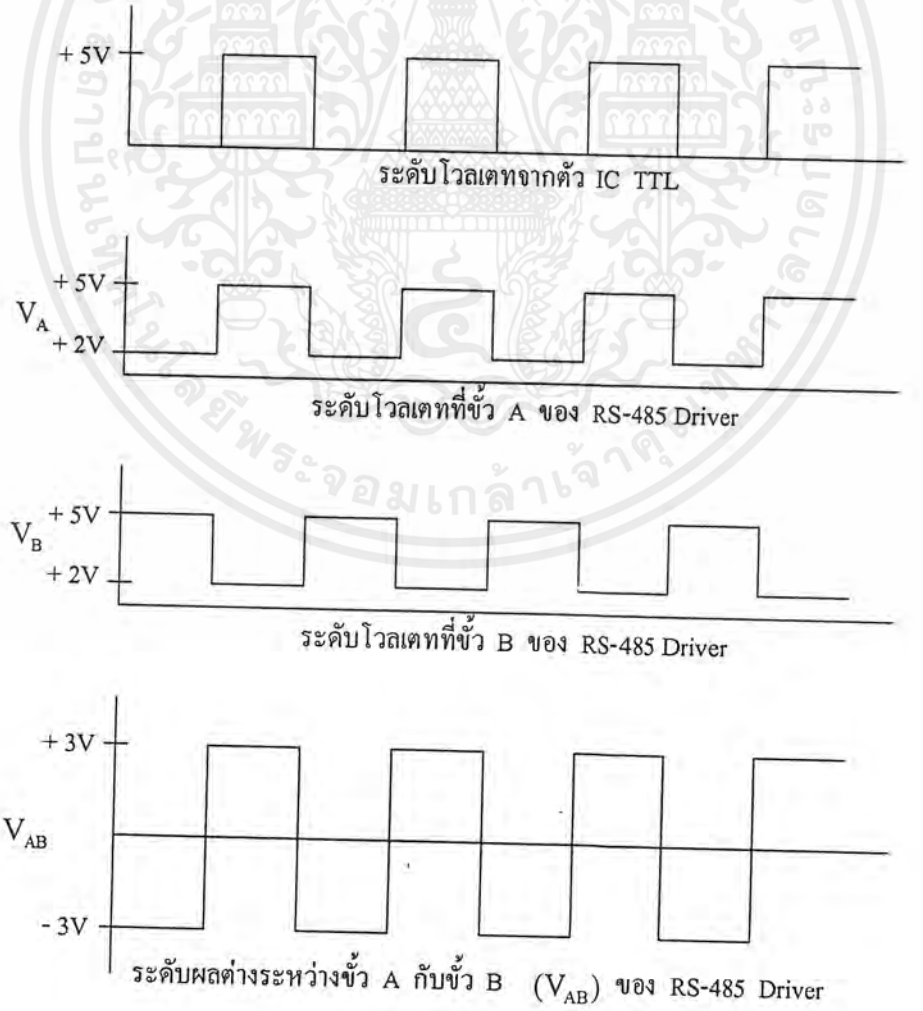
- **สายสัญญาณที่ใช้กับ RS-485**

สำหรับสายสัญญาณที่จะนำมาใช้ในการรับส่งข้อมูลด้วยระบบขับแบบ RS-485 นั้นจะต้องเป็นสายที่ออกแบบมาเพื่อใช้กับงานด้านการสื่อสารโดยเฉพาะ เช่น สายสัญญาณแบบ UTP (Unshiled Twis Pair) ซึ่งเป็นสายคู่ตีเกลียวแบบไม่มีชีลด์ ซึ่งเหมาะสำหรับใช้งานภายในอาคารซึ่งไม่มีแหล่งกำเนิดสัญญาณรบกวนแผ่กระจายออกมาในรัศมีที่สายพาดผ่านไม่มากนัก แต่ถ้าเส้นทางที่สายพาดผ่านนั้นมีระดับของสัญญาณรบกวนมากก็ต้องใช้สายแบบที่มีการป้องกันสัญญาณรบกวนไว้ด้วย คือแบบ STP

(Shiled Twis Pair) ซึ่งเป็นคู่สายตีเกลียวแบบมีชีลด์นั่นเอง โดยการนำไปต่อใช้งานต้องต่อให้ถูกด้วย เช่นสัญญาณ Tx จะมีอยู่ด้วยกันสองเส้นคือ Tx+ และ Tx- ซึ่งสัญญาณทั้งสองเส้นนี้ต้องต่อกับสายที่ตีเกลียวอยู่ภายในคู่เดียวกัน จะแยก Tx+ กับ Tx- แยกคู่กันไม่ได้ เพราะจะทำให้เกิดการไม่สมดุลย์ของสัญญาณทั้งคู่ทำให้เกิดความผิดพลาดได้ง่ายขึ้น



รูปที่ 2.5 แสดงตัวขับ RS-485

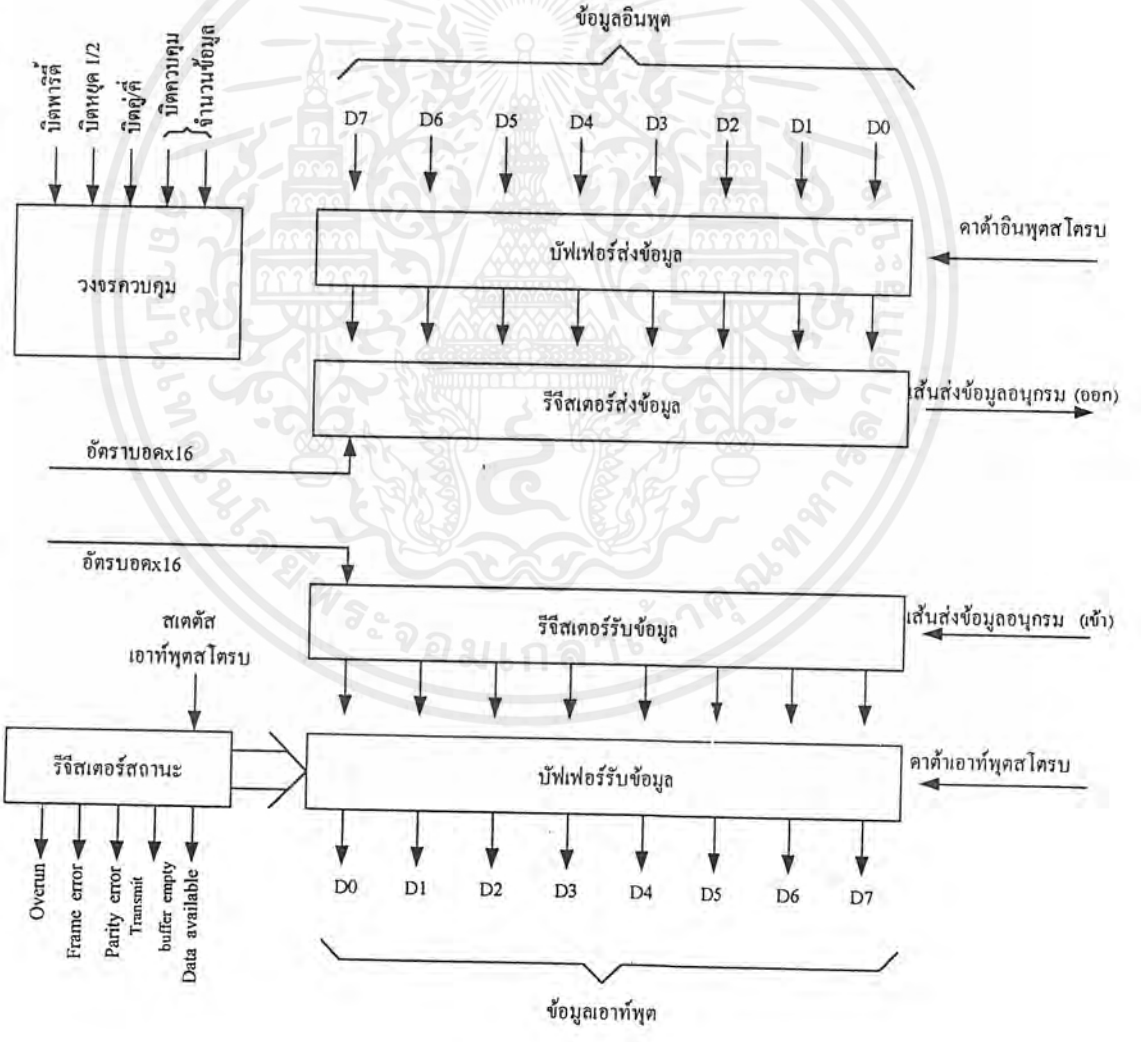


รูปที่ 2.6 แสดงระดับของโวลท์เตทของ RS-485

2.5 การสื่อสารอนุกรมยูเออาร์ที (UART : Universal Asynchronous Receiver Transmitter)

ยูเออาร์ทีเป็นอุปกรณ์ที่ทำหน้าที่รับและส่งข้อมูลแบบอะซิงโครนัส หน้าที่หลักของยูเออาร์ที คือ ทำหน้าที่แปลงข้อมูลที่อยู่ในรูปแบบขนานจากคอมพิวเตอร์ให้อยู่ในรูปแบบอนุกรมแบบอะซิงโครนัส แล้วส่งออกไป และทำหน้าที่แปลงสัญญาณอนุกรมแบบอะซิงโครนัสที่ป้อนเข้ามายัง ยูเออาร์ที ให้เป็นแบบขนานก่อนที่จะส่งเข้าคอมพิวเตอร์ ดังแสดงบล็อกไดอะแกรมของยูเออาร์ทีตามรูปที่ 2.6

ซึ่งนอกจากยูเออาร์ที จะส่งข้อมูลไปยังคอมพิวเตอร์แล้ว ยังแจ้งข้อมูลอื่น ๆ ให้คอมพิวเตอร์รับทราบด้วย เช่น อัตราเร็วในการส่งข้อมูล (บอดเรต) รูปแบบการส่งข้อมูล ความผิดพลาดที่เกิดขึ้นระหว่างการถ่ายทอดข้อมูล เป็นต้น



รูปที่ 2.7 บล็อกไดอะแกรมของยูเออาร์ที

ในการส่งข้อมูล 8 บิตแบบอนุกรมนี้จะต้องมีบิตเริ่มต้นและบิตหยุดเพิ่มขึ้นมา ซึ่งจะทำให้ข้อมูลที่ถูกส่งไปจริง ๆ มีขนาด 10 บิต โดยบิตเริ่มต้นจะมีค่าเป็น "0" เสมอ จะบอกยูเออาร์ที ที่รับข้อมูลให้รู้ว่ามีข้อมูลกำลังเข้ามา และบิตหยุดมีค่าเป็น "1" เสมอ จะบอกยูเออาร์ทีที่รับข้อมูลให้รู้ว่าการส่งข้อมูลได้เสร็จสิ้นลงแล้ว

ส่วนที่มีหน้าที่ส่งข้อมูลของยูเออาร์ทีแยกออกได้เป็น 2 ส่วนคือ บัฟเฟอร์ส่งข้อมูล (Transmited Data Output Buffer) กับรีจีสเตอร์ส่งข้อมูล (Transmit Register) โดยรจีสเตอร์ส่งข้อมูลจะมีหน้าที่เลื่อนข้อมูลส่งออกไปยังเส้นส่งข้อมูลอนุกรม โดยเริ่มจากบิตเริ่มต้น ถัดมาเป็นบิตข้อมูล และตามด้วยบิตหยุด ข้อมูลขนาด 8 บิตที่จะส่งจะถูกนำไปเก็บไว้ในบัฟเฟอร์ส่งข้อมูล เมื่อสัญญาณควบคุมดาต้าอินพุตสโตรป (Data Input Strobe) เปลี่ยนค่าจาก 1 เป็น 0 และการส่งข้อมูลแบบอนุกรมจะเริ่มขึ้นเมื่อสัญญาณควบคุมนี้เปลี่ยนค่าจาก 0 เป็น 1

ส่วนที่ทำหน้าที่รับข้อมูลของยูเออาร์ที จะเหมือนเป็นภาพสะท้อนของส่วนที่ทำหน้าที่ส่งข้อมูล ข้อมูลที่เข้ามาทางเส้นรับข้อมูลอนุกรม จะถูกเลื่อนเข้าไปเก็บใจรีจีสเตอร์รับข้อมูล โดยการเลื่อนค่า 10 หรือ 11 ครั้ง ขั้นตอนดังกล่าวจะเริ่มเมื่อมีบิตเริ่มต้นเข้า และทันทีที่ข้อมูลทั้งหมดถูกเลื่อนเข้าไปเก็บในรีจีสเตอร์รับข้อมูลแล้ว ข้อมูลในรีจีสเตอร์นี้จะถูกนำไปเก็บในบัฟเฟอร์รับข้อมูล เมื่อมีสัญญาณควบคุมดาต้าเอาต์พุตสโตรบ

ภายในยูเออาร์ทีจะมีส่วนของวงจรสร้างบอดเรตแบบโปรแกรมได้ (Programmable Buadrate Generator) โดยการกำหนดค่าตัวหารให้กับสัญญาณนาฬิกาของยูเออาร์ที โดยตัวหารนี้มีขนาด 16 บิต ดังนั้นจึงสามารถกำหนดตัวหารอยู่ในช่วง 1 - 65535 ยูเออาร์ทีสามารถรับส่งข้อมูลได้ทั้งแบบซาล์ฟดูเพล็กซ์และฟูลดูเพล็กซ์

2.5.1 ชนิดของยูเออาร์ที

ในเครื่องคอมพิวเตอร์ทั่วไปมียูเออาร์ทีที่ใช้งานกันอยู่ 2 แบบคือ 8250 ซึ่งเป็นยูเออาร์ทีมาตรฐานที่มีใช้กันมายาวนาน ยูเออาร์ทีเบอร์นี้จะมีบัฟเฟอร์สำหรับรับและส่งข้อมูลตำแหน่งเดียวกัน ทำให้การรับและส่งข้อมูลกำจัดความเร็วอยู่ที่ 57.6 กิโลบิตต่อวินาที แต่ยูเออาร์ทีเบอร์นี้ถือว่าเป็นต้นแบบของยูเออาร์ทีที่ใช้ในคอมพิวเตอร์ โดยคอมพิวเตอร์ทุกๆรุ่นจะต้องสนับสนุนการทำงานตามรูปแบบของยูเออาร์ทีเบอร์นี้

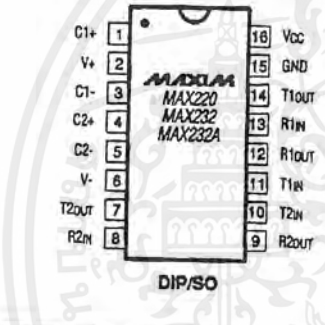
ยูเออาร์ทีอีกเบอร์หนึ่งคือ 16450 มีความสามารถรับส่งข้อมูลได้ที่ความเร็ว 115200 บิตต่อวินาที และเพิ่มรีจีสเตอร์สำหรับพักข้อมูลสำหรับยูเออาร์ที นอกจากนั้นยังเพิ่มส่วนของชิฟท์รีจีสเตอร์แบบเอฟไอเอฟโอ (First In First Out) ขนาด 16 ไบต์เข้าไป ทำให้สามารถสนับสนุนความเร็วในการรับส่งข้อมูล 256 กิโลบิตต่อวินาที โดยคอมพิวเตอร์ในปัจจุบันใช้ยูเออาร์ทีเบอร์นี้หรือใหม่กว่าเช่น เบอร์ TL16C750 ซึ่งมีรีจีสเตอร์แบบเอฟไอเอฟโอขนาด 64 ไบต์ทำงานได้ที่ระดับแรงดัน +5 โวลท์ และ +3 โวลท์ มีโหมดประหยัดพลังงาน สามารถรับส่งข้อมูลได้ที่ความเร็ว 1 เมกะบิตต่อวินาที เมื่อใช้สัญญาณนาฬิกา 16 เมกะเฮร์ต

2.6 โครงสร้างของไอซี MAX232

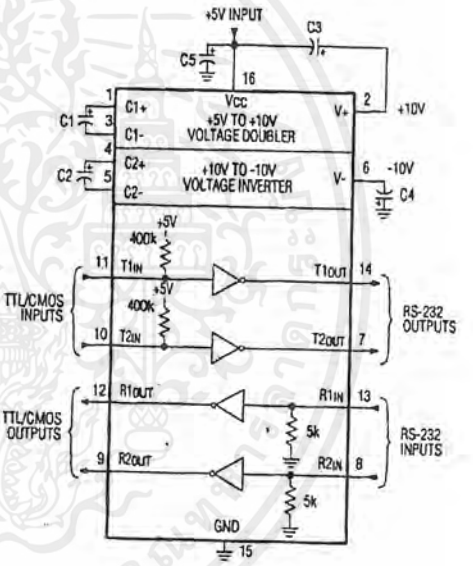
ไอซีเบอร์ MAX232 เป็นไอซีที่ทำหน้าที่ขับแรงดันไฟฟ้าจากระดับแรงดันของไอซีประเภท TTL หรือ CMOS เป็นระดับแรงดันไฟฟ้าตามมาตรฐาน RS-232 ซึ่งใช้แหล่งจ่ายไฟฟ้าเพียงแหล่งเดียว คือ +5 V เพราะมีวงจร Voltage Doubler และ Voltage Inverter คือส่วนขยายแรงดันจาก +5 โวลท์ เป็นแรงดัน +10 โวลท์ และ -10 โวลท์ อยู่ภายในตัวไอซี

คุณสมบัติทั่วไปของไอซีเบอร์ MAX232

- ใช้แหล่งจ่ายไฟฟ้าเพียงชุดเดียว (+5V)
- ประหยัดพลังงานในสภาวะที่ไม่มีสัญญาณ
- มีคุณสมบัติตามมาตรฐาน EIA/TIA-232E และ V.28
- มีตัวขับและตัวรับ 2 ชุด (Multiple Drivers and Receivers)



CAPACITANCE (μF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

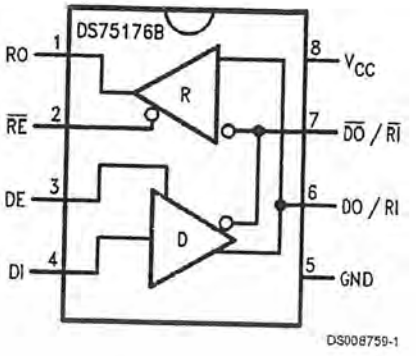


รูปที่ 2.8 แสดงตัวไอซี่ MAX232 และโครงสร้างภายใน

2.7 โครงสร้างของไอซีเบอร์ DS75176

ไอซีเบอร์ DS75176 เป็นไอซีที่ทำหน้าที่ขับแรงดันไฟฟ้าจากระดับแรงดันของไอซีประเภท TTL หรือ CMOS เป็นระดับแรงดันไฟฟ้าตามมาตรฐาน RS-485 โดยมีคุณสมบัติดังนี้

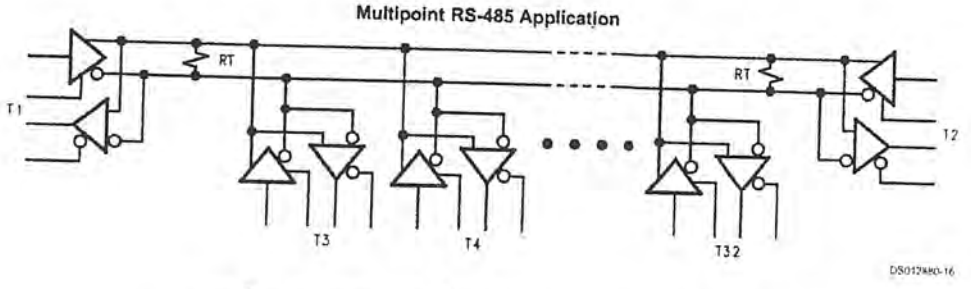
- มีคุณสมบัติตามมาตรฐาน EIA RS-485 และ RS-422
- มีช่วงของการหน่วงเวลาในการขับ 22 นาโนวินาที
- ใช้แหล่งจ่ายไฟฟ้า +5 V แหล่งเดียว
- มีส่วนป้องกันอุณหภูมิ
- มีความต้านทานในที่เอาท์พุตสูงโดยที่ตัวขับเป็นความต้านทาน (TRI-STATE) หรือในสภาวะไม่ทำงาน
- ตัวขับ 1 ตัวสามารถต่อเชื่อมกับตัวรับได้ 32 ตัว



รูปที่ 2.9 แสดงโครงสร้างภายในและตำแหน่งขาของตัวไอซี DS75176

จากรูปที่ 2.9 ซึ่งแสดงโครงสร้างภายในและตำแหน่งขาของไอซี DS75176 การใช้งานของขาต่างๆ เป็นดังนี้คือ

- ขา 1 (Receiver Output : RO) เป็นขาเอาท์พุตของสัญญาณที่รับได้โดยที่เมื่อ $A > B$ ประมาณ 200 มิลลิโวลท์จะได้ระดับลอจิกเป็น "1" เมื่อ $A < B$ ก็จะได้ระดับลอจิกเป็น "0" และเมื่อ A และ B ไม่ได้ทำการเชื่อมต่อหรือเป็นวงจรเปิดก็จะได้ระดับลอจิกเป็น "1"
- ขา 2 (Receiver Output Enable : RE) ในการรับสัญญาณอินพุททางขา 1 (RO) ที่ขานี้จะต้องเป็นลอจิก "0" ถ้าเป็นลอจิก "1" จะทำให้ขา 1 (RO) มีสภาพเป็นความต้านทาน
- ขา 3 (Driver Output Enable : DE) เมื่อต้องการส่งข้อมูลที่ขานี้จะต้องเป็นลอจิก "1" และจะอยู่ในสภาพความต้านทานเมื่อขานี้เป็นลอจิก "0"
- ขา 4 (Driver Input : DI) เป็นขาอินพุตสำหรับการขับหรือส่ง
- ขา 5 (Ground : GND)
- ขา 6 (A) เอาท์พุตของการขับแบบไม่กลับเฟส (Non-Inverting Driver) และเป็นอินพุตของการรับแบบไม่กลับเฟส (Non-Inverting Receiver)
- ขา 7 (B) เอาท์พุตของการขับแบบกลับเฟส (Inverting Driver) และเป็นอินพุตของการรับแบบกลับเฟส (Inverting Receiver)
- ขา 8 (Power Supply : VCC) ขารับแหล่งจ่ายไฟเลี้ยง



รูปที่ 2.10 แสดงการเชื่อมต่อแบบหลายจุดของ RS-485

2.8 การใช้งานพอร์ตอนุกรมของคอมพิวเตอร์ด้วยโปรแกรม Visual Basic

สำหรับการใช้งาน Visual Basic ตั้งแต่เวอร์ชั่น 2 เป็นต้นมาใน Visual Basic จะมีคัสตอมคอนโทรลสำหรับการสื่อสารอนุกรมผ่านทางพอร์ตอนุกรมของคอมพิวเตอร์มาให้ โดยใน Visual Basic เวอร์ชั่น 2 และเวอร์ชั่น 3 จะใช้ชื่อว่า MSCOMM.VBX ส่วนเวอร์ชั่น 4 ใช้ชื่อว่า MSCOMM16.OCX สำหรับการทำงานกับระบบปฏิบัติการ 16 บิต และ MSCOMM32.OCX สำหรับการทำงานกับระบบปฏิบัติการ 32 บิต สำหรับ Visual Basic เวอร์ชั่น 5 จะมีเพียง MSCOMM32.OCX ของโปรแกรมเท่านั้นเพราะถูกออกแบบมาให้ใช้งานกับระบบปฏิบัติการ 32 บิต

MSComm จัดเตรียมทางเลือกเอาไว้ 2 ทางเพื่อความสะดวกในการสื่อสารข้อมูล ทางแรกคือการสื่อสารข้อมูลที่กระตุ้นด้วยเหตุการณ์ (event-driven communication) เป็นรูปแบบการใช้งานที่มีประสิทธิภาพมากสำหรับการตอบสนองแบบทันทีทันใด เช่น เมื่อตัวอักษรถูกส่งมาที่พอร์ตอนุกรมหรือเกิดการเปลี่ยนแปลงที่ขา DCD หรือขา RTS เหตุการณ์ Oncomm ของ MSComm จะสามารถตรวจสอบสัญญาณนั้นได้ทันที ส่วนทางเลือกที่สองเป็นการคอยตรวจสอบค่าเหตุการณ์และความผิดพลาดที่เกิดขึ้นด้วยการดูค่าที่เปลี่ยนแปลงภายในคุณสมบัติ CommEvent หลังจากให้โปรแกรมทำงานในฟังก์ชันต่างๆ ไปเรียบร้อยแล้ว ซึ่งวิธีนี้ใช้งานได้ดีในการเขียนโปรแกรมที่มีขนาดเล็ก

คอนโทรล MSComm 1 ตัวสามารถควบคุมการทำงานของพอร์ตอนุกรมได้ 1 พอร์ต ถ้าในโปรแกรมที่ใช้งานต้องการติดต่อกับพอร์ตอนุกรมมากกว่า 1 พอร์ตจะต้องใช้คอนโทรล MSComm มากกว่า 1 ตัวเพื่อควบคุมพอร์ตอนุกรมในแต่ละพอร์ต แอดเดรสของพอร์ตอนุกรมและแอดเดรสของการเกิดอินเตอร์รัปต์สามารถเปลี่ยนแปลงได้จากการแก้ไขค่าที่คอนโทรลแพนเนล (Control Panel)

- **CommPort** ใช้ในการกำหนดและอ่านค่าพอร์ตอนุกรมที่ติดต่ออยู่

  (COM1, COM2, COM3, COM4)

รูปแบบการใช้งาน

Object.CommPort [= value]

โดย value เป็นค่าของพอร์ตอนุกรม ชนิดของข้อมูลเป็นจำนวนเต็ม (Integer) ค่า value สามารถกำหนดได้ในช่วง 1-16 (ค่าเริ่มต้นกำหนดไว้ที่ 1) เมื่อมีการกำหนดค่าแล้วทำการเปิดพอร์ตโดยใช้คุณสมบัติ PortOpen แต่ว่าพอร์ตนั้นไม่มีอยู่ในระบบ MSComm จะสร้างสัญญาณแสดงข้อผิดพลาด error 68 ขึ้นมา ซึ่งหมายถึงอุปกรณ์ตัวนี้ไม่มีอยู่ในระบบ ดังนั้นการเขียนโปรแกรมจึงจำเป็นต้องกำหนดตำแหน่งของพอร์ตอนุกรมก่อนที่ใช้คำสั่ง OpenPort

- **Setting** ใช้ในการกำหนดค่าอัตราบอด, พาริตี้, จำนวนของบิตข้อมูล,จำนวนของบิตปิดท้าย

รูปแบบการใช้งาน

Object.Settings [= value]

ค่า value มีชนิดข้อมูลเป็นแบบ String มีรูปแบบเป็น "BBBB,P,D,S" โดย BBBB เป็นค่าอัตราบอด, P เป็นค่าพาริตี้, D เป็นจำนวนของบิตข้อมูล และ S เป็นจำนวนบิตปิดท้ายปกติแล้วค่านี้ถูกกำหนดไว้เป็น "9600,N,8,1"

ค่าบอดเรตมาตรฐานที่ใช้กับ MSComm มีดังนี้

110 บิตต่อวินาที

300 บิตต่อวินาที

600 บิตต่อวินาที

1200 บิตต่อวินาที

2400 บิตต่อวินาที

9600 บิตต่อวินาที

14400 บิตต่อวินาที

19200 บิตต่อวินาที

28800 บิตต่อวินาที

38400 บิตต่อวินาที

56000 บิตต่อวินาที

128000 บิตต่อวินาที

256000 บิตต่อวินาที

สำหรับค่ามาตรฐานในการกำหนดค่าพาริตี้มีดังนี้

E = พาริตี้คู่

M = ลอจิก "1"

N = ไม่มีพาริตี้

O = พาริตี้คี่

S = ลอจิก "0"

ค่าที่ใช้ในการกำหนดจำนวนบิตมี 5 ค่าคือ 4, 5, 6, 7 และ 8

ค่าที่ระบุจำนวนบิตท้ายมี 3 ค่าคือ 1, 1.5 และ 2

- **PortOpen** ใช้ในการกำหนดอ่านค่าสถานะของพอร์ตอนุกรม เพื่อเปิดและปิดพอร์ตอนุกรม

รูปแบบการใช้งาน

Object.PortOpen [= value]

ค่า value มีชนิดข้อมูลเป็นแบบบูลีนคือ True กับ False โดย True หมายถึงการเปิดพอร์ตอนุกรมและ False หมายถึงการปิดพอร์ตอนุกรม สำหรับการปิดพอร์ตนั้นจะมีการเคลียร์บัฟเฟอร์รับข้อมูลและบัฟเฟอร์ส่งข้อมูลด้วย คอนโทรล MSComm จะปิดพอร์ตอนุกรมโดยอัตโนมัติเมื่อออกจากโปรแกรม ก่อนที่ใช้คุณสมบัติ PortOpen ต้องตรวจสอบให้แน่ใจก่อนว่าคุณสมบัติ CommPort นั้นได้ทำการกำหนดตำแหน่งของพอร์ตอนุกรมไว้ถูกต้องหรือไม่ มิเช่นนั้น MSComm จะแสดงข้อผิดพลาด Error 68 แจ้งแก่ผู้ใช้งาน หรือถ้าพอร์ตอนุกรมนั้นถูกเปิดเอาไว้แล้ว โปรแกรมก็จะแจ้งข้อผิดพลาดออกมาเช่นเดียวกัน

ถ้าคุณสมบัติ DTREnable หรือ RTSEnable ถูกกำหนดให้เป็น True ก่อนที่จะทำการพอร์ต ค่าคุณสมบัติของ DTREnable หรือ RTSEnable จะถูกเซตเป็น False หลังจากปิดพอร์ต แต่ถ้าเซตเป็น False หลังจากปิดโปรแกรมแล้ว ค่าที่กำหนดไว้จะเป็นค่าเดิม

ตัวอย่างการใช้คำสั่งเปิดพอร์ต เพื่อติดต่อสื่อสารกับพอร์ตอนุกรม COM1 และมีบอดเรต 9600 บิตต่อวินาที ไม่มีพาริตี้ จำนวนบิตข้อมูล 8 บิตและบิตปิดท้าย 1 บิตมีดังนี้

```
MSComm.Settings = "9600,n,8,1"
MSComm.ComPort = 1
MSComm.PortOpen = True
```

- **Input** ใช้อ่านค่าและลบค่าขบวนข้อมูลจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งาน

```
Object.Input [ = value]
```

คุณสมบัติ InputLen เป็นตัวกำหนดจำนวนของตัวอักษรที่จะอ่านโดยคุณสมบัติอินพุต การกำหนดค่าให้ InputLen เท่ากับ 0 เป็นการกำหนดให้คุณสมบัติอินพุตทำการอ่านค่าข้อมูลในบัฟเฟอร์รับข้อมูลทั้งหมด

คุณสมบัติ InputMode เป็นตัวกำหนดชนิดของข้อมูลที่คุณสมบัติอินพุตรับเข้ามา ถ้า InputMode ถูกกำหนดเป็น ComInputModeText คุณสมบัติอินพุตจะส่งค่าข้อมูลกลับมาในรูปแบบของข้อความชนิดข้อมูลเป็นแบบมีการเปลี่ยนแปลง (Variant) ถ้า InputMode กำหนดเป็น ComInputModeBinary คุณสมบัติอินพุต จะส่งข้อมูลกลับมาในรูปของไบนารีและชนิดข้อมูลเป็นแบบ Variant

- **InBufferCount** ใช้ส่งค่าจำนวนของตัวอักษรที่อยู่ในบัฟเฟอร์ภาครับ

รูปแบบการใช้คำสั่ง

```
Object.InBufferCount [=value]
```

คำสั่ง InbufferCount จะแสดงค่าจำนวนตัวอักษร ซึ่งรับมาจากภายนอกและยังเก็บอยู่ในบัฟเฟอร์ภาครับ เพื่อให้ผู้ใช้งานอ่านค่าออกไป สำหรับการเคลียค่าบัฟเฟอร์ภาครับทำได้โดยกำหนดให้อินบัฟเฟอร์เคาน์มีค่าเป็น 0

- **InBufferSize** ใช้กำหนดและคืนค่าขนาดของบัฟเฟอร์ภาครับในหน่วยเป็นไบต์

รูปแบบการใช้งานคำสั่ง

```
Object.InBufferSize [ = value ]
```

คำสั่ง InBufferSize ใช้เพื่อกำหนดขนาดของบัฟเฟอร์ภาครับ โดยค่าเริ่มต้นถูกกำหนดไว้ที่ 1024 ไบต์

- **InputLen** ใช้กำหนดค่าและคืนค่าจำนวนของตัวอักษรที่อ่านจากบัฟเฟอร์ภาครับ

รูปแบบการใช้งานคำสั่ง

```
Object.InputLen [ = value ]
```

ค่าเริ่มต้นของคุณสมบัติ InputLen มีค่าเท่ากับ "0" จะทำให้คำสั่งอินพุตของ MSComm อ่านค่าข้อมูลที่อยู่ภายในบัฟเฟอร์ภาครับทั้งหมด

ถ้าไม่มีข้อมูลอยู่ในบัฟเฟอร์ภาครับมากเท่ากับจำนวน InputLen คำสั่งอินพุตจะส่งค่าว่าง ("") กลับออกมา ผู้ใช้งานสามารถตรวจสอบข้อมูลในบัฟเฟอร์ภาครับได้โดยใช้คุณสมบัติ InbufferCount โดยกำหนดให้มีข้อมูลอยู่ในบัฟเฟอร์ภาครับก่อนแล้วจึงค่อยอ่านข้อมูลจากบัฟเฟอร์ภาครับ

- **InputMode** ใช้กำหนดค่าและคืนค่าชนิดของข้อมูลที่รับโดยคำสั่งอินพุต

รูปแบบการใช้งานคำสั่ง

Object.InputMode [= value]

คุณสมบัติอินพุตโหมดใช้กำหนดว่าข้อมูลชนิดไหนที่รับเข้ามาผ่านคำสั่งอินพุต โดยข้อมูลจะเลือกได้ 2 ประเภทคือ

ComInputModeText สำหรับข้อมูลที่อยู่ในรูปข้อความตัวอักษรตามมาตรฐาน ANSI โดยจะต้องกำหนดค่าเป็น "0" และค่าเริ่มต้นการรับข้อมูลก็จะเป็นค่านี้

ComInputModeBinary สำหรับข้อมูลอื่นๆ ซึ่งเก็บในรูปไบนารี่รวมกันอยู่เป็นไบต์ข้อมูล

- **Output** ใช้ในการส่งขบวนของข้อมูลไปยังบัฟเฟอร์ส่งข้อมูล

รูปแบบการใช้งาน คำสั่ง

object.Output [= value]

ค่า value เป็นค่าของตัวอักษรที่เขียนไปยังบัฟเฟอร์ส่งข้อมูล คุณสมบัติเอาต์พุต สามารถใช้ในการส่งข้อมูลตัวอักษรหรือข้อมูลไบนารี่ก็ได้ โดยการส่งข้อมูลเป็นรูปแบบตัวอักษรจะต้องกำหนดข้อมูลเป็นแบบ Variant และข้อมูลภายในเป็นแบบ String สำหรับการส่งข้อมูลไบนารี่จะต้องกำหนดชนิดของข้อมูลเป็นแบบ Variant และมีข้อมูลภายในเป็นแบบไบต์

- **OutBufferCount** ใช้คืนค่าจำนวนของข้อมูลตัวอักษรที่เก็บอยู่ในบัฟเฟอร์ภาคส่ง และสามารถใช้คำสั่งนี้เพื่อเคลียร์บัฟเฟอร์ภาคส่งได้ด้วย

รูปแบบการใช้งานคำสั่ง

object.OutBufferCount [= value]

ผู้ใช้งานสามารถเคลียร์บัฟเฟอร์ภาคส่งได้โดยการกำหนดค่าเอาต์พุตบัฟเฟอร์เคาน์เท่ากับ "0"

- **OutBufferSize** ใช้กำหนดค่าและคืนค่าขนาดของบัฟเฟอร์ภาคส่ง ชนิดตัวแปลเป็นแบบไบต์

รูปแบบการใช้งานคำสั่ง

object.OutBufferSize [= value]

คุณสมบัติ OutBufferSize ใช้สำหรับกำหนดขนาดของบัฟเฟอร์ภาคส่ง โดยค่าปกติที่ใช้งานจะมีค่าเท่ากับ 512 ไบต์

- **ParityReplace** ใช้กำหนดและคืนค่าตัวอักษรที่ไปวางแทนในตำแหน่งที่เกิดข้อมูลผิดพลาดจากพาริตี้

รูปแบบการใช้งานคำสั่ง

object.ParityReplace [= value]

สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

บิตพาริตี้เป็นบิตที่ทางภาคส่งข้อมูลทำการส่งมาพร้อมกับข้อมูล เพื่อตรวจสอบข้อผิดพลาดของข้อมูล โดยเมื่อมีการใช้บิตพาริตี้ คอนโทรล MSComm จะทำการบวกบิตทุกบิตที่มีค่าลอจิก "1" ในแต่ละไบต์ และทำการตรวจสอบผลลัพธ์ว่าบิตที่อ่านได้นั้นมีลอจิก "1" เป็นเลขคู่หรือคี่ และตรงกับค่าที่กำหนดไว้แต่ต้นหรือไม่ ถ้าค่าที่นำมาบวกแล้วมีพาริตี้ไม่ตรงแสดงว่าการรับส่งข้อมูลผิดพลาด

การกำหนดค่า เริ่มต้นให้กับ ParityReplace นั้นกำหนดให้ใช้เครื่องหมาย (?) ไปวางไว้ที่ตำแหน่งที่เกิดพาริตี้ผิดพลาด ถ้ากำหนดค่า ParityReplace ให้เป็นค่าว่าง ("") จะเป็นการยกเลิกการใช้งาน ParityReplace และไม่มีการป้อนข้อมูลแทนเมื่อตรวจสอบข้อผิดพลาด

ParityReplace ใช้ชนิดข้อมูลเป็นแบบสตริง แต่การกำหนดจะกำหนดได้เพียงไบต์เดียวเท่านั้น ซึ่งจะสามารถใช้ค่าใดๆก็ได้ที่เป็นโค้ด ANSI มีค่าอยู่ระหว่าง 0-255

- **DTREnable** ใช้ในการกำหนดสถานะลอจิกของขา DTR โดยสัญญาณของขา DTR จะส่งจากคอมพิวเตอร์ไปยังโมเด็มเพื่อแสดงว่าคอมพิวเตอร์พร้อมที่จะรับข้อมูลแล้วชนิดของข้อมูลเป็นแบบบูลีน

  รูปแบบการใช้งานคำสั่ง

  object.DTREnable [= value]

  ค่า Value เป็นค่าสถานะ True หรือ False เพื่อกำหนดลอจิกของขา DTR ให้เป็น "0" หรือ "1" โดย

  True หมายถึง ให้ขา DTR มีลอจิก "1"

  False หมายถึง ให้ขา DTR มีลอจิก "0"

- **RTSEnable** ใช้เพื่อกำหนดสถานะลอจิกให้ขา RTS โดยขา RTS จะเป็นสัญญาณที่ส่งมาจากคอมพิวเตอร์ไปยังโมเด็มเพื่อร้องขอส่งข้อมูล ชนิดของข้อมูลเป็นแบบบูลีน

  รูปแบบการใช้งานคำสั่ง

  object.RTSEnable [= value]

  ค่า Value เป็นค่าสถานะ True หรือ False เพื่อกำหนดลอจิก "0" หรือ "1" ให้ขา RTS โดย

  True หมายถึง ให้ขา RTS มีลอจิก "1"

  False หมายถึง ให้ขา RTS มีลอจิก "0"

- **EOFEnable** เป็นการกำหนดให้ MSComm รอสัญลักษณ์แสดงส่วนสุดของไฟล์ระหว่างการรับอินพุตเข้ามา ถ้าพบสัญลักษณ์ EOF ภาคอินพุตจะหยุดรับข้อมูล และเหตุการณ์ OnComm จะถูกกระตุ้นให้ทำงาน คุณสมบัติ CommEvent จะมีค่าเท่ากับ 7 หรือ ComEvEOF

  รูปแบบการใช้งานคำสั่ง

  object.EOFEnable [= value]

  โดย Value เป็นค่าสถานะ True หรือ False เพื่ออีนาเบิลหรือดีสอีนาเบิลการทำงานของเหตุการณ์ OnComm เมื่อตรวจพบสัญญาณ EOF โดย

  True หมายถึง เหตุการณ์ OnComm จะถูกกระตุ้นให้ทำงานด้วย EOF

  False หมายถึง เหตุการณ์ OnComm จะไม่ถูกกระตุ้นให้ทำงานด้วย EOF

  เมื่อ EOFEnable กำหนดให้เป็น False ส่วนควบคุมจะไม่มีการตรวจสอบสัญลักษณ์ EOF

- **CTSHolding** ผู้ใช้งานสามารถตรวจสอบการทำงานของขา CTS ได้ว่ามีสถานะลอจิก "0"

43218

หรือ "1"โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CTSHolding เป็น True ขา CTS จะมีสถานะลอจิกเป็น "1" ถ้าค่า CTSHolding เป็น False ขา CTS จะมีสถานะลอจิกเป็น "0"

รูปแบบการใช้งานคำสั่ง

object.CTSHolding

เมื่อขา CTS เป็นลอจิก "0" และเกิดไทม์เอาท์คอนโทรล MSComm จะกำหนดให้คุณสมบัติ ComEvent มีค่าเป็น comEventCTSTO และกระตุ้นให้เกิดเหตุการณ์ไทม์เอาต์

- **CDHolding** ผู้ใช้งานสามารถตรวจสอบการทำงานของขา DCD ได้ว่ามีสถานะลอจิกเป็น "1" หรือ "0" โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า CDHolding เป็น True ขา DCD จะมีสถานะลอจิก "1" ถ้าค่า CDHolding เป็น False ขา DCD จะมีสถานะลอจิกเป็น "0"

รูปแบบการใช้งาน

object.CDHolding

เมื่อขา DCD มีลอจิก "1" จะเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEventCDTO และกระตุ้นให้เกิดเหตุการณ์ OnComm

- **DSRHolding** ผู้ใช้งานสามารถตรวจสอบการทำงานของขา DSR ได้ว่ามีสถานะลอจิก "1" หรือ "0" โดยค่าที่อ่านได้จะเป็นบูลีน True และ False ถ้าค่า DSRHolding เป็น True ขา DSR จะมีสถานะลอจิก "1" ถ้าค่า DSRHolding เป็น False ขา DSR จะมีสถานะลอจิก "0"

รูปแบบการใช้งาน

object.DSRHolding

เมื่อขา DSR เป็นลอจิก "1" และเกิดไทม์เอาต์ คอนโทรล MSComm จะกำหนดให้คุณสมบัติ CommEvent มีค่าเป็น comEventDSRTO และกระตุ้นให้เกิดเหตุการณ์ OnComm

- **Handshaking** กำหนดคุณสมบัติและคืนค่ารูปแบบแฮนด์เช็กทางฮาร์ดแวร์

รูปแบบการใช้งาน

object.Handshaking [= value]

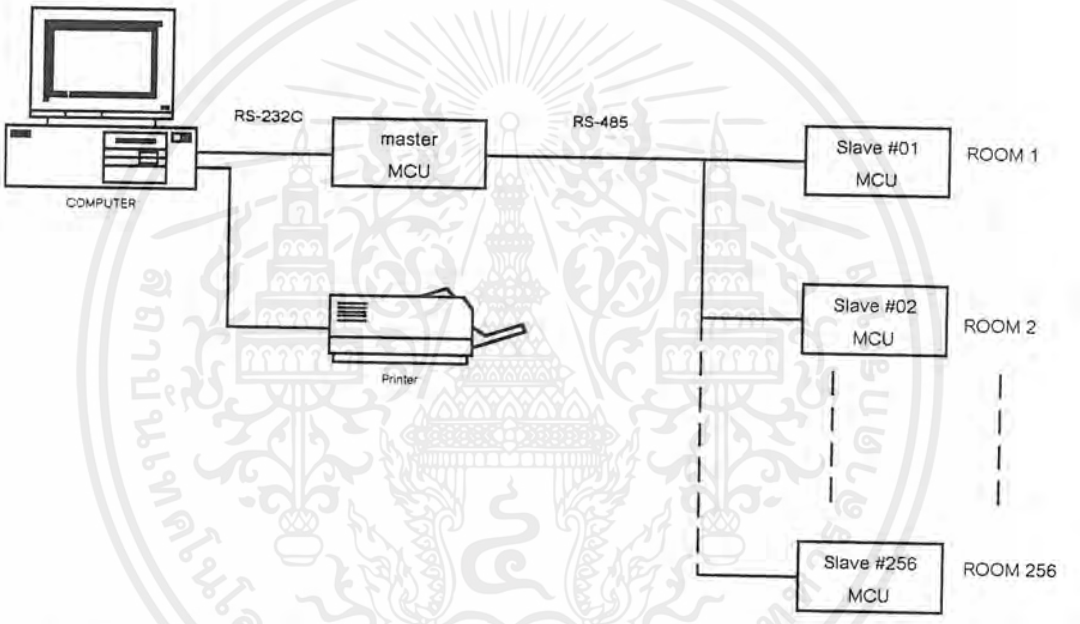
ค่าตัวแปล Value ที่ใช้กำหนดค่ากำหนดได้ 4 รูปแบบด้วยกันคือ

1. comNone ค่าที่กำหนดคือ 0 เป็นการกำหนดให้ไม่มีการแฮนเช็ก(เป็นค่าเริ่มต้น)
2. comXonXoff ค่าที่กำหนดคือ 1 เป็นการกำหนดให้ใช้แฮนเช็กแบบ XON/XOFF
3. comRTS ค่าที่กำหนดคือ 2 เป็นการกำหนดให้ใช้ขา RTS/CTS
4. comRTSXOnXOff ค่าที่กำหนดคือ 3 เป็นการกำหนดให้ใช้ทั้งแบบ RTS และ XON/XOFF

คุณสมบัติ Handshaking ใช้เพื่อกำหนดรูปแบบการสื่อสารภายในระหว่างที่ข้อมูลถูกส่งไปยังบัฟเฟอร์ภาครับ เมื่อข้อมูลตัวอักษรถูกส่งมาถึงพอร์ตอนุกรม อุปกรณ์สื่อสารข้อมูลจะทำการย้ายข้อมูลไปยังบัฟเฟอร์ภาครับ เพื่อที่จะให้โปรแกรมสามารถอ่านค่าไปใช้งานได้ ถ้าไม่มีบัฟเฟอร์ภาครับ โปรแกรมที่ใช้งานจะต้องทำการอ่านค่าข้อมูลโดยตรงจากฮาร์ดแวร์ของพอร์ตอนุกรม ซึ่งผู้ใช้งานจะเกิดปัญหาข้อมูลสูญหายได้ เนื่องจากว่าการเปลี่ยนแปลงของข้อมูลที่ส่งเข้ามามีการเปลี่ยนแปลงอย่างรวดเร็ว

บทที่ 3
การออกแบบและการสร้าง

การออกแบบระบบของเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัตินี้ ประกอบด้วยส่วนที่สำคัญ 3 ส่วนด้วยกันคือ เครื่องคอมพิวเตอร์ทำหน้าที่ในการควบคุม ประมวลผลและแสดงผลทั้งหมด ไมโครคอนโทรลเลอร์ที่ทำหน้าที่เป็นตัวแม่ควบคุมการรับส่งข้อมูลกับตัวลูก และไมโครคอนโทรลเลอร์ที่ติดตั้งอยู่ที่ตู้แช่ภายในห้องพักแต่ละห้องทำหน้าที่ตรวจสอบจำนวนเครื่องดื่มที่ช่องใส่ในตู้แช่ตามที่ได้รับคำสั่งมาจากตัวแม่



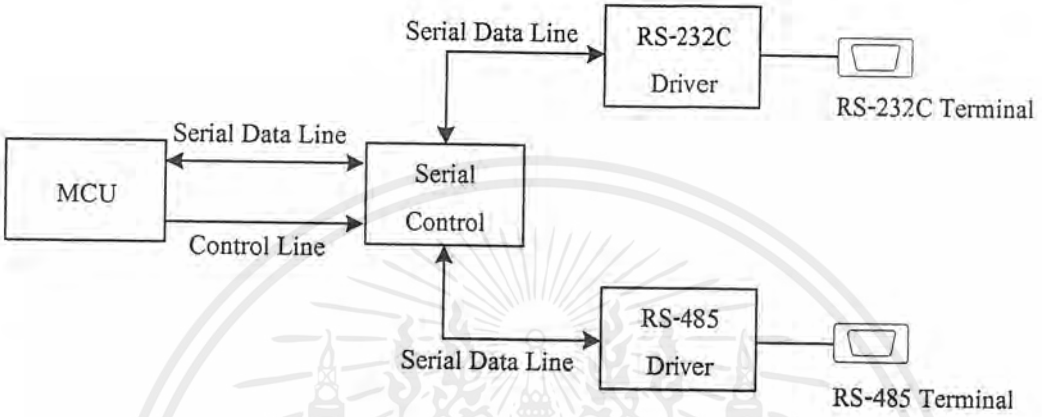
รูปที่ 3.1 แสดงบล็อกไดอะแกรมของระบบเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติ

จากรูปที่ 3.1 ซึ่งแสดงบล็อกไดอะแกรมของทั้งระบบที่สร้างขึ้นโดยมีคอมพิวเตอร์เป็นหน่วยประมวลผลและควบคุมระบบทั้งหมด จะใช้โปรแกรมที่เขียนขึ้นเองโดยใช้โปรแกรม Microsoft Visual Basic 6 เขียนขึ้น ในการควบคุมและส่งผ่านข้อมูลไปยังห้องใด ๆ นั้นจะทำการส่งผ่านข้อมูลไปยังตัวแม่ ซึ่งทำหน้าที่เป็นเสมือนเป็นสถานีแม่ข่ายรับและส่งข้อมูลจากคอมพิวเตอร์และส่งต่อไปยังสถานีลูกข่ายซึ่งก็คือตัวลูก ที่ติดตั้งอยู่ในห้องพักแต่ละห้อง จากนั้นตัวแม่ ก็จะประมวลผลและทำการติดต่อไปที่ตัวลูกตัวที่ต้องการตามที่ได้รับคำสั่งมาจากคอมพิวเตอร์ เพื่อควบคุม หรือร้องขอข้อมูลจากห้องนั้น

3.1 ส่วนของตัวแม่

ในส่วนของตัวแม่ซึ่งทำหน้าที่ในการรับคำสั่งที่ใช้ในการควบคุมจากคอมพิวเตอร์ แล้วทำการแล้วทำการส่งคำสั่งนั้นต่อไปยังตัวลูกตามค่าตำแหน่งที่ได้รับจากคอมพิวเตอร์ แล้วรอรับข้อมูลจากตัวลูกที่ส่งกลับมาพร้อมกับตรวจสอบความถูกต้องของข้อมูลด้วยการใช้เทคนิคการตรวจสอบผลรวมของข้อมูล

ทั้งหมดที่ส่งกลับมาว่ามีค่าตรงกันหรือไม่ ถ้าข้อมูลชุดนั้นถูกต้องก็จะส่งต่อไปยังคอมพิวเตอร์ต่อไป แต่ถ้าข้อมูลนั้นไม่ถูกต้องก็จะส่งรหัสความผิดพลาดไปยังคอมพิวเตอร์แล้วก็รอรับคำสั่งจากคอมพิวเตอร์ต่อไปเรื่อย ๆ

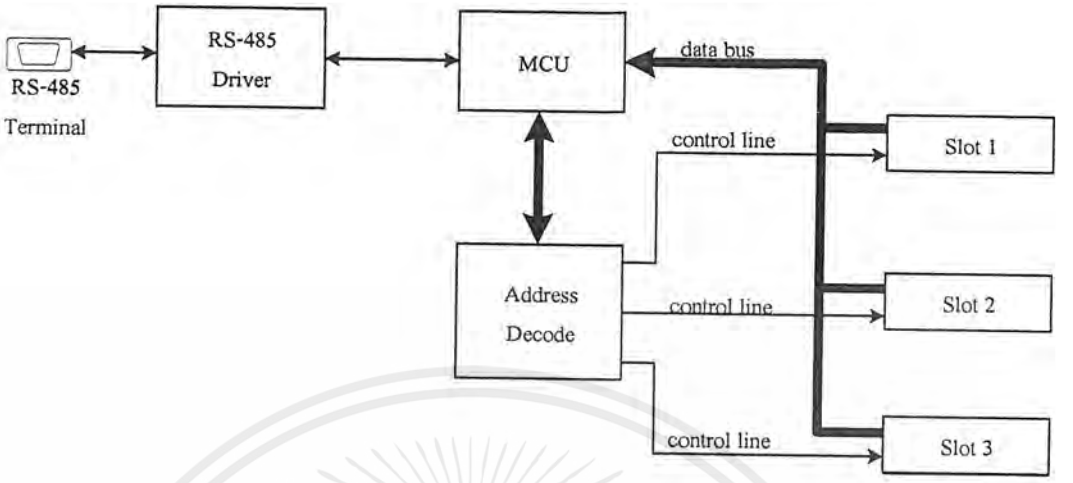


รูปที่ 3.2 แสดงบล็อกไดอะแกรมของตัวแม่

จากรูปที่ 3.2 เป็นบล็อกไดอะแกรมของไมโครคอนโทรลเลอร์ที่เป็นตัวแม่ ซึ่งมีส่วนประกอบดังนี้คือ หน่วยความจำ (Memory) สำหรับเก็บข้อมูลของห้องว่าได้มีห้องใดบ้างที่เช็คอินเข้ามาแล้ว ส่วนไมโครคอนโทรลเลอร์ (MCU) ทำหน้าที่ประมวลผลกลางในส่วนของตัวแม่และสั่งการควบคุมตัวลูกแต่ละตัวรวมทั้งการควบคุมการรับส่งข้อมูลระหว่างคอมพิวเตอร์กับการรับส่งข้อมูลระหว่างตัวลูก ส่วนการขับสัญญาณแบบ RS-232 (RS-232 Driver) ทำหน้าที่แปลงระดับสัญญาณดิจิตอลที่มีระดับแรงดันไฟฟ้า 5 โวลท์ไปเป็นระดับแรงดันไฟฟ้ามาตรฐาน RS-232 ส่วนการขับสัญญาณแบบ RS-485 (RS-485 Driver) ทำหน้าที่แปลงระดับสัญญาณดิจิตอลที่มีระดับแรงดัน 5 โวลท์ไปเป็นระดับแรงดันไฟฟ้ามาตรฐาน RS-485 ส่วนควบคุมการสื่อสารอนุกรม (Driver Control) ทำหน้าที่ในการเลือกที่จะติดต่อกับคอมพิวเตอร์หรือตัวลูก โดยการเลือกที่จะติดต่อกับคอมพิวเตอร์จะควบคุมให้ข้อมูลที่รับส่งไปในเส้นทางที่ต่อกับ RS-232 ถ้าต้องการที่จะติดต่อกับตัวลูกก็จะควบคุมให้ข้อมูลไปในเส้นทางที่ต่อกับ RS-485

3.2 ส่วนของตัวลูก

ในส่วนของตัวลูกจะทำหน้าที่ในการตรวจสอบจำนวนของเครื่องดื่มที่อยู่ตามช่องใส่ต่าง ๆ โดยแบ่งช่องใส่เป็น 6 แถว โดยแบ่งใส่เครื่องดื่มแถวละ 1 ชิ้น ซึ่งจะใส่ได้ทั้งหมด 6 ชนิด ในการตรวจสอบในแต่ละครั้งจะต้องรอคำสั่งจากตัวแม่ส่งคำสั่งมาก่อน แล้วจึงจะกระทำตามคำสั่งนั้น แล้วส่งข้อมูลกลับไปยังตัวแม่เพื่อส่งต่อไปยังคอมพิวเตอร์



รูปที่ 3.3 แสดงบล็อกไดอะแกรมของตัวลูก

จากรูปที่ 3.3 เป็นบล็อกไดอะแกรมของตัวลูก ซึ่งประกอบด้วยส่วนต่างๆ ดังนี้คือ ส่วนประมวลผลกลางทำหน้าที่ในการประมวลผลและควบคุมการทำงานของตัวลูกทั้งหมด ส่วนของการขับระดับสัญญาณลอจิก TTL เป็นระดับสัญญาณตามมาตรฐาน RS-485 ส่วนของการถอดรหัสตำแหน่งพอร์ต (Address Decode) ทำหน้าที่ในการควบคุมตำแหน่งของช่องใส่ในการตรวจสอบจำนวนเครื่องดื่มว่าต้องการอ่านข้อมูลที่ช่องใส่ใด โดยจะอ่านค่าเข้ามาทางสายข้อมูล (Data bus) และส่วนสุดท้ายคือส่วนของช่องใส่เครื่องดื่มมีอยู่ด้วยทั้งหมด 3 ช่องคือ Slot 1, Slot 2 และ Slot 3 โดยสามารถใส่ได้ช่องละ 8 ขวดหรือกระป๋อง (ชิ้น)

3.3 หลักการทำงานและวงจร

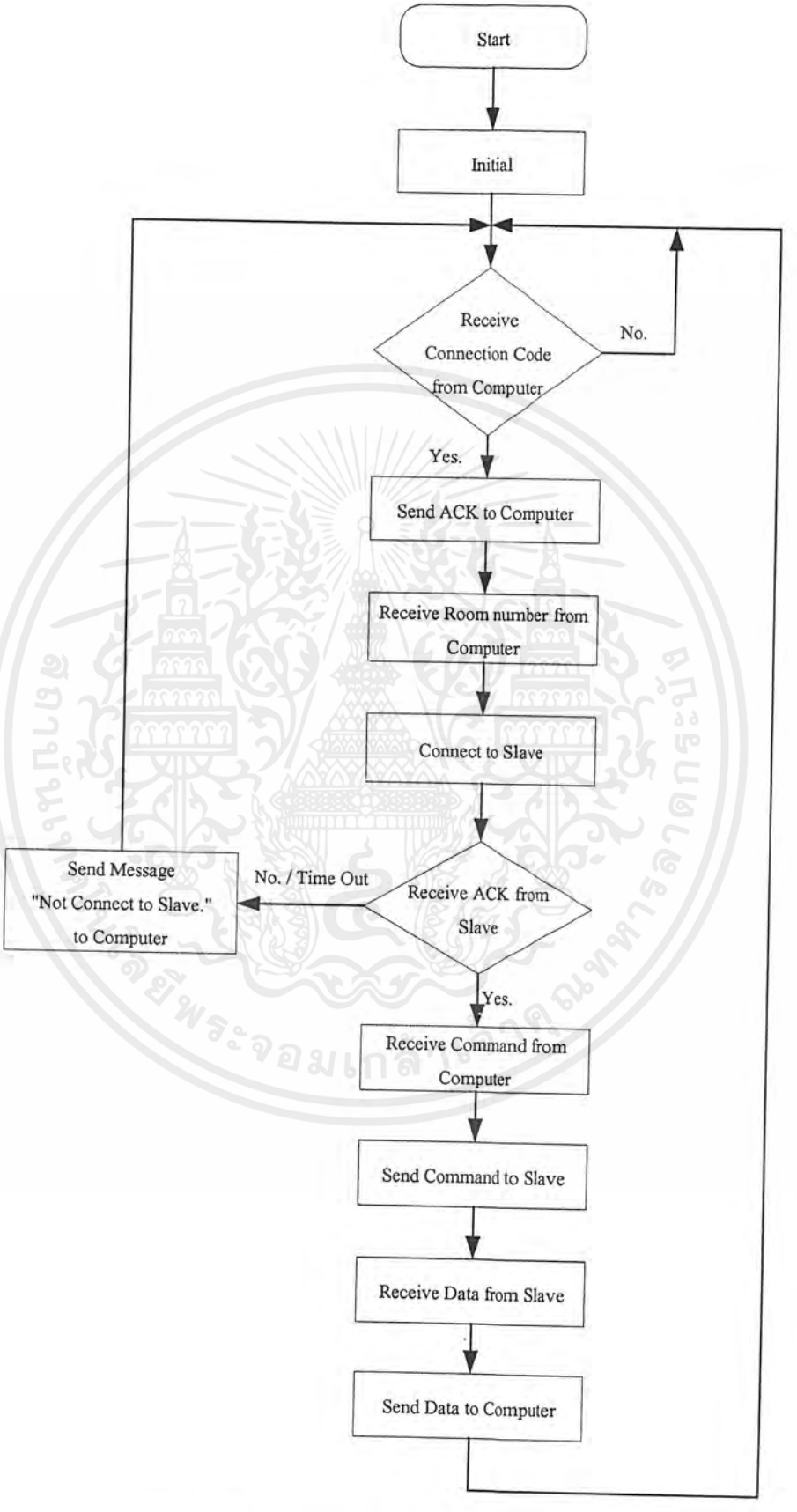
3.3.1 หลักการทำงานของตัวแม่

เริ่มต้นหลังจากรีเซ็ตระบบ จะเป็นการกำหนดค่าเริ่มต้นต่าง ๆ ของเครื่อง เช่นอัตราความเร็วในการรับส่งข้อมูล (Baud Rate) ในโครงงานนี้จะกำหนดอัตราความเร็วที่ 9600 บอด ก็จะรอรับรหัสคำสั่งและตำแหน่งของตัวลูกที่ต้องการที่จะขอข้อมูล โดยจะรอรับรหัสคำสั่งการเริ่มต้นติดต่อจากคอมพิวเตอร์ คือ รหัสแอสกีตัวที่ 32 (SP) หรือ ค่า 20 ในฐาน 16 จากนั้นก็จะส่งรหัสยืนยันการตอบรับกลับไปคือรหัสแอสกีตัวที่ 6 (ACK) และก็จะรอรับรหัสตำแหน่งของตัวลูกที่ต้องการคือหมายเลขห้องที่ต้องการที่จะดูข้อมูลนั่นเอง แล้วจะทำการเรียกไปยังตำแหน่งของตัวลูกที่ได้รับมาพร้อมกับรอรับรหัสยืนยันการตอบรับโดยกำหนดระยะเวลาเอาไว้ช่วงหนึ่งในการรอคอยการตอบรับ (Time out) เมื่อถึงกำหนดเวลาที่ตั้งไว้แล้วไม่มีการตอบรับจากตัวลูกก็จะส่งรหัสคำสั่งไปยังคอมพิวเตอร์เพื่อบอกว่าไม่สามารถติดต่อกับตัวลูกตัวนั้นได้ในขณะนี้เพื่อให้ผู้ที่ดูแลเครื่องอยู่ทำการตรวจสอบเครื่องนั้นๆ หรือสายสัญญาณที่ต่อไปยังตัวลูกตัวนั้น ในกรณีที่ทำการติดต่อไปยังตัวลูกตัวนั้นได้ก็จะส่งรหัสคำสั่งอีกตัวหนึ่ง ที่บอกว่าสามารถทำการติดต่อกับตัวลูกตัวนั้นได้กลับไปพร้อมกับรอรับรหัสคำสั่งอีกตัวหนึ่งแล้วทำการถ่ายทอดคำสั่งนั้นไปยังตัวลูกเพื่อที่

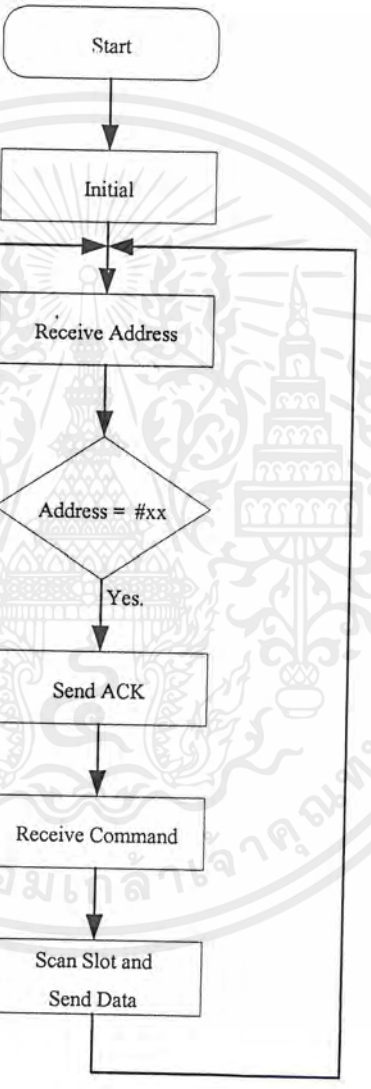
จะให้ตัวลูกตัวนั้นทำการตรวจสอบจำนวนเครื่องดื่มที่อยู่ในช่องใส่แต่ละช่องว่ามีเหลืออยู่เท่าไรแล้วส่งข้อมูลนั้นกลับไปยังตัวแม่ หรือทำการเคลียร์ค่าที่อยู่ในหน่วยความจำที่ใช้ในการป้องกันการใส่เครื่องดื่มกลับคืนไปแล้วทำการตรวจสอบจำนวนอีกครั้งแล้วส่งข้อมูลนั้นกลับไปยังตัวแม่ เพื่อที่ให้ตัวแม่ส่งข้อมูลที่ได้รับมาจากตัวลูกส่งต่อไปยังคอมพิวเตอร์และรายงานผลทางหน้าจอเพื่อให้ผู้ที่ดูแลอยู่รับทราบสถานะต่างๆ

3.3.2 หลักการทำงานของตัวลูก

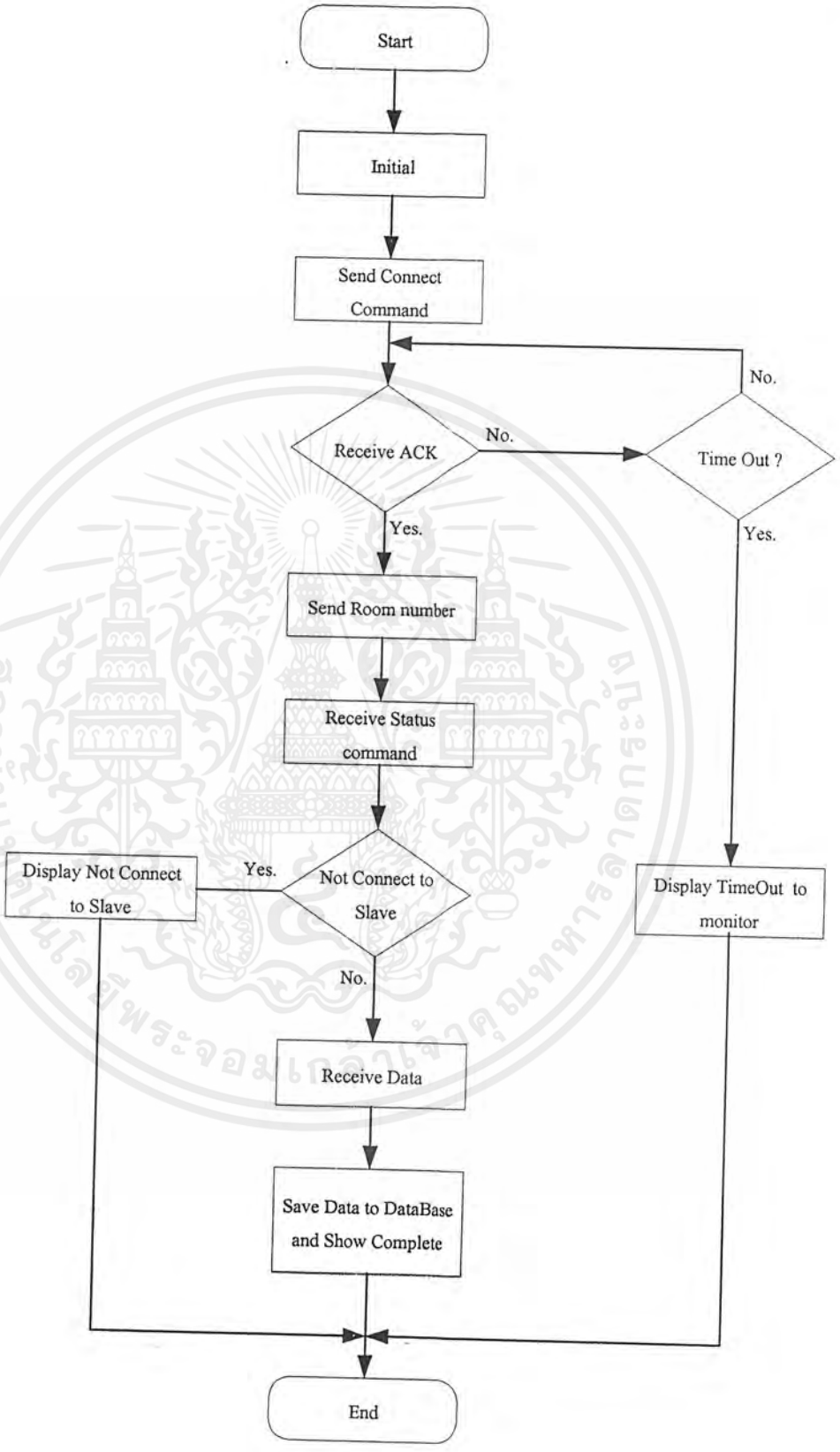
เริ่มต้นหลังจากการรีเซ็ตระบบแล้ว จะทำการกำหนดค่าเริ่มต้นต่างๆ เช่นเดียวกับตัวแม่ หลังจากนั้นก็จะเข้าสู่การรอรับข้อมูลจากตัวแม่อยู่ตลอดเวลา เมื่อตัวแม่มีการส่งข้อมูลเกิดขึ้นโดยจะส่งตำแหน่งของตัวลูกออกมาก่อน ซึ่งตัวลูกก็จะทำการตรวจสอบว่าตำแหน่งนั้นตรงกับของตนเองหรือไม่ ถ้าไม่ใช่ก็จะไม่สนใจในข้อมูลชุดนั้น แล้วก็กลับไปทำงานในสภาวะที่รอรับข้อมูลต่อไป แต่ถ้าข้อมูลชุดนั้นเป็นของตนเองก็จะส่งรหัสเพื่อบอกการยืนยันการตอบรับกลับไปยังตัวแม่เพื่อบอกว่าพร้อมที่จะรับคำสั่งแล้วก็จะรอรับคำสั่งจากตัวแม่ต่อไป เมื่อได้รับคำสั่งจากตัวแม่แล้วก็จะทำตามคำสั่งนั้น แล้วส่งผลที่ได้จากกลับไปยังตัวแม่ แล้วก็กลับเข้าสู่สภาวะรอรับตำแหน่งต่อไป



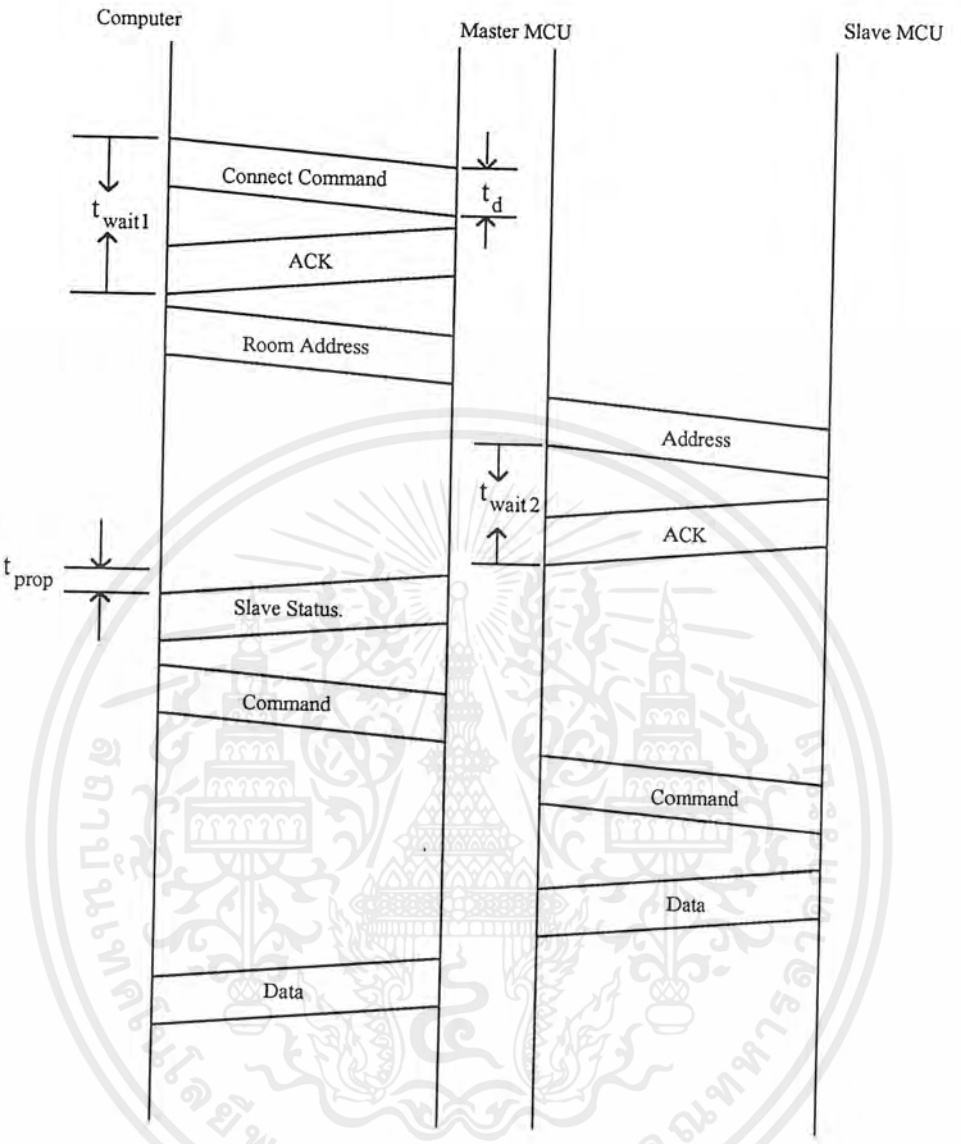
รูปที่ 3.4 แสดงโฟลว์ชาร์ทการทำงานของตัวแม่



รูปที่ 3.5 แสดงโฟลว์ชาร์ทในส่วนของตัวลูก

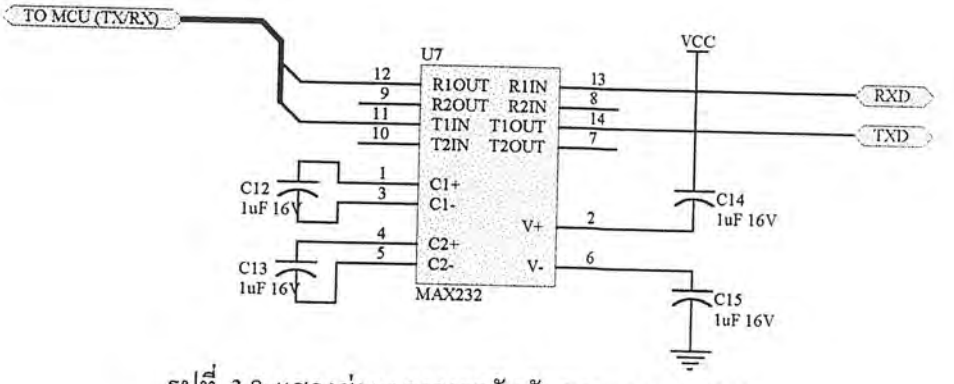


รูปที่ 3.6 แสดงโฟลว์ชาร์ทการทำงานของคอมพิวเตอร์



รูปที่ 3.7 แสดงช่วงเวลาในการรับส่งข้อมูลทั้งระบบ

3.3.3 ส่วนของวงจรขับสัญญาณแบบ RS-232

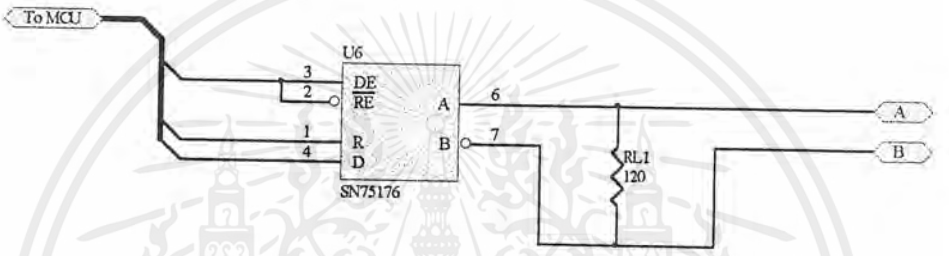


รูปที่ 3.8 แสดงส่วนของวงจรขับสัญญาณแบบ RS-232

จากรูปที่ 3.8 เป็นส่วนของวงจรขับสัญญาณแบบ RS-232 ที่ออกแบบขึ้นโดยใช้ไอซีเบอร์ MAX232 ซึ่งเป็นไอซีสำเร็จรูปเพราะใช้แหล่งจ่ายไฟ +5 โวลท์ เพียงแหล่งเดียวก็สามารถขับสัญญาณแบบลอจิก "1" และ "0" ของตัวไอซีประเภท TTL ที่มีระดับแรงดัน +5 โวลท์และ 0 โวลท์ตามลำดับเป็นระดับแรงดันประมาณ -9 โวลท์สำหรับลอจิก "1" และ +9 โวลท์สำหรับลอจิก "0"

ในส่วนของการเชื่อมต่อเข้ากันขั้วต่อแบบ DB9 จะใช้ขา 13 (R1IN) ของไอซีต่อกับขา 2 (TXD) ของ DB9 ขา 14 (T1OUT) ของไอซีต่อกับขา 3 (RXD) ของ DB9

3.3.4 ส่วนของวงจรขับสัญญาณแบบ RS-485



รูปที่ 3.9 แสดงส่วนของวงจรขับสัญญาณแบบ RS-485

จากรูปที่ 3.9 เป็นวงจรขับสัญญาณแบบ RS-485 ที่ออกแบบขึ้นโดยใช้ไอซีสำเร็จรูปเบอร์ SN75176 ซึ่งเป็นไอซีที่ใช้สำหรับการเปลี่ยนลอจิก "1" และ "0" ของไอซีประเภท TTL เป็นระดับแรงดันไฟฟ้าตามมาตรฐาน RS-485 โดยเฉพาะ ซึ่งการทำงานในส่วนนี้เมื่อต้องการที่จะส่งข้อมูลจะต้องให้ขา 2 และ 3 เป็นลอจิก "1" ส่วนในการรับข้อมูลจะต้องให้เป็นลอจิก "0"

3.3.5 ส่วนของวงจรรวมทั้งหมด

วงจรรวมทั้งหมดจะแบ่งออกเป็น 2 วงจรคือ วงจรในส่วนของตัวแม่กับวงจรในส่วนของตัวลูก สำหรับวงจรในส่วนของตัวแม่ประกอบด้วยส่วนต่าง ๆ ดังนี้คือ

- ตัวไมโครคอนโทรลเลอร์เบอร์ AT89C52 ซึ่งทำหน้าที่เป็นตัวประมวลผลและควบคุมตัวแม่ทั้งหมด
- ไอซีเบอร์ 74HC573 ทำหน้าที่ในการแลตช์ (Latch) แอดเดรสไบต์ต่ำในการเชื่อมหน่วยความจำข้อมูล หน่วยความจำโปรแกรม หรืออุปกรณ์ไอโอพอร์ทภายนอก
- หน่วยความจำโปรแกรมใช้ในการเก็บโปรแกรมที่ใช้ในการทำงาน ในกรณีที่ใช้โปรแกรมภายนอก
- หน่วยความจำข้อมูลใช้สำหรับเก็บข้อมูลของห้องต่าง ๆ
- ไอซีเบอร์ 74LS244 ทำหน้าที่ในการสวิทช์เลือกใช้พอร์ทอนุกรม RS-232 หรือ RS-485
- ไอซีเบอร์ MAX232 ทำหน้าที่ขับระดับสัญญาณ RS-232
- ไอซีเบอร์ DS75176 ทำหน้าที่ขับระดับสัญญาณ RS-485

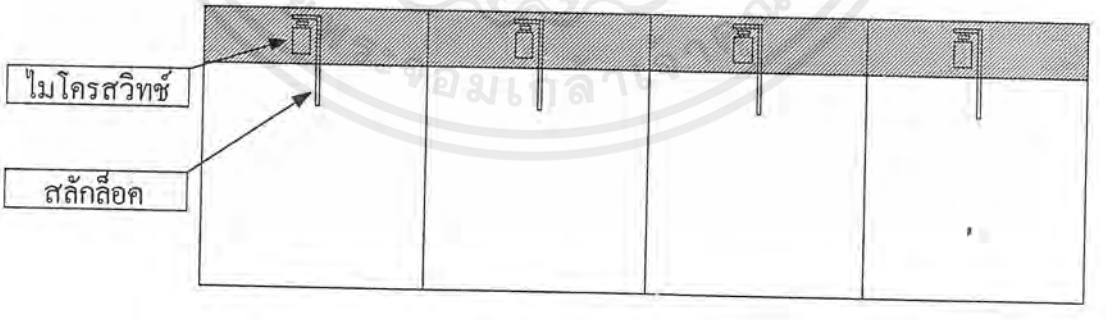
สำหรับวงจรในส่วนของตัวลูกจะแยกออกเป็นสองส่วน คือ ส่วนควบคุม และส่วนของช่องใส่เครื่องดื่ม สำหรับส่วนของส่วนควบคุมนั้นจะคล้ายกันกับวงจรของตัวแม่ แตกต่างกันที่ตัวลูกไม่มีตัวขับสัญญาณแบบ RS-232 แต่จะเป็นตัวขับแบบ RS-485 แทน และจะมีเพิ่มขึ้นมาในส่วนของดิปสวิทช์ (Dip Switch) ที่ใช้เลือกตำแหน่งของตัวเองในแต่ละตัว

ส่วนของช่องใส่เครื่องดื่มจะมีอยู่ 3 ช่องด้วยกันโดยสามารถใส่ได้ช่องละ 8 ชิ้น ซึ่งจะสามารถเพิ่มจำนวนของช่องใส่ได้อีก 5 ช่อง ช่องละ 8 ชิ้นเช่นกัน ในแต่ละช่องจะมีสวิทช์อยู่ 8 ตัวติดอยู่ที่ภายในช่องใส่เพื่อใช้ในการตรวจสอบว่ามีช่องไหนชิ้นไหนได้เอาออกไปแล้วบ้าง

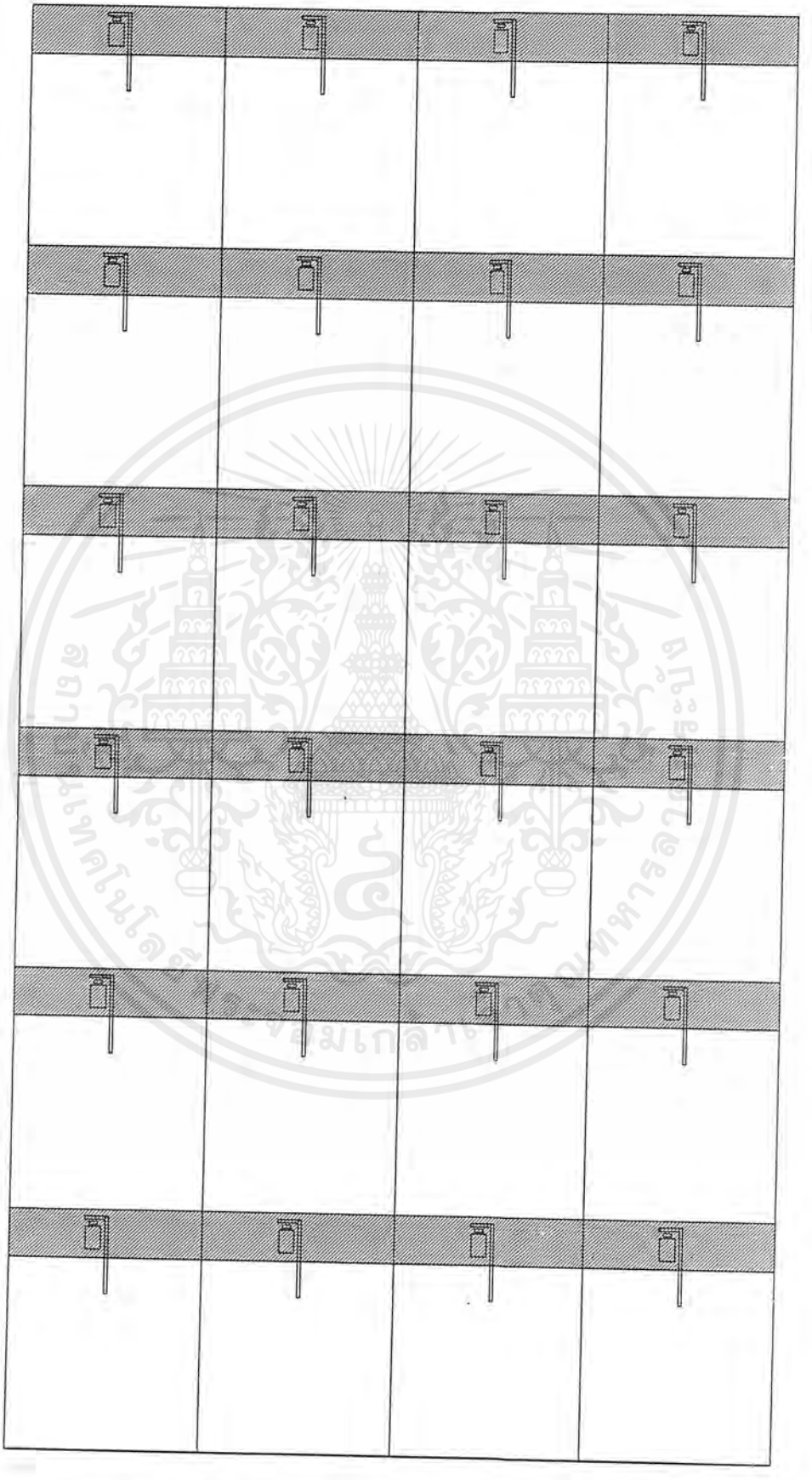
สำหรับในส่วนของการรับส่งข้อมูลระหว่างคอมพิวเตอร์กับตัวแม่และตัวแม่กับตัวลูกในรูปที่ 3.8 เป็นช่วงเวลาตั้งแต่คอมพิวเตอร์เริ่มติดต่อกับตัวแม่

3.4 ส่วนตรวจสอบจำนวนเครื่องดื่ม

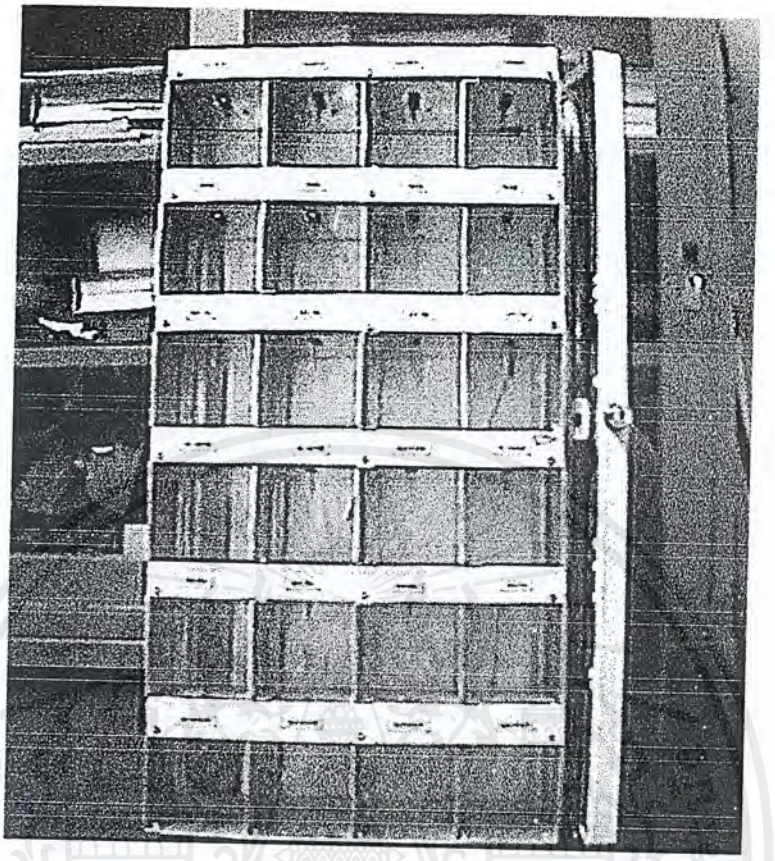
ในส่วนของการตรวจจำนวนเครื่องดื่ม จะใช้ไมโครสวิทช์ติดตั้งไว้ที่ช่องใส่แต่ละช่องเพื่อตรวจสถานะของเครื่องดื่มที่ใสอยู่ภายในช่องนั้น ซึ่งในสถานะที่มีเครื่องดื่มใส่อยู่นั้นจะมีสถานะของลอจิกเป็น "0" และเมื่อไม่มีเครื่องดื่มอยู่ก็จะมีสถานะเป็น "1" ซึ่งมีตัวลูกเป็นตัวที่คอยตรวจสอบสถานะนี้ และจะอ่านค่าสถานะเมื่อมีคำสั่งมาจากตัวแม่เท่านั้น โดยอ่านค่าเข้ามาทางพอร์ทของตัวลูกครั้งละ 8 ช่องหรือ 8 บิต เป็นจำนวน 3 ครั้งก็จะได้จำนวนของสถานะของช่องใส่ที่อ่านสถานะเข้ามาครบทั้ง 24 ช่องแล้วทำการส่งไปยังตัวแม่ต่อไป ดังรูปที่ 3.10 แสดงช่องใส่เครื่องดื่มจำนวน 1 แถว จะสามารใส่เครื่องดื่มได้จำนวน 4 กระป๋อง ซึ่งการที่ทำให้แต่ละห้องสามารถใส่ได้จำนวน 24 กระป๋องนั้นจะนำช่องใส่จำนวน 6 แถวมาเชื่อมต่อกันดังรูปที่ 3.11



รูปที่ 3.10 แสดงช่องใส่เครื่องดื่มจำนวน 1 แถว (4 ช่อง) ที่ออกแบบขึ้น

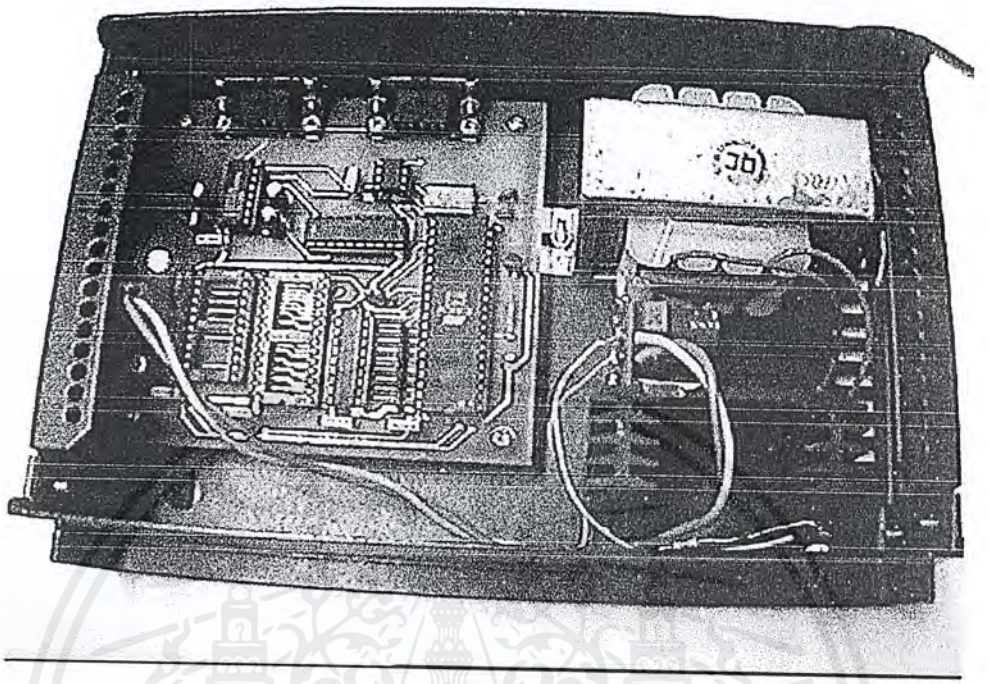


รูปที่ 3.11 แสดงช่องใส่เครื่องดื่มจำนวน 24 ช่องที่ออกแบบขึ้น

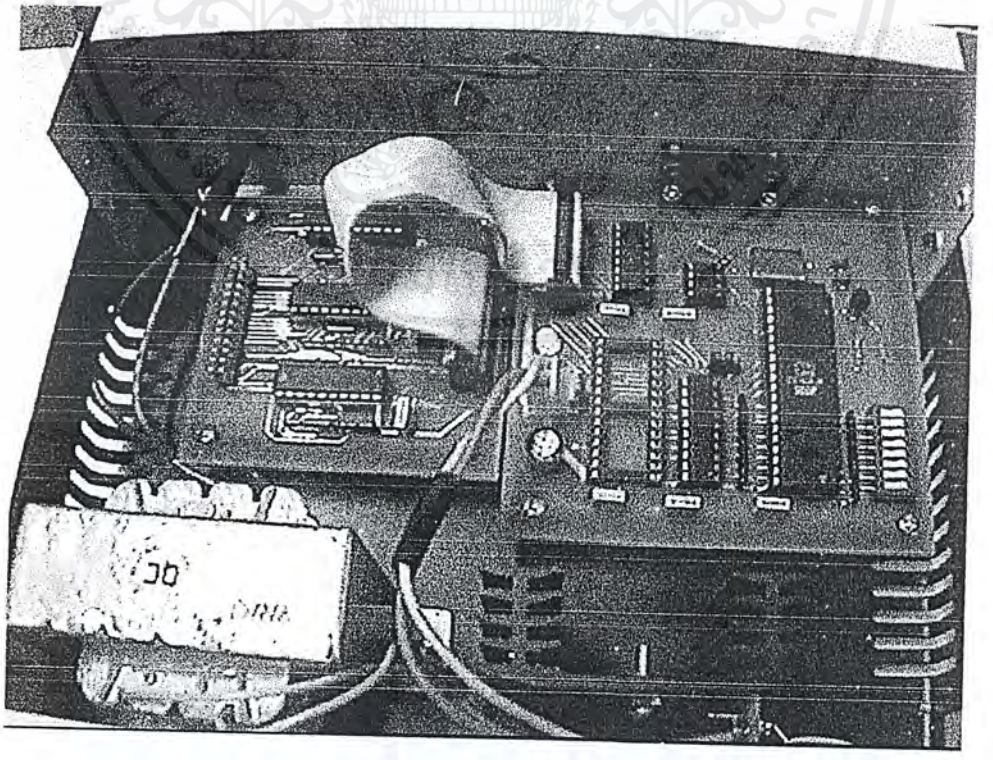


รูปที่ 3.12 แสดงช่องใส่เครื่องดื่มที่ทำสำเร็จแล้ว

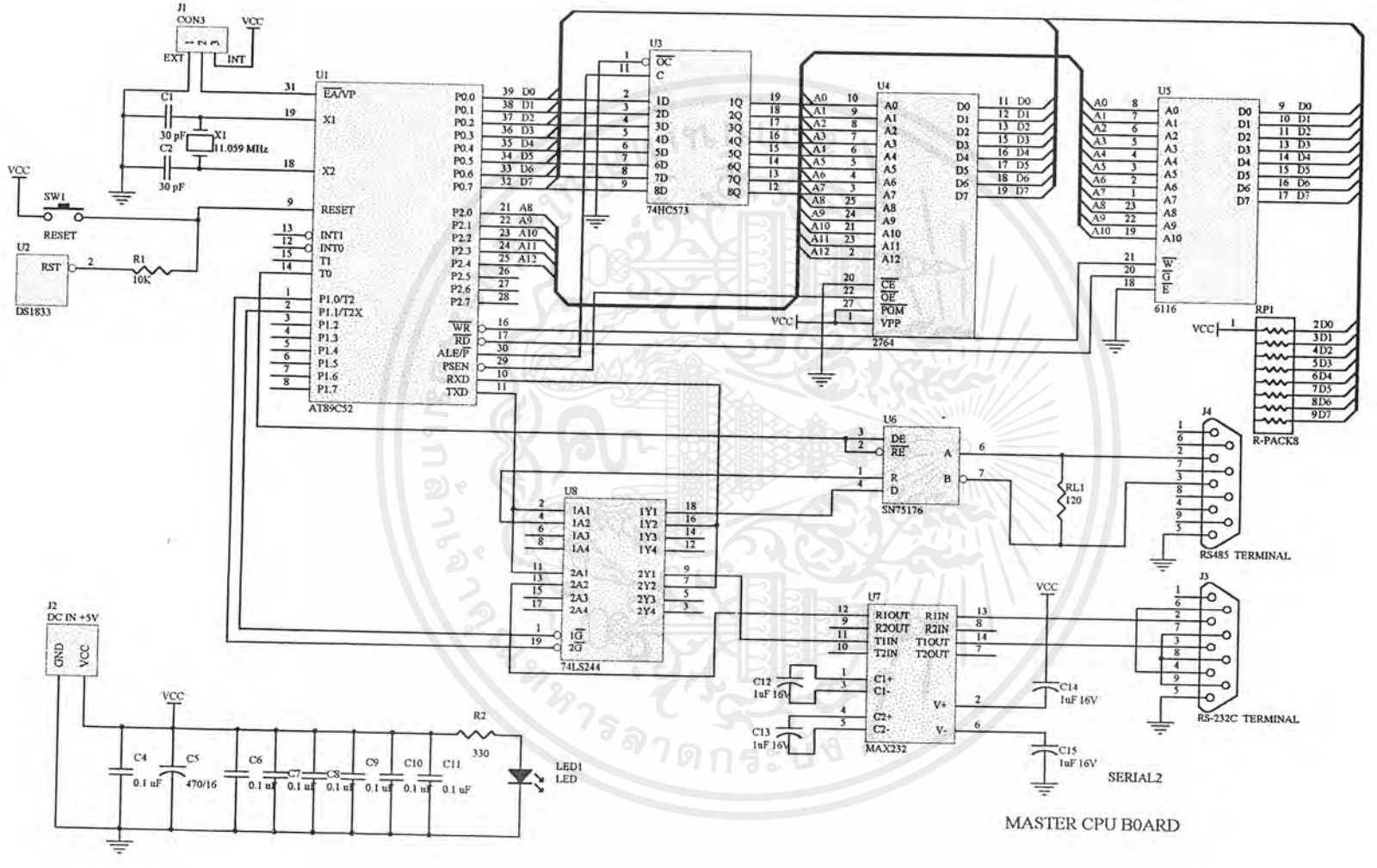
รูปที่ 3.13 แสดงช่องใส่เมื่อใส่กระป๋องเครื่องดื่มแล้ว



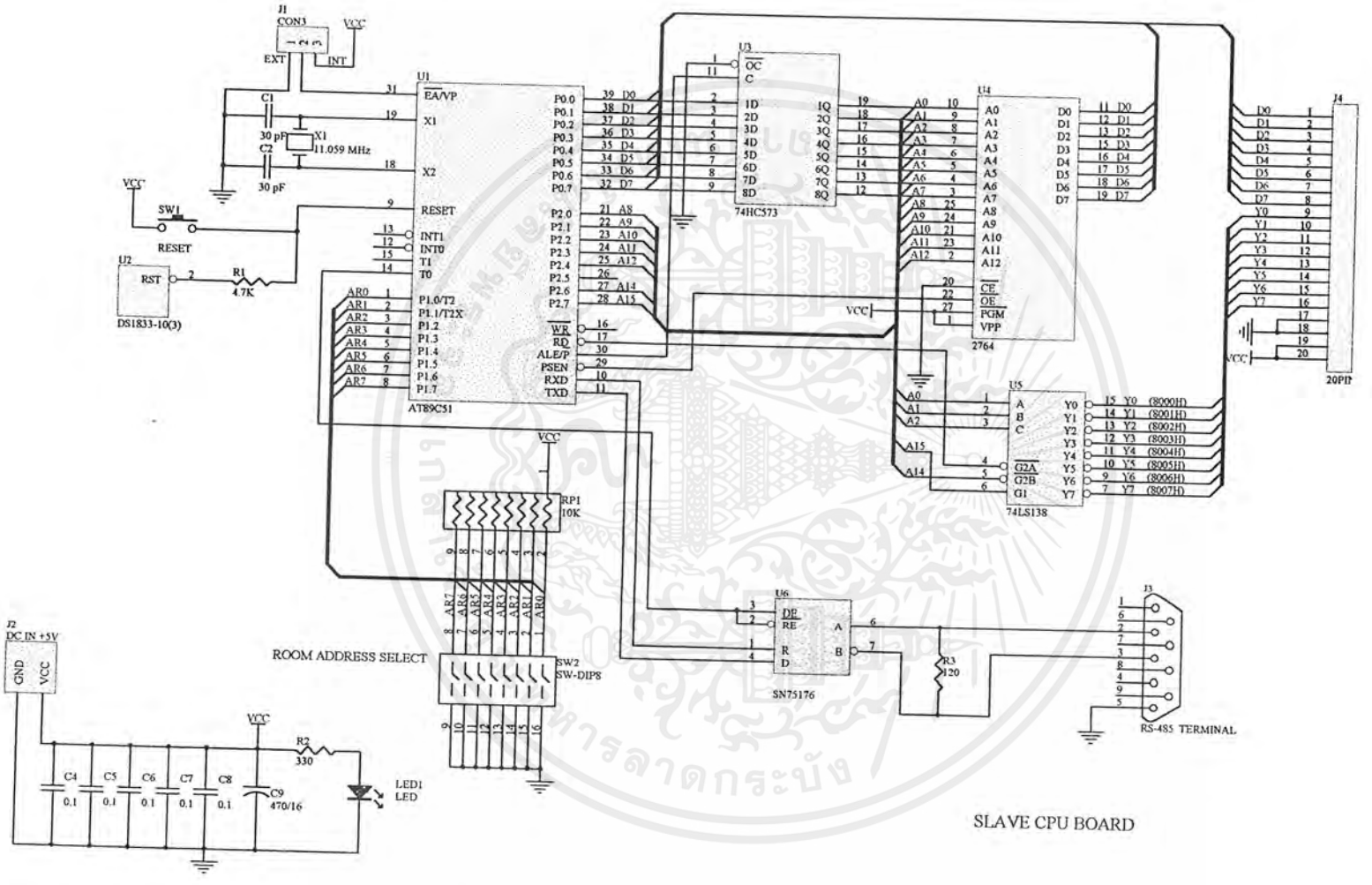
รูปที่ 3.14 แสดงภาพของตัวแม่เมื่อประกอบเสร็จ



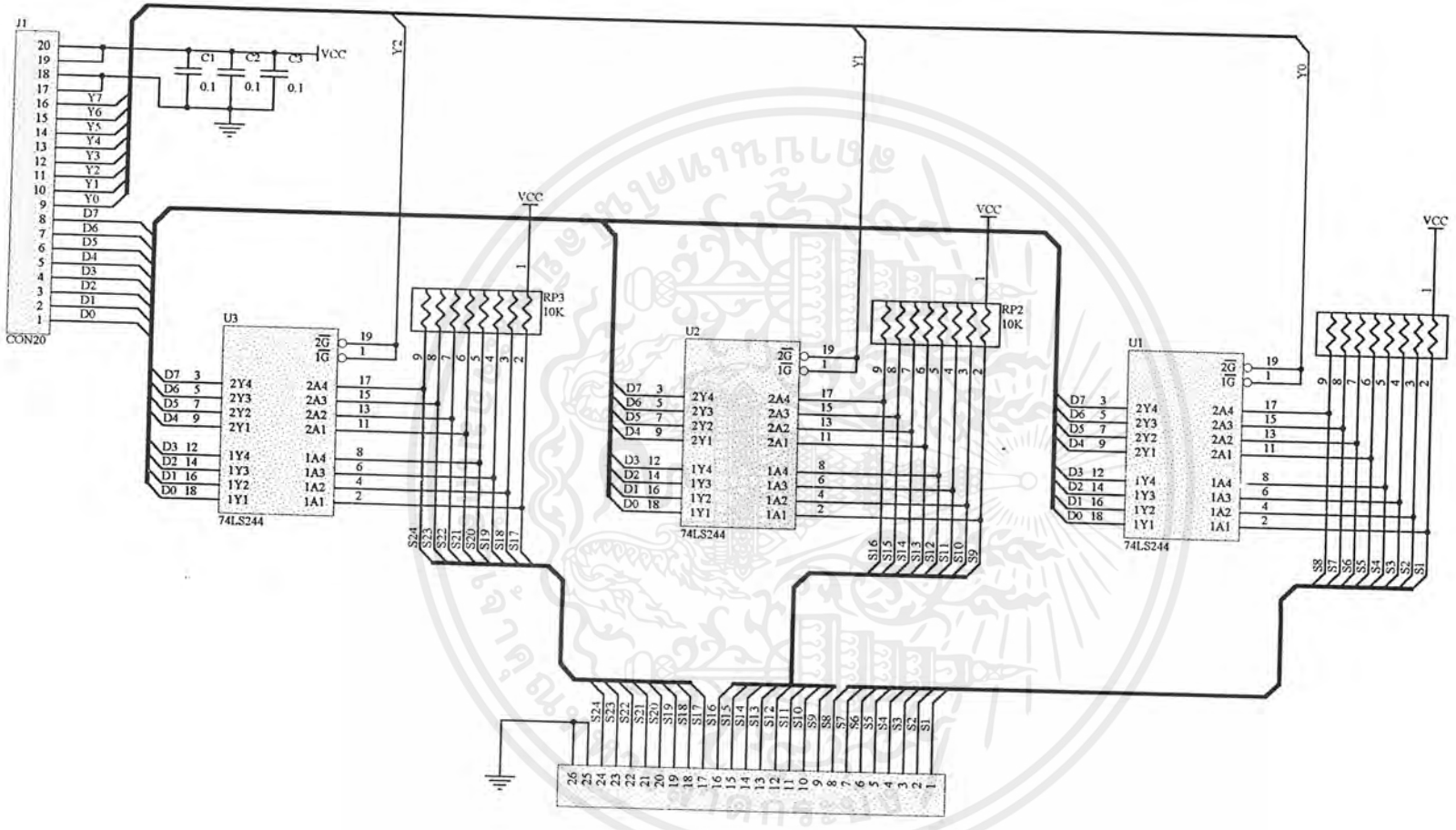
รูปที่ 3.15 แสดงภาพของตัวลูกที่ประกอบเสร็จ



รูปที่ 3.16 แสดงวงจรรวมของตัวแม่



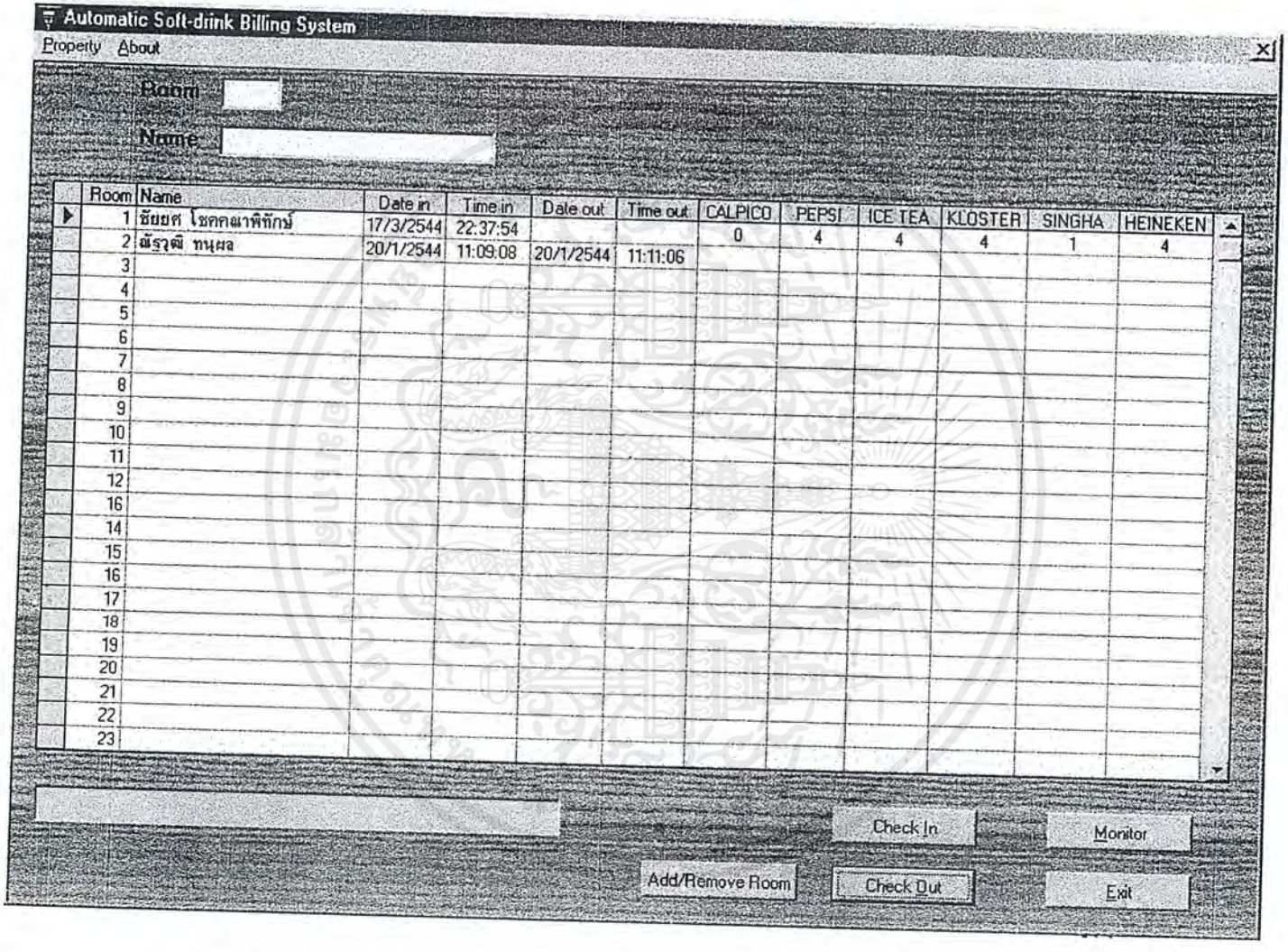
รูปที่ 3.17 แสดงวงจรรวมส่วนควบคุมของตัวลูก



รูปที่ 3.18 แสดงวงจรส่วนช่องใส่เครื่องดื่มของตัวลูก

3.5 ส่วนของโปรแกรมที่ใช้ในการควบคุมการทำงานของเครื่อง

สำหรับในส่วนของโปรแกรมที่ใช้ในการควบคุมนี้ จะไม่ขอกล่าวถึงโปรแกรมในส่วนย่อยทั้งหมดของตัวแม่ ตัวลูก และคอมพิวเตอร์ ซึ่งแสดงในรูปที่ 3.4 ถึง 3.6 แล้วซึ่งเป็นโฟล์วชาร์ทของโปรแกรมทั้งหมด จะขอกล่าวถึงในส่วนของโปรแกรมที่ควบคุมและแสดงผลทางหน้าจอคอมพิวเตอร์เท่านั้น โดยการเรียกใช้งานโปรแกรมชื่อ ASDB.EXE (Automatic Soft-Drink Checking and Billing System) ซึ่งพัฒนาขึ้นโดยใช้โปรแกรมไมโครซอฟวิชวลเบสิก เวอร์ชัน 6 (Microsoft Visual Basic 6) และได้แนบแผ่นซีดีสำหรับติดตั้งโปรแกรม ASDB มาพร้อมกับโครงงานนี้แล้ว เมื่อติดตั้งโปรแกรมแล้วเรียกใช้งานก็จะแสดงโลโกของโปรแกรมขึ้นมาก่อนประมาณ 5 วินาทีแล้วจะแสดงฟอร์มหลักของโปรแกรมขึ้นมาดังรูปที่ 3.15 ซึ่งการใช้งานหลัก ๆ มีอยู่ด้วยกัน 6 คำสั่งคือ Add/Remove Room ใช้ในการเพิ่มหรือลดจำนวนห้องที่นำเครื่องนี้ไปติดตั้ง, Check In ใช้ในการบันทึกการเข้าพักลงในฐานข้อมูลและส่งคำสั่งไปยังตัวลูกเพื่อให้ลบข้อมูลเก่าที่มีอยู่ออกไป แล้วทำการเช็คจำนวนเครื่องดื่มที่มีอยู่ว่ามีครบหรือไม่แล้วส่งข้อมูลนั้นกลับมายังคอมพิวเตอร์ ซึ่งปกติจะต้องมีค่าเป็น "0" ทั้งหมดทุกช่อง คือมีเครื่องดื่มอยู่เต็มช่องใส่ก่อนที่จะมีผู้มาเข้าพัก, Check Out ใช้ในการลบการเข้าพักออกจากฐานข้อมูลและส่งคำสั่งไปยังตัวลูกเพื่อเช็คจำนวนเครื่องดื่มครั้งสุดท้ายก่อนที่จะคิดเป็นจำนวนเงินออกมา, Monitor ใช้สำหรับการเช็คจำนวนเครื่องดื่มในแต่ละวัน เพื่อดูว่าห้องแต่ละห้องมีเครื่องดื่มที่ถูกนำไปบริโภคแล้วเป็นจำนวนเท่าใด ซึ่งสามารถเตรียมเครื่องดื่มไปเติมให้เต็มเหมือนเดิม พร้อมกับการยืนยันการเพิ่มจำนวนเครื่องดื่มลงในฐานข้อมูลในแต่ละวันได้ด้วย ส่วนคำสั่งสุดท้ายคือ Close ใช้ในการปิดโปรแกรมนี้



Room	Name	Date in	Time in	Date out	Time out	CALPICO	PEPSI	ICE TEA	KLOSTER	SINGHA	HEINEKEN
1	ชัยยศ โชคคณาพิทักษ์	17/3/2544	22:37:54			0	4	4	4	1	4
2	ลัรวุฒิ ทนุผล	20/1/2544	11:09:08	20/1/2544	11:11:06						
3											
4											
5											
6											
7											
8											
9											
10											
11											
12											
16											
14											
15											
16											
17											
18											
19											
20											
21											
22											
23											

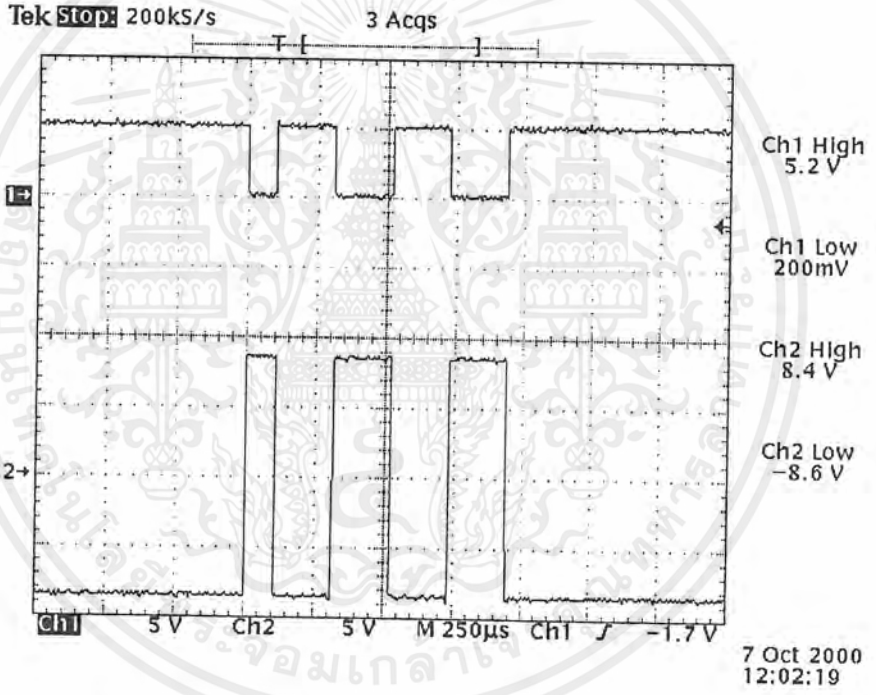
รูปที่ 3.19 แสดงฟอร์มหลักของโปรแกรมเครื่องเช็คจำนวนเครื่องดื่มอัติโนมัติ

บทที่ 4
การทดลองและผลการทดลอง

4.1 การทดลองการส่งสัญญาณดิจิตอลแบบ UART

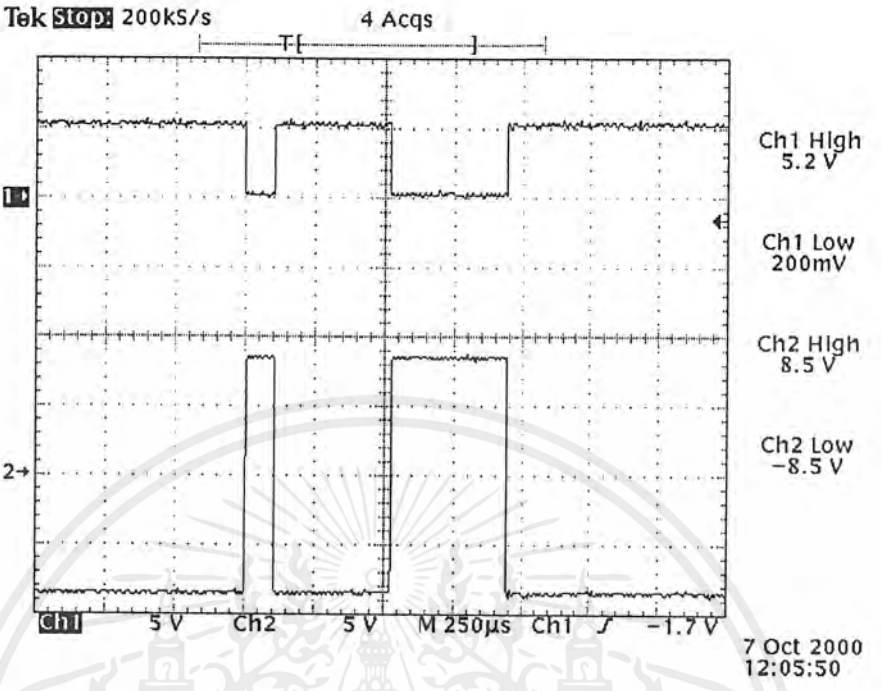
4.1.1 ทดลองวงจรขับสัญญาณแบบ RS-232

การทดลองวงจรขับสัญญาณแบบ RS-232 ทดลองโดยวงจรของตัวแม่ที่สร้างขึ้นแล้วทำการเขียนโปรแกรมส่งข้อมูลออกที่พอร์ทอนุกรมของตัวไมโครคอนโทรลเลอร์ส่งข้อมูลออกมา แล้วใช้สโคปจับสัญญาณที่ขา 14 ของตัวไอซี โดยใช้วงจรของตัวแม่รูปที่ 3.8 ที่ได้ทำการต่อและเขียนโปรแกรมให้ส่งข้อมูลตัวเดียวกันออกมา 1 ไบต์เรื่อย ๆ



รูปที่ 4.1 แสดงระดับสัญญาณดิจิตอล 11001100 ที่ทดลองส่ง
Ch1. แสดงระดับสัญญาณของ TTL
Ch2. แสดงระดับสัญญาณของ RS-232

จากรูปที่ 4.1 และ 4.2 เป็นระดับสัญญาณอนุกรมที่ส่งออกโดยบิตแรกทางซ้ายมือคือบิตที่มีนัยสำคัญน้อยที่สุด (LSB) ซึ่งเป็นบิตแรกที่ถูกชิปข้อมูลออกไปก่อน โดยรูปที่ 4.1 เป็นการส่งข้อมูล 11001100 และรูปที่ 4.2 เป็นการส่งข้อมูล 11110000 จากทั้งสองรูประดับสัญญาณลอจิก "0" ตัวแรกคือบิตเริ่มต้นหรือ Start bit ส่วนบิตถัดมาอีก 8 บิตคือข้อมูลที่ส่ง และบิตสุดท้ายคือบิตปิดท้ายหรือ Stop bit จะเห็นว่าระดับลอจิก "1" ของ RS-232 จะมีค่าเป็นลบประมาณ - 8 โวลท์ และระดับลอจิก "0" จะมีค่าเป็นบวกประมาณ + 8 โวลท์



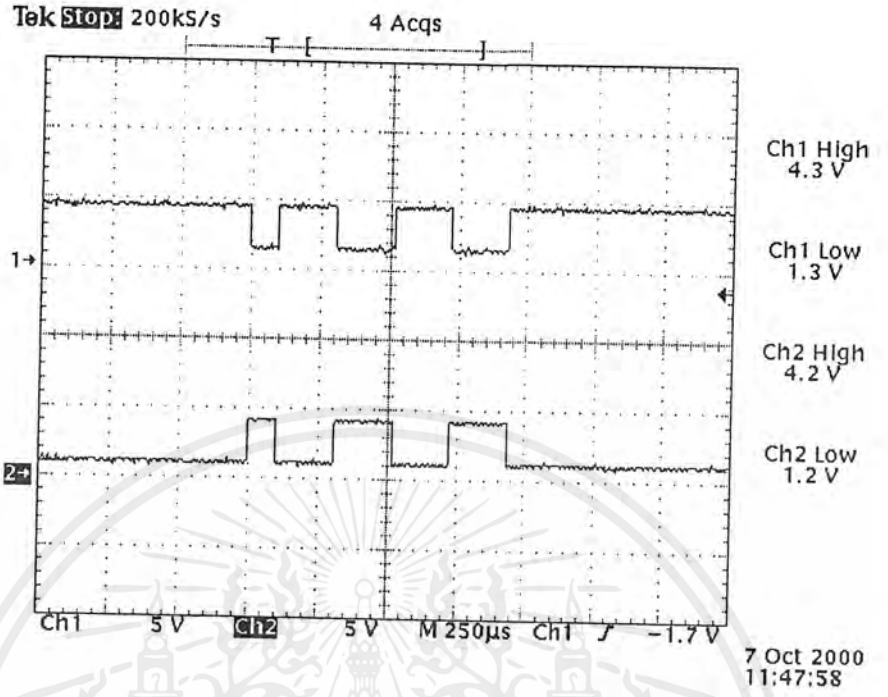
รูปที่ 4.2 แสดงระดับสัญญาณดิจิตอล 11110000 ที่ทดลองส่ง

Ch1. แสดงระดับสัญญาณของ TTL

Ch2. แสดงระดับสัญญาณของ RS-232

4.1.2 ทดลองวงจรขับสัญญาณแบบ RS-485

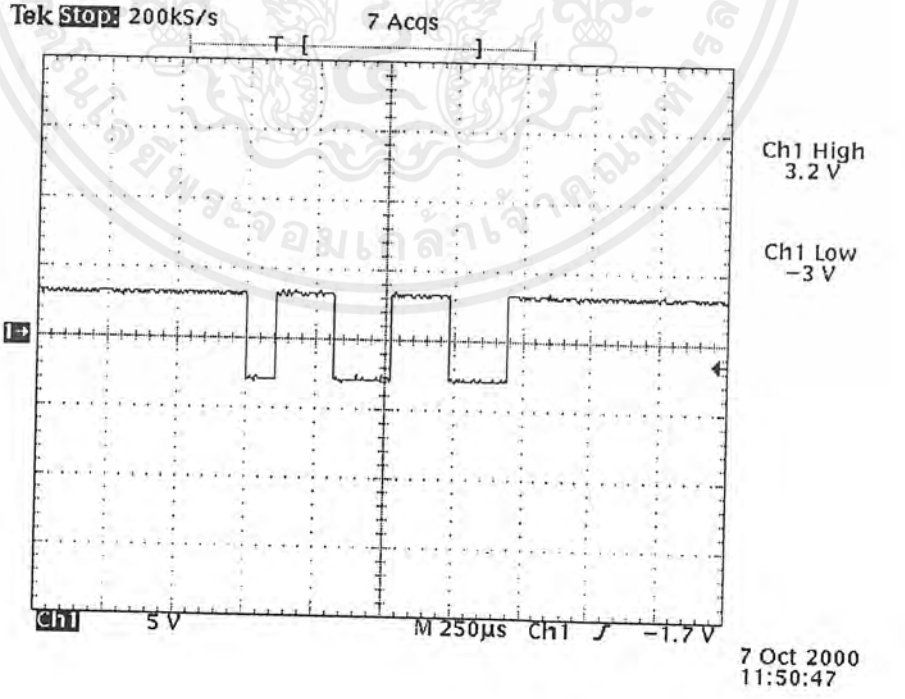
การทดลองวงจรขับสัญญาณแบบ RS-485 จะใช้วงจรของตัวแม่ในการทดสอบและเขียนโปรแกรมส่งข้อมูลเช่นเดียวกันกับการทดลองวงจรขับสัญญาณแบบ RS-232 สำหรับสัญญาณ RS-485 ซึ่งใช้ผลต่างระหว่างขั้วทั้งสอง A กับ B จะมีผลการทดลองอยู่ 3 ลักษณะคือ ระดับแรงดันระหว่างขั้ว A กับกราวด์ ระดับแรงดันระหว่างขั้ว B กับกราวด์ และผลต่างระหว่างขั้วทั้งสอง



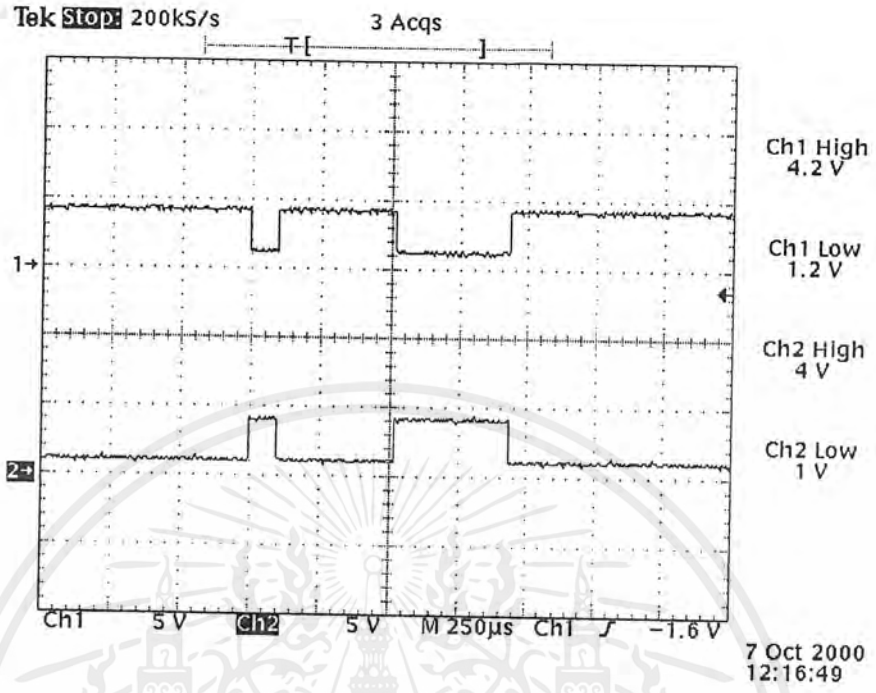
รูปที่ 4.3 แสดงระดับแรงดันของขั้ว A, B กับกราวด์ ของ RS-485 ข้อมูลที่ส่ง 11001100

Ch1. แสดงระดับสัญญาณของขั้ว A กับกราวด์

Ch2. แสดงระดับสัญญาณของขั้ว B กับกราวด์



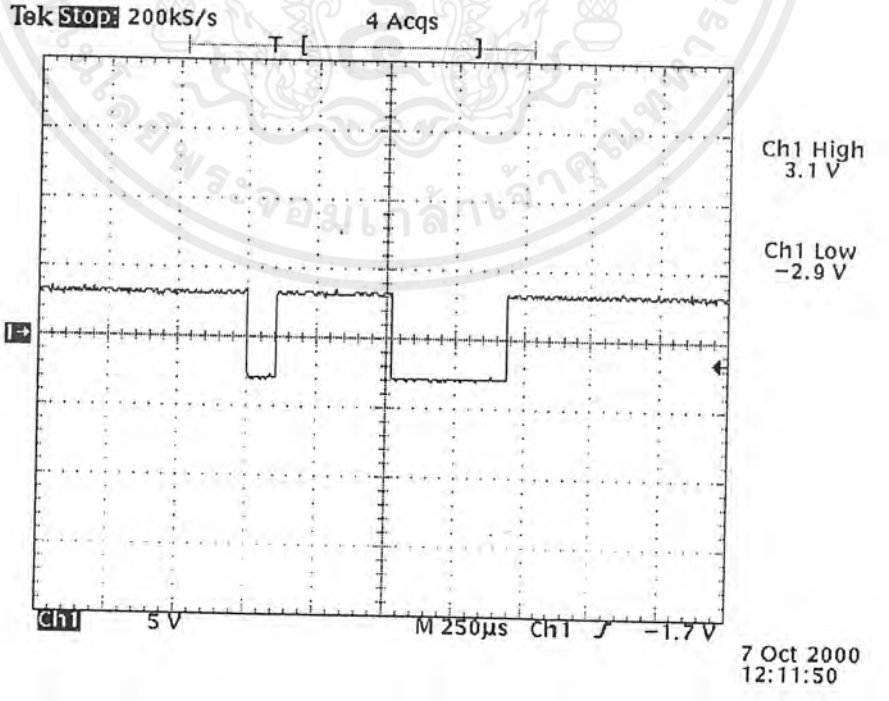
รูปที่ 4.4 แสดงระดับแรงดันระหว่างขั้ว A กับ B (V_{AB}) ของ RS-485 ข้อมูลที่ส่ง 11001100



รูปที่ 4.5 แสดงระดับแรงดันของขั้ว A, B กับกราวด์ ของ RS-485 ข้อมูลที่ส่ง 11110000

Ch1. แสดงระดับสัญญาณของขั้ว A กับกราวด์

Ch2. แสดงระดับสัญญาณของขั้ว B กับกราวด์



รูปที่ 4.6 แสดงระดับแรงดันระหว่างขั้ว A กับ B (V_{AB}) ของ RS-485 ข้อมูลที่ส่ง 11001100

จากรูปที่ 4.3 และรูปที่ 4.4 ซึ่งแสดงระดับสัญญาณเอาท์พุตของตัวขับสัญญาณแบบ RS-485 ซึ่งจากรูปที่ 4.3 จะเห็นว่าระดับแรงดันของขั้ว A (Ch. 1) กับกราวด์ที่ลอจิก "0" จะมีระดับแรงดันประมาณ 1.3 โวลท์ และระดับแรงดันของลอจิก "1" มีค่าประมาณ 4.3 โวลท์ ส่วนระดับแรงดันของขั้ว B (Ch. 2) กับกราวด์ที่ลอจิก "0" มีค่าประมาณ 4.2 โวลท์ ระดับแรงดันของลอจิก "1" มีค่าประมาณ 1.2 โวลท์ ส่วนในรูปที่ 4.4 เป็นผลต่างของขั้วทั้งสองก็จะเห็นว่าที่ระดับของลอจิก "1" มีค่าประมาณ 3.3 โวลท์ และลอจิก "0" มีค่ากลับเป็น -3 โวลท์ ซึ่งก็คือผลต่างของขั้วทั้งสองและสัญญาณแบบ RS-485 ก็คือการสลับขั้วกันระหว่างขั้ว A และขั้ว B คือที่ระดับของลอจิก "1" ผลต่างของแรงดันของขั้ว A เมื่อเทียบกับ B จะมีค่าเป็นบวก และที่ระดับของลอจิก "0" ก็จะได้ผลต่างของขั้ว A เมื่อเทียบกับ B ก็จะมีค่ากลับกันกับลอจิก "1" คือจะมีค่าเป็นลบนั่นเอง

ส่วนในรูปที่ 4.5 และรูปที่ 4.6 เป็นระดับสัญญาณเอาท์พุตของตัวขับสัญญาณแบบ RS-485 เช่นกันเพียงแต่เปลี่ยนค่าของข้อมูลที่ทดลองส่งเป็น 11110000 ซึ่งก็จะได้ผลการทดลองเช่นเดียวกันกับการทดลองที่ได้ในรูปที่ 4.3 และรูปที่ 4.4

4.2 การทดลองการทำงานของเครื่องที่ออกแบบ

จากการทดลองการทำงานของเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัติที่ออกแบบขึ้นนี้ เมื่อทำการติดตั้งโปรแกรมบนระบบปฏิบัติการวินโดวส์แล้ว สามารถเรียกใช้งานได้ที่เมนู Start ของวินโดวส์ เมื่อเรียกใช้งานแล้วก็จะแสดงโลโกที่ทำขึ้นสักครู่หนึ่งก็จะแสดงฟอร์มหลักของโปรแกรมขึ้นมา การทดลองเริ่มจากการติดตั้งช่องใส่เครื่องดื่มเข้ากับตัวลูกทางคอนเน็กเตอร์ 26 ขาที่ออกแบบไว้ และต่อสายจากตัวลูกมายังตัวแม่โดยการทดลองนี้ได้ใช้สายโทรศัพท์ธรรมดาจำนวน 1 คู่ในการเชื่อมต่อ และต่อสายระหว่างตัวแม่เข้ากับคอมพิวเตอร์ทางพอร์ทอนุกรมพอร์ท 1 (COM 1) ซึ่งในโปรแกรมนี้ได้ตั้งค่าเริ่มต้นไว้ที่คอม 1 อยู่แล้ว ในกรณีที่ต่อกับพอร์ทอื่น ก็สามารถเปลี่ยนแปลงได้จากเมนู Property ของโปรแกรมนี้ โดยสามารถเปลี่ยนได้ตั้งแต่พอร์ทคอม 1 ถึงคอม 4 เมื่อต่อสายเรียบร้อยแล้วก็ทำการทดลองโดยใส่เครื่องดื่มให้ครบทุกช่อง แล้วทดสอบโดยป้อนหมายเลขห้องที่ต้องการ (ในที่นี้กำหนดให้ตัวลูกเป็นห้องที่ 1) กับชื่อของผู้เข้าพัก แล้วใช้เม้าท์คลิ๊กที่ Check In ก็จะได้จำนวนเครื่องดื่มที่หน้าจอคอมพิวเตอร์เป็น 0 ทั้งหมดทุกประเภท และทำการทดสอบต่อไปโดย นำเครื่องดื่มออกมาทีละ 1 - 2 กระป๋อง แล้วใช้เม้าท์คลิ๊กที่ Monitor ก็จะแสดงฟอร์มขึ้นอีกฟอร์มหนึ่งโดยจะแสดงเป็นจำนวนของเครื่องดื่มที่ถูกนำออกจากช่องใส่ในแต่ละห้อง ซึ่งการทดลองนี้ได้จำนวนเครื่องดื่มที่แสดงออกมาทางหน้าจอคอมพิวเตอร์เท่ากับจำนวนที่นำออกจากช่องใส่ และได้ทดลองไปเรื่อย ๆ ก็ได้ผลการทดลองคือจำนวนเครื่องดื่มที่แสดงทางหน้าจอกับที่นำออกจากช่องใส่จะมีจำนวนที่เท่ากัน เมื่อได้ผลการทดลองที่ถูกต้องแล้วก็ทำการทดสอบอีก โดยเมื่อนำเครื่องดื่มออกจากช่องใส่ แล้วพยายามนำกลับเข้าไปใหม่ก็จะไม่มีการเปลี่ยนแปลงของจำนวน ซึ่งก็จะเห็นว่าเมื่อนำเครื่องดื่มออกมาแล้วนำกลับเข้าไปใส่ไว้ดังเดิมก็จะไม่มีผลอะไร จะมีผลเฉพาะกรณีที่เป็นการคลิ๊กที่ Check In เท่านั้นโดยการคลิ๊กที่นี่เป็นการส่งคำสั่งในการเคลียร์ข้อมูลเดิมที่มีอยู่ด้วย

AUTOMATIC SOFT-DRINK CHECKING AND BILLING SYSTEM

Room : 4 Name : ณัฐวุฒิ ทนุผล

Check in Date : 22/3/2544 Time : 17:40:22

Check out Date : 22/3/2544 Time : 17:53:03

Code	Item Name	Price/Item	Amount	Total Price
001	CALPICO	25	1	25
002	PEPSI	25	2	50
003	ICE TEA	25	2	50
004	KLOSTER	50	2	100
005	SINGHA	40	2	80
006	HEINEKEN	50	2	100

Sum Total. 405 Baht.

Signature : .....

( )

22/3/2544 17:53:03

รูปที่ 4.7 แสดงผลการทดลองการนับจำนวนและคิดเงิน

การทดลองอีกอย่างหนึ่งคือการทดลองในกรณีที่สายที่ต่อไว้หลุด หรือคอมพิวเตอร์ไม่สามารถติดต่อกับตัวแม่หรือตัวลูกได้เลย ซึ่งเริ่มต้นจากการถอดสายที่ต่อทางพอร์ทอนุกรมของคอมพิวเตอร์กับตัวแม่ออก แล้วคลิ๊กที่คำสั่งใด ๆ ก็จะฟ้องขึ้นมาว่าไม่สามารถติดต่อกับตัวควบคุมหรือตัวแม่ได้ และต่อไปคือลองต่อสายพอร์ทอนุกรมกลับเข้าไปดังเดิมแล้วถอดสายที่ต่อระหว่างตัวแม่กับตัวลูกออก แล้วคลิ๊กเหมือนเดิม ก็จะฟ้องขึ้นมาว่าไม่สามารถติดต่อไปที่ตัวลูกที่ห้องหมายเลขนั้น ๆ ได้ ซึ่งหมายถึงว่าโครงงานนี้ได้มีการตรวจสอบการเชื่อมต่อระหว่างอุปกรณ์แต่ละตัว พร้อมกันนั้นก็ได้มีการตรวจสอบความผิดพลาดของข้อมูลโดยระหว่างตัวแม่กับตัวลูกนั้นเป็นการเชื่อมต่อที่มีระยะไกลพอสมควร จึงมีการตรวจสอบความผิดพลาดของข้อมูลโดยการเพิ่มข้อมูลเข้าไปอีกไบต์หนึ่ง โดยไบต์นี้เป็นผลรวมของข้อมูลที่ทำการส่งทั้งหมดทุกไบต์ แล้วนำไปทำเป็น 2's Complement เมื่อทางตัวแม่ได้รับข้อมูลออกจากตัวลูกแล้วก็ทำการหาผลรวมของข้อมูลที่ส่งมารวมทั้งไบต์ที่เพิ่มเข้าไปด้วย ถ้าข้อมูลนั้นไม่มีข้อผิดพลาดก็จะได้ผลลัพธ์เป็น "0" แต่ถ้ามีความผิดพลาดก็จะได้ผลลัพธ์ที่ไม่เป็น "0" จากการทดลองความผิดพลาดของข้อมูลนี้โดยการทำให้เกิดความผิดพลาดระหว่างตัวลูกกับตัวแม่ เมื่อมีความผิดพลาดเกิดขึ้นก็จะฟ้องขึ้นมาทางหน้าจอคอมพิวเตอร์เช่นกัน โดยแจ้งว่า ข้อมูลที่ถูกส่งกลับมานั้นเกิดความผิดพลาดขึ้น

บทที่ 5
สรุปและวิจารณ์

ในการทดลองโครงงานนี้ ในส่วนของวงจรหรือ Hardware นั้นไม่มีปัญหาแต่อย่างใดเพราะการออกแบบวงจรทางไมโครคอนโทรเลอร์ส่วนใหญ่จะใช้โปรแกรมควบคุมการทำงานทั้งหมด เนื่องจากส่วนประกอบทาง Hardware ภายนอกนั้นมีน้อย

สำหรับหลักการทำงานของเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัตินี้ส่วนใหญ่ จะเป็นการสื่อสารข้อมูลอนุกรมแบบอะซิงโครนัส ซึ่งการเชื่อมต่อระหว่างคอมพิวเตอร์กับตัวแม่จะใช้มาตรฐาน RS-232 ส่วนในการเชื่อมต่อระหว่างตัวแม่กับตัวลูกนั้นใช้มาตรฐาน RS-485 เนื่องจากสามารถติดต่อสื่อสารได้เป็นระยะทางไกลประมาณ 1.2 กิโลเมตร

จากการทดลองส่งข้อมูลอะซิงโครนัสในส่วนของมาตรฐาน RS-232 นั้นได้ผลคือ ที่ลอจิก "1" จะได้รับดับแรงดันไฟฟ้าประมาณ -8.6 โวลท์ ส่วนที่ลอจิก "0" นั้นได้ระดับแรงดันไฟฟ้าประมาณ +8.6 โวลท์ และในส่วนของมาตรฐาน RS-485 ที่ลอจิก "1" ผลต่างของแรงดันระหว่างขั้ว A กับ B จะมีค่าเป็นบวก หรือขั้ว A มีแรงดันมากกว่าขั้ว B ส่วนที่ลอจิก "0" ก็จะกลับกันกับที่ลอจิก "1" ซึ่งก็จะเป็นไปตามมาตรฐานทั้งสองแบบ

ส่วนการทำงานของโปรแกรมที่เขียนขึ้น กับฮาร์ดแวร์ที่ออกแบบและได้ประกอบจนเป็นเครื่องที่พร้อมที่จะใช้งานนั้นสามารถทำงานได้อย่างถูกต้อง จะมีปัญหาก็ในส่วนของช่องใส่เครื่องดื่มซึ่งใช้แผ่นอะลูมิเนียมตัดเป็นชิ้นๆ แล้วนำมาต่อกัน ซึ่งก็ทำให้การประกอบเข้าด้วยกันไม่ค่อยแนบสนิทเท่าที่ควร และการติดตั้งไมโครสวิทช์ในการตรวจสอบช่องใส่แต่ละช่องนั้น ติดตั้งได้ยากลำบากมาก เพราะเนื้อที่ติดตั้งนั้นแคบ และในตอนแรกนั้นมีสวิทช์บางตัวเมื่อใส่เครื่องดื่มเข้าไปแล้วตัวสวิทช์ไม่ถูกกด ทำให้เกิดความผิดพลาดขึ้นในการตรวจสอบ แต่ในกรณีนี้จะไม่แสดงความผิดพลาดทางหน้าจอคอมพิวเตอร์ เนื่องจากไม่ถือเป็นความผิดพลาดในการส่งผ่านข้อมูล แต่จะเป็นความผิดพลาดในการอ่านค่าจากสวิทช์ของตัวลูกเท่านั้น สำหรับในโครงงานนี้ที่ได้ใช้ไมโครสวิทช์เป็นตัวตรวจสอบช่องใส่เมื่อนำไปใส่ในตู้เย็นซึ่งมีความชื้นมาก ก็อาจจะทำให้สวิทช์เกิดการเสื่อมสภาพได้เร็วและความชื้นอาจทำให้เกิดความผิดพลาดขึ้นได้

ในการพัฒนาของเครื่องเช็คยอดและคิดราคาเครื่องดื่มอัตโนมัตินี้ จะต้องพัฒนาในเรื่องของช่องใส่เครื่องดื่มเป็นสำคัญ เนื่องจากเป็นส่วนที่ทำการตรวจสอบจำนวนของเครื่องดื่มแต่ละชนิด ซึ่งติดตั้งอยู่ในตู้เย็นหรือตู้แช่ ที่มีความเย็นและความชื้น ตัวที่จะนำมาตรวจสอบช่องใส่แต่ละช่องนั้นก็จะต้องมีความทนทานต่ออุณหภูมิและความชื้นสูงพอสมควร ซึ่งอาจจะใช้แสงอินฟาเรดก็ได้ สำหรับในโครงงานนี้ไม่ได้ใช้เพราะต้องใช้งบประมาณที่สูง เนื่องจากจะต้องตรวจสอบเครื่องดื่มเป็นจำนวน 24 ช่องต่อตัวลูก 1 ตัว ดังนั้นในโครงงานนี้จึงใช้ไมโครสวิทช์แทน

ภาคผนวก

```
;*****************************************
;
;  AUTOMATIC SOFT-DRINK BILLING SYSTEM
;
;  MASTER MICROCONTROLLER SOFTWARE
;
;*****************************************

;*****************************************
;          CODE OF TRANSMISSION
;*****************************************

ACK             EQU     06H
CONNECT         EQU     20H
SL_CON          EQU     128
SCAN            EQU     130
CLEAR           EQU     132
DATAERR         EQU     134
DATANOERR       EQU     136
SL_NOCON        EQU     138

TXCON           BIT     P3.4
RS232           BIT     P1.0
RS485           BIT     P1.1
RXF             BIT     00H

;********* INTERNAL RAM BUFFER *********

                ORG     0000H
                DS      22H
ROOM:           DS      1
COMMAND:        DS      1
DATABUF:        DS      16
DATAEMP:        DS      6
FCS:            DS      2
STACKPT:        DS      56

;*********** PROGRAM ADDRESS ***********

                ORG     0000H
                LJMP    INIT

                ORG     0023H
                LJMP    TxRxISR

;*****************************************
;              MAIN PROGRAM
;*****************************************

                ORG     0030H
INIT:           MOV     SCON,#0D0H
                MOV     TMOD,#21H
                MOV     PCON,#00H
                MOV     TH1,#0FDH
                MOV     SP,#STACKPT
                MOV     IE,#80H
                SETB    TR1

;*****************************************
;              MAIN PROGRAM
```

```
;*****************************************

MAIN:           ACALL   RXD1
                CJNE    A,#CONNECT,MAIN
                MOV     A,#ACK
                ACALL   TXD1
                ACALL   RXD1
                ACALL   TXD2_ADDR
                ACALL   RXD2
                JB      RXF,CHECK
                MOV     A,#SL_NOCON
                ACALL   TXD1
                SJMP    MAIN

;************** CHECK  ************************
CHECK:          CJNE    A,#ACK,MAIN
                MOV     A,#SL_CON
                ACALL   TXD1
                ACALL   RXD1
                ACALL   TXD2_ADDR
                ACALL   RX2_RECEIVE
                CJNE    A,#DATAERR,SENDDATA
                LCALL   TXD1
                SJMP    MAIN
SENDDATA:       ACALL   TXD1
                ACALL   TXD1_SEND
                SJMP    MAIN

;*****************************************
RX2_RECEIVE:    ACALL   RXD2
                MOV     DATABUF,A
                ACALL   RXD2
                MOV     DATABUF+1,A
                ACALL   RXD2
                MOV     DATABUF+2,A
                ACALL   RXD2
                MOV     DATABUF+3,A
                ACALL   RXD2
                MOV     DATABUF+4,A
                ACALL   RXD2
                MOV     DATABUF+5,A
                ACALL   RXD2
                MOV     FCS,A
                CLR     C
                MOV     A,DATABUF
                ADD     A,DATABUF+1
                ADDC    A,DATABUF+2
                ADDC    A,DATABUF+3
                ADDC    A,DATABUF+4
                ADDC    A,DATABUF+5
                ADDC    A,FCS
                CJNE    A,#00H,DATAERROR
                MOV     A,#DATANOERR
                RET
DATAERROR:      MOV     A,#DATAERR
                RET

TXD1_SEND:      MOV     A,DATABUF
                ADD     A,#30H
                ACALL   TXD1
                MOV     A,DATABUF+1
```

```
                ADD     A,#30H
                ACALL   TXD1
                MOV     A,DATABUF+2
                ADD     A,#30H
                ACALL   TXD1
                MOV     A,DATABUF+3
                ADD     A,#30H
                ACALL   TXD1
                MOV     A,DATABUF+4
                ADD     A,#30H
                ACALL   TXD1
                MOV     A,DATABUF+5
                ADD     A,#30H
                ACALL   TXD1
                RET

;*************************************
TXD1:           SETB    RS485
                CLR     RS232
                SETB    TB8
                MOV     SBUF,A
                JNB     TI,$
                CLR     TI
                RET

;*************************************
TXD2_DATA:      CLR     RS485
                SETB    RS232
                SETB    TXCON
                CLR     TB8
                MOV     SBUF,A
                JNB     TI,$
                CLR     TI
                CLR     TXCON
                RET

;*************************************
TXD2_ADDR:      CLR     RS485
                SETB    RS232
                SETB    TXCON
                SETB    TB8
                MOV     SBUF,A
                JNB     TI,$
                CLR     TI
                CLR     TXCON
                RET

;*************************************
RXD1:           SETB    RS485
                CLR     RS232
                JNB     RI,$
                CLR     RI
                MOV     A,SBUF
                RET

;*************************************
RXD2:           CLR     RS485
                SETB    RS232
                CLR     TXCON
                SETB    ES
                CLR     RXF
```

```
                MOV     TH0,#00H
                MOV     TL0,#00H
                SETB    TR0
                JNB     TF0,$
                CLR     ES
                CLR     TF0
                CLR     TR0
                RET
;****************************************
TxRxISR:        CLR     TR0
                SETB    RXF
                CLR     ES
                SETB    TF0
                CLR     RI
                MOV     A,SBUF
                RETI

                END
```

```
;*****************************************
;
;  AUTOMATIC SOFT-DRINK BILLING SYSTEM
;
;  SLAVE MICROCONTROLLER SOFTWARE
;
;*****************************************

;*****************************************
;         SLOT PORT ADDRESS
;*****************************************

PORTSLOT1     EQU     8000H
PORTSLOT2     EQU     8001H
PORTSLOT3     EQU     8002H

;*****************************************
;         CODE OF TRANSMISSION
;*****************************************

ACK           EQU     06H
SCAN          EQU     130
CLEAR         EQU     132
RXF           BIT     00H
TXCON         BIT     P3.4

              ORG     0000H
              DS      22H
SLOT1:        DS      1
SLOT2:        DS      1
SLOT3:        DS      1
EMPTY:        DS      1
FCS:          DS      2
ITEM:         DS      6
ADDRESS:      DS      1
CODEBUFF:     DS      1
TEXT:         DS      1
BUFFER:       DS      1
STACKPT:      DS      40

              ORG     0000H
              LJMP    INIT

              ORG     0023H
              LJMP    TxRxISR

;*****************************************
;         INITIAL SERIAL PORT
;*****************************************

INIT:         MOV     SCON,#0F0H
              MOV     TMOD,#21H
              ANL     PCON,#7FH
              MOV     TH1,#0FDH
              MOV     P1,#0FFH
              MOV     SP,#STACKPT
              MOV     IE,#80H
              SETB    TR1
```

```
                CLR     TB8
                MOV     ADDRESS,P1

;****************************************
;               MAIN PROGRAM
;****************************************

MAIN:           SETB    SM2
                LCALL   RXD2
                CJNE    A,ADDRESS,MAIN
                CLR     SM2
                MOV     A,#ACK
                LCALL   TXD2
                LCALL   RXD2
                JB      RXF,MAIN1
                SJMP    MAIN
MAIN1:          CJNE    A,#CLEAR,SCAN1
                CLR     A
                MOV     SLOT1,A
                MOV     SLOT2,A
                MOV     SLOT3,A
                LCALL   SCANSLOT
                LCALL   CHKDATA
                LCALL   SENDDATA
                SJMP    MAIN

SCAN1:          CJNE    A,#SCAN,MAIN
                LCALL   SCANSLOT
                LCALL   CHKDATA
                LCALL   SENDDATA
                SJMP    MAIN

;****************************************
;           SEND DATA SUBROUTINE
;****************************************

SENDDATA:       MOV     A,ITEM
                CLR     C
                ADD     A,ITEM+1
                ADDC    A,ITEM+2
                ADDC    A,ITEM+3
                ADDC    A,ITEM+4
                ADDC    A,ITEM+5
                CPL     A
                ADD     A,#01H
                MOV     FCS,A
                MOV     A,ITEM
                LCALL   TXD2
                MOV     A,ITEM+1
                LCALL   TXD2
                MOV     A,ITEM+2
                LCALL   TXD2
                MOV     A,ITEM+3
                LCALL   TXD2
                MOV     A,ITEM+4
                LCALL   TXD2
                MOV     A,ITEM+5
                LCALL   TXD2
                MOV     A,FCS
                LCALL   TXD2
                RET
```

```
;*****************************************
;            SCAN SLOT SUBROUTINE
;*****************************************

SCANSLOT:     MOV     DPTR,#PORTSLOT1
              MOVX    A,@DPTR
              ORL     A,SLOT1
              MOV     SLOT1,A
              MOV     DPTR,#PORTSLOT2
              MOVX    A,@DPTR
              ORL     A,SLOT2
              MOV     SLOT2,A
              MOV     DPTR,#PORTSLOT3
              MOVX    A,@DPTR
              ORL     A,SLOT3
              MOV     SLOT3,A
              RET

;*****************************************
;          CHECK DATA SUBROUTINE
;*****************************************

CHKDATA:      MOV     A,SLOT1
              PUSH    ACC
              ANL     A,#0FH
              LCALL   CHKBIT
              MOV     ITEM,A
              POP     ACC
              ANL     A,#0F0H
              SWAP    A
              LCALL   CHKBIT
              MOV     ITEM+1,A
              MOV     A,SLOT2
              PUSH    ACC
              ANL     A,#0FH
              LCALL   CHKBIT
              MOV     ITEM+2,A
              POP     ACC
              ANL     A,#0F0H
              SWAP    A
              LCALL   CHKBIT
              MOV     ITEM+3,A
              MOV     A,SLOT3
              PUSH    ACC
              ANL     A,#0FH
              LCALL   CHKBIT
              MOV     ITEM+4,A
              POP     ACC
              ANL     A,#0F0H
              SWAP    A
              LCALL   CHKBIT
              MOV     ITEM+5,A
              RET

;*****************************************
;            CHECK BIT SUBROUTINE
;*****************************************

CHKBIT:       MOV     R7,#04H
              MOV     R6,#00H
CHKBIT1:      JNB     ACC.0,CHKBIT2
```

```
                INC     R6
CHKBIT2:        DJNZ    R7,CHKBIT3
                MOV     A,R6
                RET
CHKBIT3:        RR      A
                SJMP    CHKBIT1

DELAY:          CLR     RS0
                SETB    RS1
                MOV     R3,#1
DELAY1:         MOV     R2,#1
                DJNZ    R2,$
                DJNZ    R3,DELAY1
                CLR     RS1
                RET

TXD2:           SETB TXCON
                MOV SBUF,A
                JNB TI,$
                CLR TI
                RET

;*************************************
RXD2:           CLR     TXCON
                SETB    ES
                CLR     RXF
                MOV     TH0,#00H
                MOV     TL0,#00H
                SETB    TR0
                JNB     TF0,$
                CLR     ES
                CLR     TF0
                CLR     TR0
                RET

;*************************************
TxRxISR:        CLR     TR0
                SETB    RXF
                CLR     ES
                SETB    TF0
                CLR     RI
                MOV     A,SBUF
                RETI

                END
```

```
'*************************************************************

' AUTOMATIC SOFT-DRINK BILLING SYSTEM

' COMPUTER SOFTWARE FORM MAIN

'*************************************************************


Dim Datainput(10) As String
Public Buffer As String
Public ReceiveDelay As Double
Public Ack As String
Public j As Integer
Public i As Integer
Public Date_In As String
Public Time_In As String
Public Date_Out As String
Public Time_Out As String
Public Sumtotal As Double
Public RoomIn As String
Public NameIn As String
Public Comset As String
Public Connect As Boolean
Public CommPort As Boolean
Public Slave_Con As String
Dim Slave_Status As Boolean
Public Clear_Com As String
Public Scan_Com As String
Dim DataErr As String
Dim DataNoErr As String

Private Sub AddItemcmd_Click()
    AddRoom.Show
End Sub

Private Sub Form_Load()

' Set Command and Comm port

    Ack = Chr(6)                    'Ack = 06H
    Slave_Con = Chr(128)
    Scan_Com = Chr(130)
    Clear_Com = Chr(132)
    DataErr = Chr(134)
    DataNoErr = Chr(136)
    Comset = 1
    MSComm.Settings = "9600,M,8,1"
    MSComm.InputLen = 1
    MSComm.OutBufferSize = 1

End Sub

Private Sub CheckInCmd_Click()
    RoomIn = Val(RoomNumber.Text)
    If RoomIn < 1 Then
        MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
    ElseIf RoomIn < 257 Then
    If Not NameText.Text = "" Then
```

```
            RoomData.Recordset.MoveFirst
            Do Until RoomData.Recordset!room = RoomNumber.Text
                RoomData.Recordset.MoveNext
            Loop
            If RoomData.Recordset!roominstall = True Then
                If RoomData.Recordset!Status = False Then
                    Call DataClear
                    If CommPort = False Then Exit Sub
                    If Connect = True Then
                        If Slave_Status = True Then
                            If Datainput(0) = DataNoErr Then
                                With RoomData.Recordset
                                .Edit
                                !Name = NameText.Text
                                !Date_In = Date
                                !Time_In = Time
                                !Date_Out = " "
                                !Time_Out = " "
                                !water = Datainput(1)
                                !coke = Datainput(2)
                                !sp = Datainput(3)
                                !ice_tea = Datainput(4)
                                !singha = Datainput(5)
                                !hinegen = Datainput(6)
                                !Status = True
                                .Update
                                End With
                                MsgBox ("Check in Complete ! "), , "Check
in OK."
                            ElseIf Datainput(0) = DataErr Then
                                MsgBox ("Data Receive Error from this
room. => " & DataMonitor.Recordset!room), , "Data Error"
                            End If
                        Else
                            MsgBox ("Not connecting to Room => " &
DataMonitor.Recordset!room), vbCritical, "Slave in room Error"
                        End If
                    Else
                        MsgBox ("Not Connect to Control Board."),
vbCritical, "Ack Error"
                    End If
                Else
                    MsgBox ("The room is already in Used => " & RoomIn),
vbCritical, "Not live in this room"
                End If
            Else
                MsgBox ("This room is not install ! Please change the
room => " & RoomIn), vbCritical, "Room Error"
            End If
        Else
            MsgBox ("Please Enter Name "), vbCritical, "Name Error"
        End If
        Else
            MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
        End If

End Sub

Private Sub Chkoutcmd_Click()
    RoomIn = Val(RoomNumber.Text)
```

```
                .MoveFirst
                Sumtotal = 0
                For i = 1 To 6
                    .Edit
                    !total = !amount * !price
                    .Update
                    Sumtotal = Sumtotal + Val(!total)
                    .MoveNext
                Next i
                End With
                With Checkout
                .Time_Out = " "
                .Date_Out = " "
                .RoomNumber.Text = RoomIn
                .BillName.Text = NameIn
                .Date_In.Text = Date_In
                .Date_Out.Text = Date_Out
                .Time_In.Text = Time_In
                .Time_Out.Text = Time_Out
                .TotalText.Text = Sumtotal
                .DataOrder.Refresh
                .Show
                End With
                    Else
                    MsgBox ("Data Receive Error from this room. => "
& DataMonitor.Recordset!room), , "Data Error"
                    End If
                Else
                    MsgBox ("Not connecting to Room. => " &
DataMonitor.Recordset!room), vbCritical, "Slave in room Error"
                End If
            Else
                    MsgBox ("Not Connect to Control Board."),
vbCritical, "Ack Error"
            End If

        Else
            MsgBox ("This room was check out, Because No person live
in this room"), vbCritical, "Room Error"
        End If

    Else
        MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
    End If
End Sub

Private Sub mnuAbout_Click()
frmAbout.Show
End Sub

Private Sub MoniCmd_Click()
    j = 0
    ProgressBar1.Max = 256
    RoomData.Recordset.MoveFirst
    DataMonitor.Recordset.MoveFirst
    Do Until RoomData.Recordset.EOF
        If RoomData.Recordset!roominstall = True Then
        If RoomData.Recordset!Status = True Then
            RoomIn = RoomData.Recordset!room
```

```
    If RoomIn < 1 Then
        MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
    ElseIf RoomIn < 257 Then

        RoomData.Recordset.MoveFirst
        Do Until RoomData.Recordset!room = RoomIn
            RoomData.Recordset.MoveNext
        Loop

        If RoomData.Recordset!Status = True Then
            Call ReceiveData
            If CommPort = False Then Exit Sub
            If Connect = True Then
                If Slave_Status = True Then
                    If Datainput(0) = DataNoErr Then
                        With RoomData.Recordset
                        '.Edit
                        '!water = Val(!water) + Val(Datainput(1))
                        '!coke = Val(!coke) + Val(Datainput(2))
                        '!sp = Val(!sp) + Val(Datainput(3))
                        '!ice_tea = Val(!ice_tea) + Val(Datainput(4))
                        '!singha = Val(!singha) + Val(Datainput(5))
                        '!hinegen = Val(!hinegen) + Val(Datainput(6))
                        '.Update
                        Date_In = !Date_In
                        Time_In = !Time_In
                        NameIn = !Name
                        End With

                        With DataOrder.Recordset
                        .MoveFirst
                        .Edit
                        !amount = Val(RoomData.Recordset!water) +
Val(Datainput(1))
                        .Update
                        .MoveNext
                        .Edit
                        !amount = Val(RoomData.Recordset!coke) +
Val(Datainput(2))
                        .Update
                        .MoveNext
                        .Edit
                        !amount = Val(RoomData.Recordset!sp) +
Val(Datainput(3))
                        .Update
                        .MoveNext
                        .Edit
                        !amount = Val(RoomData.Recordset!ice_tea) +
Val(Datainput(4))
                        .Update
                        .MoveNext
                        .Edit
                        !amount = Val(RoomData.Recordset!singha) +
Val(Datainput(5))
                        .Update
                        .MoveNext
                        .Edit
                        !amount = Val(RoomData.Recordset!hinegen) +
Val(Datainput(6))
                        .Update
```

```
                Call ReceiveData
                If CommPort = False Then Exit Sub
                If Connect = True Then
                    If Slave_Status = True Then
                        If Datainput(0) = DataNoErr Then
                            With DataMonitor.Recordset
                            .Edit
                            !water = Datainput(1)
                            !coke = Datainput(2)
                            !sp = Datainput(3)
                            !ice_tea = Datainput(4)
                            !singha = Datainput(5)
                            !hinegen = Datainput(6)
                            .Update
                            End With
                        Else
                            With DataMonitor.Recordset
                            .Edit
                            !water = " "
                            !coke = " "
                            !sp = " "
                            !ice_tea = " "
                            !singha = " "
                            !hinegen = " "
                            .Update
                            End With
                            MsgBox ("Data Receive Error from this room.
 => " & DataMonitor.Recordset!room), , "Data Error"
                        End If
                    Else
                        With DataMonitor.Recordset
                        .Edit
                        !water = " "
                        !coke = " "
                        !sp = " "
                        !ice_tea = " "
                        !singha = " "
                        !hinegen = " "
                        .Update
                        End With
                        MsgBox ("Not connecting in Room => " &
DataMonitor.Recordset!room), vbCritical, "Slave in room Error"
                    End If
                Else
                        MsgBox ("Not Connect to Control Board."),
vbCritical, "Ack Error"
                End If
            Else
                With DataMonitor.Recordset
                .Edit
                !water = " "
                !coke = " "
                !sp = " "
                !ice_tea = " "
                !singha = " "
                !hinegen = " "
                .Update
                End With
            End If
            End If
            RoomData.Recordset.MoveNext
```

```
        DataMonitor.Recordset.MoveNext
        j = j + 1
        ProgressBar1.Value = j
    Loop
    Monitor.Text1.Text = Date
    Monitor.Text2.Text = Time
    Monitor.AddItemcmd.Enabled = True
    Monitor.Show
    Monitor.DataMonitor.Refresh
    RoomData.Refresh
    ProgressBar1.Value = 0
End Sub

Private Sub ExitCmd_Click()
    End
End Sub

Private Sub Mscommset_Click()
    Load SetMScomm
    SetMScomm.Show
End Sub

Public Sub ReceiveData()
    MSComm.CommPort = Comset
    i = 0
    ProgressBar1.Enabled = True
        On Error GoTo CommPortErr
        MSComm.PortOpen = True
        MSComm.Output = Chr(32)
        CommPort = True
        For ReceiveDelay = 1 To 100000
            If MSComm.InBufferCount Then Exit For
        Next ReceiveDelay
        Buffer = MSComm.Input
        If Buffer = Ack Then
            Connect = True
            MSComm.Output = Chr(RoomIn - 1)
            For ReceiveDelay = 1 To 100000
                If MSComm.InBufferCount Then Exit For
            Next ReceiveDelay
            Buffer = MSComm.Input
            If Buffer = Slave_Con Then
                MSComm.Output = Scan_Com
                Slave_Status = True
                Do
                    For ReceiveDelay = 1 To 10000
                        If MSComm.InBufferCount Then Exit For
                    Next ReceiveDelay
                    Datainput(i) = MSComm.Input
                    i = i + 1
                    ProgressBar1.Value = i * 10
                    If Datainput(0) = DataErr Then Exit Do
                Loop Until i = 7
            Else
                Slave_Status = False
            End If
        Else
            Connect = False
        End If
        MSComm.PortOpen = False
        ProgressBar1.Value = 0
```

```
        Exit Sub
CommPortErr:
        CommPort = False
        MsgBox ("Unable to Open Communication Port! Please Check
Communication Port Com : " & Comset), vbCritical, "Communication port
Error"
ProgressBar1.Value = 0

End Sub

Public Sub DataClear()
    MSComm.CommPort = Comset
    i = 0
    ProgressBar1.Enabled = True
        On Error GoTo CommPortErr
        MSComm.PortOpen = True
        MSComm.Output = Chr(32)
        CommPort = True
        For ReceiveDelay = 1 To 100000
            If MSComm.InBufferCount Then Exit For
        Next ReceiveDelay
        Buffer = MSComm.Input
        If Buffer = Ack Then
            Connect = True
            MSComm.Output = Chr(RoomIn - 1)
            For ReceiveDelay = 1 To 100000
                If MSComm.InBufferCount Then Exit For
            Next ReceiveDelay
            Buffer = MSComm.Input
            If Buffer = Slave_Con Then
                MSComm.Output = Clear_Com
                Slave_Status = True
                Do
                    For ReceiveDelay = 1 To 10000
                        If MSComm.InBufferCount Then Exit For
                    Next ReceiveDelay
                    Datainput(i) = MSComm.Input
                    i = i + 1
                    ProgressBar1.Value = i * 10
                    If Datainput(0) = DataErr Then Exit Do
                Loop Until i = 7
            Else
                Slave_Status = False
            End If
        Else
            Connect = False
        End If
        MSComm.PortOpen = False
        ProgressBar1.Value = 0
        Exit Sub
CommPortErr:
        CommPort = False
        MsgBox ("Unable to Open Communication Port! Please Check
Communication Port Com : " & Comset), vbCritical, "Communication port
Error"
ProgressBar1.Value = 0

End Sub

Private Sub ClearData_Click()
    RoomIn = Val(RoomNumber.Text)
```

```
    If RoomIn < 1 Then
        MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
    ElseIf RoomIn < 257 Then
        Call DataClear
        MsgBox ("Clear data in this Room Complete...")
    Else
        MsgBox ("Please Enter number of Room (1-256)"), vbCritical,
"Room Error"
    End If

End Sub
```

```
'*************************************************************
' AUTOMATIC SOFT-DRINK BILLING SYSTEM
' COMPUTER SOFTWARE ALL OTHER FORM
'*************************************************************

'*************************************************************
' FORM CHECKOUT
'*************************************************************

Dim ReceiveDelay As Long


Private Sub CloseCmd_Click()
    Main.RoomData.Refresh
    Checkout.Refresh
    Checkout.Hide
End Sub

Private Sub OK_Click()
MsgBox ("Do you want to Check out ?"), vbOKCancel, "Check out"

    RoomData.Recordset.MoveFirst
    Do Until RoomData.Recordset!room = Main.RoomIn
        RoomData.Recordset.MoveNext
    Loop

    With RoomData.Recordset
        .Edit
        !Date_Out = Date
        !Time_Out = Time
        !water = " "
        !coke = " "
        !sp = " "
        !ice_tea = " "
        !singha = " "
        !hinegen = " "
        !Status = False
        .Update
        Date_Out.Text = !Date_Out
        Time_Out.Text = !Time_Out
  End With

  AddData.Recordset.MoveFirst
  Do Until AddData.Recordset!room = Main.RoomIn
      AddData.Recordset.MoveNext
  Loop

  With AddData.Recordset
      .Edit
      !water = 0
      !coke = 0
      !sp = 0
      !ice_tea = 0
```

```
            !singha = 0
            !hinegen = 0
            .Update
        End With

        DataMonitor.Recordset.MoveFirst
        Do Until DataMonitor.Recordset!room = Main.RoomIn
            DataMonitor.Recordset.MoveNext
        Loop
        With DataMonitor.Recordset
            .Edit
            !water = " "
            !coke = " "
            !sp = " "
            !ice_tea = " "
            !singha = " "
            !hinegen = " "
            .Update
        End With
    EndOut:
    End Sub



    '*********************************************************

    ' FORM MONITOR

    '*********************************************************

    Dim Printfrm As Printer

    Private Sub AddItemcmd_Click()
        RoomData.Recordset.MoveFirst
        DataMonitor.Recordset.MoveFirst
        For i = 1 To 255
            If RoomData.Recordset!roominstall = True Then
                With RoomData.Recordset
                .Edit
                !water = Val(!water) + Val(DataMonitor.Recordset!water)
                !coke = Val(!coke) + Val(DataMonitor.Recordset!coke)
                !sp = Val(!sp) + Val(DataMonitor.Recordset!sp)
                !ice_tea = Val(!ice_tea) +
Val(DataMonitor.Recordset!ice_tea)
                !singha = Val(!singha) +
Val(DataMonitor.Recordset!singha)
                !hinegen = Val(!hinegen) +
Val(DataMonitor.Recordset!hinegen)
                .Update
                End With
                RoomData.Recordset.MoveNext
                DataMonitor.Recordset.MoveNext
            Else
                RoomData.Recordset.MoveNext
                DataMonitor.Recordset.MoveNext
            End If
        Next i
        Main.RoomData.Refresh
        AddItemcmd.Enabled = False
        DataMonitor.Recordset.MoveFirst
```

```
End Sub

Private Sub OK_Click()
Monitor.Hide
DataMonitor.Refresh
End Sub



'*************************************************************

' FORM ABOUT

'*************************************************************

Option Explicit

' Reg Key Security Options...
Const READ_CONTROL = &H20000
Const KEY_QUERY_VALUE = &H1
Const KEY_SET_VALUE = &H2
Const KEY_CREATE_SUB_KEY = &H4
Const KEY_ENUMERATE_SUB_KEYS = &H8
Const KEY_NOTIFY = &H10
Const KEY_CREATE_LINK = &H20
Const KEY_ALL_ACCESS = KEY_QUERY_VALUE + KEY_SET_VALUE +_
                       KEY_CREATE_SUB_KEY + KEY_ENUMERATE_SUB_KEYS +_
                       KEY_NOTIFY + KEY_CREATE_LINK + READ_CONTROL

' Reg Key ROOT Types...
Const HKEY_LOCAL_MACHINE = &H80000002
Const ERROR_SUCCESS = 0
Const REG_SZ = 1                        ' Unicode nul terminated string
Const REG_DWORD = 4                     ' 32-bit number

Const gREGKEYSYSINFOLOC = "SOFTWARE\Microsoft\Shared Tools Location"
Const gREGVALSYSINFOLOC = "MSINFO"
Const gREGKEYSYSINFO = "SOFTWARE\Microsoft\Shared Tools\MSINFO"
Const gREGVALSYSINFO = "PATH"

Private Declare Function RegOpenKeyEx Lib "advapi32" Alias
"RegOpenKeyExA" (ByVal hKey As Long, ByVal lpSubKey As String, ByVal
ulOptions As Long, ByVal samDesired As Long, ByRef phkResult As Long)
As Long
Private Declare Function RegQueryValueEx Lib "advapi32" Alias
"RegQueryValueExA" (ByVal hKey As Long, ByVal lpValueName As String,
ByVal lpReserved As Long, ByRef lpType As Long, ByVal lpData As
String, ByRef lpcbData As Long) As Long
Private Declare Function RegCloseKey Lib "advapi32" (ByVal hKey As
Long) As Long


Private Sub cmdSysInfo_Click()
  Call StartSysInfo
End Sub

Private Sub cmdOK_Click()
  Unload Me
End Sub
```

```
Private Sub Form_Load()
    Me.Caption = "About " & App.Title
    lblVersion.Caption = "Version " & App.Major & "." & App.Minor &
"." & App.Revision
    lblTitle.Caption = App.Title
End Sub

Public Sub StartSysInfo()
    On Error GoTo SysInfoErr

    Dim rc As Long
    Dim SysInfoPath As String

    ' Try To Get System Info Program Path\Name From Registry...
    If GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFO,
gREGVALSYSINFO, SysInfoPath) Then
    ' Try To Get System Info Program Path Only From Registry...
    ElseIf GetKeyValue(HKEY_LOCAL_MACHINE, gREGKEYSYSINFOLOC,
gREGVALSYSINFOLOC, SysInfoPath) Then
        ' Validate Existance Of Known 32 Bit File Version
        If (Dir(SysInfoPath & "\MSINFO32.EXE") <> "") Then
            SysInfoPath = SysInfoPath & "\MSINFO32.EXE"

        ' Error - File Can Not Be Found...
        Else
            GoTo SysInfoErr
        End If
    ' Error - Registry Entry Can Not Be Found...
    Else
        GoTo SysInfoErr
    End If

    Call Shell(SysInfoPath, vbNormalFocus)

    Exit Sub
SysInfoErr:
    MsgBox "System Information Is Unavailable At This Time", vbOKOnly
End Sub

Public Function GetKeyValue(KeyRoot As Long, KeyName As String,
SubKeyRef As String, ByRef KeyVal As String) As Boolean
    Dim i As Long
Counter                                                 ' Loop
    Dim rc As Long
Code                                                    ' Return
    Dim hKey As Long
To An Open Registry Key                                 ' Handle
    Dim hDepth As Long
    Dim KeyValType As Long                              '
Type Of A Registry Key                                  ' Data
    Dim tmpVal As String
Storage For A Registry Key Value                        ' Tempory
    Dim KeyValSize As Long
Registry Key Variable                                   ' Size Of
    '------------------------------------------------------------
    ' Open RegKey Under KeyRoot {HKEY_LOCAL_MACHINE...}
    '------------------------------------------------------------
    rc = RegOpenKeyEx(KeyRoot, KeyName, 0, KEY_ALL_ACCESS, hKey) '
Open Registry Key
```

```
    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle
Error...

    tmpVal = String$(1024, 0)                               ' Allocate
Variable Space
    KeyValSize = 1024                                       ' Mark
Variable Size

    '------------------------------------------------------------
    ' Retrieve Registry Key Value...
    '------------------------------------------------------------
    rc = RegQueryValueEx(hKey, SubKeyRef, 0, _
                         KeyValType, tmpVal, KeyValSize)    '
Get/Create Key Value

    If (rc <> ERROR_SUCCESS) Then GoTo GetKeyError          ' Handle
Errors

    If (Asc(Mid(tmpVal, KeyValSize, 1)) = 0) Then           ' Win95
Adds Null Terminated String...
        tmpVal = Left(tmpVal, KeyValSize - 1)               ' Null
Found, Extract From String
    Else                                                    ' WinNT
Does NOT Null Terminate String...
        tmpVal = Left(tmpVal, KeyValSize)                   ' Null
Not Found, Extract String Only
    End If
    '------------------------------------------------------------
    ' Determine Key Value Type For Conversion...
    '------------------------------------------------------------
    Select Case KeyValType                                  ' Search
Data Types...
    Case REG_SZ                                             ' String
Registry Key Data Type
        KeyVal = tmpVal                                     ' Copy
String Value
    Case REG_DWORD                                          ' Double
Word Registry Key Data Type
        For i = Len(tmpVal) To 1 Step -1                    ' Convert
Each Bit
            KeyVal = KeyVal + Hex(Asc(Mid(tmpVal, i, 1)))   ' Build
Value Char. By Char.
        Next
        KeyVal = Format$("&h" + KeyVal)                     ' Convert
Double Word To String
    End Select

    GetKeyValue = True                                      ' Return
Success
    rc = RegCloseKey(hKey)                                  ' Close
Registry Key
    Exit Function                                           ' Exit

GetKeyError:      ' Cleanup After An Error Has Occured...
    KeyVal = ""                                             ' Set
Return Val To Empty String
    GetKeyValue = False                                     ' Return
Failure
    rc = RegCloseKey(hKey)                                  ' Close
Registry Key
End Function
```

National Semiconductor

July 1998

DS75176B/DS75176BT
Multipoint RS-485/RS-422 Transceivers

General Description

The DS75176B is a high speed differential TRI-STATE® bus/line transceiver designed to meet the requirements of EIA standard RS485 with extended common mode range (+12V to −7V), for multipoint data transmission. In addition, it is compatible with RS-422.

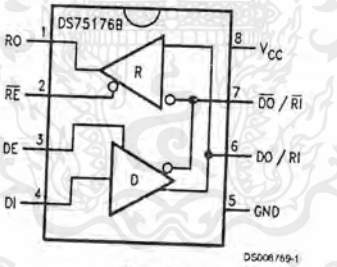
The driver and receiver outputs feature TRI-STATE capability, for the driver outputs over the entire common mode range of +12V to −7V. Bus contention or fault situations that cause excessive power dissipation within the device are handled by a thermal shutdown circuit, which forces the driver outputs into the high impedance state.

DC specifications are guaranteed over the 0 to 70°C temperature and 4.75V to 5.25V supply voltage range.

Features

- Meets EIA standard RS485 for multipoint bus transmission and is compatible with RS-422.
- Small Outline (SO) Package option available for minimum board space.
- 22 ns driver propagation delays.
- Single +5V supply.
- −7V to +12V bus common mode range permits ±7V ground difference between devices on the bus.
- Thermal shutdown protection.
- High impedance to bus with driver in TRI-STATE or with power off, over the entire common mode range allows the unused devices on the bus to be powered down.
- Pin out compatible with DS3695/A and SN75176A/B.
- Combined impedance of a driver output and receiver input is less than one RS485 unit load, allowing up to 32 transceivers on the bus.
- 70 mV typical receiver hysteresis.

Connection and Logic Diagram



DS008759-1

Top View
Order Number DS75176BN, DS75176BTN, DS75176BM or DS75176BTM
See NS Package Number N08E or M08A

TRI-STATE® is a registered trademark of National Semiconductor Corp.

© 1999 National Semiconductor Corporation

Absolute Maximum Ratings (Note 1)

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

Supply Voltage, V_{CC}	7V
Control Input Voltages	7V
Driver Input Voltage	7V
Driver Output Voltages	+15V/ −10V
Receiver Input Voltages (DS75176B)	+15V/ −10V
Receiver Output Voltage	5.5V
Continuous Power Dissipation @ 25°C	
for M Package	675 mW (Note 5)
for N Package	900 mW (Note 4)
Storage Temperature Range	−65°C to +150°C
Lead Temperature (Soldering, 4 seconds)	260°C

Recommended Operating Conditions

	Min	Max	Units
Supply Voltage, V_{CC}	4.75	5.25	V
Voltage at Any Bus Terminal (Separate or Common Mode)	−7	+12	V
Operating Free Air Temperature T_A			
DS75176B	0	+70	°C
DS75176BT	−40	+85	°C
Differential Input Voltage, VID (Note 6)	−12	+12	V

Electrical Characteristics (Notes 2, 3)

0°C ≤ T_A ≤ 70°C, 4.75V ≤ V_{CC} ≤ 5.25V unless otherwise specified

Symbol	Parameter		Conditions		Min	Typ	Max	Units
V_{OD1}	Differential Driver Output Voltage (Unloaded)		I_O = 0				5	V
V_{OD2}	Differential Driver Output Voltage (with Load)		(*Figure 1*)	R = 50Ω; (RS-422) (Note 7)	2			V
				R = 27Ω; (RS-485)	1.5			V
ΔV_{OD}	Change in Magnitude of Driver Differential Output Voltage For Complementary Output States		(*Figure 1*)	R = 27Ω			0.2	V
V_{OC}	Driver Common Mode Output Voltage						3.0	V
$\Delta\|V_{OC}\|$	Change in Magnitude of Driver Common Mode Output Voltage For Complementary Output States						0.2	V
V_{IH}	Input High Voltage	DI, DE, $\overline{RE}$, E			2			V
V_{IL}	Input Low Voltage						0.8	
V_{CL}	Input Clamp Voltage			I_{IN} = −18 mA			−1.5	
I_{IL}	Input Low Current			V_{IL} = 0.4V			−200	μA
I_{IH}	Input High Current			V_{IH} = 2.4V			20	μA
I_{IN}	Input Current	DO/RI, $\overline{DO}/\overline{RI}$	V_{CC} = 0V or 5.25V, DE = 0V	V_{IN} = 12V			+1.0	mA
				V_{IN} = −7V			−0.8	mA
V_{TH}	Differential Input Threshold Voltage for Receiver		−7V ≤ V_{CM} ≤ +12V		−0.2		+0.2	V
ΔV_{TH}	Receiver Input Hysteresis		V_{CM} = 0V			70		mV
V_{OH}	Receiver Output High Voltage	RO	I_{OH} = −400 μA		2.7			V
V_{OL}	Output Low Voltage		I_{OL} = 16 mA (Note 7)				0.5	V
I_{OZR}	OFF-State (High Impedance) Output Current at Receiver		V_{CC} = Max, 0.4V ≤ V_O ≤ 2.4V				±20	μA
R_{IN}	Receiver Input Resistance		−7V ≤ V_{CM} ≤ +12V		12			kΩ
I_{CC}	Supply Current		No Load (Note 7)	Driver Outputs Enabled			55	mA
				Driver Outputs Disabled			35	mA
I_{OSD}	Driver Short-Circuit Output Current		V_O = −7V (Note 7)				−250	mA
			V_O = +12V (Note 7)				+250	mA

Electrical Characteristics (Notes 2, 3) (Continued)

$0°C \le T_A \le 70°C$, $4.75V < V_{CC} < 5.25V$ unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Units
I_{OSR}	Receiver Short-Circuit Output Current	V_O = 0V	−15		−85	mA

Note 1: "Absolute Maximum Ratings" are those beyond which the safety of the device cannot be guaranteed. They are not meant to imply that the device should be operated at these limits. The tables of "Electrical Characteristics" provide conditions for actual device operation.

Note 2: All currents into device pins are positive; all currents out of device pins are negative. All voltages are referenced to device ground unless otherwise specified.

Note 3: All typicals are given for V_{CC} = 5V and T_A = 25°C.

Note 4: Derate linearly at 5.56 mW/°C to 650 mW at 70°C.

Note 5: Derate linearly 6.11 mW/°C to 400 mW at 70°C.

Note 6: Differential - Input/Output bus voltage is measured at the noninverting terminal A with respect to the inverting terminal B.

Note 7: All worst case parameters for which note 7 is applied, must be increased by 10% for DS75176BT. The other parameters remain valid for $-40°C < T_A < +85°C$.

Switching Characteristics

V_{CC} = 5.0V, T_A = 25°C

Symbol	Parameter	Conditions	Min	Typ	Max	Units
t_{PLH}	Driver Input to Output	R_{LDIFF} = 60Ω		12	22	ns
t_{PHL}	Driver Input to Output	$C_{L1} = C_{L2}$ = 100 pF		17	22	ns
t_r	Driver Rise Time	R_{LDIFF} = 60Ω			18	ns
t_f	Driver Fall Time	$C_{L1} = C_{L2}$ = 100 pF (*Figure 3* and *Figure 5*)			18	ns
t_{ZH}	Driver Enable to Output High	C_L = 100 pF (*Figure 4* and *Figure 6*) S1 Open		29	100	ns
t_{ZL}	Driver Enable to Output Low	C_L = 100 pF (*Figure 4* and *Figure 6*) S2 Open		31	60	ns
t_{LZ}	Driver Disable Time from Low	C_L = 15 pF (*Figure 4* and *Figure 6*) S2 Open		13	30	ns
t_{HZ}	Driver Disable Time from High	C_L = 15 pF (*Figure 4* and *Figure 6*) S1 Open		19	200	ns
t_{PLH}	Receiver Input to Output	C_L = 15 pF (*Figure 2* and *Figure 7*) S1 and S2 Closed		30	37	ns
t_{PHL}	Receiver Input to Output			32	37	ns
t_{ZL}	Receiver Enable to Output Low	C_L = 15 pF (*Figure 2* and *Figure 8*) S2 Open		15	20	ns
t_{ZH}	Receiver Enable to Output High	C_L = 15 pF (*Figure 2* and *Figure 8*) S1 Open		11	20	ns
t_{LZ}	Receiver Disable from Low	C_L = 15 pF (*Figure 2* and *Figure 8*) S2 Open		28	32	ns
t_{HZ}	Receiver Disable from High	C_L = 15 pF (*Figure 2* and *Figure 8*) S1 Open		13	35	ns

AC Test Circuits

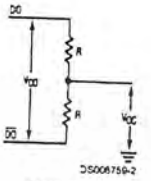
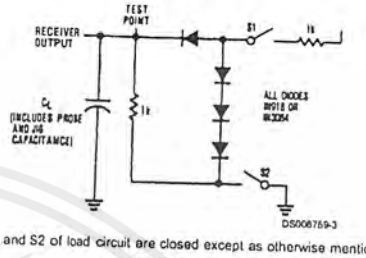


FIGURE 1.



Note: S1 and S2 of load circuit are closed except as otherwise mentioned.

FIGURE 2.

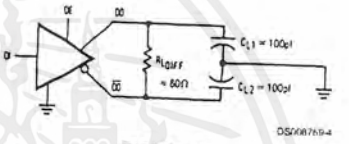
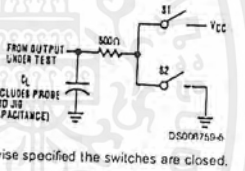


FIGURE 3.



Note: Unless otherwise specified the switches are closed.

FIGURE 4.

Switching Time Waveforms

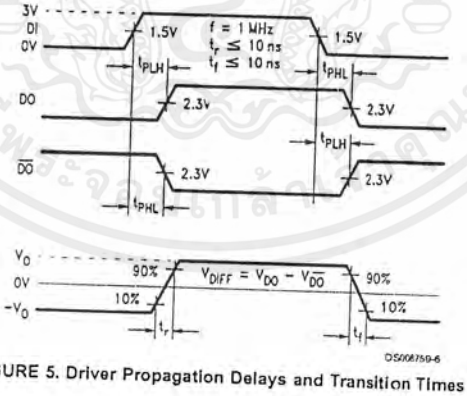


FIGURE 5. Driver Propagation Delays and Transition Times

Switching Time Waveforms (Continued)

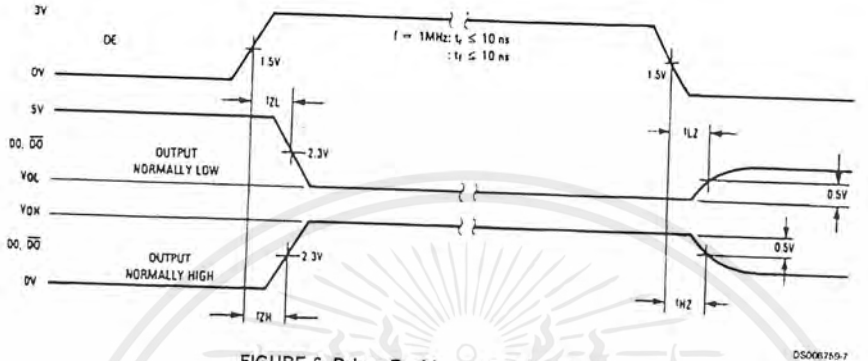
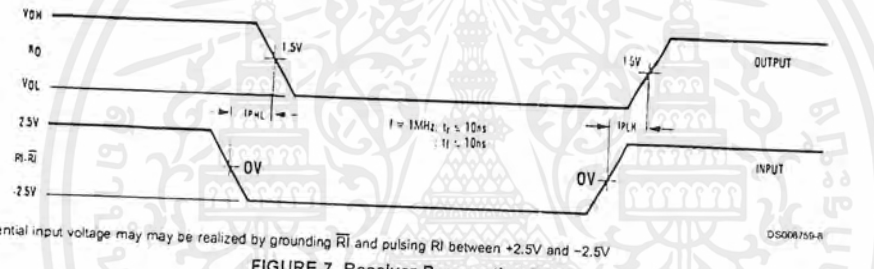


FIGURE 6. Driver Enable and Disable Times



Note: Differential input voltage may may be realized by grounding $\overline{RI}$ and pulsing RI between +2.5V and −2.5V

FIGURE 7. Receiver Propagation Delays

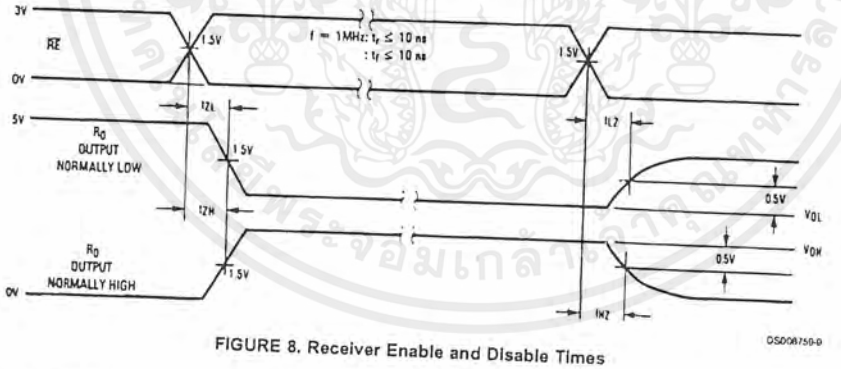


FIGURE 8. Receiver Enable and Disable Times

Function Tables

DS75176B Transmitting

Inputs			Line Condition	Outputs	
$\overline{RE}$	DE	DI		$\overline{DO}$	DO
X	1	1	No Fault	0	1
X	1	0	No Fault	1	0
X	0	X	X	Z	Z
X	1	X	Fault	Z	Z

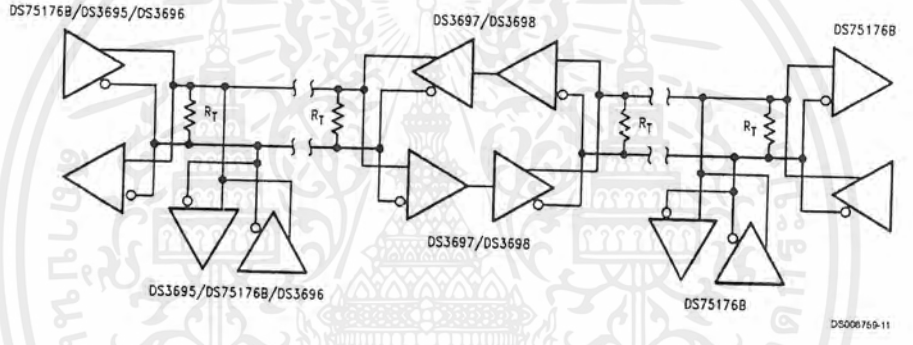
Function Tables (Continued)

DS75176B Receiving

Inputs			Outputs
$\overline{RE}$	DE	RI-$\overline{RI}$	RO
0	0	≥ +0.2V	1
0	0	≤ −0.2V	0
0	0	Inputs Open**	1
1	0	X	Z

X — Don't care condition
Z — High impedance state
Fault — Improper line conditons causing excessive power dissipation in the driver, such as shorts or bus contention situations
**This is a fail safe condition

Typical Application



Features

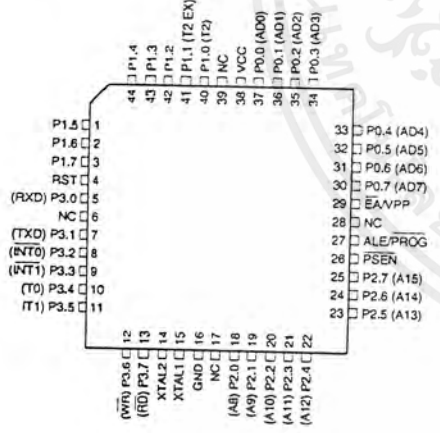
- Compatible with MCS-51™ Products
- 8K Bytes of In-System Reprogrammable Flash Memory
- Endurance: 1,000 Write/Erase Cycles
- Fully Static Operation: 0 Hz to 24 MHz
- Three-level Program Memory Lock
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight Interrupt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

Description

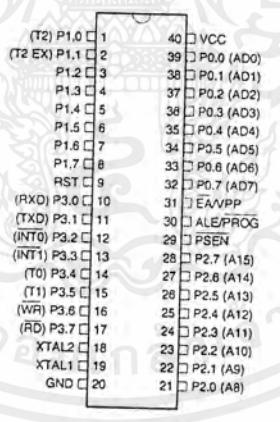
The AT89C52 is a low-power, high-performance CMOS 8-bit microcomputer with 8K bytes of Flash programmable and erasable read only memory (PEROM). The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 and 80C52 instruction 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 AT89C52 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

Pin Configurations

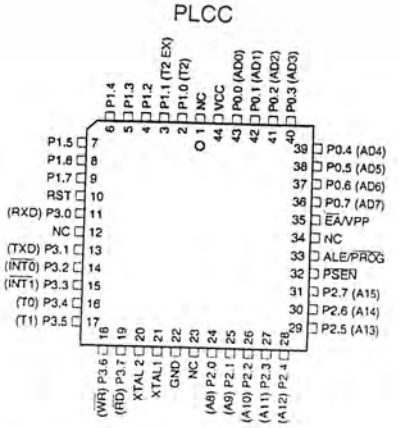
PQFP/TQFP



PDIP



PLCC

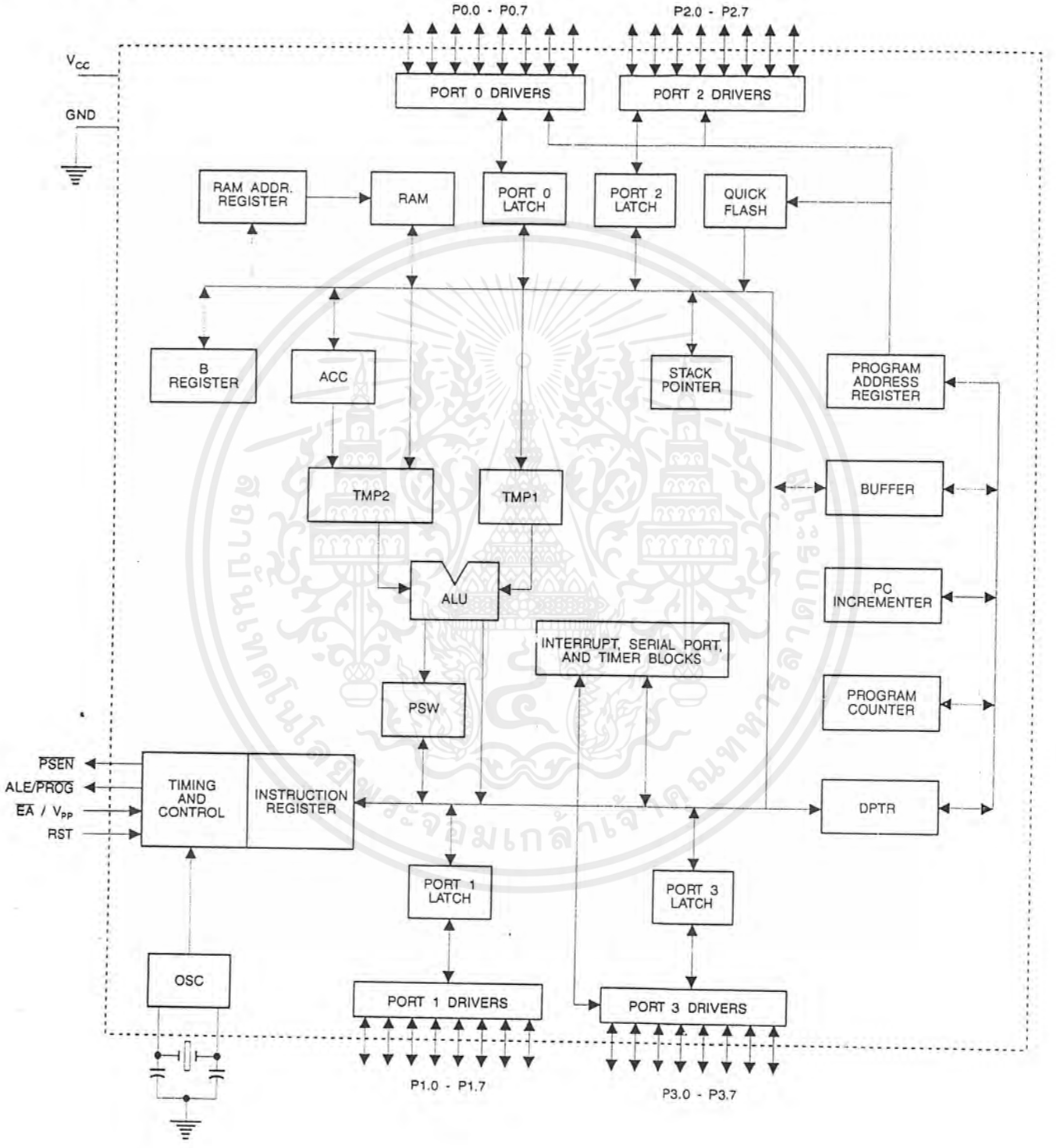


8-bit Microcontroller with 8K Bytes Flash

AT89C52

Rev. 0313H–02/00

Block Diagram



The AT89C52 provides the following standard features: 8K bytes of Flash, 256 bytes of RAM, 32 I/O lines, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full-duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89C52 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next hardware reset.

Pin Description

VCC

Supply voltage.

GND

Ground.

Port 0

Port 0 is an 8-bit open drain bi-directional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 can also be configured to be the multiplexed low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. External pullups are required during program verification.

Port 1

Port 1 is an 8-bit bi-directional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current (I_{IL}) because of the internal pullups.

In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table.

Port 1 also receives the low-order address bytes during Flash programming and verification.

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)

Port 2

Port 2 is an 8-bit bi-directional I/O port with internal pullup The Port 2 output buffers can sink/source four TTL input When 1s are written to Port 2 pins, they are pulled high b the internal pullups and can be used as inputs. As input Port 2 pins that are externally being pulled low will sourc current (I_{IL}) because of the internal pullups.

Port 2 emits the high-order address byte during fetche from external program memory and during accesses t external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pul lups when emitting 1s. During accesses to external dat memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

Port 3

Port 3 is an 8-bit bi-directional I/O port with internal pullups The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89C51, as shown in the following table.

Port 3 also receives some control signals for Flash programming and verification.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

ALE/$\overline{PROG}$

Address Latch Enable is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ($\overline{PROG}$) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external

timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

$\overline{\text{PSEN}}$

Program Store Enable is the read strobe to external program memory.

When the AT89C52 is executing code from external program memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle, except that two $\overline{\text{PSEN}}$ activations are skipped during each access to external data memory.

$\overline{\text{EA}}$/VPP

External Access Enable. $\overline{\text{EA}}$ must be strapped to GND i order to enable the device to fetch code from external pro gram memory locations starting at 0000H up to FFFFH Note, however, that if lock bit 1 is programmed, $\overline{\text{EA}}$ will b internally latched on reset.

$\overline{\text{EA}}$ should be strapped to V_{CC} for internal progran executions.

This pin also receives the 12-volt programming enable volt age (V_{PP}) during Flash programming when 12-vol programming is selected.

XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

XTAL2

Output from the inverting oscillator amplifier.

Table 1. AT89C52 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111							0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000		8FH
80H	P0 11111111	SP 00000111	DPL 00000000	DPH 00000000			PCON 0XXX0000	87H

FAIRCHILD SEMICONDUCTOR TM

August 1986
Revised March 2000

DM74LS244
Octal 3-STATE Buffer/Line Driver/Line Receiver

General Description

These buffers/line drivers are designed to improve both the performance and PC board density of 3-STATE buffers/drivers employed as memory-address drivers, clock drivers, and bus-oriented transmitters/receivers. Featuring 400 mV of hysteresis at each low current PNP data line input, they provide improved noise rejection and high fanout outputs and can be used to drive terminated lines down to 133Ω.

Features

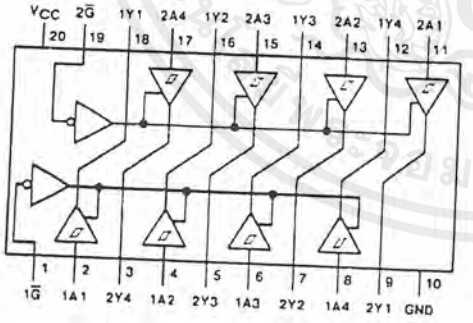
- 3-STATE outputs drive bus lines directly
- PNP inputs reduce DC loading on bus lines
- Hysteresis at data inputs improves noise margins
- Typical I_{OL} (sink current) 24 mA
- Typical I_{OH} (source current) −15 mA
- Typical propagation delay times
  - Inverting 10.5 ns
  - Noninverting 12 ns
- Typical enable/disable time 18 ns
- Typical power dissipation (enabled)
  - Inverting 130 mW
  - Noninverting 135 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS244WM	M20B	20-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS244SJ	M20D	20-Lead Small Outline Package (SOP), EIAJ TYPE II, 5.3mm Wide
DM74LS244N	N20A	20-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram



Function Table

Inputs		Output
$\overline{G}$	A	Y
L	L	L
L	H	H
H	X	Z

L = LOW Logic Level
H = HIGH Logic Level
X = Either LOW or HIGH Logic Level
Z = High Impedance

© 2000 Fairchild Semiconductor Corporation

Absolute Maximum Ratings(Note 1)

Supply Voltage	7V
Input Voltage	7V
Operating Free Air Temperature Range	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" table will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter	Min	Nom	Max	Units
V_{CC}	Supply Voltage	4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage	2			V
V_{IL}	LOW Level Input Voltage			0.8	V
I_{OH}	HIGH Level Output Current			−15	mA
I_{OL}	LOW Level Output Current			24	mA
T_A	Free Air Operating Temperature	0		70	°C

Electrical Characteristics

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions		Min	Typ (Note 2)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −18 mA				−1.5	V
HYS	Hysteresis ($V_{T+} - V_{T-}$) Data Inputs Only	V_{CC} = Min		0.2	0.4		V
V_{OH}	HIGH Level Output Voltage	V_{CC} = Min, V_{IH} = Min, V_{IL} = Max, I_{OH} = −1 mA		2.7			V
		V_{CC} = Min, V_{IH} = Min, V_{IL} = Max, I_{OH} = −3 mA		2.4	3.4		V
		V_{CC} = Min, V_{IH} = Min, V_{IL} = 0.5V, I_{OH} = Max		2			V
V_{OL}	LOW Level Output Voltage	V_{CC} = Min, V_{IL} = Max, V_{IH} = Min	I_{OL} = 12 mA			0.4	V
			I_{OL} = Max			0.5	V
I_{OZH}	Off-State Output Current, HIGH Level Voltage Applied	V_{CC} = Max, V_{IL} = Max, V_{IH} = Min	V_O = 2.7V			20	µA
I_{OZL}	Off-State Output Current, LOW Level Voltage Applied		V_O = 0.4V			−20	µA
I_I	Input Current at Maximum Input Voltage	V_{CC} = Max	V_I = 7V			0.1	mA
I_{IH}	HIGH Level Input Current	V_{CC} = Max	V_I = 2.7V			20	µA
I_{IL}	LOW Level Input Current	V_{CC} = Max	V_I = 0.4V	−0.5		−200	µA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 3)		−40		−225	mA
I_{CC}	Supply Current	V_{CC} = Max, Outputs Open	Outputs HIGH		13	23	mA
			Outputs LOW		27	46	mA
			Outputs Disabled		32	54	mA

Note 2: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 3: Not more than one output should be shorted at a time, and the duration should not exceed one second.

HARRIS SEMICONDUCTOR

CD74HC373, CD74HCT373, CD54HC573, CD74HC573, CD74HCT573

High Speed CMOS Logic
Octal Transparent Latch, Three-State Output

November 1997

Features

- Common Latch Enable Control
- Common Three-State Output Enable Control
- Buffered Inputs
- Three-State Outputs
- Bus Line Driving Capacity
- Typical Propagation Delay = 12ns at V_{CC} = 5V, C_L = 15pF, T_A = 25°C (Data to Output for HC373)
- Fanout (Over Temperature Range)
  - Standard Outputs . . . . . 10 LSTTL Loads
  - Bus Driver Outputs . . . . . 15 LSTTL Loads
- Wide Operating Temperature Range . . . -55°C to 125°C
- Balanced Propagation Delay and Transition Times
- Significant Power Reduction Compared to LSTTL Logic ICs
- HC Types
  - 2V to 6V Operation
  - High Noise Immunity: N_{IL} = 30%, N_{IH} = 30% of V_{CC} at V_{CC} = 5V
- HCT Types
  - 4.5V to 5.5V Operation
  - Direct LSTTL Input Logic Compatibility, V_{IL}= 0.8V (Max), V_{IH} = 2V (Min)
  - CMOS Input Compatibility, $I_I \leq 1\mu A$ at V_{OL}, V_{OH}

Description

The Harris CD74HC373, CD74HCT373, CD54HC573, CD74HC573, and CD74HCT573 are high speed Octal Transparent Latches manufactured with silicon gate CMOS technology. They possess the low power consumption of standard CMOS integrated circuits, as well as the ability to drive 15 LSTTL devices. The CD74HCT373 and CD74HCT573 are functionally as well as pin compatible with the standard 74LS373 and 74LS573.

The outputs are transparent to the inputs when the latch enable ($\overline{LE}$) is high. When the latch enable ($\overline{LE}$) goes low the data is latched. The output enable ($\overline{OE}$) controls the three-state outputs. When the output enable ($\overline{OE}$) is high the outputs are in the high impedance state. The latch operation is independent to the state of the output enable. The 373 and 573 are identical in function and differ only in their pinout arrangements.

Ordering Information

PART NUMBER	TEMP. RANGE (°C)	PACKAGE	PKG. NO.
CD54HC573F	-55 to 125	20 Ld CERDIP	F20.3
CD74HC373E	-55 to 125	20 Ld PDIP	F20.3
CD74HCT373E	-55 to 125	20 Ld PDIP	E20.3
CD74HC573E	-55 to 125	20 Ld PDIP	E20.3
CD74HCT573E	-55 to 125	20 Ld PDIP	E20.3
CD74HC373M	-55 to 125	20 Ld SOIC	M20.3
CD74HCT373M	-55 to 125	20 Ld SOIC	M20.3
CD74HC573M	-55 to 125	20 Ld SOIC	M20.3
CD74HCT573M	-55 to 125	20 Ld SOIC	M20.3

NOTES:

1. When ordering, use the entire part number. Add the suffix 96 to obtain the variant in the tape and reel.
2. Wafer or die for this part number are available which meets all electrical specifications. Please contact your local sales office or Harris customer service for ordering information.

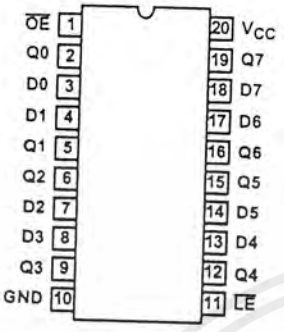
CAUTION: These devices are sensitive to electrostatic discharge. Users should follow proper IC Handling Procedures.

Copyright © Harris Corporation 1997

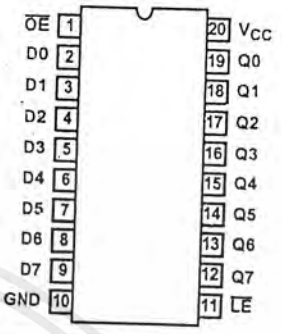
File Number 1679.1

Pinout

CD74HC373, CD74HCT373
(PDIP, SOIC)
TOP VIEW

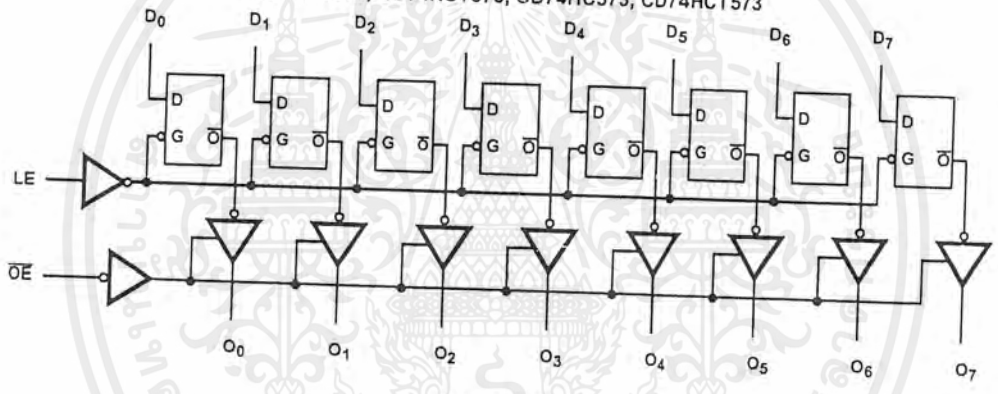


CD54HC573, CD74HC573, CD74HCT573
(PDIP, SOIC, CERDIP)
TOP VIEW

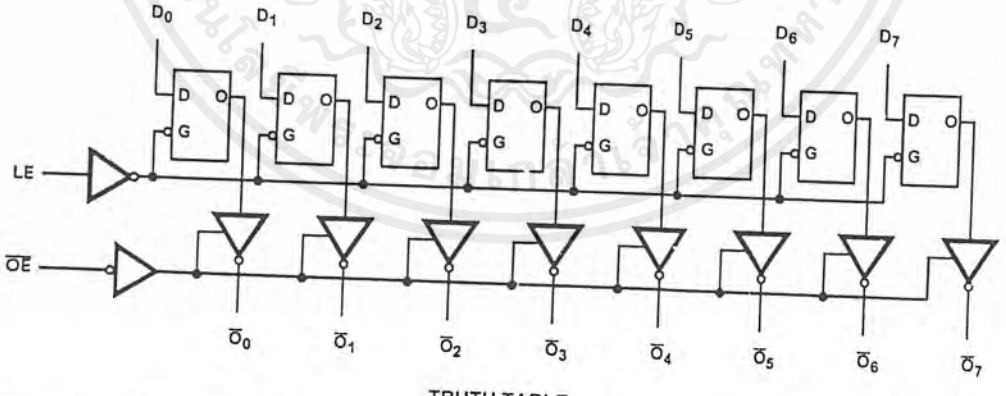


Functional Block Diagrams

CD74HC373, CD74HCT373, CD74HC573, CD74HCT573



CD74HCT573



TRUTH TABLE

OUTPUT ENABLE	LATCH ENABLE	DATA	OUTPUT
L	H	H	H
L	H	L	L
L	L	l	L
L	L	h	H
H	X	X	Z

NOTE: H = High Voltage Level, L = Low Voltage Level, X = Don't Care, Z = High Impedance State, l = Low voltage level one set-up time prior to the high to low latch enable transition, h = High voltage level one set-up time prior to the high to low latch enable transition.

Absolute Maximum Ratings

DC Supply Voltage, V_{CC} . . . . . -0.5V to 7V
DC Input Diode Current, I_{IK}
For V_I < -0.5V or V_I > V_{CC} + 0.5V . . . . . ±20mA
DC Output Diode Current, I_{OK}
For V_O < -0.5V or V_O > V_{CC} + 0.5V . . . . . ±20mA
DC Drain Current, per Output, I_O
For -0.5V < V_O < V_{CC} + 0.5V . . . . . ±35mA
DC Output Source or Sink Current per Output Pin, I_O
For V_O > -0.5V or V_O < V_{CC} + 0.5V . . . . . ±25mA
DC V_{CC} or Ground Current, I_{CC} . . . . . ±50mA

Operating Conditions

Temperature Range, T_A . . . . . -55°C to 125°C
Supply Voltage Range, V_{CC}
HC Types . . . . . 2V to 6V
HCT Types . . . . . 4.5V to 5.5V
DC Input or Output Voltage, V_I, V_O . . . . . 0V to V_{CC}
Input Rise and Fall Time
2V . . . . . 1000ns (Max)
4.5V . . . . . 500ns (Max)
6V . . . . . 400ns (Max)

Thermal Information

Thermal Resistance (Typical, Note 3)	θ_{JA} (°C/W)	θ_{JA} (°C/W)
PDIP Package	125	N/A
CERDIP Package	85	24
SOIC Package	120	N/A

Maximum Junction Temperature (Plastic Package) . . . . . 150°C
Maximum Storage Temperature Range . . . . . -65°C to 150°C
Maximum Lead Temperature (Soldering 10s) . . . . . 300°C
(SOIC - Lead Tips Only)

CAUTION: Stresses above those listed in "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress only rating and operation of the device at these or any other conditions above those indicated in the operational sections of this specification is not implied.

NOTE:

3. θ_{JA} is measured with the component mounted on an evaluation PC board in free air.

DC Electrical Specifications

PARAMETER	SYMBOL	TEST CONDITIONS V_I (V)	TEST CONDITIONS I_O (mA)	V_{CC} (V)	25°C MIN	25°C TYP	25°C MAX	-40°C TO 85°C MIN	-40°C TO 85°C MAX	-55°C TO 125°C MIN	-55°C TO 125°C MAX	UNITS
HC TYPES												
High Level Input Voltage	V_{IH}	-	-	2	1.5	-	-	1.5	-	1.5	-	V
				4.5	3.15	-	-	3.15	-	3.15	-	V
				6	4.2	-	-	4.2	-	4.2	-	V
Low Level Input Voltage	V_{IL}	-	-	2	-	-	0.5	-	0.5	-	0.5	V
				4.5	-	-	1.35	-	1.35	-	1.35	V
				6	-	-	1.8	-	1.8	-	1.8	V
High Level Output Voltage CMOS Loads	V_{OH}	V_{IH} or V_{IL}	-0.02	2	1.9	-	-	1.9	-	1.9	-	V
			-0.02	4.5	4.4	-	-	4.4	-	4.4	-	V
			-0.02	6	5.9	-	-	5.9	-	5.9	-	V
High Level Output Voltage TTL Loads			-6	4.5	3.98	-	-	3.84	-	3.7	-	V
			-7.8	6	5.48	-	-	5.34	-	5.2	-	V
Low Level Output Voltage CMOS Loads	V_{OL}	V_{IH} or V_{IL}	0.02	2	-	-	0.1	-	0.1	-	0.1	V
			0.02	4.5	-	-	0.1	-	0.1	-	0.1	V
			0.02	6	-	-	0.1	-	0.1	-	0.1	V
Low Level Output Voltage TTL Loads			6	4.5	-	-	0.26	-	0.33	-	0.4	V
			7.8	6	-	-	0.26	-	0.33	-	0.4	V
Input Leakage Current	I_I	V_{CC} or GND	-	6	-	-	±0.1	-	±1	-	±1	µA
Quiescent Device Current	I_{CC}	V_{CC} or GND	0	6	-	-	8	-	80	-	160	µA

DC Electrical Specifications (Continued)

PARAMETER	SYMBOL	TEST CONDITIONS V_I (V)	TEST CONDITIONS I_O (mA)	V_{CC} (V)	25°C MIN	25°C TYP	25°C MAX	-40°C TO 85°C MIN	-40°C TO 85°C MAX	-55°C TO 125°C MIN	-55°C TO 125°C MAX	UNITS
Three-State Leakage Current	-	V_{IL} or V_{IH}	V_O = V_{CC} or GND	6	-	-	±0.5	-	±5	-	±10	µA
HCT TYPES												
High Level Input Voltage	V_{IH}	-	-	4.5 to 5.5	2	-	-	2	-	2	-	V
Low Level Input Voltage	V_{IL}	-	-	4.5 to 5.5	-	-	0.8	-	0.8	-	0.8	V
High Level Output Voltage CMOS Loads	V_{OH}	V_{IH} or V_{IL}	-0.02	4.5	4.4	-	-	4.4	-	4.4	-	V
High Level Output Voltage TTL Loads			-6	4.5	3.98	-	-	3.84	-	3.7	-	V
			-7.8	6	5.48	-	-	5.34	-	5.2	-	V
Low Level Output Voltage CMOS Loads	V_{OL}	V_{IH} or V_{IL}	0.02	4.5	-	-	0.1	-	0.1	-	0.1	V
Low Level Output Voltage TTL Loads			6	4.5	-	-	0.26	-	0.33	-	0.4	V
			7.8	6	-	-	0.26	-	0.33	-	0.4	V
Input Leakage Current	I_I	V_{CC} to GND	-	5.5	-	-	±0.1	-	±1	-	±1	µA
Quiescent Device Current	I_{CC}	V_{CC} or GND	0	5.5	-	-	8	-	80	-	160	µA
Three-State Leakage Current	-	V_{IL} or V_{IH}	V_O = V_{CC} or GND	6	-	-	±0.5	-	±5	-	±10	µA
Additional Quiescent Device Current Per Input Pin: 1 Unit Load (Note 4)	ΔI_{CC}	V_{CC} -2.1	-	4.5 to 5.5	-	100	360	-	450	-	490	µA

NOTE:

4. For dual-supply systems theoretical worst case (V_I = 2.4V, V_{CC} = 5.5V) specification is 1.8mA.

HCT Input Loading Table

INPUT	UNIT LOADS HCT373	UNIT LOADS HCT573
$\overline{OE}$	1.5	1.25
Dn	0.4	0.3
LE	0.6	0.65

NOTE: Unit Load is ΔI_{CC} limit specified in DC Electrical Specifications table, e.g., 360µA max at 25°C.

Prerequisite For Switching Specifications

PARAMETER	SYMBOL	TEST CONDITIONS	V_{CC} (V)	25°C			-40°C TO 85°C		-55°C TO 125°C		UNITS
				MIN	TYP	MAX	MIN	MAX	MIN	MAX	
HC TYPES											
$\overline{LE}$ Pulse Width	t_W	-	2	80	-	-	100	-	120	-	ns
			4.5	16	-	-	20	-	24	-	ns
			6	14	-	-	17	-	20	-	ns
Set-up Time Data to $\overline{LE}$	t_{SU}	-	2	50	-	-	65	-	75	-	ns
			4.5	10	-	-	13	-	15	-	ns
			6	9	-	-	11	-	13	-	ns
Hold Time, Data to $\overline{LE}$ (573)	t_H	-	2	40	-	-	50	-	60	-	ns
			4.5	8	-	-	10	-	12	-	ns
			6	7	-	-	9	-	10	-	ns
Hold Time, Data to $\overline{LE}$ (373)	t_H	-	2	5	-	-	5	-	5	-	ns
			4.5	5	-	-	5	-	5	-	ns
			6	5	-	-	5	-	5	-	ns
HCT TYPES											
$\overline{LE}$ Pulse Width	t_W	-	4.5	16	-	-	20	-	24	-	ns
Set-up Time Data to $\overline{LE}$	t_W	-	4.5	13	-	-	16	-	20	-	ns
Hold Time, Data to $\overline{LE}$	t_H	-	4.5	10	-	-	13	-	15	-	ns

Switching Specifications Input t_r, t_f = 6ns

PARAMETER	SYMBOL	TEST CONDITIONS	V_{CC} (V)	25°C		-40°C TO 85°C	-55°C TO 125°C	UNITS
				TYP	MAX	MAX	MAX	
HC TYPES								
Propagation Delay, Data to Qn (HC/HCT373)	t_{PLH}, t_{PHL}	C_L = 50pF	2	-	150	190	225	ns
			4.5	-	30	38	45	ns
			6	-	26	33	38	ns
		C_L = 15pF	5	12	-	-	-	ns
Propagation Delay, Data to Qn (HC/HCT573)	t_{PLH}, t_{PHL}	C_L = 50pF	2	-	175	220	265	ns
			4.5	-	35	44	53	ns
			6	-	30	37	45	ns
		C_L = 15pF	5	14	-	-	-	ns
Propagation Delay, $\overline{LE}$ to Qn	t_{PLH}, t_{PHL}	C_L = 50pF	2	-	175	220	265	ns
			4.5	-	35	44	53	ns
			6	-	30	37	45	ns
		C_L = 15pF	5	14	-	-	-	ns
Output Enabling Time	t_{PZL}, t_{PZH}	C_L = 50pF	2	-	150	190	225	ns
			4.5	-	30	38	45	ns
			6	-	26	33	38	ns
		C_L = 15pF	5	12	-	-	-	ns

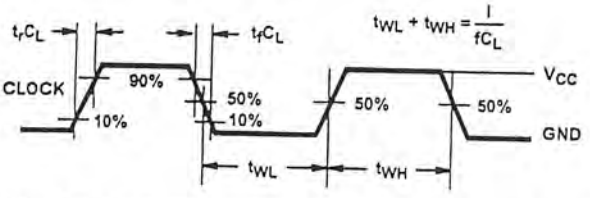
Switching Specifications Input t_r, t_f = 6ns **(Continued)**

PARAMETER	SYMBOL	TEST CONDITIONS	V_{CC} (V)	25°C		-40°C TO 85°C	-55°C TO 125°C	UNITS
				TYP	MAX	MAX	MAX	
Output Disabling Time	t_{PLZ}, t_{PHZ}	C_L = 50pF	2	-	150	190	225	ns
			4.5	-	30	38	45	ns
			6	-	26	33	38	ns
		C_L = 15pF	5	12	-	-	-	ns
Output Transition Time	t_{TLH}, t_{THL}	C_L = 50pF	2	-	60	75	90	ns
			4.5	-	12	15	18	ns
			6	-	10	13	15	ns
Input Capacitance	C_I	-	-	-	10	10	10	pF
Three-State Output Capacitance	C_O	-	-	-	20	20	20	pF
Power Dissipation Capacitance (Notes 5, 6)	C_{PD}	-	5	51	-	-	-	pF
HCT TYPES								
Propagation Delay, Data to Qn (HC/HCT373)	t_{PLH}, t_{PHL}	C_L = 50pF	4.5	-	32	40	48	ns
		C_L = 15pF	5	13	-	-	-	ns
Propagation Delay, Data to Qn (HC/HCT573)	t_{PLH}, t_{PHL}	C_L = 50pF	4.5	-	35	44	53	ns
		C_L = 15pF	5	17	-	-	-	ns
Propagation Delay, $\overline{LE}$ to Qn	t_{PLH}, t_{PHL}	C_L = 50pF	4.5	-	35	44	53	ns
		C_L = 15pF	5	14	-	-	-	ns
Output Enabling Time	t_{PZL}, t_{PZH}	C_L = 50pF	4.5	-	35	44	53	ns
		C_L = 15pF	5	14	-	-	-	ns
Output Disabling Time	t_{PLZ}, t_{PZH}	C_L = 50pF	4.5	-	35	44	53	ns
		C_L = 15pF	5	14	-	-	-	ns
Output Transition Time	t_{TLH}, t_{THL}	C_L = 50pF	4.5	-	12	15	18	ns
Input Capacitance	C_I	-	-	-	10	10	10	pF
Three-State Output Capacitance	C_O	-	-	-	20	20	20	pF
Power Dissipation Capacitance (Notes 5, 6)	C_{PD}	-	5	53	-	-	-	pF

NOTES:

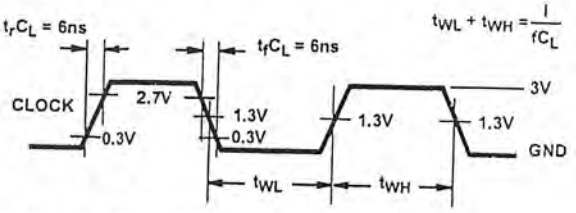
5. C_{PD} is used to determine the no-load dynamic power consumption, per latch.
6. P_D (total power per latch) = $V_{CC}^2 f_i (C_{PD} + C_L)$ where f_i = Input Frequency, C_L = Output Load Capacitance, V_{CC} = Supply Voltage.

Test Circuits and Waveforms



NOTE: Outputs should be switching from 10% V_{CC} to 90% V_{CC} in accordance with device truth table. For f_{MAX}, input duty cycle = 50%.

FIGURE 1. HC CLOCK PULSE RISE AND FALL TIMES AND PULSE WIDTH



NOTE: Outputs should be switching from 10% V_{CC} to 90% V_{CC} in accordance with device truth table. For f_{MAX}, input duty cycle = 50%.

FIGURE 2. HCT CLOCK PULSE RISE AND FALL TIMES AND PULSE WIDTH

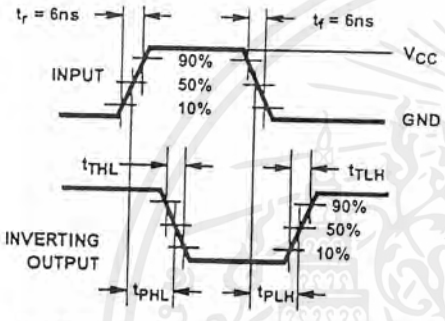


FIGURE 3. HC TRANSITION TIMES AND PROPAGATION DELAY TIMES, COMBINATION LOGIC

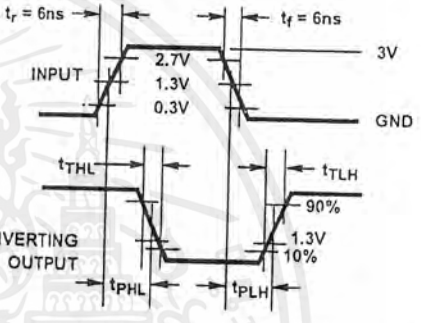


FIGURE 4. HCT TRANSITION TIMES AND PROPAGATION DELAY TIMES, COMBINATION LOGIC

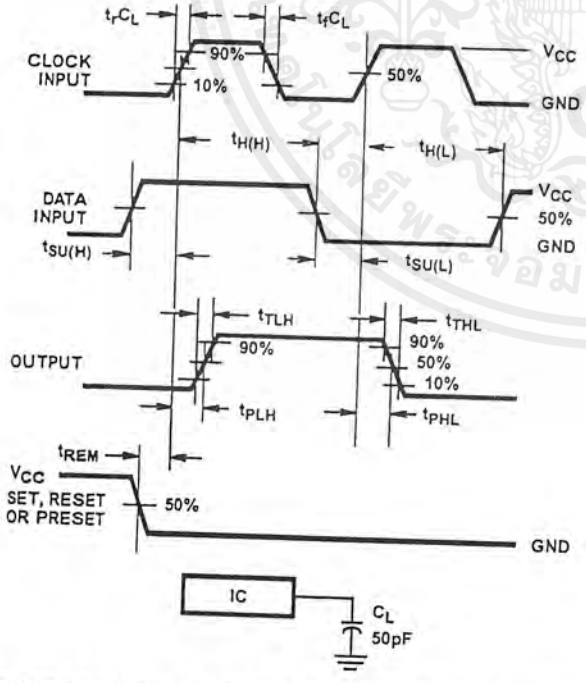


FIGURE 5. HC SETUP TIMES, HOLD TIMES, REMOVAL TIME, AND PROPAGATION DELAY TIMES FOR EDGE TRIGGERED SEQUENTIAL LOGIC CIRCUITS

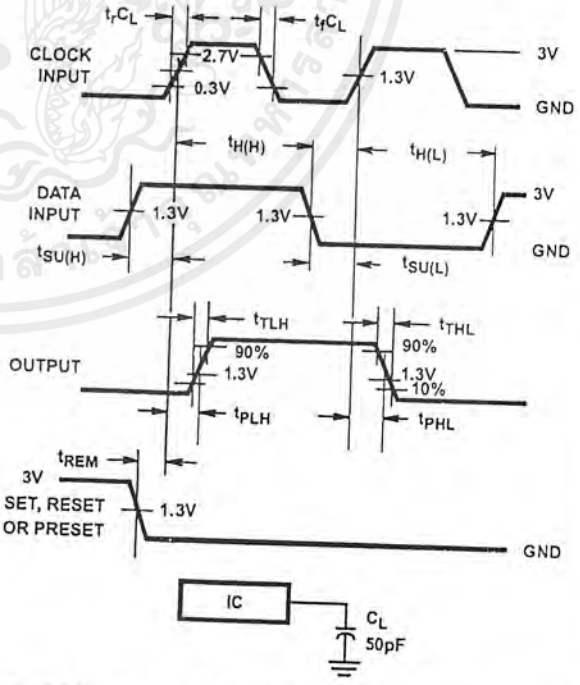


FIGURE 6. HCT SETUP TIMES, HOLD TIMES, REMOVAL TIME, AND PROPAGATION DELAY TIMES FOR EDGE TRIGGERED SEQUENTIAL LOGIC CIRCUITS

Test Circuits and Waveforms (Continued)

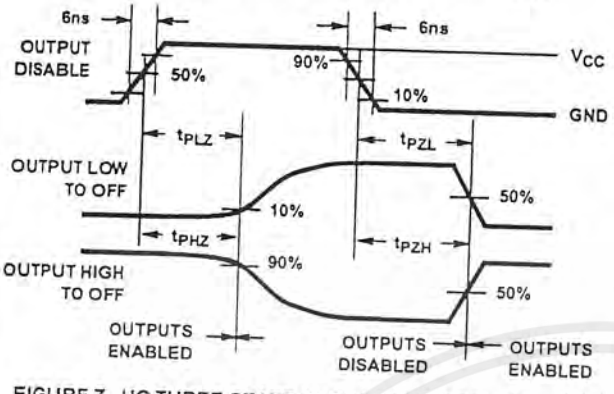


FIGURE 7. HC THREE-STATE PROPAGATION DELAY WAVEFORM

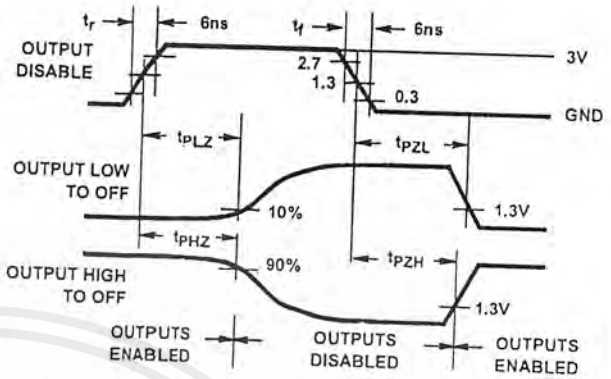
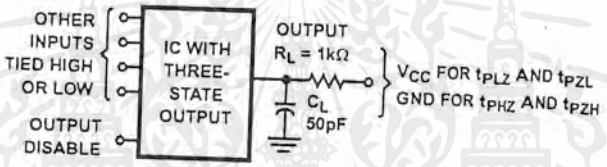


FIGURE 8. HCT THREE-STATE PROPAGATION DELAY WAVEFORM



NOTE: Open drain waveforms t_{PLZ} and t_{PZL} are the same as those for three-state shown on the left. The test circuit is Output R_L = 1kΩ to V_{CC}, C_L = 50pF.

FIGURE 9. HC AND HCT THREE-STATE PROPAGATION DELAY TEST CIRCUIT

All Harris Semiconductor products are manufactured, assembled and tested under ISO9000 quality systems certification.

Harris Semiconductor products are sold by description only. Harris Semiconductor reserves the right to make changes in circuit design and/or specifications at any time without notice. Accordingly, the reader is cautioned to verify that data sheets are current before placing orders. Information furnished by Harris is believed to be accurate and reliable. However, no responsibility is assumed by Harris or its subsidiaries for its use; nor for any infringements of patents or other rights of third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of Harris or its subsidiaries.

Sales Office Headquarters

For general information regarding Harris Semiconductor and its products, call **1-800-4-HARRIS**

NORTH AMERICA
Harris Semiconductor
P. O. Box 883, Mail Stop 53-210
Melbourne, FL 32902
TEL: 1-800-442-7747
(407) 729-4984
FAX: (407) 729-5321

EUROPE
Harris Semiconductor
Mercure Center
100, Rue de la Fusee
1130 Brussels, Belgium
TEL: (32) 2.724.2111
FAX: (32) 2.724.22.05

ASIA
Harris Semiconductor PTE Ltd.
No. 1 Tannery Road
Cencon 1, #09-01
Singapore 1334
TEL: (65) 748-4200
FAX: (65) 748-0400

HARRIS SEMICONDUCTOR

19-4323; Rev 6: 3/96

MAXIM

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

General Description

The MAX220-MAX249 family of line drivers/receivers is intended for all EIA/TIA-232E and V.28/V.24 communications interfaces, and in particular, for those applications where ±12V is not available.

These parts are particularly useful in battery-powered systems, since their low-power shutdown mode reduces power dissipation to less than 5µW. The MAX225, MAX233, MAX235, and MAX245-MAX247 use no external components and are recommended for applications where printed circuit board space is critical.

Applications

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

Features

Superior to Bipolar

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

Ordering Information

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

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

Selection Table

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

For free samples & the latest literature: http://www.maxim-ic.com, or phone 1-800-998-8800

+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

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

Supply Voltage (V_{CC}) ..... -0.3V to +6V
Input Voltages
T_{IN} ..... -0.3V to (V_{CC} - 0.3V)
R_{IN} ..... ±30V
T_{OUT} (Note 1) ..... ±15V
Output Voltages
T_{OUT} ..... ±15V
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Driver/Receiver Output Short Circuited to GND ..... Continuous
Continuous Power Dissipation (T_A = +70°C)
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
18-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) 440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C) 696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C) 842mW
Operating Temperature Ranges
MAX2_ _AC_ _, MAX2_ _C_ _ ..... 0°C to +70°C
MAX2_ _AE_ _, MAX2_ _E_ _ ..... -40°C to +85°C
MAX2_ _AM_ _, MAX2_ _M_ _ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{SHDN}$ or V_{CC} = 0V.

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

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

(V_{CC} = +5V ±10%, C1-C4 = 0.1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	Normal operation			5	40	µA
	$\overline{SHDN}$ = 0V, MAX222/242, shutdown			±0.01	±1	
Output Leakage Current	V_{CC} = 5.5V, $\overline{SHDN}$ = 0V, V_{OUT} = ±15V, MAX222/242			±0.01	±10	µA
	V_{CC} = $\overline{SHDN}$ = 0V, V_{OUT} = ±15V			±0.01	±10	
Data Rate	Except MAX220, normal operation			200	116	kbits/ sec
	MAX220			22	20	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V		300	10M		Ω
Output Short-Circuit Current	V_{OUT} = 0V		±7	±22		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±30	V
RS-232 Input Threshold Low	V_{CC} = 5V	Except MAX243 $R2_{IN}$	0.8	1.3		V
		MAX243 $R2_{IN}$ (Note 2)	-3			
RS-232 Input Threshold High	V_{CC} = 5V	Except MAX243 $R2_{IN}$		1.8	2.4	V
		MAX243 $R2_{IN}$ (Note 2)		-0.5	-0.1	
RS-232 Input Hysteresis	Except MAX243, V_{CC} = 5V, no hyst. in shdn.		0.2	0.5	1	V
	MAX243			1		
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		
TTL/CMOS Output Leakage Current	$\overline{SHDN}$ = V_{CC} or $\overline{EN}$ = V_{CC} ($\overline{SHDN}$ = 0V for MAX222), 0V ≤ V_{OUT} ≤ V_{CC}			±0.05	±10	µA

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued)

(V_{CC} = +5V ±10%, C1-C4 = 0.1µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

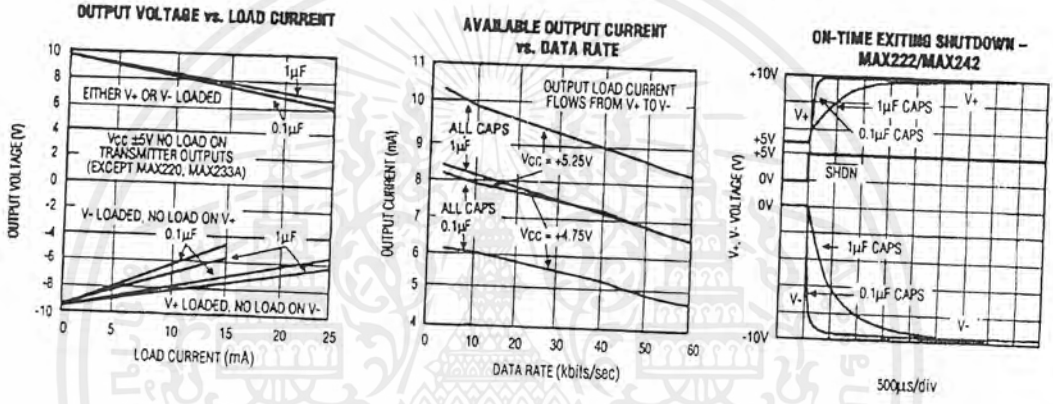
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
$\overline{EN}$ Input Threshold Low	MAX242			1.4	0.8	V
$\overline{EN}$ Input Threshold High	MAX242		2.0	1.4		V
POWER SUPPLY						
Operating Supply Voltage			4.5		5.5	V
V_{CC} Supply Current ($\overline{SHDN}$ = V_{CC}), Figures 5, 6, 9, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	T_A = +25°C		0.1	10	µA
		T_A = 0° to +70°C		2	50	
		T_A = -40° to +85°C		2	50	
		T_A = -55° to +125°C		35	100	
$\overline{SHDN}$ Input Leakage Current	MAX222/242				±1	µA
$\overline{SHDN}$ Threshold Low	MAX222/242			1.4	0.8	V
$\overline{SHDN}$ Threshold High	MAX222/242		2.0	1.4		V
AC CHARACTERISTICS						
Transition Slew Rate	C_L = 50pF to 2500pF, R_L = 3kΩ to 7kΩ, V_{CC} = 5V, T_A = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/µs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t_{PHLT}	MAX222/232A/233A/242/243		1.3	3.5	µs
		MAX220		4	10	
	t_{PLHT}	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TLL (normal operation), Figure 2	t_{PHLR}	MAX222/232A/233A/242/243		0.5	1	µs
		MAX220		0.6	3	
	t_{PLHR}	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TLL (shutdown), Figure 2	t_{PHLS}	MAX242		0.5	10	µs
	t_{PLHS}	MAX242		2.5	10	
Receiver-Output Enable Time, Figure 3	t_{ER}	MAX242		125	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}	MAX242		160	500	ns
Transmitter-Output Enable Time ($\overline{SHDN}$ goes high), Figure 4	t_{ET}	MAX222/242, 0.1µF caps (includes charge-pump start-up)		250		µs
Transmitter-Output Disable Time ($\overline{SHDN}$ goes low), Figure 4	t_{DT}	MAX222/242, 0.1µF caps		600		ns
Transmitter + to - Propagation Delay Difference (normal operation)	t_{PHLT} - t_{PLHT}	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (normal operation)	t_{PHLR} - t_{PLHR}	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

Note 2: MAX243 $R2_{OUT}$ is guaranteed to be low when $R2_{IN}$ is ≥ 0V or is floating.

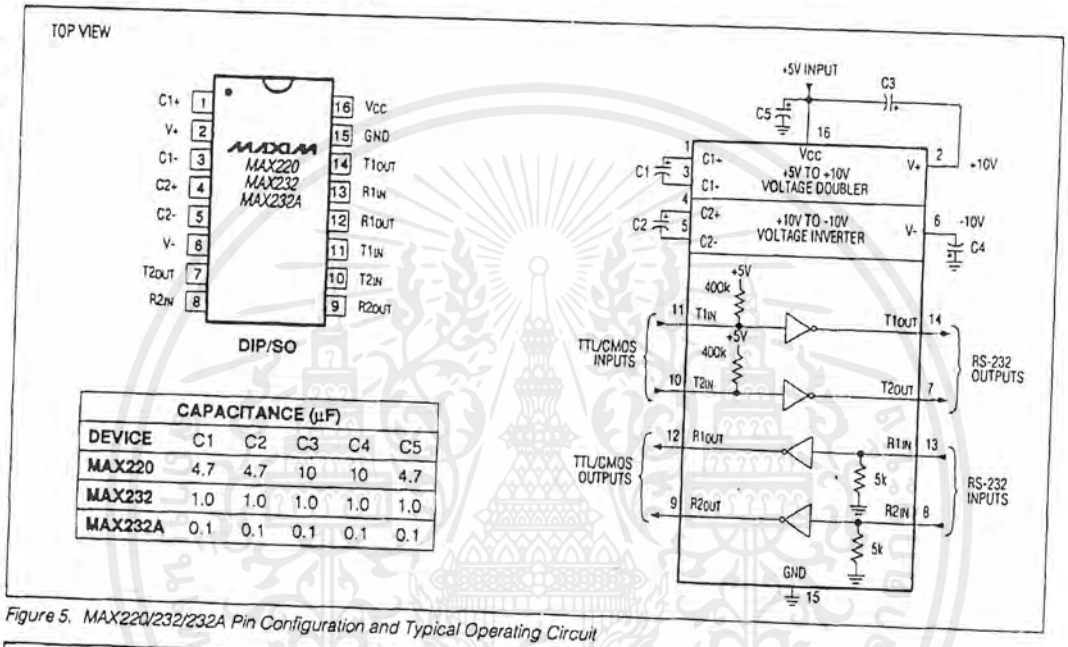
+5V-Powered, Multi-Channel RS-232 Drivers/Receivers

Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243



+5V-Powered, Multi-Channel RS-232 Drivers/Receivers



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/232/232A Pin Configuration and Typical Operating Circuit

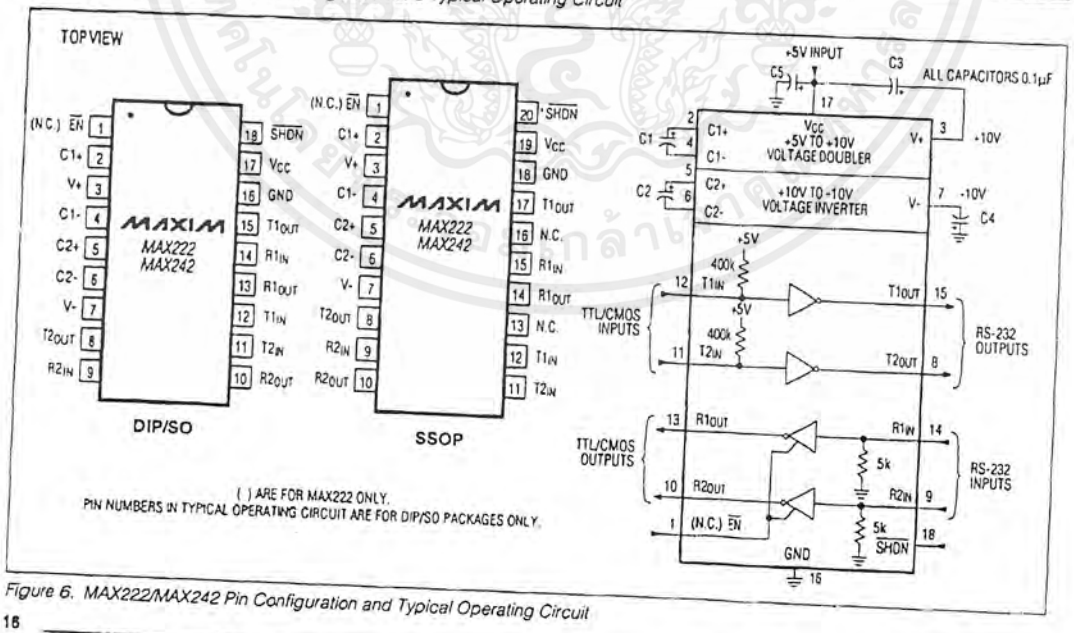


Figure 6. MAX222/MAX242 Pin Configuration and Typical Operating Circuit

กิตติกรรมประกาศ

ขอขอบพระคุณท่าน รศ. สมยศ จุณณะปิยะ ที่ให้คำปรึกษา พร้อมทั้งสถานที่และอุปกรณ์เครื่องมือต่างๆ ในการทำโครงงานนี้มาโดยตลอด

หนังสืออ้างอิง

[1] กฤษดา ใจเย็น, อรรถพล บุญยะโภคา, ชัยวัฒน์ ลิ้มพรจิตรวิไล "เรียนรู้และปฏิบัติการเชื่อมต่อ-คอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ตอนุกรม" กรุงเทพฯ, 2540

[2] ชัยวัฒน์ ลิ้มพรจิตรวิไล, วรพจน์ กรแก้ววัฒนกุล "เรียนรู้และปฏิบัติการไมโครคอนโทรลเลอร์ MCS-51" กรุงเทพฯ, 2543

[3] สมยศ จุณณะปิยะ "การใช้งานไมโครคอนโทรลเลอร์ตระกูล MCS – 51" พิมพ์ครั้งที่ 3, กรุงเทพฯ : โครงการตำราสถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, 2543