สำนักหอสมุดกลาง พระจอมเกล้าลาดกระบัง

ภาควิชาครุศาสตร์วิศวกรรม

คณะครุศาสตร์อุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ใบรับรองปริญญานิพนธ์

.....

ปริญญานิพนธ์ ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

INDUSTRIAL MICROCOMPUTER TRAINING

ชื่อนักศึกษา

1. น างสาวกนกพร กสิกพันธุ์ รหัสประจำตัว 39031301
2. นายประวิทย์ ลาวัณย์วิสุทธิ์ รหัสประจำตัว 39031316
3. นายประสิทธิ์ ภูมิภาค รหัสประจำตัว 39031317
4. นายวิชาญ เพชรมณี รหัสประจำตัว 39031327

หลักสูตร ครุศาสตร์อุตสาหกรรมบัณฑิต **สาขาวิชา** อิเล็กทรอนิกส์และคอมพิวเตอร์

อาจารย์ผู้ควบคุมปริญญานิพนธ์

1. อาจารย์สุชิน อาจหาญ
2. อาจารย์อำพล ทองระอา
3. อาจารย์กิติพงศ์ มะโน

คณะกรรมการสอบปริญญานิพนธ์	ลายมือชื่อ
1. อาจารย์สุชิน อาจหาญ	
2. อาจารย์อำพล ทองระอา	
3. อาจารย์กิติพงศ์ มะโน	
4. อาจารย์ปิยะ จิตธรรมมาภิรมย์	
5. อาจารย์ไพบูลย์ พวงวงศ์ตระกูล	

วันเดือนปีที่สอบ วันที่ 5 ธันวาคม 2540 เวลา 16.00 ถึง 18.00 น.

สถานที่สอบ ห้อง ค.310 คณะครุศาสตร์อุตสาหกรรม

ภาควิชารับรองแล้ว

.....

(ผศ.ดร.ประเทศ เทพหัสดิน ณ อยุธยา)

ประธานภาควิชาครุศาสตร์วิศวกรรม

วันที่ ... เดือน ... พ.ศ. ...

ปริญญานิพนธ์

ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

INDUSTRIAL MICROCOMPUTER TRAINING

นางสาวกนกพร กสิกพันธุ์

นายประวิทย์ ลาวัณย์วิสุทธิ์

นายประสิทธิ์ ภูมิภาค

นายวิชาญ เพชรมณี

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรครุศาสตร์อุตสาหกรรมบัณฑิต

สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์

ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

ปริญญานิพนธ์

เรื่อง ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

INDUSTRIAL MICROCOMPUTER TRAINING

ผู้จัดทำ

1. นางสาวกนกพร กสิกพันธุ์
2. นายประวิทย์ ลาวัณย์วิสุทธิ์
3. นายประสิทธิ์ ภูมิภาค
4. นายวิชาญ เพชรมณี

อาจารย์ที่ปรึกษา

ลงนาม .....

(อาจารย์สุชิน อาจหาญ)

ลงนาม .....

(อาจารย์อำพล ทองระอา)

ลงนาม .....

(อาจารย์กิติพงค์ มะโน)

หัวหน้าภาควิชาครุศาสตร์วิศวกรรม

ลงนาม .....

(ผศ. ดร. ธีระพล เทพหัสดิน ณ อยุธยา)

ปริญญานิพนธ์

เรื่อง ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

INDUSTRIAL MICROCOMPUTER TRAINING

วัตถุประสงค์

1. เพื่อใช้เป็นชุดฝึกเพื่อการศึกษาการอินเตอร์เฟสระหว่างคอมพิวเตอร์กับอุปกรณ์ภายนอก
2. เพื่อศึกษาหลักการทำงานของระบบการเชื่อมต่อของคอมพิวเตอร์กับอุปกรณ์ภายนอก
3. เพื่อออกแบบวงจรการเชื่อมต่อของคอมพิวเตอร์กับอุปกรณ์ภายนอก
4. เพื่อศึกษาการเขียนโปรแกรมที่ใช้ควบคุมการทำงานของอุปกรณ์ภายนอก

ประโยชน์ที่คาดว่าจะได้รับ

1. ได้นำชุดฝึกที่ออกแบบไว้ไปใช้ในการทดลองและเรียนรู้การอินเตอร์เฟสระหว่างคอมพิวเตอร์กับอุปกรณ์ภายนอก
2. ได้รู้และเข้าใจหลักการและระบบการเชื่อมต่อของคอมพิวเตอร์กับอุปกรณ์ภายนอก
3. ได้วงจรการเชื่อมต่อของคอมพิวเตอร์กับอุปกรณ์ภายนอก
4. ได้รู้และเข้าใจการเขียนโปรแกรมที่ใช้ควบคุมการทำงานของอุปกรณ์ภายนอก

ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

นางสาวกนกพร กสิกพันธุ์
นายประวิทย์ ลาวัณย์วิสุทธิ์
นายประสิทธิ์ ภูมิภาค
นายวิชาญ เพชรมณี

อาจารย์ที่ปรึกษา
อาจารย์สุชิน อาจหาญ
อาจารย์อำพล ทองระอา
อาจารย์กิติพงค์ มะโน

ปีการศึกษา 2540

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้นำเสนอชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม โดยใช้หลักการของการอินเตอร์เฟสระหว่างคอมพิวเตอร์กับอุปกรณ์ภายนอก หลักการทำงานของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมอาศัยการรับข้อมูลจากคอมพิวเตอร์ไปยังการ์ดเฉพาะงานและส่งผ่านไปยังช่องนำสัญญาณหรือสล็อต DB37 เพื่อนำข้อมูลจากคอมพิวเตอร์ไปควบคุมอุปกรณ์ภายนอกต่อไป

ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมสามารถนำไปใช้ประกอบเป็นชุดฝึกเพื่อใช้ในการเรียนการสอนในวิชาที่เกี่ยวข้อง โดยในชุดฝึกได้จัดทำใบงานประกอบอย่างเหมาะสมเพื่อให้การเรียนการสอนสมบูรณ์ยิ่งขึ้น

INDUSTRIAL MICROCOMPUTER TRAINING

MISS.KANOKPORN KASIKPAN
MR.PRAWIT LAWANWISUT
MR.PASIT PHUMMIPHAK
MR.WICHAN PHETMANEE

ADVISORS

MR.SUCHIN ACHARN
MR.AMPHON THONGRA-AR
MR.KITIPONG MANO

1997

ABSTRACT

This thesis is presents the project of industrial microcomputer training. It is used to Interface between computer with a peripheral. And work by receive data pass Adapter Card and send a data to DB37 slot for take a data from computer to control a peripheral from Interface Board.

Industrial microcomputer training is use to train for education. And for more perfect it has laboratory training to use with education.

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จลุล่วงไปด้วยดีด้วยความช่วยเหลือและการให้คำแนะนำปรึกษาจากท่านอาจารย์ที่เป็นที่ปรึกษาทุกท่าน รวมทั้งสมาชิกในกลุ่มที่ร่วมมือกันทำงานจนสำเร็จลุล่วงไปด้วยดี ตลอดจนบุพการีผู้สนับสนุนการศึกษาตลอดมา

นอกจากนี้ คณะผู้จัดทำขอขอบคุณภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง ที่กรุณาให้งบประมาณสนับสนุนในการจัดทำโครงงานนี้ และขอขอบคุณบริษัท ถาวรคอมพิวเตอร์ จำกัด ที่เอื้อเฟื้อเอกสารที่ใช้อ้างอิงในปริญญานิพนธ์ฉบับนี้

สารบัญ

สารบัญตาราง

สารบัญรูปภาพ

บทที่ 1
บทนำ

1.1 ความเป็นมาและความสำคัญของปริญญานิพนธ์

โรงงานอุตสาหกรรมในปัจจุบัน ได้มีการนำเครื่องจักรมาใช้ในขบวนการผลิตที่ทำงานภายใต้การควบคุมด้วยระบบอัตโนมัติมาใช้มากขึ้น ซึ่งอุปกรณ์หลักของระบบควบคุมแบบอัตโนมัตินี้ ได้แก่ คอมพิวเตอร์ ด้วยเหตุผลที่ว่า การใช้ไมโครคอมพิวเตอร์หรือคอมพิวเตอร์ส่วนบุคคลเป็นตัวควบคุมการทำงานของวงจรอิเล็กทรอนิกส์นั้น จะมีความถูกต้องเที่ยงตรงเชื่อถือได้ มีการทำงานที่รวดเร็วแม่นยำ และยังสามารถแก้ไขเปลี่ยนแปลงทางซอร์ฟแวร์ได้ง่าย ทำให้สะดวกและคล่องตัวในการใช้งานมากขึ้น

จากประโยชน์ที่มีหลากหลายของคอมพิวเตอร์ และการนำเอาคอมพิวเตอร์มาใช้ในโรงงานอุตสาหกรรม เราจะนำเอาไมโครคอมพิวเตอร์ไปควบคุมอุปกรณ์ต่าง ๆ ทั้งอุปกรณ์ที่เป็นอินพุทและอุปกรณ์ที่เป็นเอาท์พุท โดยการใช้การ์ดเฉพาะงานในการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก ซึ่งเราเรียกวิธีนี้ว่า การอินเตอร์เฟสคอมพิวเตอร์กับอุปกรณ์ภายนอก ในส่วนที่เรานำมาทำการศึกษาและทดลอง จำเป็นที่จะต้องทราบถึงโครงสร้างทางด้านฮาร์ดแวร์ โครงสร้างของช่องนำสัญญาณหรือสล็อต (SLOT) และการอินเตอร์เฟสคอมพิวเตอร์กับอุปกรณ์ภายนอก

จากที่กล่าวมาแล้ว ในการที่จะนำไมโครคอมพิวเตอร์ไปประยุกต์ใช้งานในการควบคุมให้มีประสิทธิภาพและถูกต้องนั้น จึงจะต้องมีความเข้าใจเกี่ยวกับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก ตลอดจนคำสั่งหรือซอร์ฟแวร์ที่ใช้สำหรับการควบคุมการทำงานอย่างละเอียด เพื่อให้อุปกรณ์ภายนอกนั้นทำงานได้อย่างมีประสิทธิภาพ

คณะผู้จัดทำเห็นว่า ไมโครคอมพิวเตอร์หรือคอมพิวเตอร์ส่วนบุคคลสามารถนำมาใช้เป็นเครื่องมือในการเขียนโปรแกรมทางซอร์ฟแวร์เพื่อสั่งการควบคุมอุปกรณ์ภายนอกได้ทั้งอุปกรณ์ที่เป็นอินพุทและอุปกรณ์ที่เป็นเอาท์พุท โดยจะจัดเป็นรูปแบบของชุดฝึกการใช้งานไมโครคอมพิวเตอร์ในงานอุตสาหกรรม ชุดฝึกนี้มีการจัดทำเป็นใบงานการทดลองเพื่อให้ผู้ที่สนใจใช้ฝึกการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก ซึ่งจะทำให้เกิดความชำนาญและคุ้นเคยกับคอมพิวเตอร์มากยิ่งขึ้น การสั่งการทางด้านซอร์ฟแวร์หรือโปรแกรมจะใช้

โปรแกรมภาษาซีเป็นตัวสั่งให้อุปกรณ์ภายนอกทำงาน เมื่อทำขบวนการทางด้านซอร์ฟแวร์เสร็จสิ้นก็จะเข้าสู่หลักการของการเชื่อมต่อโดยผ่านทางช่องนำสัญญาณของคอมพิวเตอร์ผ่านการ์ดเฉพาะงานไปสู่ชุดฝึกการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอก(Interface Board) ภายในชุดฝึกได้รวมเอาวงจรที่เกี่ยวข้องกับการควบคุมแบบต่างๆไว้ โดยศึกษารายละเอียดได้จากปริญญานิพนธ์ฉบับนี้

1.2 ขอบเขตของปริญญานิพนธ์

1. ใช้โปรแกรมภาษาซีควบคุมการทำงานของชุดฝึกได้
2. ใช้เป็นชุดฝึกเพื่อการศึกษาในการอินเตอร์เฟสคอมพิวเตอร์กับอุปกรณ์ภายนอก
3. สามารถควบคุมอุปกรณ์ภายนอกอย่างน้อย 5 อย่างได้
4. มีใบงานประกอบชุดฝึก

1.3 เนื้อหาโดยสังเขป

ปริญญานิพนธ์นี้มีเนื้อหาทั้งหมด 5 บทดังต่อไปนี้

บทที่ 1 บทนำ จะเป็นการกล่าวถึงรายละเอียดในแต่ละบทของปริญญานิพนธ์ฉบับนี้โดยสังเขป

บทที่ 2 ทฤษฎีและหลักการ จะกล่าวถึงเนื้อหาที่นำมาใช้อ้างอิงและใช้เป็นแนวทางในการออกแบบชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมทั้งฮาร์ดแวร์และซอร์แวร์

บทที่ 3 การออกแบบและการสร้าง จะอธิบายถึงการออกแบบวงจรและการสร้างในแต่ละส่วนของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

บทที่ 4 การทดลองและผลการทดลอง ในบทนี้เป็นการนำโปรแกรมที่ได้ออกแบบไว้ในบทที่ 3 มาทำการทดสอบและบันทึกผลการทดลอง

บทที่ 5 บทสรุป วิจารณ์ และแนวทางในการพัฒนา ในบทนี้มุ่งเน้นถึงแนวทางในการแก้ปัญหาต่างๆ ที่เกิดขึ้นนับตั้งแต่การเริ่มศึกษาปริญญานิพนธ์จนกระทั่งเสร็จสมบูรณ์ พร้อมทั้งเสนอแนวทางการพัฒนา เพื่อให้ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมประสิทธิภาพสูงต่อไป

บทที่ 2

ทฤษฎีและหลักการ

เนื้อหาของปริญญานิพนธ์ในบทนี้เป็นทฤษฎีและหลักการที่นำมาใช้ประกอบการสร้างโครงงานโดยประกอบด้วย โครงสร้างทางฮาร์ดแวร์ของไมโครคอมพิวเตอร์,สัญญาณต่างๆ บนสล็อตของเครื่องไมโครคอมพิวเตอร์ ,การอ้างแอดเดรสของพอร์ท I/O และหลักการโปรแกรมภาษาซี ซึ่งจะกล่าวดังต่อไปนี้

2.1 โครงสร้างทางฮาร์ดแวร์ของไมโครคอมพิวเตอร์

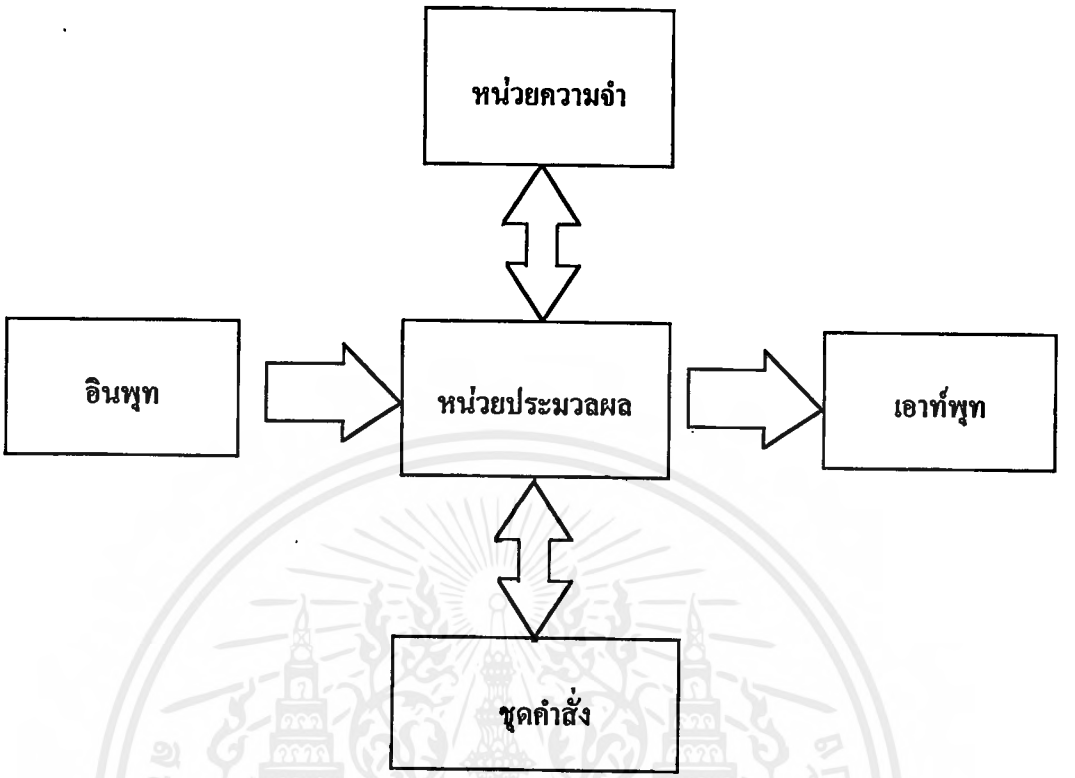
ฮาร์ดแวร์ของไมโครคอมพิวเตอร์ คือ องค์ประกอบของเครื่องคอมพิวเตอร์ที่สามารถจับต้องได้ เช่น แผงวงจร จอภาพ คีย์บอร์ด เป็นต้น แต่การที่จะใช้งานไมโครคอมพิวเตอร์นั้นจำเป็นที่จะต้องรู้ถึงการทำงานของระบบคอมพิวเตอร์เสียก่อน ในปริญญานิพนธ์ฉบับนี้จึงขอกล่าวถึงส่วนของไมโครคอมพิวเตอร์ ที่มีความเกี่ยวข้องกับโครงงานในปริญญานิพนธ์ซึ่งมีดังต่อไปนี้

2.1.1 ส่วนประกอบและการทำงานของระบบคอมพิวเตอร์

ส่วนประกอบของระบบคอมพิวเตอร์โดยทั่วๆไป สามารถที่แบ่งออกเป็นส่วนประกอบใหญ่ ๆ 4 ส่วนคือ

1. หน่วยความจำ (Memory)
2. หน่วยประมวลผล (Processor Unit)
3. หน่วยอินพุท / เอาท์พุท (Input - Output Unit)
4. ชุดคำสั่ง (Program)

ซึ่งสามารถแสดงส่วนประกอบของระบบคอมพิวเตอร์ได้ดังรูปที่ 2.1



รูปที่ 2.1 ส่วนประกอบของระบบคอมพิวเตอร์

1. หน่วยความจำ (Memory)

ทำหน้าที่เก็บข้อมูลทุกชนิดที่คอมพิวเตอร์รับเข้ามา ไม่ว่าจะรับเข้ามาโดยวิธีใดก็ตาม หน่วยความจำเปรียบเสมือนสมองของคอมพิวเตอร์เลยก็ว่าได้ เหมือนกับสมองของคนเรา ถ้าคนเราไม่มีสมองก็ไม่สามารถทำงานได้ เช่นเดียวกับคอมพิวเตอร์ ถ้าไม่มีหน่วยความจำก็ไม่สามารถทำงานใด ๆ ได้เช่นกัน

นอกจากนี้ หน่วยความจำที่มีไว้ในไมโครคอมพิวเตอร์ ยังสามารถแบ่งออกได้เป็น 2 ชนิด คือ

ROM (Random Access Memory) หรือหน่วยความจำแบบชั่วคราว ความจำชนิดนี้ไม่สามารถที่จะเก็บข้อมูลไว้ได้ตลอดเวลา หน่วยความจำชั่วคราวจะเก็บข้อมูลไว้เฉพาะเวลามีแหล่งจ่ายไฟให้เท่านั้นเองเมื่อเลิกใช้งานปิดเครื่องไมโครคอมพิวเตอร์ข้อมูลทั้งหมดที่ถูกเก็บไว้ในหน่วยความจำชั่วคราวก็จะหายไป

. ROM (Read Only Memory) หรือหน่วยความจำถาวร หน่วยความจำประเภทนี้จะสามารถเก็บรักษาข้อมูลไว้ตลอดเวลาแม้กระทั่งไม่มีไฟจ่ายให้ก็ตาม หน่วยความจำประเภทนี้ มักจะทำหน้าที่เกี่ยวกับการเก็บสภาวะแวดล้อมของระบบไมโครคอมพิวเตอร์ เช่น ไบออส (BIOS) เป็นต้น

2. หน่วยประมวลผล (Processor Unit)

หรือหน่วยประมวลผลกลาง (Central Processing Unit, CPU) มีหน้าที่นำข้อมูลที่เก็บอยู่ในหน่วยความจำมากระทำชุดคำสั่งที่เขียนหรือกำหนดขึ้น จากนั้นจะส่งผลลัพธ์แสดงออกที่หน่วยเอาท์พุทขึ้นอยู่กับว่าเอาท์พุทนั้นจะเป็นอะไร เช่น มอนิเตอร์(Monitor), ปริ้นเตอร์ (Printer) เป็นต้น

3. หน่วยอินพุท / เอาต์พุท (Input / Output)

ถ้าจะยกตัวอย่างให้มองเห็นอุปกรณ์ในหน่วยอินพุท / เอาท์พุทชัดเจนที่สุดงหนีไม่พ้นคีย์บอร์ดที่ทำหน้าที่เป็นอินพุทและจอแสดงภาพที่ทำหน้าที่เป็นอุปกรณ์เอาต์พุท อุปกรณ์อินพุทจะมีหน้าที่นำชุดคำสั่งและข้อมูล (Data) เข้าไปเก็บไว้ในหน่วยความจำ จากนั้นหน่วยประมวลผลจะนำชุดคำสั่งไปประมวลผล เมื่อเรียบร้อยแล้วจะส่งผลนั้นออกที่อุปกรณ์เอาท์พุท ในที่นี้ก็คือจอแสดงผลนั่นเอง แต่อุปกรณ์อินพุท / เอาท์พุทไม่ได้มีเพียงแค่คีย์บอร์ดและจอแสดงผลเท่านั้น ยังมีอีกมากมายหลายชนิด ซึ่งจะได้กล่าวต่อไปในหัวข้อของส่วนประกอบของเครื่องไมโครคอมพิวเตอร์

4. ชุดคำสั่ง (Program)

คอมพิวเตอร์จะสามารถทำงานได้ก็ต่อเมื่อมีชุดคำสั่ง หรือโปรแกรมคอยควบคุมการทำงานของมันเท่านั้น และชุดคำสั่งที่จะนำมาควบคุมการทำงานของคอมพิวเตอร์นั้นก็ได้มาจากผู้เขียนโปรแกรม โดยผู้เขียนจะเขียนโปรแกรมเพื่อสั่งการให้คอมพิวเตอร์ทำงานตามความต้องการหรือตามรูปแบบที่ได้วางไว้ เช่น ในปริญญานิพนธ์นี้ได้ใช้โปรแกรมภาษาซีซึ่งเป็นภาษาคอมพิวเตอร์ระดับสูง ในการเขียนเป็นโปรแกรมควบคุมการทำงานของอุปกรณ์ควบคุมภายนอก

2.1.2 ส่วนประกอบของเครื่องไมโครคอมพิวเตอร์

ในปริญญานิพนธ์ฉบับนี้ส่วนประกอบของไมโครคอมพิวเตอร์และองค์ประกอบพื้นฐานต่างๆ จะใช้ IBM PC เป็นตัวอ้างอิงเพื่อให้สะดวกและเกิดความเข้าใจไปในแนวทาง

เดียวกัน องค์ประกอบพื้นฐานของ IBM PC ประกอบด้วยองค์ประกอบต่อไปนี้ คือ System Unit , คีย์บอร์ด , จอแสดงผล ซึ่งจะอธิบายได้ดังต่อไปนี้

1. ซีสเต็มยูนิต (System Unit)

ส่วนของซีสเต็มยูนิตประกอบด้วยไมโครโปรเซสเซอร์ขนาด 16 บิท หน่วยความจำที่ประกอบด้วย ROM (Read Only Memory) และ (Random Access Memory) , แหล่งจ่ายไฟ, ลำโพงซึ่งใช้ส่งสัญญาณเสียงหรือดนตรีและช่องนำสัญญาณหรือสล็อต 5 สล็อต เราใช้สล็อตทั้ง 5 นี้ในการขยายระบบ

2. โปรเซสเซอร์บอร์ด (Processor Board)

ส่วนสำคัญของซีสเต็มยูนิต คือ โปรเซสเซอร์บอร์ด ซึ่งโปรเซสเซอร์บอร์ด จะเป็นส่วนที่บรรจุอุปกรณ์อิเล็กทรอนิกส์พื้นฐานที่ใช้ในระบบคือ ไมโครโปรเซสเซอร์ 8088, ส่วนของหน่วยความจำซึ่งประกอบด้วย RAM ขนาด 640 กิโลไบท์และ ROM ขนาด 40 กิโลไบท์ และ I / O Adapter ที่ใช้กับคีย์บอร์ด (Adapter คือ อุปกรณ์ที่เป็นตัวช่วยขยายความสามารถในการทำงานของระบบบนโปรเซสเซอร์บอร์ด)นอกจากนี้เครื่อง IBM PC ยังเตรียมสล็อตเพื่อใช้ในการขยายระบบให้อีก 5 สล็อต ซึ่งสามารถเพิ่มความสามารถของระบบได้โดยการต่ออุปกรณ์ที่ใช้ในการอินเตอร์เฟส หรือการ์ดต่าง ๆ เข้าทางสล็อตเหล่านี้

3. แหล่งจ่ายไฟ (Power Supply)

แหล่งจ่ายไฟที่ใช้ในการจ่ายพลังงานให้แก่อุปกรณ์ต่างๆ ในซีสเต็มยูนิตและยังจ่ายพลังงานไปที่สล็อตต่าง ๆ ทั้ง 5 สล็อตอีกด้วย ดังนั้นการ์ดต่างๆ ที่นำมาเสียบกับสล็อตจะได้รับไฟจากแหล่งจ่ายไฟด้วย สำหรับแรงดันไฟตรงที่แหล่งจ่ายไฟชุดนี้สามารถจ่ายพลังงานได้ทั้งหมด 200 วัตต์

4. ดิสก์เก็ตไดรฟ์ (Diskette Drive)

ในการอินเตอร์เฟสดิสก์เก็ตไดรฟ์เข้ากับโปรเซสเซอร์บอร์ดนั้นจะต้องใช้ Diskette Adapter ช่วยในการอินเตอร์เฟสโดยเสียบ Diskette Adapter ลงในสล็อตใดสล็อตหนึ่งจากสล็อตทั้งหมด 5 สล็อต

5. ลำโพง (Speaker)

ปกติมักถูกใช้แสดงสัญญาณเตือนต่างๆใช้ในการเล่นเกมส์หรือใช้โปรแกรมที่ต้องการเอาท์พุทเป็นสัญญาณเสียง

6. ระบบคีย์บอร์ด (Keyboard Unit)

คีย์บอร์ดทั้งหมดของ IBM PC มีทั้งหมด 101 คีย์ โดยส่วนกลางของคีย์บอร์ดจะมีลักษณะเหมือนกับแป้นพิมพ์ของเครื่องพิมพ์ดีด สำหรับส่วนคีย์ด้านข้างของคีย์บอร์ดนั้นจะมีลักษณะพิเศษต่างจากคีย์บอร์ดของเครื่องพิมพ์ดีด ทางด้านขวาจะเป็นคีย์ที่ใช้ อ่านค่าตัวเลขและใช้ควบคุมการเลื่อนตำแหน่งของเคอร์เซอร์พร้อมกันไปด้วย สำหรับคีย์ทางด้านซ้ายจะเป็นฟังก์ชั่นคีย์ซึ่งสามารถโปรแกรมฟังก์ชั่นการทำงานให้แก่ฟังก์ชั่นคีย์แต่ละคีย์ได้ การอินเตอร์เฟสคีย์บอร์ดเข้ากับซีสเต็มยูนิตจะทำผ่านทางสายไฟจำนวนสี่เส้นซึ่งใช้ส่งสัญญาณและจ่ายไฟให้แก่คีย์บอร์ด คีย์บอร์ดของเครื่อง IBM PC จะส่งรหัสสัญญาณขนาดแปดบิทจำนวนหนึ่งรหัสทุกครั้งที่คีย์ถูกกด และทุกครั้งที่ปล่อยคีย์ ถ้ากดคีย์ใดค้างไว้ คีย์บอร์ดจะส่งรหัสเมื่อคีย์ถูกกดจำนวนหนึ่งรหัสแล้วหยุดชั่วขณะแล้วส่งรหัสตัวเดิมซ้ำออกไปอีกตามจังหวะของการพิมพ์จนกว่าคีย์ตัวนั้นจะถูกปล่อย ขณะที่ปล่อยคีย์ คีย์บอร์ดจะส่งรหัสอีกตัวหนึ่งที่ต่างจากกันออกไป ลักษณะการทำงานแบบนี้ช่วยให้ระบบทราบถึงการทำงานของคีย์บอร์ดสามารถระบุได้ว่าขณะใดที่ใช้ SHIFT Key อยู่รวมทั้งการกดคีย์ค้าง

7. จอแสดงผล (Monitor Display)

จอแสดงผลนี้ต่อเข้ากับเครื่อง PC ผ่านทางการ์ด ซึ่งถูกเสียบลงในสล็อตหนึ่งจากจำนวนทั้งหมด 5 สล็อตที่อยู่บนซีสเต็มยูนิต จอภาพนี้ได้รับไฟกระแสสลับ (AC) จากซีสเต็มยูนิต โดยผ่านทางสายเคเบิลสั้น ๆ ที่ต่ออยู่ข้างหลังเครื่อง ดังนั้นการเปิดปิดสวิทช์ที่ซีสเต็มยูนิต จะควบคุมการจ่ายไฟให้แก่จอภาพด้วย สัญญาณที่ใช้ในการแสดงผลนั้นเป็นแบบ Direct - drive ซึ่งประกอบด้วยสัญญาณต่อไปนี้คือ สัญญาณที่ใช้ควบคุมการสแกนทางแนวนอน , สัญญาณที่ใช้ควบคุมสแกนทางแนวตั้ง , สัญญาณหรือข้อมูลและสัญญาณควบคุมความเข้ม, สำหรับความละเอียดของมอนิเตอร์จะมีรายละเอียด 720 จุดทางแนวนอน และ 350 จุดทางแนวตั้ง จอภาพจะถูกรีเฟรชด้วยความถี่ 50 Hz ความถี่ที่ใช้สแกนภาพมีค่าประมาณ 17 KHz ข้อมูลจะถูกส่งออกด้วยความถี่ 16.257 MHz จอของเครื่อง IBM สามารถแสดงตัวอักษรได้ 80 ตัวต่อบรรทัดได้ 25 บรรทัด

8. การ์ดทำหน้าที่เฉพาะงาน

การ์ดเฉพาะงานต่างๆ ที่ใช้ในการขยายหน่วยความจำ , การอินเตอร์เฟสกับอุปกรณ์ภายนอกอื่นๆ หรือใช้ขยายความสามารถของระบบตามความต้องการของผู้ใช้นั้นสามารถเชื่อมต่อเข้ากับซีสเต็มบอร์ดโดยผ่านทางสล็อตทั้ง 5 สล็อต

Diskette Drive Attachment Adapter Card เราสามารถใช้การ์ดเฉพาะนี้ต่อดิสเก็ตไดรฟ์ขนาด 5 ¼ นิ้ว จำนวน 2 ตัว (ซึ่งปกติดิสเก็ตไดรฟ์ทั้งสองตัวนี้จะอยู่ในซีสเต็มยูนิต) เข้ากับโปรเซสเซอร์บอร์ดได้นอกจากนี้เรายังสามารถเพิ่มจำนวนดิสเก็ตไดรฟ์เข้าไปได้อีก 2 ตัว การใช้งานของการ์ดแผ่นนี้กินเนื้อที่ของสล็อตเพียงสล็อตเดียวเท่านั้น

Parallel Printer Port Attachment Card เราใช้การ์ดนี้ในการอินเตอร์เฟสเครื่องพิมพ์ที่มีขนาดมาตราฐานของการอินเตอร์เฟสเป็นแบบขนาดตามมาตราฐาน Centronix การ์ดนี้สามารถส่งข้อมูลตามคำสั่งที่โปรแกรมไว้หรือโดยการอินเตอร์รัพท์ การ์ดนี้กินเนื้อที่เพียงสล็อตเดียว

Serial Port Attachment Card พอร์ทบนการ์ดแผ่นนี้เป็นพอร์ทอนุกรมแบบอะซิงโครนัส ซึ่งเป็นไปตามมาตราฐานการอินเตอร์เฟสแบบ RS-232 - C ซึ่งเราสามารถนำไปเชื่อมต่อกับโมเด็มเพื่อใช้ในการส่งข้อมูลผ่านทางสายโทรศัพท์ได้ เราสามารถโปรแกรมอัตราเร็วในการส่งข้อมูล (band rate) ของการ์ดนี้ให้มีอัตราเร็วระหว่าง 50 ถึง 9600 band เราสามารถโปรแกรมขนาดของข้อมูลที่จะส่งว่าต้องการให้มีขนาด 5, 6, 7 หรือ 8 บิทได้เช่นเดียวกัน จำนวน stop bit ก็สามารถโปรแกรมได้ โดยมีขนาด 1, 1 1/2 หรือ 2 บิท สำหรับบิทพาริตี้ เราสามารถเลือกได้ว่าต้องการให้เป็นแบบพาริตี้คู่ , คี่ หรือไม่มีการสร้างบิทพาริตี้ นอกจากนี้เราสามารถส่งข้อมูลโดยอาศัยการอินเตอร์เฟสแบบ Current - loop ได้โดยการปรับตัวจัมเปอร์(jumper) เช่นเดียวกับการ์ดอื่น ๆ ที่กล่าวมาการ์ดแผ่นนี้ใช้สล็อตเพียงสล็อตเดียวบนซีสเต็มยูนิต

Prototyping Card ในบางครั้งเราต้องการอินเตอร์เฟสเครื่อง IBM PC เข้ากับอุปกรณ์หรือเครื่องจักรบางชนิด แต่เนื่องจากเป็นการใช้เฉพาะงาน เราจึงต้องออกแบบวงจรส่วนอินเตอร์เฟสเองโดยการใช้ Prototyping Card นี้ เราสามารถนำวงจรของเราประกอบลงการ์ดแผ่นนี้ ลักษณะของการ์ดแผ่นนี้จะมีลักษณะเป็นรูปสี่เหลี่ยมผืนผ้าเจาะรู

ไว้เป็นแถว ๆ เพื่อใช้เสียบอุปกรณ์และบัดกรีสาย Wire Wrap นอกจากนี้ยังมีการเดินลายทองแดงบนการ์ด ซึ่งเมื่อการ์ดถูกเสียบลงในสล็อตมันจะถูกต่อเข้ากับบัสของระบบ (System Bus) ลักษณะของการเดินสายปริ้นจะช่วยให้เราสมารถบัฟเฟอร์ข้อมูลได้จากบัสของระบบ และช่วยให้การถอดรหัสแอดเดรสทำได้ง่ายโดยใช้อุปกรณ์เพียงไม่กี่ตัว

2.2 สัญญาณต่าง ๆ บนสล็อตของเครื่องไมโครคอมพิวเตอร์

ภายในเครื่องไมโครคอมพิวเตอร์ ได้ออกแบบให้สามารถที่จะที่จะเพิ่มเติมวงจรอินเตอร์เฟสเข้าไปในภายหลังได้ โดยผ่านทางสล็อตที่อยู่บนเมนบอร์ดนี้จะมีจำนวน 5 สล็อต ซึ่งแต่ละสล็อตจะมีจำนวนขาทั้งสิ้น 62 ขา แบ่งออกเป็น 2 ข้าง ๆ ละ 31 ขา ส่วนการเรียกตำแหน่งขาของสล็อตเหล่านี้จะขึ้นอยู่กับว่าขานั้นอยู่ข้างใด(ซ้ายหรือขวา) ของสล็อตโดยขาที่อยู่ทางด้านซ้ายของสล็อตจะเรียกโดยใช้อักษร “B” นำหน้าเลขตำแหน่งของขา เช่น ขา B16 (นับจากทางด้านท้ายของเครื่อง) ส่วนขาที่อยู่ทางด้านขวาของสล็อตจะเรียกโดยใช้อักษร “A” นำหน้าเลขตำแหน่งของขา เช่น ขา A24 ก็คือขาทางด้านขวาของสล็อตขาที่ 24 (นับจากทางด้านท้ายของเครื่อง)

แต่ละขาของสล็อตเหล่านี้จะเชื่อมต่อกับเส้นสัญญาณต่างๆ บนเมนบอร์ด ทำให้การสร้างวงจรอินเตอร์เฟสกับ IBM PC สามารถทำได้โดยสะดวก ซึ่งเส้นสัญญาณที่เชื่อมต่อกับขาของสล็อตเหล่านี้จะประกอบไปด้วย เส้นสัญญาณของบัสแอดเดรส (Address Bus), บัสข้อมูล (Data Bus) , บัสควบคุมสำหรับการเขียน / อ่านข้อมูลจากหน่วยความจำ หรือพอร์ท I / O , เส้นสัญญาณสำหรับการขอการอินเตอร์รัพท์ของวงจรอินเตอร์เฟส, เส้นสัญญาณสำหรับการขอ DMA, สัญญาณฐานเวลา(Timing Signal) ต่าง ๆ ที่ใช้ในระบบ , เส้นสัญญาณแสดงการรีเฟรชหน่วยความจำ และเส้นสัญญาณสำหรับการตรวจสอบความผิดพลาด (I / O CHCK)

นอกจากเส้นสัญญาณเหล่านี้แล้ว สล็อตเมนบอร์ดยังเชื่อมต่อกับแหล่งจ่ายไฟต่างๆ ที่ใช้ในระบบอีกด้วย คือ +5 VDC, -5 VDC , +12 VDC และ -12 VDC

รายละเอียดเกี่ยวกับสัญญาณต่าง ๆแสดงดังรูปที่ 2.2 มีดังต่อไปนี้

2. 2.1 Address bus และสัญญาณต่าง ๆ ที่เกี่ยวข้อง

A0 - A19 (Address Bus : ขา A31 - A12) :

ขาสัญญาณทั้ง 12 ขานี้ เป็นเอาท์พุทซึ่งใช้สำหรับกำหนดแอดเดรสของหน่วยความจำหรืออุปกรณ์ I/O ที่ 8088 ต้องการติดต่อด้วย โดยที่สัญญาณ A0 จะมีนัยสำคัญต่ำสุด (Least Signifioant Bit) และ A19 จะมีนัยความสำคัญสูงสุด (Most Significant Bit) สำหรับค่าแอดเดรสบัส A0-A19 นี้จะถูกกำหนดโดย 8088 ในระหว่างขบวนการอ่าน / เขียนข้อมูลลงในหน่วยความจำหรืออุปกรณ์ I/O แต่ในช่วงของขบวนการ DMA นั้น DMA-Controller จะเป็นผู้กำหนดค่าแอดเดรสบนบัสแอดเดรสเอง (ในระหว่างนี้ 8088 จะถูกตัดออกจากระบบ)

จะเห็นได้ว่าจำนวนแอดเดรสมีอยู่ 20 เส้น ซึ่งสามารถที่จะอ้างแอดเดรสของหน่วยความจำได้ถึง 1 เมกะไบท์ แต่อย่างไรก็ตามจะมีแอดเดรสบางแอดเดรสที่ถูกใช้งานโดย IBM PC อยู่ก่อนแล้ว คือแอดเดรสของหน่วยความจำ RAM บนเมนบอร์ดที่ถูกใช้โดยระบบจำนวน 64 กิโลไบท์ และแอดเดรสสำหรับหน่วยความจำ ROM อีก 48 กิโลไบท์ ซึ่งถูกจัดในช่วงของแอดเดรสบนสุดใน 1 เมกะไบต์ คือ 0FC00H จนถึง 0FFFFH

สำหรับการอ้างแอดเดรสของพอร์ท I/O นั้น จะใช้เส้นแอดเดรสเพียง 16 เส้น คือ A0-A15 ซึ่งจะทำให้อ้างแอดเดรสของพอร์ทได้ 64 กิโลพอร์ท โดยผ่านทางชุดคำสั่ง IN และ OUT ส่วนเส้นแอดเดรสที่เหลือคือ A16 -A19 นั้นจะไม่ถูกใช้งานอย่างไรก็ตามภายใน IBM PC จะใช้เส้นแอดเดรสในการอ้างแอดเดรสของพอร์ทเพียง 10 เส้นคือ จาก A0- A9 และค่าแอดเดรสที่ใช้งานสำหรับวงจรเพิ่มเติมภายนอกต้องอยู่ในช่วง 0200H จนถึง 03FFH เท่านั้น

D0-D7 (Data Bus : ขา A9 - A2) :

ขาสัญญาณนี้จะเป็นแบบ Bi-Directional ซึ่งต่อกับบัสข้อมูลของระบบเพื่อทำหน้าที่ในการส่งผ่านข้อมูลระหว่างพอร์ท I/O กับ IBM PC โดยบิท D0 จะมีนัยสำคัญต่ำสุดและบิท D7 จะมีนัยสำคัญสูงสุด

สำหรับในบัสไซเคิลของการเขียนข้อมูลที่สร้างขึ้นโดย 8088 นั้น ข้อมูลจะถูกส่งออกมาบนบัสข้อมูล ก่อนสัญญาณ IOW (ในกรณีที่ต้องการส่งข้อมูลให้กับพอร์ท) หรือ MEMW (ในกรณีที่ต้องการส่งข้อมูลให้กับหน่วยความจำ) จะเปลี่ยนจากลอจิก " 0 " เป็น

ลอจิก " 1 " (ขอบขาขึ้น) ซึ่งโดยทั่วไปขอบขาขึ้นของสัญญาณ IOW หรือ MEMW นี้ จะถูกใช้เพื่อสั่งให้พอร์ท I/O หรือหน่วยความจำที่มีแอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้นรับข้อมูลไปเก็บไว้

สำหรับบัสไซเกิลของการอ่านข้อมูลที่สร้างขึ้นโดย 8088 นั้น พอร์ท I/O หรือหน่วยความจำที่ถูกอ้างถึงจะต้องส่งข้อมูลออกมาบนบัสข้อมูล ก่อนที่สัญญาณ IOR (ในกรณีที่ต้องการอ่านข้อมูลจากพอร์ท) หรือMEMW (ในกรณีที่ต้องการอ่านข้อมูลจากหน่วยความจำ) จะเปลี่ยนจากลอจิก " 0 " เป็นลอจิก " 1 " (ขอบขาขึ้น)

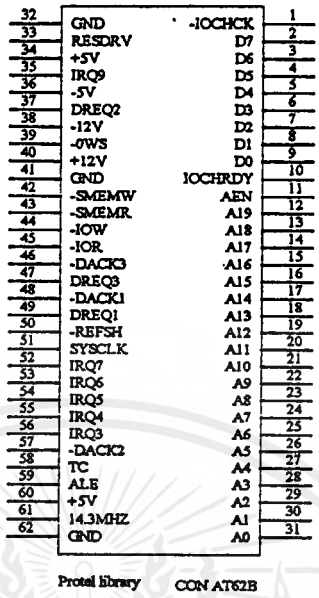
AEN (address Enable : ขา A11):

สัญญาณนี้เป็นสัญญาณเอาท์พุทที่ใช้ในการแสดงว่าบัสไซเกิลที่เกิดขึ้น ในช่วงเวลาที่มีสัญญาณ AEN แอคตีฟลอจิก " 1 " นั้น เป็นบัสไซเคิลของขบวนการ DMA

สำหรับบนเมนบอร์ดของ IBM PC นั้น จะใช้สัญญาณในการ Disable 8288 Bus Controller และจะใช้ Disable พอร์ท I/O ต่าง ๆ ที่ไม่เกี่ยวข้องกับขบวนการ DMA นั้น 8237A-5 จะส่งแอดเดรสของหน่วยความจำออกมาบนบัสแอดเดรส (ซึ่งเป็นแอดเดรสของหน่วยความจำ)นั้นทำการอ่านหรือส่งข้อมูลออกมาบนบัสข้อมูลทำให้เกิดความผิดพลาดขึ้นได้

ALE (Address Latch Enable : ขา B28) :

สัญญาณนี้เป็นสัญญาณเอาท์พุทที่ 8288 Bus Controller สร้างขึ้นเพื่อใช้สำหรับแสดงการเริ่มต้นของบัสไซเกิล และแสดงให้อุปกรณ์ภายนอกทราบว่าแอดเดรสที่ 8088 ต้องการจะติดต่อด้วยนั้นถูกส่งออกมาบนบัสแอดเดรสแล้ว โดยที่สัญญาณ ALE นี้จะเปลี่ยนจากลอจิก " 1 " เป็น " 0 "เมื่อค่าแอดเดรสที่ถูกต้องถูกส่งออกมาบนบัสข้อมูลเรียบร้อยแล้ว ดังนั้นขอบขาลงสัญญาณ ALE นี้ถูกใช้ในการแลทช์ค่าแอดเดรสจากบัสแอดเดรส/ ข้อมูล (Address / Data Bus AD0-AD7) ของ 8088 ทำให้สามารถแยกค่าแอดเดรส (A0-A19) และข้อมูล(D0-D7) ออกจากกันได้ อย่างไรก็ตามสัญญาณ ALE จะแอคตีฟเฉพาะในบัสไซเกิลที่สร้างขึ้นโดย 8088 เท่านั้นโดยจะไม่แอคตีฟในระหว่างขบวนการ DMA



รูปที่ 2.2 สัญญาณต่าง ๆ บนสล็อต

2.2.2 สัญญาณอินเตอร์รัพท์

$\overline{\text{I/O CHCK}}$ (I/O Channal Check : ขา A1) :

ขาสัญญาณนี้เป็นอินพุทที่ใช้ในการแสดงความผิดพลาดเกี่ยวกับพาริตี้ ที่เกิดขึ้นในการทำงานของวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O เมื่อขาสัญญาณนี้ได้รับลอจิก " 0 " จะทำให้ 8088 ถูกอินเตอร์รัพท์แบบ Non-Maskable (NMI) อย่างไรก็ตามเราสามารถที่จะกำหนดให้วงจรภายในของ IBM PC ทำการขออินเตอร์รัพท์ (เมื่อได้รับสัญญาณ $\overline{\text{I/O CHCK}}$) หรือไม่ก็ได้โดยการกำหนดลอจิกของบิทข้อมูลของพอร์ทที่ควบคุมการขออินเตอร์รัพท์แบบ NMI คือ บิท D7 ของพอร์ท 00A0H ในกรณี D7 ของพอร์ท 00A0H ถูกเซตเป็น " 1 " ก็จะทำให้วงจรภายนอกขออินเตอร์รัพท์แบบ NMI ได้ (Enable) แต่ถ้าบิท D7 ของพอร์ท 00A0H ถูกเซตเป็น " 0 " ก็จะเป็นการดิสเอเบิล (Disable) การขออินเตอร์รัพท์แบบ NMI

IRQ2-IRQ7 (Interrupt Request 2 Through 7 : ขา B4 และ B25-B21) :

ขาสัญญาณทั้ง 6 นี้เป็นขาอินพุทที่ใช้สำหรับการขออินเตอร์รัพท์จาก 8088 โดยสัญญาณเหล่านี้จะต่อเข้ากับ 8259A บนเมนบอร์ดโดยตรง โปรแกรมในส่วนไบออส (BIOS) ของ IBM PC จะทำการโปรแกรม 8259A ให้ IRQ2 มีลำดับความสำคัญสูงสุด (Highest, Priority) และ IRQ7 มีลำดับความสำคัญต่ำสุด ในกรณีที่มีการขออินเตอร์รัพท์เกิดขึ้น คือ

ระดับลอจิกที่ขา IRQ ขาใดขาหนึ่งถูกเปลี่ยนจากลอจิก " 0 " เป็นลอจิก " 1 " (ขอบขาขึ้น) 8259A ก็จะทำการส่งสัญญาณ INT ให้กับ 8088 เพื่อทำการขออินเตอร์รัพท์

2.2.3 สัญญาณที่ใช้สร้าง Wait States

I/O CHRDY (I/O Channel Ready : ขา A10) :

ขาสัญญาณนี้เป็นอินพุทที่ใช้เพิ่มช่วงเวลาในบัสไซเคิลในกรณีที่อุปกรณ์ I/O หรือหน่วยความจำที่เกี่ยวข้องกับขบวนการในบัสไซเกิลที่เกิดขึ้นนั้น ไม่สามารถทำงานทันตามช่วงเวลาปกติของบัสไซเคิลนั้น ๆ ได้

เมื่ออุปกรณ์ I/O หรือหน่วยความจำต้องการที่จะเพิ่มช่วงเวลาในบัสไซเกิลให้นานขึ้นอีกนั้นจะสามารถทำได้โดยการป้อนลอจิก " 0 " ให้กับขา I/O CHRDY ในช่วงเวลาที่ I/O หรือหน่วยความจำที่ถูกกำหนดขึ้นได้รับสัญญาณจากการดีโค้ดแอดเดรสและสัญญาณ $\overline{\text{MEMR}}$, $\overline{\text{MEMW}}$, $\overline{\text{IOR}}$ หรือ $\overline{\text{IOW}}$ แอคตีฟ

2.2.4 สัญญาณนาฬิกา

CLK (Clock : ขา B20) :

ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งต่อกับสัญญาณคล็อกที่ถูกสร้างขึ้นโดยการหารสัญญาณ OSC ด้วย 3 ทำให้ได้ความถี่ประมาณ 4.77 MHz (14.31818 MHz /3) หรือมีช่วงเวลาใน 1 คาบ (ช่วงเวลาของคล็อก 1 ลูก) เท่ากับ 210 nanosec. (1/4.77 MHz) สำหรับค่า Duty Cycle ของสัญญาณนี้จะมีค่าประมาณ 1/3 คือ 1 คาบจะมีช่วงเวลาที่เป็นลอจิก " 1 " เท่ากับ 1/3 ของคาบเวลาทั้งหมด หรือประมาณ 70 nanosec. และช่วงเวลาที่เป็นลอจิก " 0 " เท่ากับ 2/3 ของคาบเวลาทั้งหมด หรือประมาณ 140 nanosec. สัญญาณนี้เป็นสัญญาณที่ถูกใช้เป็นคล็อกของระบบ

OSC (Oscilator : ขา B30) :

ขานี้เป็นเอาท์พุทที่เชื่อมต่อกับสัญญาณคล็อก ที่มีค่าความถี่สูงสุดบนเมนบอร์ดคือ 14.31816 MHz ซึ่งมีคาบเวลาประมาณ 70 nanosec. และมี Duty Cycle (ช่วงเวลาใน 1 คาบที่สัญญาณคล็อกมีลอจิกเป็น "1" หารด้วยคาบเวลาทั้งหมด) ประมาณ 50 สัญญาณคล็อกอื่น ๆ ของระบบ เช่น คล็อกที่ป้อนให้กับ 8088 หรือ ชิฟซัพพอร์ทต่าง ๆ นั้นจะถูกสร้างขึ้นโดยการหารสัญญาณคล็อกนี้ อย่างไรก็ตามสิ่งหนึ่งที่จะต้องคำนึงถึงในการใช้งานสัญญาณ OSC ก็คือสัญญาณนี้จะไม่ซิงโครนัส กับสัญญาณอื่นๆ บนบัสของระบบ ดังนั้นจึงไม่ควรที่จะนำ

สัญญาณจากขา OSC นี้ไปใช้เป็นสัญญาณคล็อกสำหรับวงจรภายนอกอื่น ๆ ที่ทำงานร่วมกับระบบ

2. 2. 5 สัญญาณควบคุมต่าง ๆ

$\overline{\text{MEMR}}$ (Memory Read : ขา B12) :

ขานี้เป็นเอาท์พุทจาก 8088 ซึ่งสัญญาณนี้จะแอคตีฟที่ลอจิก " 0 " ในระหว่างบัสไซเกิลของการอ่านข้อมูลจากหน่วยความจำของ 8088 เพื่อให้หน่วยความจำที่มีแอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้น ทำการส่งข้อมูลออกมาบนบัสข้อมูล โดยหน่วยความจำนั้นจะต้องส่งข้อมูลออกมาในช่วงเวลา 30 nanosec. ก่อนที่สัญญาณ $\overline{\text{MEMW}}$ จะกลับเป็นลอจิก " 1 " ทั้งนี้ก็เพื่อให้ 8088 ได้รับข้อมูลที่ถูกต้อง

สำหรับในระหว่างขบวนการ DMA นั้น DMA-Controller จะควบคุมบัสต่างๆ ของระบบแทน 8088 และสัญญาณ $\overline{\text{MEMW}}$ จะถูกใช้บัสไซเกิลของการอ่านข้อมูลจากหน่วยความจำ (ข้อมูลถูกส่งจากหน่วยความจำไปให้กับอุปกรณ์ I/O)

$\overline{\text{MEMW}}$ (Memory Write : ขา B11) :

ขานี้เป็นเอาท์พุทแอคตีฟที่ลอจิก " 0 " ซึ่ง 8288 BUS Controller สร้างขึ้นในระหว่างบัสไซเคิลในการเขียนข้อมูลลงในหน่วยความจำของ 8088 สัญญาณ $\overline{\text{MEMW}}$ นี้จะถูกส่งออกมาเพื่อให้หน่วยความจำที่แอดเดรสตรงกับค่าแอดเดรสบนบัสแอดเดรสนั้น ทำการรับข้อมูลที่อยู่บนบัสข้อมูลเก็บไว้ โดยทั่วไปหน่วยความจำจะรับข้อมูลในช่วงขอบขาขึ้นของสัญญาณ $\overline{\text{MEMW}}$

สำหรับในขบวนการ DMA นั้น 82837A -5 DMA - Controller จะทำการควบคุมบัสต่างๆ ของระบบแทน 8088 และสัญญาณ $\overline{\text{MEMW}}$ จะถูกใช้ในบัสไซเกิลของการเขียนข้อมูลลงในหน่วยความจำ (ข้อมูลถูกส่งจากอุปกรณ์ I/O ไปให้กับหน่วยความจำ)

$\overline{\text{IOR}}$ (I/O Read : ขา B14) :

ขาสัญญาณนี้เป็นเอาท์พุทแอคตีฟที่ลอจิก " 0 " ที่สร้างขึ้นโดย 8288 Bus Controller เพื่อใช้ในการแสดงว่าบัสไซเกิลที่เกิดขึ้นนี้เป็นบัสไซเกิลของการอ่านข้อมูลจากพอร์ท I/O เพื่อให้พอร์ท I/O ที่มีแอดเดรสตรงกับแอดเดรสบนบัสแอดเดรสนั้นส่งข้อมูลออกมาบนบัสข้อมูลโดยข้อมูลจะต้องถูกส่งออกมาบนบัสข้อมูลก่อนขอบขาขึ้นของสัญญาณ $\overline{\text{IOR}}$ ประมาณ 30 nanosec. เพื่อให้มั่นใจได้ว่า 8088 สามารถรับข้อมูลได้ถูกต้อง สำหรับในขบวนการ DMA

8237A- 5 DMA Controller จะทำการสร้างสัญญาณ $\overline{\text{IOR}}$ เอง โดยที่ค่าแอดเดรสที่อยู่บนบัสแอดเดรสจะเป็นค่าแอดเดรสของหน่วยความจำที่พอร์ท I/O ที่ขอ DMA ต้องการจะนำข้อมูลไปเก็บ การที่พอร์ทใดจะส่งข้อมูลออกมาบนบัสข้อมูลนั้น จะอาศัยสัญญาณ DACK จาก DMA Controller เป็นตัวกำหนด เช่นกรณีที่สัญญาณ DACK1 แอคตีฟก็จะแสดงว่าพอร์ท I/O จะต้องส่งข้อมูลออกมาบนบัสข้อมูลก็คือพอร์ท I/O ที่ขอ DMA ผ่านทาง DRQ1 เป็นต้น

$\overline{\text{IOW}}$ (I/O Writer : ขา B13) :

ขาสัญญาณเป็นเอา?พุทแอคตีฟที่ลอจิก "0" ซึ่งถูกสร้างขึ้นโดย 8288 Bus Controller เพื่อใช้แสดงว่าบัสไซเคิลที่เกิดขึ้นนี้เป็นบัสไซเคิลของการเขียนข้อมูลลงพอร์ท I/O เพื่อให้พอร์ท I/O ที่มีแอดเดรสตรงกับแอดเดรสบัสแอดเดรสนั้นรับข้อมูลที่อยู่บนบัสข้อมูลไผเก็บไว้ อย่างไรก็ตามเนื่องจากในชวงเวลาที่สัญญาณ $\overline{\text{IOW}}$ นี้ แอคตีฟ (ลอจิก "0") นั้นข้อมูลบนบัสข้อมูลอาจจะยังไม่สมบูรณ์ ดังนั้นการออกแบบจึงควรใช้ขอบขาขึ้นของสัญญาณ $\overline{\text{IOW}}$ แทนขอบขาลงในการทำให้พอร์ท I/O ที่เกี่ยวข้องรับข้อมูลไปเก็บไว้เพื่อให้ข้อมูลบนบัสข้อมูลสมบูรณ์เสียก่อน สำหรับในขบวนการ DMA Controller ทำการสร้างสัญญาณ $\overline{\text{IOW}}$ เองโดยที่ค่าแอดเดรสที่อยู่บนบัสแอดเดรสจะเป็นค่าแอดเดรสของหน่วยความจำที่พอร์ท I/O ที่ขอ DMA ต้องการจะอ่านข้อมูล

RESET DRV (ขา B2) :

ขาสัญญาณนี้เป็นเอาท์พุท ซึ่งจะแอคตีฟ (ลอจิก " 1 ") ในช่วงที่เราเริ่มจ่ายไฟให้กับระบบ และยังจะคงแอคตีฟไปจนกว่าระบบต่าง ๆ ภายใน IBM PC พร้อมที่จะทำงานได้จากนั้นสัญญาณนี้ก็จะเปลี่ยนกลับเป็นลอจิก " 0 " นอกจากนี้ในระหว่างการทำงานของ IBM PC ถ้าระดับแรงดันของแหล่งจ่ายไฟตกลง สัญญาณนี้จะถูกทำให้แอคตีฟเช่นกัน โดยทั่วไปแล้วสัญญาณนี้จะถูกนำไปใช้ในการรีเซตวงจรอินเตอร์เฟสหรืออุปกรณ์ I/O ต่าง ๆ ในช่วงที่เริ่มจ่ายไฟให้กับระบบซึ่งจะเป็นการทำให้วงจรหรืออุปกรณ์เหล่านั้นถูกปรับ ให้อยู่ในสภาวะที่แน่นอนก่อนที่จะเริ่มต้นการทำงานในระบบ

2.2.6 สัญญาณที่ใช้ในขบวนการ DMA

DRQ1 - DRQ3 (DMA Request 1-3 : ขา B6 และขา B16) :

ขาสัญญาณทั้งสามนี้เป็นสัญญาณอินพุทแอคตีฟที่ลอจิก " 1 " ซึ่งอุปกรณ์ภายนอกสามารถใช้ในการขอ DMA จากระบบ โดยการป้อนระดับลอจิก " 1 " ให้กับขา DRQ ขาใด

ขาหนึ่ง (ขา DRQ ทั้งสามนี้จะต่อเข้ากับ DRQ1 - DRQ3 ของ 8237A-5) เมื่อ 8237A-5 ได้รับสัญญาณนี้แล้วก็จะตรวจสอบว่ามีการขอ DMA ในช่องทางที่มีลำดับความสำคัญ (Priority) สูงกว่าหรือไม่ ถ้าไม่มีก็จะทำการขอ DMA จาก 8088 และตอบรับการขอ DMA จากอุปกรณ์ภายนอก แต่ถ้ามี 8237A-5 ก็จะทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญสูงกว่าก่อนแล้วจึงทำการขอ DMA ให้กับแชนแนลที่มีลำดับความสำคัญต่ำกว่าภายใน ROM BIOS ของ IBM PC จะโปรแกรม 8237A-5 ให้ DRQ1 ที่มีลำดับความสำคัญสูงสุด และ DRQ3 มีลำดับความสำคัญต่ำสุดดังนั้นถ้ามีการขอ DMA ของอุปกรณ์ภายนอกผ่านทางแชนแนลที่ 1 (DRQ1) และแชนแนลที่ 2 (DRQ2) 8237A-5 ก็จะทำการขอ DMA ของแชนแนลที่ 1 แล้วจึงจะทำการขอ DMA ให้กับแชนแนลที่ 2 อย่างไรก็ตาม 8237A-5 ยังมีช่องทางสำหรับการขอ DMA อยู่อีก 1 ช่องทาง คือ แชนแนล 0(DRQ0) ซึ่งในความเป็นจริงแล้วแชนแนลนี้จะมีลำดับความสำคัญที่สูงกว่าแชนแนลที่ 1 แต่จะไม่ถูกต่อออกมายังขาของสล็อต เนื่องจาก IBM PC จะใช้แชนแนล 0 นี้ในการรีเฟรชหน่วยความจำที่เป็น Dynamic RAM ในการขอ DMA นั้นสัญญาณ DRQ นี้ จะต้องแอคตีฟอยู่ในช่วงระยะเวลาหนึ่งเท่านั้นถ้าสัญญาณนี้แอคตีฟนานเกินไป จะทำให้เกิดขบวนการ DMA ขึ้นมากกว่า 1 ขบวนการได้

สัญญาณ DACK ของแชนแนลที่ขอ DMA นั้น ในการรีเซตสัญญาณ DRQ เช่น อุปกรณ์ภายนอกที่ขอ DMA ผ่านทางแชนแนลที่ 1 (DRQ1) ก็จะคอยตรวจสอบการตอบรับการขอ DMA จากสัญญาณ $\overline{\text{DACK}}$ ของแชนแนลที่ 1 ($\overline{\text{DRQ1}}$) เมื่อได้รับสัญญาณจาก $\overline{\text{DRQ1}}$ แล้วก็จะรีเซตสัญญาณ DRQ1 (เปลี่ยนจากลอจิก " 1 " เป็น " 0 ")

$\overline{\text{DACK0}}$ - $\overline{\text{DACK3}}$ (DMA Acknowledge 0-3 : ขา B19,B17,B26และ B15) :

สัญญาณทั้ง 4 นี้เป็นเอาท์พุทแอคตีฟที่ลอจิก " 0 " ซึ่ง 8237A-5 สร้างขึ้นเพื่อแสดงให้วงจรภายนอกที่ขอ DMA ทราบว่าการขอ DMA นั้นได้รับการตอบสนองแล้วและ 8237A-5 จะเข้าสู่ขบวนการ DMA เพื่อให้การส่งผ่านข้อมูลระหว่างอุปกรณ์ I/O ที่ขอ DMA กับหน่วยความจำที่เกิดขึ้นได้โดยตรง (คือไม่ต้องผ่าน 8088) โดยสัญญาณ $\overline{\text{DACK}}$ นี้จะ แอคตีฟในแชนแนลใดก็ขึ้นอยู่กับว่าขบวนการ DMA ที่เกิดขึ้นนั้น เป็นการตอบสนองต่อการขอ DMA ในแชนแนลที่ 2 (DRQ2) สัญญาณ $\overline{\text{DACK2}}$ ก็จะแอคตีฟ เป็นต้น

ดังที่กล่าวแล้วว่าสัญญาณ DRQ0 นั้นจะไม่ถูกต่อออกมายังขาของสล็อต ดังนั้นวงจรอินเตอร์เฟสจึงไม่สามารถจะขอ DMA ผ่านทางแชนแนล 0 ได้ แต่สัญญาณ $\overline{\text{DACK0}}$ จะถูกต่อออกมายังสล็อตด้วย (ขา B19) ทั้งนี้ก็เพื่อจะแสดงให้วงจรอินเตอร์เฟสต่าง ๆ ทราบว่า

ขบวนการ DMA ที่เกิดขึ้นในช่วงเวลาที่ $\overline{\text{DACK0}}$ แอคตีฟนั้นเป็นขบวนการที่ใช้สำหรับการรีเฟรชหน่วยความจำที่เป็น Dynamic RAM ซึ่งวงจรอินเตอร์เฟสที่ใช้หน่วยความจำประเภทนี้สามารถจะนำไปใช้ในวงจรได้

โดยที่การรีเฟรชหน่วยความจำนั้นจะเกิดขึ้นในทุก ๆ 15.12 usec. หรือทุก ๆ 72 คล็อก ดังนั้นสัญญาณ $\overline{\text{DACK0}}$ นี้จะแอคตีฟในทุก ๆ 15.12 usec. ด้วย

T / C (Terminal Count : B27):

สัญญาณนี้ถูกสร้างขึ้นจากการนำเอาสัญญาณเอาท์พุทที่ขา EOP ของ 8237A -5 มากลับลอจิก (โดยใช้เกท Inverter) ทำให้สัญญาณ T/C นี้แอคตีฟที่ลอจิก " 1 " สำหรับสัญญาณนี้จะแอคตีฟเมื่อจำนวนไบท์ในการส่งผ่านข้อมูลของขบวนการDMA ในแชนแนลใดแชนแนลหนึ่งครบตามจำนวนที่กำหนดไว้ โดยทั่วไปแล้วสัญญาณที่จะถูกใช้ในการสิ้นสุดขบวนการ DMA ที่ทำการส่งผ่านข้อมูลเป็นบล็อก เนื่องจากสัญญาณนี้จะแอคตีฟไม่แสดงว่าเป็นสัญญาณของแชนแนลใดดังนั้นจึงต้องทำการนำสัญญาณ T/C นี้ผ่านเกท Inverter แล้วนำไป OR กับสัญญาณ DACK เพื่อให้สามารถทราบได้ว่าสัญญาณ T/C ที่เกิดขึ้นนั้นเป็นสัญญาณของแชนแนลใด สำหรับในแชนแนลที่ 0 นั้น สัญญาณ T/C จะแอคตีฟในช่วงเวลาที่คงที่คือ ทุก ๆ 990.804 millsec. ซึ่งก็คือช่วงเวลาที่ใช้ในการรีเฟรชหน่วยความจำขนาด 64 กิโลไบท์

2.2.7 บัสแหล่งจ่ายไฟของระบบ

+5 VDC (ขา B3 และ B29) :

ขาทั้งสองนี้ต่อกับแหล่งจ่ายไฟ DC +5V ของระบบ โดยจะมีค่าความเที่ยงตรง (Regulated) ±5% คืออยู่ในช่วง +4.75 ถึง +5.25 VDC

+12 VDC (ขา B9) :

ขานี้ต่อกับแหล่งจ่ายไฟ DC +12 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Regulated) ± 5% คืออยู่ในช่วง +11.4 ถึง +12.6 VDC

-5 VDC (ขา B5) :

ขานี้ต่อกับแหล่งจ่ายไฟ DC -5 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Regulated) ± 10% คืออยู่ในช่วง -5.5 ถึง -4.5 VDC

-12 VDC (ขา B7) :

ขานี้ต่อกับแหล่งจ่ายไฟ DC -12 V ของระบบ โดยจะมีค่าความเที่ยงตรง (Regulated) ±10% คืออยู่ในช่วง -13.2 ถึง -4.5 VDC

GND (ขา B1, B10 และ B31) :

ขาทั้งสามนี้จะต่อเข้ากับกราวด์ (Ground) ของระบบ

2.3 การอ้างแอดเดรสของพอร์ท I/O

ในการควบคุมและตรวจสอบสภาวะการทำงาน รวมทั้งการอ่านข้อมูลจากอุปกรณ์ที่เป็นชิพซัพพอร์ทหรือการ์ดต่าง ๆ ที่ใช้ในระบบของ IBM PC นั้น จะกระทำโดยผ่านทางพอร์ท I/O ของระบบ ดังนั้นในการที่จะใช้งานหรือควบคุมการทำงานของอุปกรณ์เหล่านี้จึงจำเป็นต้องศึกษาถึงวิธีการควบคุมพอร์ท I/O ต่างๆ ของระบบด้วยและเนื่องจากการควบคุมหรือติดต่อกับพอร์ทเหล่านี้ต้องกระทำโดยการอ้างถึงแอดเดรสของพอร์ท I/O เหล่านั้นโดยตรง จึงจำเป็นต้องศึกษาถึงหลักการอ้างแอดเดรสของ 8088 ใน IBM PC ด้วย

สำหรับแอดเดรสของพอร์ท I/O ต่าง ๆ นั้น จะเป็นแอดเดรสที่ถูกสร้างขึ้นโดย 8088 ซึ่งแอดเดรสเหล่านี้เป็นแอดเดรสที่จัดไว้สำหรับพอร์ท I/O โดยเฉพาะ คือแยกจากแอดเดรสของหน่วยความจำโดยเด็ดขาด ส่วนการส่งข้อมูลเหล่านี้จะทำได้โดยการใช้คำสั่ง OUT ของ 8088 ส่งข้อมูลนั้นไปยังแอดเดรสของพอร์ทที่ต้องการ และสำหรับการตรวจสอบหรือการอ่านข้อมูลจากพอร์ทก็จะทำได้โดยการใช้คำสั่ง IN ของ 8088อ่านข้อมูลจากแอดเดรสของพอร์ทที่ต้องการเช่นกัน

ภายในไมโครโปรเซสเซอร์เบอร์ 8088 นี้จะมีแอดเดรสสำหรับใช้กับพอร์ท I/O อยู่ทั้งสิ้น 65,536 หรือ 64 กิโลแอดเดรส (ในขณะที่มีแอดเดรสสำหรับหน่วยความจำอยู่ 1 เมกะไบท์) ซึ่งทำให้การอ้างแอดเดรสของพอร์ท I/O ที่ทำงานร่วมกับ 8088 นั้น ต้องใช้จำนวนเส้นแอดเดรสในบัสแอดเดรสทั้งสิ้น 16 เส้น คือ A0-A15 แต่สำหรับใน IBM PC นี้ ถูกออกแบบมาให้ใช้เส้นแอดเดรสเฉพาะ 10 เส้นล่าง คือ A0-A9 เท่านั้น ดังนั้นในการอ้างถึงแอดเดรสของพอร์ทของอุปกรณ์หรือชิพซัพพอร์ทใด ๆ ที่ใช้ร่วมกับ IBM PC จึงใช้จำนวนเส้นแอดเดรสเพียง 10 เส้นด้วย โดยเส้นแอดเดรสที่เหลืออยู่คือ A10-A15 นั้นจะไม่ถูกนำไปใช้งาน แต่ค่าแอดเดรสบนเส้นแอดเดรสเหล่านี้ยังคงเปลี่ยนแปลงตามค่าแอดเดรสของพอร์ท

ที่กำหนดไว้ในคำสั่ง OUT หรือ IN อยู่ด้วย เพียงแต่ไม่ได้ถูกนำมาดีโค้ดร่วมกับแอดเดรส A0-A9 เท่านั้น ตัวอย่างเช่นในการใช้คำสั่ง OUT ส่งข้อมูลไปยังพอร์ทที่ตรงกับแอดเดรส 0010H นั้นจะให้ผลเหมือนกันกับการส่งข้อมูลไปยังพอร์ทที่ตรงกับแอดเดรส 0410H , 0810H, 0C10H ทั้งนี้เนื่องจากแอดเดรส 6 บิทบนไม่ได้ถูกใช้งาน จึงทำให้การเปลี่ยนแปลงค่าแอดเดรสบนเส้นแอดเดรส A10 - A15 นั้น ไม่ทำให้เกิดความแตกต่างใด ๆ ขึ้น

เนื่องจาก IBM PC ได้ใช้งานเส้นแอดเดรสเพียง 10 เส้น (คือ A0-A9) ดังนั้นจึงสามารถอ้างแอดเดรสของพอร์ทได้สูงสุดเพียง 1024 พอร์ท (จากจำนวน 64 กิโลพอร์ท) เท่านั้น นอกจากนี้ในกรณีที่เป็นการอ่านข้อมูลจากพอร์ทของ IBM PC ข้อมูลในบิท A9 จะถูกจัดให้มีหน้าที่ในการแบ่งพอร์ททั้ง 1024 พอร์ท ออกเป็น 2 ส่วน (ส่วนละ 512 พอร์ท) อีกด้วยกล่าวคือ ถ้าข้อมูลในบิท A9 เป็น "0" แล้ว จะทำการอ่านข้อมูลได้เฉพาะจากพอร์ทของอุปกรณ์หรือชิพซัพพ์พอร์ทต่าง ๆ ที่อยู่บนเมนบอร์ด (Main Board) ของ IBM PC เช่น 8253-5, 8237-5 หรือ 8259A เท่านั้น แต่ถ้าข้อมูลในบิท A9 นี้เป็น "1" ก็จะทำการอ่านข้อมูลได้เฉพาะจากพอร์ทที่อยู่บนการ์ดต่าง ๆ เท่านั้น

พอร์ทบน IBM PC ทั้ง 1024 พอร์ทถูกแบ่งออกเป็น 2 กลุ่มโดยที่กลุ่มแรกเป็นกลุ่มของพอร์ทที่อยู่บนเมนบอร์ด และกลุ่มที่สองเป็นกลุ่มที่จัดเตรียมไว้สำหรับพอร์ทที่อยู่บนการ์ดต่าง ๆ สำหรับในกรณีของการส่งข้อมูลให้กับพอร์ททั้ง 1024 พอร์ท ซึ่งสามารถที่จะเลือกส่งไปยังพอร์ทใด ๆ ใน IBM PC ได้ ดังนั้นการเลือกแอดเดรสสำหรับพอร์ทที่อยู่บนการ์ดจึงสามารถทำได้โดยสะดวก แต่อย่างไรก็ตามสิ่งหนึ่งที่จะต้องคำนึงถึงก็คือถ้าแอดเดรสที่เลือกให้กับพอร์ทนี้ตรงกับค่าแอดเดรสเดิมที่อยู่ บนเมนบอร์ดแล้ว เมื่อทำการส่งข้อมูลให้กับพอร์ทที่อยู่ในตำแหน่งแอดเดรสนี้ ก็จะเท่ากับเป็นการส่งข้อมูลให้กับทั้งพอร์ทที่อยู่บนเมนบอร์ด และพอร์ทที่อยู่บนการ์ดด้วยซึ่งในกรณีเช่นนี้อาจจะก่อให้เกิดความผิดพลาดขึ้นได้เช่นกัน ดังนั้นในการกำหนดค่าแอดเดรสให้กับพอร์ทที่ถูกสร้างขึ้นบนการ์ดต่าง ๆ จึงควรจะใช้ค่าแอดเดรสที่แอดเดรสบิท A9 มีค่าเป็น " 1 " คือ แอดเดรส 0FE00H และ 0FFFFH เท่านั้น (แอดเดรสบิท A10-A15 ไม่ถูกใช้ในการดีโค้ด แต่เพื่อความสะดวกจึงกำหนดให้มีค่าเป็น " 1 " ในฐานสองทั้งหมด แต่ในการใช้งานจริงอาจเปลี่ยนแปลงให้แอดเดรส A0-A15 แต่ละบิทมีค่าเป็น " 1 " หรือ " 0 " ก็ได้) การใช้งานแอดเดรสบิทต่างๆ ในการอ้างของพอร์ทแสดงได้ดังรูปที่ 2.3

รูปที่ 2.3 การใช้งานแอดเดรสบิทต่างๆ ในการอ้างของพอร์ท

2.4 หลักการเขียนโปรแกรมภาษาซี

ภาษาซีจัดเป็นภาษาระดับกลาง (Middle - Language) ที่นำมาเขียนโปรแกรมระบบปฏิบัติการยูนิกซ์ (UNIX) ภาษาระดับกลางถึงแม้ว่าจะเป็นภาษาที่เหมือนคำพูดภาษาอังกฤษธรรมดาๆ ซึ่งคล้ายกับภาษาระดับสูงแล้ว ก็ยังมีบางคำสั่งไปคล้ายคำสั่งของภาษาระดับต่ำอยู่ หรือเรียกได้ว่า ภาษาซีมีความสามารถเหมือนภาษาแอสเซมบลี้ (Assembly) คือมีคำสั่งที่สามารถเข้าถึงบิท, ไบท์ และตำแหน่งของหน่วยความจำของเครื่องได้ แต่เขียนง่ายเหมือนกับภาษาระดับสูงทั่วไป เช่น ภาษาปาสคาล (Pascal) เป็นต้น

2.4.1 คุณลักษณะของภาษาซี

1. ภาษาซีเป็นภาษาที่มีการโปรแกรมเป็นแบบลำดับขั้น (Procedure) ที่มีประสิทธิภาพมากที่สุด เพราะโปรแกรมสามารถสั่งงานได้เร็วกว่าทุกภาษา (ยกเว้นภาษาแอสเซมบลี้) ด้วยเหตุนี้โปรแกรมในสมัยปัจจุบันมักเขียนด้วยภาษาซี และถ้าส่วนไหนของโปรแกรมต้องการความเร็วเพิ่มขึ้น จึงจะใช้ภาษาแอสเซมบลี้เขียนในส่วนนั้นๆ ของโปรแกรมแทน

2. ภาษาซีเป็นภาษาที่ใช้ติดต่อกับส่วนต่าง ๆ ของคอมพิวเตอร์ได้ดีกว่าภาษาอื่นๆ

3. ถึงแม้ว่าตัวแปลงภาษาซีให้เป็นภาษาเครื่องจะมีอยู่มากแต่แทบทั้งหมดยึดเอามาตราฐานของ ANSI เป็นหลักจะมีต่างกันก็ในเรื่องของกราฟฟิก (Graphics) เท่านั้น

4. ภาษาซีเป็นภาษาที่มีความสามารถที่จะนำไปสั่งงานคอมพิวเตอร์ที่มีซีพียูต่างชนิดกันได้ (Portability สูง หมายถึง มีความสะดวกสูง) ดังนั้นโปรแกรมที่เขียนด้วยภาษาซีจึงง่ายต่อการที่จะนำไปใช้กับเครื่องต่างชนิดกัน เช่น คอมพิวเตอร์ส่วนบุคคลไปสู่มินิคอมพิวเตอร์ เป็นต้น

5. ภาษาซีเป็นภาษาที่สั่งงานให้คอมพิวเตอร์ทำงานได้แทบทุกอย่าง เท่าที่ภาษาสั่งงานจะทำได้ โดยอาจใช้ภาษาซีในการเขียนตัวแปลงภาษา (Compiler) ของภาษาอื่นๆ หรือเกี่ยวกับปัญญาประดิษฐ์ (Artificial Intelligence)

6. ภาษาซีเป็นภาษาที่มีโปรแกรมสนับสนุนมากและราคาต่ำ

7. ภาษาซีเป็นภาษาที่ใช้กำหนดรูปแบบของข้อมูลได้อย่างกว้างขวาง

2.4.2 ข้อจำกัดของภาษาซี

1. เมื่อเปรียบเทียบภาษาซีกับภาษาชั้นสูงอื่นๆ จะเห็นว่าภาษาซีอยู่ในระดับต่ำกว่า ดังนั้นโปรแกรมภาษาซีจึงยาวกว่าภาษาอื่นๆ โดยปกติเวลาจะเขียนโปรแกรมสำเร็จรูปจริงๆ อาจจำเป็นต้องซื้อโปรแกรมสนับสนุนมาช่วยเพื่อทำให้โปรแกรมสมบูรณ์เร็วขึ้น ซึ่งนั่นก็หมายความว่าค่าใช้จ่ายต้องเพิ่มขึ้น

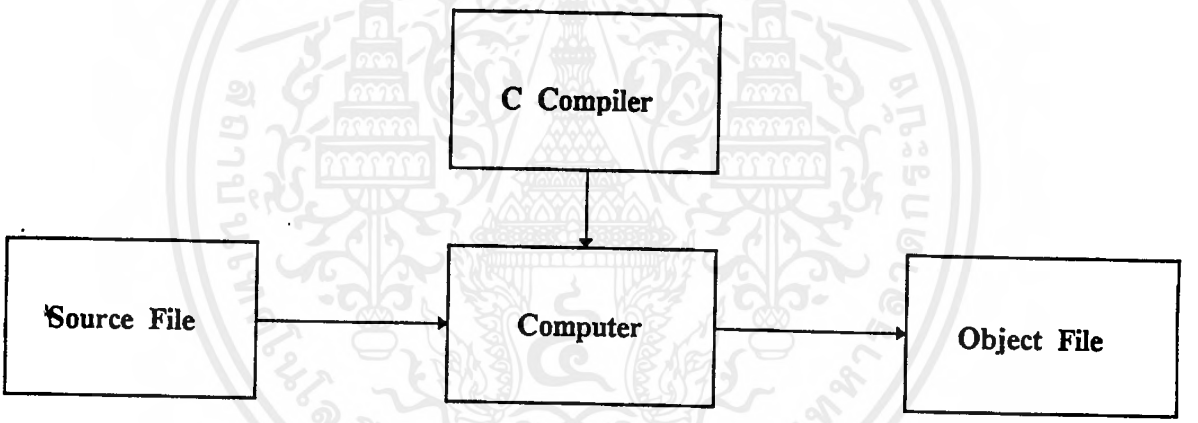
2. ภาษาซีเป็นภาษาที่มีการสั่งการแบบมีลำดับขั้นนั่นคือก่อนที่จะทำการแก้ปัญหาใดๆ จะต้องรู้ขั้นตอนของการแก้ปัญหานั้นก่อน

3. ภาษาซีเป็นภาษาที่ตัวโปรแกรมอ่านเข้าใจได้ยากกว่าภาษาชั้นสูงอื่นๆ

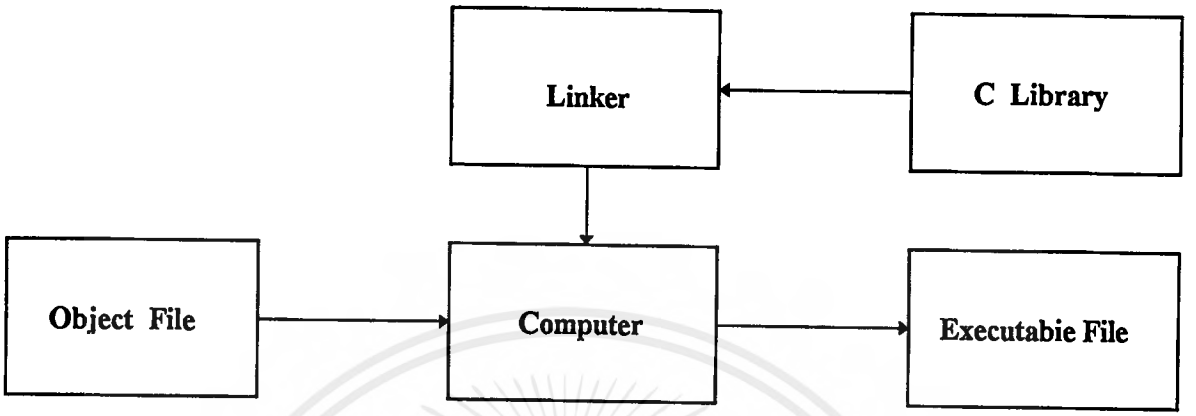
4. ถึงแม้ว่าภาษาซีจะถูกทำให้เข้าใจได้กับมาตราฐานเดียวกัน แต่ก็ยังมีสิ่งที่ต่างกันอยู่อีกมาก

2.4.3 ขั้นตอนการทำโปแกรมโดยใช้โปรแกรมภาษาซี

โปรแกรมต่างๆ ที่เขียนขึ้นเพื่อสั่งงานคอมพิวเตอร์นั้นเรียกว่า ซอสโค้ด (Source Code) หรือ ซอสโปรแกรม (Source Program) โดยใช้อิดิเตอร์ (Editor)สร้างซอสโปรแกรม เมื่อเขียนเสร็จเรียบร้อยจะมีขั้นตอนในการทำงาน เพื่อให้ได้ผลลัพธ์ออกมา 2 ขั้นตอนซึ่งแสดงได้ดังรูปที่ 2.4 และ 2 . 5 ตามลำดับดังนี้



รูปที่ 2.4 การแปลโปรแกรมในภาษาซี



รูปที่ 2.5 การเชื่อมโยงในโปรแกรมภาษาซี

ขั้นตอนที่ 1 การแปลโปรแกรม (Compile) ในรูปที่ 2.4 เป็นการแปลโปแกรมในภาษาซีโดยนำโปรแกรมที่เขียนขึ้น (Source File) มาทำการแปลเป็นโปรแกรมภาษาเครื่อง (Object File) โปรแกรมที่เขียนขึ้นนี้มีกฎการตั้งชื่อเหมือนกับการตั้งชื่อแฟ้มข้อมูลในดอส แต่จะมีชนิดของแฟ้มข้อมูลเป็นนามสกุล จุดซีเท่านั้น เช่น TOP.C เมื่อโปรแกรมนี้ผ่านการแปลโปรแกรมแล้วจะได้เป็นโปรแกรมภาษาเครื่องที่มีแฟ้มข้อมูลเป็น TOP.OBJ

ขั้นตอนที่ 2 การเชื่อมโยง (Linker) ในรูปที่ 2.5 เป็นการเชื่อมโยงในโปรแกรมภาษาซี คือการนำโปรแกรมภาษาเครื่องมาทำการเชื่อมโยงกับฟังก์ชั่นที่ต้องการจากไลบรารี่ (Library) เพื่อสร้างให้เป็นแฟ้มข้อมูลในการสั่งปฏิบัติงาน (Executable File) จะเป็นแฟ้มข้อมูลที่สามารถสั่งให้ปฏิบัติงานได้โดยตรงในขณะที่อยู่ในดอส แฟ้มข้อมูลชนิดนี้จะเป็น .EXE

2.4.4 ชุดคำสั่งของภาษาซีที่ใช้ในการติดต่อกับพอร์ท

โปรแกรมภาษาซีมีชุดคำสั่งที่ใช้อำนวยความสะดวกในการเขียนโปรแกรมติดต่อกับระบบหรือติดต่อกับอุปกรณ์ภายนอกเช่นเดียวกับโปรแกรมภาษาอื่นๆ และในการติดต่อกับอุปกรณ์ภายนอกคอมพิวเตอร์จะใช้พอร์ท (PORT) เป็นตัวรับส่งข้อมูล

จากที่ได้กล่าวมาแล้วว่า คอมพิวเตอร์ประกอบไปด้วยหน่วยประมวลผลกลาง หน่วยความจำ และหน่วยรับส่งข้อมูลเข้าออก ในเครื่องคอมพิวเตอร์หน่วยประมวลผลกลางก็คือ

ซีพียูในตระกูล 80XXX หน่วยความจำในที่นี้ก็คือหน่วยความจำหลักของระบบ การที่ซีพียูจะเขียนหรืออ่านข้อมูลกับส่วนอื่นใดที่นอกเหนือไปจากหน่วยความจำ จะต้องกระทำผ่านพอร์ทเสมอ

ในกรณีที่ผู้เขียนโปรแกรมต้องการจัดการอุปกรณ์เหล่านี้ด้วยตนเอง โดยไม่ผ่านระบบปฏิบัติการ หรือต้องการจัดส่วนที่ระบบปฏิบัติการไม่มีความสามารถในการจัดการในส่วนนั้น ผู้เขียนจึงต้องเขียนหรืออ่านข้อมูลกับพอร์ทโดยตรง

ภาษาซีกำหนดฟังก์ชั่นสำหรับติดต่อกับพอร์ทไว้ซึ่งสามารถดูได้จากตารางที่ 2.1 ดังต่อไปนี้

ฟังก์ชั่น	อธิบาย
unsigned inpw(unsigned portid); int inp(unsigned portid);	ทำหน้าที่อ่านข้อมูลขนาด 16 บิท โดยอ่านข้อมูล 8 บิทล่างจากพอร์ทหมายเลข และข้อมูล 8 บิทบนจากพอร์ทหมายเลข portid +1 โดยจะส่งค่ากลับเป็นข้อมูลที่อ่านค่าได้
unsigned outpw(unsigned portid, unsigned value); int outp(unsigned portid , int value);	ทำหน้าที่อ่านข้อมูลขนาด 8 บิทจากพอร์ทหมายเลข portid โดยส่งค่ากลับเป็นข้อมูลที่อ่านได้
int inport(int portid); unsigned char inportb(int portid);	ทำหน้าที่อ่านข้อมูลขนาด 16 บิท ไปที่พอร์ทหมายเลข portid และ portid+1 โดยจะส่งค่า 8 บิทล่างไปที่หมายเลข portid และส่ง 8 บิท บนไปที่ port+1 ค่าที่มาโคร outpw ( ) ส่งกลับก็คือข้อมูลที่ส่งออกไปนั่นเอง
void outport(int portid, int portid); void outportb(int portid, unsigned char value);	ทำหน้าที่เขียนข้อมูล 8 บิทไปที่พอร์ทหมายเลข portid โดยฟังก์ชั่น outp ( ) จะส่งค่ากลับมาเป็นค่าข้อมูลที่ส่งออกไป

ตารางที่ 2.1 ฟังก์ชั่นสำหรับติดต่อกับพอร์ท

บทที่ 3

การออกแบบและการสร้าง

3.1 ส่วนประกอบโดยทั่วไป

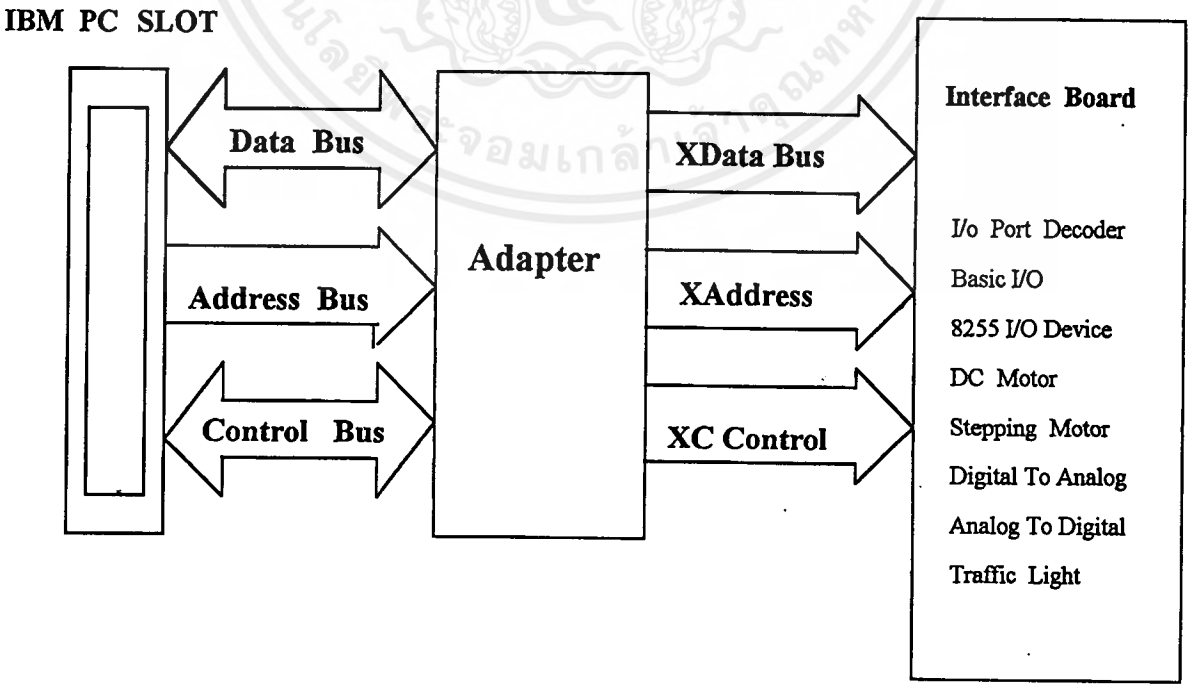
ชุดฝึกในปริญญานิพนธ์ฉบับนี้มีส่วนประกอบด้วยกัน 3 ส่วน ซึ่งแสดงดังตารางที่ 3.1

ส่วนประกอบ	คำอธิบาย
1. ส่วนของอะแดปเตอร์การ์ด	เป็นส่วนที่ใช้เป็นสัญญาณในการติดต่อกับอุปกรณ์ภายนอกตามต้องการซึ่งประกอบด้วย 1. บัสข้อมูล 2. บัสควบคุม 3. แอดเดรสบัส
2. อินเตอร์เฟสบอร์ด	เป็นส่วนของบอร์ดที่ใช้ในการทดลองซึ่งประกอบด้วย 1. วงจรพอร์ทดีโค้ด 2. วงจรควบคุมการสร้างเวทสเตท 3. วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น 4. วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น 5. วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 6. วงจรดีซีมอเตอร์

ตารางที่ 3.1 ส่วนประกอบที่สำคัญของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

ส่วนประกอบ	คำอธิบาย
2. อินเตอร์เฟสบอร์ด	7. วงจรสเต็ปปิ้งมอเตอร์ 8. วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก 9. วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล 10. วงจรควบคุมสัญญาณไฟจราจร
3. ส่วนของแหล่งจ่ายไฟ	ทำหน้าที่แปลงแรงดันไฟฟ้ากระแสสลับ 220 โวลท์ เป็นแรงดันไฟฟ้ากระแสตรงขนาด 5, -5, 12,-12 โวลท์ เพื่อไปเลี้ยงวงจรในส่วนต่าง ๆ

ตารางที่ 3.1 ส่วนประกอบที่สำคัญของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม (ต่อ)



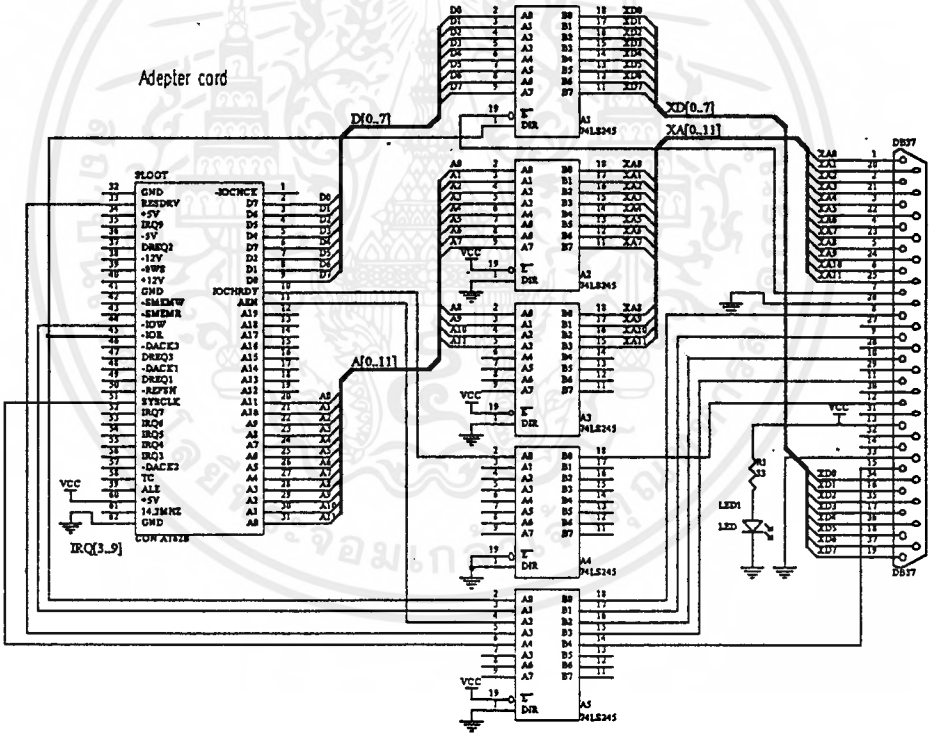
รูปที่ 3.1 ผังการทำงานของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

3. 2 การออกแบบและการทำงานในส่วนต่าง ๆ

จากรูปที่ 3.1 เป็นผังการทำงานที่ใช้ในการออกแบบชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมซึ่งสามารถแยกอธิบายในส่วนต่าง ๆ ของการออกแบบและการสร้างได้ดังต่อไปนี้

3.2.1 ส่วนของอะแดปเตอร์การ์ด

ส่วนของวงจรบนอะแดปเตอร์การ์ดเป็นส่วนของวงจรที่ทำหน้าที่ในการควบคุม การติดต่อระหว่าง IBM PC กับวงจรอินเตอร์เฟสภายนอก ทำหน้าที่เป็นวงจรบัฟเฟอร์ (Buffer) การออกแบบวงจรบนอะแดปเตอร์การ์ด แสดงได้ดังรูปที่ 3.2 เพื่อให้ง่ายขึ้นขอแยกอธิบายการทำงานของวงจรคือ

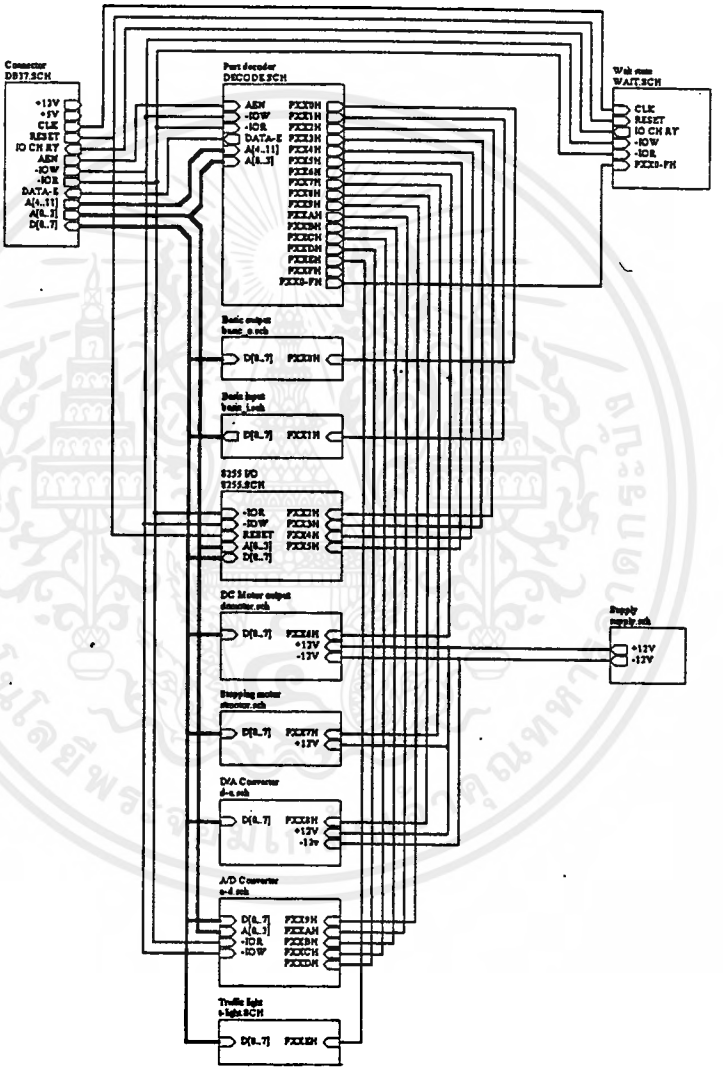


รูปที่ 3.2 วงจรบนอะแดปเตอร์การ์ด

ไอซี 74LS245 ทำหน้าที่เป็นวงจรบัฟเฟอร์เพื่อให้จ่ายกระแสได้สูงขึ้นโดย A2 และ A3 จะเป็นตัว บัฟเฟอร์ให้กับแอดเดรสบัส A0-A11 โดย A1 เป็นบัฟเฟอร์ให้กับบัสข้อมูล D0-D7 ซึ่ง A4 เป็นบัฟเฟอร์ให้กับอินพุทของบัสควบคุม คือ IO CHRDY และ A5 เป็นบัฟเฟอร์ให้กับเอาท์พุทของบัสควบคุม คือ RESET, $\overline{IOW}$, $\overline{IOR}$, SYSCLK, AEN

3.2.2 ส่วนของอินเตอร์เฟสบอร์ด

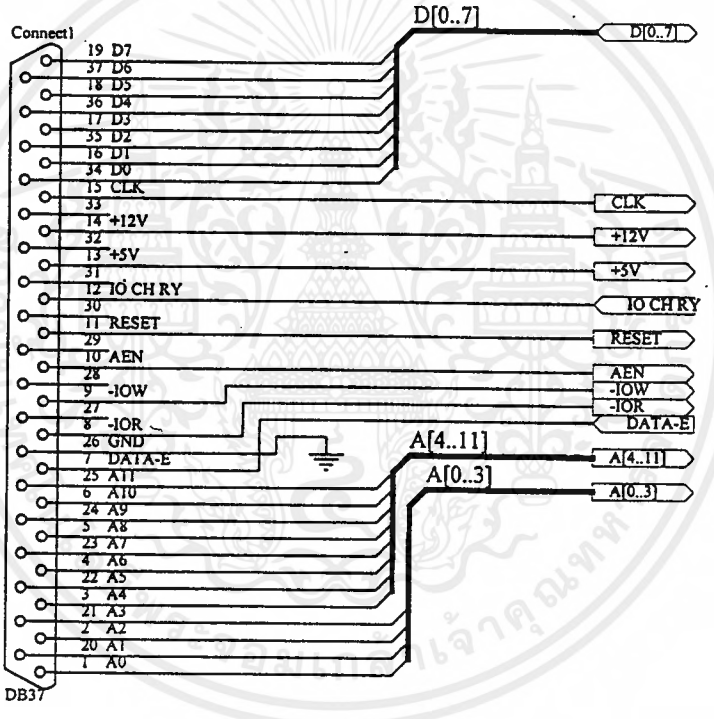
จากรูปที่ 3.3 เป็นการออกแบบชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมในส่วนของอินเตอร์เฟสบอร์ดซึ่งเป็นวงจรที่ใช้ในการทดลองใบงานประกอบด้วย



รูปที่ 3.3 อินเตอร์เฟสบอร์ด (Interface Board)

3.2.3 ส่วนของคอนเน็กเตอร์ DB37

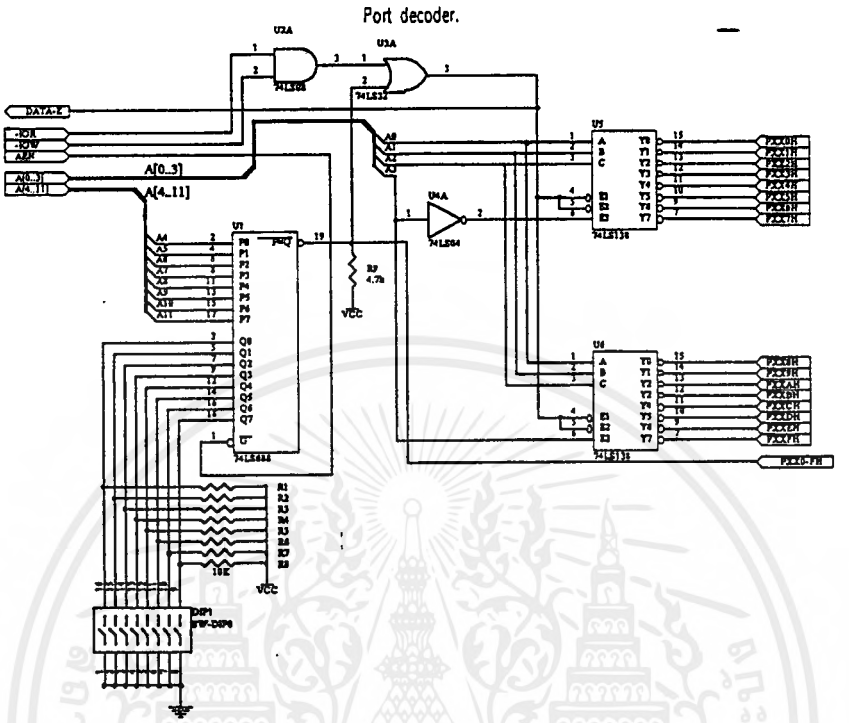
จากรูปที่ 3.4 เป็นการออกแบบในส่วนของคอนเน็กเตอร์ DB37 ซึ่งใช้ในการเชื่อมต่อสัญญาณต่าง ๆ ของ IBM PC สล็อต ที่จะนำมาควบคุมการทำงานของวงจรบนอินเตอร์เฟสบอร์ดผ่านมาทางสายแพร ซึ่งสัญญาณที่จะถูกนำมาควบคุมการทำงานของวงจรต่างๆ บนอินเตอร์เฟสบอร์ดนี้ จะขึ้นอยู่กับว่าในขณะนั้นผู้เขียนโปรแกรมได้สั่งการให้โปรแกรมทำงานในส่วนของวงจรใด



รูปที่ 3.4 คอนเน็กเตอร์ DB37

3.2.4 ส่วนของวงจรพอร์ทดีโค้ด

จากรูปที่ 3.5 เป็นการออกแบบวงจรพอร์ทดีโค้ด (PORT DECODE) เป็นส่วนของวงจรที่ทำหน้าที่ถอดรหัสแอดเดรสพอร์ท



รูปที่ 3.5 วงจรพอร์ทดีโค้ด

ซึ่งในการออกแบบไม่ต้องการที่จะให้มีหมายเลขพอร์ทซ้ำกัน ดังนั้นการออกแบบวงจรในส่วนนี้จำเป็นต้องอาศัยตารางการจัดสรรแอดเดรสที่ใช้ติดต่อกับอุปกรณ์ภายนอกบน IBM PC ในการถอดรหัส

ในการสร้างวงจรพอร์ทดีโค้ด ประกอบด้วย IC U1,U2A,U3A,U4A,U5,U6 ,R1-R8 และดิฟสวิทช์ 1 (DIP. 1) ทำหน้าที่ดีโค้ดตำแหน่งของพอร์ทได้ตั้งแต่แอดเดรส ที่ 00XH ถึง FFXH (อยู่ที่การกำหนดการดีโค้ดให้เหมาะสมกับตำแหน่ง I/O MAP ของ IBM PC)

จากรูปที่ 3.5 ส่วนของวงจรจะเห็นได้ว่ามีการนำสัญญาณ -IOR , -IOW , AEN มาประกอบในการดีโค้ดด้วย และเมื่อมีการเขียนหรืออ่านข้อมูลผ่านพอร์ทก็จะเป็นลอจิก "0" และเมื่อไม่มีขบวนการ DMA เกิดขึ้น สัญญาณ AEN ก็จะเป็นลอจิก "0"

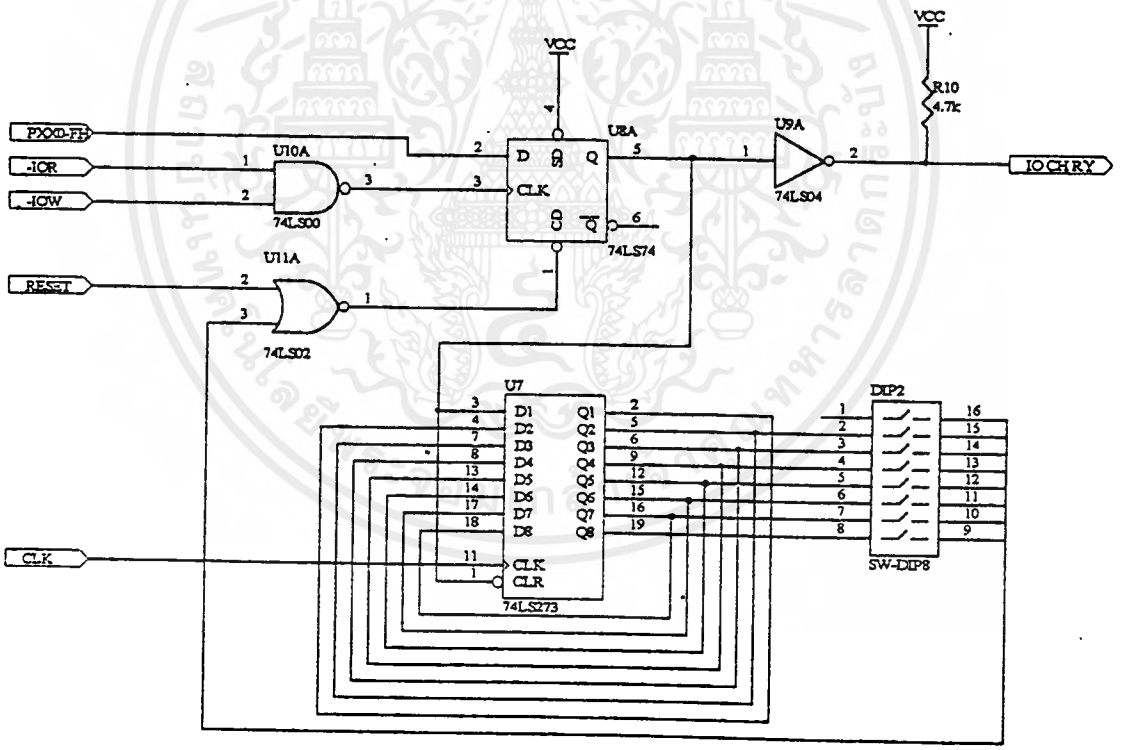
และเมื่อทำการโยกดิฟสวิทช์ 1 ก็จะทำการเปรียบเทียบแอดเดรสบัส (A4-A11) เพื่อทำการเลือกแอดเดรส และเมื่อการเปรียบเทียบเป็นไปอย่างถูกต้อง และ AEN เป็น "0" ก็จะมี

ซึ่งเมื่อ -IOR , -IOW , เป็นลอจิก "0" ด้วยแล้ว U5 และ U6 จะทำการดีโค้ดให้กับวงจรต่าง ๆ ในอินเตอร์เฟสบอร์ด

3.2.5 วงจรควบคุมการสร้างสัญญาณเวทสเตท

เนื่องจากขาสัญญาณ I/O CHRDY (I/O Channel Ready) เป็นอินพุทที่ใช้เพิ่มช่วงเวลาในบัสไซเคิลในกรณีที่อุปกรณ์ I/O หรือหน่วยความจำที่เกี่ยวข้องกับขบวนการในบัสไซเกิลที่เกิดขึ้นนั้นไม่สามารถทำงานทันตามช่วงเวลาปกติ

เมื่ออุปกรณ์ I/O หรือหน่วยความจำต้องการที่จะเพิ่มช่วงเวลาในบัสไซเกิลให้นานขึ้นอีกนั้นจะสามารถทำได้โดยการป้อนลอจิก "0" ให้กับขา I/O CHRDY (WAIT STATE CONTROL) ในช่วงเวลาที่ I/O หรือหน่วยความจำถูกกำหนดขึ้นได้รับสัญญาณการดีโค้ดแอดเดรสและสัญญาณ $\overline{MEMR}$, $\overline{MEMW}$, $\overline{IOR}$ และ $\overline{IOW}$ ซึ่งแสดงส่วนของวงจรควบคุมการสร้างสัญญาณเวทสเตทได้ดังรูปที่ 3. 6



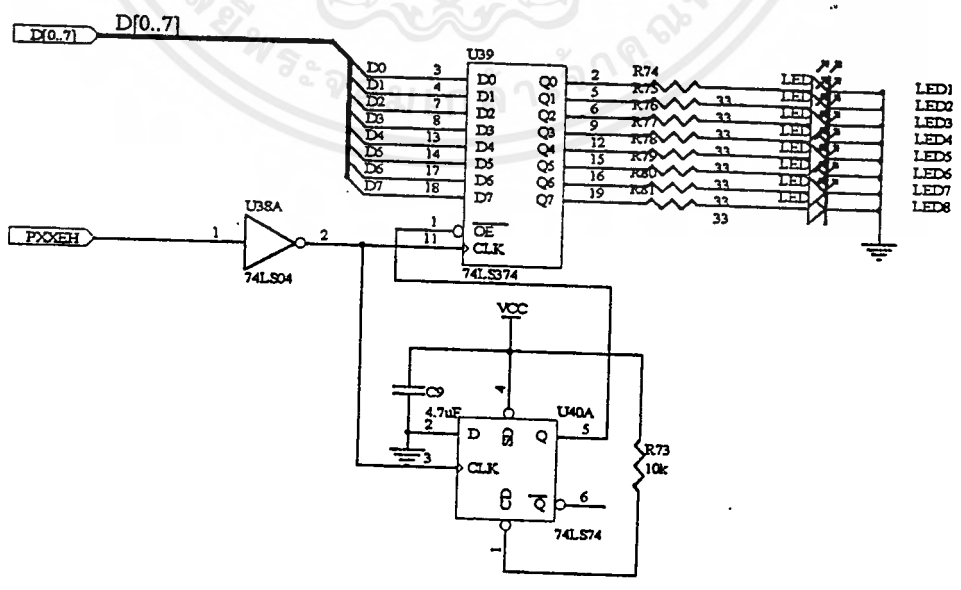
รูปที่ 3.6 วงจรควบคุมการสร้างสัญญาณเวทสเตท

จากรูปที่ 3.6 เป็นการออกแบบวงจรควบคุมการสร้างสัญญาณเวทสเตท ประกอบด้วยไอซีเบอร์ 74LS00 , 74LS02 , 74LS04 , 74LS74 , 74LS273 และดิฟสวิทช์ 2 (DIP.2)

เป็นวงจรที่สามารถใช้ในการขอให้วงจรบนเมนบอร์ดทำการสร้างสัญญาณ เวทสเตท ในไซเคิลของการอ่านและการเขียนข้อมูลบนอินพุท / เอาท์พุทพอร์ท ได้ตั้งแต่ 1-8 ลูก การทำงานของวงจรเริ่มจากที่ขา 3 ของไอซีเบอร์ 74LS74 มีสัญญาณนาฬิกาทุกครั้งที่มีการอ้างถึงพอร์ทที่ได้ทำการดีโค้ดไว้ ทำให้ไอซี U9A เป็นลอจิก "1" ในขณะเดียวกันสมมุติว่าขณะนี้ดิฟสวิทช์ 2 ทุกตัวอยู่ในตำแหน่งปิด (OFF) ที่ขา 13 ของ ไอซีเบอร์ 74LS32 จะเป็นลอจิก " 1 " ทำให้ขา 8 ของ ไอซีเบอร์ 74LS00 เป็นลอจิก "0" ไปทำการเคลียร์ ไอซีเบอร์ 74LS74 ให้ที่ไอซี U9A เป็นลอจิก "0" ไอซี U9A จะปิดทำให้ I/O CHRDY มีลอจิก "1" วงจรเมนบอร์ดจะสร้างสัญญาณเวทสเตทขึ้น 1 ลูก ซึ่งเป็นไปตามปกติ คือถ้ามีกระบวนการติดต่อกับอุปกรณ์อินพุทและอุปกรณ์เอาท์พุท และที่ I/O CHRDY เป็นลอจิก "1" วงจรเมนบอร์ดจะสร้างสัญญาณเวทสเตท ขึ้น 1 ลูก แต่ถ้ามีการโยกดิฟสวิทช์ 2 ตัวที่ 1,2,3...7 ก็จะทำให้วงจรเมนบอร์ดสร้าง สัญญาณเวทสเตท ขึ้น 2,3,4,5...8 ลูกตามลำดับ

3.2.6 วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

ในส่วนของวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น เป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำเครื่องคอมพิวเตอร์ไปควบคุมอุปกรณ์ภายนอกอย่างง่ายและแสดงให้เห็นถึงการเขียนโปรแกรมเพื่อส่งข้อมูลไปควบคุมให้ไดโอดเปล่งแสงติด - ดับตามต้องการ ซึ่งการทำงานของวงจรสามารถอธิบายได้ดังรูปที่ 3.7

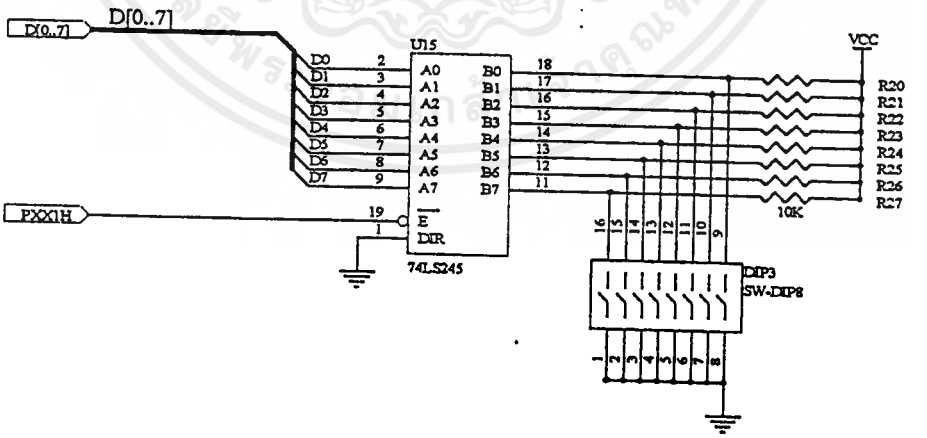


รูปที่ 3.7 วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

จากรูปที่ 3.7 ประกอบด้วยอุปกรณ์ตัวหลัก ๆ คือ U13 ไอซีเบอร์ 74LS374 ทำหน้าที่ในการค้างข้อมูลที่ส่งเข้ามา โดยแสดงผลด้วยไดโอดเปล่งแสง (LED1 - LED8) , U14A ไอซี 74LS74 เป็นไอซีดีฟลิปฟลอป(D-Flip Flop) ทำหน้าที่เคลียร์ไอซี 74LS374 ในจังหวะเริ่มแรกที่จ่ายไฟให้กับวงจร การทำงานคือ เมื่อเริ่มเปิดไฟเข้าเครื่องครั้งแรก R19 และ C1 จะทำให้ขา Q ของ IC 74LS74 มีลอจิกเป็น "1" เป็นผลให้ Q0 - Q7 ของไอซี 74LS374 อยู่ในสภาวะ HIGH IMPEDANCE ทำให้ไดโอดเปล่งแสงดับหมดทุกดวง แต่ถ้ามีการส่งข้อมูลมายังพอร์ท ก็จะทำให้ขา Q ของไอซี 74LS74 มีลอจิกเป็น "0" ทำให้ข้อมูลที่ขา D0 -D7 ของไอซี 74LS374 ปรากฏที่ขา Q0 - Q7 ของไอซี 74LS374 และข้อมูลจะคงค้างอยู่ทำให้ไดโอดเปล่งแสงติดอยู่เช่นนั้นจนกว่าจะส่งข้อมูลชุดใหม่ออกมาหรือปิดไฟที่เข้าเครื่อง

3.2.7 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

ในส่วนของวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้นเป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำคอมพิวเตอร์ไปควบคุมอุปกรณ์อินพุทอย่างง่าย โดยการเขียนโปรแกรมเพื่อรับข้อมูลจากอุปกรณ์อินพุทประเภทสวิทช์ ซึ่งการสร้างและการทำงานของวงจรสามารถอธิบายได้ดังรูปที่ 3.8

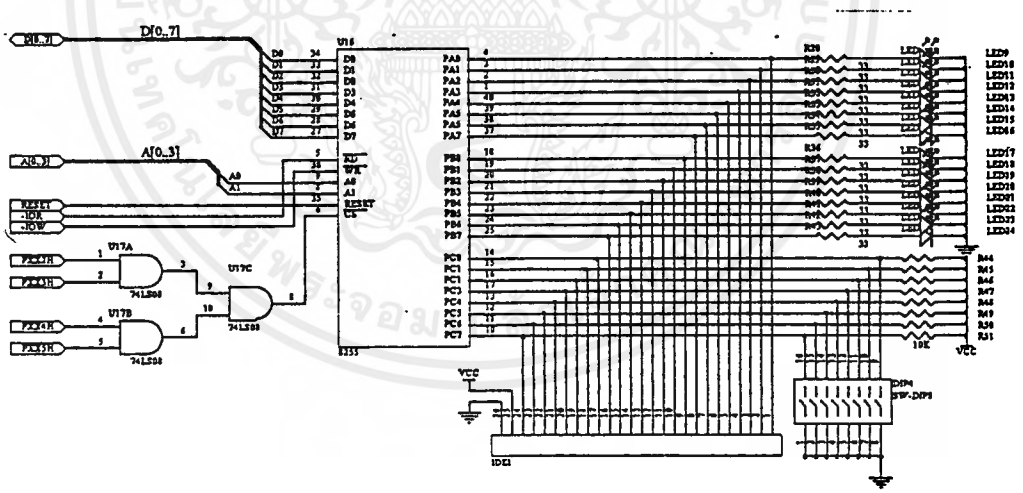


รูปที่ 3.8 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

จากรูปที่ 3.8 วงจรประกอบด้วยอุปกรณ์หลัก คือ U15 ไอซี 74LS245 ซึ่งเป็นไอซีทำหน้าที่เป็นตัวรับข้อมูลจากดิฟสวิทช์ 3 (DIP.3) เพื่อส่งให้กับคอมพิวเตอร์ ไอซี 74LS245 ถูกดีโค้ดไว้ที่พอร์ท XX1H การทำงานในสภาวะปกติ ถ้าดิฟสวิทช์ 3 อยู่ในตำแหน่งของสวิทช์ 3 ปิด (OFF) ที่ B0 ถึง B3 และ B4 ถึง B7 จะมีลอจิกเป็น "1" เนื่องจากถูกดึงเข้ามาด้วย R20 - R27 แต่ถ้าภายในดิฟสวิทช์ 3 ซึ่งมีอยู่ด้วยกัน 8 ตัวและตัวในตัวหนึ่งอยู่ในตำแหน่งเปิด(ON) ก็จะได้ลอจิก "0" ที่ตำแหน่งนั้น และถ้ามีการเขียนโปรแกรมเพื่อรับข้อมูลจากพอร์ท ก็จะได้ข้อมูลที่มีลอจิกเป็น "0" ณ. ตำแหน่งนั้นด้วย

3.2.8 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255

เป็นการใช้ 8255 ควบคุมอุปกรณ์อินพุทและอุปกรณ์เอาท์พุท ซึ่งให้แสดงผลผ่านทางไดโอดเปล่งแสง รูปที่ 3.9 แสดงส่วนของวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255

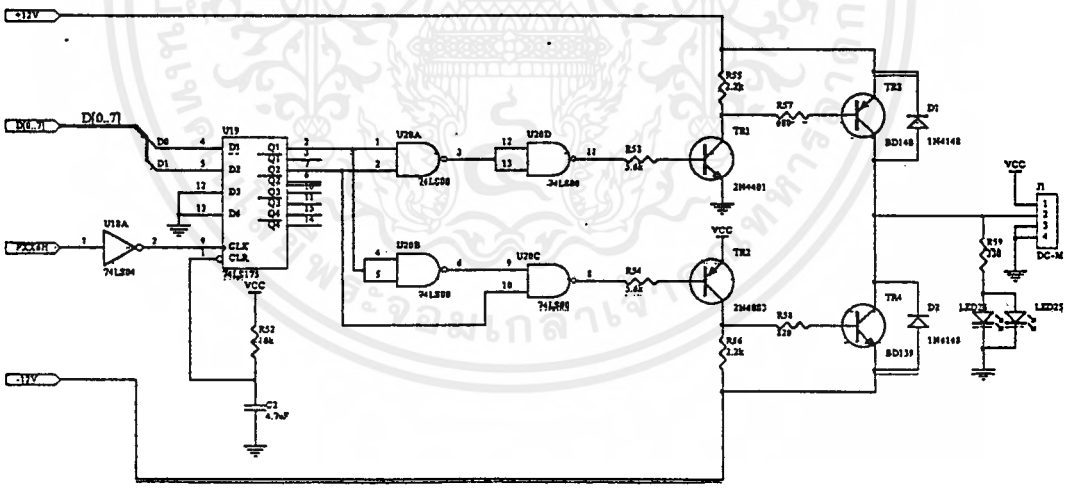


รูปที่ 3.9 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255

จากรูปที่ 3.9 วงจรประกอบด้วยอุปกรณ์หลัก คือ ไอซี 8255 ซึ่งถูกดีโค้ดไว้ที่ XX2H - XX5H จากวงจรกำหนดให้พอร์ท A และ พอร์ท B เป็นเอาท์พุทพอร์ท และพอร์ท C เป็น อินพุทพอร์ท ทำงานในโหมด 0 แต่ถ้าต้องการใช้งาน 8255 ในโหมด 1และโหมด 2 ก็สามารถทำได้โดยการเปลี่ยนค่าของรหัสควบคุม (Control Word) สัญญาณจาก 8255 ทั้งพอร์ท A,B,C จะสามารถถูกนำไปใช้โดยการต่อที่ IDE1

3.2.9 วงจรดีซีมอเตอร์

ในส่วนของวงจรดีซีมอเตอร์เป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำคอมพิวเตอร์ไปควบคุมอุปกรณ์ประเภทมอเตอร์ ซึ่งเป็นการควบคุมทิศทางการหมุนของดีซีมอเตอร์เท่านั้น ว่าจะให้ดีซีมอเตอร์หมุนแบบตามเข็มนาฬิกา (Forward) หรือหมุนแบบทวนเข็มนาฬิกา (Reverse) และสามารถเขียนโปรแกรมเพื่อควบคุมทิศทางการหมุนของดีซีมอเตอร์ได้ ซึ่งการสร้างและการทำงานของวงจรสามารถอธิบายได้จากรูปที่ 3.10



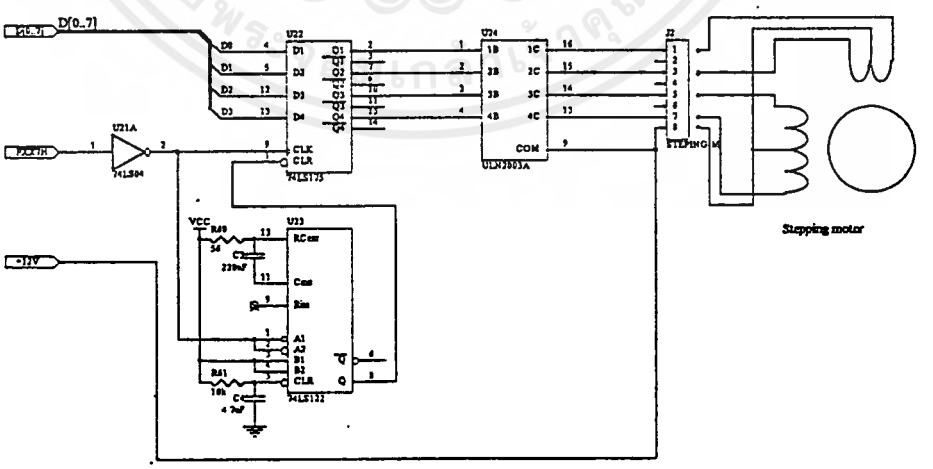
รูปที่ 3.10 วงจรดีซีมอเตอร์

จากรูปที่ 3.10 วงจรที่ใช้ควบคุมการทำงานของดีซีมอเตอร์ จะถูกดีโค้ดไว้ที่พอร์ท XX6H วงจรประกอบด้วยอุปกรณ์หลัก ๆ คือ U19 ไอซี 74LS175 , U20 ไอซี 74LS00 ทรานซิสเตอร์ขับมอเตอร์, ไดโอดเปล่งแสงเพื่อแสดงทิศทางการหมุนของมอเตอร์

การทำงานของวงจรเริ่มจากถ้ามีการส่งข้อมูลมาที่พอร์ท XX6H ซึ่งในวงจรจะใช้ข้อมูลจาก D0 - D7 มาใช้ในการควบคุมทิศทางการหมุนของดีซีมอเตอร์ โดยทิศทางการหมุนสัมพันธ์กับข้อมูลที่ D0-D7 ถ้ามีการเขียนโปรแกรมและทำให้ D0 เป็นลอจิก "0" และ D1 เป็นลอจิก "1" ข้อมูลจะถูกแลทช์ไว้ที่ U19 ทำให้ที่ขา 11 ของ U20D เป็นลอจิก "0" และขาที่ 8 ของ U20C เป็นลอจิก "0" เป็นผลให้ ทรานซิสเตอร์ตัวที่ 1 ปิด; ทรานซิสเตอร์ตัวที่ 2 เปิด; ทรานซิสเตอร์ตัวที่ 3 เปิด ; ทรานซิสเตอร์ตัวที่ 4 ปิด; ทำให้มีกระแสไหลจาก +12 โวลท์ ไปยังดีซีมอเตอร์ ทำให้ดีซีมอเตอร์หมุนแบบตามเข็มนาฬิกา และในการทำให้ดีซีมอเตอร์หมุนแบบทวนเข็มนาฬิกา การทำงานของวงจรก็เช่นเดียวกับการทำงานในช่วงที่ดีซีมอเตอร์หมุนแบบตามเข็มนาฬิกา ต่างกันเพียงแต่การสลับการเปิด-ปิดของทรานซิสเตอร์เท่านั้น

3.2.10 วงจรสเตปปิ้งมอเตอร์

ในงานอุตสาหกรรมที่ต้องการความแม่นยำ ในการขับเคลื่อนสเตปปิ้งมอเตอร์จึงถูกนำมาใช้งานอย่างกว้างขวาง และในส่วนของวงจรนั้นเป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำคอมพิวเตอร์ไปควบคุมการขับเคลื่อนสเตปปิ้งมอเตอร์ซึ่งการสร้างและการทำงานของวงจรสามารถอธิบายได้ดังรูปที่ 3.11



รูปที่ 3.11 วงจรสเตปปิ้งมอเตอร์

การควบคุมการหมุนของสเตปปิ้งมอเตอร์ สามารถควบคุมได้ทั้งทิศทาง (Direction) และตำแหน่ง(Position) จากวงจรประกอบด้วยอุปกรณ์หลัก ๆ คือ U22 ไอซี 74LS175, U24 ไอซี ULN2003A , U23 ไอซี 72LS122

หน้าที่ของอุปกรณ์แต่ละตัวมีดังนี้

U22 ไอซี 74LS175 ทำหน้าที่ค้างข้อมูลที่ส่งมาที่พอร์ท XX7H

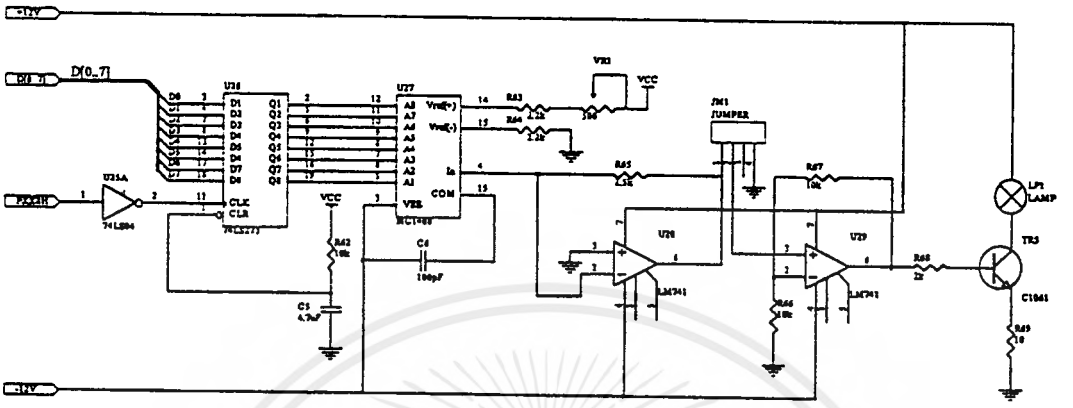
U23 ไอซี 72LS122 เป็นวงจรรีทริกกิ้งโมโนสเตเบิลมัลติไวเบรเตอร์ ทำหน้าที่เคลียร์ U22 ไอซี 74LS175 ในกรณีที่ไม่มีการส่งข้อมูลมาที่พอร์ม XX7H เพื่อไม่ให้ U22 ค้างข้อมูลอยู่ตลอดเวลาเพราะถ้ามีการค้างข้อมูลที่ ลอจิก "1" ที่ U22 จะทำให้มีกระแสไหลผ่านสเตปปิ้งมอเตอร์ตลอดเวลาเช่นกัน จะทำให้สเตปปิ้งมอเตอร์ร้อนขึ้นเรื่อย ๆ จึงต้องทำการเคลียร์ข้อมูลที่ U22

U24 ULN2003A เป็นตัวขับทำหน้าที่ในการจ่ายกระแสให้แก่ขดลวดของสเตปปิ้งมอเตอร์

การทำงานเริ่มจากการส่งข้อมูลไปที่พอร์ท XX7H ต่อเนื่องเป็นลำดับกันไป ซึ่งข้อมูลที่ส่งออกไปจะเป็นอะไรก็แล้วแต่ความต้องการที่จะให้สเตปปิ้งมอเตอร์หมุนในลักษณะไหน

3.2.11 วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

ในส่วนของวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกนั้นเป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำคอมพิวเตอร์ไปควบคุมวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกเพื่อนำข้อมูล (Data) จากคอมพิวเตอร์มาแปลงเป็นสัญญาณแอนาลอก ซึ่งแสดงได้ดังรูปที่ 3.12



รูปที่ 3.12 วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

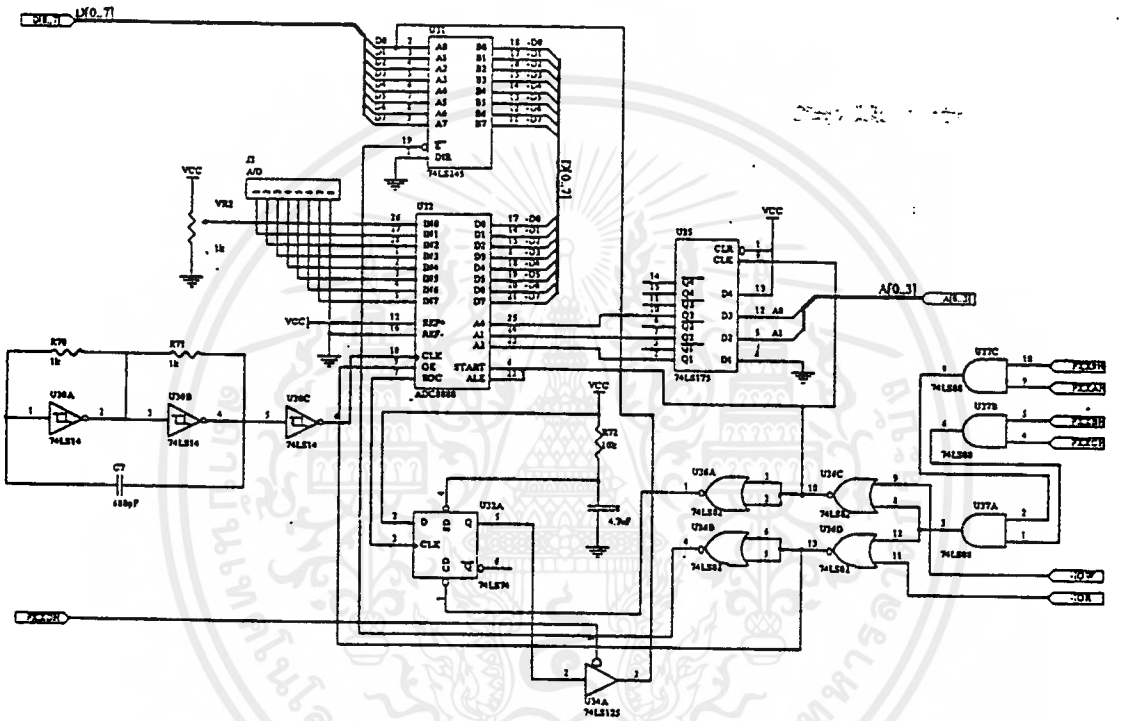
จากวงจรประกอบด้วย U26 ไอซี 74LS273, U27 ไอซี MC1408 , U28 ไอซี LM741 ประกอบกับ R-C เพื่อทำหน้าที่เป็นวงจรเปลี่ยนสัญญาณจาก ดิจิตอลเป็นแอนาลอก ซึ่งวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกในวงจรนี้มีทั้งหมด 4 ชุดแต่ละชุดถูกดีโค้ดไว้ที่พอร์ท XX8H วงจรข้างต้นจะให้สัญญาณเอาท์พุทเป็นระดับแรงดันที่เปลี่ยนแปลงไปตามการเปลี่ยนแปลงของสัญญาณแอนาลอกที่เข้ามาที่อินพุทของวงจร

U26 จะทำหน้าที่ค้างข้อมูล ไว้ที่ Q1 -Q8 จากนั้น U27 จะทำการแปลงข้อมูลนี้ให้เป็นสัญญาณแอนาลอกซึ่งอยู่ในรูปของกระแส ปรากฏที่ขา 4 แล้ว U28 จะทำการเปลี่ยนกระแสนี้ให้อยู่ในรูปของแรงดันส่งออกไปที่จัมเปอร์ JM1

สัญญาณจากจัมเปอร์ JM1 จะต่อเข้ากับวงจรขับหลอดไฟขนาด 12 โวลท์ เพื่อให้เห็นความสัมพันธ์ระหว่างสัญญาณดิจิตอลที่เข้ามากับสัญญาณแอนาลอกที่ได้ โดยการใช้มัลติมิเตอร์ในการวัดค่า

3.2.12 วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล

ในส่วนของวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอลนั้น เป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการนำคอมพิวเตอร์มาควบคุมวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล เพื่อนำสัญญาณแอนาลอกมาแปลงเป็นสัญญาณดิจิตอลและส่งให้เครื่องคอมพิวเตอร์ประมวลผลต่อไป ซึ่งการสร้างและการทำงานของวงจรสามารถอธิบายได้ดังรูปที่ 3.13



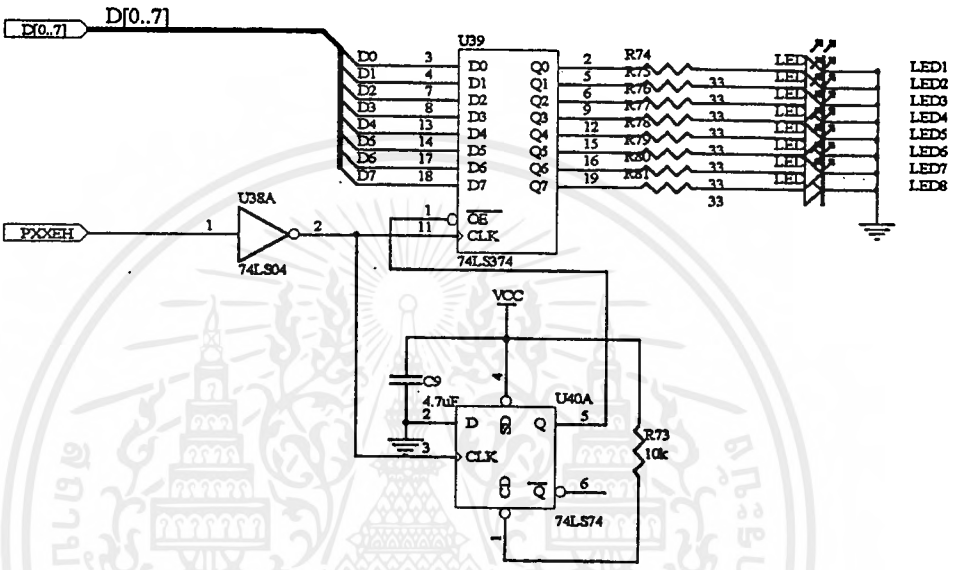
รูปที่ 3.13 วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล

จากวงจรจะทำการต่อ VREF (+) กับ VREF (-) ลงกราวด์ เพื่อเป็นตัวกำหนดแรงดันอ้างอิงให้กับวงจรภายในของ U32 ADC0808 ขา CLK ต่อกับวงจรกำเนิดสัญญาณนาฬิกา ขาของ วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอลต่อกับ U31 ไอซี 74LS245 ขา A0-A2 ต่อกับ U35 ไอซี 74LS175

การทำงานของวงจรเริ่มจากการส่งข้อมูลกำหนดแอดเดรส เพื่อกำหนดแชนแนลของวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล ว่าจะรับข้อมูลอนาลอกจากแชนแนลไหน

3.2.13 วงจรควบคุมสัญญาณไฟจราจร

ส่วนของวงจรควบคุมสัญญาณไฟจราจรนั้นจะมีแนวคิดการออกแบบเหมือนกับวงจรเบสิกเอาท์พุท คือเป็นหลักการเบื้องต้นในการนำคอมพิวเตอร์ไปควบคุมอุปกรณ์เอาท์พุท การสร้างและการทำงานของวงจรสามารถอธิบายได้ดังรูปที่ 3.14



รูปที่ 3.14 วงจรควบคุมสัญญาณไฟจราจร

จากรูปที่ 3.14 จะเห็นว่าเป็นวงจรลักษณะเดียวกับวงจรเบสิกเอาท์พุท ในการทดลองเพียงแต่นำเอาไดโอดเปล่งแสงมาจัดเรียงตำแหน่งใหม่ให้เป็นลักษณะของสัญญาณไฟจราจร แล้วควบคุมจังหวะการส่งข้อมูลไปยังไดโอดเปล่งแสงนั้น ให้สัมพันธ์กับลักษณะการติดดับของไฟจราจร

วงจรจะประกอบด้วย U39 ไอซี 74LS374 ทำหน้าที่เป็นตัวรับข้อมูลจากบัสข้อมูล ที่ D0-D7 แล้วส่งข้อมูลไปยังไดโอดเปล่งแสงที่ Q0-Q7 ไอซี 74LS374 ถูกดีโค้ดไว้ที่พอร์ท XXEH ไอซี U40A เบอร์ 74LS74 ทำหน้าที่เป็นตัวทำให้เอาท์พุท Q0-Q7 ของ U39 อยู่ในสภาวะ HIGH IMPEDANCE ในจังหวะเริ่มเปิดไฟเข้าเครื่องครั้งแรก

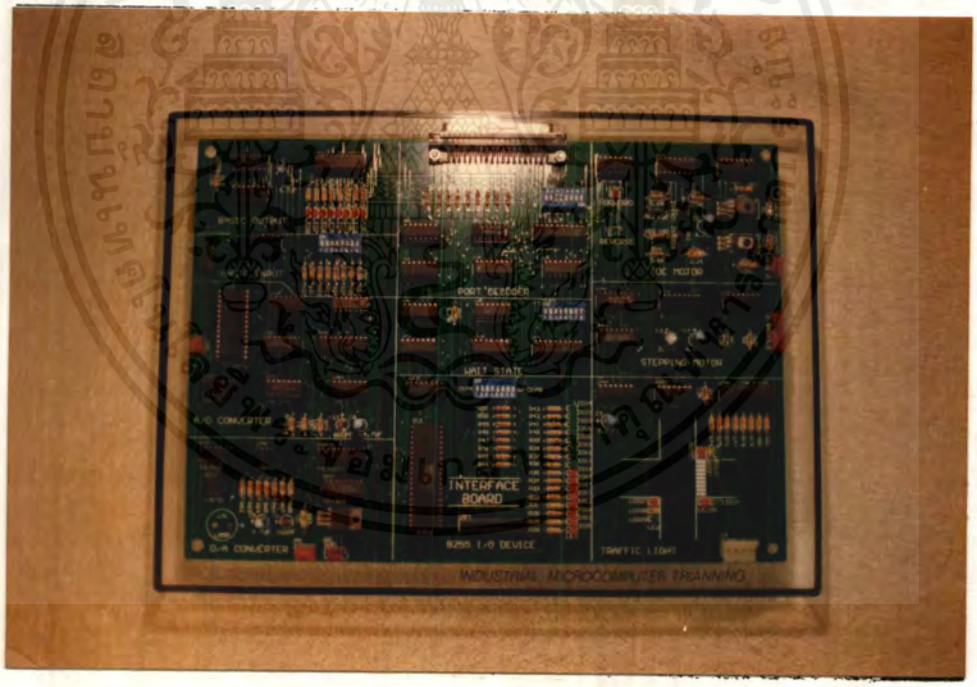
3. 3 ส่วนของแหล่งจ่ายไฟ

เป็นชุดประกอบสำเร็จ ซึ่งเป็นแหล่งจ่ายไฟประเภทสวิทช์ชิ่งเพาเวอร์ซัพพลาย (Switching Power Supply) อย่างเดียวกับที่ใช้ใน IBM PC ทั่วไปซึ่งสามารถจ่ายแรงดันไฟฟ้า +5 ,-5,+12,-12 โวลท์

3.4 การสร้างชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

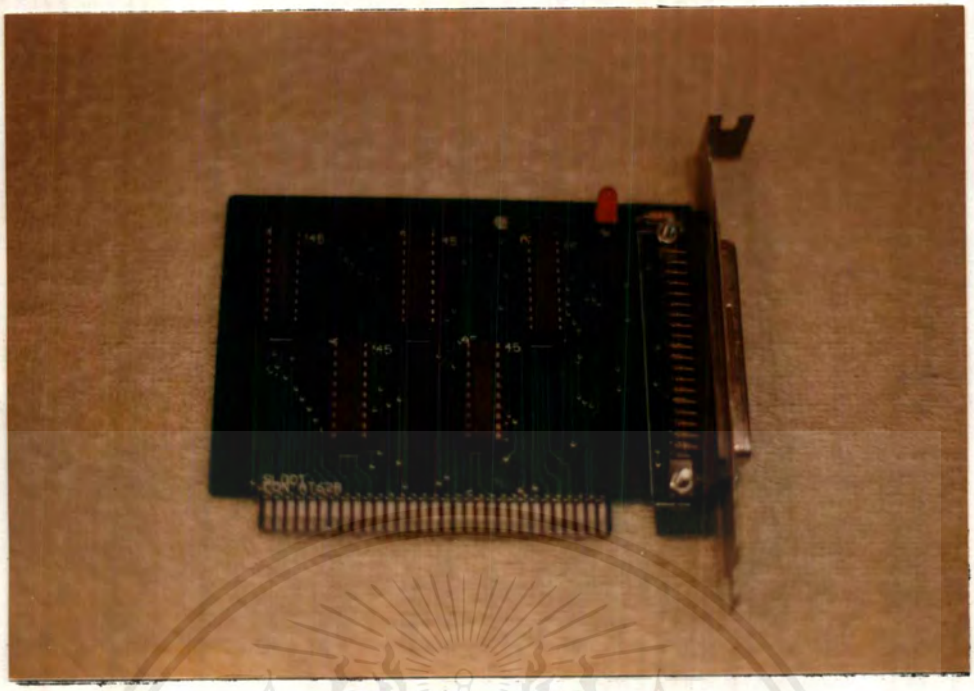
การประกอบชุดต่าง ๆ เข้าด้วยกันตลอดจนลักษณะโดยรวมของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรมมีรูปแบบแสดงให้เห็นดังต่อไปนี้

ในรูปที่ 3.15 เป็นการประกอบวงจรต่างๆ ลงบนแผ่นวงจรพิมพ์แผ่นเดียวกัน ซึ่งในโครงงานนี้เราจะเรียกในส่วนนี้ว่า ชุดอินเตอร์เฟสบอร์ดของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม



รูปที่ 3.15 ชุดอินเตอร์เฟสบอร์ดของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

ในรูปที่ 3.16 รูปแสดงแผงวงจรของอะแดปเตอร์การ์ดที่ใช้ในการควบคุมการติดต่อระหว่างคอมพิวเตอร์กับวงจรอินเตอร์เฟสภายนอก และจะทำหน้าที่เป็นตัวบัฟเฟอร์ข้อมูลด้วย รูปของแผงวงจรอะแดปเตอร์การ์ดสามารถแสดงได้ดังรูปที่ 3.16

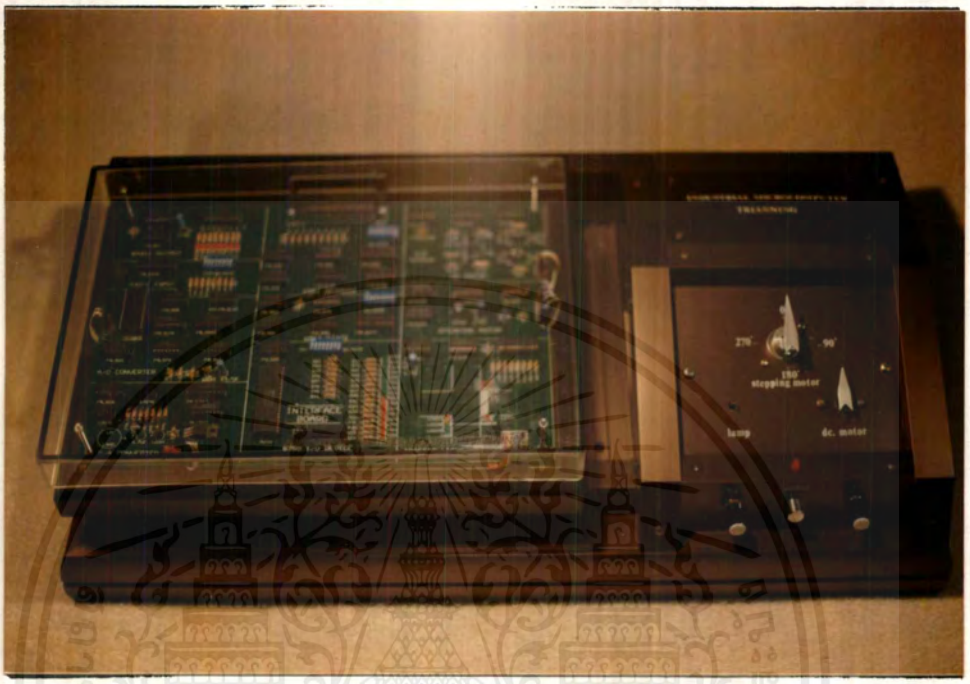


รูปที่ 3.16 อะแดปเตอร์การ์ด

อีกส่วนหนึ่งที่อยู่บนชุดฝึกก็คือ ส่วนที่ใช้ในการแสดงผลการทำงานของ สเตปปิ้งมอเตอร์ ดีซีมอเตอร์ และหลอดไฟ ซึ่งในส่วนที่กล่าวมานี้จะอยู่ทางด้านขวามือของผู้ใช้งาน ในรูปที่ 3.17 จะเป็นรูปที่แสดงถึงส่วนแสดงผลเหล่านี้

รูปที่ 3.17 สเตปปิ้งมอเตอร์ ดีซีมอเตอร์ และหลอดไฟ

ในรูปที่ 3.18 นี้เป็นการแสดงลักษณะโดยสมบูรณ์ของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม ซึ่งแสดงได้ดังนี้



รูปที่ 3.18 ลักษณะโดยสมบูรณ์ของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

บทที่ 4

การทดลองและผลการทดลอง

ในบทนี้จะเป็นการทดลอง วงจรส่วนต่าง ๆ ของอินเตอร์เฟสบอร์ด (Interface board) เช่น วงจรเบสิกอินพุท , ควบคุมอุปกรณ์ภายนอกเบื้องต้น เป็นต้น ด้วยโปรแกรมภาษาซี ซึ่งจะทำการทดสอบส่วนต่าง ๆ ดังนี้

4.1 การทดสอบวงจรโดยใช้โปรแกรมบนเครื่องคอมพิวเตอร์

ลำดับขั้นการทดลอง

1. นำอะแดปเตอร์การ์ดไปเสียบในสล็อตของคอมพิวเตอร์
2. ต่อสายแพรเข้ากับคอนเน็กเตอร์ DB37 ระหว่างอะแดปเตอร์การ์ดกับอินเตอร์เฟสบอร์ด
3. ทำการดีโค้ด (Decode) ส่วนของอินเตอร์เฟสบอร์ดให้ตรงกับตำแหน่งของแอดเดรสที่ต้องการ
4. เปิดสวิทช์แหล่งจ่ายไฟของ อินเตอร์เฟสบอร์ด
5. ทำการเขียนโปรแกรมภาษาซี เพื่อทดสอบวงจรส่วนต่าง ๆ ดังนี้

หมายเหตุ

โปรแกรมการทดสอบดังต่อไปนี้จะใช้ตำแหน่งของการดีโค้ด(Decode) ที่ 300H-30FH

4.1.1 การทดสอบวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

ในการทดลองนี้เป็นการทดสอบการส่งสัญญาณข้อมูลไปยังวงจรควบคุมอุปกรณ์ภาย-นอกเบื้องต้นเพื่อแสดงผลการติด-ดับของไดโอดเปล่งแสง (LED) โดยถ้ามีการส่งค่าข้อมูลมายังพอร์ท 300H และยังคงมีข้อมูลค้างอยู่จะทำให้ไดโอดเปล่งแสงติดอยู่เช่นนั้น ซึ่งมีลำดับขั้นการทดลองดังต่อไปนี้

1.ป้อนโปรแกรมการทดสอบวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น ซึ่งวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้นต่ออยู่กับพอร์ท 300H ทำการป้อนโปรแกรมทดสอบที่ 4.1 ดังนี้

โปรแกรมทดสอบที่ 4.1 วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

```
#include < stdio.h>
#include < dos.h>
main( )
{
 int  a= 0xff ;
 outportb(0X300, a);
}
```

2. ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.1

โปรแกรมทดสอบที่	ผลการทดลอง
4.1วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น	จากการทดลองวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้นผลที่ได้คือไดโอดเปล่งแสงติดทุกดวง

ตารางที่ 4.1 ผลการทดลองโปรแกรมทดสอบที่ 4.1 วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

4.1.2 การทดสอบวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

การทดลองนี้เป็นการการรับข้อมูลจากวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น เข้ามายังคอมพิวเตอร์ เพื่อแสดงค่าของข้อมูลเป็นเลขฐานสิบและเลขฐานสิบหกออกทางหน้าจอคอมพิวเตอร์ โดยต้องทำการปรับดิฟสวิทช์ 3 (DIP SW. 3) ตั้งแต่ตัวที่ 1-8 ไปที่ตำแหน่ง 0ff ก่อนที่จะทำการคอมไพล์โปรแกรม เมื่อทำการปรับดิฟสวิทช์ 3 ผลที่จะได้รับก็คือ ค่าของข้อมูลจากดิฟสวิทช์ 3 จะถูกส่งผ่านทางพอร์ท 301H ไปดีโค้ดที่วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น และจะแสดงตำแหน่งของ off ออกมาทางจอคอมพิวเตอร์เป็นเลขฐานสิบและเลขฐานสิบหก ซึ่งในการทดลองมีลำดับขั้นตอนดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น ซึ่งวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น ต่ออยู่กับพอร์ท 301H ทำการป้อนโปรแกรมทดสอบที่ 4.2 ดังนี้

โปรแกรมทดสอบที่ 4.2 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a ;
clrscr( );
a = inportb(0x301);
printf(" Data = %dD  or %%xH " ,a,a);
getch( );
}
```

2. ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9
3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.2

โปรแกรมทดสอบที่	ผลการทดลอง
4.2 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น	ผลจะปรากฏที่หน้าจอคอมพิวเตอร์ดังนี้ Data = 255D or ffH

ตารางที่ 4.2 ผลการทดลองโปรแกรมทดสอบที่ 4.2 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

4.1.3 การทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0

การทดลองนี้เป็นการทดลองการส่งข้อมูลระหว่างคอมพิวเตอร์กับวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0 เพื่อแสดงผลการติด - ดับของไดโอดเปล่งแสง ซึ่งในโหมด 0 นี้การใช้งาน 8255 จะต้องส่งรหัสควบคุม (Control Word)ให้แก่ 8255 ก่อน โดยพอร์ท A ,พอร์ท B เป็นเอาท์พุทพอร์ท และพอร์ท C เป็นอินพุทพอร์ท เมื่อมีการส่งข้อมูลผ่านทางพอร์ท A และ พอร์ท B ผลที่จะได้คือจะทำให้ไดโอดเปล่งแสงของวงจร (LED9-LED24)ติดทุกดวง ซึ่งลำดับขั้นตอนในการทดลองมีดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0 ซึ่งวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0 ต่ออยู่กับพอร์ทต่างๆ ดังนี้

พอร์ท A ของ 8255 อยู่ที่ 304H
พอร์ท B ของ 8255 อยู่ที่ 305H
พอร์ท C ของ 8255 อยู่ที่ 302H
พอร์ทควบคุม ของ 8255 อยู่ที่ 303H

ป้อนโปรแกรมทดสอบที่ 4.3 ดังนี้

โปรแกรมทดสอบที่ 4.3 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a = 0xff;
  outportb(0x303,0x89);
  outportb(0x305,a);
  outportb(0x304,a);
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.3

โปรแกรมทดสอบที่	ผลการทดลอง
4.3 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0	ผลที่ได้คือไดโอดเปล่งแสง ของวงจร ติดทุกดวง (LED9 - LED24 ติดทุกดวง)

ตารางที่ 4.3 ผลการทดลองโปรแกรมทดสอบที่ 4.3 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 0

4.1.4 การทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1

การทดลองนี้เป็นการทดลองการส่งข้อมูลระหว่างคอมพิวเตอร์กับวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1 เพื่อแสดงผลการติด - ดับของไดโอดเปล่งแสง ซึ่งในโหมด 1 นี้การใช้งาน 8255 จะต้องส่งรหัสควบคุม (Control Word)ให้แก่ 8255 ก่อน โดยพอร์ท A ,พอร์ท B เป็นพอร์ทข้อมูล และพอร์ท C เป็นแฮนด์เช็ค เมื่อมีการส่งข้อมูลผ่านทางพอร์ท A และ พอร์ท B ผลที่จะได้คือจะทำให้ไดโอดเปล่งแสงของวงจร (LED9-LED24)ติดทุกดวง ซึ่งลำดับขั้นตอนในการทดลองมีดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1 ซึ่งวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1 ต่ออยู่กับพอร์ทต่างๆ ดังนี้

พอร์ท A ของ 8255 อยู่ที่ 304H

พอร์ท B ของ 8255 อยู่ที่ 305H

พอร์ท C ของ 8255 อยู่ที่ 302H

พอร์ทควบคุม ของ 8255 อยู่ที่ 303H

ป้อนโปรแกรมทดสอบที่ 4.4 ดังนี้

โปรแกรมทดสอบที่ 4.4 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a = 0xff;
  outportb(0x303,0xa0);
  outportb(0x305,a);
  outportb(0x304,a);
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.4

โปรแกรมทดสอบที่	ผลการทดลอง
4.4 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1	ผลที่ได้คือไดโอดเปล่งแสง ของวงจร ติดทุกดวง (LED9 - LED24 ติดทุกดวง)

ตารางที่ 4.4 ผลการทดลองโปรแกรมทดสอบที่ 4.4 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 1

4.1. 5 การทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2

การทดลองนี้เป็นการทดลองการส่งข้อมูลระหว่างคอมพิวเตอร์กับวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2 เพื่อแสดงผลการติด - ดับของไดโอดเปล่งแสง ซึ่งในโหมด 2 นี้การใช้งาน 8255 จะต้องส่งรหัสควบคุม (Control Word)ให้แก่ 8255 ก่อน โดยพอร์ท A เป็นพอร์ทข้อมูลแบบสองทิศทาง ในการทดลองนี้จะให้พอร์ท A เป็นเพอร์ทเอาท์พุท เมื่อมีการส่งข้อมูลผ่านทางพอร์ท A ผลที่จะได้คือจะทำให้ไดโอดเปล่งแสงของวงจร (LED9-LED16)ติดทุกดวง ซึ่งลำดับขั้นตอนในการทดลองมีดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2 ซึ่งวงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2 ต่ออยู่กับพอร์ทต่างๆ ดังนี้

พอร์ท A ของ 8255 อยู่ที่ 304H

พอร์ทควบคุม ของ 8255 อยู่ที่ 303H

ป้อนโปรแกรมทดสอบที่ 4. 5 ดังนี้

โปรแกรมทดสอบที่ 4.5 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a = 0xff;
  outportb(0x303,0xc0);
  outportb(0x305,a);
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.5

โปรแกรมทดสอบที่	ผลการทดลอง
4.5 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2	ผลที่ได้คือไดโอดเปล่งแสง ของวงจร ติดทุกดวง (LED9 - LED16 ติดทุกดวง)

ตารางที่ 4.5 ผลการทดลองโปรแกรมทดสอบที่ 4.5 วงจรการควบคุมอุปกรณ์อินพุท/เอาท์พุทโดยใช้ 8255 ในโหมด 2

4.1.6 การทดสอบวงจรดีซีมอเตอร์

การทดลองนี้เป็นการทดลองควบคุมทิศทางการหมุนของดีซีมอเตอร์ว่าต้องการให้หมุนแบบตามเข็มนาฬิกาหรือแบบทวนเข็มนาฬิกา โดยใช้คอมพิวเตอร์เขียนโปรแกรมสั่งการเพื่อให้ดีซีมอเตอร์หมุน แต่ถ้ามีการกดคีย์ใดๆ ดีซีมอเตอร์จะหยุดหมุนทันที ซึ่งมีลำดับขั้นการทดลองดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรดีซีมอเตอร์ ซึ่งวงจรดีซีมอเตอร์ต่ออยู่กับพอร์ท 306H ทำการป้อนโปรแกรมทดสอบที่ 4.6 ดังนี้

โปแกรมทดสอบที่ 4.6 วงจรดีซีมอเตอร์

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a= 0x02,c=0x00;
clrscr( );
outportb(0x306,a);
printf(" Press enter to stop DC motor " );
getch( );
outportb(0x306,c); }
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.6

โปรแกรมทดสอบที่	ผลการทดลอง
4.6 วงจรดีซีมอเตอร์	ผลที่ได้คือดีซีมอเตอร์จะหมุนแบบตามเข็มนาฬิกา แต่เมื่อมีการกดคีย์ใดๆ บนคีย์บอร์ดทำให้มอเตอร์หยุดหมุนทันที

ตารางที่ 4.6 ผลการทดลองโปรแกรมทดสอบที่ 4.6 วงจรดีซีมอเตอร์

4.1.7 การทดสอบวงจรสเต็ปปิ้งมอเตอร์

การทดลองนี้เป็นการทดลองควบคุมทิศทางและตำแหน่งการหมุนของสเต็ปปิ้งมอเตอร์ โดยใช้คอมพิวเตอร์เขียนโปรแกรมสั่งการเพื่อให้สเต็ปปิ้งมอเตอร์หมุนตามทิศทางและตำแหน่งที่กำหนดในโปรแกรม แต่ถ้ามีการกดคีย์ใดๆ สเต็ปปิ้งมอเตอร์จะหยุดหมุนทันที ซึ่งมีลำดับขั้นการทดลองดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรสเต็ปปิ้งมอเตอร์ ซึ่งต่ออยู่กับพอร์ท 307H ป้อนโปรแกรมทดสอบที่4.7 ดังนี้

โปรแกรมทดสอบที่ 4.7 วงจรสเต็ปปิ้งมอเตอร์

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a= 0x01, b=0x02, c=0x00, d=0x08, e=100, f=0x307;
 clrscr( );
 printf(" Press  anykey  to  stop the  Stepping  motor " );
 while(!kbhit( ) ) {
    outportb( f ,a);
```

```
                delay( e);
                outportb( f ,b);
                delay( e);
                outportb( f ,c);
                delay( e);
        outportb( f ,d);
        }
}
```

2. ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.7

โปรแกรมทดสอบที่	ผลการทดลอง
4.7 วงจรสเต็ปปิ้งมอเตอร์	ในการทดลองสเต็ปปิ้งมอเตอร์หมุนผิดปกติ

ตารางที่ 4.7 ผลการทดลองโปรแกรมทดสอบที่ 4.7 วงจรสเต็ปปิ้งมอเตอร์

การแก้ไข

ทำการตรวจสอบความผิดพลาดของวงจรโดยพบความผิดพลาดเป็นที่ตัวของสเตปปิ้งมอเตอร์ จึงได้ทำการเปลี่ยนสเตปปิ้งมอเตอร์แล้วทำการทดสอบอีกครั้งและผลที่ได้คือสเต็ปปิ้งมอเตอร์จะหมุนตามทิศทางและตำแหน่งที่กำหนดให้ในโปรแกรม และเมื่อมีการกดคีย์ใด ๆ บนคีย์บอร์ดของคอมพิวเตอร์ก็จะทำให้สเต็ปปิ้งมอเตอร์หยุดหมุน

4.1.8 การทดสอบวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

การทดลองนี้เป็นการทดลองการแปลงสัญญาณข้อมูลจากคอมพิวเตอร์ผ่านทางวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก (D / A) ให้เป็นสัญญาณแรงดันทางไฟฟ้าไปควบคุมการติด-ดับของหลอดไฟ โดยการส่งข้อมูลจะกระทำผ่านทางพอร์ท 308H ซึ่งมีลำดับขั้นตอนในการทดลองดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก ซึ่งต่ออยู่กับพอร์ท 308H ป้อนโปรแกรมทดสอบที่ 4. 8 ดังนี้

โปแกรมทดสอบที่ 4.8 วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a= 0xff, b=0x00;
clrscr( );
outportb(0x308,a);
printf(" Press anykey to close the light. " );
getch( );
outportb(0x308,b);
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.8

โปรแกรมทดสอบที่	ผลการทดลอง
4.8วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก	ผลที่ได้คือ หลอดไฟจะติดสว่าง และเมื่อกดคีย์ใดๆ บนคีย์บอร์ดหลอดไฟก็จะดับ

ตารางที่ 4.8 ผลการทดลองโปรแกรมทดสอบที่ 4.8 วงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

4.1.9 การทดสอบวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล

การทดลองนี้เป็นการทดลองการแปลงสัญญาณข้อมูลจากการปรับวอลุ่ม (VR2)ผ่านทางวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล(A/D) ให้เป็นสัญญาณข้อมูลเพื่อส่งให้คอมพิวเตอร์และคอมพิวเตอร์จะทำการแสดงผลออกทางจอภาพเป็นเลขฐานสิบและเลขฐานสิบหก โดยทำการคอมไพล์โปรแกรมก่อน แล้วจึงทำการปรับวอลุ่ม และจะสังเกตุเห็นการเปลี่ยนแปลงของข้อมูลบนจอคอมพิวเตอร์ ถ้าปรับวอลุ่มมาทางซ้ายมือข้อมูลจะมีค่าเป็นศูนย์ทั้งเลขฐานสิบและเลขฐานสิบหก แต่ถ้ามีการปรับวอลุ่มมาทางขวามือค่าของข้อมูลจะเพิ่มขึ้นเรื่อย ๆ ซึ่งลำดับขั้นตอนในการทดลองมีดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล ซึ่งจะต่ออยู่กับพอร์ท 30CH ป้อนโปรแกรมทดสอบที่ 4. 9 ดังนี้

โปรแกรมทดสอบที่ 4.9 วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล

```
#include < stdio.h>
#include < dos.h>
#include < conio.h>
main( )
{ int  a=0, b;
 clrscr( );
outportb(0x30c, 0x00);
```

```
 while(!kbhit( ) )
 b=inportb(0x30d);
 if (b!=254) {
 b=inportb(0x30c);
gotoxy(20,10);
 printf(" Data = %dD  or %xH    " ,b,b);
     delay( 100);
   }
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.9

โปรแกรมทดสอบที่	ผลการทดลอง
4.9 วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล	ผลที่ได้คือ เมื่อปรับวอลุ่ม (VR2) จะสังเกตุเห็นการเปลี่ยนแปลงจากจอคอมพิวเตอร์ เช่น เมื่อปรับ VR2 ไปทางซ้ายสุดแล้วที่หน้าจอคอมพิวเตอร์จะแสดงผลคือ Data = 0D or 0H และเมื่อปรับ VR2 ไปทางขวาทีละน้อยตัวเลขก็จะค่อย ๆ เพิ่มขึ้น

ตารางที่ 4.9 ผลการทดลองโปรแกรมทดสอบที่ 4.9 วงจรแปลงสัญญาณแอนาลอกเป็นสัญญาณดิจิตอล

4.1.10 การทดสอบวงจรควบคุมสัญญาณไฟจราจร

การทดลองนี้เป็นการทดลองซึ่งคล้ายกับวงจรเบสิกเอาท์พุทคือเป็นการส่งข้อมูลจากคอมพิวเตอร์เพื่อไปแสดงผลที่ไดโอดเปล่งแสง โดยผ่านทางพอร์ท 30eH ซึ่งมีลำดับขั้นตอนการทดลองดังต่อไปนี้

1. ป้อนโปรแกรมการทดสอบวงจรควบคุมสัญญาณไฟจราจร ซึ่งต่ออยู่กับพอร์ท 30eH ป้อนโปรแกรมทดสอบที่ 4.10 ดังนี้

โปรแกรมทดสอบที่ 4.10 วงจรควบคุมสัญญาณไฟจราจร

```
#include < stdio.h>
#include < dos.h>
main( )
{
 int  a= 0xff ;
outportb(0x30e, a);
}
```

2.ทำการคอมไพล์โปรแกรมโดยการกดคีย์ Ctrl+F9

3. สังเกตุผลและบันทึกผลการทดลองลงตารางที่ 4.10

โปรแกรมทดสอบที่	ผลการทดลอง
4.10 วงจรควบคุมสัญญาณไฟจราจร	ผลที่ได้คือ ไดโอดเปล่งแสง (LED27 - LED34)ติดทุกดวง

ตารางที่ 4.10 ผลการทดลองโปรแกรมทดสอบที่ 4.10 วงจรควบคุมสัญญาณไฟจราจร

4.2 สรุปผลการทดลอง

การติดต่อระหว่างคอมพิวเตอร์กับอุปกรณ์ภายนอก เราจะต้องทราบถึงตำแหน่งแอดเดรสของอุปกรณ์ภายนอกนั้นๆ เช่น วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้นออกแบบให้อยู่ที่ตำแหน่งแอดเดรส 300H และเมื่อทำการเขียนโปรแกรมส่งข้อมูลออกที่ตำแหน่ง 300H ผลที่ได้ก็ไปแสดงผลที่ส่วนของวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น ส่วนของวงจรอื่นๆก็เช่นกัน จะต้องทำการกำหนดตำแหน่งแอดเดรสของวงจรนั้น ๆ เสียก่อนที่จะส่งข้อมูลออกไป

บทที่ 5
บทสรุป วิจารณ์ และแนวทางในการพัฒนา

5.1 บทสรุป

ชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม (Industrial Microcomputer Training) ในปริญญานิพนธ์ฉบับนี้ สร้างขึ้นเพื่อการศึกษาและพัฒนาในส่วนของการใช้ซอร์ฟแวร์ควบคุมฮาร์ดแวร์ หรือการอินเตอร์เฟสระหว่างอุปกรณ์ภายนอกกับคอมพิวเตอร์ IBM PC

เมื่อพิจารณาถึงส่วนของฮาร์ดแวร์ในงานอุตสาหกรรมแล้ว ชุดฝึกนี้สามารถทดแทนอุปกรณ์ต่าง ๆ ในการใช้งานจริงอย่างกว้างขวางอีกทั้งยังสามารถเพิ่มอุปกรณ์ภายนอก ซึ่งต่อผ่านพอร์ทที่จัดเตรียมไว้ในบอร์ดอีกด้วย และเมื่อพิจารณาในส่วนของซอร์ฟแวร์ในการเขียนโปรแกรมเพื่อติดต่อกับอุปกรณ์ภายนอกนั้นผู้ฝึกจะได้รับทั้งประสบการณ์ในการเขียนโปรแกรมและเทคนิคในการเขียนโปรแกรมติดต่ออุปกรณ์ภายนอกอย่างมีประสิทธิภาพ

5.2 ปัญหาและแนวทางการแก้ไข

ในระหว่างทำโครงงานพิเศษในครั้งนี้ คณะผู้จัดทำได้พบกับปัญหาและอุปสรรคต่าง ๆ ที่ทำให้สูญเสียเวลาในการแก้ไขข้อบกพร่องผิดพลาดเหล่านั้น บางปัญหาอาจจะมองดูเล็กน้อย แต่ถ้าไม่จัดการแก้ไขก็จะทำให้เสียเวลามากในการแก้ไขในภายหลัง ซึ่งปัญหาต่าง ๆ และแนวทางในการแก้ไขแสดงให้เห็นดังตารางที่ 5.1 ดังต่อไปนี้

ปัญหา	แนวทางการแก้ไข
1. ในส่วนของการออกแบบและการสร้าง เนื่องจากเมื่อวงจรทั้งหมดรวมกันแล้วจะเป็นวงจรที่ใหญ่พอสมควร ดังนั้นในการออกแบบหลายวงจรจึงจำเป็นต้องใช้โปรแกรมคอมพิวเตอร์เข้าช่วย ในการออกแบบจึงเกิดความยุ่งยากในการใช้งาน	ควรศึกษาโปรแกรมคอมพิวเตอร์ ในการออกแบบให้ละเอียด รวมทั้งควรทดลองออกแบบจริง เพื่อเป็นการพบปัญหาที่แท้จริง
2. ในส่วนของฮาร์ดแวร์ เนื่องจากการนำทุกวงจรรวมเข้าด้วยกันบนแผ่นลายวงจรแผ่นเดียวและคอมพิวเตอร์ในปัจจุบันมีความเร็วเพิ่มสูงขึ้นมาก จึงเกิดปัญหาในเรื่องความผิดพลาดในการทดลอง	ควรทดลองออกแบบในส่วนของแต่ละวงจรให้สามารถ ทำงานได้อย่างถูกต้องเสียก่อนที่จะนำมารวมเป็นวงจรเดียวกัน

ตารางที่ 5.1 ปัญหาและแนวทางการแก้ไขปัญหาในการทำปริญญานิพนธ์

หมายเหตุ

วงจรทั้งหมดในชุดทดลองนี้ถูกออกแบบด้วยโปรแกรม Protel for window 1.5 ซึ่งโปรแกรมชุดนี้ยังมีข้อผิดพลาด (BUG) อยู่หลายจุด ดังนั้นจึงการตรวจสอบวงจรที่ได้ถูกออกแบบอย่างสมบูรณ์ด้วยโปรแกรมตัวนี้อีกครั้งอย่างถี่ถ้วน

ข้อควรระวัง

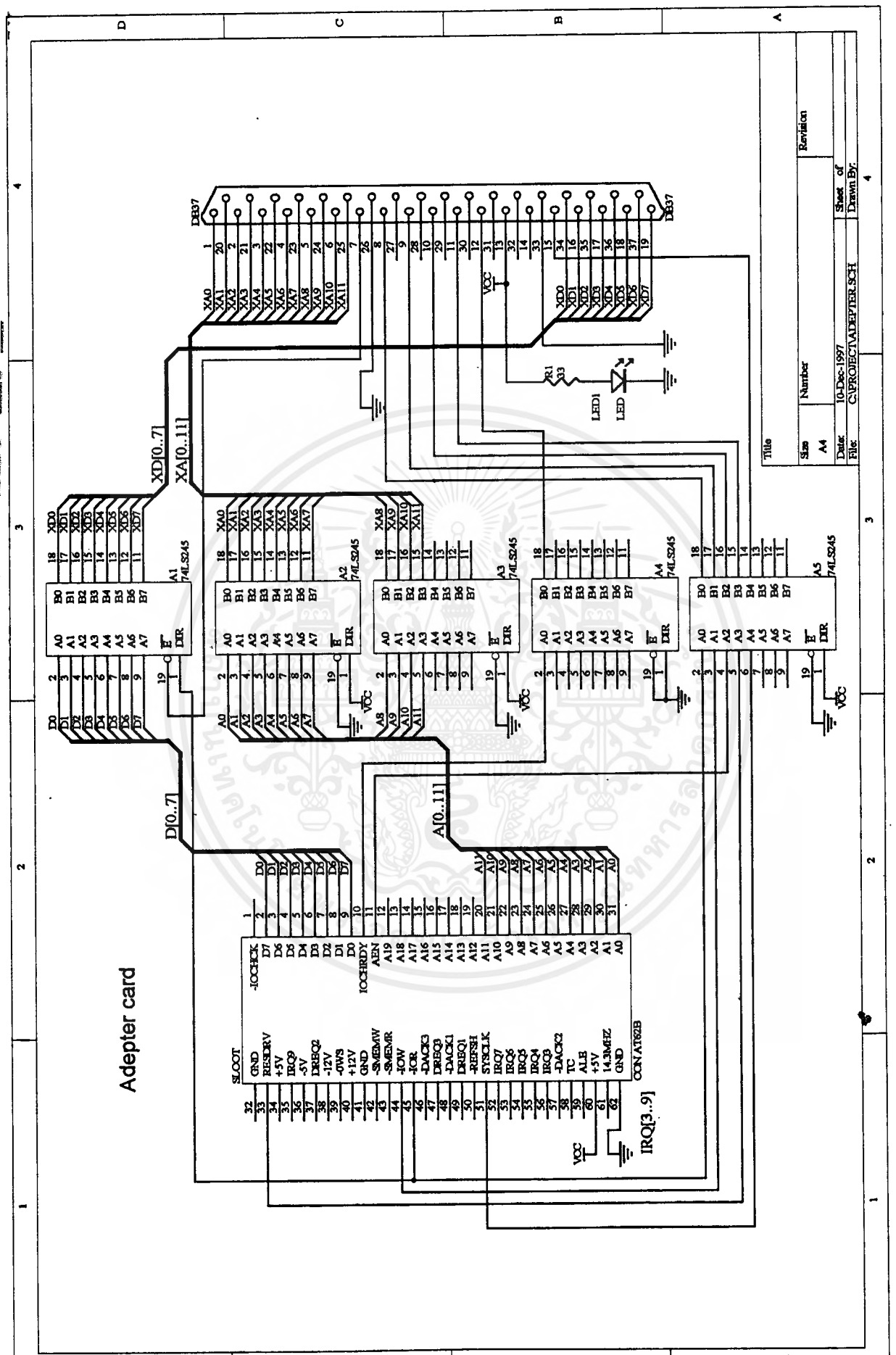
ในการออกแบบวงจรอะแดปเตอร์การ์ด (Adapter Card) นั้นควรระวังในส่วนของความถูกต้องของตำแหน่งของการ์ดที่เสียบลงในสล็อตเพราะว่า ถ้ามีการลัดวงจรเกิดขึ้นอาจทำให้วงจรของคอมพิวเตอร์เสียหายอย่างถาวรได้

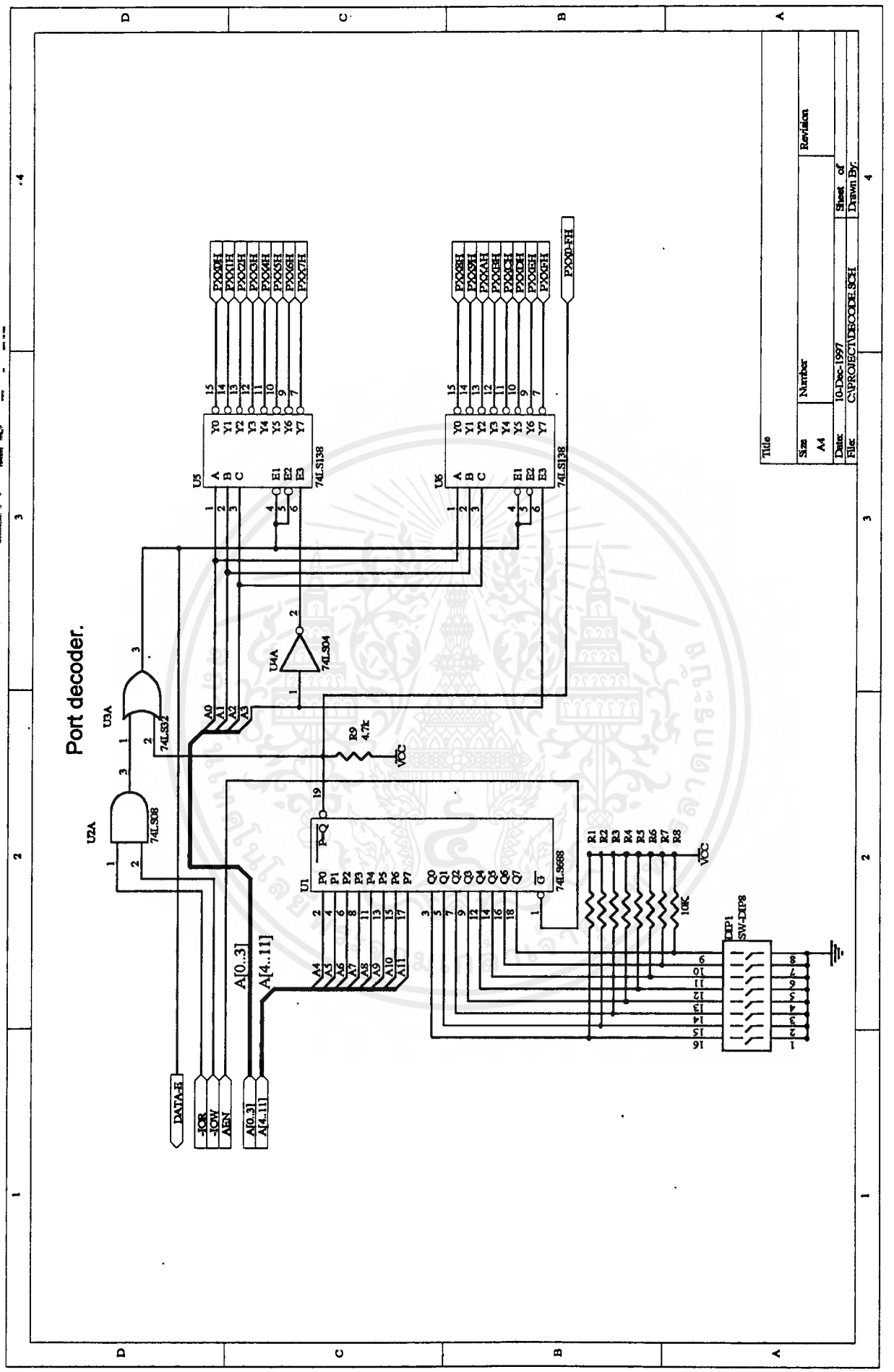
5.3 แนวทางในการพัฒนา

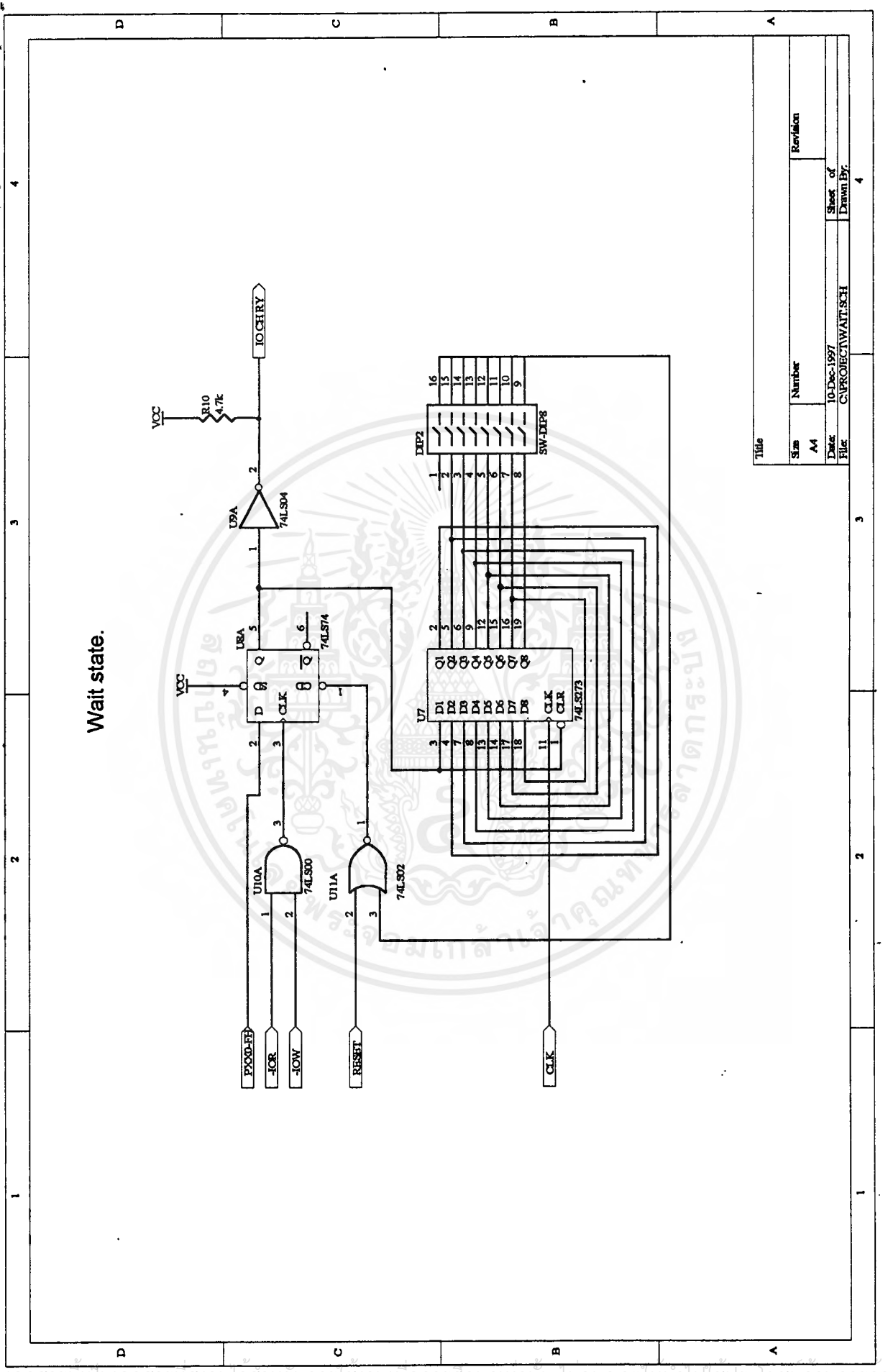
1.การปรับปรุงและเพิ่มเติมในส่วนของวงจร เพื่อให้เกิดความหลากหลายและการทำงานที่ดียิ่งขึ้น

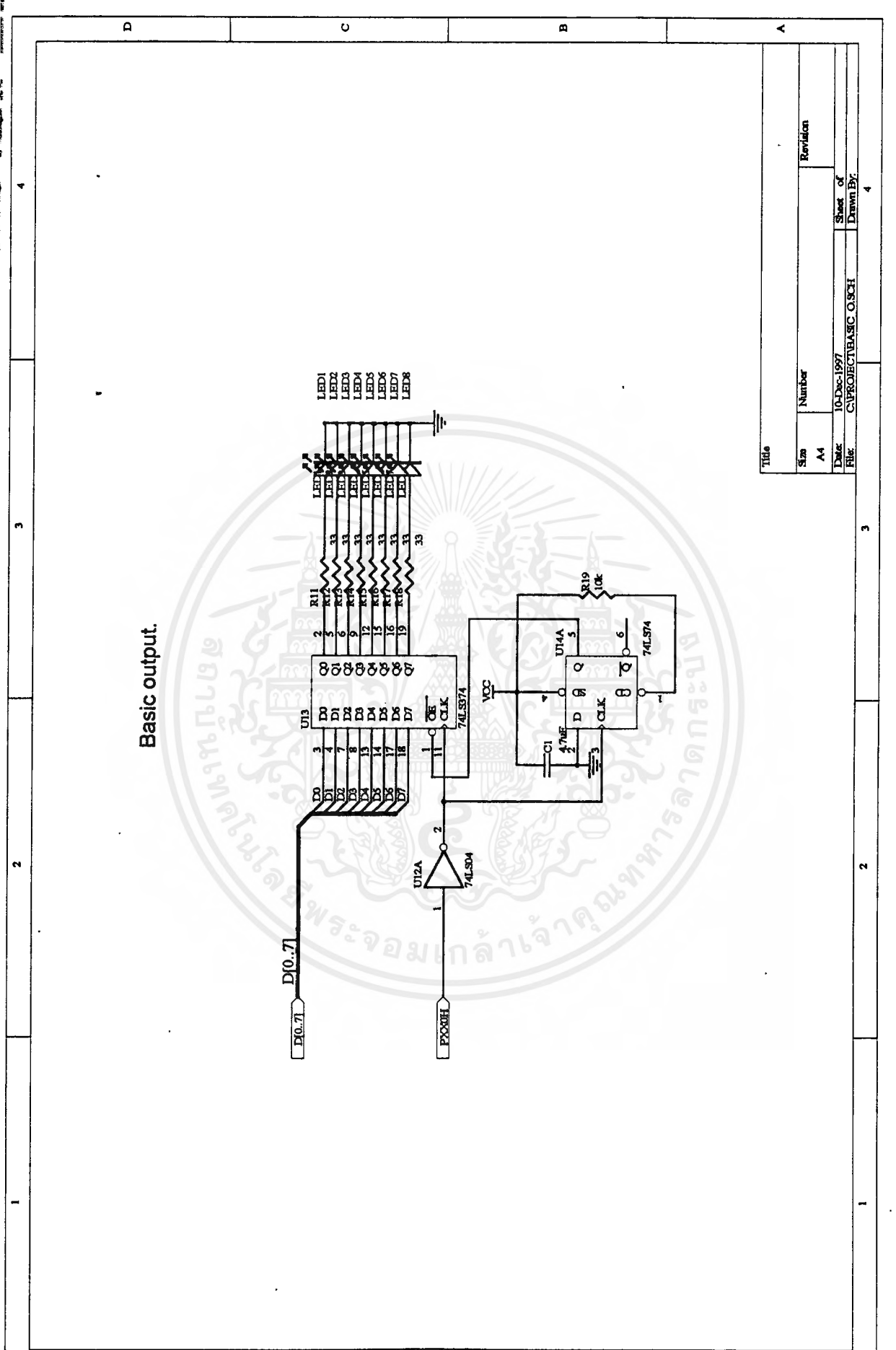
2.พัฒนาการใช้โปรแกรมด้วยการใช้โปรแกรมภาษาต่างๆ ในการเขียนโปรแกรมควบคุมเพื่อการศึกษาในโปรแกรมภาษาอื่นๆ ต่อไป

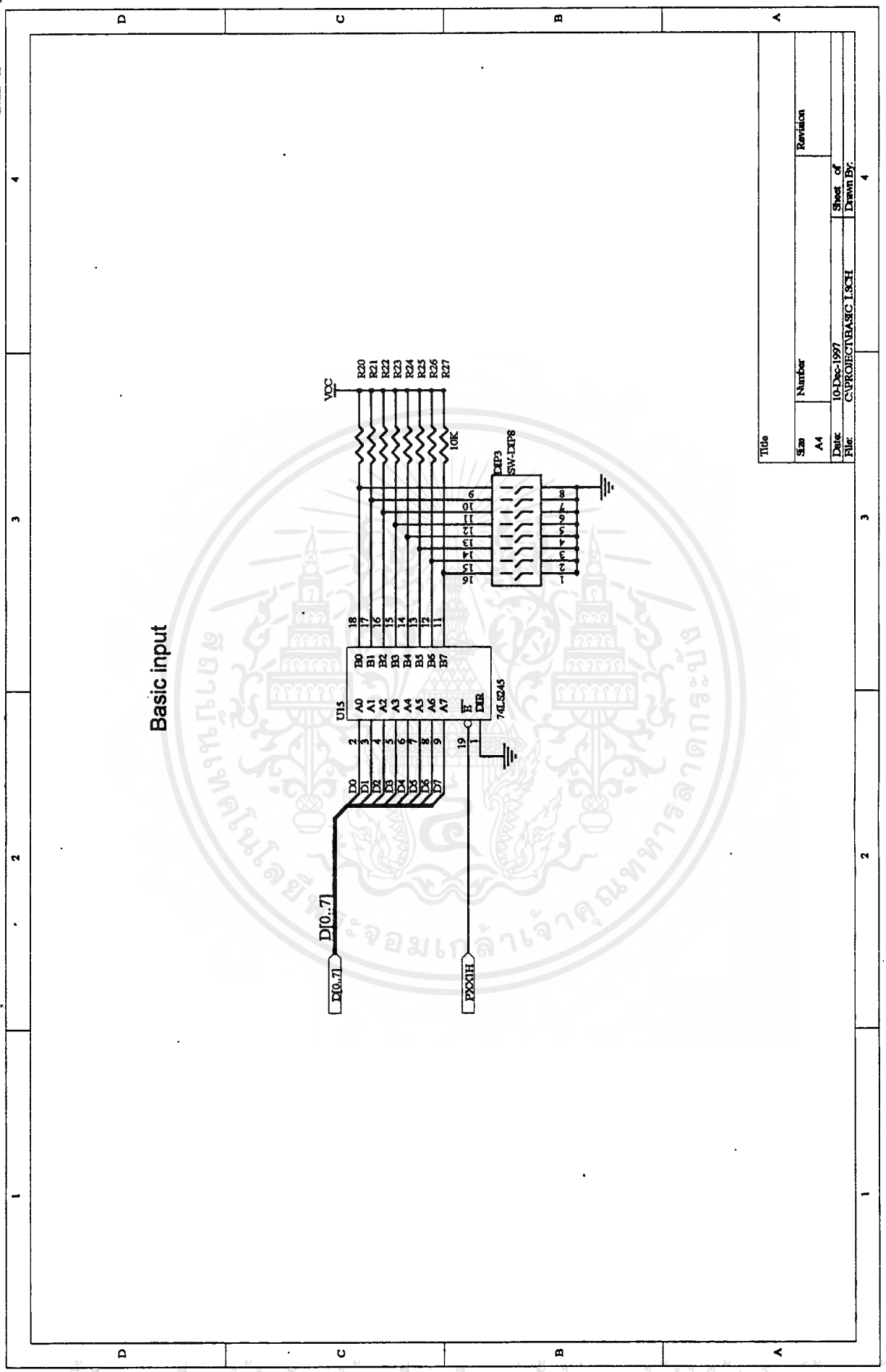
ภาคผนวก ก
วงจรส่วนต่าง ๆ ของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

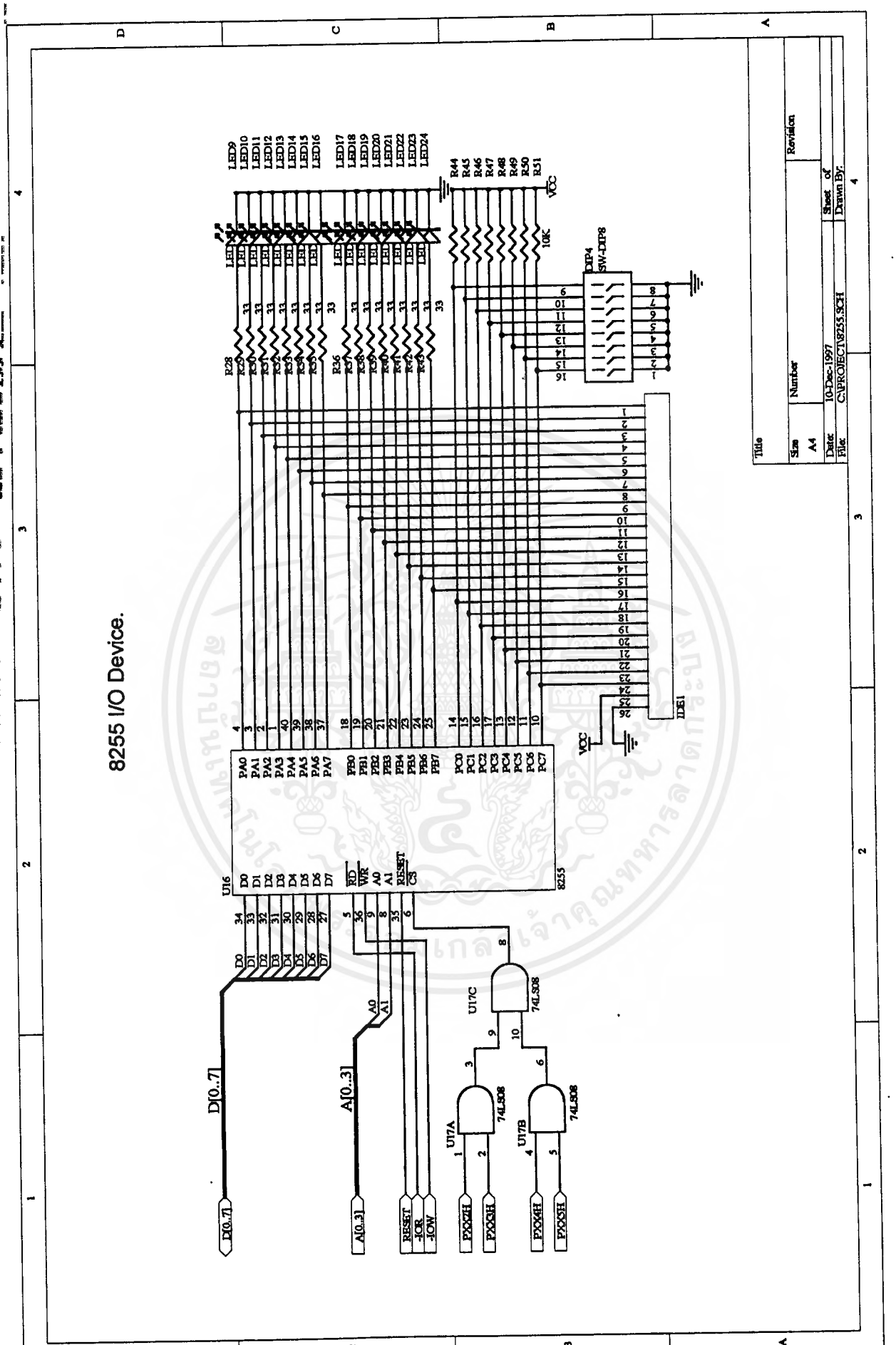


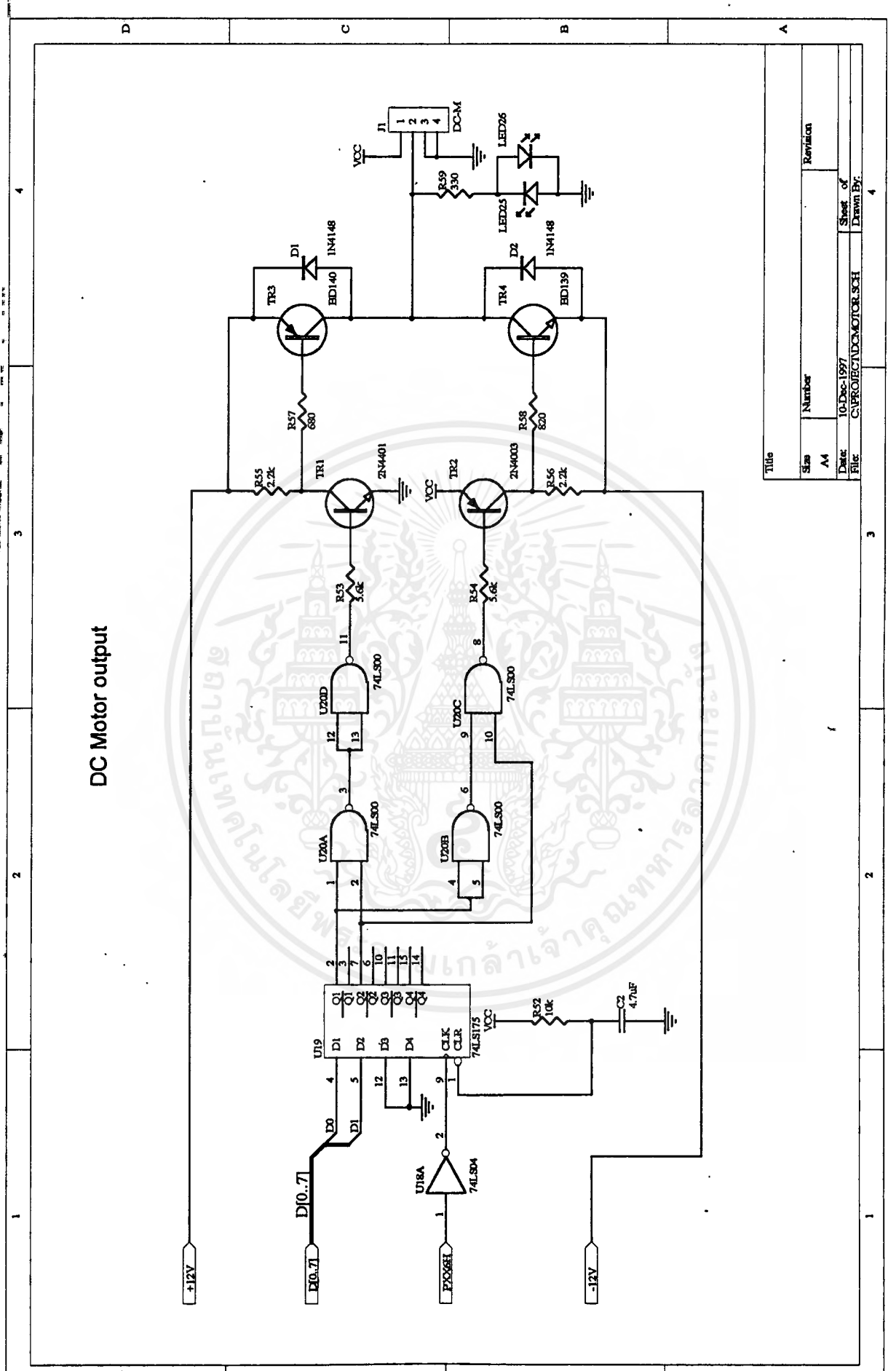


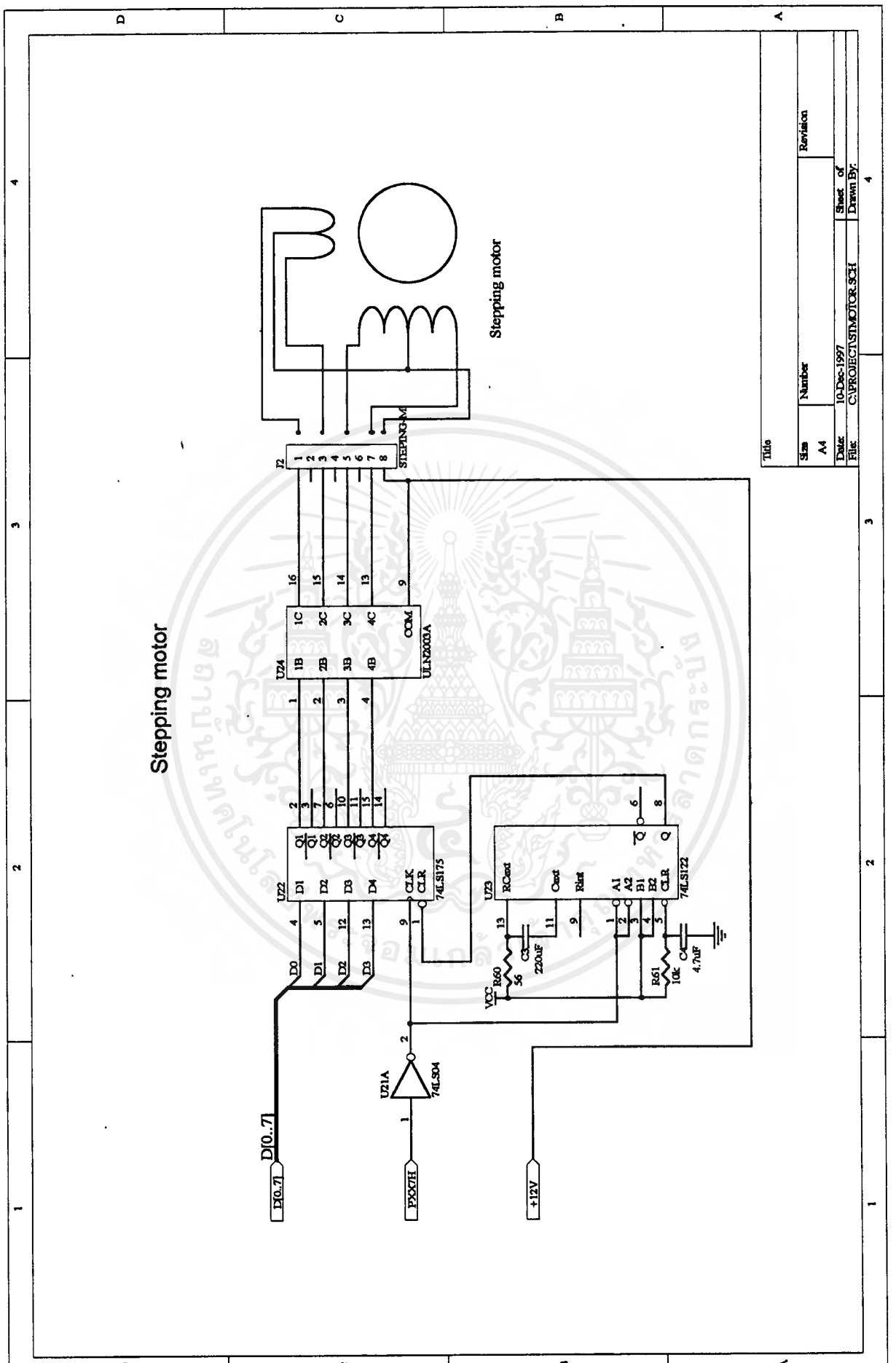


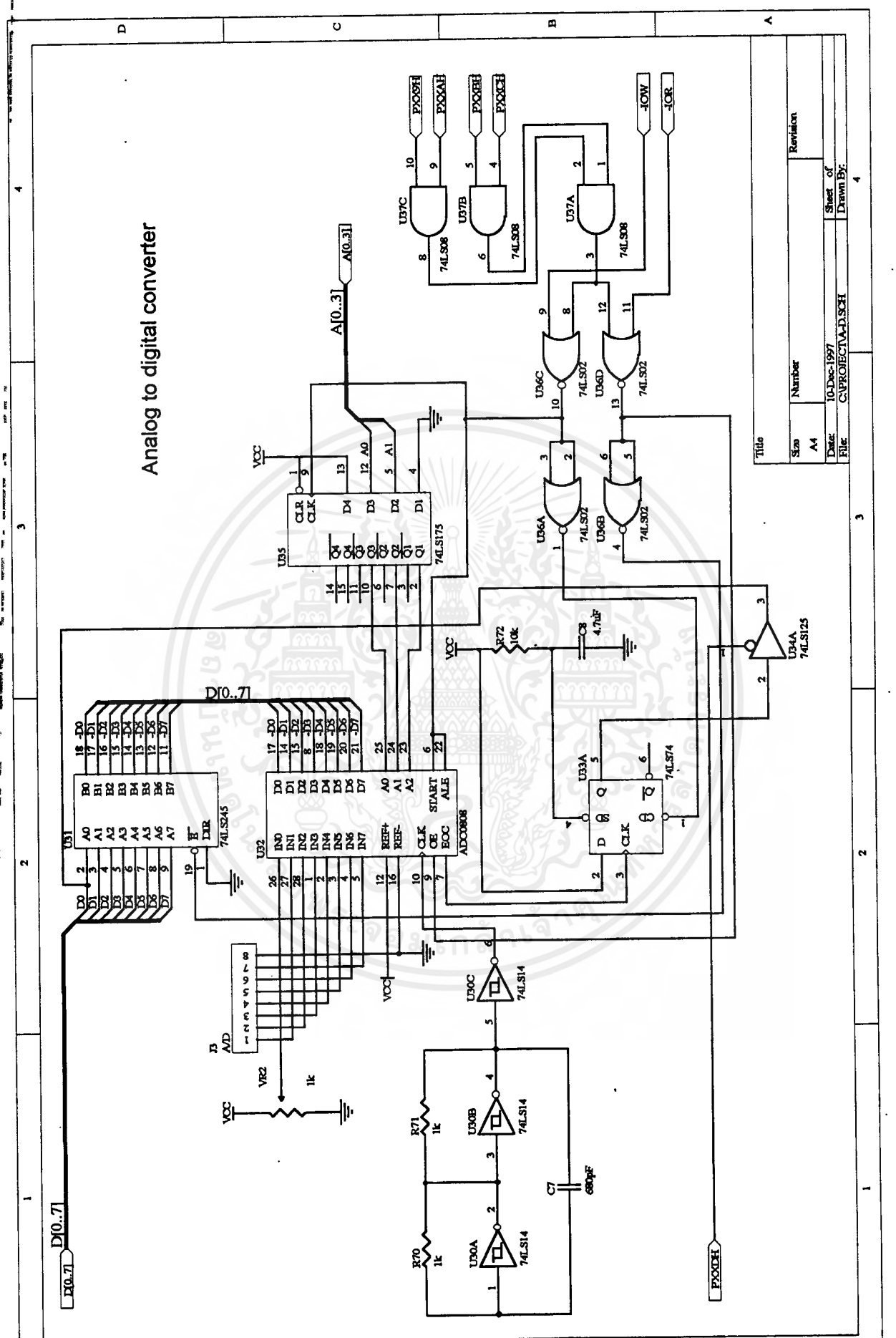


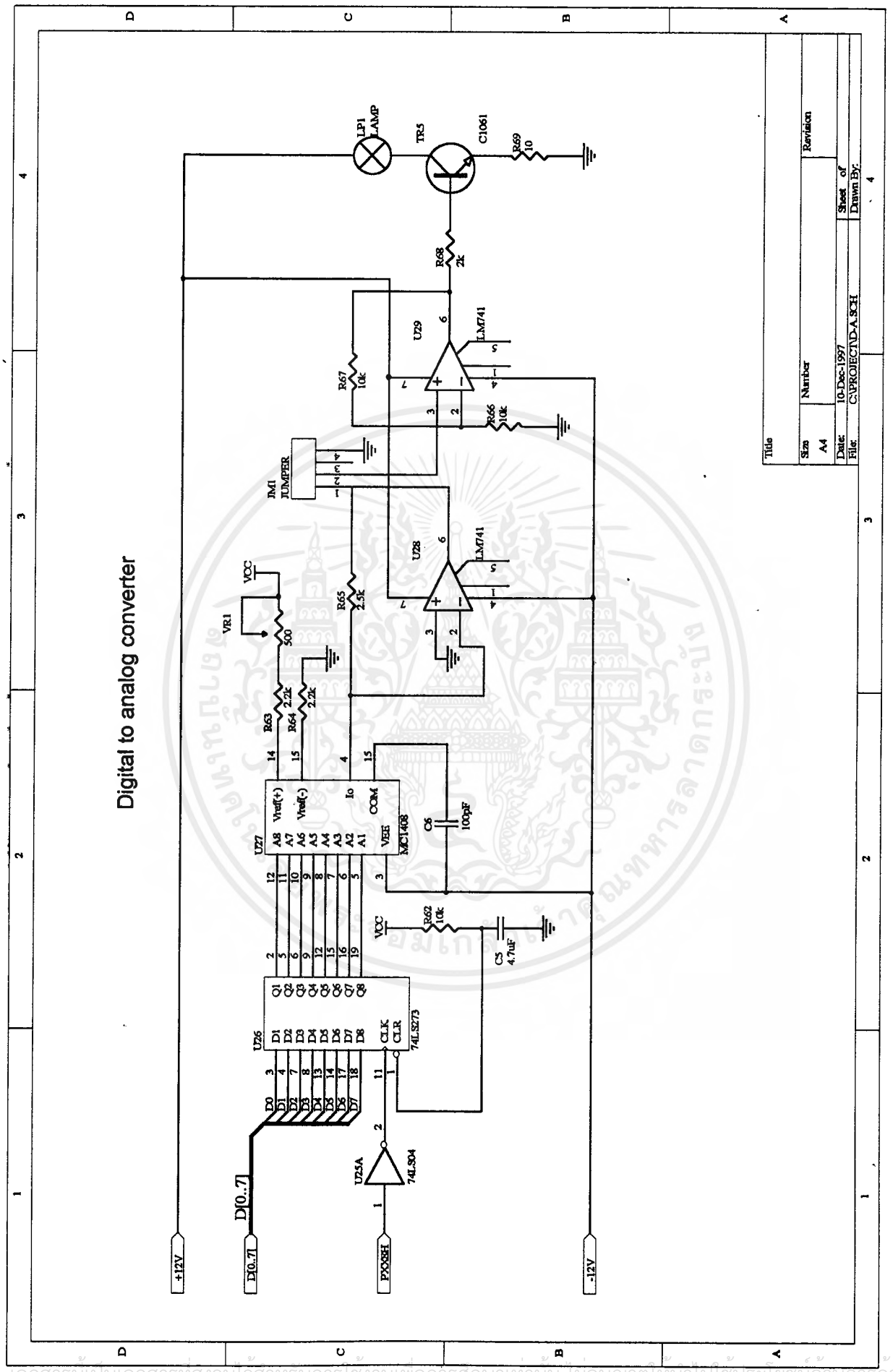


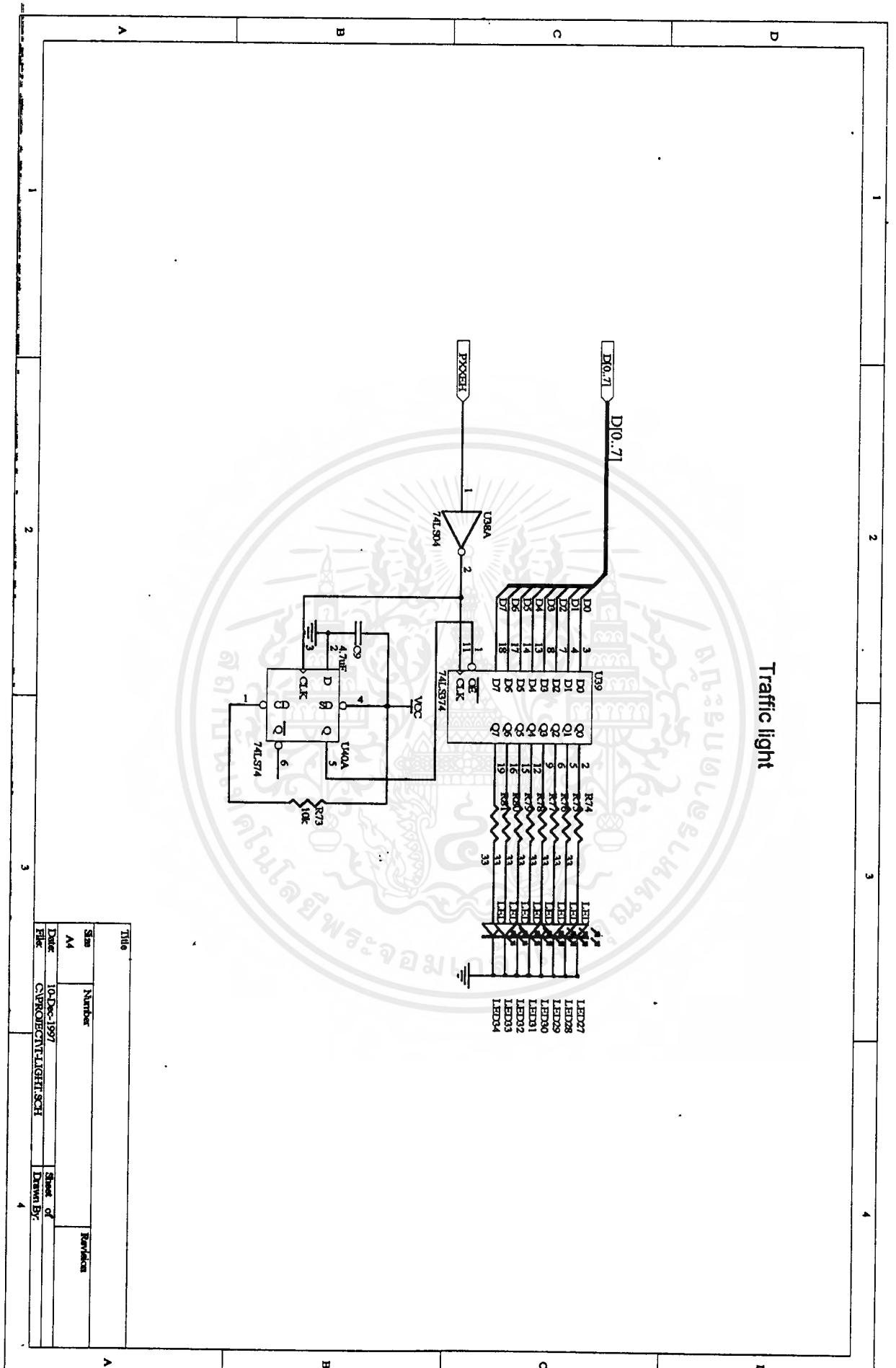




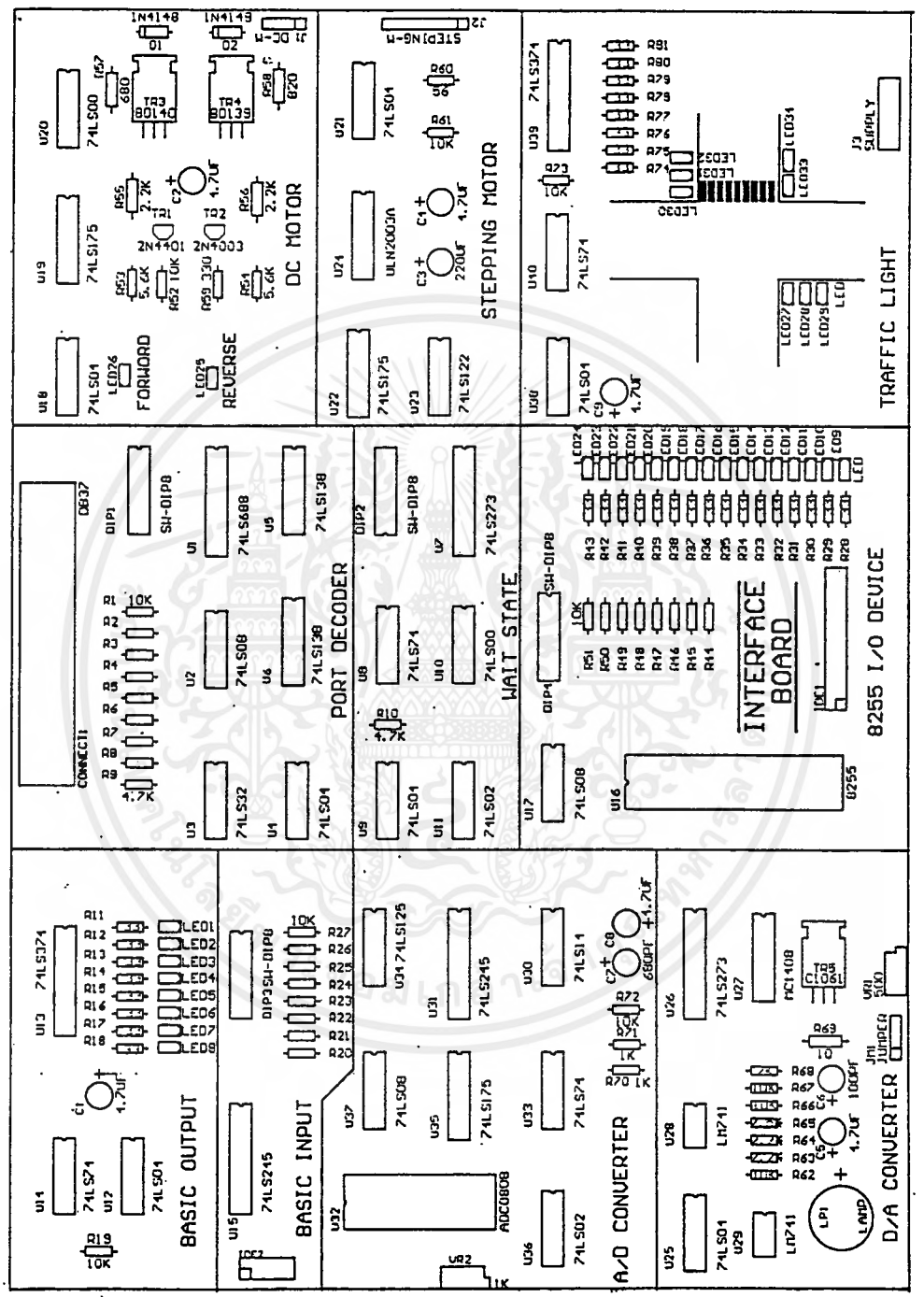








ภาคผนวก ข
ลักษณะการวางอุปกรณ์บนบอร์ดในส่วนต่าง ๆ



ลักษณะการวางอุปกรณ์ของอินเตอร์เฟสบอร์ด

ภาคผนวก ค
ใบงานการทดลองของชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม

การทดลองที่ 1

อินพุท/เอาท์พุท พอร์ทดีโค้ดเดอร์

(INPUT / OUTPUT PORT DECODER)

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถอธิบายวิธีการดีโค้ดพอร์ทของการ์ดอินเตอร์เฟสได้
2. เพื่อให้นักศึกษาสามารถออกแบบวงจรพอร์ทดีโค้ดเดอร์ได้
3. เพื่อให้นักศึกษาสามารถทดสอบการทำงานเบื้องต้นของวงจรดีโค้ดได้

ทฤษฎีเบื้องต้น

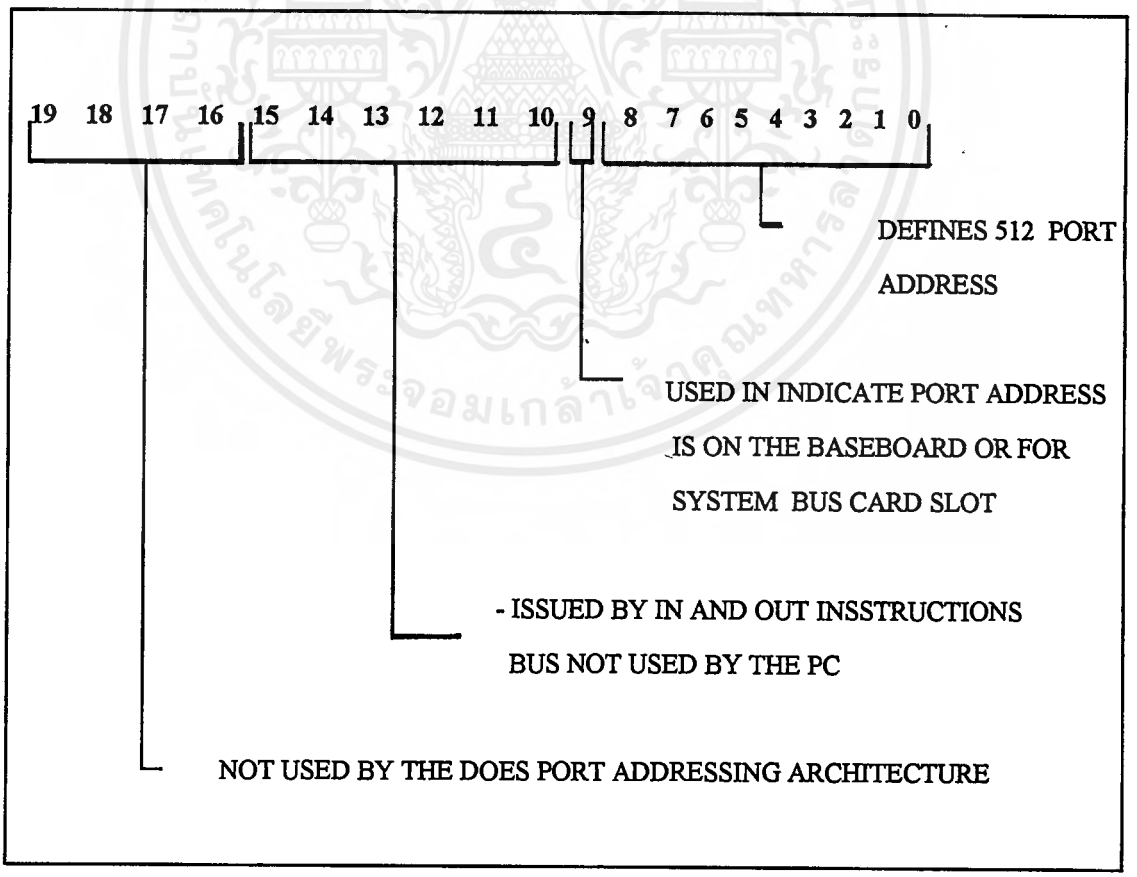
การควบคุมและตรวจสอบสภาวะการทำงาน รวมทั้งการอ่านข้อมูลจากอุปกรณ์ที่เป็นชิพซัพพอร์ทหรือการ์ดต่างๆ ที่ใช้ในระบบของ IBM PC นั้น จะกระทำโดยผ่านทางพอร์ทอินพุท/เอาท์พุทของระบบ เนื่องจากการควบคุมหรือติดต่อกับพอร์ทเหล่านี้ต้องกระทำโดยการอ้างถึง แอดเดรสของพอร์ทอินพุท/เอ้าท์พุทเหล่านั้นโดยตรง จึงจำเป็นต้องศึกษาถึงหลักการอ้างแอดเดรสของ 8088 ใน IBM PC ด้วย

8088 มีแอดเดรสสำหรับใช้กับพอร์ทอินพุท/เอ้าท์พุท อยู่ทั้งสิ้น 65,536 หรือ 64 K ซึ่งทำให้การอ้างแอดเดรสของพอร์ทอินพุท/เอ้าท์พุท ที่ทำงานร่วมกับ 8088 นั้น ต้องใช้จำนวนเส้นแอดเดรสในบัสแอดเดรสทั้งสิ้น 16 เส้น คือ A0-A15 แต่สำหรับใน IBM PC นี้ ถูกออกแบบมาให้ใช้เส้นแอดเดรสเฉพาะ 10 เส้นล่าง คือ A0-A9 เท่านั้น สำหรับแอดเดรสของพอร์ทอินพุท/เอ้าท์พุท ต่างๆ นั้น จะเป็นแอดเดรสที่ถูกสร้างขึ้นโดย 8088 ซึ่งแอดเดรสเหล่านี้เป็นแอดเดรสที่จัดไว้สำหรับพอร์ทอินพุท/เอ้าท์พุทโดยเฉพาะ คือแยกจากแอดเดรสของหน่วยความจำโดยเด็ดขาด ส่วนการส่งข้อมูลเหล่านี้จะทำได้โดยการใช้คำสั่ง OUT ของ 8088 ส่งข้อมูลนั้นไปยังแอดเดรสของพอร์ทที่ต้องการ และสำหรับการตรวจสอบหรือการอ่านข้อมูลจากพอร์ทก็จะทำได้โดยการใช้คำสั่ง IN ของ 8088 อ่านข้อมูลจากแอดเดรสของพอร์ทที่ต้องการเช่นกัน

ค่าแอดเดรสบนเส้นแอดเดรสเหล่านี้ยังคงเปลี่ยนแปลงตามค่าแอดเดรสของพอร์ทที่กำหนดไว้ในคำสั่ง OUT หรือ IN อยู่ด้วยเพียงแต่ไม่ได้ถูกนำมาดีโค้ดร่วมกับแอดเดรส A0-A9 เท่านั้น

พอร์ทบน IBM PC จึงสามารถอ้างแอดเดรสของพอร์ทเพียง 1024 พอร์ท โดยพอร์ทถูกแบ่งออกเป็น 2 กลุ่ม โดยที่กลุ่มแรกเป็นกลุ่มของพอร์ทที่อยู่บนเมนบอร์ด และกลุ่มที่สองเป็นกลุ่มที่จัดเตรียมไว้สำหรับพอร์ทที่อยู่บนการ์ดต่างๆ

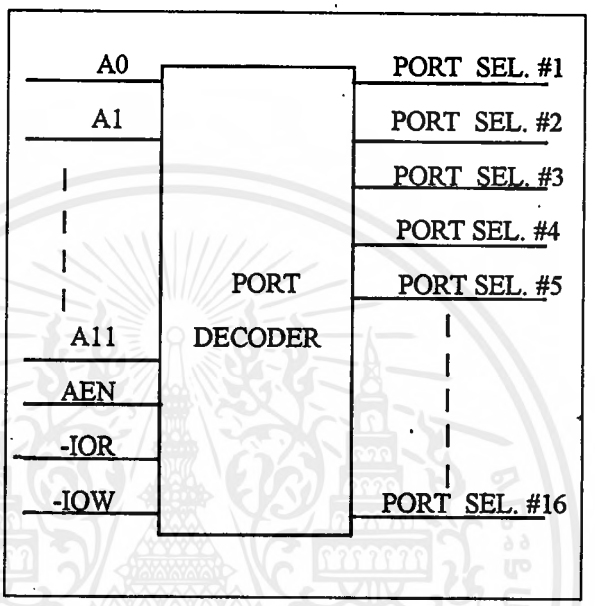
สิ่งหนึ่งที่ต้องคำนึงถึงก็คือถ้าแอดเดรสที่เลือกให้กับพอร์ทนี้ตรงกับค่าแอดเดรสที่มีอยู่บนเมนบอร์ดแล้ว เมื่อทำการส่งข้อมูลให้กับพอร์ทที่อยู่ในตำแหน่งแอดเดรสนี้ ก็จะเท่ากับเป็นการส่งข้อมูลให้กับทั้งพอร์ทที่อยู่บนเมนบอร์ด และพอร์ทที่อยู่บนการ์ดด้วยซึ่งกรณีเช่นนี้อาจจะให้เกิดความผิดพลาดขึ้นได้เช่นกัน ดังนั้นในการกำหนดค่าแอดเดรสให้กับพอร์ทที่ถูกสร้างขึ้นบนการ์ดต่างๆ จึงควรจะใช้ค่าแอดเดรสที่แอดเดรสบิท A9 มีค่าเป็น 1 คือแอดเดรส 0FE00H จนถึง 0FFFFH เท่านั้น (แอดเดรสบิท A0-A15 ไม่ถูกใช้ในการดีโค้ด แต่เพื่อความสะดวกจึงกำหนดให้มีค่าเป็น 1 ในฐานสองทั้งหมด แต่ในการใช้งานจริงอาจเปลี่ยนแปลงให้ แอดเดรส A10-A15 แต่ละบิทมีค่าเป็น 1 หรือ 0 ก็ได้)



รูปที่1.1 การใช้งานและแอดเดรสบิทต่างๆ ในการอ้างแอดเดรสของพอร์ทใน IBM PC

โครงสร้างของ อะแดปเตอร์การ์ด ที่ใช้ในการทดลอง

อะแดปเตอร์การ์ดที่ออกแบบใช้งานในการทดลองนี้ใช้วิธีการดีโค้ดแบบ NON - FIXED ดังผังการทำงานต่อไปนี้



รูปที่1.2 แผนผังการดีโค้ดแบบ NON-FIXED

ในการออกแบบ อะแดปเตอร์การ์ดได้ดีโค้ดพอร์ทไว้ที่ หมายเลข 00XH ถึง FFXH ซึ่งถ้าดูจากการกำหนดหมายเลขพอร์ทในระบบว่าพอร์ท หมายเลข 30XH-30XH เป็นช่วงที่ไม่ได้ใช้งาน (NOT USED) และเหมาะสม ดังนั้นจึงสามารถใช้งานในช่วงนี้ได้ ซึ่งในการออกแบบ ดีโค้ด หมายเลขพอร์ทที่ใช้งานโดย IBM PC โดยสามารถดูได้จากการกำหนดหมายเลขพอร์ทในระบบ ดังต่อไปนี้

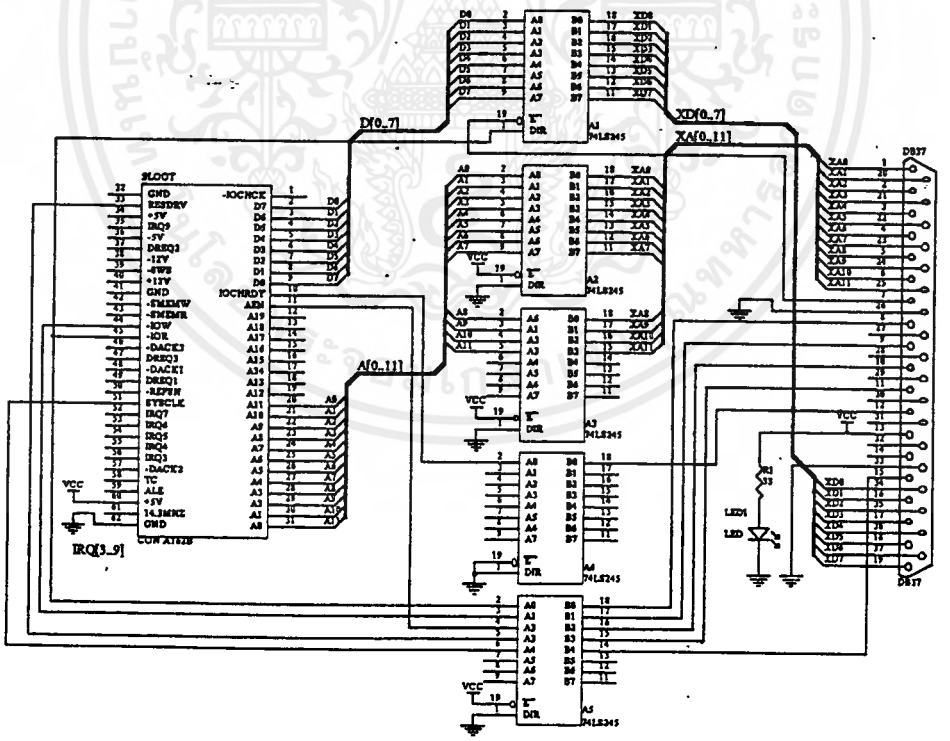
หมายเลขพอร์ทที่ใช้ ได้รับการกำหนดดังนี้

000-00F	พอร์ทของชิป DMA 8237A
020-021	พอร์ทของชิปอินเตอร์รัพต์คอนโทรลเลอร์ 8259A
040-043	พอร์ทของไอซีไทเมอร์ 8253
060-063	พอร์ทของ 8255A ที่อยู่บนเมนบอร์ด
080-083	พอร์ทของดีเอ็มเอที่ใช้กำหนดเพจ (page register)
0AX	รีจิสเตอร์ที่ใช้สำหรับ NMI
0CX	สงวนไว้
0EX	สงวนไว้
200-20F	พอร์ทที่ใช้ในการควบคุมเกม
210-217	ส่วนขยายเพิ่มต่อ
220-20F	สงวนไว้
278-27F	สงวนไว้
2F0-2F7	สงวนไว้
2F8-2FF	พอร์ทสื่อสาร COM2
200-21F	โปรโตไทม์การ์ด
320-32F	วงจรควบคุมฮาร์ดดิสก์
378-37F	เครื่องพิมพ์แบบขนาน
380-38F	วงจรสื่อสาร SDLC
3A0-3AF	สงวนไว้
3B0-3BF	วงจรควบคุมการแสดงผลบน CRT แบบโมโนโครม
3C0-3CF	สงวนไว้
3D0-3DF	วงจรควบคุมการแสดงผลบน CRT แบบสี
3E0-3EF	สงวนไว้
3F0-3F7	วงจรควบคุมดีสไดร์ฟ
3F8-3FF	วงจรควบคุมพอร์ทสื่อสาร COM1

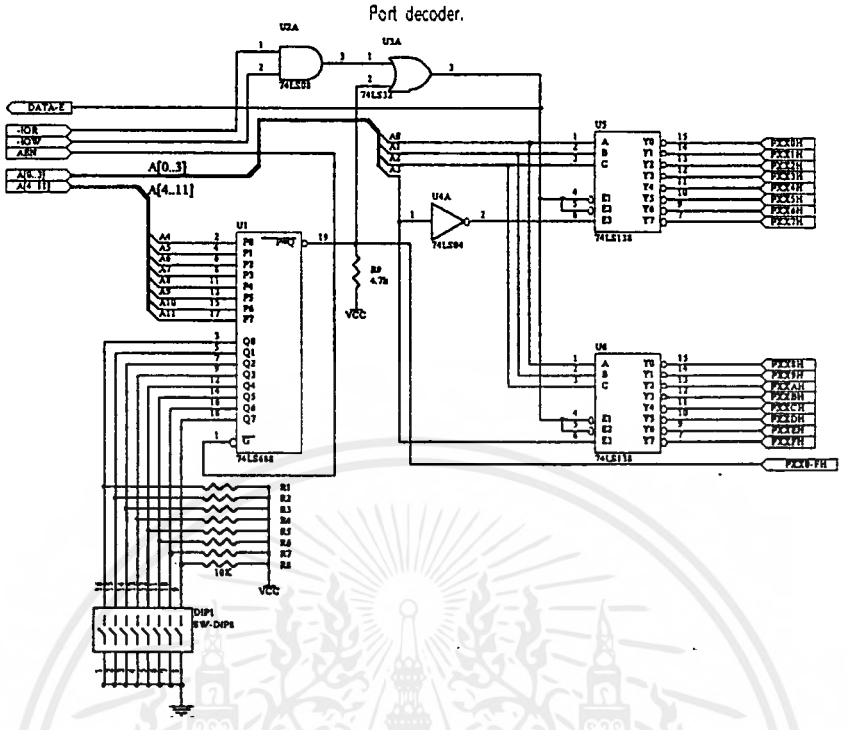
วงจรบน อะแดปเตอร์การ์ด เป็นส่วนของวงจรที่ทำหน้าที่ในการควบคุมการติดต่อระหว่าง IBM PC กับวงจรอินเตอร์เฟสภายนอก ทำหน้าที่เป็นวงจรบัฟเฟอร์ เพื่อให้ง่ายขึ้นจะขอแยกอธิบายการทำงานของวงจรคือ

ไอซี 74LS245 ทำหน้าที่เป็นบัฟเฟอร์เพื่อให้จ่ายกระแสได้สูงขึ้นโดย A2 และ A3 จะเป็นตัว บัฟเฟอร์ ให้กับ แอดเดรสบัส A0-A11 โดย A1 เป็น บัฟเฟอร์ ให้กับบัสข้มูล D0-D7 ซึ่ง A4 เป็น บัฟเฟอร์ให้กับอินพุทของบัสบัสควบคุม คือ IO CHRDY และ A5 เป็น บัฟเฟอร์ ให้กับ เอาท์พุทของบัสควบคุม คือ $\overline{\text{RESET}}$, $\overline{\text{IOW}}$,$\overline{\text{IOR}}$, $\overline{\text{SYSCLK}}$, $\overline{\text{AEN}}$

วงจรบนบอร์ดอินเตอร์เฟส ทำหน้าที่ในการดีโค้ดพอร์ทหมายเลขต่างๆและมีวงจรสร้างสัญญาณรอ (WAIT) ให้กับ IBM PC ในกรณีที่อุปกรณ์ภายนอกทำงานได้ช้า รวมถึงวงจรที่ใช้ในการทดลองในใบงานชุดนี้ทั้งหมด และเพื่อให้ง่ายขึ้นจะขอแยกอธิบายการทำงานในส่วนของวงจรที่เกี่ยวข้องกับการ ดีโค้ดพอร์ท ดังนี้



รูปที่1.3 วงจรบนอะแดปเตอร์การ์ด



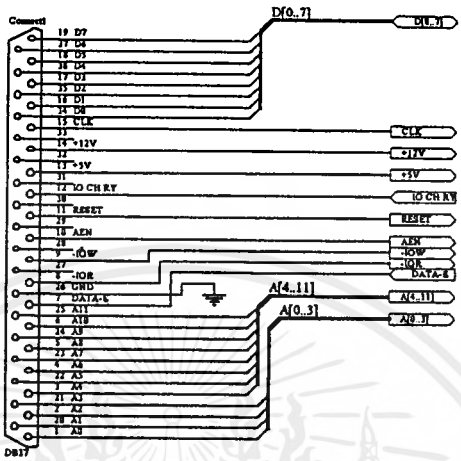
รูปที่1.4 วงจร พอร์ทดีโค้ดเดอร์

วงจรพอร์ทดีโค้ดเดอร์ ประกอบด้วย ไอซี U1,U2,U3,U4,U5,U6 ,R1-R9 และ DIP SW.1 ทำหน้าที่ดีโค้ดตำแหน่งของพอร์ท ได้ตั้งแต่ ตำแหน่งที่ 00XH ถึง FFXH (อยู่ที่การกำหนดการดีโค้ดให้เหมาะสมกับตำแหน่ง อินพุท/เอาท์พุท แม็บ ของ IBM PC)

จากวงจรจะเห็นได้ว่ามีการนำสัญญาณ -IOR , -IOW , AEN มาประกอบในการดีโค้ดด้วย และเมื่อมีการเขียนหรืออ่านข้อมูลผ่านพอร์ท ก็จะเป็นลอจิก 0 และเมื่อไม่มีขบวนการ DMA เกิดขึ้น สัญญาณ AEN ก็จะเป็นลอจิก 0

และเมื่อทำการโยกสวิทช์ DIP SW.1 ก็จะทำการเปรียบเทียบ บัสแอดเดรส (A4-A11) เพื่อทำการเลือก แอดเดรส และเมื่อการเปรียบเทียบเป็นไปอย่างถูกต้อง และ AEN เป็น 0 ก็จะมีลอจิก 0 ที่ขา 19 ของ U1 ซึ่งเมื่อ -IOR , -IOW , เป็นลอจิก 0 ด้วยแล้ว U5 และ U6 ก็จะทำการดีโค้ดให้กับวงจรต่าง ๆ ใน บอร์ดอินเตอร์เฟส

สัญญาณที่คอนเนคเตอร์



รูปที่1.5 สัญญาณที่คอนเนคเตอร์

จากรูปเป็นสัญญาณที่ได้จากอะแดปเตอร์การ์ดที่เสียบบนสล็อตของ IBM PC โดยใช้คอนเนคเตอร์ 37 ขา ผ่านสาย แพร์(PAIR) เพื่อส่งไปยังบอร์ดอินเตอร์เฟส

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีโปรแกรมภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ด อินเตอร์เฟส
4. ออสซิลโลสโคป
5. สายแพร์ (PAIR)
6. โลจิกโพรบ (LOGIC PROBE)

ลำดับขั้นการทดลอง

1. เสียบ อะแดปเตอร์การ์ด ลงที่ สล็อตใด สล็อตหนึ่งของเครื่อง IBM PC ต่อสายแพร์ ไปยัง บอร์ดอินเตอร์เฟส

2. ป้อนโปรแกรมภาษาซี ดังต่อไปนี้

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int  Port=0x300,Data=0x00;
main(){
        while(!kbhit()) {
        outportb(Port,Data); }
}
```

3. EXECUTE โปรแกรมแล้วใช้ LOGIC PROBE วัดที่ขา 15 เทียบกับขา 7,9,10,11,12,13,14 ของ U5 (74LS138) บันทึกผลการทดลอง

ผลการทดลอง.....

.....

.....

4. ป้อนโปรแกรมภาษา C ดังต่อไปนี้

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int  Port=0x300,Data;
main(){
        while(!kbhit()) {
Data=inportb(Port); }}
```

5. EXECUTE โปรแกรมแล้วใช้ลอจิกโพรบ (LOGIC PROBE) วัดที่ขา 15 เทียบกับขา 7,9,10,11,12,13,14 ของ U5 (74LS138) บันทึกผลการทดลอง

ผลการทดลอง.....

.....

.....

6. ทดลองเปลี่ยนหมายเลขพอร์ท เป็นหมายเลขอื่นอยู่ในย่าน 0x300 - 0x30f โดยใช้โปรแกรมในข้อ 2 แล้วใช้ลอจิกโพรบ วัดที่ขา 7,9,10,11,12,13,14,15 ของ U5 และ U6 ของวงจรในรูปที่ 4 บันทึกผลการทดลอง

ผลการทดลอง.....

.....

.....

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

คำถาม

1. จงออกแบบวงจร พอร์ทดีโค้ดเดอร์ให้มีหมายเลข พอร์ทเป็น &H200

2.เปรียบผลการทดลองในข้อ 3 และข้อ 5 ว่าเหมือนหรือแตกต่างกันอย่างไร

หมายเหตุ

เพื่อความเข้าใจที่ดียิ่งขึ้นควรจะทำการศึกษาเกี่ยวกับการจัดแอดเดรสสำหรับพอร์ทอินพุท/เอาท์พุทใน IBM PC : หนังสือเทคโนโลยีฮาร์ดแวร์ IBM PC ของ ยืน ภู่วรวรรณ

การทดลองที่ 2
วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น
(BASIC OUTPUT)

วัตถุประสงค์

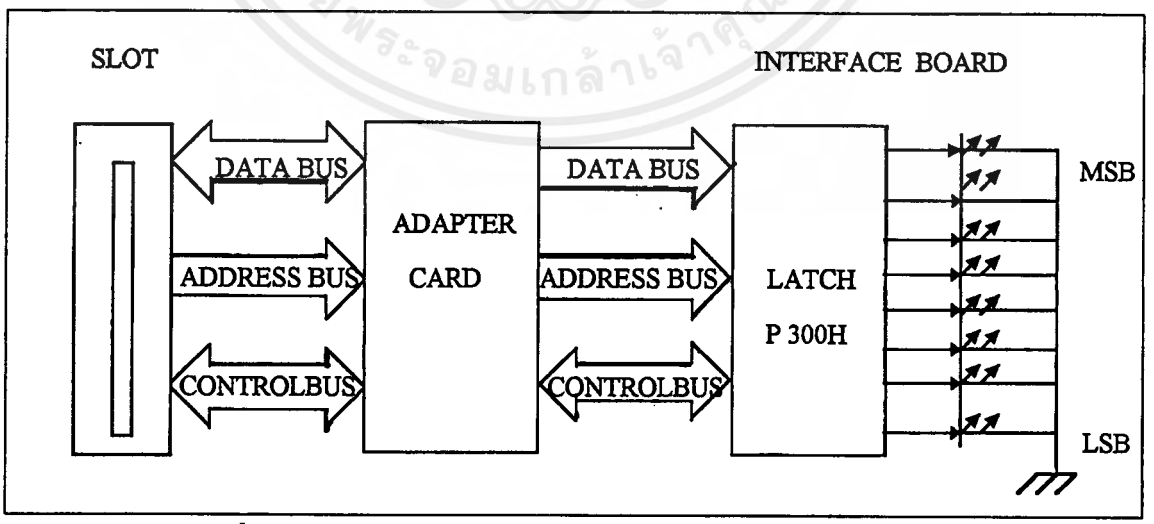
1.เพื่อให้นักศึกษาสามารถอธิบายการทำงานของวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้นของคอมพิวเตอร์ได้

2.เพื่อให้นักศึกษาสามารถเขียนโปรแกรมควบคุมอุปกรณ์ภายนอกที่เป็นไดโอดเปล่งแสงได้

3.เพื่อให้นักศึกษาสามารถเขียนโปรแแกรมภาษาซีประยุกต์ใช้งานกับอุปกรณ์ภายนอกที่เป็นไดโอดเปล่งแสงได้

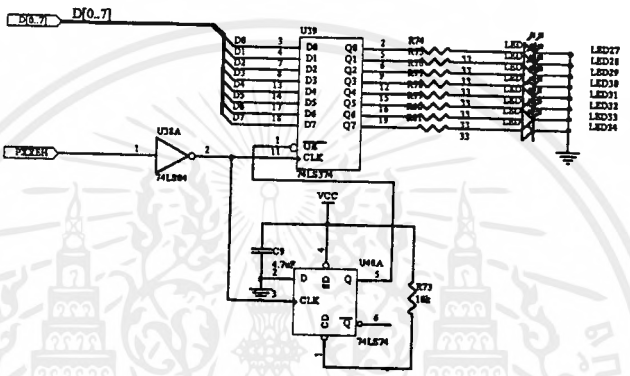
ทฤษฎีเบื้องต้น

วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น เป็นการแสดงให้เห็นถึงหลักการเบื้องต้นในการควบคุมอุปกรณ์ภายนอกอย่างง่าย และแสดงให้เห็นถึงการเขียนโปรแกรมเพื่อส่งข้อมูลไปควบคุมให้ อุปกรณ์ภายนอกที่เป็นไดโอดเปล่งแสง ติด-ดับ ตามต้องการ



รูปที่2.1 ผังการทำงานของวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

จากผังการทำงาน สัญญาณจากสล็อตจะเข้าไปยังอะแดปเตอร์การ์ด เพื่อทำการเปลี่ยนเป็นสัญญาณที่จะส่งให้กับการ์ดอินเตอร์เฟส ซึ่งประกอบด้วยสัญญาณของ บัสข้อมูล , บัสตำแหน่ง และบัส ควบคุม ซึ่งวงจรวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้นถูกดีโค้ดไว้ที่ พอร์ท 0x300



รูปที่ 2.1 วงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น

วงจรประกอบด้วยอุปกรณ์ตัวหลักๆ คือไอซี U13 ไอซี เบอร์ 74LS374 ทำหน้าที่ในการค้างข้อมูลที่ส่งมา โดยแสดงผลด้วยไดโอดเปล่งแสง (LED 1 - LED) 8 , U14A ไอซี 74LS74 ทำหน้าที่เคลียร์ ไอซี 74LS374 เมื่อเริ่มเปิดไฟเข้าเครื่องครั้งแรก R19 และ C1 จะทำให้ขา ของ U14A มีลอจิกเป็น 1 เป็นผลให้ Q0-Q7 ของ U13 อยู่ในสภาวะไฮอิมพิแดนซ์ ไดโอดเปล่งแสงจะดับหมดทุกดวง แต่ถ้ามีการส่งข้อมูลที่ขา D0-D7 ของ U13 ปรากฏที่ขา Q0-Q7 ของ U13 และข้อมูลจะคงค้างอยู่ทำให้ ไดโอดเปล่งแสง ติดอยู่เช่นนั้นจนกว่าจะส่งข้อมูลชุดใหม่มาให้หรือปิดไฟเข้าเครื่อง

การเขียนโปรแกรมเพื่อส่งข้อมูลไปยังพอร์ท 0x300 ในการทดลองนี้มีวิธีการส่งข้อมูลใน 2 ลักษณะคือ

1. การส่งข้อมูลเป็นเลขฐานสิบ

น้ำหนัก	128	64	32	16	8	4	2	1	
บิท	7	6	5	4	3	2	1	0	
ข้อมูล	1	0	0	0	0	0	0	0	128
ข้อมูล	1	1	1	1	0	0	0	0	240

ค่าน้ำหนักประจำตำแหน่งของข้อมูล

ถ้าต้องการส่งข้อมูลเพื่อให้ไดโอดเปล่งแสง(LED) ดวงขวามือสุดติด (LED ทางขวาสุดคือหลัก MSB มีน้ำหนักเป็น 128) จะต้องเขียนโปรแกรมดังนี้

```
outportb(768,128);
```

ต้องการส่งข้อมูลให้ ไดโอดเปล่งแสง 4 บิทบนติดจะต้องส่งข้อมูล 240 (128 + 64 + 32 + 16) ดังนี้

```
outportb(768,240);
```

2. การส่งข้อมูลเป็นเลขฐานสิบหก

บิทที่	7	6	5	4	3	2	1	0	
ข้อมูล	1	0	0	0	0	0	0	0	0x80
ข้อมูล	1	1	1	1	0	0	0	0	0xf0
ข้อมูล	1	0	1	0	0	1	1	0	0xA6

ถ้าต้องการส่งข้อมูลเพื่อให้ไดโอดเปล่งแสง ดวงขวามือสุดติด (ทางขวาสุดคือหลัก MSB) จะต้องเขียนโปรแกรมดังนี้

```
outportb(0x300,0x80);
```

หรือถ้าต้องการส่งข้อมูลให้ไดโอดเปล่งแสง 4 บิทบน ติดจะต้องส่งข้อมูล &HF0 ดังนี้

```
outportb(0x300,0xF0);
```

ตัวอย่างโปรแกรม

เป็นโปรแกรมภาษาซี เพื่อให้ไดโอดเปล่งแสง (LED) ติด-ดับ ต่อเนื่องกันเป็นไฟวิ่ง

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    Port=0x300,Delay=100;
char
a,Data[]={0x01,0x02,0x40,0x80,0x10,0x20,0x40,0x80};
main(){
        while(!kbhit()) {
        outportb(Port,Data[a++]);
        if(a>=8)  a=0;
        delay(Delay); }
}
```

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีภาษาซี
2. อะแดปเตอร์การ์ด
3. บอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมต่อไปนี้ แล้วรัน โปรแกรม บันทึกผลที่เกิดขึ้น

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int  Port=0x300,Data=1;
main(){
        outportb(Port,Data);
        getch();
}
```

ผลการทดลอง.....

.....

2. **ทดลองเปลี่ยน Data เป็น 2, 4, 16, 32, 64, 128 แล้วรันโปรแกรม บันทึกผลการทดลองลงในตารางที่ 1**

DATA	LED ที่ติด
2	
4	
8	
16	
32	
64	
128	

ตารางที่ 1

สรุปและวิจารณ์ผลการทดลอง.....

.....

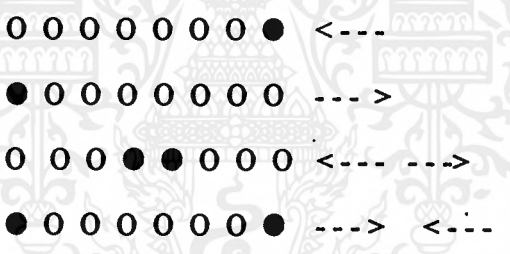
.....

.....

.....

คำถาม

1. จากโปรแกรมไฟวิ่งดังตัวอย่างถ้าต้องการให้ไดโอดเปล่งแสง (LED) วิ่งเร็วขึ้นต้องแก้ไขโปรแกรมอย่างไร
2. ถ้าต้องการให้ไดโอดเปล่งแสง (LED) เป็นไฟวิ่งดังรูป จงเขียนโปรแกรม



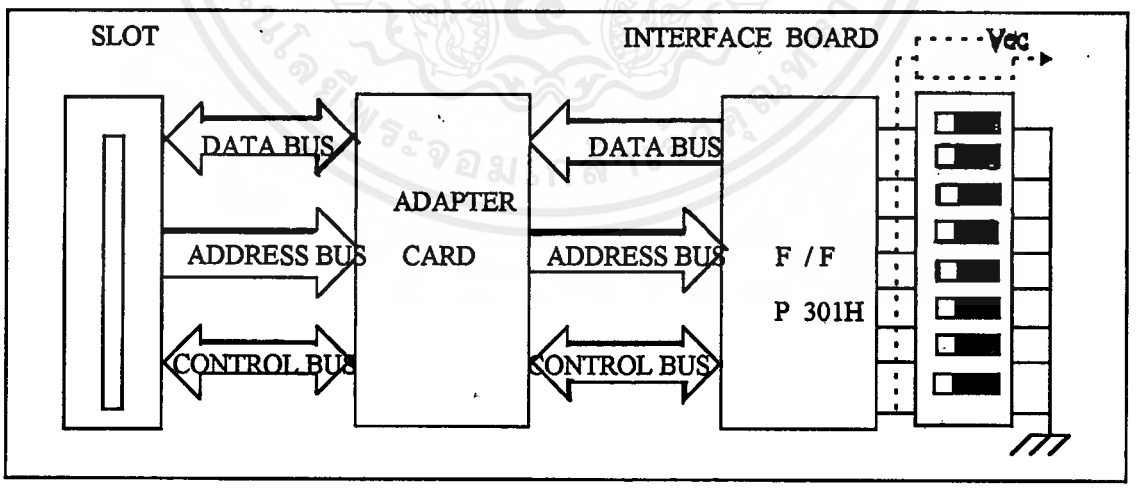
การทดลองที่ 3
วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น
(BASIC INPUT)

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถอธิบายการทำงานของวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้นของคอมพิวเตอร์
2. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมเพื่อรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้นได้
3. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมประยุกต์เพื่อรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้นไปควบคุมอุปกรณ์ภายนอกได้

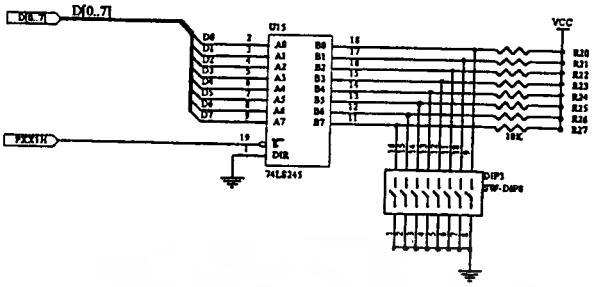
ทฤษฎีเบื้องต้น

ส่วนของวงจรดิพสวิทช์(DIP SW. INPUT) แสดงให้เห็นถึงหลักการเบื้องต้นในการรับข้อมูลจากอุปกรณ์อินพุทประเภทสวิทช์



รูปที่3.1 ผังการทำงานของวงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

การทำงานของวงจร



รูปที่3.2 วงจรรับข้อมูลจากอุปกรณ์อินพุทเบื้องต้น

วงจรประกอบด้วยอุปกรณ์หลัก คือไอซี U15 IC 74LS245 ทำหน้าที่เป็นตัวรับข้อมูลจาก DIP SW.3 เพื่อส่งให้กับ IBM PC ไอซี 74LS245 ถูก ดีโค้ด ไว้ที่พอร์ท 0x301 การทำงานในสภาวะปกติ ถ้า SW.2 อยู่ในตำแหน่งปิด (OFF) ที่ 1A1 ถึง 1A4 และ 2A1 ถึง 2A2 จะมีลอจิกเป็น 1 เนื่องจากถูกดึงขึ้น (PULL UP) ด้วย RP2 แต่ถ้า SW3 ตัวใดตัวหนึ่งอยู่ในตำแหน่งเปิด (ON) ก็จะได้ลอจิก 0 ที่ตำแหน่งนั้น และถ้ามีการเขียนโปรแกรมเพื่อรับข้อมูลจากพอร์ท 0x301 ก็จะได้ข้อมูลที่มีลอจิก 0 ตำแหน่งนั้นด้วย

<u>ตัวอย่างโปรแกรม</u>

เป็นโปรแกรมเพื่อรับข้อมูลจากดิฟสวิตช์ที่ พอร์ท 0x301 แล้วแสดงผลที่หน้าจอภาพ โดยแสดงผลทั้งในรูป เลขฐาน 10 และฐาน 16 (สับดิฟสวิตช์ 3 ก่อนแล้วรันโปรแกรม).

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int  Port=0x301,Data;
main(){
        clrscr();
        Data=inportb(Port);
        printf(" Data = %dD or  %xH",Data,Data);
        getch();
```

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมดังต่อไปนี้แล้วทำการรัน ทดลองเปลี่ยนตำแหน่ง DIP SW. ตามตารางที่ 1 บันทึกผลการทดอง

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int  Port=0x301,Data;
main(){
        clrscr();
        while(!kbhit()){
        Data=inportb(Port);
        gotoxy(10,10);
        printf(" Data = %dD or  %xH
        ",Data,Data);
delay(100); }
}
```

DIP SW.								ค่าที่อ่านได้
8	7	6	5	4	3	2	1	
OFF	OFF	OFF	OFF	OFF	OFF	OFF	ON	
OFF	OFF	OFF	OFF	OFF	OFF	ON	OFF	
OFF	OFF	OFF	OFF	OFF	ON	OFF	OFF	
OFF	OFF	OFF	OFF	ON	OFF	OFF	OFF	
OFF	OFF	OFF	ON	OFF	OFF	OFF	OFF	
OFF	OFF	ON	OFF	OFF	OFF	OFF	OFF	
OFF	ON	OFF	OFF	OFF	OFF	OFF	OFF	
ON	OFF	OFF	OFF	OFF	OFF	OFF	OFF	

ตารางที่ 3.1 การกดดิฟสวิตช์

สรุปและวิจารณ์ผลการทดลอง.....

.....

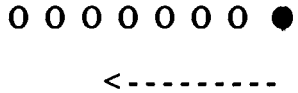
.....

.....

.....

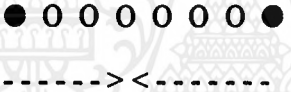
คำถาม

1. จงเขียนโปรแกรมเพื่อใช้ดิฟสวิตช์เป็นตัวควบคุมความเร็วไฟวิ่ง
2. จงเขียนโปรแกรมที่ใช้ดิฟสวิตช์ ตัวที่ 1 และ 2 เป็นตัวควบคุมรูปแบบของไฟวิ่งดังนี้



สวิตช์ 2	สวิตช์ 1
เปิด	เปิด

สวิตช์ 2	สวิตช์ 1
เปิด	ปิด



สวิตช์ 2	สวิตช์ 1
ปิด	เปิด

สวิตช์ 2	สวิตช์ 1
ปิด	ปิด

การทดลองที่ 4

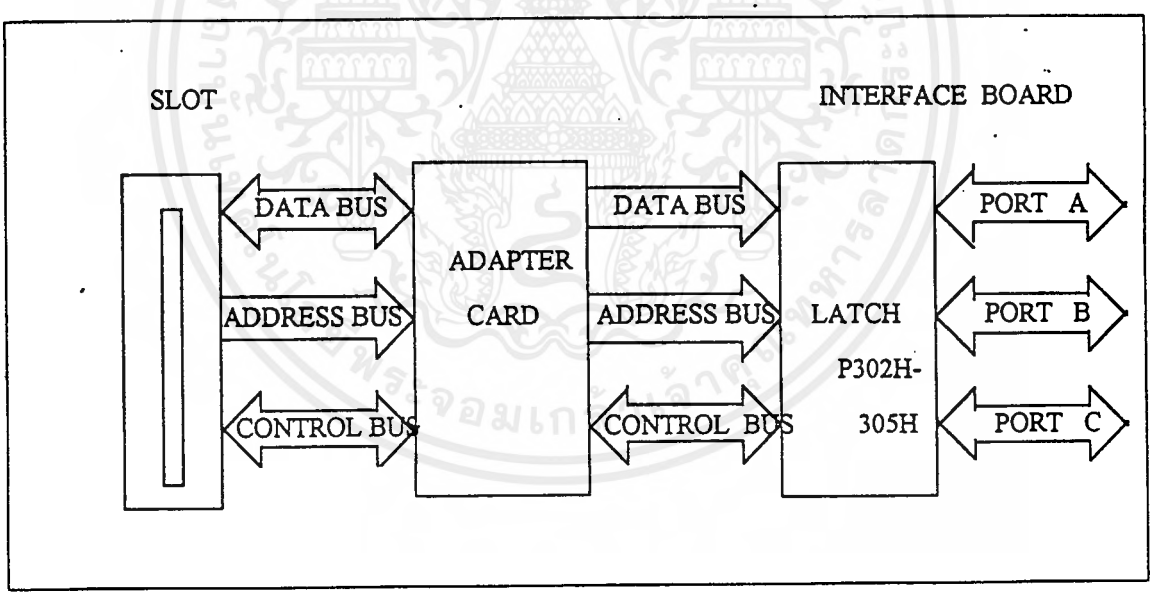
การใช้งาน 8255 เป็นพอร์ทอินพุท/เอาท์พุท

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถใช้งาน 8255 เป็น พอร์ทอินพุท/เอาท์พุทได้
2. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมเพื่อสั่งงาน 8255 ได้
3. เพื่อให้นักศึกษาสามารถทดสอบการทำงานเบื้องต้นของ 8255 เป็นพอร์ทอินพุท/เอาท์พุทได้

ทฤษฎีเบื้องต้น

การใช้งาน 8255 เป็นพอร์ทอินพุท/เอาท์พุทสามารถต่อใช้งานดังผังการทำงานต่อไปนี้



รูปที่ 4.1 ผังการทำงาน ของการใช้งาน 8255

ไอซี 8255 เป็น ไอซีพอร์ทที่สามารถโปรแกรมการทำงานได้ สร้างขึ้นมาใช้กับ 8088 แม้ว่าภายหลังจะมีการประยุกต์ไปใช้งานกับ Z-80 ก็ตามในการทดลองจะใช้กับ 8088

<u>รายละเอียดของการจัดเรียงขา 8255</u>

ในส่วนนี้จะพิจารณาหน้าที่ของขาแต่ละขาของ 8255 ดังนี้

D0 - D7 : เป็นสายข้อมูล อินพุท / เอาท์พุท แบบสองทิศทาง (BI-DIRECTIONAL BUS) ใช้สำหรับวัดหรือส่งข้อมูลระหว่าง CPU กับ 8255

CS : (CHIP SELECT INPUT) เป็นขาที่เป็นตัวกำหนดให้ 8255 ทำงานหรือไม่ กล่าวคือ ถ้าขานี้ได้รับลอจิก 0 ก็สามารถ อ่าน หรือ เขียน ข้อมูลกับ 8255 ได้

RD : (READ INPUT) เป็นกระบวนการอินพุท คือเมื่อขามีลอจิกเป็น '0' และสัญญาณ CS มีลอจิกเป็น 0 ข้อมูลจาก 8255 จะปรากฏ สู่ระบบบัสข้อมูล CPU ก็จะสามารถอ่านข้อมูลออกไปได้

WR : (WRITE INPUT) เป็นกระบวนการเอาท์พุท คือเมื่อขานี้มีลอจิกเป็น 0 และสัญญาณ CS มีลอจิกเป็น 0 ข้อมูลจาก ระบบบัสข้อมูลจะถูกส่งเข้าไปยัง 8255

RESET : เมื่อขานี้มีสถานะเป็น 1 8255 จะถูกรีเซ็ท

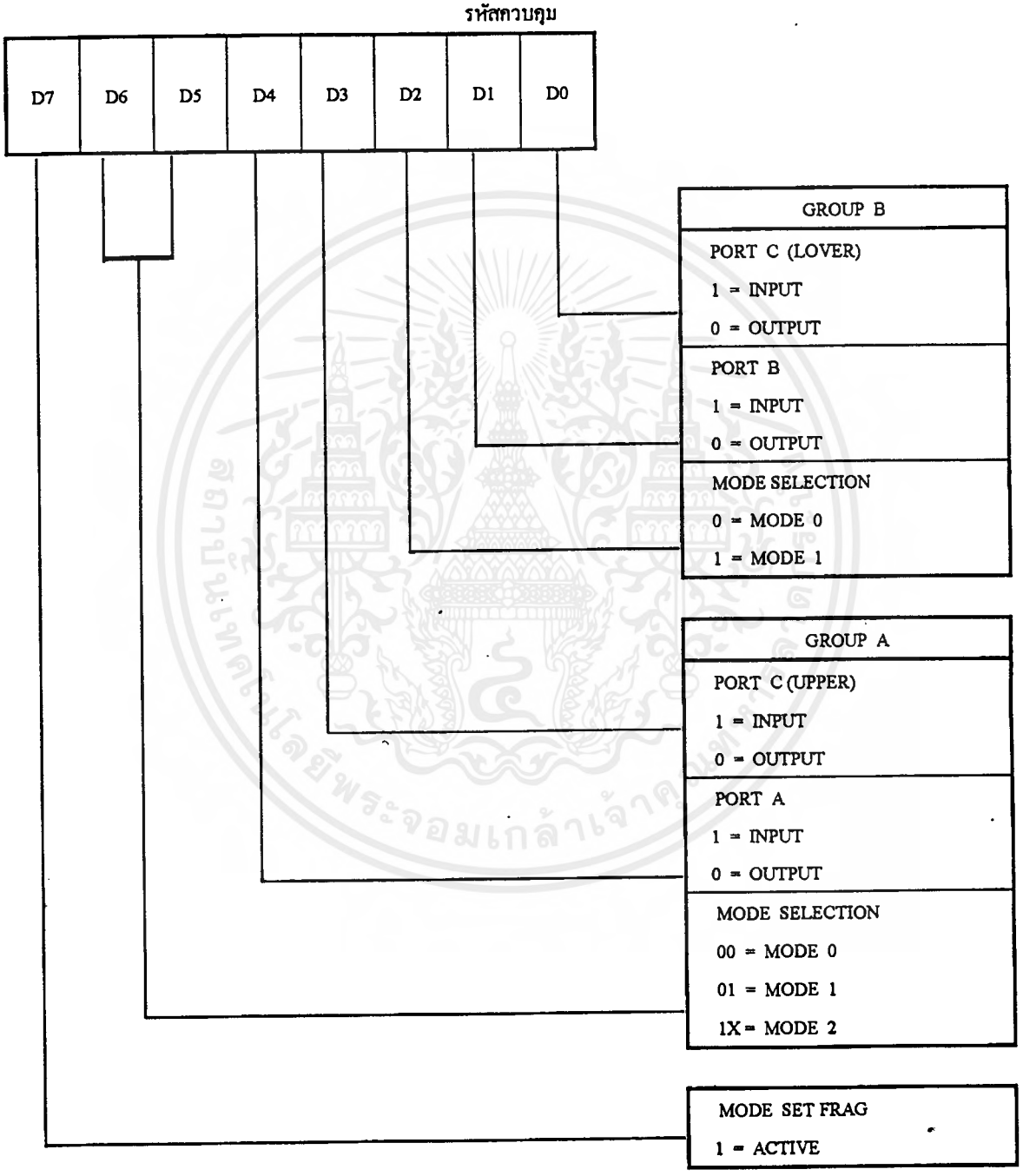
PA0 PA7, PB0-PB7 : เป็น พอร์ทอินพุท/เอ้าท์พุท ขนาด 8 บิท ใช้ต่อกับอุปกรณ์ภายนอกอื่นๆ

PC0-PC7 : เป็น พอร์ทอินพุท/เอ้าท์พุท ขนาด 8 BIT เช่นเดียวกับ PA0-PA7 และ PB0-PB7 แต่ PC0-PC7 สามารถแบ่งเป็นสองกลุ่มคือ PC0-PC3 และ PC4-PC7

A0 - A1 : (ADDRESS INPUT) เป็นตัวกำหนดการติดต่อระหว่าง CPU กับ พอร์ทต่างๆ ของ 8255 ว่า CPU ต้องการติดต่อกับ A, B, C หรือ พอร์ทคอนโทรล โดยพิจารณาร่วมกับสัญญาณ $\overline{RD}$ และ $\overline{WR}$ ดังนี้

รีจิสเตอร์ควบคุมของ 8255

ในการใช้งาน 8255 จะต้องส่งรหัสควบคุม (CONTROL WORD) ไปยังพอร์ทควบคุม (CONTROL PORT) เพื่อควบคุมการทำงานของ 8255 ว่าให้ทำงานในโหมดไหน และให้แต่ละพอร์ทเป็น อินพุท หรือเอาท์พุท



ความหมายของบิทต่างๆ ของรหัสควบคุม

ตัวอย่าง

กำหนดให้ 8255 เป็น พอร์ทอินพุท/เอาท์พุท แบบธรรมดา (BASIC I/O PORT) โดย พอร์ท A และ พอร์ท B เป็นเอาท์พุท พอร์ท C เป็น อินพุทจะได้รหัส สำหรับรหัสควบคุม (CONTROL WORD) ดังนี้

D7	D6	D5	D4	D3	D2	D1	D0	
1	0	0	0	1	0	0	1	
8				9				ฐาน 16 (HEX)
137								ฐาน 10 (DECIMAL)

จากนั้นก็นำค่านี้ส่งไปที่ พอร์ทควบคุม ซึ่งตามการทดลองนี้พอร์ทควบคุม ถูกกำหนดไว้ที่ พอร์ท 303H (A0 และ A1 เป็น 1)

หมายเหตุ

DECODE	ADDRESS BIT								8255
PORT	A7	A6	A5	A4	A3	A2	A1	A0	PORT
304H	0	0	0	0	0	1	0	0	A
305H	0	0	0	0	0	1	0	1	B
302H	0	0	0	0	0	0	1	0	C
303H	0	0	0	0	0	0	1	1	Control P

ตัวอย่างโปรแกรมภาษาซีเพื่อส่งรหัสควบคุมไปยังพอร์ทควบคุม

1. กรณีเป็นเลขฐานสิบหก

```
outportb(0x303,0x89);
```

2. ในกรณีส่งเป็นเลขฐานสิบ

```
outportb(0x303, 137);
```

หรือ

```
outportb(777,137);
```

โหมด

แบ่งออกเป็น 3 โหมด คือ

4.1 การทดลอง 8255 ในโหมด 0

ในการกำหนดให้ 8255 ทำงานใน โหมด 0 นั้น จะต้องมีการส่งรหัสควบคุม (CONTROL WORD) ให้แก่ 8255 ก่อน รหัสควบคุมนี้จะกำหนดลักษณะการทำงานให้แต่ละพอร์ทของ 8255

ตัวอย่างหนึ่งของคำสั่งที่จะสั่งให้ 8255 ทำงานอยู่ในโหมด 0 ได้แก่

D7	D6	D5	D4	D3	D2	D1	D0
1	0	0	0	0	0	0	0

บิท D7 เป็นตัวกำหนดให้พอร์ท A เป็นพอร์ทเอาท์พุท

บิท D6, D5 เป็นตัวกำหนดโหนดการทำงานของพอร์ท A คือ D5 , D6 มีค่าเป็น 0 แสดงว่าอยู่ในโหมด 0

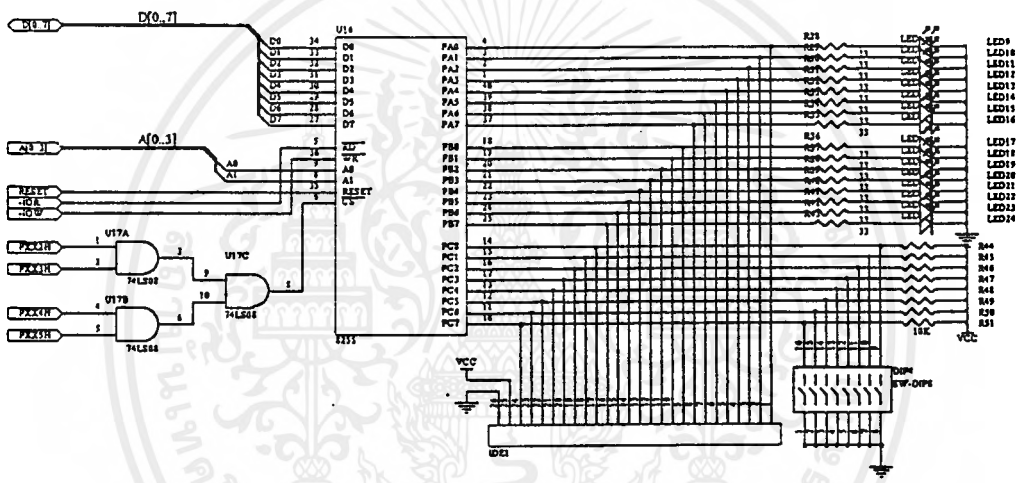
บิท D4 เป็น 0 กำหนดให้พอร์ท A เป็นพอร์ทเอาท์พุท

บิท D3 เป็น 0 เซ็ทพอร์ท C 4บิทบนเป็นพอร์ทเอาท์พุท

บิท D2 เป็น 0 เซ็ทโหมดของพอร์ท B ให้พอร์ท B อยู่ในโหมด 0

บิท D1 เป็น 0 เซ็ทพอร์ท B ให้เป็นพอร์ทเอาท์พุท

บิท D0 เป็น 0 เซ็ทพอร์ท C ให้ 4 บิทล่างเป็นเอาท์พุท



รูปที่4.2 วงจรใช้งาน 8255 เป็นอินพุท/เอาท์พุท

วงจรที่ใช้ในการทดลองประกอบด้วยอุปกรณ์หลักคือ 8255 ซึ่งถูก ดีโค้ดไว้ที่พอร์ท 302H - 305H จากวงจรกำหนดให้ พอร์ท A และ พอร์ท B เป็น พอร์ทเอาท์พุท และ พอร์ท C เป็น พอร์ทอินพุท ทำงานใน โหมด 0 แต่ถ้าต้องการใช้งาน 8255 ในลักษณะอื่น ก็สามารถทำได้โดยการเปลี่ยนรหัสควบคุมตามความเหมาะสม

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์
5. มัลติมิเตอร์ หรือออสซิลโลสโคป

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมตามตัวอย่างจากนั้นรันโปรแกรม
2. ทดลองเปลี่ยนค่าดิฟสวิตช์(SW.1)สังเกตผลที่เกิดขึ้น

ตัวอย่างโปรแกรม โปรแกรมต่อไปนี้เป็นโปรแกรมในการรับข้อมูลจากพอร์ท C (ในที่นี้คือดิฟสวิทช์ SW.4) แล้วส่งข้อมูลที่ได้ออกไปที่พอร์ท A และ พอร์ท B เพื่อทำให้ไดโอดเปล่งแสง (LED) ติด-ดับ ตามการเปลี่ยนแปลงของ SW.1 ที่ พอร์ท C

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int     A=0x304,B=0x305,C=0x302,CtrP=0x303;
char   Data,CtrW=0x89;
main(){
        outportb(CtrP,CtrW);
        while(!kbhit()){
                Data=inportb(C);
                outportb(A,Data);
                outportb(B,Data);
                delay(10); }
}
```

ผลการทดลอง.....

.....

.....

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

.....

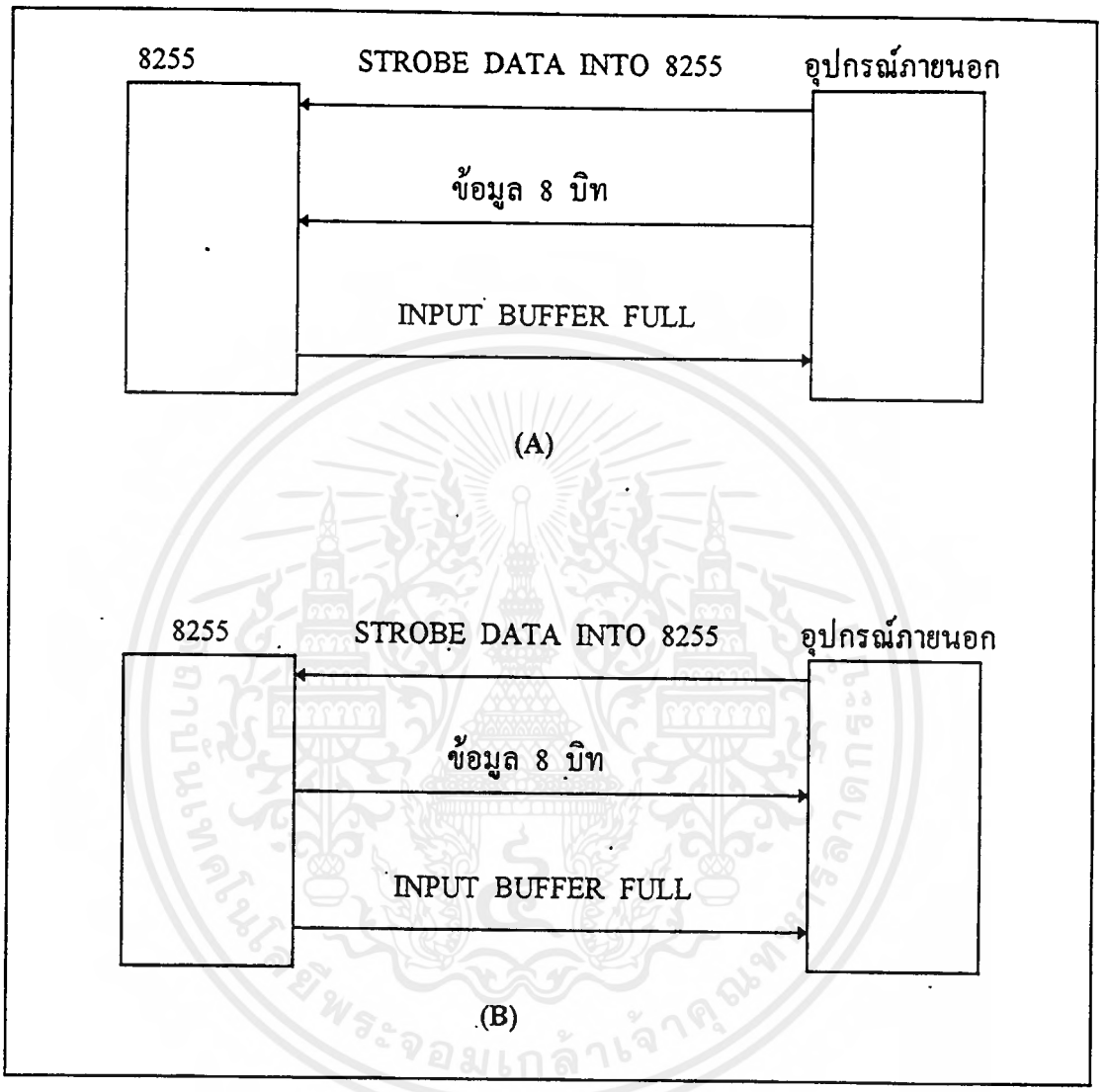
คำถาม

1. จงเขียนโปรแกรมเพื่อใช้ดิฟสวิตช์เป็นตัวควบคุมความเร็วไฟวิ่งที่ พอร์ท A และ พอร์ท B
2. จงเขียนโปรแกรมเพื่อใช้ดิฟสวิตช์เป็นตัวควบคุมรูปแบบของไฟวิ่งที่ พอร์ท A และ พอร์ท B
3. จงอธิบายวิธีการหา รหัสควบคุม เพื่อ ตั้งค่าเริ่มต้น (INITIALIZE) ของ 8255 ถ้ากำหนดให้ พอร์ท A เป็นพอร์ทอินพุท และ พอร์ท B , C เป็นพอร์ทเอาท์พุท

4.2 การทดลอง 8255 ในโหมด 1

ในการทำงานของ 8255 ใน โหมด 1 นี้เป็นการทำงานในลักษณะของการที่ทำให้อินพุทและ เอาท์พุทมีการตรวจสอบสัญญาณ (HANDSHAKE) พอร์ท A และ พอร์ท B จะเป็นพอร์ทข้อมูล ส่วนพอร์ท C จะเป็นสัญญาณแฮนด์เช็ค (HANDSHAKE) โดย 4 พอร์ท บนของพอร์ท C จะเป็นสัญญาณแฮนด์เชคให้กับพอร์ท A และ 4 บิทล่าง จะเป็นสัญญาณแฮนด์เช็คให้กับ พอร์ท B

หลักการรับส่งข้อมูลในในวิธีการของแฮนด์เช็คนี้คือการให้อุปกรณ์ภายนอกส่งสัญญาณแสดงสภาวะความพร้อมให้กับ 8255 ดังรูปที่ 4.3



รูปที่ 4.3 (A) และ (B) แผนผังลักษณะการทำงานของการติดต่อระหว่าง 8255 กับอุปกรณ์ภายนอกในลักษณะแฮนด์เช็ค

รูปที่ 4.3 (A) ข้อมูลจะถูกส่งออกจากอุปกรณ์ภายนอกเข้าสู่ 8255 ก่อนที่อุปกรณ์ภายนอกจะส่งข้อมูลให้แก่ 8255 จะต้องตรวจสอบอินพุท บัฟเฟอร์ ฟูล แฟลก (INPUT BUFFER FULL FLAG) ก่อน ถ้าแฟลกเป็นจริงแสดงว่าข้อมูลในบัฟเฟอร์ของ 8255 นี้ยังไม่ถูกอ่านในซีพียู คือข้อมูลจากอุปกรณ์ภายนอกส่งข้อมูลให้กับ 8255 และซีพียูยังไม่ได้อ่านข้อมูลเข้าไป ถ้าแฟลกเป็นเท็จแสดงว่าซีพียูอ่านข้อมูลออกไปแล้ว อุปกรณ์ภายนอกก็จะส่งข้อมูลใหม่ให้กับ 8255 ได้ ทำให้อินพุท บัฟเฟอร์ ฟูล เป็นจริงอีกครั้ง

ในรูปที่ 4.3 (B) เป็นตัวส่งข้อมูลให้กับอุปกรณ์ภายนอกก่อนที่ 8255 จะส่งข้อมูลให้กับอุปกรณ์ภายนอกนั้นจะต้องเซ็ทเอาท์พุท บัฟเฟอร์ ฟูล แฟลก (OUTPUT BUFFER FULL FLAG) ก่อน เพื่อบ่งบอกให้อุปกรณ์ภายนอกทราบว่า 8255 มีข้อมูลพร้อมที่จะส่งออกไปแล้ว เมื่ออุปกรณ์ภายนอกส่งสัญญาณ สโตรบ (STROBE) รับเอาข้อมูลเข้าไปแล้ว อินพุท บัฟเฟอร์ ฟูล แฟลก เป็นเท็จ เพื่อบอกให้อุปกรณ์ภายนอกรู้ว่า ขณะนี้ไม่มีข้อมูลอยู่ใน 8255 ซีพียูสามารถส่งข้อมูลใหม่ให้กับ 8255 ได้

วิธีการทำแฮนด์เช็คนี้มีประโยชน์มากในกรณีที่อุปกรณ์ภายนอกทำงานช้ากว่าระบบไมโครคอมพิวเตอร์ เช่นการการรับส่งข้อมูลระหว่าง คอมพิวเตอร์ กับ เครื่องพิมพ์ เมื่อโปรแกรม 8255 โหมด 1 พอร์ท C จะมีความหมายดังนี้

ขา	กรณีอินพุท	กรณีเอาท์พุท
PC0	INTRb	INTRb
PC1	IBFb	OBFb
PC2	STBb	ACKb
PC3	INTRa	INTRa
PC4	STBa	I / O
PC5	IBFa	I / O
PC6	I / O	ACKa
PC7	I / O	OBFa

ตารางที่ 4.1 สัญญาณที่สำคัญในการทำแฮนด์เชคในโหมด 1

ส่วนอินพุท

IBF เป็นเอาท์พุท ทำงานที่ลอจิก 1 เป็นตัวบอกว่าตอนนี้มีข้อมูลที่รับเข้ามาแล้ว และจะถูกรีเซ็ท เป็น 0 เมื่อซีพียู อ่านข้อมูลจาก 8255 ออกไปแล้ว (RESET ที่ขอบขาขึ้นของสัญญาณ RD

STB เป็น อินพุท ทำงานที่ลอจิก 0 ถ้า STB เป็น 0 8255 จะอ่านข้อมูลจากอุปกรณ์ภายนอกเข้ามา

INTR ปีนเอาท์พุท ทำงานที่ ลอจิก 1 จะทำเมื่อขาขอบขึ้นของ เอสทีบี โดยที่ไอบีเอฟต้องเป็น 1 และ ไอเอ็นทีอี จะต้องถูกอีนาเบิ้ล (ENABLE) ไว้

ส่วนเอาท์พุท

OBF เป็นเอาท์พุท ทำงานที่ลอจิก 0 เป็นตัวบอกว่าซีพียูได้ทำการเขียนข้อมูลไปยัง 8255 แล้ว พร้อมที่จะให้อุปกรณ์ภายนอกรับไปได้ โดยสัญญาณนี้จะทำงานที่ขอบขาขึ้นของสัญญาณ WR และจะถูก รีเซ็ท ก็ต่อเมื่อที่ขอบขาลงของสัญญาณ ACK

ACK เป็นอินพุท ทำงานที่ลอจิก 0 เป็นสัญญาณที่ส่งมาจากอุปกรณ์ภายนอกเพื่ออ่านข้อมูลจาก 8255

INTR เป็นเอาท์พุท ทำงานที่ลอจิก 1 และทำงานที่ขอบขาขึ้นของสัญญาณ ACK โดยที่โอบีเอฟ เป็น 0 และ ไอเอ็นทีอีถูกอีนาเบิ้ล

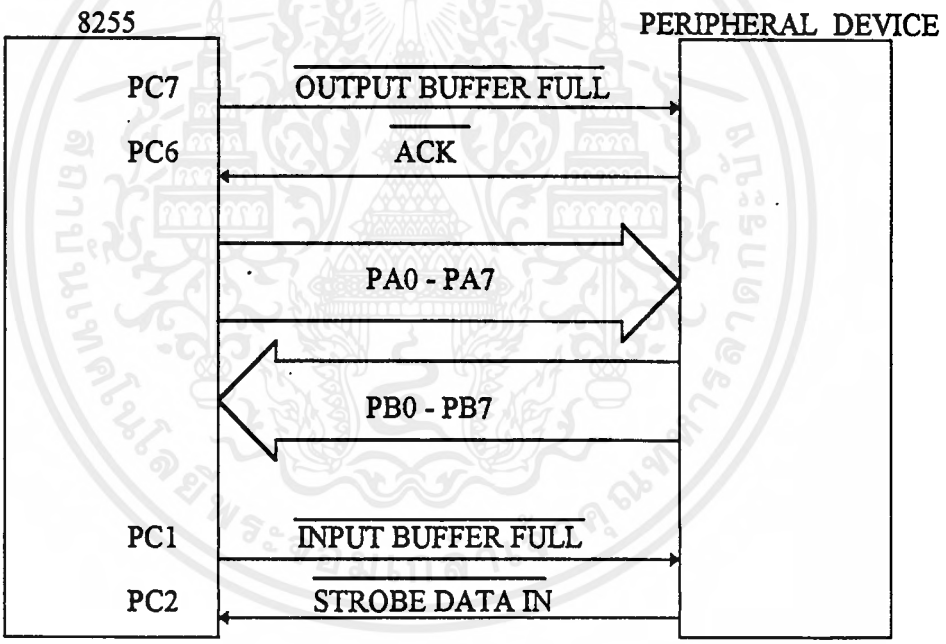
ตัวอย่างโปรแกรม

ตัวอย่างโปรแกรมนี้เป็นการทดลอง 8255 ในโหมด 1 โดยกำหนดให้พอร์ท A เป็นพอร์ทเอาท์พุท พอร์ท B เป็นพอร์ทอินพุท และ พอร์ท C เป็นแฮนด์เช็ค โดย 4 บิทบนเป็นแฮนด์เช็คให้กับพอร์ท A ส่วน บิทล่างเป็นแฮนด์เช็คให้กับพอร์ท B

ทำการรันโปรแกรมแล้วทำการทดสอบการแฮนด์เช็คได้ กรณีพอร์ท A เป็นพอร์ทเอาท์พุทคือ จะมีข้อมูลอยู่ที่ PA0-PA7 ของ 8255 ในการส่งข้อมูลไปยังอุปกรณ์ภายนอก 8255 จะต้องทำการตรวจสอบ เอาท์พุท บัฟเฟอร์ ฟูล จะต้องมีลอจิกเป็น 0 หรือ 1 สามารถทำการตรวจสอบค่าได้ โดยการใช้มิเตอร์หรือออสซิลโลสโคป วัดที่ขา PC7 ของ 8255 ถ้ามีลอจิกเป็น 1 อุปกรณ์ภายนอกได้รับข้อมูลออกไปแล้ว แต่ถ้ามีค่าเป็นลอจิก 0 ข้อมูลอยู่ใน 8255 ยังไม่ถูกอุปกรณ์ภายนอกอ่านออกไป

ถ้าอุปกรณ์ภายนอกพร้อมที่จะรับเอาข้อมูลที่อยู่ใน 8255 ทำการตรวจสอบได้โดยการใช้มิเตอร์วัดที่ขา PC6 ของ 8255 จะต้องมีลอจิกเป็น 0

พอร์ท B เป็นอินพุทพอร์ท ในการรับข้อมูลจากอุปกรณ์ภายนอกนั้นจะต้องทำการตรวจสอบ อินพุท บัฟเฟอร์ ฟูล ว่าพร้อมที่จะรับเอาข้อมูลหรือไม่ โยใช้มิเตอร์หรือออสซิลโลสโคปวัดที่ขา PC1 ของ 8255 ถ้ามีลอจิกเป็น 1 แสดงว่ารับเอาข้อมูลเข้ามาแล้วและจะทำการรีเซ็ทเป็น 0 เมื่อซีพียูอ่านข้อมูลจาก 8255 ออกไปแล้ว ถ้าอุปกรณ์ภายนอกพร้อมที่จะส่งข้อมูลให้ 8255 จะส่งสัญญาณสโตรบ (STROBE) ให้กับ 8255 ทำการตรวจสอบได้โดยการใช้มิเตอร์วัดที่ขา PC2 ของ 8255 จะต้องมีลอจิก 0 ซึ่งในการทดลองนี้สามารถแสดงได้ดังรูปที่ 4.4



รูปที่ 4.4 การเชื่อมต่อระหว่าง 8255 กับอุปกรณ์ภายนอกในลักษณะแฮนด์เช็คในโหมด 1

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมตามตัวอย่าง
2. ทำการรันโปรแกรม

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int     A=0x304,B=0x305,CtrP=0x303;
char   Data,CtrW=0xa3;
main(){
        outportb(CtrP,CtrW);
                Data=inportb(B);
                outportb(A,Data);
                getch();
}
```

ผลการทดลอง.....

.....

.....

สรุปผลการทดลอง.....

.....

.....

คำถาม

1. อธิบายวิธีการทำแฮนด์เช็ค
2. ให้เขียนโปรแกรมที่สามารถทำการแฮนด์เช็คมา 1 โปรแกรม

4.3 การทดลอง 8255 ในโหมด 2

การทำงานของ 8255 ในโหมด 2 เป็นการใช้งานในลักษณะทำให้ พอร์ท A เป็นพอร์ทข้อมูลแบบสองทิศทาง การทำงานใช้พอร์ท A เป็นอินพุทและเอาท์พุทแลทช์ (INPUT / OUTPUT LATCH) เอาท์พุทแลทช์ หมายถึง การส่งข้อมูลไว้เพื่อรออุปกรณ์ภายนอกรับเอา

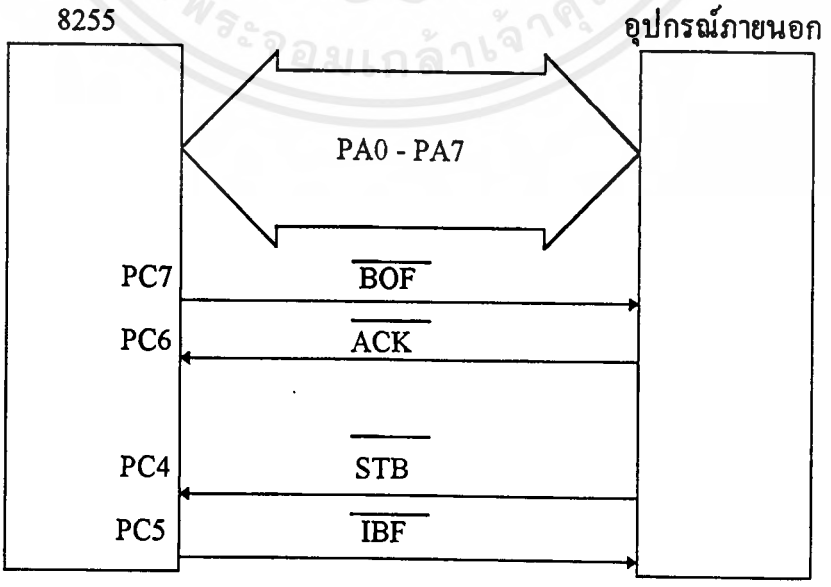
ข้อมูลออก ส่วนอินพุทแลทช์ หมายถึงการเก็บข้อมูลที่อุปกรณ์ภายนอกส่งเข้ามาเพื่อรอให้ซีพียูอ่านเข้าไปไปนั้น

ตัวอย่างโปรแกรม

ตัวอย่างโปรแกรมนี้เป็นการใช้งาน 8255 ในโหมดที่ 2 ซึ่งสามารถส่งข้อมูลให้อุปกรณ์ภายนอกและรับข้อมูลจากอุปกรณ์ภายนอกได้โดยการแลทช์ไว้ในพอร์ท A

ในการส่งข้อมูลให้อุปกรณ์ภายนอกนั้น สามารถทำการตรวจสอบได้ว่า 8255 มีข้อมูลพร้อมจะส่งหรือไม่ได้โดยใช้มิเตอร์หรือออสซิลโลสโคปวัดที่ขา PC7 ของ 8255 ถ้ามีลอจิก 0 แสดงว่ามีข้อมูลพร้อมที่จะส่งให้อุปกรณ์ภายนอก ถ้ามีลอจิกเป็น 1 แสดงว่าอุปกรณ์ภายนอกรับเอาข้อมูลไปแล้ว ในการส่งข้อมูลออกให้อุปกรณ์ภายนอกนั้นจะต้องตรวจสอบความพร้อมของอุปกรณ์ภายนอกเสียก่อน ถ้าอุปกรณ์ภายนอกพร้อมที่จะรับข้อมูลจะส่งสัญญาณ ACK (ACKNOWLEDGE) ให้กับ 8255 ซึ่งสามารถตรวจสอบได้โดยใช้มิเตอร์วัดที่ขา PC6 ของ 8255 จะต้องมีลอจิกเป็น 0

ในการรับข้อมูลจากอุปกรณ์ภายนอก อุปกรณ์ภายนอกจะต้องตรวจสอบ อินพุท บัฟเฟอร์ ฟูล เสียก่อนด้วยการส่งสัญญาณ สโตรบ (STROBE) มาทำการตรวจสอบว่าพร้อมจะรับข้อมูลหรือยัง ทำการตรวจสอบโดยการใช้มิเตอร์วัดที่ขา PC5 ของ 8255 ถ้าลอจิก 0 แสดงว่าพร้อมที่จะรับเอาข้อมูลจากอุปกรณ์ภายนอก และสามารถตรวจสอบสัญญาณสโตรบ โดยใช้มิเตอร์วัดที่ขา PC4 ของ 8255 ถ้าเป็นลอจิก 1 แสดงว่าพร้อมที่รับข้อมูลแล้ว ในการทดลองนี้สามารถแสดงได้ดังรูปที่ 4.5



รูปที่ 4.5 การเชื่อมต่อระหว่าง 8255 กับอุปกรณ์ภายนอกในโหมด 2

ลำดับขั้นการทดลอง

1. **เขียนโปรแกรมตามตัวอย่าง**
2. **ทำการรันโปรแกรม**

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int     A=0x304,C=0x302,CtrP=0x303;
char   Data=0x55,CtrW1=0xc8, CtrW2=0xc0;
main(){
        outportb(CtrP,CtrW1);
                outportb(A,Data);
        outportb(CtrP,CtrW2);
                Data=inportb(A);
        printf("Data=%x",Data);
        getch();
}
```

ผลการทดลอง.....

.....

.....

สรุปผลการทดลอง.....

.....

.....

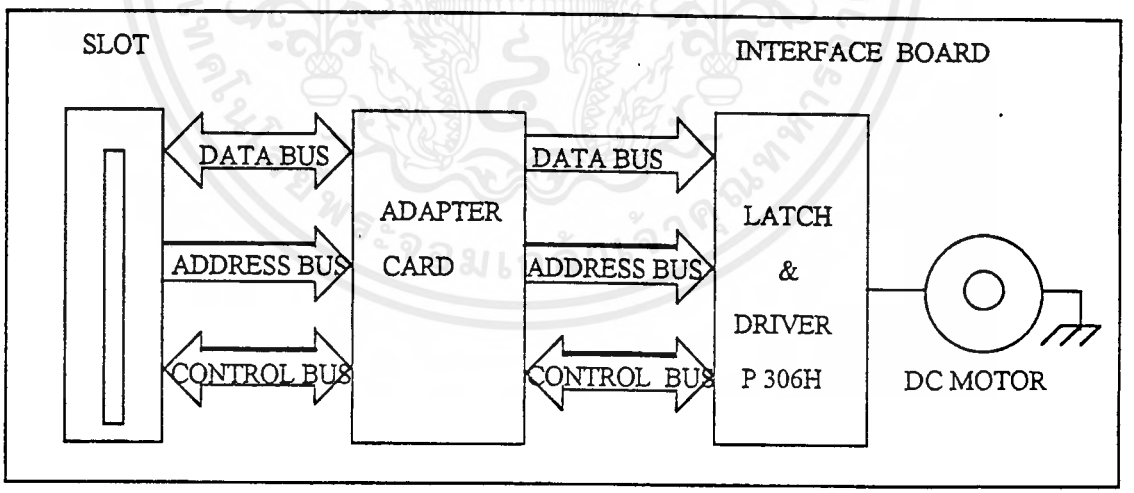
การทดลองที่ 5
การ อินเตอร์เฟส กับ ดีซีมอแตอร์

วัตถุประสงค์

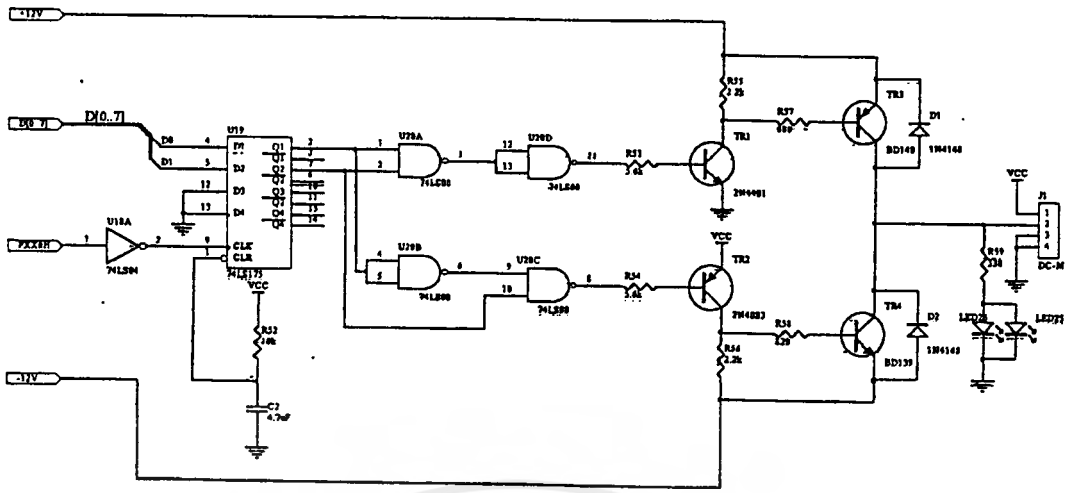
1. ให้นักศึกษาสามารถทำการเชื่อมต่อระหว่าง คอมพิวเตอร์ กับ ดีซีมอเตอรได้
2. ให้นักศึกษาสามารถเขียนโปรแกรมเพื่อควบคุมทิศทางของการหมุนของดีซีมอเตอร์ได้

ทฤษฎีเบื้องต้น

อุปกรณ์เอาท์พุทนอกจากจะควบคุมโดยใช้ รีเลย์ แล้ว ดีซีมอเตอร์ ถือว่าเป็นอุปกรณ์เอาท์พุทอย่างหนึ่งซึ่งสามารถควบคุมด้วยไมโครคอมพิวเตอร์ ซึ่งในการทดลองนี้จะเป็นตัวควบคุมทิศทางการหมุนของดีซีมอเตอร์ เพียงเท่านั้นยังไม่สามารถควบคุมตำแหน่งได้และดีซีมอเตอร์ที่ใช้ในการทดลองก็เป็นดีซีมอเตอร์ขนาดเล็กเท่านั้น แต่ก็สามารถใช้หลักการเดียวกันนี้ไปควบคุม ดีซีมอเตอร์ ขนาดใหญ่ได้



รูปที่ 5.1 ผังการทำงานของวงจร อินเตอร์เฟสกับดีซีมอเตอร์



รูปที่ 5.2 วงจรดีซีมอเตอร์เอาท์พุท

วงจรที่ใช้ในการทดลอง

การควบคุมการทำงานของดีซีมอเตอร์ ในการทดลองนี้ถูกดีโค้ดไว้ที่ พอร์ท 306H วงจรประกอบด้วยอุปกรณ์หลักๆ คือไอซี U19 IC 74LS175 , U20 IC 74LS00 , ทรานซิสเตอร์ ไดร์มอเตอร์ , ไดโอดเปล่งแสง (LED) แสดงทิศทางการหมุนของมอเตอร์

การทดลองของวงจรเริ่มจากถ้ามีการส่งข้อมูลมาที่พอร์ท 306H ซึ่งในวงจรจะใช้ข้อมูลจาก D0 และ D1 มาใช้ในการควบคุมทิศทางการหมุนของ ดีซีมอเตอร์ โดยทิศทางการหมุนสัมพันธ์กับข้อมูลที่ D0 และ D1 ดังแสดงในตาราง

ข้อมูล		มอเตอร์
D1	D0	
0	0	หยุด
0	1	หยุด
1	0	ตามเข็มนาฬิกา
1	1	ทวนเข็มนาฬิกา

ตารางที่5.1

สมมุติว่าทำการเขียนโปรแกรมเพื่อส่งข้อมูลไปยังพอร์ท &H306 ดังนี้

```
outportb(0x306,0x02);
```

จากโปรแกรมทำให้ D0 เป็นลอจิก 0 และ D1 เป็นลอจิก 1 ข้อมูล 0x02 จะถูกแลช (LATCH) ไว้ที่ไอซี U19 ทำให้ที่ขา 11 ของ U20 เป็นลอจิก 0 เป็นผลทำให้ Q4 ปิด(OFF) ; Q5 เปิด(ON) ; Q2 เปิด(ON) ; Q3 ปิด(OFF) ทำให้มีกระแสไหลจาก +12 V. ไปยัง ดีซีมอเตอร์ ทำให้ ดีซีมอเตอร์ หมุนแบบตามเข็มนาฬิกาและในการทำให้ ดีซีมอเตอร์หมุนแบบทวนเข็มนาฬิกา การทำงานของวงจรก็เช่นเดียวกันกับการทำงานในช่วง ตามเข็มนาฬิกา ต่างกันเพียงแต่การสลับการทำงานของทรานซิสเตอร์เท่านั้น

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีโปรแกรมภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมต่อไปนี้ เปลี่ยนค่าข้อมูล ตามตารางที่7.1 EXECUTE โปรแกรมทุกครั้งที่เปลี่ยนข้อมูล บันทึกผลการทดลอง

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int   Port=0x306,Data=0x00;
main(){
        outportb(Port,Data);
}
```

ข้อมูล(Data)	ทิศทางการหมุน
0x00	
0x01	
0x02	
0x03	

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

คำถาม

1. จงเขียนโปรแกรมภาษาซีเพื่อรับข้อมูลจากคีย์บอร์ดของเครื่องคอมพิวเตอร์เพื่อควบคุมทิศทางการหมุนของ ดีซีมอเตอร์ กำหนดให้ คีย์ R หมุนทวนเข็มนาฬิกา คีย์ F หมุนตามเข็มนาฬิกา และ คีย์ S หยุดหมุน

การทดลองที่ 8
การ อินเตอร์เฟส กับ สเต็บปิ้งมอเตอร์

วัตถุประสงค์

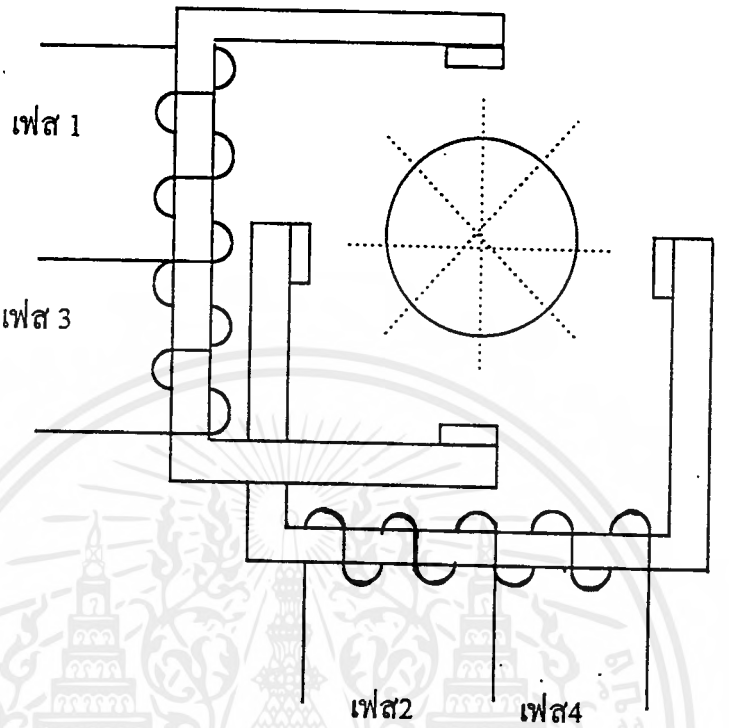
1. เพื่อให้นักศึกษาสามารถทำการอินเตอร์เฟสระหว่างคอมพิวเตอร์กับสเต็บปิ้งมอเตอร์ได้
2. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมเพื่อควบคุมลักษณะการหมุนของสเต็บปิ้งมอเตอร์ได้

ทฤษฎีเบื้องต้น

สเต็บปิ้งมอเตอร์ ถือว่าเป็นอุปกรณ์เอาท์พุทอย่างหนึ่งซึ่งสามารถควบคุมได้ด้วยไมโครคอมพิวเตอร์ ลักษณะการทำงานของ สเต็บปิ้งมอเตอร์จะเคลื่อนที่เป็นขั้น (STEP) ซึ่งอาจเป็นขั้นละ 1.8 , 5 , 7.5 องศา ก็แล้วแต่ชนิดของมอเตอร์ส่วนใหญ่ สเต็บปิ้งมอเตอร์จะใช้ในการควบคุมระบบดิจิตอล เช่น อุปกรณ์ประกอบคอมพิวเตอร์ต่างๆ เช่น ปริ้นเตอร์) , พล็อตเตอร์ , ดีสก์-ไดร์ฟ ตลอดจนอุปกรณ์ในระบบงานอิเล็กทรอนิกส์อุตสาหกรรม หรือเครื่องมือวัดและระบบควบคุมอื่นๆ

มีส่วนประกอบสำคัญ 2 ส่วน คือ

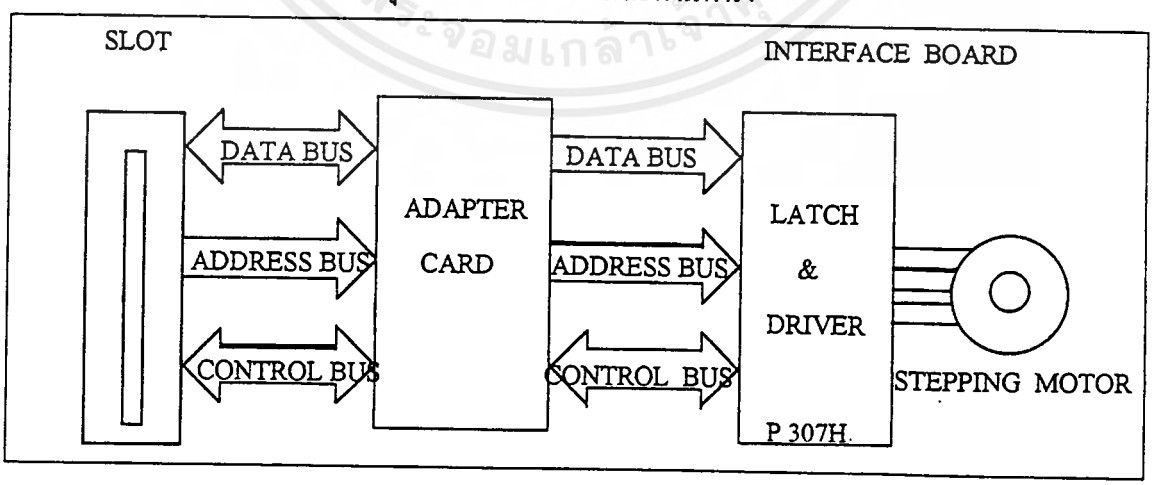
1. ส่วนที่หมุนได้ (ROTOR) จะเป็นแม่เหล็กถาวร หรืออื่นๆ
2. ส่วนที่อยู่กับที่ (STATOR) เป็นขดลวดหลายๆ ขด



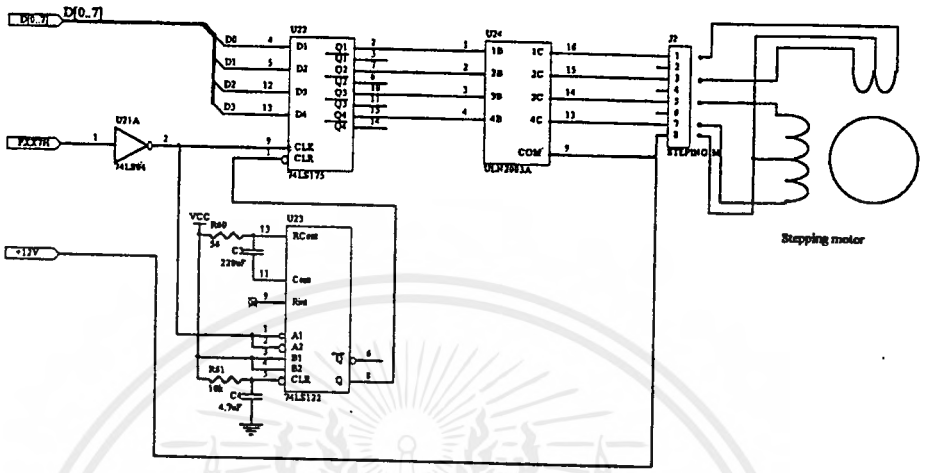
รูปที่ 6.1 สเต็บปิ้งมอเตอร์ 4 เฟสแบบ ยูนิโพลาร์ เพอร์มาแนนท์ แม็คเน็ต

วงจรที่ใช้ในการทดลอง

การทดลองนี้จะเป็นการทดลองใช้คอมพิวเตอร์ ควบคุมการหมุนของสเต็บปิ้งมอเตอร์ ซึ่งจะสามารถควบคุมได้ทั้งทิศ0ทาง และตำแหน่ง



รูปที่ 6.2 ผังการทำงานของวงจรอินเตอร์เฟส สเต็บปิ้งมอเตอร์



รูปที่ 6.3 วงจรสเต็บปิ้งมอเตอร์

จากวงจรประกอบด้วยอุปกรณ์หลักๆ คือ U22 74LS175 , U24 ULN2033 , U23 74LS122

หน้าที่ของอุปกรณ์แต่ละตัวมีดังนี้

- U22 74LS175 ทำหน้าที่ค้างข้อมูลที่ส่งมาที่ พอร์ท 307H
- U23 74LS122 เป็นวงจรรีทริกเกอร์ริ่ง โมโนสเตเบิ้ล มัลติไวเบรเตอร์ ทำหน้าที่เคลียร์ U22 ในกรณีที่ไม่มีการส่งข้อมูลมาที่ พอร์ท 307H เพื่อไม่ให้ U22 ค้างข้อมูลตลอดเวลา เพราะถ้ามีการค้างข้อมูลที่ลอจิก 1 ที่ U22 จะทำให้มีกระแสไหลผ่านสเต็บปิ้งมอเตอร์ ตลอดเวลาเช่นเดียวกัน จะทำให้สเต็บปิ้งมอเตอร์ ร้อนขึ้นเรื่อยๆ จึงต้องทำการเคลียร์ ข้อมูลที่ U22
- U24 ULN2003 เป็นตัวไดร์เวอร์ ทำหน้าที่ในการจ่ายกระแสให้แก่ขดลวดของสเต็บปิ้งมอเตอร์

การทำงานเริ่มจากการส่งข้อมูลไปที่ พอร์ท 307H ต่อเนื่องเป็นลำดับกันไป ซึ่งข้อมูลที่ส่งออกไปจะเป็นอะไรก็แล้วแต่ความต้องการที่จะให้สเต็บปิ้งมอเตอร์หมุนไปลักษณะไหน ก็ทำการส่งข้อมูลตามตารางดังที่กล่าวมาแล้วข้างต้น

ตัวอย่างโปรแกรม เพื่อให้ สเต็บปิ้งมอเตอร์หมุน

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    Port=0x307,Delay=100;
char  a,Data[]={0x01,0x02,0x04,0x08};
main(){
        while(!kbhit()) {
        outportb(Port,Data[a++]);
        if(a>=4)  a=0;
        delay(Delay); }
}
```

จากตัวอย่างโปรแกรมเป็นการทำให้ สเต็บปิ้งมอเตอร์ หมุนตามเข็มนาฬิกา ถ้าต้องการทำให้ สเต็บปิ้งมอเตอร์ หมุนทวนเข็มนาฬิกา ก็สามารถทำได้โดยการเปลี่ยนข้อมูลใน70 เป็น Data[ ]={0x08,0x04,0x02,0x01} ตามลำดับ หรือถ้าต้องการให้ สเต็บปิ้งมอเตอร์ หมุนช้าลงหรือเร็วขึ้น ก็สามารถทำได้โดยการเปลี่ยนค่าของดีเลย์ (Delay)

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

ป้อนโปรแกรมดังต่อไปนี้ รัน โปรแกรมแล้วสังเกตผลที่เกิดขึ้น

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    Port=0x306,Delay=100;
char  a,Data[]={0x01,0x02,0x04,0x08};
main(){
        while(!kbhit()) {
        outportb(Port,Data[a++]);
        if(a>=4)  a=0;
        delay(Delay); }
}
```

ผลการทดลอง.....

.....

.....

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

คำถาม

1. จากโปรแกรมในข้อที่ 1 ถ้าต้องการทำให้สเต็บปิ้งมอเตอร์ หมุนกลับทาง จะต้องเขียนโปรแกรมอย่างไร
2. ถ้าต้องการให้ สเต็บปิ้งมอเตอร์ หมุน 1 รอบ (360) แล้วหยุดจะต้องเขียนโปรแกรมอย่างไร
3. จงเขียนโปรแกรมเพื่อรับข้อมูลจากคีย์บอร์ด เพื่อควบคุมทิศทางการหมุนของสเต็บปิ้งมอเตอร์ โดยกำหนดให้ คีย์ F หมุนตามเข็มนาฬิกา คีย์ R หมุนทวนเข็มนาฬิกา คีย์ S หยุดหมุน

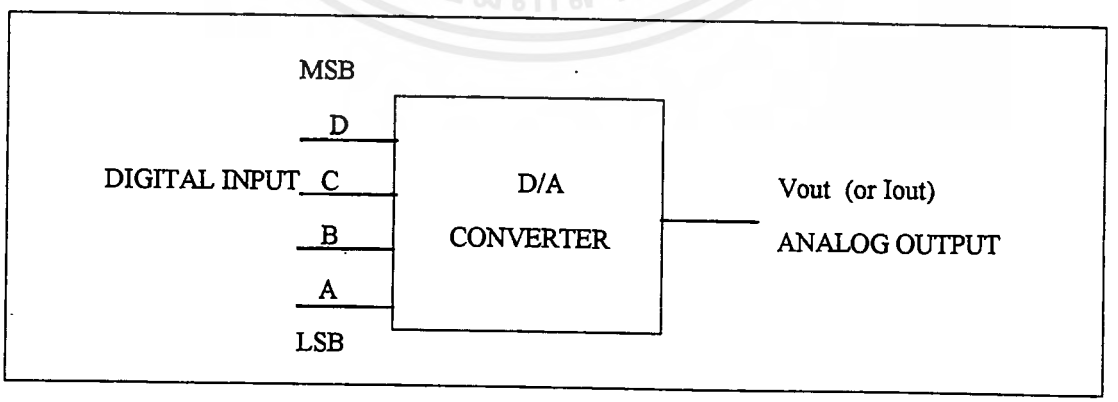
การทดลองที่ 7
วงจรแปลงสัญญาณดิจิตอลเป็นแอนาลอก
(DIGITAL TO ANALOG CONVERTOR)

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถอธิบายการทำงานของวงจรแปลงสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก
2. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมเพื่อควบคุมการทำงานของวงจรแปลงสัญญาณดิจิตอลเป็นแอนาลอกได้
3. เพื่อให้นักศึกษาสามารถประยุกต์ใช้งานวงจรแปลงสัญญาณดิจิตอลเป็นแอนาลอกได้

ทฤษฎีเบื้องต้น

การควบคุมอุปกรณ์ในงานอุตสาหกรรม มักจะต้องเกี่ยวกับสัญญาณที่เป็น สัญญาณแอนาลอก ซึ่งสัญญาณแอนาลอกก็คือสัญญาณที่มีความแตกต่างหลายๆระดับ เช่น ระดับแรงดัน หรือกระแส ในการควบคุมอุปกรณ์ที่ต้องการสัญญาณแอนาลอกโดยใช้ไมโครโปรเซสเซอร์ หรือ ไมโครคอมพิวเตอร์ สัญญาณจากไมโครโปเซสเซอร์ หรือ ไมโครคอมพิวเตอร์ จะเป็นสัญญาณแอนาลอก ดั้งนั้นจึงต้องมีการเปลี่ยนสัญญาณดิจิตอลให้เป็นสัญญาณแอนาลอก เพื่อนำไปใช้ควบคุมอุปกรณ์เหล่านั้นต่อไป



รูปที่ 7.1 ผังการทำงานของสัญญาณดิจิตอลเป็นแอนาลอก ขนาด 4 บิท

ผังการทำงานของวงจรแปลงสัญญาณดิจิตอลเป็นแอนาลอก แบบ 4 บิท แอนาลอกเอาท์พุทที่ได้จะอยู่ในรูปของ แรงดันหรือกระแส

D	C	B	A	Vout (VOLTS)
0	0	0	0	0
0	0	0	1	1
0	0	1	0	2
0	0	1	1	3
0	1	0	0	4
0	1	0	1	5
0	1	1	0	6
0	1	1	1	7
1	0	0	0	8
1	0	0	1	9
1	0	1	0	10
1	0	1	1	11
1	1	0	0	12
1	1	0	1	13
1	1	1	0	14
1	1	1	1	15

ตารางที่ 7.1 ค่าแรงดันเอาท์พุทที่ใช้ในรูปของแอนาลอก

ตารางที่7.1 แสดงให้เห็นถึงความสัมพันธ์โดยตรงระหว่างสัญญาณทั้งสอง เช่นที่สัญญาณดิจิตอลอินพุท 1001 ซึ่งมีค่าเท่ากับ 9 ในระบบเลขฐานสิบ จะมีแรงดันเป็นสัญญาณแอนาลอกเท่ากับ 9 V. แต่ความจริงแล้วอาจมีค่าเป็น 2 เท่าหรือเท่ากับเท่าไหร่ก็ได้แล้วแต่การออกแบบวงจร ว่าจะให้มีอัตราส่วนของสัญญาณทั้งสองเป็นเท่าไร

อินพุทเวท (INPUT WEIGHTS)

อินพุทเวท คือ น้ำหนักประจำตำแหน่งแต่ละบิท ของสัญญาณดิจิตอล

MSB D	C	B	LSB A	WEIGHTS
0	0	0	1	1
0	0	1	0	2
0	1	0	0	4
1	0	0	0	8

ตารางที่ 7.2 แสดงน้ำหนักในแต่ละบิท ของสัญญาณดิจิตอล

จากตารางที่ 7.2 แสดงน้ำหนักในแต่ละบิท

ที่บิท A มีน้ำหนักเท่ากับ $1 * 2 = 1$

บิท B มีน้ำหนักเท่ากับ $1 * 2 = 2$

บิท C มีน้ำหนักเท่ากับ $1 * 2 = 4$

และ บิท D มีน้ำหนักเท่ากับ $1 * 2 = 8$

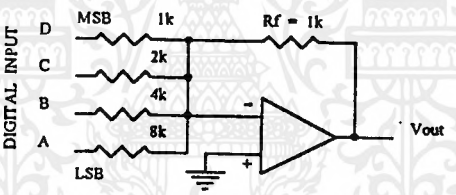
พิจารณาดูน้ำหนักแต่ละบิทจะมีน้ำหนักมากว่ากันเป็น 2 เท่า โดยเริ่มจากบิท LSB ดังนั้นแรงดันเอาท์พุทจะมีค่าเท่ากับผลรวมของน้ำหนักแต่ละบิท เช่นที่สัญญาณดิจิตอล 1001 จะมีแรงดันเอาท์พุทเท่ากับ $(1*8)+ (0*4)+ (0*2)+ (1*1) = 8+1 = 9$ V.

รีโซลูชั่น (RESOLUTION ; STEP SIZE)

รีโซลูชั่น คือการเปลี่ยนแปลงต่ำสุดของสัญญาณแอนาลอก จากระดับแรงดันหนึ่งไปยังอีกระดับแรงดันหนึ่งเมื่อสัญญาณดิจิตอลอินพุทเปลี่ยนแปลงไป พิจารณาตารางรูปที่ 2 จะพบว่าในการเปลี่ยนแปลงของสัญญาณดิจิตอลแต่ละครั้ง จะทำให้สัญญาณแอนาลอกเปลี่ยนไป 1 V. เรียกค่านี้ว่า รีโซลูชั่นหรือ สเต็ปไซซ์ ปกติแล้วค่ารีโซลูชั่นจะมีค่าเท่ากับน้ำหนักของ LSB เสมอ

วิธีการเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก D/A CONVERTER CIRCUITRY

วิธีการเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก มีด้วยกันหลายวิธี แต่วิธีการอันหนึ่งที่นิยมกันได้แก่ การใช้วงจรตามรูปที่ 7.2

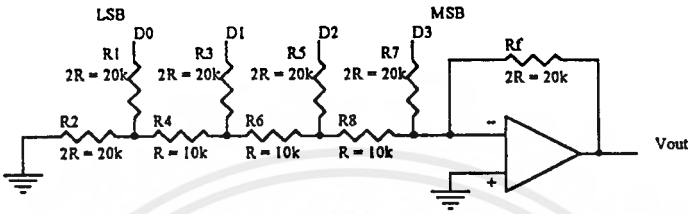


รูปที่ 7.2 วงจรวงจรขยายออปแอมป์แบบซัมมิ่ง ที่ใช้ทำหน้าที่เป็นวงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

วงจรตามรูปที่ 7.2 เป็นวงจรขยายซัมมิ่ง มีอัตราการขยายสูง มีเอาท์พุทอิมพิแดนซ์ต่ำและอินพุทอิมพิแดนซ์สูงมาก ทั้งนี้เพื่อจะไม่ให้โหลดวงจรมากเกินไป ค่าแรงดันเอาท์พุท (Vout) จะหาได้จากสูตร

$$Vout = -(Vd + 0.5Vc + 0.25Vb + 0.125Va)$$

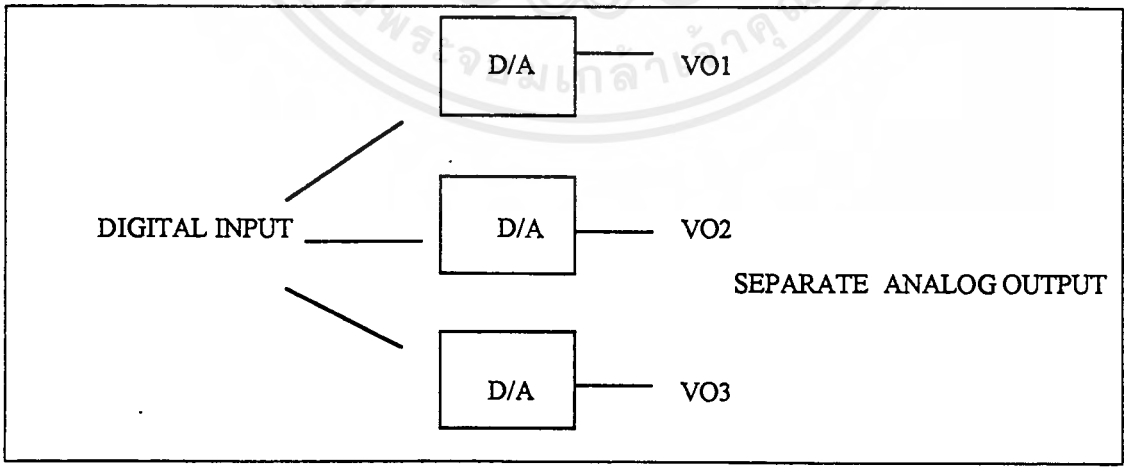
วงจรตามรูปที่ 7.2 นั้น จะมีปัญหาเกิดขึ้นในกรณีที่สัญญาณดิจิตอลมีจำนวนบิทมากขึ้น เนื่องจากการหาค่าความต้านทานที่มีขายตามท้องตลาดได้ลำบาก จึงนิยมใช้วิธีการของ R/2R LADDER แทน ดังรูปที่ 7.3



รูปที่ 7.3 วงจรแปลงสัญญาณดิจิตอลเป็นแอนาลอกแบบ R/2R LADDER

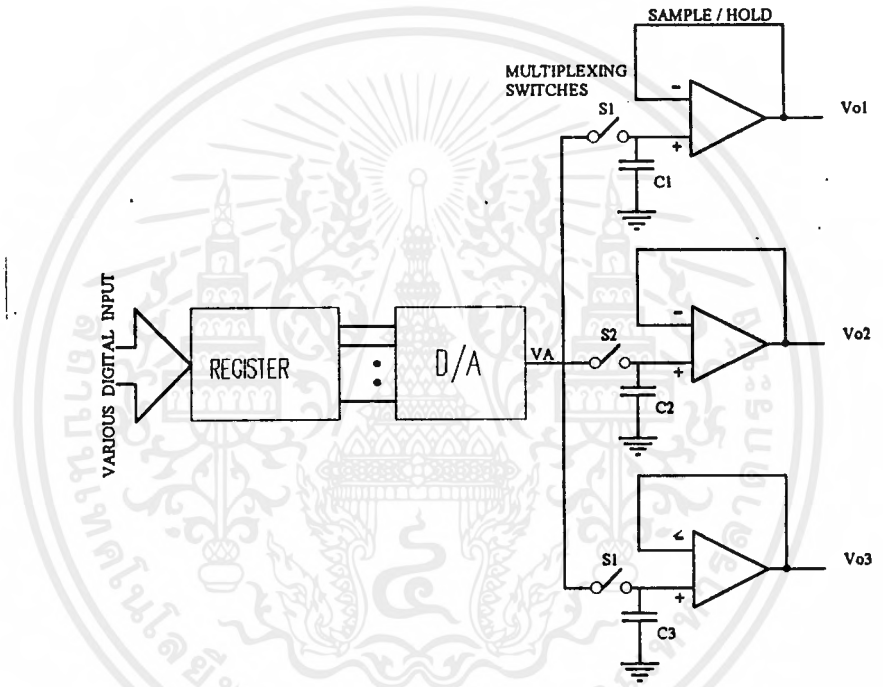
<u>วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกแบบมัลติเพล็กซ์</u>

ในกรณีการใช้งานที่มีดิจิตอลอินพุทหลายๆ ชุดเพื่อนำไปใช้ควบคุมอุปกรณ์หลายชนิด เช่น ใช้ควบคุมมอเตอร์ หรือวาวล์ต่างๆ การต่อวงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกจะกระทำได้สองแบบ คือ แบบแรกได้แก่การใช้วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก 1 ชุดต่อสัญญาณดิจิตอลอินพุท 1 ชุด แบบที่สอง ได้แก่ แบบมัลติเพล็กซ์



รูปที่ 7.4วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกแบบเซพเพอร์เรท

วงจรรูปที่ 7.4 เรียกว่าวงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกแบบเซพเพอร์เรท วงจรแบบนี้จะมีข้อดีคือสัญญาณดิจิตอลอินพุทแต่ละชุดจะถูกป้อนให้แก่ วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอกแต่ละชุดอย่างต่อเนื่อง วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก แต่ละชุดจะแยกกันทำงานโดยอิสระไม่ขึ้นต่อกัน ดังนั้นจึงไม่จำเป็นต้องใช้อุปกรณ์หน่วยความจำเลย อุปกรณ์ต่างๆ เช่น ตัวต้านทาน หรือ แอมปริไฟร์ต่างๆต้องมีความแม่นยำสูงเพื่อเป็นการแก้ไขข้อเสียของแบบแรก จึงได้อาศัยหลักการของมัลติเพล็กซ์ เพื่อลดความสิ้นเปลืองลง ดังแสดงให้เห็นตามรูปที่ 7.5



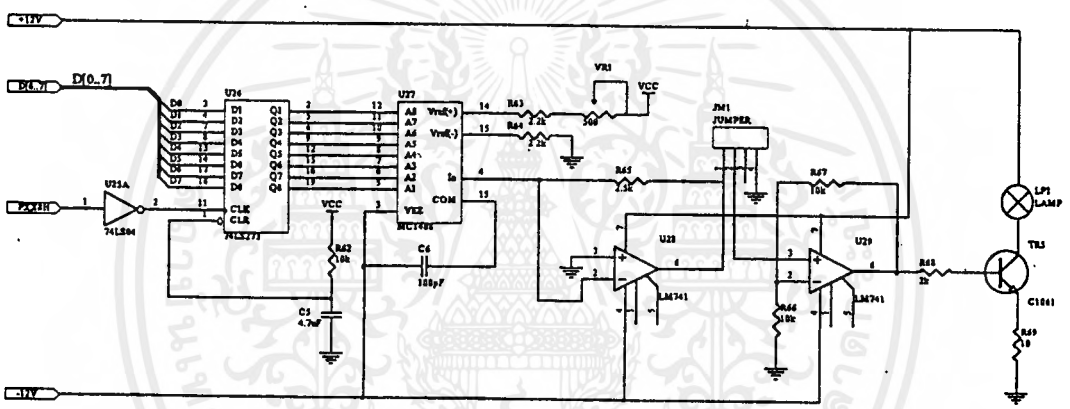
รูปที่ 7.5 วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก อาศัยหลักการของมัลติเพล็กเซอร์

หลักการของมัลติเพล็กซ์ก็คือ การใช้วิธีของช่วงเวลาชาริ่ง (TIME-SHARING) กล่าวคือในช่วงเวลาต่างๆ จะผลัดการทำงานเมื่อมีอินพุทนั้นเข้ามา เช่นครั้งแรกทำงานที่อินพุท A ต่อมาทำงานที่สัญญาณอินพุท B เป็นต้น ด้วยวิธีการนี้เองทำให้วงจรนี้สามารถใช้วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก เพียงชุดเดียวเท่านั้น

อัตราการมัลติเพล็กซ์ (MULTIPLEXING RATE)

อัตราการมัลติเพล็กซ์ (MULTIPLEXING RATE) หมายถึงการทำงานในหนึ่งรอบของดิจิตอลอินพุท แต่ละชุด การทำงานจะเริ่มจากการเปลี่ยนข้อมูลสัญญาณดิจิตอลอินพุทชุดใหม่เข้าไปยังรีจิสเตอร์ และชุด วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก ทำการเปลี่ยนข้อมูลนี้เป็นสัญญาณแอนาลอก Va มัลติเพล็กซ์สวิทช์ต่อวงจรที่เหมาะสม และเปิดสวิทช์อันอื่น

วงจรที่ใช้ในการทดลอง



รูปที่ 7.6 วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก

จากวงจรประกอบด้วย U26 ไอซี 74LS273 , U27 ไอซี MC1408 , U28 ไอซี LM741 ประกอบกับ R-C เพื่อทำหน้าที่เป็นวงจรเปลี่ยนสัญญาณจาก ดิจิตอล เป็นสัญญาณแอนาลอก ซึ่งวงจร วงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก ในวงจรนี้ถูก ดีโค้ดไว้ที่พอร์ท 308H วงจรข้างต้นจะให้สัญญาณ เอาท์พุทเป็นระดับแรงดันที่เปลี่ยนแปลงไปตามการเปลี่ยนแปลงของสัญญาณแอนาลอกที่
เข้ามาที่เอาท์พุทของวงจร.

การทำงานของวงจรเริ่มต้นจาก เมื่อส่งข้อมูลมาที่พอร์ท 308H ด้วยคำสั่ง

```
outportb(0x308,0x80);
```

U26 จะทำหน้าที่ค้างข้อมูล 80H ไว้ที่ Q1-Q8 จากนั้น U27 จะทำการแปลงข้อมูลนี้ให้เป็นสัญญาณแอนาลอกซึ่งอยู่ในรูปของกระแส ปรากฏที่ขา 4 (ขา Iout) U28 จะทำการเปลี่ยนกระแสนี้ให้อยู่ในรูปของแรงดันส่งออกไปที่ JM1

สัญญาณจาก JM1 จะต่อเข้ากับวงจรขับหลอดไฟ ขนาด 12V. เพื่อให้เป็นความสัมพันธ์ระหว่างสัญญาณดิจิตอลที่เข้ามา กับสัญญาณแอนาลอกที่ได้ โดยสังเกตจากความสว่างของหลอดไฟ

<u>อุปกรณ์การทดลอง</u>

1. เครื่องคอมพิวเตอร์ที่มีภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์.
5. ดีซี.โวลท์มิเตอร์

<u>ลำดับขั้นการทดลอง</u>

1. เสียบจุดต่อบนบอร์ดอินเตอร์เฟสส่วนของวงจรเปลี่ยนสัญญาณดิจิตอลเป็นสัญญาณแอนาลอก (JUMPER) ดังรูป

2. ป้อนโปรแกรมต่อไปนี้ แล้วรันโปรแกรม แล้วเปลี่ยนค่าตามตารางที่ 7.3 สังเกตความสว่างของหลอดไฟ และแรงดันที่วัดจากขา 1,2 ของJM 1 บันทึกผลที่เกิดขึ้น

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    Port=0x308;
char  Data=0xff;
main(){
        outportb(Port,Data); }
```

ข้อมูล(Data)	แรงดัน(V)
0x20	
0x30	
0x40	
0x50	
0x60	
0x70	
0x80	
0x90	
0xa0	
0xb0	
0xc0	
0xd0	
0xe0	
0xf0	

ตารางที่7.3

ผลการทดลอง.....

.....

.....

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

.....

คำถาม

1. จงอธิบายความสัมพันธ์ของสัญญาณดิจิตอลกับระดับแรงดันเอาท์พุท ที่ได้จากการทดลอง
2. ถ้าต้องการเปลี่ยนระดับแรงดันเอาท์พุทสูงสุดของวงจรจะต้องดัดแปลงวงจรอย่างไร
3. จงเขียนโปรแกรมเพื่อให้หลอดไฟสว่างจากมากไปหาน้อย

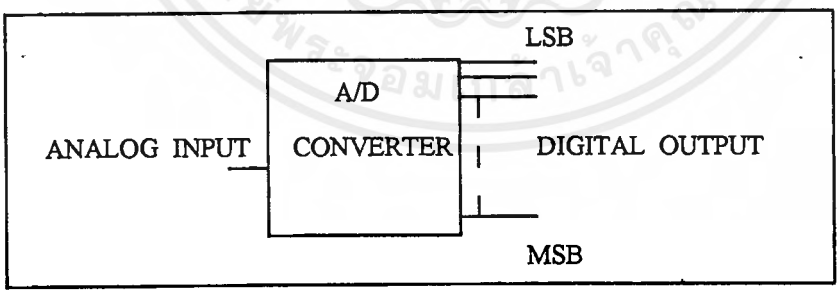
การทดลองที่ 8
วงจรแปลงสัญญาณจาก แอนาลอกเป็น ดิจิตอล
(ANALOG TO DIGITAL CONVERTER)

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถอธิบายการทำงานการทำงานของวงจรแปลงสัญญาณจาก แอนาลอกเป็น ดิจิตอลได้
2. เพื่อให้นักศึกษาสามารรถเขียนโปรแกรมเพื่อควบคุมการทำงานของวงจรแปลงสัญญาณจาก แอนาลอกเป็น ดิจิตอลได้
3. เพื่อให้นักศึษาสามารถประยุกต์ใช้งานวงจรแปลงสัญญาณจาก แอนาลอกเป็นดิจิตอลได้

ทฤษฎีเบื้องต้น

การเก็บข้อมูลจากอุปกรณ์เซ็นเซอร์ บางชนิด เช่น อุปกรณ์เซ็นเซอร์อุณหภูมิ ความดัน น้ำหนัก เป็นต้น ปกติข้อมูลเหล่านี้ไม่สามารถเก็บไว้ในหน่วยความจำ หรือนำไปประมวลผลด้วยไมโครโปรเซสเซอร์ได้เพราะเป็นสัญญาณแอนาลอก ดังนั้นจึงต้องมีการแปลงสัญญาณเหล่านี้ ให้อยู่ในรูปของสัญญาณดิจิตอล

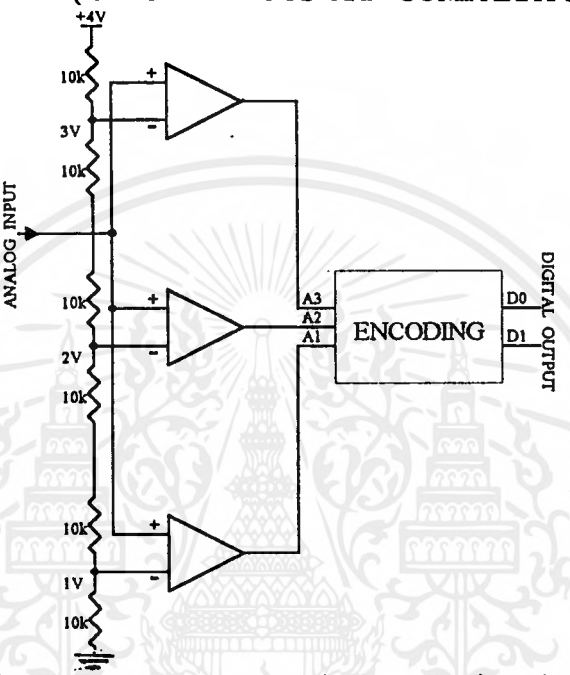


รูปที่8.1 ผังการทำงานของวงจรแปลงสัญญาณจาก แอนาลอกเป็น ดิจิตอล

<u>วงจรแปลงสัญญาณจากแอนาลอกเป็นดิจิตอลเปรียบเทียบแบบขนาน</u>

(PARALLEL COMPARATOR A/D CONVERTER)

วงจรวงจรแปลงสัญญาณจากแอนาลอกเป็นดิจิตอล แบบง่ายที่สุดได้แก่แบบ เปรียบเทียบแบบขนาน หรือเรียกอีกชื่อหนึ่งว่า วงจรแปลงสัญญาณจากแอนาลอกเป็นดิจิตอลเปรียบเทียบเกิดขึ้นในเวลาเดียวกัน (SIMULTANEOUS A/D COMPARATOR) ดังรูปที่ 8.2



รูปที่8.2 วงจรแปลงสัญญาณจากแอนาลอกเป็นดิจิตอลเปรียบเทียบแบบขนาน

	COMPARATOR OUTPUT			DIGITAL OUTPUT	
Vin	A1	A2	A3	D1	D0
0 1	0	0	0	0	0
1 - 2	1	0	0	0	1
2 - 3	1	1	0	1	0
3 - 4	1	1	1	1	1

ตารางที่8.1 ค่าสัญญาณที่จุดต่างๆเมื่อ Vin เปลี่ยนแปลง

จากรูปที่8.2 และตารางที่8.1 สามารถอธิบายการทำงานของวงจรได้ดังนี้

ตัวต้านทาน 10 Kohm จำนวน 4 ตัวจะทำหน้าที่เป็นวงจรแบ่งแรงดันให้แก่อินพุทขั้วลบของออปแอมป์ แต่ละตัวซึ่ง ออปแอมป์ นี้เป็นตัวเปรียบเทียบแรงดัน (VOLTAGE COMPARATOR) ดังนั้นแรงดันที่อินพุทขั้วลบก็คือแรงดันอ้างอิง (REFERENCE VOLTAGE) ของตัวเปรียบแต่ละตัวถ้าแรงดันที่อินพุทขั้วบวกมีค่าสูงกว่าแรงดันอ้างอิง ก็จะทำให้เอาท์พุทของตัวเปรียบเทียบเปลี่ยนสภาวะจาก 0 เป็น 1

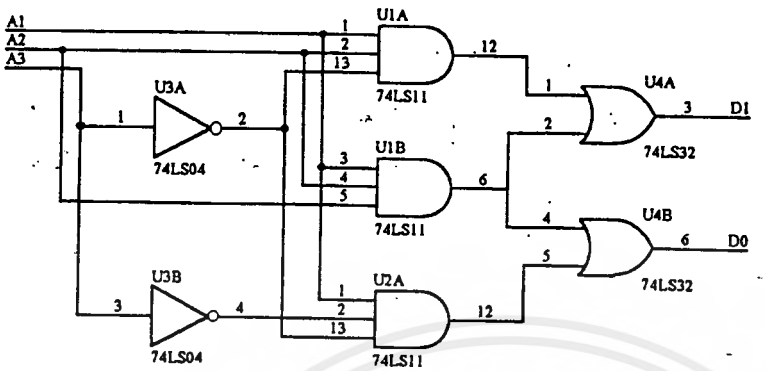
จากวงจรจะพบว่าวงจรแบ่งแรงดันจะแบ่งแรงดันให้แก่อินพุทขั้วลบ ของตัวเปรียบเทียบ A3 เท่ากับ 3V. ; A2 เท่ากับ 2V. ; A1 เท่ากับ 1V. และเอาท์พุทของตัวเปรียบแต่ละตัวจะต่อเข้ากับวงจรถอดรหัส (ENCODING GATES) เพื่อทำหน้าที่ถอดรหัสในรูปเลขฐานสอง พิจารณาดูที่อินพุท Vin จากตารางรูปที่ 3 เมื่อ Vin เท่ากับ 0 ถึง 1V. เอาท์พุทของวงจรเปรียบเทียบจะอยู่ในสภาวะ 0 เพราะแรงดันไม่สูงกว่าแรงดันอ้างอิงของตัวเปรียบเทียบใด ต่อมา Vin = 1-2 V. จะพบว่าแรงดัน Vin จะมากกว่าแรงดันของตัวเปรียบเทียบ A1 ดังนั้นเอาท์พุท A1 จะเปลี่ยนมาอยู่ในสภาวะ 1 ในทำนองเดียวกันก็จะได้เอาท์พุท A1,A2,A3 ตามตารางที่ 3 โดยพิจารณา Vin เช่นเดียวกับที่กล่าวมาแล้วข้างต้น คือยึดหลักว่าแรงดันที่อินพุทขั้วบวกมากกว่าแรงดันอ้างอิงที่อินพุทขั้วลบ เอาท์พุทของตัวเปรียบนั้นก็จะเปลี่ยนสภาวะเป็น 1 สำหรับเอาท์พุท D1 - D0 ก็มีวิธีการออกแบบดังนี้

เขียน ฟังก์ชั่นบูลีน (BOOLEAN FUNCTION) ของ D0 และ D1 โดยดูจากเอาท์พุทในตารางที่10.1 ที่เป็น 1

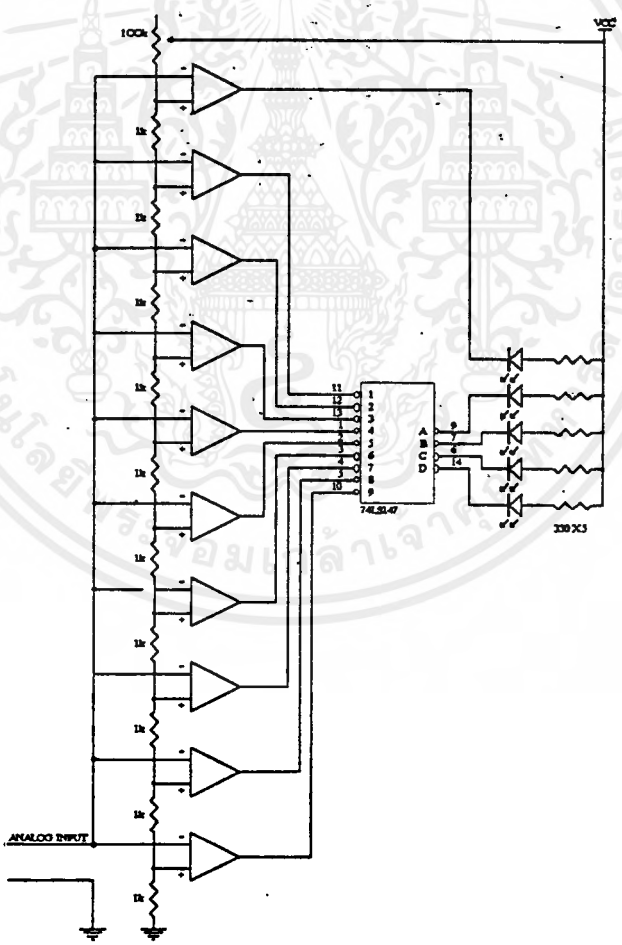
$$D1 = A1A2\overline{A3} + A1A2A3$$

$$D0 = \overline{A1}\,\overline{A2}A3 + A1A2\overline{A3}$$

เขียนลอจิกไดอะแกรมได้ดังรูปที่ 8.3



รูปที่ 8.3 วงจรถอดรหัส



รูปที่ 8.4 วงจรแปลงสัญญาณจากแอนาลอกเป็นดิจิตอลเปรียบเทียบแบบขนานแสดงผลเป็นรหัสไบนารี่ โค้ด เดซิมอล (BCD CODE)

รูปที่ 8.4 เป็นวงจรวงจรแปลงสัญญาณจากอนาล็อกเป็นดิจิตอลเปรียบเทียบแบบขนาน แสดงผลเป็น รหัสไบนารี่ โค้ด เดซิมอล (BCD CODE) โดยใช้ไอซีเบอร์ 74LS147 เป็นตัวถอดรหัส ไอซีเบอร์ 74LS147 เป็นไอซีทีทีแอล มีตารางการทำงานตามรูปที่ 8.2

INPUTS									OUTPUTS			
1	2	3	4	5	6	7	8	9	D	C	B	A
H	H	H	H	H	H	H	H	H	H	H	H	H
X	X	X	X	X	X	X	X	L	L	H	H	L
X	X	X	X	X	X	X	L	H	L	H	H	H
X	X	X	X	X	X	L	H	H	H	L	L	L
X	X	X	X	X	L	H	H	H	H	L	L	H
X	X	X	X	L	H	H	H	H	H	L	H	L
X	X	X	L	H	H	H	H	H	H	L	H	H
X	X	L	H	H	H	H	H	H	H	H	L	L
X	L	H	H	H	H	H	H	H	H	H	L	H
L	H	H	H	H	H	H	H	H	H	H	H	L

ตารางที่ 8.2 แสดงตารางการทำงานของไอซีเบอร์ 74LS174

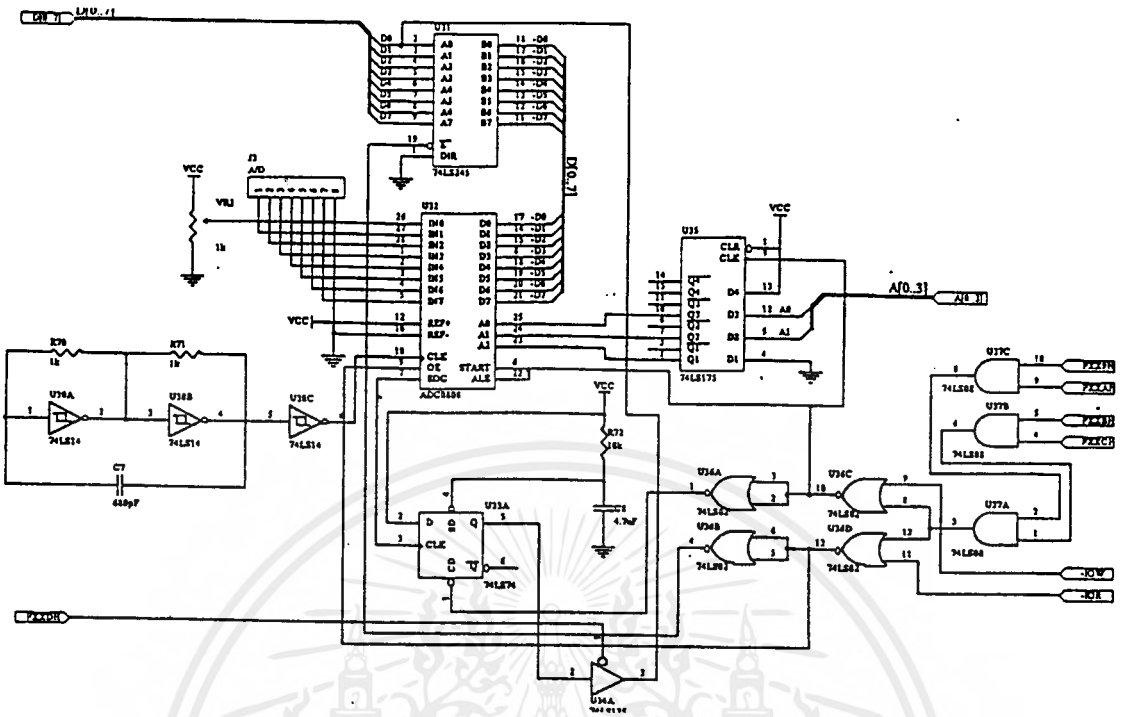
ตามรูปที่ 8.4 ตัวต้านทานปรับค่าได้ทำหน้าที่ปรับแรงดันสูงสุดที่จะจ่ายให้แก่ขั้วไม่กลับสัญญาณ (NON INVERTING) ของ ออปแอมป์ ตัวบนสุด กล่าวง่ายๆก็คือ เป็นตัวกำหนดความไวของสัญญาณอินพุทนั่นเอง การทำงานของวงจรจะเริ่มเมื่อมีสัญญาณอนาล็อกป้อนให้แก่ อินพุทของระดับแรงดันชองสัญญาณอนาล็อกที่มีค่ามากกว่าแรงดันที่ขา 11 ของออปแอมป์ ตัวล่างสุด ก็จะทำให้เอาท์พุทของออปแอมป์ เปลี่ยนสภาวะจาก 1 มาเป็น 0 ทำให้ได้เอาท์พุทจาก ไอซี 74LS147 เป็น HHHL หรือ 1110ไดโอดเปร่งแสง หลัก A จะสว่างดวงเดียว ต่อมาถ้าแรงดันอินพุทมากกว่าแรงดันที่ขา 9 ของออปแอมป์ ตัวถัดไป ก็จะทำให้

เอาท์พุทของ ออปแอมป์ เปลี่ยนสภาวะจาก 1 มาเป็น 0 โดยเอาท์พุทของออปแอมป์ตัล่สุด เป็น 0 ด้วย แต่เนื่องจาก ไอซีเบอร์ 74LS7 มีตารางการทำงานในลักษณะ จัดลำดับความสำคัญ คือ สมมุติวงจร 2 เป็น 0 จะได้เอาท์พุทเป็น HHLH โดยไม่สนใจว่าอินพุท 1 จะเป็นอะไร ดังนั้นในสภาวะนี้ จึงได้เอาท์พุทเป็น HHLH ไดโอดเปร่งแสง หลัก B ก็สว่าง สำหรับการทำงานในระดับแรงดันอื่นๆ ก็มี หลักการทำงานเช่นเดียวกับที่ อธิบายข้างต้น

ในการทดลองนี้ใช้ไอซีเบอร์ ADC0808

<u>คุณสมบัติของ ADC0808</u>

1. รีโซลูชั่น (RESOLUTION) 8 BIT
2. ความคลาดเคลื่อน (TOTAL UNADJUSTED ERROR)± 1/2 LSB
3. ไม่มีรหัสมิซซิ่ง (NO MISSING CODE)
4. ช่วงเวลาการเปลี่ยนแปลง (CONVERTION TIME) 100 uS
5. แรงดัน (SINGLE SUPPLY) 5 Vdc
6. รับสัญญาณอินพุทได้ 8 CHANNEL
7. มีเอาท์พุทแบบ (TRI-STATE WITH LATCHED CONTROL LOGIC)
8. รับแรงดันแอนาลอกอินพุท 0-5 Vdc.
9. กำลังงาน (POWER CONTROL) 15 mW.
10. ใช้วิธีการแปลงสัญญาณแบบ SUCCESSIVE APPROXIMATE
11. อุณหภูมิ (TEMPERATURE RANGE) -40 °C ถึง +85 °C



รูปที่8.5 วงจรแปลงสัญญาณแอนาลอกเป็นดิจิตอล

วงจรที่ใช้ในการทดลอง

วงจรจะทำการต่อ VREF (+) กับ VREF (-) ลงกราวด์ เพื่อเป็นตัวกำหนดแรงดันอ้างอิงให้กับวงจรภายในของไอซี U32 ADC0808 ขา CLK ต่อกับวงจรกำเนิดสัญญาณนาฬิกา ขาของ A/D ต่อกับไอซี U31 IC 74LS245 ขา A0-A2 ต่อกับไอซี U35 IC 74LS175 การทำงานของวงจรเริ่มจากการส่งข้อมูลกำหนดแอดเดรส เพื่อกำหนดช่อง (CHANNEL) ของวงจรแปลงสัญญาณแอนาลอกเป็นดิจิตอล ว่าจะรับข้อมูลอนาลอกจาก ช่องไหน ด้วยการเขียนโปรแกรมดังนี้

```
outportb(0x30c,0x00);
```

	A9	A8	A7	A6	A5	A4	A3	A2	A1	A0
0x30C	1	1	0	0	0	0	1	1	0	0

จากโปรแกรมทำให้สัญญาณ PORT 30CH และ -IOW เป็นลอจิก "0" ทำให้มีสัญญาณ PULSE (LOGIC "0") ที่ขา 10 ของ U 36C ไปกระตุ้นให้ U35 ทำการค้างข้อมูลจาก Address Bus A0-A1 ซึ่งมีค่าเป็น "100" (ดังตาราง) ข้อมูลนี้จะส่งให้กับ A0-A1 ของ ADC0808 เพื่อกำหนดให้ ADC0808 รับสัญญาณแอนาลอกจาก CHANNEL "0" (IN0) ซึ่งในที่นี้ IN0 ต่อกับ VR2 เพื่อเป็นตัวปรับระดับแรงดัน

ตัวอย่างโปรแกรม เพื่อแปลงสัญญาณแอนาลอก เป็นดิจิตอล

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    PortA_D=0x30c,PortCheck=0x30d
char  Data;
        main(){
              outportb(PortA_D,0x00);
              while(!kbhit()) {
                    Data=inport(PortCheck);
                    if (Data!=254){
                          outportb(PortA_D,0x00);
                          Data=inport(PortA_D);
                          gotoxy(10,10);
                          printf("Data = %dD or %xH      ");
                          delay(100);
        }}}
```

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีโปรแกรมภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

1. ป้อนโปรแกรมตามตัวอย่าง EXECUTE โปรแกรม ทดลองปรับ VR2 บันทึกผลการทดลอง

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int   PortA_D=0x30c,PortCheck=0x30d
char  Data;
        main(){
            outportb(PortA_D,0x00);
            while(!kbhit()) {
                Data=inport(PortCheck);
                if (Data!=254){
                    outportb(PortA_D,0x00);
                    Data=inport(PortA_D);
                    gotoxy(10,10);
                    printf("Data = %dD or %xH      ");
                    delay(100);
                }}}
```

ผลการทดลอง.....

.....

.....

สรุปผลการทดลอง.....

.....

.....

คำถาม

1. จงเขียนผังการทำงานของวงจรควบคุมอุณหภูมิโดยใช้ไมโครคอมพิวเตอร์ พร้อมทั้งอธิบายการทำงานของแต่ละจุดการทำงาน

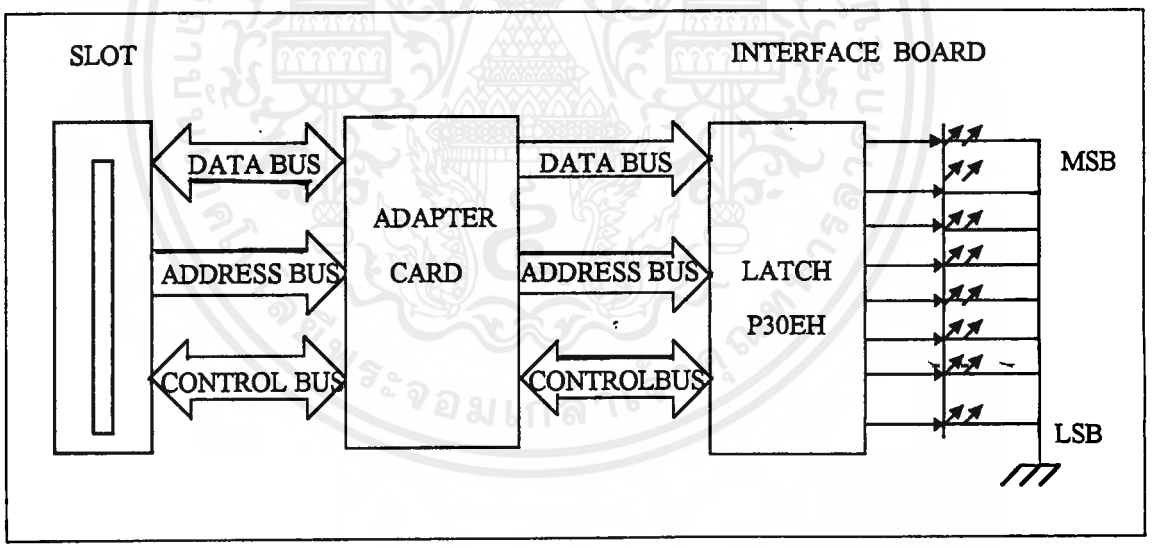
การทดลองที่ 9
วงจรควบคุมไฟจราจร
(TRAFFIC LIGHT)

วัตถุประสงค์

1. เพื่อให้นักศึกษาสามารถเขียนโปรแกรมเพื่อควบคุมการติดดับของไฟจราจรได้
2. เพื่อให้นักศึกษาสามารถประยุกต์โปรแกรมควบคุมอุปกรณ์ภายนอกอย่างอื่นได้

ทฤษฎีเบื้องต้น

การใช้งานไมโครคอมพิวเตอร์ เพื่อควบคุมไฟจราจร ก็คือการส่งข้อมูลไปยังอุปกรณ์เอาท์พุทที่ไดโอดเปล่งแสงโดยมีการควบคุมจังหวะการติดดับ ให้มีระยะเวลาที่แน่นอนและสัมพันธ์กัน

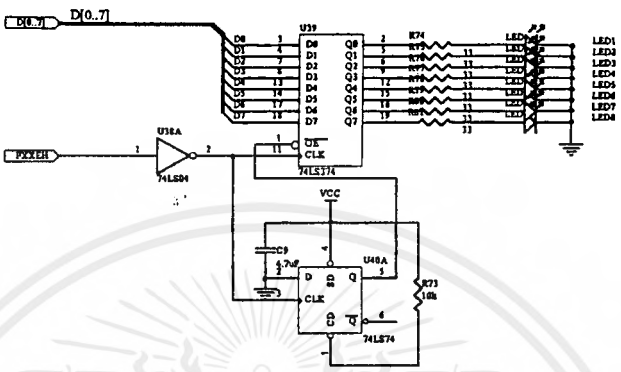


รูปที่ 9.1 ผังการทำงานของวงจรควบคุมไฟจราจร

จากวงจรจะเห็นว่าเป็นวงจรลักษณะเดียวกันกับวงจรควบคุมอุปกรณ์ภายนอกเบื้องต้น เพียงแต่อุปกรณ์ภายนอกเป็นไดโอดเปล่งแสงทำการจัดเรียงตำแหน่งใหม่ให้เป็นลักษณะของ

สัญญาณไฟจราจร แล้วควบคุมจังหวะการส่งข้อมูลไปยังไดโอดเปล่งแสง ให้สัมพันธ์กับลักษณะการติดดับของไฟจราจร

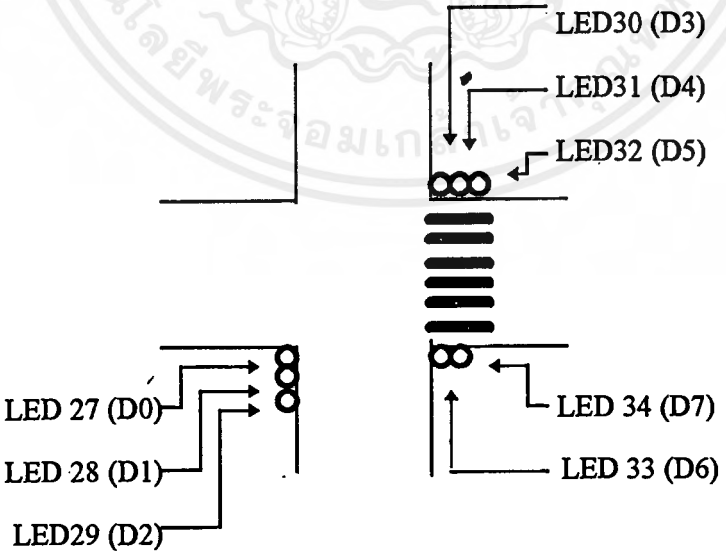
การทำงานของวงจร



รูปที่ 9.2 วงจรควบคุมไฟจราจร

ประกอบด้วย IC U39 74LS374 ทำหน้าที่เป็นตัวรับข้อมูลจาก DATA BUS ที่ D0-D7 แล้วส่งข้อมูลไปยัง LED ที่ Q0-Q7 IC 74LS374 ถูก DECODE ไว้ที่ PORT 30EH

IC U40 เบอร์ 74LS74 ทำหน้าที่เป็นตัวทำให้ OUTPUT Q0-Q7 ของ U39 อยู่ในสภาวะ HIGH IMPEDANCE ในจังหวะเริ่มเปิดไฟเข้าเครื่องครั้งแรก



รูปที่ 9.2 แผนผังตำแหน่งของไดโอดเปล่งแสงเมื่อเทียบกับบัสข้อมูล

อุปกรณ์การทดลอง

1. เครื่องคอมพิวเตอร์ที่มีโปรแกมภาษาซี
2. อะแดปเตอร์การ์ด
3. ชุดบอร์ดอินเตอร์เฟส
4. สายแพร์

ลำดับขั้นการทดลอง

1. ทดลองป้อนโปรแกรมต่อไปนี้แล้วสังเกตตำแหน่งของไดโอดเปล่งแสงที่สว่าง

```
#include <stdio.h>
#include <conio.h>
#include <dos.h>
int    Port=0x30e,Delay=100;
char  a,Data[]={0x01,0x02,0x40,0x80,0x10,0x20,0x40,0x80};
        main(){
                while(!kbhit()) {
outportb(Port,Data[a++]);
if(a>=8)  a=0;
delay(Delay); }
        }
```

สรุปและวิจารณ์ผลการทดลอง.....

.....

.....

คำถาม

1. เขียนโปรแกรมเพื่อควบคุมการติดดับของไฟให้มีการติดดับเหมือนกับไฟจราจรจริง

บรรณานุกรม

บริษัทถาวรคอมพิวเตอร์ จำกัด,"คู่มือการทดลองชุดฝึกไมโครคอมพิวเตอร์ในงานอุตสาหกรรม ",กรุงเทพมหานคร,2534

ยืน ภู่วรวรรณ, " เทคโนโลยีฮาร์ดแวร์ IBM PC ",กรุงเทพมหานคร, หจก. เอช-เอน การพิมพ์,2533

ศูนย์ภาษาคอมพิวเตอร์, "การใช้งาน Z-80 ",กรุงเทพมหานคร,หจก. สำนักพิมพ์ฟิสิกส์เซ็นเตอร์, 2534

ธานินทร์ ถาวรศาสนวงศ์ และ ทินกร ดุ๊ก, " การอินเตอร์เฟส IBM PC " ,กรุงเทพมหานคร,หจก. สำนักพิมพ์ฟิสิกส์เซ็นเตอร์, 2534

ธันวา ศรีประมง, " การโปรแกรมภาษาซีสำหรับวิศวกรรม " , กรุงเทพมหานคร,มหาวิทยาลัยมหานคร,2539

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นางสาวกนกพร กสิกพันธุ์
วันเดือนปีเกิด	วันที่ 15 พฤศจิกายน พ.ศ. 2519
สถานที่เกิด	จังหวัดปราจีนบุรี
ภูมิลำเนาเดิม	56/4 ตำบลบ้านสร้าง อำเภอบ้านสร้าง จังหวัดปราจีนบุรี 25150
ที่อยู่ปัจจุบัน	56/4 ตำบลบ้านสร้าง อำเภอบ้านสร้าง จังหวัดปราจีนบุรี 25150
โทรศัพท์	(037)271412

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนเมืองปราจีนบุรี
มัธยมศึกษาตอนต้น	โรงเรียนปราจีนกัลยาณี
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคปราจีนบุรี
ประกาศนียบัตรวิชาชีพชั้นสูง(ปวส.)	วิทยาลัยเทคนิคปราจีนบุรี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	-
คติพจน์	ทำวันนี้ให้ดีที่สุด

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายประวิทย์ ลาวัณย์วิสุทธิ์
วันเดือนปีเกิด	วันที่ 1 กรกฎาคม พ.ศ. 2518
สถานที่เกิด	จังหวัดเพชรบุรี
ภูมิลำเนาเดิม	48 หมู่ 7 ตำบลท่าไม้รวก อำเภอท่ายาง จังหวัดเพชรบุรี 76130
ที่อยู่ปัจจุบัน	600 หมู่ 1 บ้านริมสวน ถนนอ่อนนุช-ลาดกระบัง กรุงเทพมหานคร
โทรศัพท์	(032)458250

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนบ้านท่าลาว
มัธยมศึกษาตอนต้น	โรงเรียนหนองชุมแพรวิทยา
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคราชบุรี
ประกาศนียบัตรวิชาชีพชั้นสูง(ปวส.)	วิทยาลัยเทคนิคราชบุรี
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	ทุนเรียนดี
คติพจน์	ทำวันนี้ให้ดีที่สุด เพื่อวันพรุ่งนี้

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายประสิทธิ์ ภูมิภาค
วันเดือนปีเกิด	วันที่ 1 พฤษภาคม พ.ศ. 2518
สถานที่เกิด	จังหวัดร้อยเอ็ด
ภูมิลำเนาเดิม	20 หมู่ 11 บ้านโนนคำ ตำบลสระนกแก้ว อำเภอโพนทอง จังหวัดร้อยเอ็ด 45110
ที่อยู่ปัจจุบัน	600 หมู่ 1 บ้านริมสวน ถนนอ่อนนุช-ลาดกระบัง กรุงเทพมหานคร
โทรศัพท์	-

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนบ้านโนนแก้ว
มัธยมศึกษาตอนต้น	โรงเรียนโพนทองวิทยายน
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคร้อยเอ็ด
ประกาศนียบัตรวิชาชีพชั้นสูง(ปวส.)	วิทยาลัยเทคนิคร้อยเอ็ด
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	
คติพจน์	อย่าเห็นแก่ตัว อย่ากลัวเสียเปรียบ

ประวัติผู้แต่ง

ชื่อผู้ทำปริญญานิพนธ์	นายวิชาญ เพชรมณี
วันเดือนปีเกิด	วันที่ 9 ตุลาคม พ.ศ. 2518
สถานที่เกิด	จังหวัดสุราษฎร์ธานี
ภูมิลำเนาเดิม	138/ 8 หมู่ 4 ตำบลมะขามเตี้ย อำเภอเมือง จังหวัดสุราษฎร์ธานี
ที่อยู่ปัจจุบัน	600 หมู่ 1 บ้านริมสวน ถนนอ่อนนุช-ลาดกระบัง กรุงเทพมหานคร
โทรศัพท์	(077)285226

ประวัติการศึกษา

ประถมศึกษา	โรงเรียนมานิตานุเคราะห์
มัธยมศึกษาตอนต้น	โรงเรียนสุราษฎร์ธานี
ประกาศนียบัตรวิชาชีพ (ปวช.)	วิทยาลัยเทคนิคสุราษฎร์ธานี
ประกาศนียบัตรวิชาชีพชั้นสูง(ปวส.)	สถาบันเทคโนโลยีราชมงคลวิทยาเขตภาคใต้ (สงขลา)
ปริญญาตรี	สาขาวิชาอิเล็กทรอนิกส์และคอมพิวเตอร์ ภาควิชาครุศาสตร์วิศวกรรม คณะครุศาสตร์อุตสาหกรรม
ผลงานที่ได้รับรางวัล	-
ทุนการศึกษา	ทุนผูกพันธ์สถาบันเทคโนโลยีราชมงคล
คติพจน์	...