

บอร์ดแสดงผลแบบดอทเมตริกซ์สามสี

Three Colors Dot Matrix Display Board



โดย

นายพิเชฐ เหล่าพินนา

นายไพฑูรย์ ทองจันทร์

เลขที่.....

เลขทะเบียน.....42162

วัน, เดือน, ปี.....14 พ.ค. 2545

.b.....

.i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

บอร์ดแสดงผลแบบด็อตเมทริกซ์สามสี
Three Colors Dot Matrix Display Board



โดย

นายพิเชฐ เหล่าพินนา 41013058

นายไพฑูรย์ ทองจันทร์ 41013062

อาจารย์ที่ปรึกษา

รศ.ดร.ถวิด พิงมา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง บอร์ดแสดงผลแบบดอตเมทริกซ์สามสี

Three Colors Dot Matrix Display Board

ผู้จัดทำ

1. นายพิเชฐ เหล่าพินนา 41013058
2. นายไพฑูรย์ ทองจันทร์ 41013062

.....
(รศ.ดร. ถวิล พึ่งมา)

อาจารย์ที่ปรึกษา



บอร์ดแสดงผลแบบดอทเมตริกซ์สามสี

Three Colors Dot Matrix Display Board

โดย นายพิเชฐ เหล่าพินนา 41013058

นายไพฑูรย์ ทองจันทร์ 41013062

อาจารย์ที่ปรึกษา รศ.ดร.ถวิล พึ่งมา

บทคัดย่อ

ปริญญานิพนธ์ฉบับนี้เป็นการนำเสนอ การออกแบบและการสร้างบอร์ดแสดงผลแบบ ดอทเมตริกซ์ ซึ่งสามารถแสดงข้อมูลได้ทั้งตัวอักษรภาษาไทยและภาษาอังกฤษ ตลอดจนการแสดงรูปภาพเคลื่อนไหวแบบต่างๆ ได้สามสี คือ สีแดง สีเขียวและสีส้ม โดยมีขนาด 64 X 128 จุด (64 แถว X 128 หลัก) สามารถเลือกรูปแบบการแสดงผลได้สองแบบ คือ แบบบรรทัดเดียวและแบบ 2 บรรทัด โดยใช้ไมโครคอนโทรลเลอร์ ตระกูล MCS-51 เป็นตัวควบคุมการทำงานและสามารถควบคุมการแสดงผลเปลี่ยนแปลงข้อความผ่านทางคอมพิวเตอร์ส่วนบุคคล ด้วยโปรแกรมที่จัดทำขึ้นเพื่อให้ง่ายต่อการใช้งาน

ABSTRACT

This project presented about design and make prototype of Three colors dot matrix display board. The display can show Thai and English characters, Graphic data in three colors (Red, Green and Orange) by using LED display capacity by 64 X 128 (64 rows X 128 columns). The format of display can set by single row and double row. The system is controlled by micro-controller MCS-51 family. The user can control and change data via serial port of personal computer linked to the main system with the simple software.

สารบัญรูปภาพ

สารบัญตาราง

บทที่ 1 บทนำ

i

บทที่ 2 ทฤษฎีและหลักการ

2

2.1 ไดโอดเปล่งแสง (Light Emitting Diode)

2

2.2 การสแกน

7

2.3 การสื่อสารแบบอนุกรม

7

2.4 หลักการพื้นฐานของ Delphi

12

2.5 การอินเตอร์รัปต์ของ MCS-51

21

บทที่ 3 การคำนวณและการสร้าง

25

3.1 แนวคิดและการทำงาน

25

3.2 แนวคิดในการออกแบบ

25

3.3 โครงสร้างและการทำงานของชุดควบคุมตัวสเลฟ

25

3.4 โครงสร้างและการทำงานของชุดควบคุมตัวมาสเตอร์

34

3.5 การแมปข้อมูลตัวอักษร ตัวเลข และรูปภาพต่างๆ

35

3.6 โปรแกรมสำเร็จรูปที่ใช้ในการควบคุมการแสดงผล

36

3.7 โฟลว์ชาร์ตของโปรแกรมต่างๆ ที่ใช้ควบคุมการแสดงผลของบอร์ดแสดงผล

37

บทที่ 4 การทดลองและผลการทดลอง

46

4.1 การทดลองวงจรเลดซ์

46

4.2 การทดลองวงจรควบคุมการสแกนด้านแถว

47

4.3 การทดลองวัดกระแสของแหล่งจ่าย 5 โวลต์

47

4.4 การทดลองวัดกระแสของแหล่งจ่าย 12 โวลต์

48

4.5 การทดลองการแสดงผลในรูปแบบต่างๆ

48

บทที่ 5 บทวิจารณ์และบทสรุป

52

ภาคผนวก

กิตติกรรมประกาศ

หนังสืออ้างอิง

สารบัญรูปภาพ

	หน้าที่
รูปที่ 2.1 แสดงการเกิด Electroluminescence ของหลอดแอลอีดี	2
รูปที่ 2.2 แสดงสัญลักษณ์ทางไฟฟ้าและโครงสร้าง	3
รูปที่ 2.3 แสดงความสัมพันธ์ของความเข้มแสงกับความยาวคลื่น	3
รูปที่ 2.4 แสดงความสัมพันธ์ของกระแสฟอว์เวิร์ดกับแรงดันฟอว์เวิร์ด	4
รูปที่ 2.5 แสดงความสัมพันธ์ของความเข้มแสงกับกระแสฟอว์เวิร์ด	5
รูปที่ 2.6 แสดงความสัมพันธ์ของประสิทธิภาพกับกระแสฟีด (Peak)	5
รูปที่ 2.7 แสดงกระแสฟีดสูงสุดกับการเปลี่ยนแปลงความกว้างพัลส์	5
รูปที่ 2.8 แสดงความสัมพันธ์ของการส่องแสงกับมุมมองของการมองเห็น	6
รูปที่ 2.9 แสดงรูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรม	8
รูปที่ 2.10 แสดงรูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรมแบบอะซิงโครนัส	8
รูปที่ 2.11 แสดงคอนเน็คเตอร์แบบ DB-25 และ DB-9	10
รูปที่ 2.12 แสดงการต่ออุปกรณ์ภายนอกกับพอร์ตอนุกรมของคอมพิวเตอร์ในลักษณะต่างๆ	12
รูปที่ 2.13 แสดงวินโดว์ของDelphi	16
รูปที่ 2.14 แสดงวินโดว์หลัก	16
รูปที่ 2.15 แสดงวินโดว์ออกแบบเจ็ทอินสเป็คเตอร์	18
รูปที่ 2.16 แสดงวินโดว์ฟอร์ม	18
รูปที่ 2.17 แสดงวินโดว์โค้ดเอ็ดิเตอร์	19
รูปที่ 2.18 แสดงแหล่งที่มาสัญญาณอินเตอร์รัปต์	21
รูปที่ 3.1 แสดงบล็อกไดอะแกรมของบอร์ดแสดงผลแบบดิจิตอลเมตริกซ์ 3 สี	26
รูปที่ 3.2 แสดงบล็อกไดอะแกรมของชุดควบคุมตัวสลับ	27
รูปที่ 3.3 แสดงวงจรเลตซ์	28
รูปที่ 3.4 แสดงลักษณะการเลตซ์ข้อมูล	28
รูปที่ 3.5 แสดง Timing Diagramme ของวงจรเลตซ์	29
รูปที่ 3.6 แสดงการต่อวงจรที่ใช้ควบคุมการสแกนทางด้านแถว	29
รูปที่ 3.7 แสดงการต่อวงจรควบคุมการเลตซ์ข้อมูล	30
รูปที่ 3.8 แสดงวงจรป้องกันแอลอีดีเสียหาย	31
รูปที่ 3.9 แสดง Timing Diagramme ของวงจรป้องกันแอลอีดีเสียหาย	31
รูปที่ 3.10 แสดงวงจรขับแอลอีดี	32
รูปที่ 3.11 แสดงบล็อกไดอะแกรมของชุดควบคุมตัวมาสเตอร์	34
รูปที่ 3.12 แสดงการแม็ปเลข 1	35
รูปที่ 3.13 แสดงหน้าต่างของโปรแกรมสำเร็จรูป	36
รูปที่ 3.14 แสดงขั้นตอนการรับข้อมูลของตัวอักษรและรูปภาพ	37

สารบัญรูปภาพ (ต่อ)

	หน้า
รูปที่ 3.15 แสดงการอินเตอร์รัปต์ไทม์เมอร์พร้อมทั้งสแกนข้อมูล	38
รูปที่ 3.16 แสดงการทำงานของโปรแกรมหลัก	39
รูปที่ 3.17 แสดงการทำงานของโปรแกรมรูปภาพเคลื่อนไหว	40
รูปที่ 3.18 แสดงการทำงานของโปรแกรมแสดงตัวอักษรตัวใหญ่	41
รูปที่ 3.19 แสดงการทำงานของโปรแกรมแสดงตัวอักษรตัวเล็ก	42
รูปที่ 3.20 แสดงการทำงานของโปรแกรมแสดงตัวอักษรตัวเล็ก (ต่อ)	43
รูปที่ 3.21 แสดงการทำงานของโปรแกรมเลื่อนข้อมูล	43
รูปที่ 3.22 แสดงการการทำงานของโปรแกรมส่งข้อมูล	44
รูปที่ 3.23 แสดงบอร์ดแสดงผลแบบค็อตเมตริกซ์ 3 สี	45
รูปที่ 4.1 แสดงรูปสัญญาณที่ได้จากการแลตซ์ข้อมูล	46
รูปที่ 4.2 แสดงรูปสัญญาณที่ใช้ในการควบคุมการสแกนด้านแถว	47
รูปที่ 4.3 แสดงผลการทดลองการวัดกระแสของแหล่งจ่าย 5 โวลต์	48
รูปที่ 4.4 แสดงผลการทดลองการวัดกระแสของแหล่งจ่าย 12 โวลต์	48
รูปที่ 4.5 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสีแดงบรรทัดเดียว	49
รูปที่ 4.6 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสีเขียวบรรทัดเดียว	49
รูปที่ 4.7 แสดงผลการทดลองการแสดงผลแบบรูปภาพ (รูปมิกกี้เมาส์)	50
รูปที่ 4.8 แสดงผลการทดลองการแสดงผลแบบรูปภาพ (รูปกระต่าย)	50
รูปที่ 4.9 แสดงผลการทดลองการแสดงผลแบบรูปภาพ (รูปสุนัข)	51
รูปที่ 4.10 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสองบรรทัด	51

สารบัญตาราง

	หน้าที่
ตารางที่ 2.1 แสดงค่า Absolute Maximum Ratings at $T_A = 25^\circ\text{C}$	3
ตารางที่ 2.2 แสดงคุณสมบัติทางไฟฟ้าและทางแสง ($T_A = 25^\circ\text{C}$)	4
ตารางที่ 2.3 แสดงการเปรียบเทียบข้อดีและข้อเสียของแอลอีดี	6
ตารางที่ 2.4 แสดงการจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232	11
ตารางที่ 2.5 แสดงข้อสัญญาณอินเทอร์รัปต์	22
ตารางที่ 2.6 แสดงความหมายบิตต่างๆใน (Interrupt Enable Register)	22



บทที่ 1

บทนำ

เนื่องจากโลกปัจจุบันนี้เป็นโลกของข่าวสารข้อมูล ผู้ที่มีข้อมูลมากกว่าย่อมได้เปรียบกว่าในทำนองเดียวกัน ผู้ที่สามารถเสนอข่าวสารข้อมูลแก่กลุ่มเป้าหมายได้มากกว่าก็ย่อมได้เปรียบกว่าเช่นกัน โดยเฉพาะอย่างยิ่งด้านธุรกิจและการโฆษณาซึ่งมีการแข่งขันกันมาก ดังนั้นจึงต้องอาศัยสื่อที่สามารถแสดงข้อมูลข่าวสาร และภาพเคลื่อนไหวต่างๆ ที่สามารถดึงดูดความสนใจจากผู้พบเห็น และสามารถสื่อสารให้ทุกคนเข้าใจในข้อมูลข่าวสารนั้นๆ ได้

ป้ายแสดงข่าวสาร(Display Board) เป็นสื่อชนิดหนึ่งที่นิยมใช้กันมากในปัจจุบัน ซึ่งสามารถพบเห็นได้ตามแหล่งชุมชน ย่านธุรกิจ ศูนย์การค้า โรงพยาบาล บริษัทห้างร้านต่างๆ และหน่วยงานราชการบางแห่ง ซึ่งการแสดงผลมีหลายรูปแบบ มีแบบข้อความนิ่งและข้อความที่เลื่อนได้ เช่น จากซ้ายไปขวา จากขวาไปซ้าย บนลงล่างฯ

ในอดีตการแสดงผลจะใช้เซเวนเซกเมนต์(7-Segment) มีความละเอียดน้อย ต่อมาได้มีการพัฒนาเป็นแบบจุด(Dot matrix display) ซึ่งสามารถแสดงผลได้ละเอียดและสวยงามมากขึ้น แต่ในคอนนั้น การแสดงผลแบบจุดจะใช้หลอดแอลอีดี (LED) ที่เปล่งแสงได้สีเดียว จึงแสดงผลได้แค่เพียงสีเดียว ต่อมาเทคโนโลยีทางด้านอิเล็กทรอนิกส์มีความก้าวหน้ามากขึ้นจนสามารถผลิตหลอดแอลอีดี ที่เปล่งแสงได้หลายสีในหลอดเดียวกัน ทำให้สามารถแสดงผลได้หลายสี มีความสวยงามมากขึ้น

ดังนั้นในโครงการนี้จึงนำหลอด แอลอีดี ซึ่งสามารถเปล่งแสงได้สามสี คือ สีแดง สีเขียว และสีส้ม มาประยุกต์สร้างเป็นบอร์ดแสดงผลแบบจุด(Dot matrix display board) ขนาด 64 X 128 จุด (64 แถว X 128 คอลัมน์) สามารถแสดงผลได้ทั้งภาษาไทย ภาษาอังกฤษและรูปภาพ การแสดงผลมีหลายรูปแบบ เช่น แบบตัวใหญ่บรรทัดเดียว แบบตัวเล็กสองบรรทัด เลื่อนจากขวาไปซ้าย ภาพเคลื่อนไหวและภาพนิ่ง เนื่องจากเป็นบอร์ดแสดงผลขนาดใหญ่ เพื่อความสะดวกในการออกแบบและควบคุมการทำงานจึงแบ่งออกเป็นบอร์ดย่อยๆ จำนวนสี่บอร์ด โดยใช้ไมโครคอนโทรลเลอร์ ตระกูล MCS-51 จำนวนห้าตัวในการควบคุมการทำงานเป็นมาสเตอร์ (Master) หนึ่งตัวและเป็นสเลฟ(Slave) สี่ตัว สามารถควบคุมการแสดงผลผ่านทางคอมพิวเตอร์ ด้วยซอฟต์แวร์ที่จัดทำขึ้น เพื่อให้ง่ายต่อการใช้งาน

บทที่ 2

ทฤษฎีและหลักการ

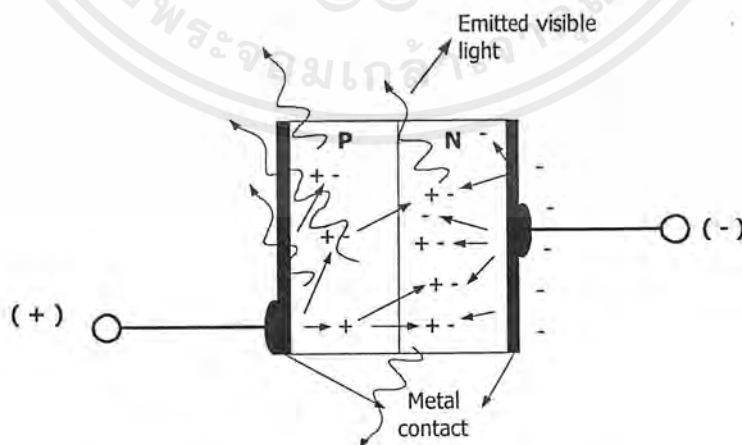
2.1 ไดโอดเปล่งแสง (Light Emitting Diode)

บอร์ดแสดงผลอิเล็กทรอนิกส์ในโครงงานนี้ ส่วนแสดงผลที่สำคัญคือ แอลอีดี ฉะนั้นจึงจะต้องมาศึกษาและทำความเข้าใจเกี่ยวกับคุณสมบัติต่างๆ และการทำงานของแอลอีดีเพื่อจะได้นำไปใช้งานได้อย่างถูกต้องตามความต้องการ

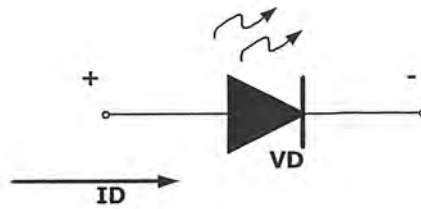
2.1.1 คุณสมบัติของแอลอีดี

แอลอีดีเป็นไดโอดชนิดหนึ่งซึ่งสามารถเปล่งแสงได้ ซึ่งโครงสร้างของแอลอีดี จะมีลักษณะเป็นสารกึ่งตัวนำชนิดพี (P-Type) และชนิดเอ็น (N-Type) ต่อกันเป็น พี-เอ็น จังก์ชัน (P-N Junction) เมื่อมีการจ่ายกระแสฟอร์เวิร์ดไบอัส (Forward Bias) ให้แอลอีดี แอลอีดีจะเปล่งแสงออกมา ซึ่งกระแสฟอร์เวิร์ดไบอัสนี้จะไปกระตุ้นอิเล็กตรอนและโฮลข้าม พี-เอ็น จังก์ชัน เพื่อมารวมตัวกันซึ่งในการรวมตัวกันนี้ จะมีการปลดปล่อย พลังงานออกมาในรูปของโฟตอนซึ่งเป็นอนุภาคของแสง

สารกึ่งตัวนำที่นำมาใช้ในการสร้างแอลอีดีได้มีการเลือกชนิดที่มีคุณสมบัติง่ายต่อการสร้างและมีเสถียรภาพในสภาวะแวดล้อมต่างๆ สำหรับสารกึ่งตัวนำที่นิยมนำมาสร้างแอลอีดีจะมีอยู่หลายชนิด เช่น แกลเลียมอาร์เซไนด์ฟอสไฟด์ (Gallium Arsenidephosphide GaSaP) ซึ่งมีความยาวคลื่น 6600 อังสตรอม (Angstrom) เป็นช่วงความถี่สีแดงซึ่งอยู่ในช่วงความถี่ที่ตาคนเรามองเห็น หรือแกลเลียมฟอสไฟด์ (Gallium Phosphide GaP) จะมีความยาวคลื่น 5700 อังสตรอม เป็นช่วงความถี่เขียว ความถี่ที่ได้จากสารทั้งสองชนิดค่อนข้างจะคงที่แต่อาจจะมีการเปลี่ยนแปลงไปบ้าง สำหรับสารกึ่งตัวนำทั้งสองชนิดนี้จะใช้กระแสไฟฟ้าไม่มากในการไบอัสเพื่อให้เกิดการปลดปล่อยโฟตอนออกมา การให้แสงของแอลอีดีโดยการจ่ายกระแสไฟฟ้าเรียกว่า อิเล็กโตรลูมินิเซนส์ (Electroluminescence) ดังรูปที่ 2.1



รูปที่ 2.1 แสดงการเกิด Electroluminescence ของหลอดแอลอีดี

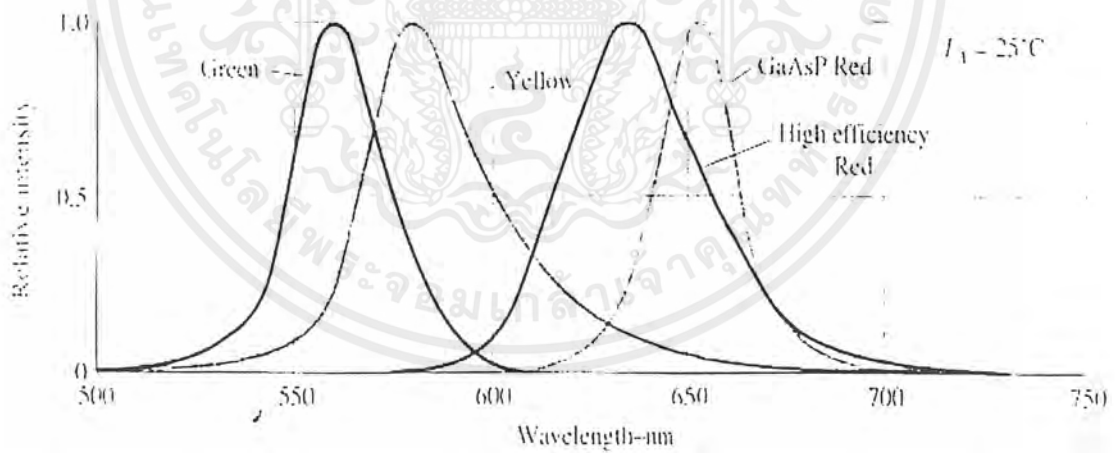


รูปที่ 2.2 แสดงสัญลักษณ์ทางไฟฟ้าและโครงสร้าง

ตารางที่ 2.1 แสดงค่า Absolute Maximum Ratings at $T_A = 25^\circ\text{C}$

Parameter	High Eff. Red (4160)	Units
Power dissipation	120	mW
Average forward current	2 [1]	mA
Peak forward current	60	mA
Operating and storage temperature range	-55°C to 100°C	
Lead soldering temperature [1.6 mm. (0.063 in.) from body]	230°C for 3 sec.	

[1] Derate from 50°C at $0.2\text{ mA}/^\circ\text{C}$



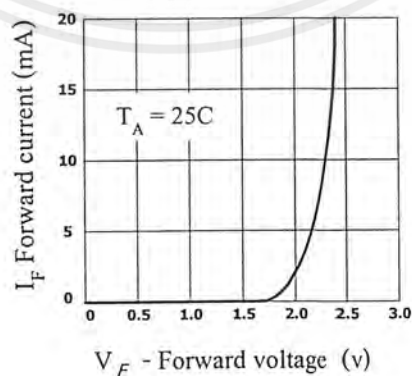
รูปที่ 2.3 แสดงความสัมพันธ์ของความเข้มแสงกับความยาวคลื่น

ตารางที่ 2.2 แสดงคุณสมบัติทางไฟฟ้าและทางแสง ($T_A = 25^\circ\text{C}$)

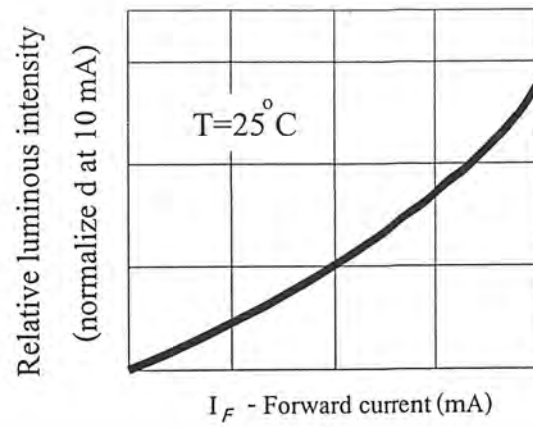
Symbol	Description	High Eff. Red (4160)			Units	Test Conditions
		Min.	Typ.	Max.		
I_v	Axial luminous intensity	1.0	3.0		Mcd	$I_F = 10\text{ mA}$
$2\theta_{1/2}$	Included angle between half luminous intensity point		80		Deg.	Note 1
λ_{peak}	Peak wavelength		635		Nm.	Measurement at peak
λ_d	Dominant wavelength		628		Nm.	Note 2
τ_s	Speed of response		90		Ns.	
C	Capacitance		11		PF.	$V_F = 0$; $f = 1\text{ MHz}$
θ_{JC}	Thermal resistance		120		$^\circ\text{C}/\text{W}$	Junction to cathode lead at 0.79 mm.(.031 in.) from body
V_F	Forward voltage		2.2	3.0	V	$I_F = 10\text{ mA}$.
BV_R	Reverse breakdown voltage	5.0			V	$I_R = 100\text{ }\mu\text{A}$
η_v	Luminous efficacy		147		lm/W	Note 3

NOTES:

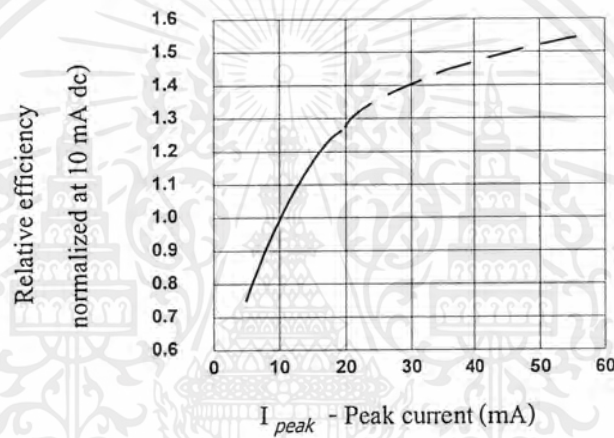
- $\theta_{1/2}$ is the off-axis angle at which the luminous intensity is half the axial luminous intensity.
- The dominant wavelength, λ_d , is derived from the CIE chromaticity diagram and represents the single wavelength that defines the color of the device.
- Radiant intensity, I_e , in watts/steradian, may be found from the equation $I_e = I_v / \eta_v$, where I_v is the luminous intensity in candelas and η_v is the luminous efficacy in lumens/watt.



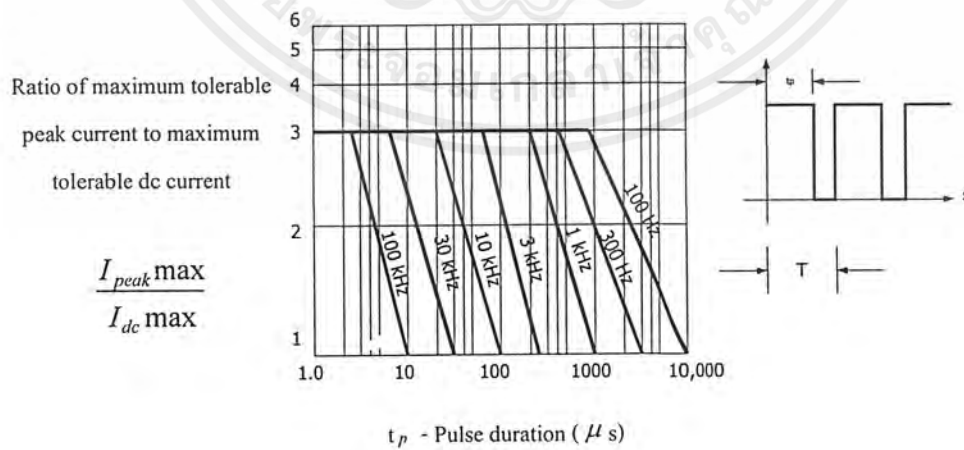
รูปที่ 2.4 แสดงความสัมพันธ์ของกระแสฟอว์เวิร์ดกับแรงดันฟอว์เวิร์ด



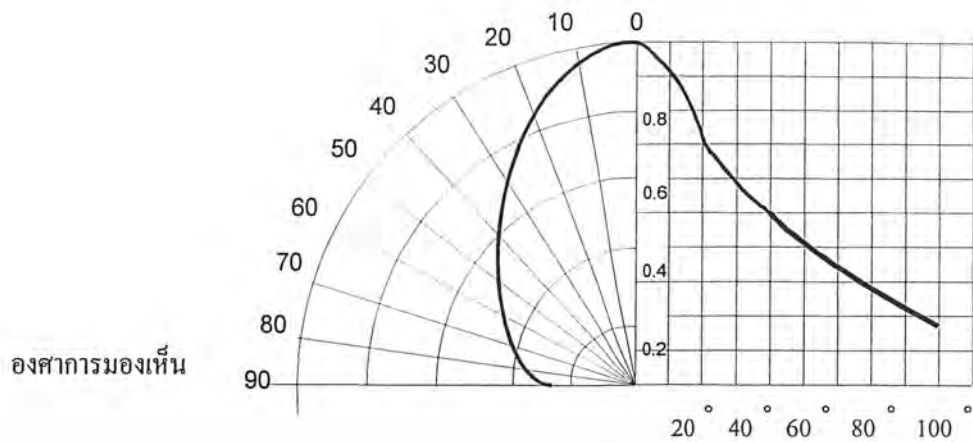
รูปที่ 2.5 แสดงความสัมพันธ์ของความเข้มแสงกับกระแสฟอร์เวิร์ด



รูปที่ 2.6 แสดงความสัมพันธ์ของประสิทธิภาพกับกระแสพีค (Peak)



รูปที่ 2.7 แสดงความสัมพันธ์ของกระแสพีคสูงสุดกับการเปลี่ยนแปลงความกว้างพัลส์



รูปที่ 2.8 แสดงความสัมพันธ์ของการส่องแสงกับมุมมองของการมองเห็น

รูปที่ 2.3 ถึง 2.8 แสดงคุณสมบัติและกราฟคุณลักษณะของแอลอีดี ของบริษัท ฮิวเลทแพคการ์ด แบบ ไฮเอฟิซิเอนซี (High-efficiency)

แอลอีดีโดยทั่วไปจะไม่สามารถมองเห็นได้ทุกแง่มุม กล่าวคือเราจะเห็นแสงที่ปล่อยออกมาจากแอลอีดีในมุมแคบๆที่ทำกับแกนของหลอดเท่านั้น

2.1.2 ข้อดีและข้อเสียของแอลอีดี

ตารางที่ 2.3 แสดงการเปรียบเทียบข้อดีและข้อเสียของแอลอีดี

ข้อดีของแอลอีดี	ข้อเสียของแอลอีดี
1. สามารถสว่างและดับได้ในเวลาที่สั้นมากเพียงไม่กี่นาโนวินาที ดังนั้นจึงสามารถให้แสงจากแอลอีดี เป็นตัวนำข่าวสารหรือข้อมูลไปใช้ในการสื่อสารที่มีความถี่เป็น MHz 2. ใช้แรงดันต่ำและกินกระแสเพียงเล็กน้อยจึงทำให้เหมาะที่จะใช้ควบคู่กับ IC 3. มีอายุการใช้งานมากกว่าหลอดไฟ 4. มีกำลังส่องสว่างแปรผันตามกระแสที่ไหลผ่าน ดังนั้นจึงสามารถส่งข้อมูลในรูปของกระแสจากจุดหนึ่งไปยังอีกจุดหนึ่งที่ห่างไกลออกไปโดยใช้แสง 5. ทนทานต่อการกระแทกกระเทือน	1. สำหรับการใช้งานที่ต้องการกำลังส่องสว่างเต็มที่ ต้องมีวงจรควบคุมกำลังส่องสว่างของแอลอีดี 2. เสียหายได้ง่ายถ้าแรงดันและกระแสสูงกว่าที่ตัวมันจะรับได้

2.2 การสแกน

2.2.1 หลักในการทำให้เกิดภาพบนจอแสดงผล

เราสามารถทำให้มีภาพบนจอแสดงผลแบบจุด (Dot Matrix) ได้จากการอาศัยหลักของการสแกน คือภาพที่ปรากฏนั้นจะประกอบด้วยการรวมเส้นที่เกิดจากการสแกนครั้งละหนึ่งเส้นเป็นจำนวนเท่ากับ จำนวนหลัก (Column) หรือจำนวนแถว (Row) ของบอร์ดแสดงผลแล้วแต่รูปแบบของการสแกน ซึ่ง การมองเห็นของมนุษย์ที่จะเห็นเป็นภาพหนึ่งได้นั้น ภาพที่เกิดจะต้องเกิดซ้ำกันหลายๆรอบ ความเร็วที่ใช้ นั้นจะต้องมากกว่า 25 ภาพต่อวินาที โดยการสแกนจะถูกควบคุมด้วยซอฟต์แวร์ซึ่งสามารถกำหนดความเร็ว ได้ เทคนิคในการสแกนภาพจอแสดงผลแบบจุด สามารถทำได้ 2 วิธีด้วยกัน คือ

2.2.2 การสแกนทางหลัก (Column)

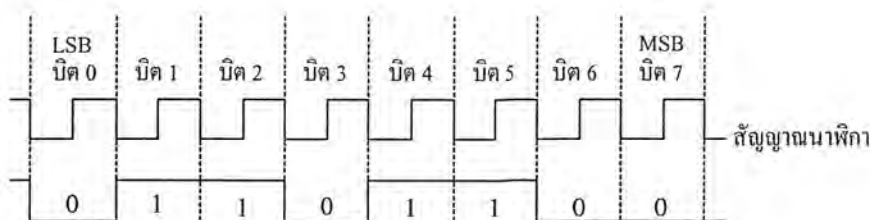
ในการสแกนทางหลัก จะทำการส่งข้อมูลออกไปทางแถว โดยจะส่งข้อมูลตัวที่ 1 ออกไปแล้วให้ หลักที่ 1 แอคทีฟ จากนั้นก็ทำการส่งข้อมูลตัวที่ 2 ออกไปแล้วให้หลักที่ 2 แอคทีฟ ทำเช่นนี้ไปจนกระทั่ง ข้อมูลส่งออกครบทุกหลักก็จะเป็นการสแกนครบ 1 รอบ จะสังเกตว่าถ้าจำนวนหลักที่จะแสดงผลออกมา เป็นตัวอักษรที่มีจำนวนหลายหลักวิธีการสแกนแบบนี้จะไม่เหมาะสมนักที่จะนำมาใช้งานเพราะเมื่อให้แอล อีดีหลักที่หนึ่งติด กว่าที่แอลอีดีหลักสุดท้ายจะติดต้องใช้เวลาานาน

2.2.3 การสแกนทางแถว (Row)

ในการสแกนทางแถว จะทำการส่งข้อมูลของแถวที่ 1 ออกไปจนครบทุกหลักก่อนแล้วให้แถวที่ 1 แอคทีฟพร้อมกัน จากนั้นก็ทำการส่งข้อมูลของแถวที่ 2 ออกไปจนครบทุกหลักแล้วให้แถวที่ 2 แอคทีฟ พร้อมกัน ทำเช่นนี้จนกระทั่งข้อมูลถูกส่งออกไปครบทุกแถวก็จะเป็นการสแกนครบ 1 รอบ วิธีนี้มีข้อดีคือ สามารถแสดงผลเป็นตัวอักษรพร้อมกันได้หลายหลักและถ้าจัดเวลาให้เหมาะสมแล้วเวลาทำการสแกนจะ ไม่เกิดอาการพริ้ว แต่มีข้อเสียคือการเขียนโปรแกรมควบคุมให้ตัวอักษรเลื่อนทำได้ยากกว่าการสแกนแบบ หลัก สำหรับโครงการนี้จะใช้วิธีการสแกนทางแถวเพื่อควบคุมตัวอักษรและรูปภาพ

2.3 การสื่อสารแบบอนุกรม

การสื่อสารแบบอนุกรมนั้นจะแบ่งออกได้เป็น 2 แบบ คือการสื่อสารอนุกรมแบบซิงโครนัสและการสื่อสารอนุกรมแบบอะซิงโครนัส การสื่อสารแบบซิงโครนัสจะมีสัญญาณนาฬิกา ร่วมอยู่กับการรับ และส่งสัญญาณด้วย ตัวอย่างการส่งข้อมูลแบบซิงโครนัสก็คือคีย์บอร์ดของคอมพิวเตอร์ ซึ่งสายเส้นหนึ่ง จะเป็นสายของสัญญาณนาฬิกา ส่วนสายอีกเส้นจะเป็นสายของข้อมูล ดังนั้นการติดต่อกันแบบซิงโครนัส จะต้องใช้สายในการเชื่อมต่ออย่างน้อยที่สุด 3 เส้นคือ สัญญาณนาฬิกา ข้อมูลและกราวด์ จากรูปที่ 2.9 แสดงไทมิงไดอะแกรมของการส่งข้อมูลแบบซิงโครนัส



รูปที่ 2.9 แสดงรูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรม

2.3.1 การสื่อสารข้อมูลแบบอะซิงโครนัส

การสื่อสารข้อมูลแบบอะซิงโครนัสคือการรับและส่งข้อมูลไปในสายโดยไม่จำเป็นต้องมีสัญญาณนาฬิกาที่ร่วมด้วยเหมือนกับการส่งข้อมูลแบบซิงโครนัส แต่จะใช้การกำหนดค่าสัญญาณนาฬิกาทั้งภาคส่งและภาครับให้มีค่าเท่ากัน ซึ่งเรียกสัญญาณนาฬิกาที่ใช้ในการกำหนดค่าให้ภาครับและภาคส่งนี้ว่า อัตราการถ่ายเทข้อมูลหรือบอดเรต (Baudrate) มีหน่วยเป็นบิตต่อวินาที (bit per second: bps)

รูปแบบของข้อมูลที่ใช้ในการรับส่งแบบอะซิงโครนัสประกอบด้วย 4 ส่วนด้วยกันคือ

1. บิตเริ่มต้น (Start Bit) ซึ่งจะมีขนาด 1 บิต
2. บิตข้อมูลแบบอนุกรมจะมีขนาด 5, 6, 7 หรือ 8 บิต
3. บิตตรวจสอบพาริตี (Parity Bit) จะมีขนาด 1 บิต หรือไม่มี
4. บิตปิดท้าย (Stop Bit) จะมีขนาด 1, 1.5 หรือ 2 บิต

จากรูปที่ 2.10 แสดงรูปแบบของข้อมูลอนุกรมแบบอะซิงโครนัส เมื่อไม่มีข้อมูลที่จะส่ง DATA จะมีสถานะลอจิก “1” ซึ่งจะเรียกสถานะนี้ว่าสถานะหยุดรอ (waiting stage) การเริ่มต้นส่งข้อมูลจะเริ่มจากการให้ขา DATA มีลอจิก “0” ด้วยช่วงระยะเวลา 1 บิต ซึ่งจะเรียกบิตนี้ว่าบิตเริ่มต้นจากนั้นบิตข้อมูลจะถูกส่งออกไป โดยเริ่มจากบิตที่มีนัยสำคัญต่ำสุด (LSB) ก่อน ซึ่งข้อมูลในไบต์ที่จะส่งอาจมีจำนวนบิต 5, 6, 7 หรือ 8 บิตก็ได้ จากนั้นจะตามด้วยบิตพาริตีซึ่งใช้เพื่อตรวจสอบความผิดพลาดที่เกิดขึ้นจากการส่งข้อมูล บิตสุดท้ายที่จะส่งคือบิตปิดท้าย ซึ่งจะทำให้ขา DATA มีสถานะลอจิก “1” อีกครั้งหนึ่งด้วยระยะเวลาอย่างน้อย 1 บิต, 1.5 บิต หรือ 2 บิต เพื่อเป็นการแสดงว่าสิ้นสุดข้อมูล



รูปที่ 2.10 แสดงรูปแบบอย่างง่ายที่สุดของข้อมูลอนุกรมแบบอะซิงโครนัส

อุปกรณ์พิเศษที่ได้รับการออกแบบมาสำหรับการรับและส่งข้อมูลแบบอะซิงโครนัสเรียกว่า Universal Asynchronous Receiver/Transmitter หรือ UART อัตราความเร็วในการรับและส่งข้อมูลของการรับส่งข้อมูลแบบอะซิงโครนัสคือค่าบอดเรต ซึ่งก็คือค่าจำนวนบิตต่อวินาทีที่ใช้ในการรับและส่งข้อมูล

บอดเรตมาตรฐานที่ใช้สำหรับพอร์ตอนุกรม RS-232 ได้แก่ 110,150,300,600,1200,2400,4800,6900 และ 19200 บิตต่อวินาที และมีค่าเพิ่มมากขึ้นตามเทคโนโลยีของคอมพิวเตอร์ซึ่งการรับและส่งแบบอนุกรมโดยไม่ผ่านโมเด็มอาจจะสามารถกำหนดค่าบอดเรตได้สูงถึง 115200 บิตต่อวินาที เนื่องจากบอดเรตคือจำนวนบิตของข้อมูลที่สามารถถ่ายทอดได้ภายใน 1 วินาที ยกตัวอย่าง ข้อมูลอนุกรมถูกส่งในลักษณะ 8 บิต ไม่มีการตรวจสอบพาริตี มีบิตเริ่มต้น 1 บิต และบิตปิดท้าย 1 บิต ความยาวของข้อมูลที่รับส่งนี้เท่ากับ 10 บิต ถ้าใช้บอดเรตในการส่งข้อมูลเท่ากับ 9600 บิตต่อวินาที ก็จะสามารถรับส่งข้อมูลได้ด้วยความเร็ว 960 ไบต์ต่อวินาที และถ้ามีการใช้พาริตีความเร็วในการรับส่งข้อมูลจะเหลือเป็น 872 ไบต์ต่อวินาที

การตรวจสอบพาริตีสามารถกำหนดให้เป็นแบบคี่ (odd), แบบคู่ (even) หรือไม่มีการตรวจสอบพาริตีก็ได้ การตรวจสอบพาริตีเป็นการตรวจสอบจำนวนรวมของบิตที่เป็นลอจิก “1” ภายในข้อมูลที่ส่งออกไป 1 ไบต์ว่ามีจำนวนรวมเป็นเลขคู่หรือคี่โดยต้องรวมพาริตีบิตเข้าไปด้วย ยกตัวอย่างข้อมูลที่ทำการส่งมีขนาด 8 บิต และมีค่าเท่ากับ 99ฐานสิบหก หรือ 10011001 ฐานสอง จะเห็นว่าข้อมูลในไบต์นี้มีจำนวนลอจิก “1” จำนวน 4 ตัว ซึ่งเป็นเลขคู่ดังนั้นถ้ากำหนดพาริตีเป็นคู่ค่าในบิตพาริตีจะต้องมีค่าลอจิกเป็น “0” แต่ถ้าพาริตีเป็นคี่ค่าที่บิตพาริตีต้องมีค่าเป็น “1” เพื่อให้ข้อมูล 1 ไบต์รวมทั้งบิตพาริตีมีจำนวนบิตที่เป็นลอจิก “1” มีจำนวนรวมกันเป็นเลขคี่

บิตพาริตีถูกสร้างขึ้นจากภาคส่งข้อมูลของ UART ซึ่งทางภาครับจะต้องทำการกำหนดคุณสมบัติการตรวจสอบพาริตีให้ตรงกันว่าจะตรวจสอบพาริตีคี่หรือพาริตีคู่ จากนั้นภาครับของ UART จะทำการตรวจสอบพาริตีที่เกิดขึ้นว่าเป็นคู่หรือคี่ โดยการนับจำนวนลอจิก “1” ทั้งหมดรวมทั้งพาริตีบิตด้วย ถ้ากำหนดพาริตีไว้เป็นคู่แต่อ่านค่าตัวเลขในการนับออกมาได้เป็นเลขคี่ทางภาครับจะแสดงข้อผิดพลาดออกมาให้ผู้ใช้งาน นับเป็นการตรวจสอบความผิดพลาดที่เกิดขึ้นเท่านั้น ถ้าข้อมูลที่ทำการส่งมีบิตที่ผิดพลาดมากกว่า 1 บิต การตรวจสอบวิธีนี้จะไม่ได้ผล สำหรับการตั้งพาริตีบิตเป็น NONE นั้นทั้งภาครับและภาคส่ง จะไม่มีการตรวจสอบพาริตี

2.3.2 มาตรฐานพอร์ตอนุกรมแบบ RS-232

มาตรฐานการเชื่อมต่อแบบอนุกรม RS-232 เป็นมาตรฐานอุตสาหกรรมที่ออกแบบมาเพื่อใช้ในการส่งข้อมูลอนุกรมแบบอะซิงโครนัส 2 ทิศทาง โดยมาตรฐาน RS-232 ในอดีตนั้นถูกออกแบบมาเพื่อการส่งผ่านข้อมูลจากคอมพิวเตอร์ไปยังโมเด็มเพียงอย่างเดียวเพื่อที่จะนำข้อมูลจากโมเด็มนี้สื่อสารผ่านสายโทรศัพท์ไปยังคอมพิวเตอร์อีกชุดหนึ่งซึ่งอยู่ห่างไกลกัน โดยคณะกรรมการที่เรียกว่า สมาคมอุตสาหกรรมอิเล็กทรอนิกส์ (Electronic Industries Association: EIA) ได้วางมาตรฐานที่มีชื่อเรียกกันว่า EIA RS-232 มาตรฐานนี้ในช่วงแรกจะใช้คอนเนคเตอร์เป็นแบบ DB-25 โดยกำหนดความยาวสูงสุดไว้ที่ 50 ฟุต มีระดับสัญญาณตั้งแต่ -3 ถึง -12 V แสดงว่ามีข้อมูล (Mark) และ +3 ถึง +12V แสดงว่าเป็นช่องว่าง (Space)

มาตรฐาน RS-232 ได้กำหนดรูปแบบของอุปกรณ์เชื่อมต่อข้อมูล (Data Terminal Equipment : DTE) กับวงจรข้อมูลปลายทาง (Data Communication Equipment : DCE) ไว้ว่า อุปกรณ์ DTE จะต้องเป็นอุปกรณ์ที่มีการประมวลผลในตัวเช่น ไมโครคอนโทรลเลอร์หรือไมโครคอมพิวเตอร์ ซึ่งมีความสามารถ

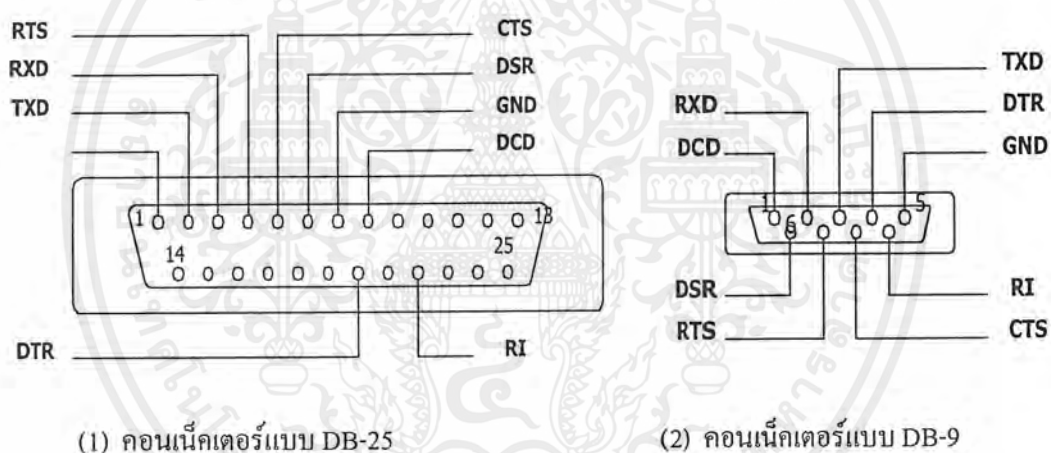
ในการสร้างบิตข้อมูลแบบอนุกรมได้ ส่วนอุปกรณ์ DCE จะทำหน้าที่เป็นเพียงตัวรับข้อมูลที่ส่งมาจาก DTE เท่านั้น โดยการรับส่งข้อมูลระหว่างอุปกรณ์ทั้งสองจะกระทำผ่านมาตรฐาน RS-232

ข้อแตกต่างของอุปกรณ์ DTE และอุปกรณ์ DCE อย่างที่เห็นได้ชัดคือ คอนเน็กเตอร์ของ DTE จะเป็นตัวผู้ส่วนคอนเน็กเตอร์ของ DCE จะเป็นตัวเมีย ซึ่งพอร์ตอนุกรมของคอมพิวเตอร์ที่ใช้กันอยู่ทั่วไปจะเป็นแบบ DTE ส่วนคอนเน็กเตอร์ที่โมเด็มจะเป็นแบบ DCE

สำหรับการใช้งานบนคอมพิวเตอร์ พอร์ตอนุกรม RS-232 มักถูกใช้เชื่อมต่อกับโมเด็มหรือเมาส์ โดยสามารถรับส่งข้อมูลได้ด้วยความยาวของสายสัญญาณสูงสุด 20 เมตร

2.3.3 คอนเน็กเตอร์สำหรับพอร์ต RS-232 และการเชื่อมต่อ

มาตรฐานการเชื่อมต่อแบบ RS-232 จะใช้คอนเน็กเตอร์แบบ DB-9 ตัวผู้หรือ DB-25 ตัวผู้ซึ่งคอนเน็กเตอร์แบบ DB-25 จะมีขาต่อใช้งานเพียง 9 เส้นเช่นเดียวกับคอนเน็กเตอร์แบบ DB-9 เนื่องจากขาอื่นๆที่เคยใช้งานในอดีต ปัจจุบันมีการใช้งานไม่มากนัก จึงถูกยกเลิกไป โดยแสดงรูปร่างและตำแหน่งขาในรูปที่ 2.11



รูปที่ 2.11 แสดงคอนเน็กเตอร์แบบ DB-25 และ DB-9

สำหรับการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกแสดงดังในรูปที่ 2.12 ลูกศรในรูปแสดงถึงทิศทางของข้อมูล ในรูปที่ 2.12 (ก) เป็นการเชื่อมต่อแบบ Null modem หรือการเชื่อมต่อโดยไม่ต้องผ่านโมเด็ม โดยมีการตรวจสอบหรือแฮนด์เช็กเต็มรูปแบบ ส่วนในรูปที่ 2.12 (ข) เป็นการเชื่อมต่อแบบ Null modem ในลักษณะที่ใช้สายสัญญาณเพียง 3 เส้น โดยเส้นหนึ่งสำหรับส่งข้อมูล อีกเส้นหนึ่งสำหรับรับข้อมูลและเส้นสุดท้ายเป็นกราวด์ สำหรับรายละเอียดหน้าที่การทำงานในแต่ละขาของพอร์ตอนุกรม RS-232 มีดังนี้

Data Carrier Detect : DCD หรืออาจเรียกว่า **Carrier Detect : CD** ขานี้จะแอกทีฟเมื่อมีการส่งสัญญาณพาห้จากอุปกรณ์สื่อสารข้อมูล เช่น โมเด็ม สำหรับการใช้งานปกติ ขานี้จะไม่ได้ถูกใช้งานมากนัก

Receive Data : RD หรือ RxD ขานี้ใช้เพื่อรับสัญญาณอนุกรมเข้ามายังคอมพิวเตอร์โดยนำข้อมูลที่อ่านได้เก็บไว้ในรีจิสเตอร์บัฟเฟอร์

ตารางที่ 2.4 แสดงการจัดขาของคอนเน็กเตอร์พอร์ตอนุกรมตามมาตรฐาน RS-232

คอนเน็กเตอร์ DB-9	คอนเน็กเตอร์ DB-25	ชื่อของสายสัญญาณ	ชนิดของสาย สัญญาณ
1	8	Data Carrier Detect : DCD	อินพุต
2	3	Received Data : RxD	อินพุต
3	2	Transmitted Data : TxD	เอาต์พุต
4	20	Data Terminal Ready : DTR	เอาต์พุต
5	7	Signal Ground : GND	-
6	6	Data Set Ready : DSR	อินพุต
7	4	Request To Send : RTS	เอาต์พุต
8	5	Clear To Send : CTS	อินพุต
9	22	Ring Indicator : RI	อินพุต

Transmitted Data : TD หรือ TxD ขานี้ใช้เพื่อส่งข้อมูลออกจากคอมพิวเตอร์ โดยนำข้อมูลที่เก็บอยู่ในบัฟเฟอร์สำหรับส่งข้อมูลออกไป

Data Terminal Ready : DTR เป็นขาสัญญาณที่ส่งออกจากคอมพิวเตอร์เพื่อให้อุปกรณ์ปลายทางรับรู้ว่า ต้องการติดต่อด้วย โดยขา DTR นี้จะต้องเชื่อมต่อกับขา DSR ของอุปกรณ์ปลายทางและขา DTR ของอุปกรณ์ปลายทางจะต้องเชื่อมต่อกับขา DSR ของคอมพิวเตอร์ ถ้าใช้การเชื่อมต่อแบบ Null modem ซึ่งใช้สายในการเชื่อมต่อเพียง 3 เส้นจะต้องต่อขา DTR และ DSR ของตัวมันเองเข้าด้วยกันและต้องต่อกับขา DCD ด้วยในกรณีที่โปรแกรมสื่อสารที่ใช้มีการตรวจจับสัญญาณพาห์

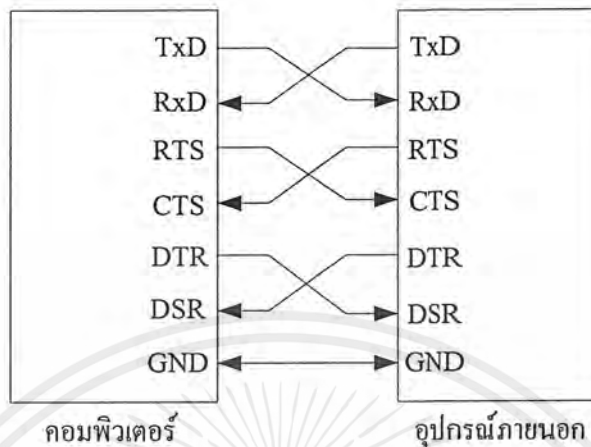
Signal Ground : GND ขากราวด์ของระบบ

Data Set Ready : DSR ขานี้ใช้คู่กับขา DTR เพื่อตรวจสอบการเชื่อมต่อกันระหว่างคอมพิวเตอร์กับอุปกรณ์ปลายทาง ซึ่งขา DSR นี้จะเป็นขาสำหรับรับข้อมูลจากภายนอก ซึ่งถูกส่งมาจากขา DTR

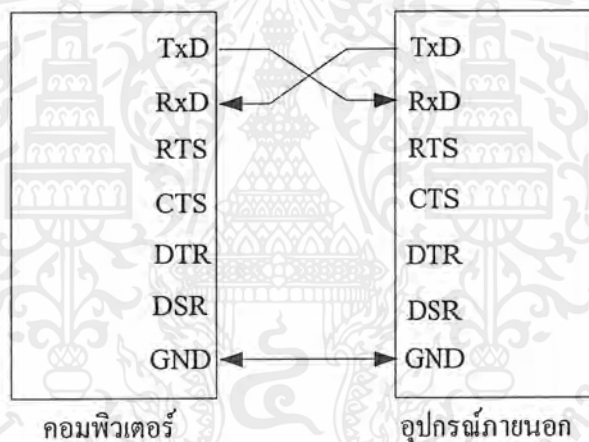
Request To Send : RTS เป็นขาสำหรับส่งสัญญาณร้องขอทางอุปกรณ์ปลายทางส่งข้อมูลกลับมายังคอมพิวเตอร์ โดยขาที่รับสัญญาณ RTS ก็คือขา CTS ในกรณีที่ใช้การเชื่อมต่อแบบ Null modem 3 สาย จะต้องเชื่อมต่อกับขา RTS และ CTS ของตัวมันเองเข้าด้วยกัน เพื่อจะให้การรับและส่งข้อมูลสามารถเกิดขึ้นได้ตลอดเวลา

Clear To Send : CTS ขานี้จะคอยรับสัญญาณจากขา RTS เมื่อรับสัญญาณได้ ข้อมูลที่ขา TxD จะถูกส่งออกไป ดังนั้นที่ขานี้จึงถูกใช้เพื่อตรวจสอบอุปกรณ์ต่อพ่วงว่าพร้อมที่จะรับข้อมูลหรือไม่

Ring Indicator : RI ใช้แสดงสถานะสัญญาณเรียกจากสายโทรศัพท์ ปกติในการสื่อสารโดยทั่วไปสายนี้จะไม่ถูกใช้งาน จะใช้งานก็ต่อเมื่อมีการเชื่อมต่อกับโมเด็มและโปรแกรมมีการตรวจสอบสัญญาณนี้เท่านั้น



(ก) แสดงการต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ Null modem



(ข) แสดงการต่ออุปกรณ์ภายนอกเข้ากับคอมพิวเตอร์แบบ RS-232 โดยใช้สายสัญญาณเพียง 3 เส้น
รูปที่ 2.12 แสดงการต่ออุปกรณ์ภายนอกกับพอร์ตอนุกรมของคอมพิวเตอร์ในลักษณะต่างๆ

2.4 หลักการพื้นฐานของ Delphi

การสร้างแอปพลิเคชันบน Windows ในยุคปัจจุบัน สามารถทำได้ง่ายสะดวกกว่าในยุคแรกๆ ที่จะต้องเขียนโค้ดใส่ไว้ในไฟล์เป็นจำนวนมาก แล้วจึงค่อยสั่งให้แปลภาษา และยุ่งยากในการแก้ไขข้อผิดพลาดของโปรแกรม (debug) อีกทั้งยากที่จะเห็นผลลัพธ์ที่ได้ว่าจะเป็นไปตามที่ต้องการหรือไม่ แต่ในยุคนี้จะมีเครื่องมือในการสร้างแอปพลิเคชันมากมาย ที่สามารถเห็นได้ตั้งแต่ในขณะที่กำลังสร้าง มีทั้งที่ใช้ภาษา C ภาษา Basic และภาษา Pascal ซึ่งก็คือ Delphi นั่นเอง

Delphi เป็นเครื่องมือในการพัฒนาแอปพลิเคชัน ที่นอกจากจะใช้ง่ายและสามารถเห็นผลลัพธ์ที่ได้ตั้งแต่ในขณะที่กำลังสร้างแล้ว ยังมีข้อดีที่เหนือกว่าภาษาอื่น ๆ คือ ในแง่ของการเขียนโค้ดทำได้ง่าย

กว่าภาษา C และแอปพลิเคชันที่ได้จะสามารถทำงานได้ประสิทธิภาพสูงเกือบเท่าภาษา C ซึ่งเร็วกว่าภาษา Basic มาก

Delphi สามารถใช้พัฒนาแอปพลิเคชันได้ทุกระดับทุกประเภทครบถ้วนในตัวเดียว โดยไม่ต้องใช้ภาษาอื่นเพิ่มเติมอีกเลยไม่ว่าจะเป็นการสร้างโปรแกรมง่ายๆ สำหรับมือสมัครเล่นหรือใช้ทำงานที่สลับซับซ้อนสำหรับมืออาชีพ อีกทั้งยังรองรับระบบปฏิบัติการทั้ง Windows95, 98 และแม้แต่ Windows NT

Delphi เป็นเครื่องมือที่ใช้ในการพัฒนาโปรแกรมบน Windows โดยใช้ภาษาปาสคาล (Pascal) เป็นหลักในการพัฒนาโปรแกรม ซึ่งเป็นโครงสร้างภาษาที่เขียนง่าย และถึงแม้ว่าผู้ใช้จะไม่มีความรู้เกี่ยวกับภาษาปาสคาลเลยก็ไม่ได้เป็นอุปสรรคต่อการพัฒนาโปรแกรมแต่อย่างใด และเนื่องจาก Delphi มีเครื่องมือช่วยเหลือในการนำคำสั่งต่าง ๆ มาใช้งานได้อย่างสะดวกและรวดเร็ว

Delphi เป็นเครื่องมือสำหรับการพัฒนาโปรแกรม มีสิ่งอำนวยความสะดวกไว้อย่างครบถ้วน โดยมีสภาพแวดล้อมในการทำงาน (Development Environment) ที่ช่วยให้สามารถทำทุกอย่างได้จากใน Delphi เองมีเครื่องมือทุกชนิดที่จำเป็นสำหรับการสร้างแอปพลิเคชันบน Windows ทั้งในส่วนของการติดต่อกับผู้ใช้ การแสดงผลกราฟิก การติดต่อกับฐานข้อมูล การจัดการระบบ ตลอดจนการพัฒนาแอปพลิเคชันเพื่อทำงานบนอินเทอร์เน็ต และด้วยคุณสมบัติที่มีอยู่อย่างมากมายของ Delphi พอดีแยกได้เป็นข้อ ๆ ดังนี้

2.4.1 วิชาลโปรแกรมมิง (Visual Programming)

การพัฒนาโปรแกรมแบบวิชวล ก็คือการพัฒนาโดยเห็นผลที่จะเกิดขึ้นเมื่อรันโปรแกรมได้ตั้งแต่ในขณะที่กำลังสร้าง โดยการนำชิ้นส่วนต่าง ๆ ที่ต้องการ ได้แก่ ปุ่ม (Button), ข้อความ (Label), รูปภาพ (Image) ฯลฯ ซึ่งเหล่านี้เรียกโดยรวมๆ ว่าคอมโพเนนต์ (Component) นำมาวางบนวินโดว์ที่เรียกว่าฟอร์ม (Form) ปรับขนาดและตำแหน่งรวมทั้งคุณสมบัติต่าง ๆ ของคอมโพเนนต์ และแม้แต่ฟอร์มเองก็ได้ผลตามที่ต้องการ และการเปลี่ยนคุณสมบัติเหล่านี้จะมีผลตั้งแต่ในขณะที่กำลังออกแบบและเมื่อรันโปรแกรมก็จะได้ผลลัพธ์เหมือนกับที่เห็นในขณะที่ยังออกแบบ

2.4.2 การโปรแกรมเชิงวัตถุ (Object Oriented Programming - OOP)

การโปรแกรมเชิงวัตถุเป็นการพัฒนาโปรแกรมโดยการสร้าง วัตถุ หรือ ออบเจ็กต์ (Object) ที่ต้องการ ในมุมมองของตัววัตถุเองว่าต้องการให้มีลักษณะเป็นอย่างไรและสามารถทำอะไรได้บ้างแทนที่จะมองที่การสร้างรูทีน (Routine) หรือ โพรซีเจอร์ (Procedure) เป็นหลักเช่นดังก่อน ๆ นี้ ประโยชน์ที่ได้ก็คือเราสามารถสร้างวัตถุโดยเริ่มจากวัตถุที่ง่ายไม่ซับซ้อนเป็นพื้นฐานขึ้นมาเสียก่อน จากนั้นจึงนำวัตถุเหล่านั้นมาตกแต่งปรับปรุงให้มีความสามารถมากขึ้นหรือทำงานได้หลากหลายขึ้น โดยนำสิ่งที่เหมือนกันหรือใช้ร่วมกันมาไว้ในวัตถุ ซึ่งจะเรียกว่า Base Object หรือ Base Class จากนั้นจึงแต่งเติม Base Object นี้ให้กลายเป็นออบเจ็กต์อื่น ๆ ตามที่ต้องการ เราสามารถนำวัตถุที่ได้นี้ก็กลับมาใช้ใหม่ได้เรื่อย ๆ ในแอปพลิเคชันต่าง ๆ และแต่งเติมต่อไปได้เรื่อย ๆ เช่นกัน

สิ่งสำคัญในการเขียนโปรแกรมเชิงวัตถุคือ จะต้องมีความเข้าใจเกี่ยวกับโครงสร้างของออบเจ็กต์ และหลักการของ OOP ดังนี้

1. โครงสร้างของออบเจกต์

ออบเจกต์ทุก ๆ ออบเจกต์จะต้องมีโครงสร้างดังต่อไปนี้

ชนิดของออบเจกต์ ออบเจกต์แต่ละออบเจกต์จะถือว่าเป็นคนละชนิด (Type) กัน ก็ต่อเมื่อนำออบเจกต์ไปสร้างต่อให้เป็นออบเจกต์ใหม่ก็จะต้องเป็นชนิดใหม่เสมอหรือสามารถเรียกได้ว่าเป็นคนละคลาส (Class) กันนั่นเอง ชนิดของออบเจกต์ได้แก่ ออบเจกต์ชนิดป้อนและชนิดข้อความ เป็นต้น

คุณสมบัติหรือเรียกว่า “พร็อพเพอร์ตี้” (Property) หมายถึง คุณลักษณะของออบเจกต์แต่ละตัวสามารถกำหนดให้แตกต่างกันไปตามความต้องการที่แตกต่างกัน เช่น ขนาดและสีของปุ่ม หรือแม้แต่ข้อความของออบเจกต์ที่แสดงอยู่บนปุ่ม เป็นต้น

พฤติกรรมหรือเมธอด (Method) คือความสามารถในการทำงานของออบเจกต์ ตัวอย่างเช่น การแสดงปุ่ม (Show) หรือการซ่อนปุ่ม (Hide) เป็นต้น

2. หลักการของ OOP

คุณสมบัติของการ โปรแกรมที่จะถือได้ว่าเป็นเชิงวัตถุได้นั้นจะต้องมีคุณสมบัติในการซ่อนเร้นการทำงานนี้ไว้ภายใน สามารถนำคุณสมบัติที่มีไปดัดแปลงใช้งานต่อๆไปได้ และอาจจะเปลี่ยนแปลงเป็นออบเจกต์ใหม่ได้ ดังรายละเอียดต่อไปนี้

Encapsulation เป็นการซ่อนเร้นส่วนของการทำงานภายในออบเจกต์ ที่ไม่เกี่ยวข้องกับภายนอกไว้ไม่ให้เห็นและไม่ให้แก้ไขเปลี่ยนแปลงส่วนที่ซ่อนไว้ ซึ่งเรานำออบเจกต์นี้ไปใช้หรือนำไปดัดแปลงได้เฉพาะส่วนที่ออบเจกต์นั้นยอมให้เท่านั้น

Inheritance เป็นการสืบทอดคุณสมบัติของออบเจกต์ เมื่อนำออบเจกต์ใดๆ ไปสร้างเป็นออบเจกต์ใหม่ คุณสมบัติของออบเจกต์เดิมจะยังคงมีอยู่ และยังสามารถเรียกใช้และทำงานได้อย่างครบถ้วน

Polymorphism คือลักษณะการทำงานที่แตกต่างกันของคุณสมบัติหรือพฤติกรรมเดียวกันแต่เป็นของออบเจกต์คนละชนิดกัน ตัวอย่างเช่น เมธอด Save To File ของเมโมจะได้ text file ที่เก็บข้อความนั้น ในขณะที่ Save To File ของอิมเมจจะได้ไฟล์รูปภาพ ซึ่งแม้จะเป็นการบันทึกข้อมูลไว้เป็นไฟล์เหมือนกัน แต่วิธีการบันทึกก็ต่างกัน

Delphi เป็นคอมไพเลอร์ที่ใช้แปลภาษาโปรแกรมเป็นภาษาเครื่อง ซึ่งหลังจากการคอมไพล์ โปรแกรมเราจะได้ไฟล์ .exe ซึ่งเก็บภาษาของเครื่องนั้น ๆ และสามารถทำงานได้เลยโดยไม่ต้องผ่านขั้นตอนการแปลภาษาในระหว่างการทำงานอีก ทำให้สามารถทำงานได้เร็วและไม่มีขีดจำกัด

2.4.3 คอมโพเนนต์ไลบรารี (Component Library)

คอมโพเนนต์เป็นส่วนประกอบย่อย ๆ ที่เราจะนำมาใช้ในการสร้างแอปพลิเคชัน ซึ่งใน Delphi มีคอมโพเนนต์ให้เลือกใช้อยู่เป็นจำนวนมาก โดยจะเก็บอยู่ใน “ห้องสมุด” หรือคอมโพเนนต์ไลบรารี (Component Library) และจัดแยกเป็นกลุ่มๆ ตามประเภทของการใช้งาน

2.4.4 OCX และ ActiveX

OCX คือออบเจกต์ที่ใช้ใน Visual Basic ของไมโครซอฟท์ ซึ่งมีลักษณะคล้ายกับคอมโพเนนต์

ของ Delphi ส่วน ActiveX ก็เป็นออบเจกต์ที่ถูกพัฒนามาในรูปแบบเดียวกับ OCX แต่มีวัตถุประสงค์เพื่อใช้งานบนอินเทอร์เน็ต ซึ่งทั้ง OCX และ ActiveX ต่างก็เป็นออบเจกต์ที่นำมาใช้กับ Delphi ได้

2.4.5 Wizard และ Object Repository

Delphi มี Wizard ที่ใช้ในการสร้างฟอร์มและออบเจกต์ที่ต้องใช้บ่อย ๆ หรือสร้างได้ยาก ไว้ให้ใช้เป็นจำนวนมาก ตัวอย่างเช่น Wizard ในการสร้างรายงานแบบต่างๆ หรือฟอร์ม เช่น About Box และ Dialog Box ซึ่งจะเรียกอีกอย่างหนึ่งว่า เทมเพลต (Template) ก็ได้ เหล่านี้ล้วนช่วยให้การสร้างแอปพลิเคชันเป็นไปได้อย่างรวดเร็วและมีความกลมกลืนมากขึ้น นอกจากนี้ยังสามารถเก็บฟอร์มหรือออบเจกต์ที่คิดว่าต้องใช้บ่อย เพิ่มเข้าไปไว้ให้เรียกใช้ในแอปพลิเคชันอื่นๆ ได้อีกด้วยโดยการบันทึกไว้ใน Object Repository จากนั้นเมื่อต้องการใช้งานก็เรียกใช้ได้เลยเหมือนกับการใช้ Wizard ทั่วไป

2.4.6 การติดต่อกับฐานข้อมูล

การติดต่อกับฐานข้อมูลนั้น Delphi มีคอมโพเนนต์ที่สามารถเชื่อมต่อ เพื่อจัดการกับข้อมูลที่เก็บอยู่ในฐานข้อมูลทุกประเภท ไม่ว่าจะเป็นการเพิ่ม ลบ แก้ไขหรือการเรียกดูข้อมูล โดยผู้ใช้ไม่ต้องเขียนชุดคำสั่งใดๆ ในโปรแกรมเลยก็สามารถสร้างแอปพลิเคชันอย่างง่าย ๆ ที่ทำงานกับฐานข้อมูลขึ้นมาได้แล้ว

เราสามารถให้ Delphi จัดการกับฐานข้อมูลที่เป็นแบบง่าย ๆ ซึ่งได้แก่ dBase หรือ Paradox และ MS Access ซึ่งพวกนี้จะเรียกว่าเป็น Local Database คือ Database ที่ทำงานในเครื่องนั่นเองหรือเรียกอีกอย่างหนึ่งว่าเป็นแบบ File oriented Database ก็ได้เพราะเป็นการเก็บข้อมูลไว้ในโครงสร้างของไฟล์นอกจากนี้ยังสามารถใช้งานกับระบบฐานข้อมูลที่เป็น Database Server ต่างๆ เช่น SQL Server หรือ InterBase ซึ่งอาจจะทำงานอยู่ในเครื่องเดียวกันกับแอปพลิเคชันในกรณีของ Local InterBase หรืออาจจะทำงานอยู่บนเครื่อง Server เครื่องอื่น ซึ่งเรียกว่าเป็นการใช้ฐานข้อมูลในแบบ Client/Server ก็ยังได้

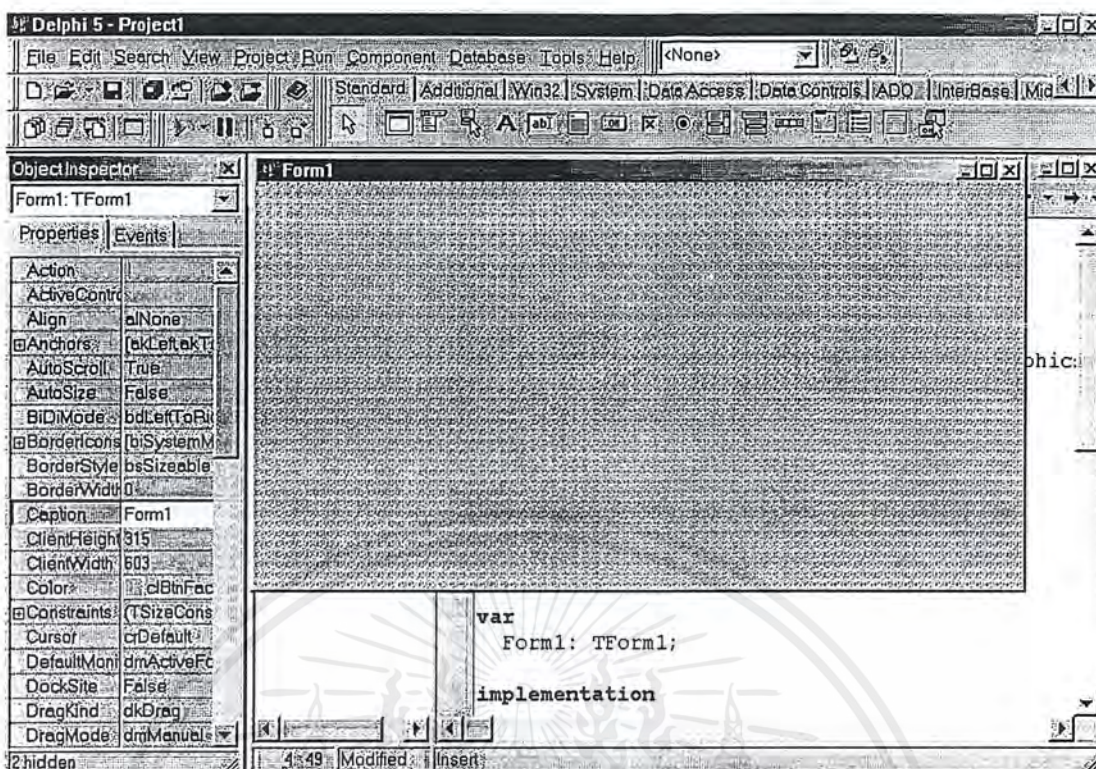
2.4.7 สภาพแวดล้อมของ Delphi

การสร้างโปรแกรมใน Delphi จะมีรูปแบบที่เรียกว่า Integrated Development Environment (IDE) คือเป็นการสร้างโปรแกรมโดยการทำงานทุกอย่างสามารถทำได้จากในที่แห่งเดียวโดยไม่จำเป็นต้องใช้ Editor ตัวหนึ่งเขียนโปรแกรม และคอมไพล์ด้วยคำสั่งที่เป็น Command Line

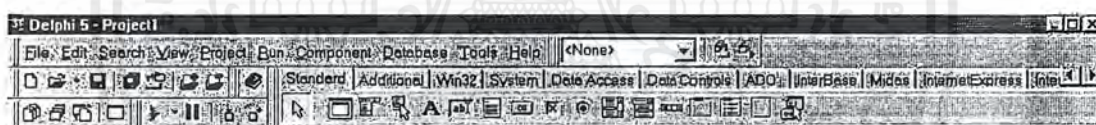
หลังจากที่รันโปรแกรม Delphi แล้วเราจะเห็นหน้าต่างของ Delphi ซึ่งภายในประกอบด้วยวินโดว์ต่าง ๆ ดังนี้

1. วินโดว์หลัก (Main Window)

วินโดว์หลักทำหน้าที่เป็นศูนย์กลางสำหรับควบคุมกระบวนการในการพัฒนาโปรแกรม ไม่ว่าจะเป็นการจัดการเกี่ยวกับไฟล์ภายในแอปพลิเคชัน การคอมไพล์ และการตรวจสอบข้อผิดพลาด เมื่อรันโปรแกรม Delphi จะเห็นวินโดว์ที่อยู่บนสุด โดยภายในวินโดว์หลักจะแบ่งออกเป็น 4 ส่วนคือ ไตเติลบาร์ เมนูบาร์ สปีดบาร์ และคอมโพเนนต์พาเลต ดังนี้



รูปที่ 2.13 แสดงวินโดว์ของDelphi



รูปที่ 2.14 แสดงวินโดว์หลัก

ไตเติลบาร์ (Title Bar) เป็นส่วนที่อยู่บนสุดแสดงชื่อโปรแกรม Delphi และชื่อของโปรเจกต์ที่ทำงานอยู่ในขณะนั้น

เมนูบาร์ (Menu Bar) เป็นส่วนประกอบที่จะสามารถพบได้ในแอปพลิเคชันบน Windows ทั่วไป โดยจะอยู่ใต้ไตเติลบาร์ภายในเมนูบาร์จะเป็นส่วนที่แสดงเมนูคำสั่งของ Delphi ซึ่งจะใช้ในการทำงานต่างๆ เกือบทั้งหมด เช่น เมนู File ใช้ทำงานกับแฟ้มข้อมูล ไม่ว่าจะเป็นการเปิด การปิด หรือการบันทึกแฟ้มข้อมูล เป็นต้น

สปีดบาร์ (Speed Bar) คือกลุ่มของคำสั่งที่ใช้อยู่ ซึ่งจะแสดงอยู่ใต้เมนูบาร์ทางด้านซ้ายของวินโดว์หลัก ภายในสปีดบาร์จะประกอบด้วยปุ่มแทนรายการต่างๆ ของเมนู เมื่อคลิกเมาส์ที่ปุ่มใดก็จะเหมือนกับการเลือกคำสั่งจากเมนู และเมื่อเลื่อนเมาส์ไปหยุดบนปุ่มใดก็จะเห็นกรอบข้อความ (tooltips) แสดงชื่อเมนู และฟังก์ชันคีย์ซึ่งเป็นคีย์ลัดสำคัญสำหรับใช้แทนปุ่มนั้น

คอมโพเนนต์พาเลตต์ (Component Palette) คือส่วนที่อยู่ถัดจากสปีดบาร์ไปทางด้านขวาประกอบด้วยคอมโพเนนต์สำหรับสร้างแอปพลิเคชัน โดยจัดกลุ่มไว้ในแท็บต่างๆ คลิกที่แท็บเพื่อแสดงคอมโพ

เนตในแต่ละกลุ่มและเมื่อเลื่อนเมาส์ไปหยุดบนคอมโพเนนต์ใดก็จะปรากฏกรอบข้อความแสดงชื่อคอมโพเนนต์นั้นขึ้น

2. วินโดว์ออบเจกต์อินสเปกเตอร์ (Object Inspector Window)

วินโดว์ออบเจกต์อินสเปกเตอร์เป็นวินโดว์ซึ่งอยู่ใต้วินโดว์หลักทางด้านซ้าย ใช้สำหรับแสดงและปรับแต่งค่าพรีออเพอร์ตี้ (property) และอีเวนต์ (event) ของคอมโพเนนต์ จากรูปที่ 2.15 จะเห็นว่าได้ไคเดิลบาร์ของวินโดว์ออบเจกต์อินสเปกเตอร์ จะเป็นคอมโบบ็อกซ์สำหรับแสดงชื่อและชนิดของคอมโพเนนต์ที่ทำงานอยู่ในขณะนั้น ในรูปเป็น “Form1 TForm1” แสดงว่าเรากำลังทำงานกับคอมโพเนนต์ชื่อ Form1 และมีชนิดของคอมโพเนนต์เป็น TForm1 ถัดลงมาจะมีแท็บ 2 แท็บคือแท็บ Properties และแท็บ Events

แท็บ Properties ใช้สำหรับแสดงและกำหนดค่าคุณสมบัติหรือพรีออเพอร์ตี้ (property) ของคอมโพเนนต์ภายในแท็บแบ่งออกเป็น 2 คอลัมน์โดยคอลัมน์ทางซ้ายนั้นจะแสดงชื่อพรีออเพอร์ตี้ ส่วนคอลัมน์ทางขวานั้นมีไว้สำหรับแสดงและแก้ไขค่าของพรีออเพอร์ตี้

แท็บ Events ใช้สำหรับกำหนดการทำงาน เมื่อมีเหตุการณ์เกิดขึ้นหรือการกระทำใดๆ เกิดขึ้นกับคอมโพเนนต์ ซึ่งอาจเกิดจากการกระทำของผู้ใช้หรือตัวโปรแกรมเอง ตัวอย่างเช่น การใช้เมาส์คลิกที่ปุ่ม จะเกิดอีเวนต์ OnClick ขึ้นกับปุ่มนั้นหรือการกด Enter บน EditBox ก็เกิดอีเวนต์ OnKeyDown, OnKeyUp และ OnKeyPress ขึ้นมาเมื่อใช้เมาส์คลิกที่แท็บอีเวนต์ในวินโดว์ออบเจกต์อินสเปกเตอร์ดังที่แสดงในรูปที่ 2.15 จะเห็นว่าภายในแบ่งออกเป็น 2 คอลัมน์ คอลัมน์ทางซ้ายแสดงชื่ออีเวนต์ของคอมโพเนนต์ ส่วนคอลัมน์ทางขวาจะแสดงชื่อโพรซีเยอร์ที่จะทำงานเมื่อเกิดอีเวนต์ขึ้น

3. วินโดว์ฟอร์ม (Form Window)

ฟอร์มเป็นวินโดว์ที่ใช้สำหรับออกแบบส่วนที่ติดต่อกับผู้ใช้ โดยการนำคอมโพเนนต์ต่างๆ จากคอมโพเนนต์พาเลตมาวางลงบนฟอร์ม

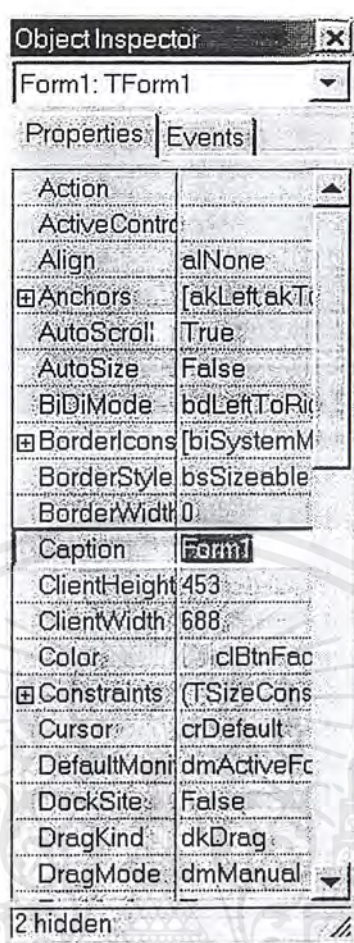
เมื่อเปิดโปรแกรม Delphi ขึ้นมาหรือสร้างแอปพลิเคชันใหม่ Delphi จะสร้างฟอร์มให้หนึ่งฟอร์มเสมอในแอปพลิเคชันหนึ่งๆ จะประกอบด้วยฟอร์มอย่างน้อยหนึ่งฟอร์มเสมอ

4. วินโดว์โค้ดเอดิเตอร์ (Code Editor Window)

วินโดว์โค้ดเอดิเตอร์มีไว้สำหรับเขียนโปรแกรมโดยอาจจะถูกบังคับอยู่ได้ฟอร์ม เมื่อเลื่อนวินโดว์ฟอร์มออกไปหรือคลิกที่ปุ่ม minimize ของฟอร์ม จะเห็นวินโดว์โค้ดเอดิเตอร์ปรากฏดังรูปที่ 2.17

ภายในวินโดว์โค้ดเอดิเตอร์ จะประกอบด้วยส่วนต่างๆ ดังต่อไปนี้

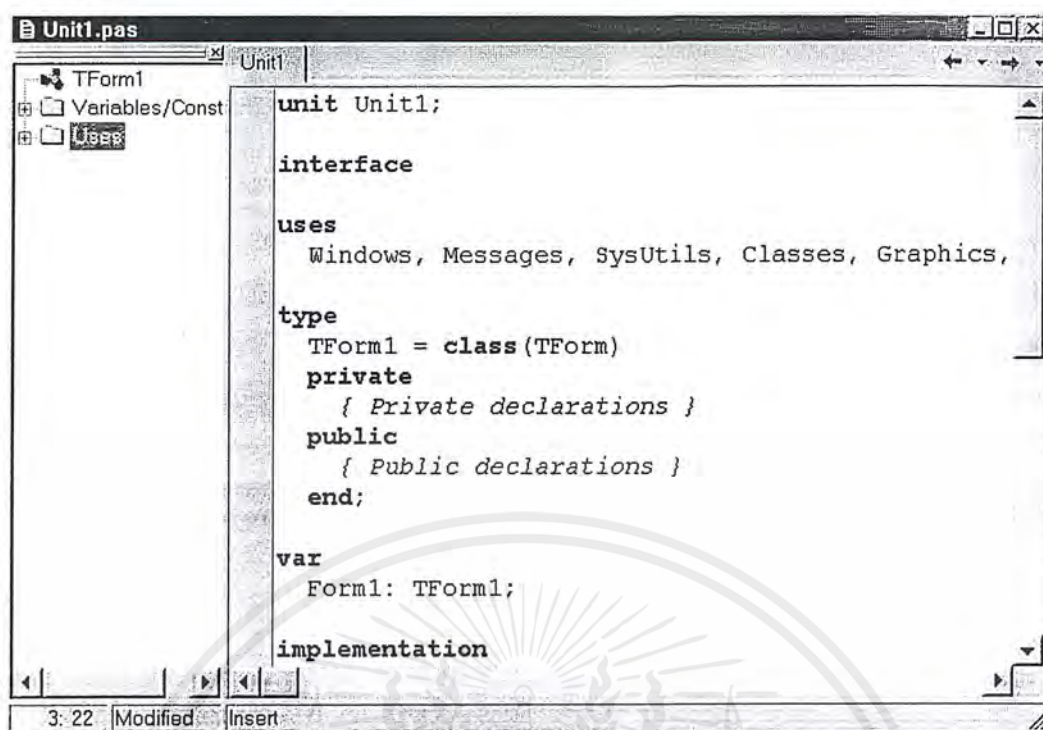
Unit Explorer เป็นกรอบที่อยู่ทางด้านซ้าย ซึ่งใช้สำหรับแสดงสิ่งต่างๆ ที่มีอยู่ในยูนิิต เช่น ออบเจกต์ ตัวแปร ค่าคงที่ และชื่อของยูนิิตต่างๆ ที่ถูกอ้างอิงถึงในยูนิิต ลักษณะของการแสดงรายละเอียดจะอยู่ในรูปแบบของโครงสร้างต้นไม้ (Tree View)



รูปที่ 2.15 แสดงวินโดวออบเจกต์อินสเปกเตอร์



รูปที่ 2.16 แสดงวินโดวโฟร์ม



รูปที่ 2.17 แสดงวินโดวโค้ดเอดิเตอร์

Code editor โค้ดเอดิเตอร์เป็นส่วนที่ใช้เขียนคำสั่ง โดยมีแท็บของชื่อไฟล์ที่ใช้เก็บยูนิตนั้น สำหรับเลือกยูนิตที่ต้องการ

สถานะการทำงาน อยู่ทางด้านล่างของวินโดวโค้ดเอดิเตอร์ ซึ่งประกอบด้วยส่วนต่าง ๆ ดังนี้

ตำแหน่งของเคอร์เซอร์ ประกอบด้วยตัวเลข 2 ตัวคั่นด้วยเครื่องหมาย ตัวแรกแสดง หมายเลขบรรทัดและตัวที่สองแสดงหมายเลขของคอลัมน์ที่เคอร์เซอร์อยู่บน โค้ดเอดิเตอร์

สถานะการแก้ไข ใช้บอกว่ายูนิตนี้มีการแก้ไขหรือไม่ กรณีที่แก้ไขและยังไม่ได้บันทึกในช่องนี้จะแสดงคำว่า Modified แต่ถ้ายังไม่มีแก้ไข หรือการบันทึกการแก้ไขไปแล้วก็จะไม่มีข้อความใด ๆ ปรากฏในช่องนี้

สถานะการพิมพ์ ใช้บอกสถานะการพิมพ์ในขณะนั้น เช่น เป็นการพิมพ์แทรก (Insert) หรือพิมพ์ทับ (Overwrite)

5. ไฟล์ต่างๆ ในโปรเจก

ในโปรเจกหนึ่งๆ ที่สร้างขึ้นจะประกอบไปด้วยไฟล์ต่างๆ ซึ่งมีหน้าที่และการทำงานที่แตกต่างกัน บางไฟล์จะถูกสร้างขึ้นในตอนออกแบบ แต่บางไฟล์จะเกิดขึ้นหลังจากการบันทึก หรือคอมไพล์โปรเจก ซึ่งไฟล์หลักๆ มีดังนี้

- Project File (.dpr)
- Unit File (.pas)
- Form File (.dfm)

- Resource File (.res)
- Project GroupFile (.bpg)

Project File เป็นไฟล์ที่รวบรวมรายละเอียดของโปรเจกต์ ว่าภายในโปรเจกต์หนึ่งนั้นประกอบไปด้วยฟอร์มหรือยูนิตอะไรบ้างนอกจากนั้นยังทำหน้าที่สร้างฟอร์มและเรียกฟอร์มหลักของแอปพลิเคชันขึ้นมาทำงานอีกด้วย ในแอปพลิเคชันหนึ่งๆจะมีโปรเจกต์ไฟล์เพียงไฟล์เดียว โดยมีนามสกุลเป็น .dpr และจะถูกสร้างขึ้นเมื่อเราสร้างโปรเจกต์ใหม่ ซึ่งถ้าเป็นโปรเจกต์แรก Delphi จะกำหนดชื่อดีฟอลต์สำหรับโปรเจกต์เป็น Project1.dpr และเมื่อคอมไพล์โปรเจกต์เราจะได้ไฟล์ .exe ที่มีชื่อเดียวกับชื่อโปรเจกต์

Unit File Delphi จะแยกการทำงานภายในโปรเจกต์ออกเป็นโมดูลย่อยๆ เรียกว่ายูนิต (Unit) ซึ่งแต่ละยูนิตจะเก็บรายละเอียดของตัวแปร ค่าคงที่ ชนิดข้อมูล โปรซีเยอร์และฟังก์ชัน เพื่อใช้ในการทำงานภายในยูนิตนั้นๆ ปกติ Delphi จะกำหนดให้ยูนิตไฟล์มีนามสกุลเป็น .pas ซึ่งจริงๆแล้วเราสามารถกำหนดเป็นนามสกุลอื่นๆก็ได้ เพียงแต่ว่า .pas เป็นนามสกุลมาตรฐานของภาษาปาสคาล หลังจากที่เราคอมไพล์หรือรันโปรเจกต์ Delphi ก็สร้างไบনারีไฟล์ให้สำหรับแต่ละยูนิต โดยมีชื่อเหมือนกับชื่อยูนิตไฟล์แต่นามสกุลเป็น .dcu ภายในยูนิตไฟล์จะมีส่วนที่บอกรายละเอียดเกี่ยวกับฟอร์มดังนี้

- type เป็นส่วนที่ใช้สำหรับประกาศชนิดข้อมูล โดยมีการประกาศให้ TForm1 เป็นคลาส (ชนิดของออพเจกต์) ที่สืบทอดคุณสมบัติมาจากออพเจกต์ TForm และภายในคลาสมีกการกำหนดขอบเขตของคลาสด้วยว่า private และ public โดยคำว่า private หมายถึงสิ่งต่างๆที่ประกาศไว้ในส่วนนี้ จะเป็นที่ยุ้กและใช้งานได้เฉพาะภายในคลาสนี้เท่านั้น ส่วนคำว่า public หมายถึงสิ่งที่ประกาศไว้ในส่วนนี้จะเป็นที่ยุ้กและใช้งานได้ในคลาสนี้และที่อื่นๆ ด้วย

- var เป็นการประกาศตัวแปร Form1 ให้มีชนิดข้อมูลเป็นออพเจกต์ TForm1 โดยชื่อของตัวแปรนี้ได้มาจากคำพรีอเพอร์ดี Name ของฟอร์มนั้นเอง

- {\$R *.DFM} เป็นคำสั่งสำหรับคอมไพเลอร์ (compiler directive) โดยเครื่องหมาย \$ แสดงว่าเป็นคอมไพเลอร์ไคเร็กทีฟและ R คือ คำสั่งที่บอกให้รวมเอา Resource จากไฟล์ของฟอร์มซึ่งมีชื่อเดียวกับยูนิตแต่นามสกุลเป็น .dfm เข้ามาด้วย

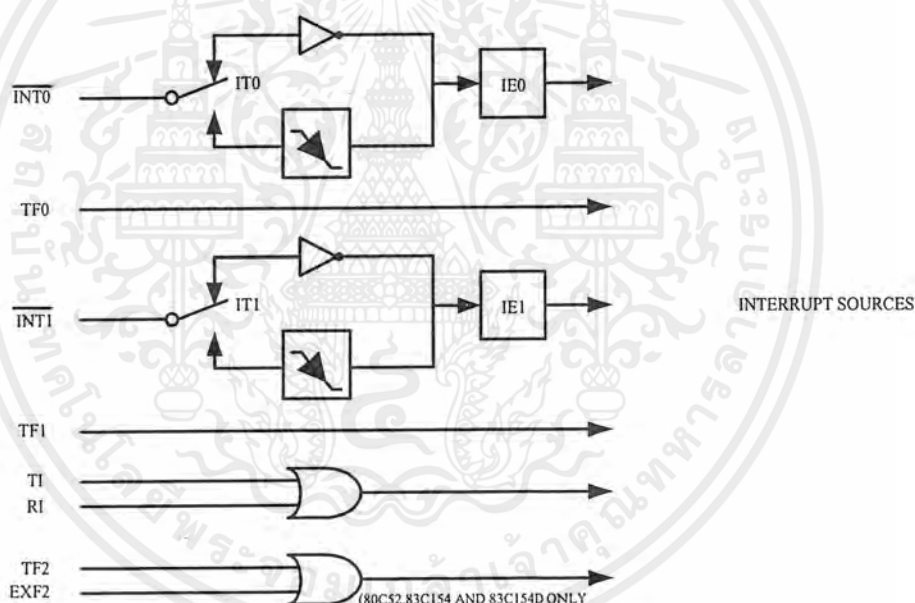
Form File เป็นไฟล์ที่เก็บรายละเอียดของออพเจกต์ต่างๆที่อยู่บนฟอร์ม รวมทั้งตัวฟอร์มเองด้วย เช่น พร็อพเพอร์ตี้ของฟอร์ม พร็อพเพอร์ตี้ของคอมโพเนนต์ต่างๆที่อยู่บนฟอร์มเป็นต้น เมื่อเราบันทึกโปรเจกต์ Delphi จะสร้างไฟล์นี้ให้โดยจะมีชื่อเหมือนกับชื่อยูนิตแต่นามสกุลเป็น .dfm

Resource File เป็นไฟล์ที่มีรูปแบบการเก็บข้อมูลที่ใช้สำหรับ Windows ซึ่งปกติจะใช้เก็บส่วนประกอบต่างๆที่เป็นไบนารี เช่น ไอคอนของแอปพลิเคชัน ที่เรากำหนดในไดอะล็อกบ็อกซ์

Project Group File เป็นไฟล์ที่ใช้เก็บรวบรวมรายการโปรเจกต์ที่ได้กำหนดไว้ให้อยู่ในโปรเจกต์เมเนเจอร์ ไฟล์นี้จะมีนามสกุลเป็น .bpg และเมื่อเราตั้ง Open Project เราก็สามารถเลือกเปิดโปรเจกต์เดี่ยวๆ (ไฟล์ .dpr) หรือเปิดโปรเจกต์กรุป (ไฟล์ .bpg) ก็ได้

2.5 การอินเทอร์รัปต์ของ MCS-51

คือการขัดจังหวะโปรแกรมชั่วคราวแล้วมาทำโปรแกรมบริการอินเทอร์รัปต์ (Interrupt Service Routine; ISR) การตรวจสอบสัญญาณการร้องขออินเทอร์รัปต์จะตรวจสอบที่ตำแหน่ง S5P2 ของทุกๆ แมชชีนไซเคิลเมื่อพบแล้วในช่วงแมชชีนไซเคิลที่ 2 จะเป็นการตรวจสอบว่าเป็นของอุปกรณ์ใด แมชชีนไซเคิลที่ 3 จะกระโดดไปทำโปรแกรมบริการอินเทอร์รัปต์ (Interrupt Service Routine; ISR) ดังแสดงค่า Vector ดังตารางที่ 2.5 อินเทอร์รัปต์ของ MCS-51 ได้มาจาก 6 แหล่งแต่มีเพียง 5 Vector สำหรับ 8031,8051 สำหรับเบอร์ 8032,8052 ขึ้นไปจะมี 8 แหล่ง 6 Vector (เพิ่ม Timer/Counter 2 เข้ามา) ดังรูปที่ 2.18 ซึ่งแสดงแหล่งที่มาของสัญญาณอินเทอร์รัปต์ จากอุปกรณ์ต่างๆ ซึ่งมีทั้งภายในชิพและภายนอกชิพ



รูปที่ 2.18 แสดงแหล่งที่มาสัญญาณอินเทอร์รัปต์

2.5.1 IE (Interrupt Enable Register)

ใช้ควบคุมอินเทอร์รัปต์ได้ 6 แหล่ง 5 Vector คือจาก (TI,RI), TF0, TF1, IE0, IE1 เราสามารถสั่งห้ามหรือไม่ห้ามการอินเทอร์รัปต์ได้จากรีจิสเตอร์ชุดนี้ ดังมีในตารางที่ 2.6

ตารางที่ 2.5 แสดงชื่อสัญญาณอินเทอร์รัปต์

ลำดับ	ชื่อสัญญาณอินเทอร์รัปต์	Vector Address	Priority
1	IE0 (จากขา INT0)	0003H	Highest
2	TF0 (Timer/Counter 0)	000BH	
3	IE1 (จากขา INT1)	0013H	
4	TF1 (Timer/Counter 0)	001BH	
5	TI + RI (จาก Serial Port)	0023H	
6	TF2 + EXF2 (จาก Timer/Counter 2)	002BH	Lowest

(TF2 + EXF2) มีเฉพาะในเบอร์ 8052 ขึ้นไป

ตารางที่ 2.6 แสดงความหมายบิตต่างๆใน (Interrupt Enable Register)

บิต	ชื่อบิต	การทำงาน
IE.7	EA	= 1 หมายถึงยอมให้เลือกการทำอินเทอร์รัปต์จากอินเทอร์รัปต์จากแหล่งต่าง ๆ ได้ = 0 หมายถึงไม่ยอมให้ทำอินเทอร์รัปต์จากแหล่งใด ๆ ทั้งสิ้น
IE.6	X	ไม่ได้ใช้งาน
IE.5	ET2	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์เมื่อ TF2 เกิดโอเวอร์โฟลว์ = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์เมื่อ TF2 เกิดโอเวอร์โฟลว์
IE.4	ES	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์จากพอร์ตสื่อสารอนุกรมได้ = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์จากพอร์ตสื่อสารอนุกรม
IE.3	ET1	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์เมื่อ TF1 เกิดโอเวอร์โฟลว์ = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์เมื่อ TF1 เกิดโอเวอร์โฟลว์
IE.2	EX1	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์จากสัญญาณภายนอกหมายเลข 1 = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์จากสัญญาณภายนอกหมายเลข 1
IE.1	ET0	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์เมื่อ TF0 เกิดโอเวอร์โฟลว์ = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์เมื่อ TF0 เกิดโอเวอร์โฟลว์
IE.0	EX0	= 1 หมายถึง ยอมให้ทำอินเทอร์รัปต์จากสัญญาณภายนอกได้ (ขา INT0) = 0 หมายถึง ไม่ยอมให้ทำอินเทอร์รัปต์จากสัญญาณภายนอก (ขา INT0)

ความหมายของสัญลักษณ์

E = Enable หรือ External ; T = Timer

0,1,2 = Channel 0, Channel 1, Channel 2

อินเทอร์รัปต์ภายใน MCS-51 ได้จาก Timer 0, Timer 1 โดยตรวจสอบที่ TF0 และ TF1 และอินเทอร์รัปต์จาก Serial Port โดยตรวจสอบที่ TI และ RI ขา INT0 และ INT1 เมื่อทำอินเทอร์รัปต์จะเป็นขา External Interrupt Input เมื่อไม่ทำอินเทอร์รัปต์ จะใช้ Start Timer/Counter หรือที่เราเรียกว่า Hardware Start การขัดจังหวะของการอินเทอร์รัปต์

เราสามารถกำหนดลำดับความสำคัญได้จาก **Interrupt Priority Register (IP)**

คำอธิบายความหมายในแต่ละบิตใน (IP)

- PT2 : 0 Timer 2 มีลำดับความสำคัญต่ำสุด
1 Timer 2 มีลำดับความสำคัญสูงสุด
- PS : 0 พอร์ตสื่อสารอนุกรมมีลำดับความสำคัญต่ำสุด
1 พอร์ตสื่อสารอนุกรมมีลำดับความสำคัญสูงสุด
- PT1 : 0 Timer 1 มีลำดับความสำคัญต่ำสุด
1 Timer 1 มีลำดับความสำคัญสูงสุด
- PT0 : 0 Timer 0 มีลำดับความสำคัญต่ำสุด
1 Timer 1 มีลำดับความสำคัญสูงสุด
- PX0 : 0 อินเทอร์รัปต์ภายนอกชนิด 0 มีลำดับความสำคัญต่ำสุด
1 อินเทอร์รัปต์ภายนอกชนิด 0 มีลำดับความสำคัญสูงสุด
- PCT : (83C154/83C154D Only)
0 ยอมให้มีการจัดลำดับความสำคัญของการอินเทอร์รัปต์ (Priority)
1 ไม่ยอมให้มีการจัดลำดับความสำคัญของการอินเทอร์รัปต์การทำหรือไม่ทำ
อินเทอร์รัปต์ควบคุมจาก IE

ถ้าทุกบิตเป็น 1 หมดจะถือว่าทุกตัวมีลำดับความสำคัญสูงสุดเท่ากัน เมื่อสัญญาณมาพร้อมกันจะ
ต้องตัดสินใจตามตารางที่ 2.5

การเลือกลักษณะของการทริก (Interrupt Type Control) ไม่ได้อยู่ในรีจิสเตอร์ที่ใช้ควบคุมการ
อินเทอร์รัปต์แต่ไปอยู่ในรีจิสเตอร์ TCON แทนดังรายละเอียดดังนี้

- Timer/Counter Register (TCON) อยู่ใน SFR ตำแหน่งที่ (088H) ใช้ทำงานหลายหน้าที่คือ
- TFx ใช้เป็นที่เก็บ Flag ของ Timer 0,1
- TRx ใช้ Start Timer Counter 0,1
- IEx ใช้เป็นที่เก็บ Flag ของสัญญาณอินเทอร์รัปต์ภายนอกชนิด 0,1
- ITx ใช้เลือกลักษณะของการทริกที่ขา INTx

บิต TF1(Timer Over Flow Flag) แสดงการเกิดโอเวอร์โฟลว์ของไทม์เมอร์เคาน์เตอร์ 1 จะเซต
เมื่อเกิดโอเวอร์โฟลว์และจะถูกเคลียร์เองเมื่อซีพียูย้ายการไปที่โปรแกรมบริการอินเทอร์รัปต์

บิต TR1 บิตควบคุมการนับของไทม์เมอร์ 1 ควบคุมจากโปรแกรม

- 1 ไทม์เมอร์เคาน์เตอร์ 1 เริ่มทำงานต่อ
0 ไทม์เมอร์เคาน์เตอร์ 1 หยุดทำงาน

บิต TF0 (Timer Over Flow Flag) บิตแสดงการเกิดโอเวอร์โฟลว์ของไทม์เมอร์เคาน์เตอร์ 0
ถูกเซตเมื่อเกิดโอเวอร์โฟลว์ เช่นเดียวกับ TF1

บิต TR0 เช่นเดียวกับ TR1 แต่ใช้ควบคุมไทม์เมอร์ เคาน์เตอร์ 0

บิต IE1 บิตแสดงสัญญาณอินเตอร์รัปต์ภายนอกหมายเลข 1 เมื่อมีสัญญาณอินเตอร์รัปต์เข้ามาที่ขา INT1 และถูกเคลียร์เองโดยคำสั่ง RETI ที่อยู่ในโปรแกรมส่วนบริการอินเตอร์รัปต์

บิต IT1 บิตเลือกประเภทการตรวจสอบสัญญาณอินเตอร์รัปต์ ที่เข้ามาทางขา INT1

1 จะตรวจสอบการเปลี่ยนระดับสัญญาณจาก 1 เป็น 0 ที่ขา INT1

0 จะตรวจสอบระดับศูนย์ของสัญญาณ ที่ขา INT1

ข้อควรสังเกต

T หมายถึง Timer/Counter หรือ Type

R หมายถึง RUN

F หมายถึง Flag

1,0 หมายถึง Channel หรือ Channel 1 บางที่ใช้คำว่า x เช่น Trx



บทที่ 3

การคำนวณและการสร้าง

3.1 แนวคิดและการทำงาน

จุดมุ่งหมายในการออกแบบและสร้างบอร์ดแสดงผล 3 สี เพื่อใช้เป็นสื่อในการนำเสนอข้อมูลต่างๆ ซึ่งจะต้องมีลักษณะเด่นสะดุดตา ดึงดูดใจผู้พบเห็นได้ จึงได้ใช้หลอดแอลอีดีที่สามารถแสดงผลได้ 3 สี มาจัดทำเป็นส่วนแสดงผล เหตุที่ไม่ใช้แอลอีดีที่เป็นโมดูลเพราะว่า มุมมองของหลอดแอลอีดีจะกว้างกว่าแบบแอลอีดีที่เป็นโมดูลและการแสดงผลจะเป็นธรรมชาติมากกว่า ในการแสดงผลหลอดแอลอีดีจะใช้เทคนิคการสแกนคือหลอดแอลอีดีจะไม่ติดตลอดเวลา จะติดและดับสลับกันทีละ 2 แถวด้วยอัตราความเร็วที่สายตามนุษย์ไม่สามารถมองเห็นการกระพริบของหลอดได้ในขณะที่ข้อมูลที่ส่งมาก็จะมีการเปลี่ยนแปลงไปเรื่อยๆ

3.2 แนวคิดในการออกแบบ

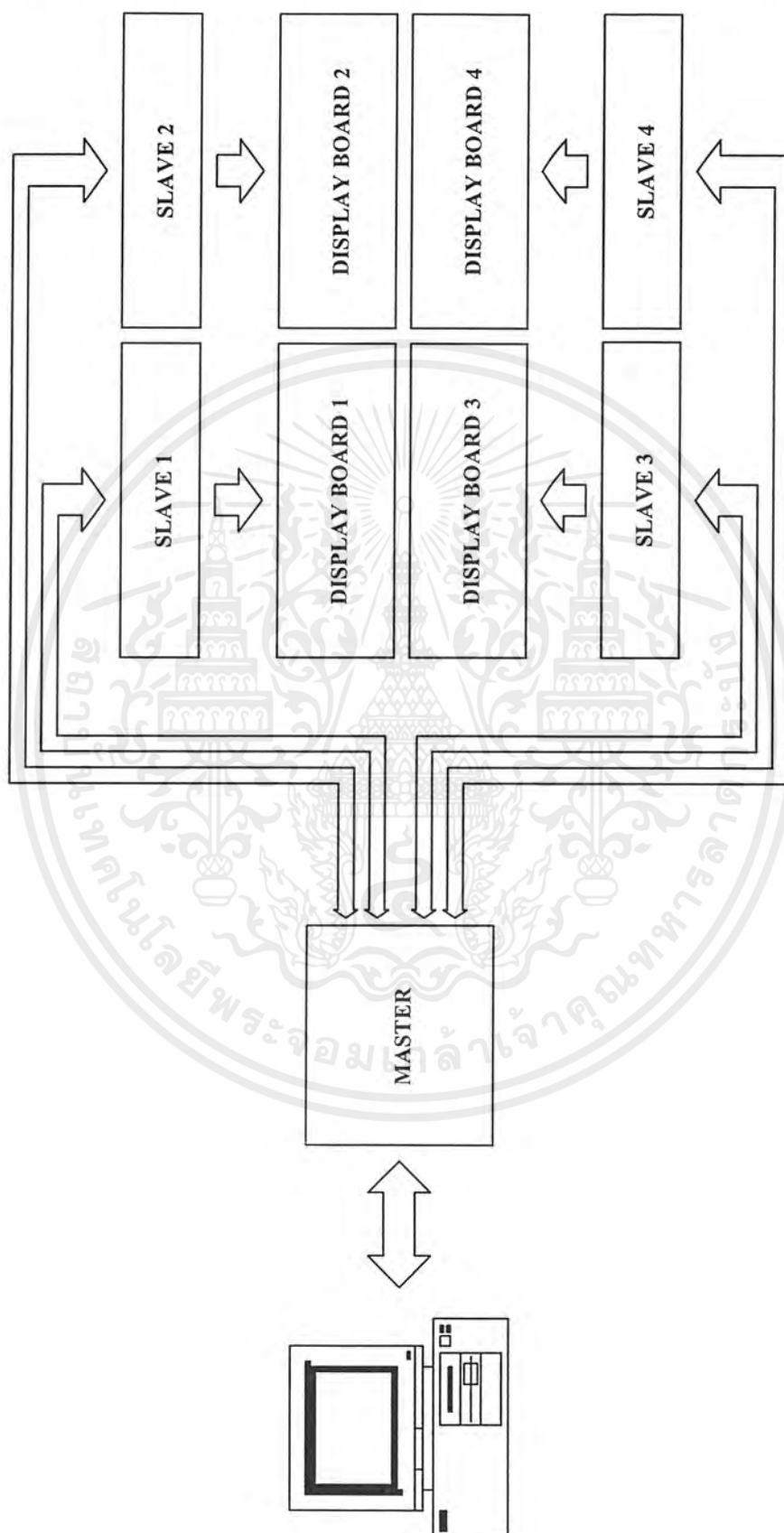
เนื่องจากบอร์ดแสดงผลมีขนาด 64×128 จุด ซึ่งเป็นขนาดใหญ่ทำให้เป็นการยากในการที่จะสร้างเป็นบอร์ดเดี่ยว ทางคณะผู้จัดทำจึงทำการออกแบบแยกเป็นบอร์ดย่อยๆ จำนวน 4 บอร์ด แต่ละบอร์ดมีขนาด 32×64 จุด เพื่อความสะดวกในการออกแบบและการสร้าง โดยการควบคุมการแสดงผลจะมีชุดควบคุมทั้งหมด 5 ชุด แบ่งเป็นมาสเตอร์ 1 ชุดและสเลฟ 4 ชุด โดยชุดควบคุมตัวสเลฟ 1 ชุด จะควบคุมบอร์ดแสดงผล 1 บอร์ด มีหน้าที่การทำงานโดยรวมคือรับข้อมูลที่ส่งมาจากชุดควบคุมตัวมาสเตอร์และนำข้อมูลนั้นออกมาแสดงผล บอร์ดแสดงผลจะแยกส่วนของแอลอีดีและส่วนชุดวงจรควบคุมออกจากกัน และใช้สายสัญญาณในการส่งข้อมูลจากชุดวงจรควบคุมตัวสเลฟไปยังบอร์ดแสดงผล ชุดควบคุมตัวมาสเตอร์จะทำหน้าที่ประมวลผลข้อมูลที่ส่งมาจากคอมพิวเตอร์แล้วทำการส่งข้อมูลในรูปแบบต่างๆ ตามข้อมูลที่ได้รับไปยังชุดควบคุมตัวสเลฟทั้ง 4 ชุด ในชุดควบคุมทั้ง 5 ชุดจะใช้ไมโครคอนโทรลเลอร์ตระกูล MCS-51 เป็นตัวประมวลผลโดยเลือกใช้เบอร์ AT89C52 ไมโครคอนโทรลเลอร์ทุกตัวจะทำงานที่ความถี่ 24 MHz ซึ่งจะทำให้สามารถประมวลผลได้รวดเร็ว บล็อกไดอะแกรมของบอร์ดแสดงผลสามสีนี้แสดงดังในรูปที่ 3.1

3.3 โครงสร้างและการทำงานของชุดควบคุมตัวสเลฟ

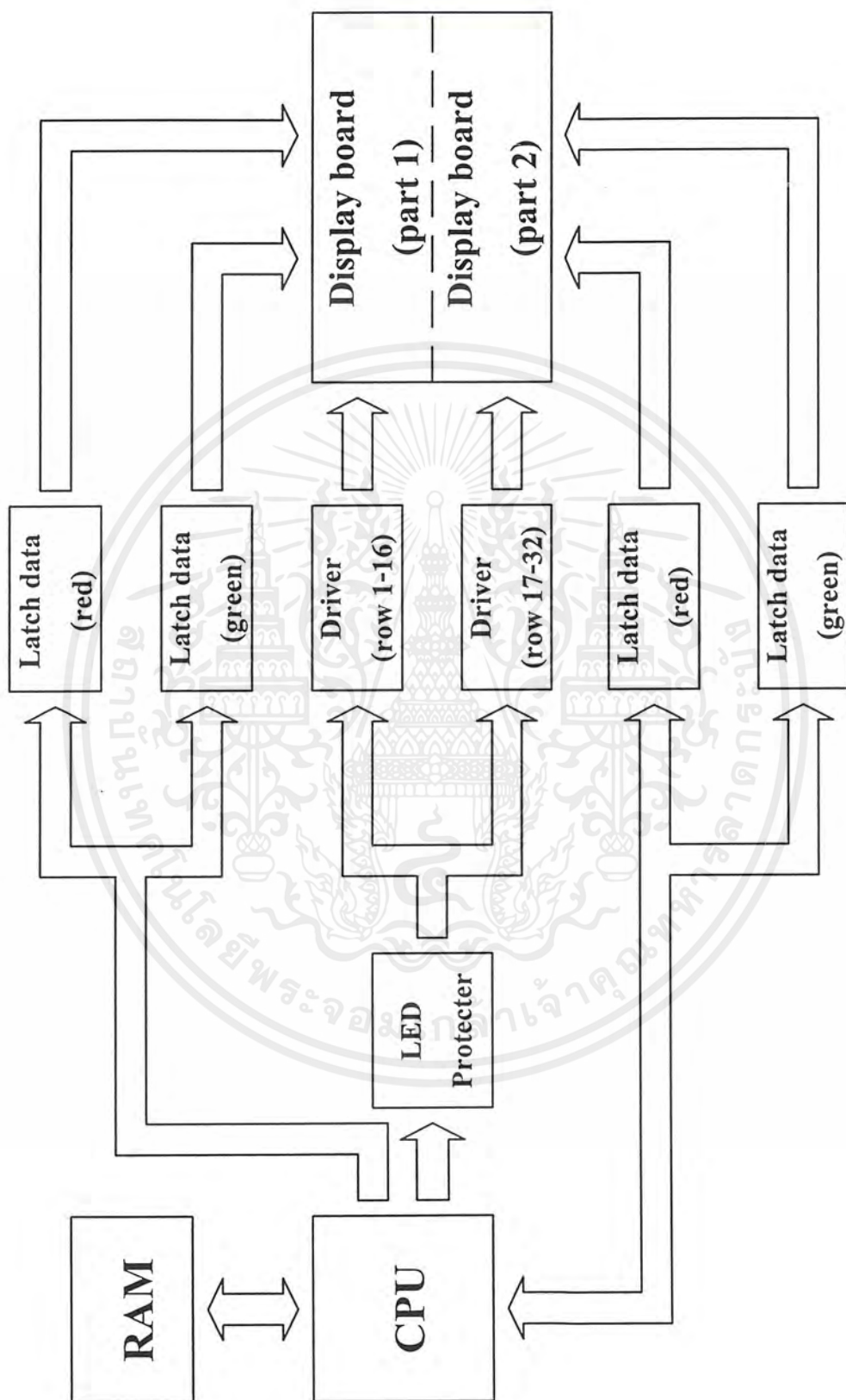
3.3.1 วงจรแลตซ์

เนื่องจากในแต่ละบอร์ดมีจำนวนแถวของแอลอีดีทั้งหมด 32 แถวดังนั้นเพื่อให้แอลอีดีมีช่วงเวลาการติดสว่างนานขึ้นและเป็นการลดเวลาที่ใช้ในการสแกนทางด้านแถวลงจึงได้มีการออกแบบการแลตซ์ข้อมูลโดยชุดไอซีแลตซ์จะแยกเป็น 2 ชุด คือชุดด้านบน (16แถวบน) และชุดด้านล่าง (16แถวล่าง) ซึ่งแต่ละตัวจะทำหน้าที่รับข้อมูลจากบัสข้อมูลภายใต้สัญญาณควบคุมการเลือกวงจรแลตซ์ ข้อมูลที่เชื่อมต่อกับวงจรส่วนนี้จะป้อนผ่านบัฟเฟอร์มาก่อน การที่ไอซีแลตซ์จะทำการแลตซ์ข้อมูลที่ได้รับมาจากบัสข้อ

มูลค่าที่ขา LE (Latch Enable) ต้องได้รับสัญญาณที่มีการเปลี่ยนสถานะจาก Low เป็น High โครงสร้างภายในของไอซีเบอร์ 74LS374 จะมีลักษณะเป็น ดี-ฟลิปฟล็อป (D Flip-Flop)

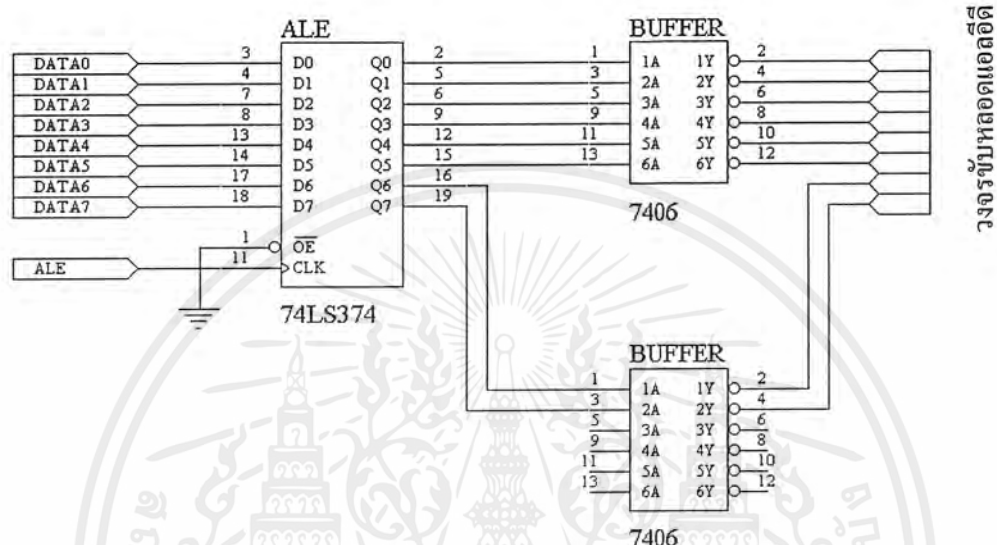


รูปที่ 3.1 แสดงบล็อก ไดอะแกรมของบอร์ดแสดงผลแบบไดนามิกสี่บิต



รูปที่ 3.2 แสดงบล็อกไดอะแกรมของชุดควบคุมตัวเลข

การทำงานของวงจรสามารถอธิบายได้ตาม Timing Diagram ดังในรูปที่ 3.5 ในวงจรจะมีการต่อขา OE ลงกราวด์เพื่อทำให้มีสถานะ Low ขณะที่ข้อมูลเข้ามาทางอินพุตและถ้าขา LE สัญญาณมีการเปลี่ยนสถานะจาก Low เป็น High ณ เวลาดังกล่าวข้อมูลที่เอาต์พุตจะมีสถานะเดียวกันกับข้อมูลที่อินพุต และจะค้างสถานะนี้ตลอดไปแม้ว่าข้อมูลที่อินพุตจะมีการเปลี่ยนแปลงแล้วก็ตาม ข้อมูลเอาต์พุตจะมีการเปลี่ยนตามข้อมูลอินพุตอีกครั้งเมื่อสัญญาณที่ขา LE มีการเปลี่ยนสถานะจาก Low เป็น High

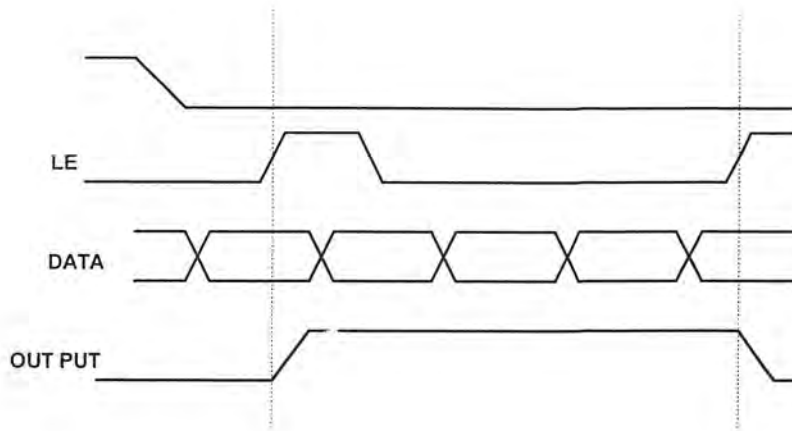


รูปที่ 3.3 แสดงวงจรแลตซ์

เอาที่พูดที่มีการเลตซ์ข้อมูล



รูปที่ 3.4 แสดงลักษณะการแลตซ์ข้อมูล

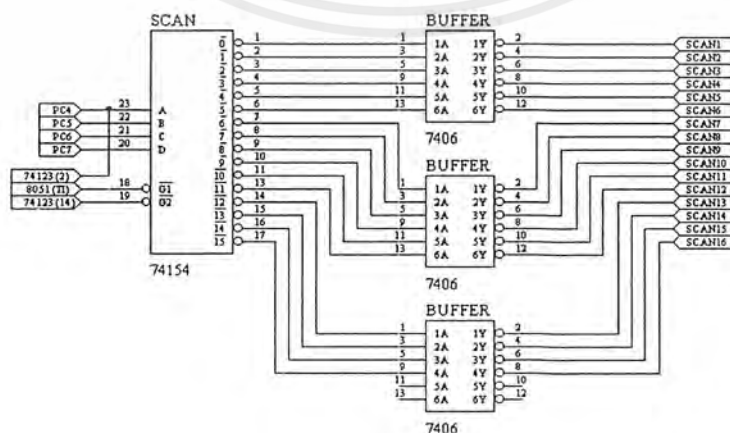


รูปที่ 3.5 แสดง Timing Diagram ของวงจรแสดง

3.3.2 วงจรควบคุมการสแกนทางด้านแนว

การสแกนทางด้านแนวของแต่ละบอร์ดนั้นจะทำการส่งข้อมูลในแถวที่ 1 และ 17 ของทุกๆหลักจนครบแล้วจะควบคุมให้แอลทีฟพร้อมกันโดยการเช็ตสัญญาณที่พอร์ท P3.5 ซึ่งต่อเข้ากับขา $\overline{G1}$ (18) ของไอซี 74LS154 ต่อจากนั้นก็ส่งข้อมูลชุดต่อไปคือข้อมูลของแถวที่ 2 และ 18 ออกไปทุกหลักแล้วให้แอลทีฟพร้อมกัน ทำเช่นนี้ไปเรื่อยๆ จนข้อมูลทั้ง 32 แถวถูกส่งออกไปก็จะถือเป็นการสแกนครบหนึ่งรอบ โดยจะกำหนดช่วงเวลาการแอลทีฟของแอลอีดีในแต่ละแถวโดยใช้อินเตอร์รัปต์ไทม์เมอร์แฟล็ก (TFO, TF1) เพื่อให้แอลอีดีทุกหลอดมีระยะเวลาการแอลทีฟเท่ากันทั้ง 16 แถว

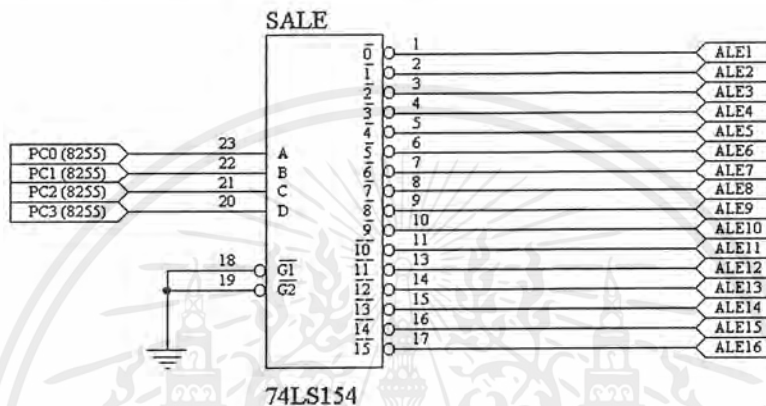
ในส่วนของสัญญาณที่ใช้ควบคุมการสแกนนั้น จะใช้สัญญาณจาก 4 บิตบนของพอร์ท C ของไอซี 8255 ป้อนเข้าทางด้านอินพุตของไอซีเบอร์ 74LS154 (4-Line-to 16-Line Decoder/Demultiplex) ซึ่งจะทำหน้าที่ถอดรหัสในการเลือกแถวที่จะทำการสแกนและด้วยเหตุที่ไอซีเบอร์นี้จะให้สถานะทางเอาต์พุตเป็น Low จึงจำเป็นต้องใช้ไอซีเบอร์ 74LS06 ซึ่งทำหน้าที่เป็นอินเวอร์เตอร์มาต่อเข้าซึ่งจะทำให้สัญญาณเปลี่ยนสถานะมาเป็น High เพื่อใช้เป็นแรงดันไบอัสให้กับทรานซิสเตอร์ชนิด เอ็นพีเอ็น ที่อยู่ในส่วนของวงจรขับ



รูปที่ 3.6 แสดงการต่อวงจรที่ใช้ควบคุมการสแกนทางด้านแนว

3.3.3 วงจรควบคุมการแลตซ์ข้อมูล

การทำงานจะเป็นไปในลักษณะของการส่งสัญญาณไปควบคุมให้ข้อมูลที่ส่งมาจากชุดมาสเตอร์อยู่ในหลักที่ถูกต้อง โดยจะใช้สัญญาณจาก 4 บิตล่างของพอร์ท C ของไอซีเบอร์ 8255 เป็นอินพุตให้กับไอซีเบอร์ 74LS154 เนื่องจากว่าเอาต์พุตทั้ง 16 ขาของไอซีเบอร์ 74LS154 ถูกต่อเข้ากับขา ALE ของไอซีแลตซ์ทุกตัวในวงจรแลตซ์ วงจรควบคุมนี้จึงเป็นตัวเลือกตำแหน่งของหลักที่ต้องการแลตซ์ข้อมูลที่ส่งมา ดังนั้นในการส่งสัญญาณเลือกตำแหน่งการแลตซ์ข้อมูลในแต่ละหลักซึ่งควบคุมโดยโปรแกรมจะต้องทำงานสัมพันธ์กับข้อมูลที่ส่งมาทางบัสข้อมูลว่าควรจะเป็นข้อมูลของหลักใด



รูปที่ 3.7 แสดงการต่อวงจรควบคุมการแลตซ์ข้อมูล

3.3.4 วงจรป้องกันแอลอีดีเสียหาย

ทำหน้าที่ป้องกันไม่ให้แอลอีดีติดค้างนานเกินไปเพราะอาจเกิดความเสียหายกับหลอดได้ โดยใช้ไอซีเบอร์ 74LS123 (Retriggerable Monostable Multivibrator) มาต่อเป็นวงจรโมโนสเตเบิล สัญญาณที่ใช้ในการทริกที่ขา B(2) จะต่อมาจากบิตสุดท้ายของอินพุตที่ใช้ควบคุมการแลตซ์ข้อมูลของไอซีเบอร์ 74LS154 โดยสัญญาณเอาต์พุตของวงจรป้องกันจะต่อเข้ากับขาที่ $\bar{Q}$ (19) ของไอซีเบอร์ 74LS154

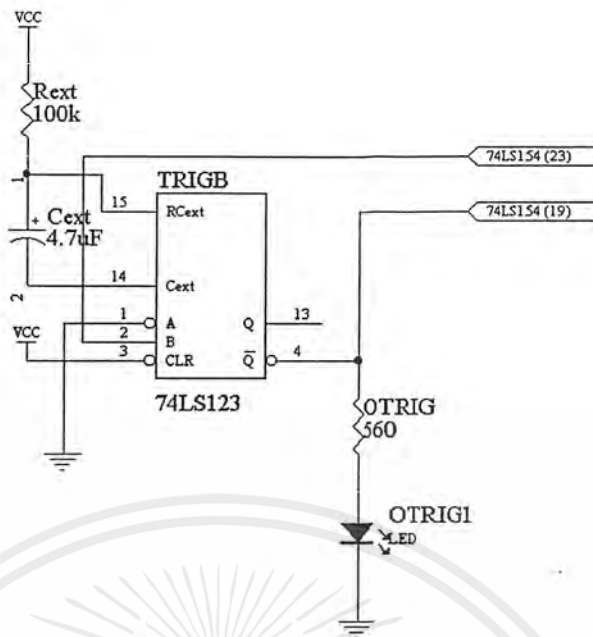
ในการสแกนแต่ละแถว สัญญาณที่เป็นอินพุตของไอซีเบอร์ 74LS154 ที่อยู่ในส่วนของวงจรควบคุมการสแกนด้านแถวจะมีการเปลี่ยนแปลงสถานะไปตามลำดับของแถวที่จะสแกน ถ้าหากการสแกนมีการทำงานผิดพลาดเกิดขึ้นหรือซีพียูมีอาการแฮงค์ เป็นเหตุให้ไอซีเบอร์ 74LS123 ไม่ได้ถูกทริกภายในช่วงเวลาที่กำหนดโดยค่า $R_{ext} C_{ext} (t_w)$ จะทำให้เอาต์พุต $\bar{Q}$ (4) มีสถานะเป็น High และที่ขา $\bar{G}2$ (19) ของไอซีเบอร์ 74LS154 ก็จะมีสถานะเป็น High ด้วย มีผลทำให้เอาต์พุตทั้ง 16 ขาของไอซีเบอร์นี้มีสถานะเป็น High ทรานซิสเตอร์เบอร์ 2SC1061 ที่ต่ออยู่ในวงจรจับทุกตัวจะ “OFF”

การคำนวณหาค่าเวลา t_w สามารถหาได้ดังนี้

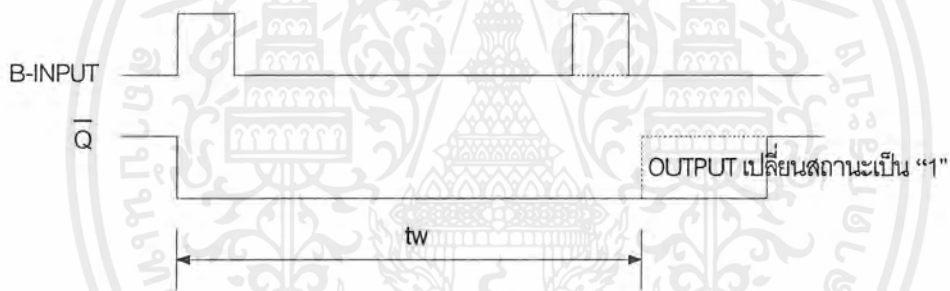
$$t_w = R_{ext} \times C_{ext} \times K$$

ค่าของ K ที่เหมาะสม = 0.25

$$\begin{aligned} t_w &= 100 \times 10^3 \times 4.7 \times 10^{-6} \times 0.25 \\ &= 0.1175 \text{ วินาที} \end{aligned}$$



รูปที่ 3.8 แสดงวงจรป้องกันแอลอีดีเสียหาย



รูปที่ 3.9 แสดง Timing Diagramme ของวงจรป้องกันแอลอีดีเสียหาย

3.3.5 วงจรขับแอลอีดี (Drive Circuit LED Board)

เนื่องจากได้แบ่งส่วนของบอร์ดแสดงผลออกเป็น 4 ส่วน โดยแต่ละส่วนมีขนาดจำนวนจุดทางด้านแถวและด้านหลักเท่ากับ 64 x 32 จุด ดังนั้นจะอธิบายการทำงานเพียงส่วนเดียวเท่านั้น

ในบอร์ดแสดงผลแอลอีดีจะต้องมีการควบคุมการจ่ายกระแสไฟฟ้าให้แต่ละหลอดสว่างตามรูปแบบตัวอักษรหรือรูปภาพต่างๆที่ต้องการ ดังนั้นจึงจำเป็นต้องมีส่วน Driver เพื่อทำหน้าที่ขยายกระแสและเป็นเหมือนสวิตช์ปิด/เปิด ซึ่งในโครงงานนี้ใช้ทรานซิสเตอร์ทำหน้าที่เป็นตัวควบคุมในการจ่ายกระแสไฟฟ้าให้หลอดแอลอีดีแต่ละหลอด ส่วนวงจรตัวขับจะแบ่งออกเป็น 2 ส่วน ได้แก่

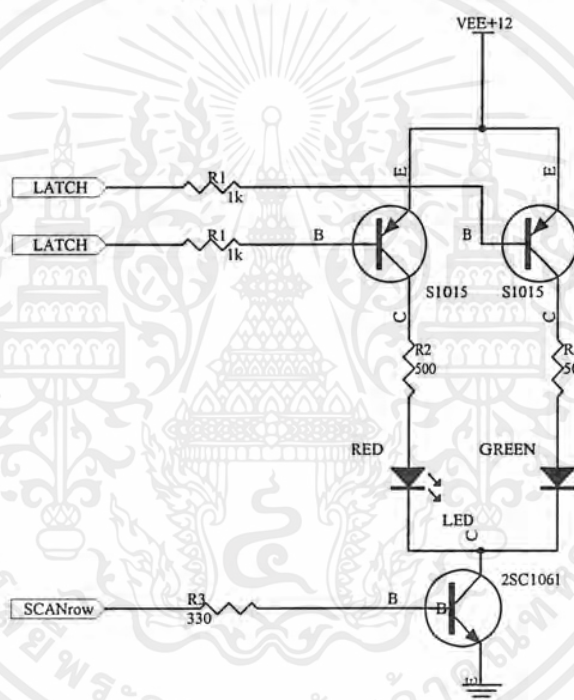
1. วงจรตัวขับด้านหลัก (Column Drive Circuit)

วงจรตัวขับด้านหลักจะประกอบด้วยส่วนขับกระแสให้แอลอีดีสีแดงและส่วนขับกระแสให้แอลอีดีสีเขียว ทรานซิสเตอร์ที่ใช้ขับได้แก่เบอร์ 2SA1015 ซึ่งเป็นชนิด พีเอ็นพี ความต้านทาน R1 และ R2 โดยมีการจัดไบอัสให้กับทรานซิสเตอร์เป็นแบบคอมมอนอิมิเตอร์ (Common Emitter) ความต้านทาน R2 จะ

ทำหน้าที่จำกัดกระแสที่จะไหลเข้าสู่หลอดแอลอีดีแต่ละหลอดในหลักนั้นๆ ความต้านทาน $R1$ จะเป็นตัวจำกัดกระแสไบอัสที่จะไหลเข้าที่ขาเบสของทรานซิสเตอร์ เมื่ออินพุตของไอซีเบอร์ 7406 ที่ต่อกับวงจรแลตช์มีลอจิกเป็น 1 ทรานซิสเตอร์ที่ต่อกับเอาต์พุตตรงกับตำแหน่งนั้นจะทำงาน (on) แต่ว่าแอลอีดีจะติดหรือไม่มันต้องขึ้นอยู่กับวงจรขับเคลื่อนแถวด้วย ซึ่งจะได้อธิบายในต่อไป

2. วงจรตัวขับเคลื่อนแถว (Row Drive Circuit)

วงจรขับเคลื่อนแถวจะประกอบด้วยทรานซิสเตอร์เบอร์ 2SC1061 เป็นชนิดเอ็นพีเอ็น มีการต่อแบบคอมมอนอีมิเตอร์ (Common Emitter) และความต้านทาน $R3$ ทำหน้าที่ในการควบคุมกระแสไบอัสที่จะไหลเข้าทางขาเบสของทรานซิสเตอร์ ทรานซิสเตอร์ตัวนี้จะทำหน้าที่เหมือนสวิตช์คอยควบคุมกระแสที่จะไหลลงกราวด์ โดยขาคอลเลกเตอร์ถูกต่อออกจากขาคอมมอนของหลอดแอลอีดี การทำงานของทรานซิสเตอร์จะขึ้นอยู่กับสถานะของสัญญาณที่ส่งมาจากวงจรควบคุมการสแกนด้านแถว



รูปที่ 3.10 แสดงวงจรขับเคลื่อนแอลอีดี

การเชื่อมต่อระหว่างส่วนของวงจรขับเคลื่อนกับส่วนของ Dot Matrix Display Board จะใช้สายแพรกับคอนเน็คเตอร์เป็นตัวเชื่อมโยงสัญญาณทั้งนี้เพื่อความยืดหยุ่น สะดวกต่อการแยกส่วนและทำให้ขนาดของบอร์ดเล็กลงอีกทั้งยังสามารถทำการตรวจสอบข้อผิดพลาดต่างๆ ที่เกิดขึ้นกับแต่ละบอร์ดได้ง่าย

การคำนวณหาค่าความต้านทานต่างๆ ในวงจรได้ออกแบบโดยกำหนดให้ขนาดของกระแสที่จะไหลผ่านแอลอีดีแต่ละหลอดมีค่าเท่ากับ 20 มิลลิแอมป์ ทรานซิสเตอร์เบอร์ 2SA1015 จะมีกระแสคอลเลกเตอร์เท่ากับ 320 มิลลิแอมป์ (20 มิลลิแอมป์ x 16 แถว) มีอัตราขยายกระแส (β) 100 ทรานซิสเตอร์เบอร์ 2SC1061 มีอัตราขยายกระแส (β) 60

การคำนวณหาค่าความต้านทาน R_1

$$\begin{aligned} I_B &= \frac{I_C}{\beta} \\ &= \frac{320mA}{100} \\ &= 3.2 \text{ มิลลิแอมป์} \end{aligned}$$

$$V_{EE} - V_{EB} - V_B - I_B R_1 = 0$$

$$R_1 = \frac{V_{EE} - V_{BE} - V_B}{I_B}$$

$$\begin{aligned} \text{แทนค่า } R_1 &= \frac{12 - 0.7 - 0.1}{3.2 \times 10^{-3}} \\ &= \frac{11.2}{3.2 \times 10^{-3}} \\ &= 3500 \, \Omega \end{aligned}$$

การคำนวณหาค่าความต้านทาน R_2

$$V_{EE} - V_{EC1} - I_C R_2 - V_D - V_{CE2} = 0$$

$$R_2 = \frac{V_{EE} - V_{EC1} - V_{CE2} - V_D}{I_C}$$

$$\begin{aligned} \text{แทนค่า } R_2 &= \frac{12 - 0.1 - 0.1 - 2.4}{20 \times 10^{-3}} \\ &= \frac{9.4}{20 \times 10^{-3}} \\ &= 470 \, \Omega \end{aligned}$$

การคำนวณหาค่าความต้านทาน R_3

เนื่องจากในเวลาแสดงผลจริงหลอดแอลอีดีจะไม่สว่างหมดทุกหลอด จึงสมมุติให้มีหลอดที่สว่างประมาณ 40 หลอด ทรานซิสเตอร์เบอร์ 2SC1601 จะมีกระแสคอลเลกเตอร์เท่ากับ 800 มิลลิแอมป์

$$\begin{aligned} I_B &= \frac{800mA}{60} \\ &= 13.33 \text{ มิลลิแอมป์} \end{aligned}$$

$$V_B - V_{BE} - I_B R_3 = 0$$

$$R_3 = \frac{V_B - V_{BE}}{I_B}$$

$$= \frac{5 - 0.7}{13.33 \times 10^{-3}}$$

$$= \frac{4.3}{13.33 \times 10^{-3}}$$

$$= 322.6 \, \Omega$$

ในวงจรจริงเนื่องจากแอลอีดีไม่สว่างเท่าที่ควรจึงปรับเปลี่ยนค่าความต้านทานต่างๆเป็น

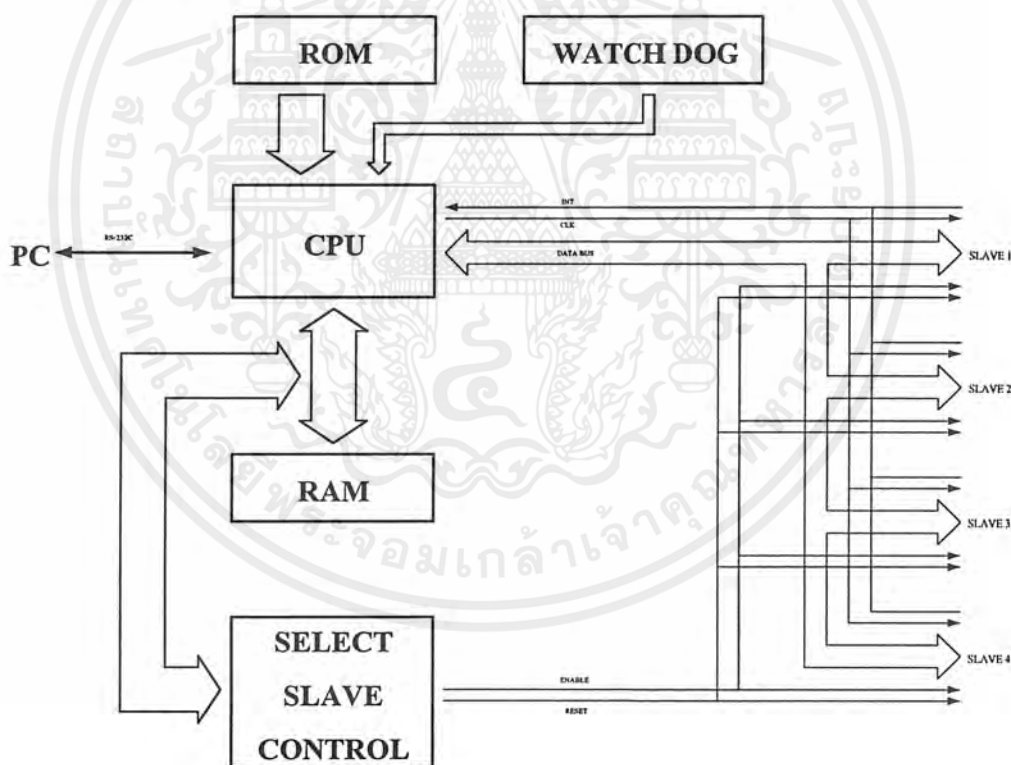
$$R1 = 1 \, k\Omega$$

$$R2 = 500 \, \Omega$$

$$R3 = 330 \, \Omega$$

3.4 โครงสร้างและการทำงานของชุดควบคุมตัวมาสเตอร์

การทำงานของชุดควบคุมตัวมาสเตอร์จะใช้ไมโครคอนโทรลเลอร์ตระกูล MCS-51 เบอร์ AT89C52 ในการควบคุมการทำงานทั้งหมด บล็อกไดอะแกรมของชุดควบคุมตัวมาสเตอร์แสดงดังในรูปที่ 3.11



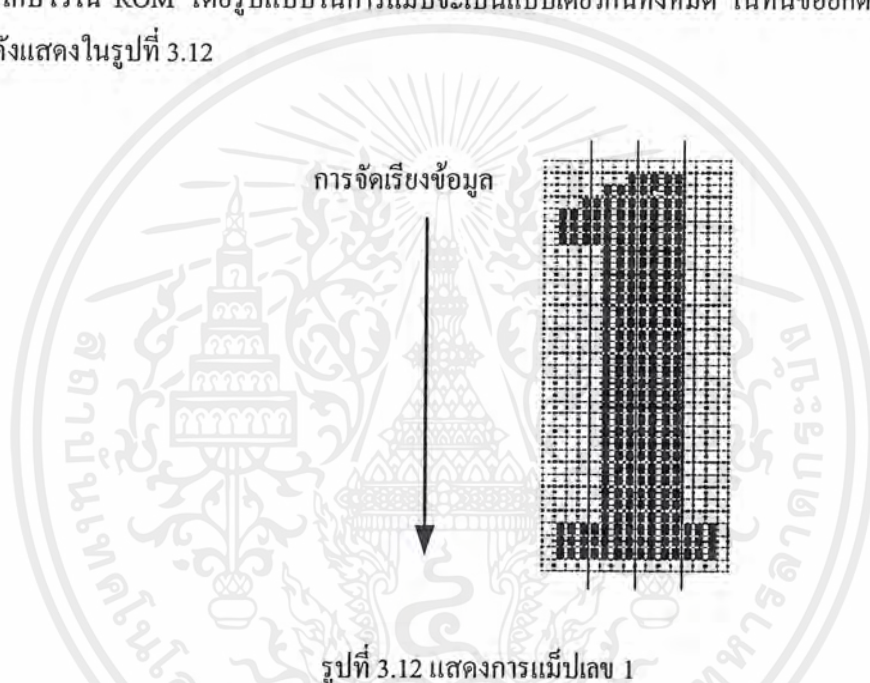
รูปที่ 3.11 แสดง บล็อกไดอะแกรมของชุดควบคุมตัวมาสเตอร์

ข้อมูล อักษรทั้งไทยและอังกฤษ ตัวเลขและรูปภาพต่างๆ ที่ออกแบบขึ้นมาเพื่อใช้ในการแสดงผล จะถูกเก็บไว้ใน ROM เมื่อซีพียู ได้รับข้อมูลจากคอมพิวเตอร์ ซึ่งมีทั้งรหัสแอสกีของตัวอักษรและโค้ดต่างๆที่กำหนดขึ้นเพื่อใช้ในการเลือกรูปแบบการแสดงผล ก็จะทำการประมวลผลโดยจะโหลดข้อมูลใน

ROM ที่ตรงกับรหัสแอสกีออกมา เช่น ถ้าได้รับรหัสแอสกี 41 ก็จะโหลดข้อมูลตัวอักษร A ใน ROM ออกมา จากนั้นก็จะส่งข้อมูลต่างๆ ออกไปยังชุดควบคุมตัวสเลฟ ณ เวลาใดเวลาหนึ่งจะมีเพียงชุดควบคุมตัวสเลฟเพียงตัวเดียวที่สามารถรับข้อมูลได้ โดยจะมีบล็อก SELECT SLAVE CONTROL เป็นตัวควบคุมการเลือกชุดควบคุมตัวสเลฟ ในส่วนของวงจรวอตช์ด็อก (WATCH DOG) จะทำหน้าที่คอยเฝ้าดูการทำงานของชิพ ถ้าชิพทำงานผิดพลาดหรือติดอยู่ในลูปใดลูปหนึ่งของโปรแกรม วงจรวอตช์ด็อก ก็จะทำให้การส่งสัญญาณมารีเซ็ตชิพให้เริ่มต้นทำงานใหม่

3.5 การแมปข้อมูลตัวอักษร ตัวเลข และรูปภาพต่างๆ

รูปแบบลักษณะของตัวอักษร ตัวเลข รวมทั้งรูปภาพต่างๆ จะใช้การแมปเป็นเลขฐานสิบหก เพื่อทำการเก็บไว้ใน ROM โดยรูปแบบในการแมปจะเป็นแบบเดียวกันทั้งหมด ในที่นี้ขอยกตัวอย่างการแมปเลข 1 ดังแสดงในรูปที่ 3.12



จากตัวอย่างในรูปที่ 3.12 ซึ่งเป็นเลข 1 จะกำหนดให้กรณีทีหลดแอลอีดีสว่างแทนด้วยสถานะ “1” ส่วนกรณีทีหลดแอลอีดีดับจะแทนด้วยสถานะ “0” โดยจะให้ทางขวามือเป็น MSB และทางด้านซ้ายมือเป็น LSB เมื่อทำการแมปแล้วจะทำการเก็บข้อมูลลงใน ROM ลักษณะการเรียงข้อมูลจะเรียงจากบนลงล่างดังนี้

```
DB 80H,0E0H,0F8H,0FEH,0FEH,0FEH,0E0H,0E0H,0E0H,0E0H
DB 0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH
DB 0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0E0H
DB 0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0E0H,0FEH,0FEH,0FEH
DB 0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH
DB 0FH,0FH,0FH,0FH,0FH,0FH,0FH,0FH,7FH,7FH,7FH
```

3.6 โปรแกรมสำเร็จรูปที่ใช้ในการควบคุมการแสดงผล

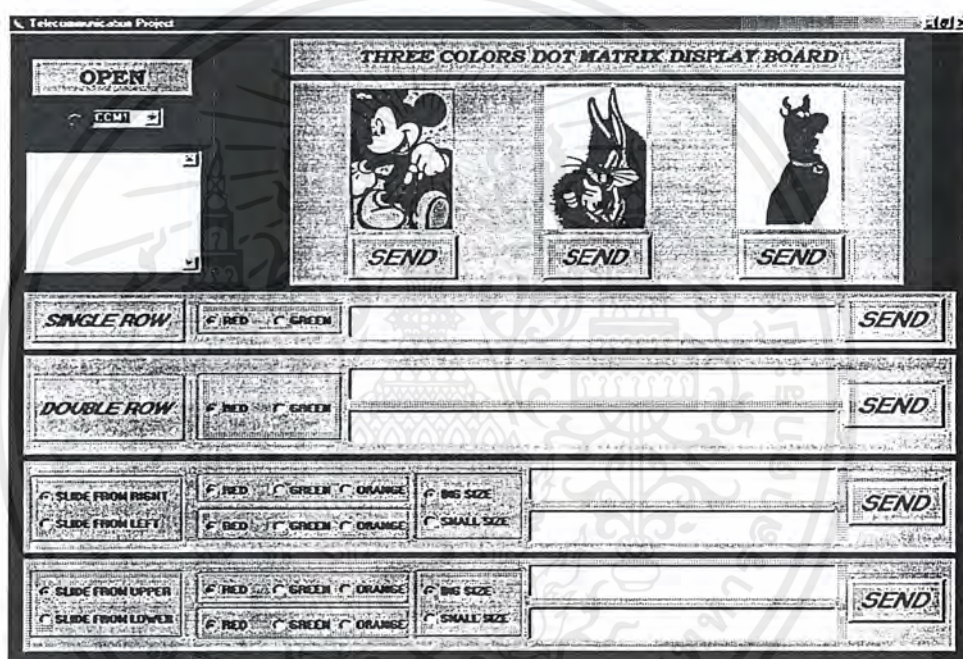
การส่งข้อมูลจากคอมพิวเตอร์ไปยังชุดควบคุมตัวมาสเตอร์จะใช้การส่งแบบอะซิงโครนัส บอกระยะเท่ากับ 9600 bps โดยโปรแกรมสำเร็จรูปนี้จะใช้ Delphi ในการพัฒนา โดยเน้นให้ใช้งานง่าย ซึ่งในหน้าต่างของโปรแกรมจะประกอบไปด้วยสามส่วนย่อยคือ

ส่วนที่หนึ่ง ใช้ในการควบคุมการติดต่อระหว่างโปรแกรมกับพอร์ตอนุกรมของคอมพิวเตอร์

ส่วนที่สอง ใช้ในการควบคุมการแสดงผลแบบรูปภาพนิ่ง และภาพเคลื่อนไหว

ส่วนที่สาม ใช้ในการควบคุมการแสดงผลแบบตัวอักษร

ในส่วนที่สามจะมีช่องว่างสำหรับรับข้อมูลจากคีย์บอร์ด ฟังก์ชันในการเลือกสีของตัวอักษรและปุ่มต่างๆที่ใช้ในการควบคุม หน้าต่างของโปรแกรมแสดงดังในรูปที่ 3.13



รูปที่ 3.13 แสดงหน้าต่างของโปรแกรมสำเร็จรูป

การใช้งานโปรแกรม

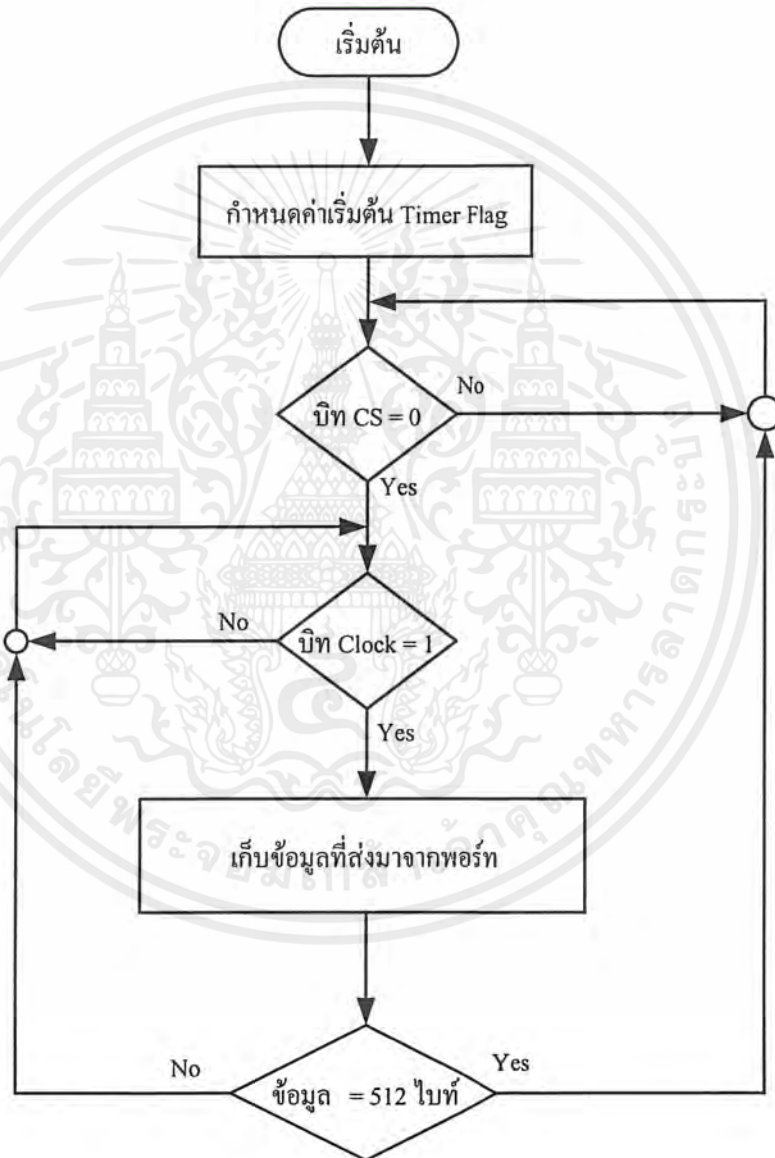
- 1.ทำการเปิดโปรแกรมขึ้นมา จะปรากฏหน้าต่างโปรแกรมดังแสดงในรูปที่ 3.13
- 2.เลือกพอร์ตอนุกรมของคอมพิวเตอร์ที่จะ ใช้ในส่งข้อมูล โดยสามารถกำหนดให้เป็นCOM1หรือ COM2 จากนั้นทำการคลิกบนปุ่ม OPEN ถ้าการเชื่อมต่อสมบูรณ์จะปรากฏข้อความ “Connect to com1(or com2).Devices are ready!” และปุ่ม OPEN จะเปลี่ยนเป็น “COLSE”
- 3.เลือกรูปแบบการแสดงผล ถ้าจะส่งรูปภาพก็ทำการส่งได้เลยโดยการคลิกปุ่ม SEND ได้รูปภาพนั้นๆ ถ้าต้องการส่งข้อความให้นำเมาส์ไปคลิกในช่องสำหรับกรอกข้อความแล้วทำการป้อนข้อความที่ต้องการแสดง เลือกสีที่ต้องการ จากนั้นทำการคลิกปุ่ม SEND ข้อความก็จะถูกส่งไปยังฮาร์ดแวร์ เพื่อทำการแสดงผลต่อไป

3.7. โฟลว์ชาร์ตของโปรแกรมต่างๆ ที่ใช้ควบคุมการแสดงผลของบอร์ดแสดงผล

โฟลว์ชาร์ตของโปรแกรมต่างๆ จะแบ่งออกเป็น 2 ส่วน คือ โฟลว์ชาร์ตของโปรแกรมในส่วนของชุดสเลฟและโฟลว์ชาร์ตของโปรแกรมในส่วนของชุดมาสเตอร์

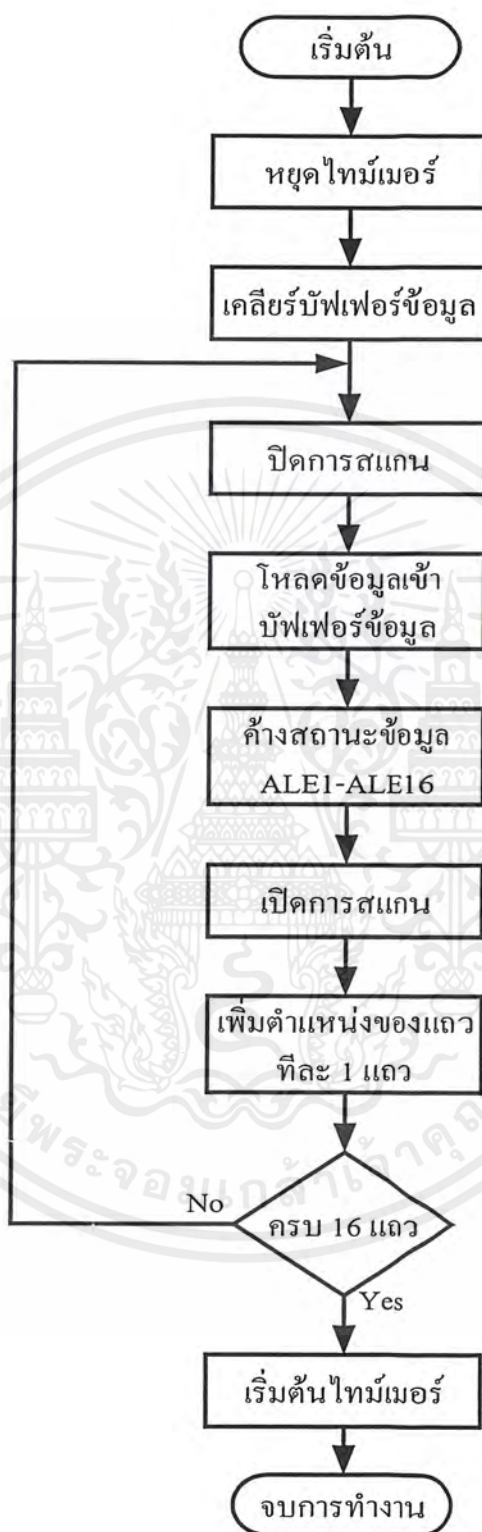
3.7.1 โฟลว์ชาร์ตของโปรแกรมในส่วนของชุดสเลฟ

1. โฟลว์ชาร์ตของโปรแกรมรับข้อมูล จะแสดงขั้นตอนการรับข้อมูลที่ตัวมาสเตอร์ส่งมาให้ในแต่ละบอร์ด



รูปที่ 3.14 แสดงขั้นตอนการรับข้อมูลของตัวอักษรและรูปภาพ

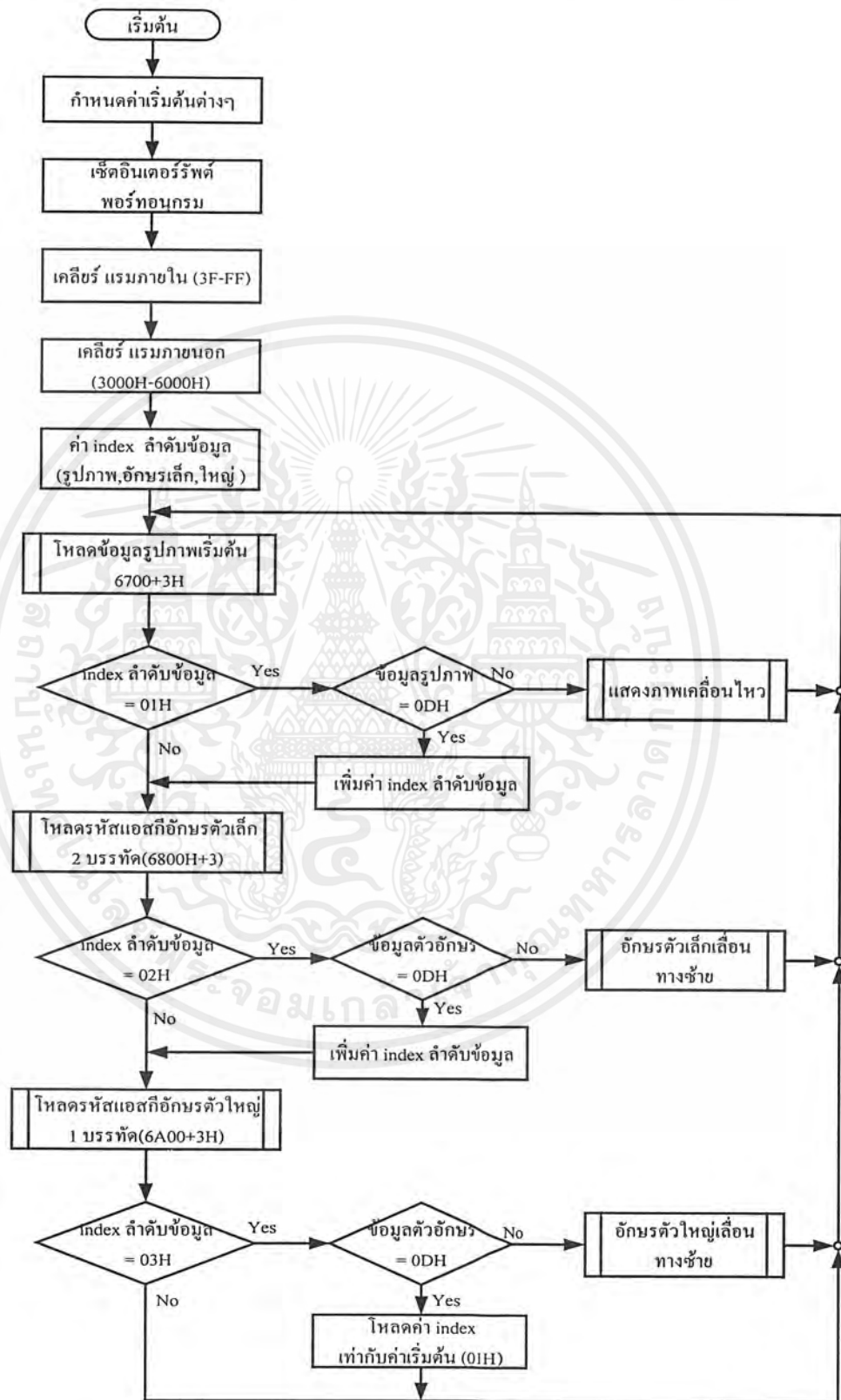
2. โฟลว์ชาร์ตของโปรแกรมอินเตอร์รัฟไทม์เมอร์พร้อมทั้งสแกนข้อมูล



รูปที่ 3.15 แสดงการอินเตอร์รัฟไทม์เมอร์พร้อมทั้งสแกนข้อมูล

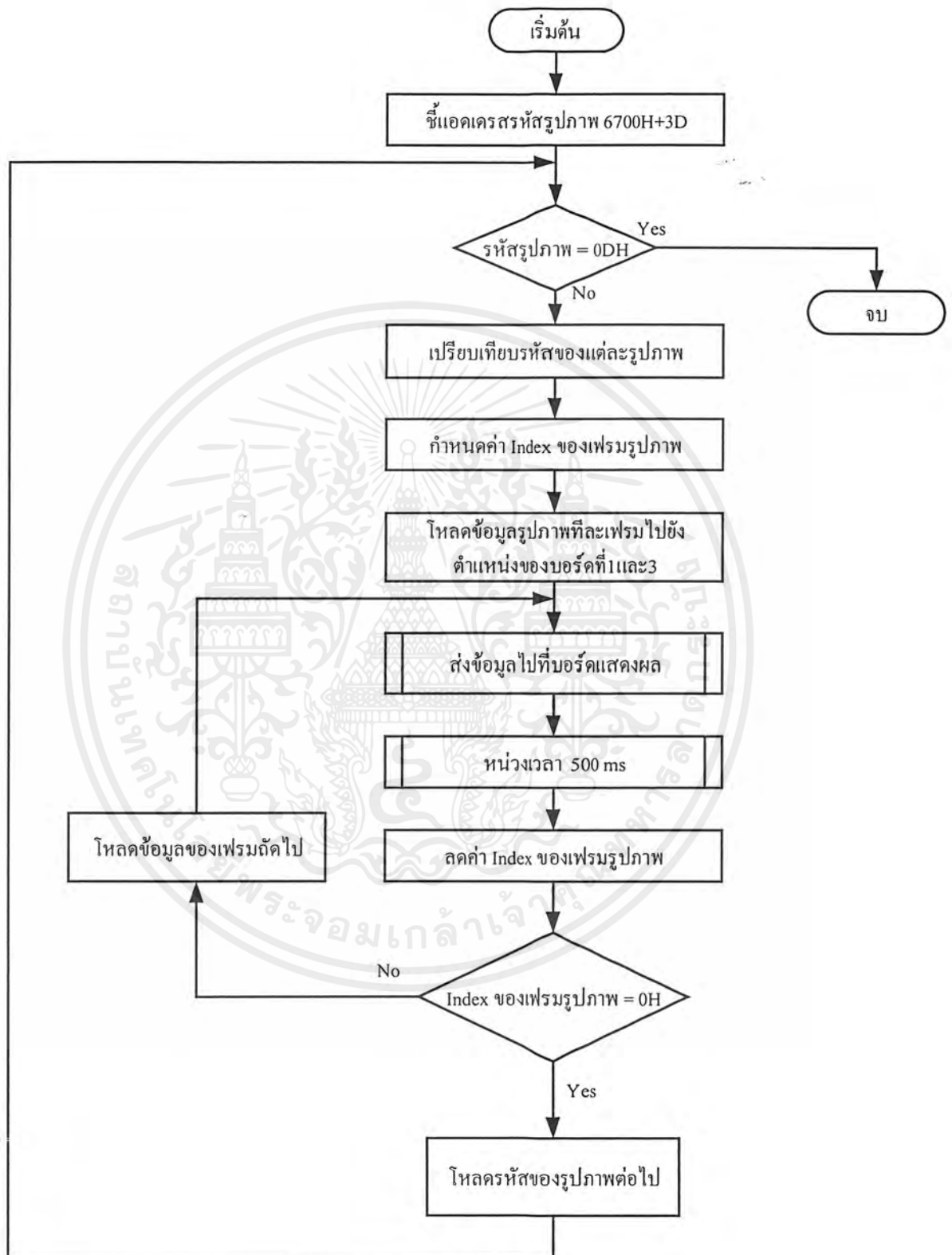
3.7.2 โฟลว์ชาร์ตของโปรแกรมในส่วนของชุดมัสเตอร์

1. โฟลว์ชาร์ตของโปรแกรมหลัก ซึ่งจะแสดงการทำงาน โดยรวมของบอร์ดแสดงผล



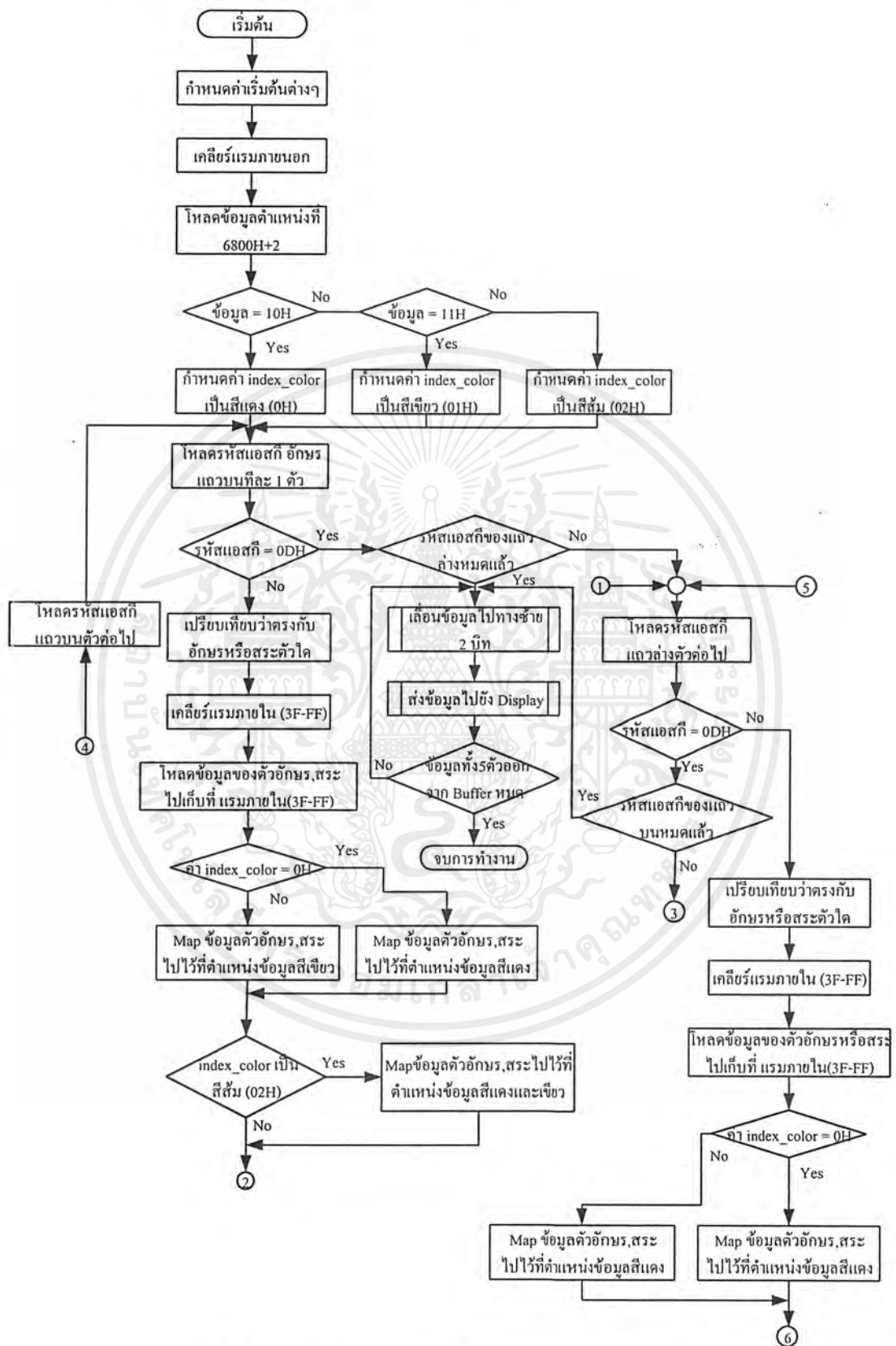
รูปที่ 3.16 แสดงการทำงานของโปรแกรมหลัก

2. โพลีชาร์ตของโปรแกรมแสดงรูปภาพเคลื่อนไหว

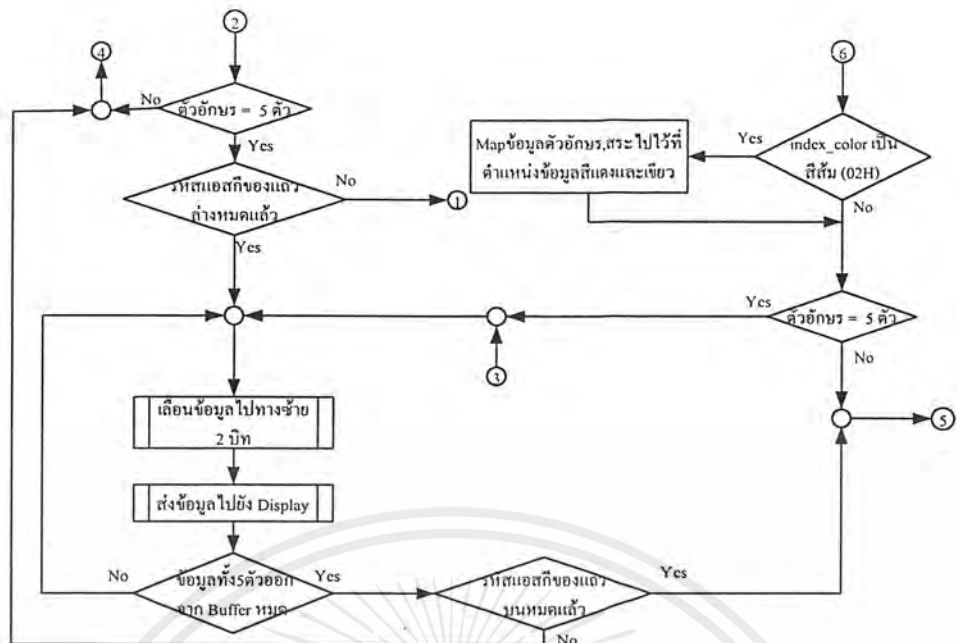


รูปที่ 3.17 แสดงการทำงานของโปรแกรมรูปภาพเคลื่อนไหว

4. โพลีชาร์ตของโปรแกรมแสดงตัวอักษรตัวเล็ก

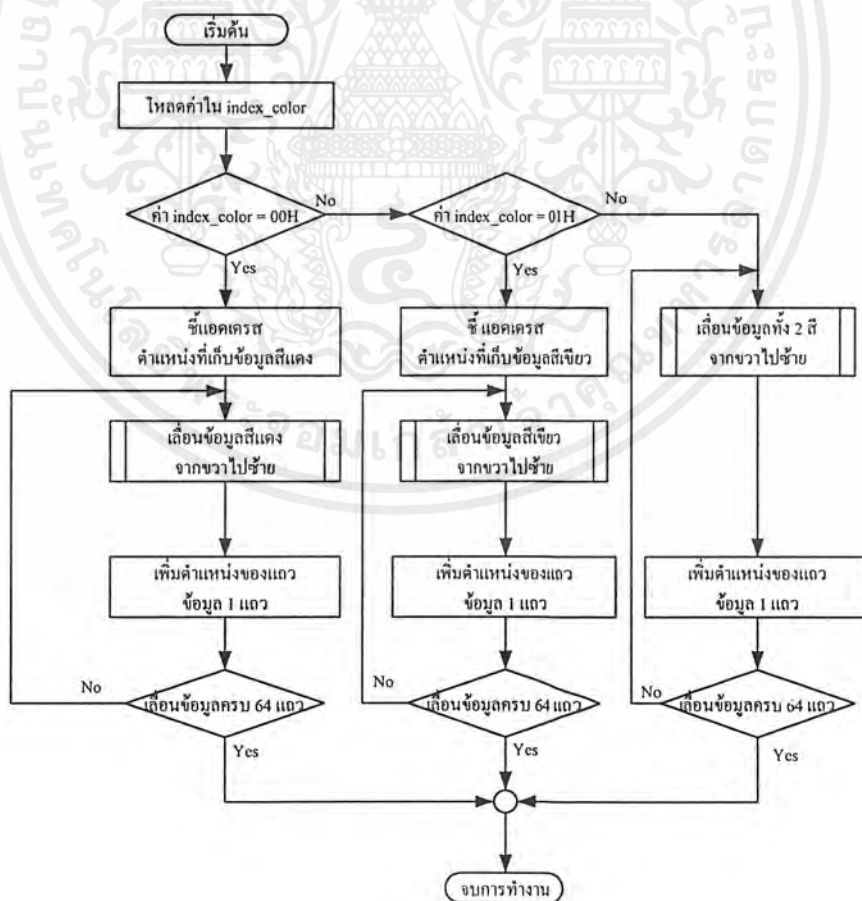


รูปที่ 3.19 แสดงการทำงานของ โปรแกรมแสดงตัวอักษรตัวเล็ก



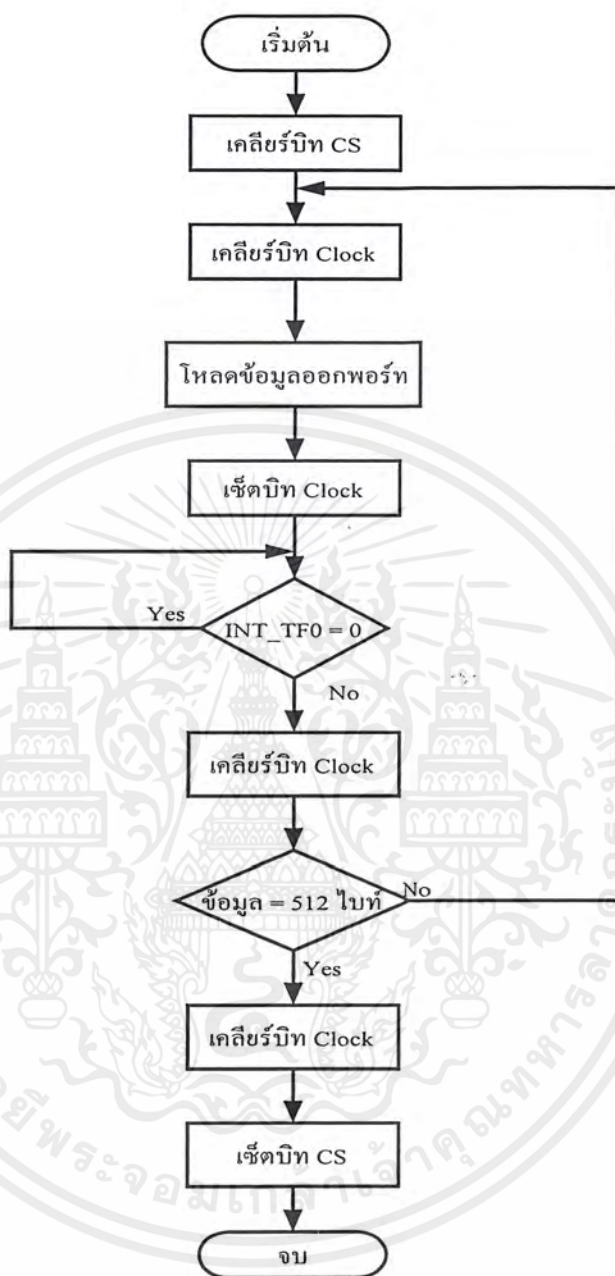
รูปที่ 3.20 แสดงการทำงานของโปรแกรมแสดงตัวอักษรตัวเล็ก (ต่อ)

5. โฟล์ชาร์ตของโปรแกรมแสดงการเลื่อนข้อมูล

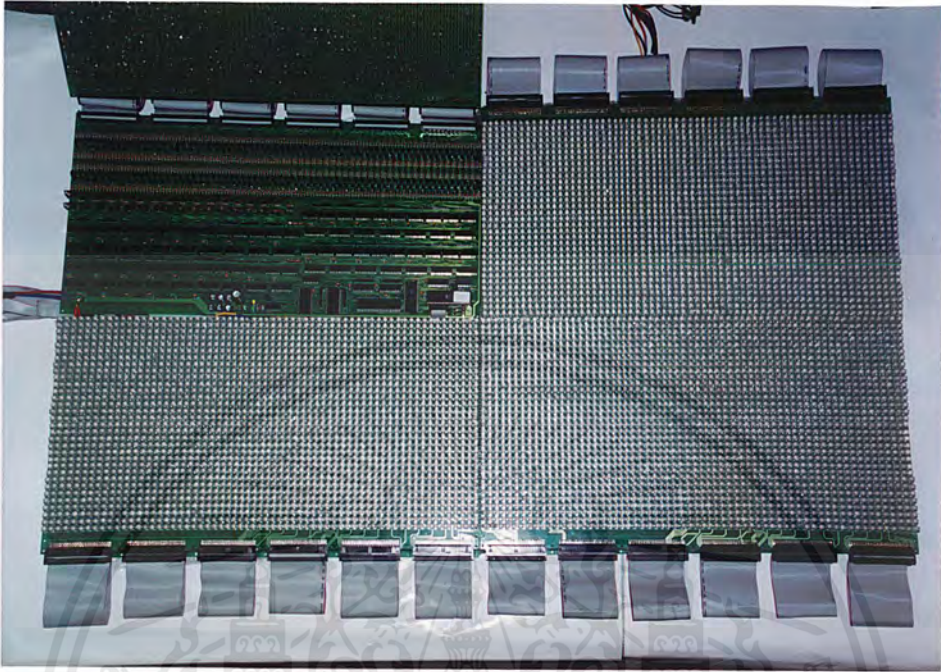


รูปที่ 3.21 แสดงการทำงานของโปรแกรมเลื่อนข้อมูล

6. โฟลว์ชาร์ตของโปรแกรมแสดงการส่งข้อมูล



รูปที่ 3.22 แสดงการทำงานของโปรแกรมส่งข้อมูล



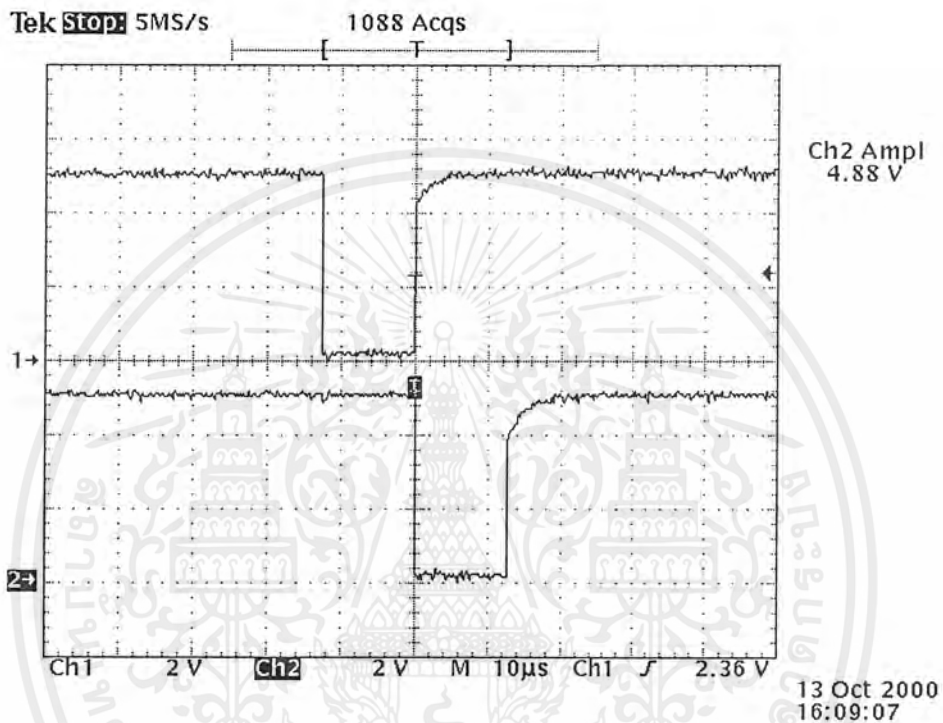
รูปที่ 3.23 แสดงบอร์ดแสดงผลแบบไดอเมตริกซ์ 3 สี

บทที่ 4

การทดลองและผลการทดลอง

4.1 การทดลองวงจรแลตซ์

การทดลองจะทำการวัดสัญญาณช่วงเวลาการแลตซ์ข้อมูลของแต่ละหลัก จะได้ผลการทดลองดังรูปที่ 4.1



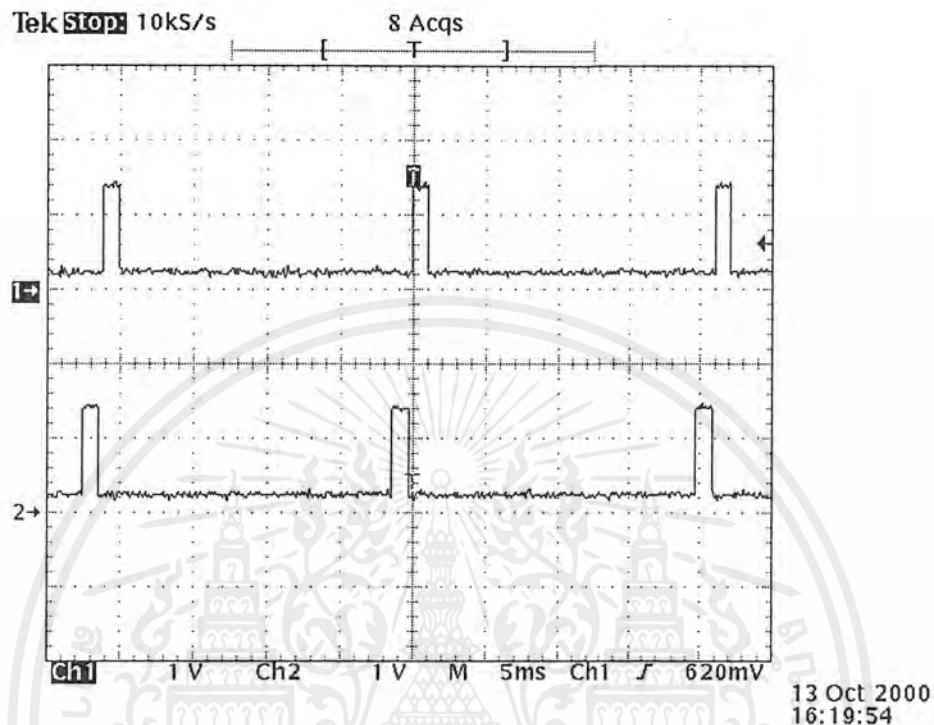
รูปที่ 4.1 แสดงรูปสัญญาณที่ได้จากการแลตซ์ข้อมูล

วงจรแลตซ์ของทุกหลัก จะมีอินพุตต่อกับบัสข้อมูลเดียวกัน ดังนั้นจะต้องมีการแลตซ์ข้อมูลเพื่อให้ข้อมูลที่รับได้อยู่ในตำแหน่งที่ถูกต้อง จากรูปที่ 4.1 เป็นสัญญาณการแลตซ์ข้อมูล ข้อมูลของหลักที่ 1 ถูกส่งเข้ามา เมื่อสถานะที่ขา LE ของไอซีแลตซ์หลักที่ 1 มีการเปลี่ยนสถานะจาก Low เป็น High ข้อมูลที่ออกทางเอาต์พุตของไอซีตัวนี้จะมีสถานะเดียวกันกับข้อมูลอินพุต ช่วงเวลานี้ข้อมูลจะยังไม่ถูกแลตซ์ ดังรูปที่วัดได้ใน channel ที่ 1

เมื่อมีข้อมูลของหลักที่ 2 เข้ามา สถานะที่ขา LE ของไอซีแลตซ์หลักที่ 1 จะเป็น High ซึ่งช่วงเวลานี้ไอซีแลตซ์หลักที่ 1 จะแลตซ์ข้อมูลของหลักที่ 1 ไว้ สถานะที่ขา LE ของไอซีแลตซ์หลักที่ 2 จะมีการเปลี่ยนสถานะจาก Low เป็น High ข้อมูลที่ออกทางเอาต์พุตของไอซีตัวนี้จะมีสถานะเดียวกันกับข้อมูลอินพุตซึ่งเป็นข้อมูลของหลักที่ 2 ช่วงเวลานี้ข้อมูลจะยังไม่ถูกแลตซ์จนกว่า สถานะที่ขา LE ของไอซีแลตซ์หลักที่ 2 จะมีสถานะ เป็น high ดังรูปที่วัดได้ใน channel ที่ 2

4.2 การทดลองวงจรควบคุมการสแกนด้านแนว

การทดลองจะทำการวัดสัญญาณที่ใช้ควบคุมการสแกนเพื่อให้แถวใดแถวหนึ่งแอกทีฟ จะได้ผลการทดลองดังรูปที่ 4.2



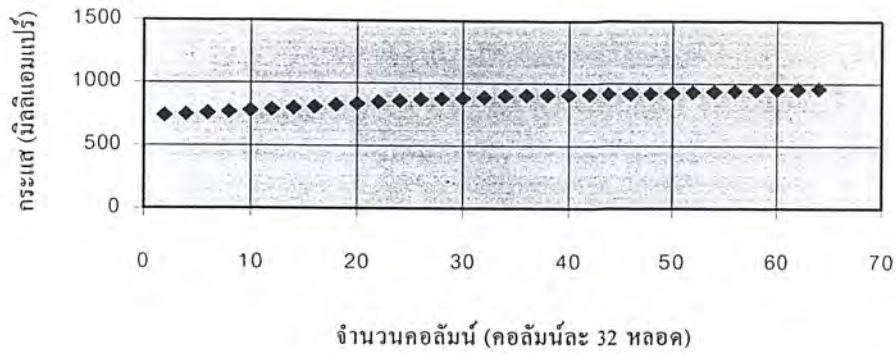
รูปที่ 4.2 แสดงรูปสัญญาณที่ใช้ในการควบคุมการสแกนด้านแนว

สัญญาณทั้ง 2 channel เป็นการวัดเปรียบเทียบระหว่างเอาต์พุตของวงจรควบคุมการสแกนทางด้านแนว โดยวัด 2 แถวที่อยู่ติดกัน ซึ่งสัญญาณนี้จะใช้เป็นแรงดันไบอัสให้กับทรานซิสเตอร์เบอร์ 2SC1061 ที่อยู่ในวงจรขับ สัญญาณที่ใช้ควบคุมการสแกน 1 รอบ จะแบ่งเป็นช่วงการแอกทีฟของแต่ละแถวออกเป็น 16 ช่วงเวลา ในหนึ่งช่วงเวลาของสัญญาณ จะใช้ในการสแกนด้านแนวจำนวน 2 แถว

4.3 การทดลองวัดกระแสของแหล่งจ่าย 5 โวลต์

บอร์ดชุดควบคุมตัวสเลฟแต่ละบอร์ดจะใช้แหล่งจ่ายไฟ 5 โวลต์ จ่ายให้กับอุปกรณ์ประเภทไอซี ซึ่งส่วนใหญ่จะใช้ในการแลตซ์ข้อมูลและการควบคุมการสแกนทางด้านแนว ในการทดลองนี้จะวัดกระแสที่จ่ายให้แต่ละบอร์ดโดยทำการโปรแกรมให้หลอดแอลอีดี (สีแดง) ของบอร์ดติดสว่างครั้งละ 2 คอลัมน์ แล้วทำการเพิ่มขึ้นเรื่อยๆ จนหลอดแอลอีดีติดครบทุกครบทั้ง 64 คอลัมน์ ผลการทดลองที่ได้แสดงดังในรูปที่ 4.3

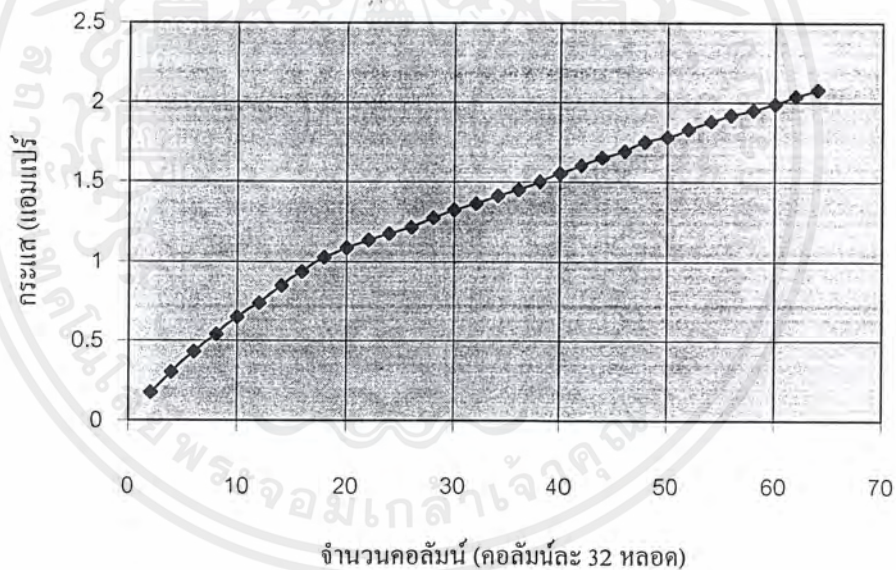
หมายเหตุ การทดลองนี้ใช้แหล่งจ่ายไฟที่ประกอบด้วยหม้อแปลงขนาด 1 แอมป์



รูปที่ 4.3 แสดงผลการทดลองการวัดกระแสของแหล่งจ่าย 5 โวลต์

4.4 การทดลองวัดกระแสของแหล่งจ่าย 12 โวลต์

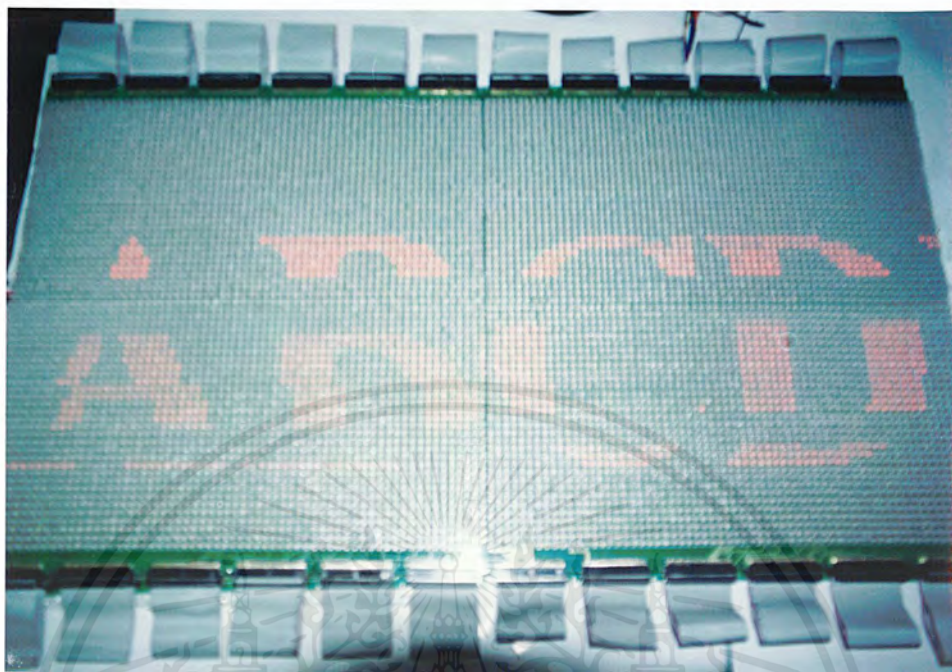
บอร์ดแสดงผลแต่ละบอร์ดจะใช้แหล่งจ่ายไฟ 12 โวลต์ในการขับหลอดแอลอีดี การทดลองนี้จะคล้ายกับการทดลองหัวข้อที่ 4.3 แต่ใช้แหล่งจ่ายไฟแบบสวิตต์ซิ่ง ผลการทดลองที่ได้แสดงดังในรูปที่ 4.4



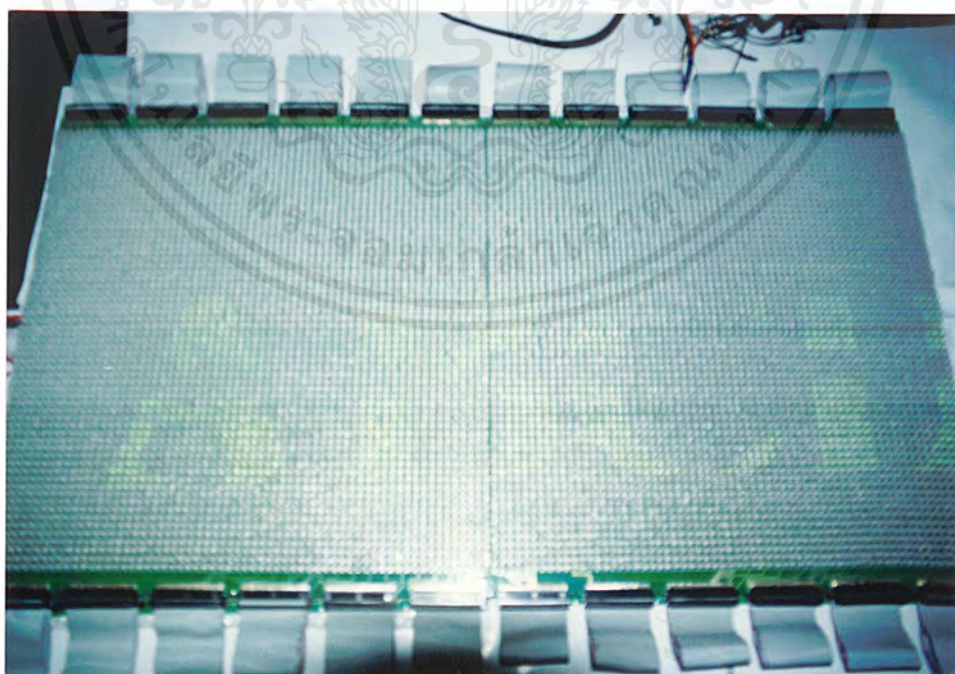
รูปที่ 4.4 แสดงผลการทดลองการวัดกระแสของแหล่งจ่าย 12 โวลต์

4.5 การทดลองการแสดงผลในรูปแบบต่างๆ

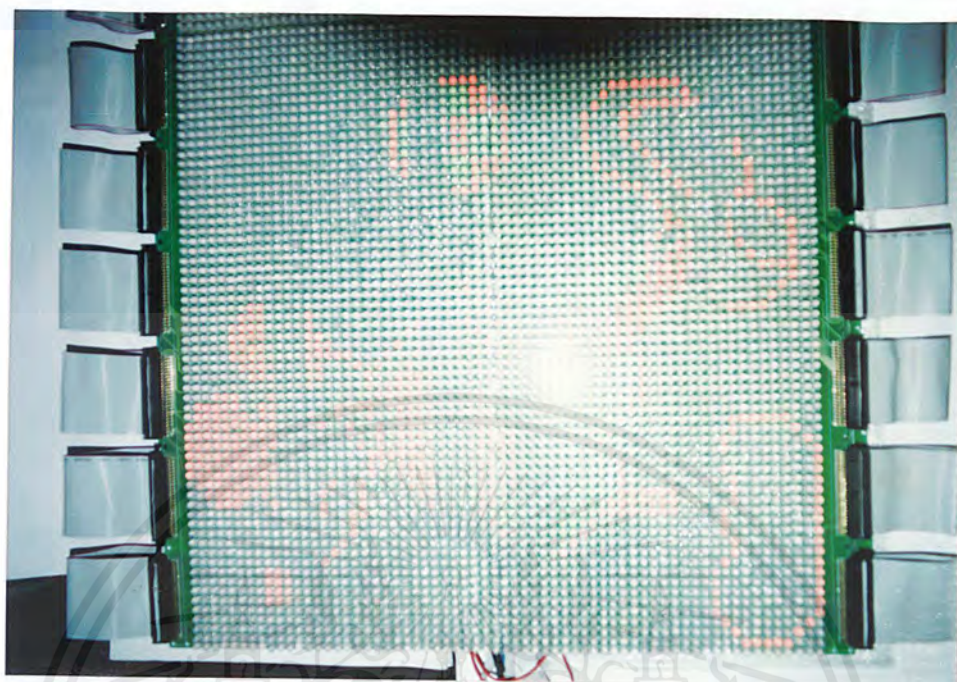
ในการทดลองนี้จะทำการควบคุมการแสดงผลโดยส่งข้อมูลจากคอมพิวเตอร์ไปให้บอร์ดแสดงผลทำการประมวลผลและขับหลอดแอลอีดีให้สว่างตามรูปแบบที่ต้องการ ซึ่งมีผลการทดลองแต่ละแบบดังรูปต่อไปนี้



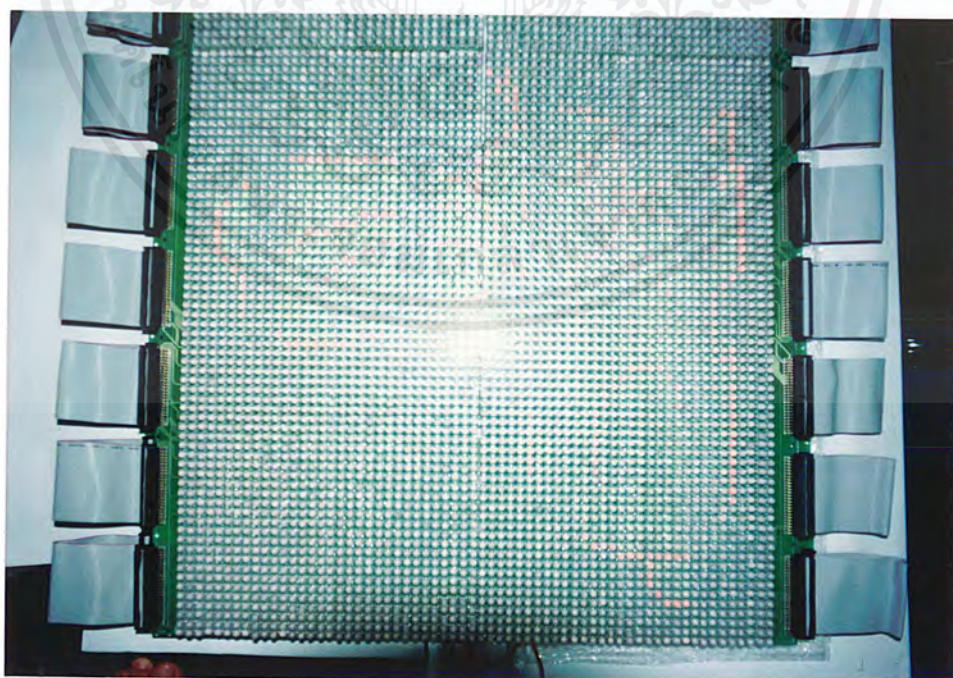
รูปที่ 4.5 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสีแดงบรรทัดเดียว



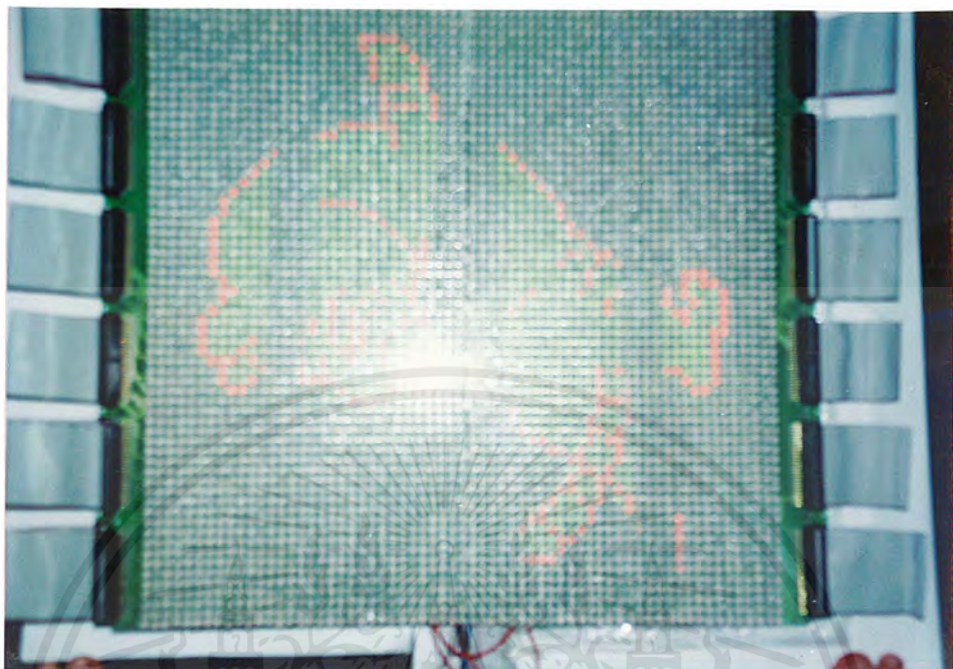
รูปที่ 4.6 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสีเขียวบรรทัดเดียว



รูปที่ 4.7 แสดงผลการทดลองการแสดงผลแบบรูปภาพ (รูปมิกกี้เมาส์)



รูปที่ 4.8 แสดงผลการทดลองการแสดงผลแบบรูปภาพหนึ่ง (รูปกระต่าย)



รูปที่ 4.9 แสดงผลการทดลองการแสดงผลแบบรูปภาพนิ่ง (รูปสุนัข)



รูปที่ 4.10 แสดงผลการทดลองการแสดงผลแบบตัวอักษรสองบรรทัด

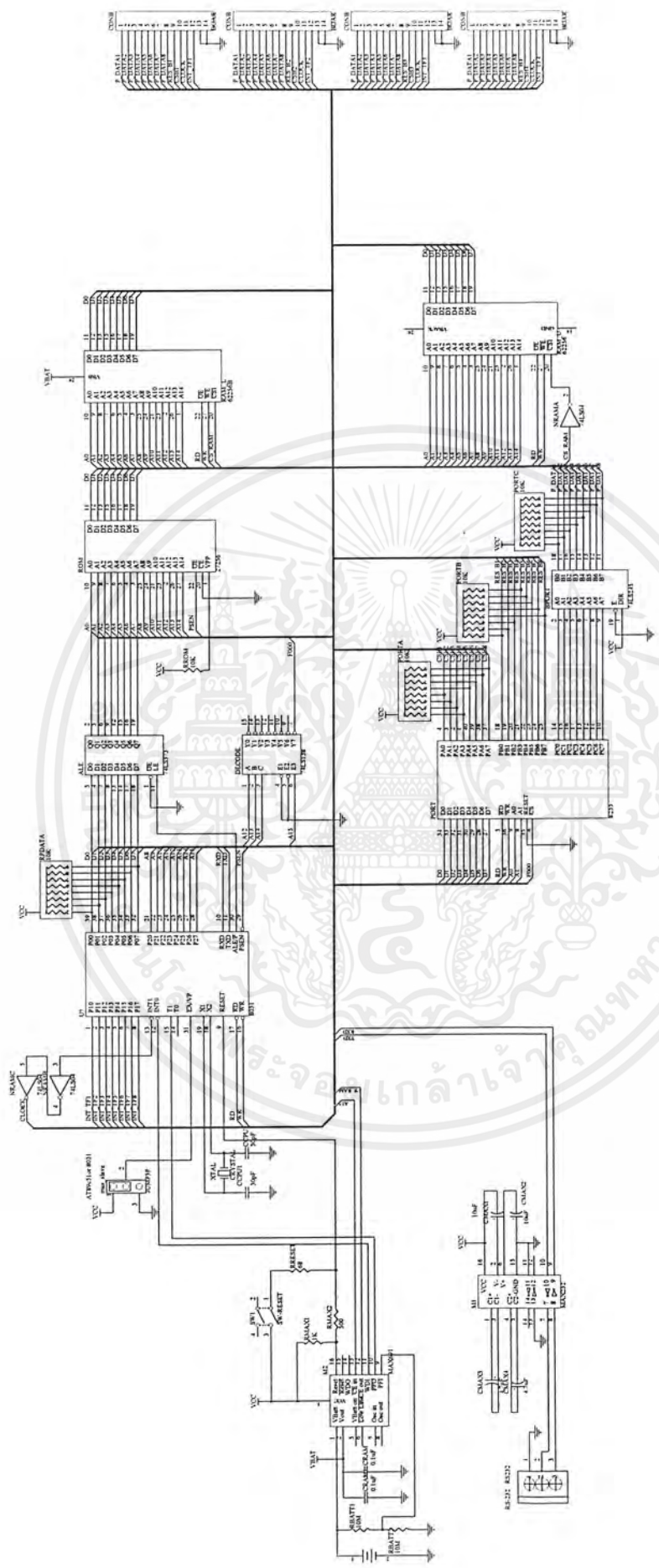
บทที่ 5

บทวิจารณ์และบทสรุป

ด้านการออกแบบและการสร้างฮาร์ดแวร์ซึ่งได้ประยุกต์จากบอร์ดแสดงผลขนาดเล็กเป็นขนาดใหญ่ ได้พบปัญหาและอุปสรรคหลายประการด้วยกัน คือ ในการออกแบบบอร์ดแสดงผลจะต้องให้หลอดแอลอีดีแต่ละหลอดวางใกล้กันให้มากที่สุด เพื่อให้แสดงผลได้ละเอียด ทำให้ลายวงจรต้องมีขนาดเล็ก จึงเกิดปัญหาในขั้นตอนการลงอุปกรณ์และทำให้เกิดการลัดวงจรได้ง่าย ส่วนปัญหาในการแสดงผล คือ ในแต่ละบอร์ดย่อยหลอดแอลอีดีแต่ละหลอดสว่างไม่เท่ากัน โดยเฉพาะเมื่อแสดงผลเป็นสีส้ม ซึ่งสาเหตุมาจากคุณภาพของหลอดแต่ละหลอดไม่เท่ากัน เมื่อนำบอร์ดย่อยทั้งสี่บอร์ดมารวมกันและมีการแสดงผลออกมาถ้าบอร์ดไหนมีจำนวนหลอดแอลอีดีที่ติดมาก ความสว่างของหลอดจะน้อยกว่าบอร์ดที่มีจำนวนหลอดแอลอีดีที่ติดน้อย ทางด้านแหล่งจ่ายไฟ เนื่องจากบอร์ดแสดงผลแบบด็อตเมทริกซ์สามสีนี้จะกินกระแสสูง ถ้าใช้แหล่งจ่ายไฟที่ประกอบด้วยหม้อแปลงธรรมดาจะต้องใช้หม้อแปลงที่มีขนาดใหญ่ น้ำหนักมาก จึงเลือกใช้ชุดจ่ายไฟแบบสวิตชิ่งชนิดที่ใช้กับคอมพิวเตอร์จำนวนสองตัวมาเป็นแหล่งจ่ายไฟให้กับโครงการนี้เพราะมีขนาดเล็ก น้ำหนักเบาและระดับของแรงดันยังเรียกว่า โดยให้บอร์ดแสดงผลย่อยสองบอร์ดใช้แหล่งจ่ายไฟชุดเดียวกัน

ปัญหาทางด้านโปรแกรม ในส่วนของแอสเซมบลีการเขียนโปรแกรมเลื่อนตัวอักษรจะกำหนดให้มีการจัดวางตัวอักษรครั้งละ 5 ตัวเก็บไว้ในบัฟเฟอร์ก่อนแล้วจึงมีการเลื่อนออกมาแสดงผล เนื่องจากการเขียนโปรแกรมทำได้ไม่รัดกุมพอ ทำให้การเลื่อนตัวอักษรแต่ละรอบมีการสะดุดเล็กน้อยทำให้มองดูไม่ราบเรียบ ส่วนการแสดงผลภาพเคลื่อนไหวถ้าต้องการได้ภาพที่ละเอียดเคลื่อนไหวต่อเนื่องสามารถทำได้โดยการเพิ่มจำนวนเฟรมให้มากขึ้นในขั้นตอนของการแมปภาพ แต่เนื่องด้วยทางคณะผู้จัดทำมีเวลาที่จำกัด จึงได้กำหนดให้แต่ละภาพมีจำนวนเฟรมที่เคลื่อนไหวเพียง 5 เฟรมซึ่งภาพที่แสดงออกมาก็พอใช้ได้ ภาพที่นำมาแมปไม่ควรเป็นภาพที่ละเอียดมาก เพราะจำนวนจุดบนบอร์ดแสดงผลมีน้อย ภาพที่ออกมาจะดูไม่เหมือนจริง ความเร็วในการเลื่อนข้อมูลก็เป็นอีกปัญหาหนึ่งที่เกิดขึ้นเพราะขั้นตอนในการประมวลผลมีมากประกอบด้วยโปรแกรมย่อยหลายส่วนและถึงแม้ว่าจะกำหนดให้ไมโครคอนโทรลเลอร์ทำงานที่ความถี่ 24 เมกะเฮิร์ตซ์ มีการเลื่อนข้อมูลทีละ 2 บิต แต่ข้อมูลที่เลื่อนออกมาแสดงผลก็ยังช้าอยู่ แนวทางในการพัฒนาส่วนนี้ควรเลือกไมโครคอนโทรลเลอร์ที่มีความเร็วในการทำงานสูงและเขียนโปรแกรมให้สั้นที่สุดเท่าที่เป็นไปได้





วงจรควบคุมส่วนนาฬิกา

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//                                     Telecommunication project                      //
//                                     Dot matrix display board                      //
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
```

```
unit Liverpool;
```

```
interface
```

```
uses
```

```
    Windows, Messages, SysUtils, Classes, Graphics, Controls, Forms,
    Dialogs,
    StdCtrls, ExtCtrls, CPortCtl, CPort, jpeg;
```

```
type
```

```
    TForm1 = class(TForm)
        Panel1: TPanel;
        Panel2: TPanel;
        ComboBox1: TComboBox;
        Memo1: TMemo;
        Panel3: TPanel;
        Panel4: TPanel;
        Panel5: TPanel;
        Panel6: TPanel;
        RadioGroup1: TRadioGroup;
        Edit1: TEdit;
        Edit2: TEdit;
        Edit3: TEdit;
        RadioGroup2: TRadioGroup;
        Panel12: TPanel;
        Panel13: TPanel;
        Timer1: TTimer;
        ComPort1: TComPort;
        Button1: TButton;
        Button2: TButton;
        Button3: TButton;
        Button4: TButton;
        Button5: TButton;
        ComLed1: TComLed;
        Image1: TImage;
        Image2: TImage;
        Image3: TImage;
        Panel7: TPanel;
        Edit4: TEdit;
        Edit5: TEdit;
        RadioGroup4: TRadioGroup;
        Button6: TButton;
        Panel8: TPanel;
        Edit6: TEdit;
        Edit7: TEdit;
        Button7: TButton;
        RadioGroup5: TRadioGroup;
        RadioGroup6: TRadioGroup;
        RadioGroup7: TRadioGroup;
        RadioGroup8: TRadioGroup;
        RadioGroup9: TRadioGroup;
        RadioGroup10: TRadioGroup;
        RadioGroup11: TRadioGroup;
```

```

procedure Timer1Timer(Sender: TObject);
procedure Panel2Click(Sender: TObject);
procedure RadioGroup1Click(Sender: TObject);
procedure RadioGroup2Click(Sender: TObject);
procedure ComPort1AfterOpen(Sender: TObject);
procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure ComPort1AfterClose(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure RadioGroup5Click(Sender: TObject);
procedure RadioGroup6Click(Sender: TObject);
procedure RadioGroup7Click(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure RadioGroup9Click(Sender: TObject);
procedure RadioGroup10Click(Sender: TObject);
procedure RadioGroup11Click(Sender: TObject);
procedure Button7Click(Sender: TObject);
procedure FormClose(Sender: TObject; var Action: TCloseAction);

private
  { Private declarations }
public
  { Public declarations }
end;

var
  Form1: TForm1;

implementation

{$R *.DFM}

procedure TForm1.Timer1Timer(Sender: TObject);
begin
  if comPort1.Connected then
  begin
    Timer1.Interval := 500; //Define time 500 ms.
    panel3.Visible:= not panel3.Visible;
  end;
end;

procedure TForm1.Panel2Click(Sender: TObject);
begin
  EnumComPorts(ComboBox1.Items); //initial port
  if ComboBox1.ItemIndex > -1 then
    ComPort1.Port := ComboBox1.Items[ComboBox1.ItemIndex];

  if Comport1.connected then
    Comport1.Close
  else
    Comport1.Open;
end;

procedure TForm1.RadioGroup1Click(Sender: TObject);

```

```

begin
    case radiogroup1.ItemIndex of
    0:edit1.Font.Color:=clred;
    1:edit1.Font.Color:=clgreen;
    end;
end;

procedure TForm1.RadioGroup2Click(Sender: TObject);
begin
    case radiogroup2.ItemIndex of
    0:edit2.Font.Color:=clred;
    1:edit2.Font.Color:=clgreen;

    end;
    case radiogroup2.ItemIndex of
    0:edit3.Font.Color:=clred;
    1:edit3.Font.Color:=clgreen;

    end;
end;

procedure TForm1.ComPort1AfterOpen(Sender: TObject);
begin
    memol.Text:='Connect to'+combobox1.Text+' Devices are ready!';
    panel2.Caption:='CLOSE';
end;

procedure TForm1.Button1Click(Sender: TObject);
var str1:string;
begin
    str1:=edit1.Text;
    if comport1.Connected then
    begin
        if radiogroup1.ItemIndex=0 then
            comport1.WriteString(#$9+#$11+#$10+str1+#$20+#$20+#13);
        if radiogroup1.ItemIndex=1 then
            comport1.WriteString(#$9+#$11+#$11+str1+#$20+#$20+#13);

    end;
end;

procedure TForm1.Button2Click(Sender: TObject);
var str2,str3:string;
begin
    str2:=edit2.Text;
    str3:=edit3.Text;
    begin

        if radiogroup2.ItemIndex=0 then
        begin
            comport1.WriteString(#$7+#$15+#$10+str2+#$20+#$20+#13);
            comport1.WriteString(#$8+#$15+#$10+str3+#$20+#$20+#13);
        end;

        if radiogroup2.ItemIndex=1 then

```

```

        begin
            comport1.WriteString(#$7+#$15+#$11+str2+#$20+#$20+#13);
            comport1.WriteString(#$8+#$15+#$11+str3+#$20+#$20+#13);
        end;

    end;

end;

procedure TForm1.Button3Click(Sender: TObject);
    var str1:string;
begin
    if comport1.Connected then
        str1:='A';
        comport1.WriteString(#$6+#$13+#$10+str1+#13);
end;

procedure TForm1.Button4Click(Sender: TObject);
    var str2:string;
begin
    if comport1.Connected then
        str2:='B';
        comport1.WriteString(#$6+#$13+#$10+str2+#13);
end;

procedure TForm1.ComPort1AfterClose(Sender: TObject);
begin
    memol.Text:='disconnect';
    panel2.Caption:='OPEN';
end;

procedure TForm1.Button5Click(Sender: TObject);
    var str3:string;
begin
    if comport1.Connected then
        str3:='C';
        comport1.WriteString(#$6+#$13+#$10+str3+#13);
end;

////////////////////////////////////
//////////////////////////////////// SLIDE FROM LEFT TO RIGHT //////////////////////////////////////
//////////////////////////////////// OR //////////////////////////////////////
//////////////////////////////////// SLIDE FROM RIGHT TO LEFT //////////////////////////////////////
////////////////////////////////////

procedure TForm1.Radiogroup5Click(Sender: TObject);
begin
    case radiogroup5.ItemIndex of
        0:edit4.Font.Color:=clred;
        1:edit4.Font.Color:=clgreen;
        2:edit4.Font.Color:=clred+clgreen;
    end;

    if radiogroup7.ItemIndex = 1 then
        case radiogroup5.ItemIndex of
            0:edit5.Font.Color:=clred;
            1:edit5.Font.Color:=clgreen;
        end;
    end;
end;

```

```

2:edit5.Font.Color:=clred+clgreen;
end;
end;

procedure TForm1.Radiogroup6Click(Sender: TObject);
begin
    case radiogroup6.ItemIndex of
        0:edit5.Font.Color:=clred;
        1:edit5.Font.Color:=clgreen;
        2:edit5.Font.Color:=clred+clgreen;
    end;
end;

procedure TForm1.Radiogroup7Click(Sender: TObject);
begin
    if radiogroup7.ItemIndex = 1 then
    begin
        radiogroup6.Visible:=false;
        case radiogroup5.ItemIndex of
            0:edit5.Font.Color:=clred;
            1:edit5.Font.Color:=clgreen;
            2:edit5.Font.Color:=clred+clgreen;
        end;
    end;

    if radiogroup7.ItemIndex = 0 then
        radiogroup6.Visible:=true;
    if radiogroup7.ItemIndex = 0 then
    begin
        case radiogroup6.ItemIndex of
            0:edit5.Font.Color:=clred;
            1:edit5.Font.Color:=clgreen;
            2:edit5.Font.Color:=clred+clgreen;
        end;
    end;
end;

procedure TForm1.Button6Click(Sender: TObject);
var str4,str5 : string;
begin
    str4:= edit4.Text;
    str5:= edit5.Text;

    ///BIG SIZE ///

    if radiogroup4.ItemIndex = 0 then // slide from left to right
    begin
        if radiogroup7.ItemIndex = 0 then
        begin
            if radiogroup5.ItemIndex = 0 then
                comport1.WriteString('$4+$11+$10+str4+$D');
            if radiogroup5.ItemIndex = 1 then
                comport1.WriteString('$4+$11+$11+str4+$D');
            if radiogroup5.ItemIndex = 2 then
                comport1.WriteString('$4+$11+$12+str4+$D');
            if radiogroup6.ItemIndex = 0 then
                comport1.WriteString('$5+$11+$10+str5+$D');

```

```

        if radiogroup6.ItemIndex = 1 then
            comport1.WriteString(#$5+#$11+#$11+str5+#$D);
        if radiogroup6.ItemIndex = 2 then
            comport1.WriteString(#$5+#$11+#$12+str5+#$D);
        end;
    end;

end;

if radiogroup4.ItemIndex = 1 then // slide from right to left
begin
    if radiogroup7.ItemIndex = 0 then
        begin
            if radiogroup5.ItemIndex = 0 then
                comport1.WriteString(#$4+#$12+#$10+str4+#$D);
            if radiogroup5.ItemIndex = 1 then
                comport1.WriteString(#$4+#$12+#$11+str4+#$D);
            if radiogroup5.ItemIndex = 2 then
                comport1.WriteString(#$4+#$12+#$12+str4+#$D);
            if radiogroup6.ItemIndex = 0 then
                comport1.WriteString(#$5+#$12+#$10+str5+#$D);
            if radiogroup6.ItemIndex = 1 then
                comport1.WriteString(#$5+#$12+#$11+str5+#$D);
            if radiogroup6.ItemIndex = 2 then
                comport1.WriteString(#$5+#$12+#$12+str5+#$D);
            end;
        end;
    end;

    /// SMALL SIZE ///

    if radiogroup4.ItemIndex = 0 then // slide from left to right
    begin
        if radiogroup7.ItemIndex = 1 then
            begin
                if radiogroup5.ItemIndex = 0 then
                    comport1.WriteString(#$4+#$18+#$10+str4+#$D);
                if radiogroup5.ItemIndex = 1 then
                    comport1.WriteString(#$4+#$18+#$11+str4+#$D);
                if radiogroup5.ItemIndex = 2 then
                    comport1.WriteString(#$4+#$18+#$12+str4+#$D);
                if radiogroup5.ItemIndex = 0 then
                    comport1.WriteString(#$5+#$18+#$10+str5+#$D);
                if radiogroup5.ItemIndex = 1 then
                    comport1.WriteString(#$5+#$18+#$11+str5+#$D);
                if radiogroup5.ItemIndex = 2 then
                    comport1.WriteString(#$5+#$18+#$12+str5+#$D);
                end;
            end;
        end;

        if radiogroup4.ItemIndex = 1 then // slide from right to left
        begin
            if radiogroup7.ItemIndex = 1 then
                begin
                    if radiogroup5.ItemIndex = 0 then
                        comport1.WriteString(#$4+#$19+#$10+str4+#$D);
                    if radiogroup5.ItemIndex = 1 then

```

```

        comport1.WriteString(#$4+#$19+#$11+str4+#$D);
    if radiogroup5.ItemIndex = 2 then
        comport1.WriteString(#$4+#$19+#$12+str4+#$D);
    if radiogroup5.ItemIndex = 0 then
        comport1.WriteString(#$5+#$19+#$10+str5+#$D);
    if radiogroup5.ItemIndex = 1 then
        comport1.WriteString(#$5+#$19+#$11+str5+#$D);
    if radiogroup5.ItemIndex = 2 then
        comport1.WriteString(#$5+#$19+#$12+str5+#$D);
    end;

end;

end;

end;

////////////////////////////////////
//////////////////////////////////// SLIDE FROM UPPER ROW //////////////////////////////////////
//////////////////////////////////// OR //////////////////////////////////////
//////////////////////////////////// SLIDE FROM LOWER ROW //////////////////////////////////////
////////////////////////////////////

procedure TForm1.RadioGroup9Click(Sender: TObject);
begin
    if radiogroup11.ItemIndex = 0 then
        begin
            case radiogroup9.ItemIndex of
                0:edit6.Font.Color:=clred;
                1:edit6.Font.Color:=clgreen;
                2:edit6.Font.Color:=clred+clgreen;
            end;
        end;
    if radiogroup11.ItemIndex = 1 then
        begin
            case radiogroup9.ItemIndex of
                0:edit6.Font.Color:=clred;
                1:edit6.Font.Color:=clgreen;
                2:edit6.Font.Color:=clred+clgreen;
            end;
            case radiogroup9.ItemIndex of
                0:edit7.Font.Color:=clred;
                1:edit7.Font.Color:=clgreen;
                2:edit7.Font.Color:=clred+clgreen;
            end;
        end;
    end;

end;

procedure TForm1.RadioGroup10Click(Sender: TObject);
begin
    if radiogroup11.ItemIndex = 0 then
        begin
            case radiogroup10.ItemIndex of
                0:edit7.Font.Color:=clred;
                1:edit7.Font.Color:=clgreen;
                2:edit7.Font.Color:=clred+clgreen;
            end;
        end;
    end;
end;

```



```

        comport1.WriteString(#$3+#$13+#$10+str7+#$D);
    if radiogroup10.ItemIndex = 1 then
        comport1.WriteString(#$3+#$13+#$11+str7+#$D);
    if radiogroup10.ItemIndex = 2 then
        comport1.WriteString(#$3+#$13+#$12+str7+#$D);
    end;

end;

end;

if radiogroup8.ItemIndex = 1 then // slide from lower
begin
    if radiogroup11.ItemIndex = 0 then
        begin
            if radiogroup9.ItemIndex = 0 then
                comport1.WriteString(#$2+#$14+#$10+str6+#$D);
            if radiogroup9.ItemIndex = 1 then
                comport1.WriteString(#$2+#$14+#$11+str6+#$D);
            if radiogroup9.ItemIndex = 2 then
                comport1.WriteString(#$2+#$14+#$12+str6+#$D);
            if radiogroup10.ItemIndex = 0 then
                comport1.WriteString(#$3+#$14+#$10+str7+#$D);
            if radiogroup10.ItemIndex = 1 then
                comport1.WriteString(#$3+#$14+#$11+str7+#$D);
            if radiogroup10.ItemIndex = 2 then
                comport1.WriteString(#$3+#$14+#$12+str7+#$D);
            end;
        end;
    end;

    /// SMALL SIZE ///

    if radiogroup8.ItemIndex = 0 then // slide from upper row
        begin
            if radiogroup11.ItemIndex = 1 then
                begin
                    if radiogroup9.ItemIndex = 0 then
                        comport1.WriteString(#$2+#$16+#$10+str6+#$D);
                    if radiogroup9.ItemIndex = 1 then
                        comport1.WriteString(#$2+#$16+#$11+str6+#$D);
                    if radiogroup9.ItemIndex = 2 then
                        comport1.WriteString(#$2+#$16+#$12+str6+#$D);
                    if radiogroup10.ItemIndex = 0 then
                        comport1.WriteString(#$3+#$16+#$10+str7+#$D);
                    if radiogroup10.ItemIndex = 1 then
                        comport1.WriteString(#$3+#$16+#$11+str7+#$D);
                    if radiogroup10.ItemIndex = 2 then
                        comport1.WriteString(#$3+#$16+#$12+str7+#$D);
                    end;
                end;
            end;
        end;

        end;

        if radiogroup8.ItemIndex = 1 then // slide from lower row
            begin
                if radiogroup11.ItemIndex = 1 then
                    begin
                        if radiogroup9.ItemIndex = 0 then
                            comport1.WriteString(#$2+#$17+#$10+str6+#$D);

```

```

        if radiogroup9.ItemIndex = 1 then
            comport1.WriteString(#$2+#$17+#$11+str6+#$D);
        if radiogroup9.ItemIndex = 2 then
            comport1.WriteString(#$2+#$17+#$12+str6+#$D);
        if radiogroup10.ItemIndex = 0 then
            comport1.WriteString(#$3+#$17+#$10+str7+#$D);
        if radiogroup10.ItemIndex = 1 then
            comport1.WriteString(#$3+#$17+#$11+str7+#$D);
        if radiogroup10.ItemIndex = 2 then
            comport1.WriteString(#$3+#$17+#$12+str7+#$D);
    end;

end;

end;

procedure TForm1.FormClose(Sender: TObject; var Action: TCloseAction);
begin
    comport1.Close;
end;

end.

```



DM74LS154 4-Line to 16-Line Decoder/Demultiplexer

General Description

Each of these 4-line-to-16-line decoders utilizes TTL circuitry to decode four binary-coded inputs into one of sixteen mutually exclusive outputs when both the strobe inputs, G1 and G2, are LOW. The demultiplexing function is performed by using the 4 input lines to address the output line, passing data from one of the strobe inputs with the other strobe input LOW. When either strobe input is HIGH, all outputs are HIGH. These demultiplexers are ideally suited for implementing high-performance memory decoders. All inputs are buffered and input clamping diodes are provided to minimize transmission-line effects and thereby simplify system design.

Features

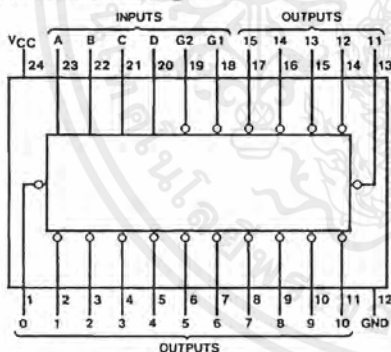
- Decodes 4 binary-coded inputs into one of 16 mutually exclusive outputs
- Performs the demultiplexing function by distributing data from one input line to any one of 16 outputs
- Input clamping diodes simplify system design
- High fan-out, low-impedance, totem-pole outputs
- Typical propagation delay
 - 3 levels of logic 23 ns
 - Strobe 19 ns
- Typical power dissipation 45 mW

Ordering Code:

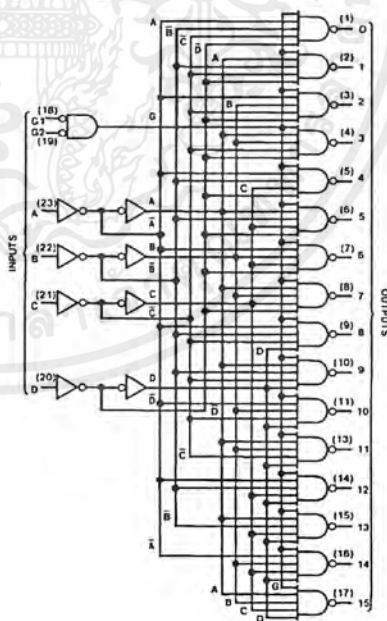
Order Number	Package Number	Package Description
DM74LS154WM	M24B	24-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-013, 0.300 Wide
DM74LS154N	N24A	24-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-010, 0.600 Wide

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

Connection Diagram



Logic Diagram



Function Table

Inputs					Outputs																
G1	G2	D	C	B	A	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
L	L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	L	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	L	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H	H
L	L	L	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H	H
L	L	L	H	L	L	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H	H
L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H
L	L	L	H	L	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	L	H	H	H	H	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L	H
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H	H	H	H	H	H	H	H	H	H	H	H	H	H	H	L
L	L	L	L	L	L	H															

H = HIGH Level
L = Low Level
X = Don't Care

Absolute Maximum Ratings(Note 1)

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

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

Recommended Operating Conditions

Symbol	Parameter	Min	Nom	Max	Units
V_{CC}	Supply Voltage	4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage	2			V
V_{IL}	LOW Level Input Voltage			0.8	V
I_{OH}	HIGH Level Output Current			-0.4	mA
I_{OL}	LOW Level Output Current			8	mA
T_A	Free Air Operating Temperature	0		70	°C

Electrical Characteristics

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

Symbol	Parameter	Conditions	Min	Typ (Note 2)	Max	Units
V_I	Input Clamp Voltage	$V_{CC} = \text{Min}, I_I = -18 \text{ mA}$			-1.5	V
V_{OH}	HIGH Level Output Voltage	$V_{CC} = \text{Min}, I_{OH} = \text{Max}$ $V_{IL} = \text{Max}, V_{IH} = \text{Min}$	2.7	3.4		V
V_{OL}	LOW Level Output Voltage	$V_{CC} = \text{Min}, I_{OL} = \text{Max}$		0.25	0.4	V
		$V_{IL} = \text{Max}, V_{IH} = \text{Min}$		0.35	0.5	
		$I_{OL} = 4 \text{ mA}, V_{CC} = \text{Min}$		0.25	0.4	
I_I	Input Current @ Max Input Voltage	$V_{CC} = \text{Max}, V_I = 7V$			0.1	mA
I_{IH}	HIGH Level Input Current	$V_{CC} = \text{Max}, V_I = 2.7V$			20	μA
I_{IL}	LOW Level Input Current	$V_{CC} = \text{Max}, V_I = 0.4V$			-0.4	mA
I_{OS}	Short Circuit Output Current	$V_{CC} = \text{Max}$ (Note 3)	-20		-100	mA
I_{CC}	Supply Current	$V_{CC} = \text{Max}$ (Note 4)		9	14	mA

Note 2: All typicals are at $V_{CC} = 5V, T_A = 25^\circ\text{C}$.

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

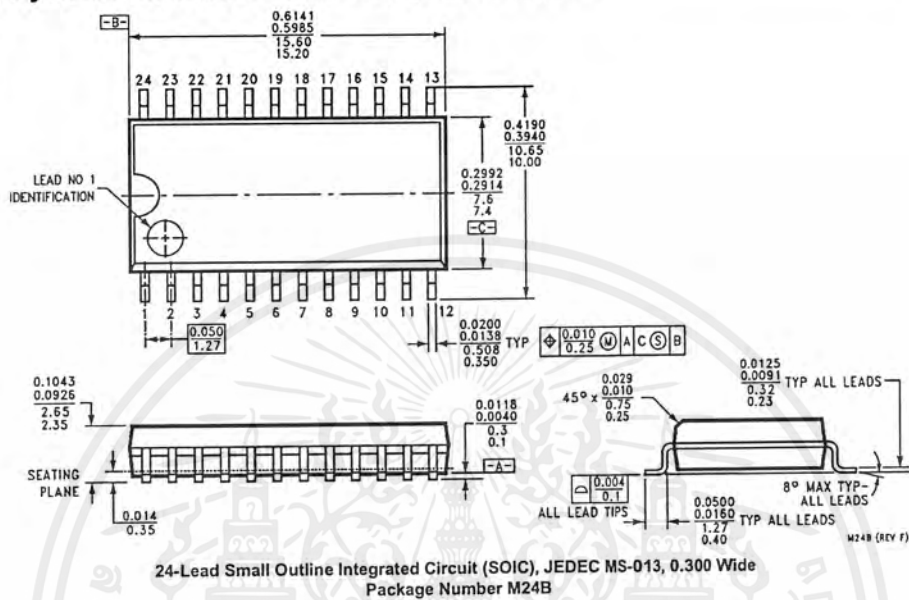
Note 4: I_{CC} is measured with all outputs OPEN and all inputs GROUNDED.

Switching Characteristics

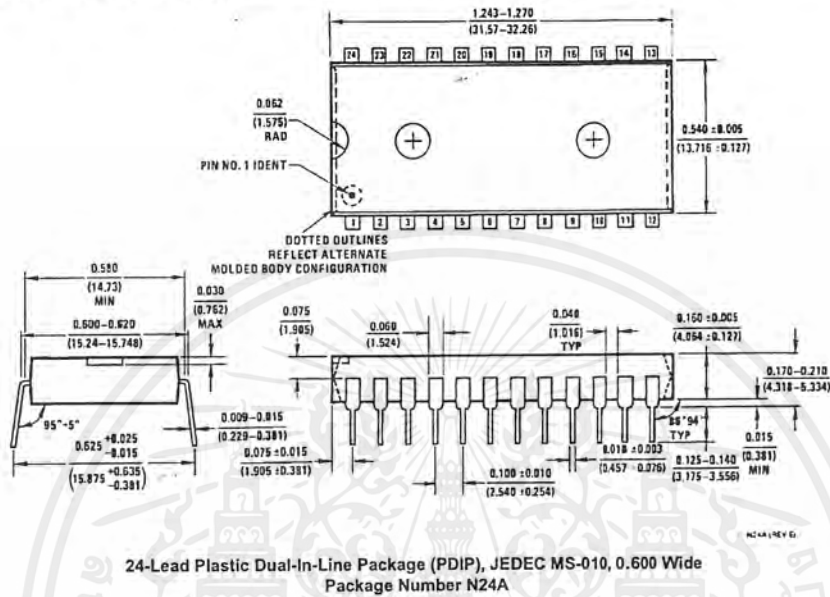
at $V_{CC} = 5V$ and $T_A = 25^\circ\text{C}$

at $V_{CC} = 5V$ and $I_A = 25^{\circ}C$							
Symbol	Parameter	From (Input) To (Output)	$R_L = 2\text{ k}\Omega$				Units
			$C_L = 15\text{ pF}$		$C_L = 50\text{ pF}$		
			Min	Max	Min	Max	
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Data to Output		30		35	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Data to Output		30		35	ns
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Strobe to Output		20		25	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Strobe to Output		25		35	ns

Physical Dimensions inches (millimeters) unless otherwise noted



Physical Dimensions inches (millimeters) unless otherwise noted (Continued)



Fairchild does not assume any responsibility for use of any circuitry described, no circuit patent licenses are implied and Fairchild reserves the right at any time without notice to change said circuitry and specifications.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF THE PRESIDENT OF FAIRCHILD SEMICONDUCTOR CORPORATION. As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, and (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in a significant injury to the user.
2. A critical component in any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

www.fairchildsemi.com

SN74LS373, SN74LS374

Octal Transparent Latch with 3-State Outputs; Octal D-Type Flip-Flop with 3-State Output

The SN74LS373 consists of eight latches with 3-state outputs for bus organized system applications. The flip-flops appear transparent to the data (data changes asynchronously) when Latch Enable (LE) is HIGH. When LE is LOW, the data that meets the setup times is latched. Data appears on the bus when the Output Enable (OE) is LOW. When OE is HIGH the bus output is in the high impedance state.

The SN74LS374 is a high-speed, low-power Octal D-type Flip-Flop featuring separate D-type inputs for each flip-flop and 3-state outputs for bus oriented applications. A buffered Clock (CP) and Output Enable (OE) is common to all flip-flops. The SN74LS374 is manufactured using advanced Low Power Schottky technology and is compatible with all ON Semiconductor TTL families.

- Eight Latches in a Single Package
- 3-State Outputs for Bus Interfacing
- Hysteresis on Latch Enable
- Edge-Triggered D-Type Inputs
- Buffered Positive Edge-Triggered Clock
- Hysteresis on Clock Input to Improve Noise Margin
- Input Clamp Diodes Limit High Speed Termination Effects

GUARANTEED OPERATING RANGES

Symbol	Parameter	Min	Typ	Max	Unit
V _{CC}	Supply Voltage	4.75	5.0	5.25	V
T _A	Operating Ambient Temperature Range	0	25	70	°C
I _{OH}	Output Current – High			–2.6	mA
I _{OL}	Output Current – Low			24	mA



ON Semiconductor

<http://onsemi.com>

LOW
POWER
SCHOTTKY

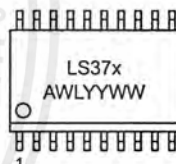
MARKING DIAGRAMS



PDIP–20
N SUFFIX
CASE 738



SOIC–20
DW SUFFIX
CASE 751D



x = 3 or 4
A = Assembly Location
WL = Wafer Lot
YY = Year
WW = Work Week

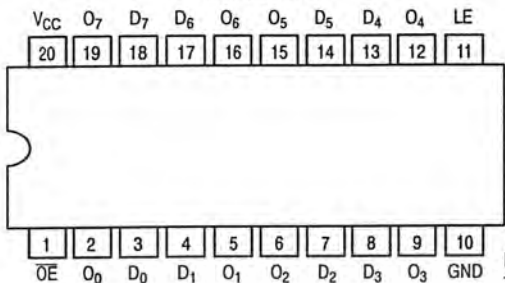
ORDERING INFORMATION

Device	Package	Shipping
SN74LS373N	PDIP–20	1440 Units/Box
SN74LS373DW	SOIC–20	2500/Tape & Reel
SN74LS374N	PDIP–20	1440 Units/Box
SN74LS374DW	SOIC–20	2500/Tape & Reel

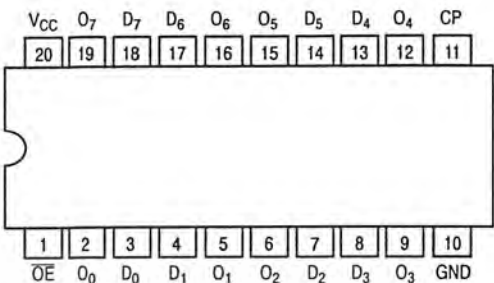
SN74LS373, SN74LS374

CONNECTION DIAGRAM DIP (TOP VIEW)

SN74LS373



SN74LS374



NOTE:
The Flatpak version
has the same pinouts
(Connection Diagram) as
the Dual In-Line Package.

PIN NAMES

$D_0 - D_7$	Data Inputs
LE	Latch Enable (Active HIGH) Input
CP	Clock (Active HIGH Going Edge) Input
$\overline{OE}$	Output Enable (Active LOW) Input
$O_0 - O_7$	Outputs

NOTES:

a) 1 TTL Unit Load (U.L.) = 40 μ A HIGH/1.6 mA LOW.

LOADING (Note a)

HIGH	LOW
0.5 U.L.	0.25 U.L.
0.5 U.L.	0.25 U.L.
0.5 U.L.	0.25 U.L.
0.5 U.L.	0.25 U.L.
65 U.L.	15 U.L.

TRUTH TABLE

LS373

D_n	LE	$\overline{OE}$	O_n
H	H	L	H
L	H	L	L
X	L	L	Q_0
X	X	H	Z*

LS374

D_n	LE	$\overline{OE}$	O_n
H		L	H
L		L	L
X	X	H	Z*

H = HIGH Voltage Level

L = LOW Voltage Level

X = Immaterial

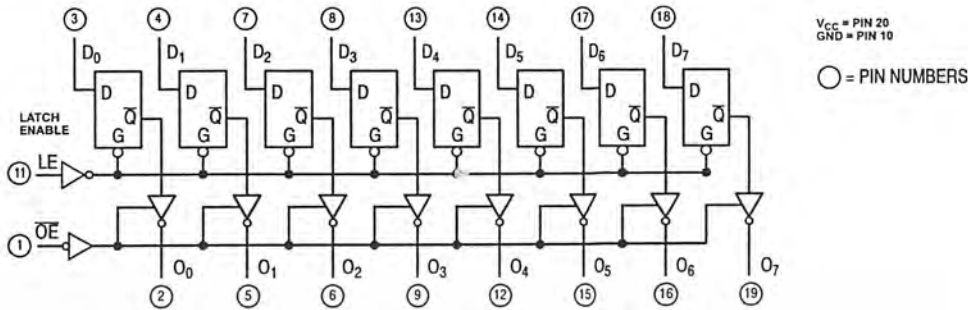
Z = High Impedance

* Note: Contents of flip-flops unaffected by the state of the Output Enable input ($\overline{OE}$).

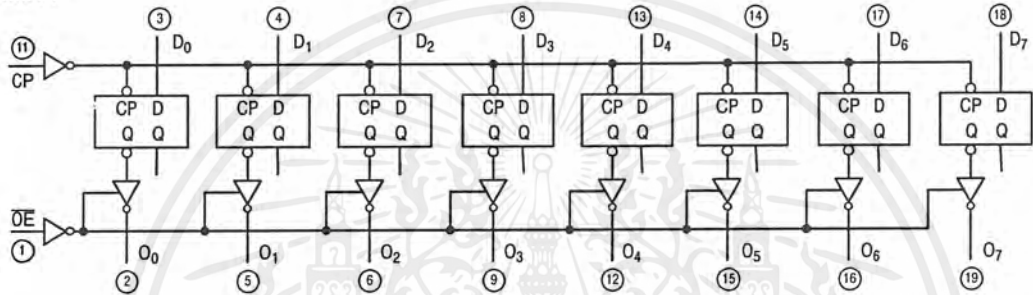
SN74LS373, SN74LS374

LOGIC DIAGRAMS

SN74LS373



SN74LS374



DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

Symbol	Parameter	Limits			Unit	Test Conditions
		Min	Typ	Max		
V _{IH}	Input HIGH Voltage	2.0			V	Guaranteed Input HIGH Voltage for All Inputs
V _{IL}	Input LOW Voltage			0.8	V	Guaranteed Input LOW Voltage for All Inputs
V _{IK}	Input Clamp Diode Voltage		-0.65	-1.5	V	V _{CC} = MIN, I _{IN} = -18 mA
V _{OH}	Output HIGH Voltage	2.4	3.1		V	V _{CC} = MIN, I _{OH} = MAX, V _{IN} = V _{IH} or V _{IL} per Truth Table
V _{OL}	Output LOW Voltage		0.25	0.4	V	I _{OL} = 12 mA
			0.35	0.5	V	I _{OL} = 24 mA
I _{OZH}	Output Off Current HIGH			20	μA	V _{CC} = MAX, V _{OUT} = 2.7 V
I _{OZL}	Output Off Current LOW			-20	μA	V _{CC} = MAX, V _{OUT} = 0.4 V
I _{IH}	Input HIGH Current			20	μA	V _{CC} = MAX, V _{IN} = 2.7 V
				0.1	mA	V _{CC} = MAX, V _{IN} = 7.0 V
I _{IL}	Input LOW Current			-0.4	mA	V _{CC} = MAX, V _{IN} = 0.4 V
I _{OS}	Short Circuit Current (Note 1.)	-30		-130	mA	V _{CC} = MAX
I _{CC}	Power Supply Current			40	mA	V _{CC} = MAX

1. Not more than one output should be shorted at a time, nor for more than 1 second.

SN74LS373, SN74LS374

AC CHARACTERISTICS ($T_A = 25^\circ\text{C}$, $V_{CC} = 5.0\text{ V}$)

Symbol	Parameter	Limits						Unit	Test Conditions
		LS373			LS374				
		Min	Typ	Max	Min	Typ	Max		
t _{MAX}	Maximum Clock Frequency				35	50		MHz	C _L = 45 pF, R _L = 667 Ω
t _{PLH} t _{PHL}	Propagation Delay, Data to Output		12 12	18 18				ns	
t _{PLH} t _{PHL}	Clock or Enable to Output		20 18	30 30		15 19	28 28	ns	
t _{PZH} t _{PZL}	Output Enable Time		15 25	28 36		20 21	28 28	ns	
t _{PHZ} t _{PLZ}	Output Disable Time		12 15	20 25		12 15	20 25	ns	C _L = 5.0 pF

AC SETUP REQUIREMENTS ($T_A = 25^\circ\text{C}$, $V_{CC} = 5.0\text{ V}$)

Symbol	Parameter	Limits				Unit
		LS373		LS374		
		Min	Max	Min	Max	
t _w	Clock Pulse Width	15		15		ns
t _s	Setup Time	5.0		20		ns
t _h	Hold Time	20		0		ns

DEFINITION OF TERMS

SETUP TIME (t_s) — is defined as the minimum time required for the correct logic level to be present at the logic input prior to LE transition from HIGH-to-LOW in order to be recognized and transferred to the outputs.

HOLD TIME (t_h) — is defined as the minimum time following the LE transition from HIGH-to-LOW that the logic level must be maintained at the input in order to ensure continued recognition.

SN74LS373, SN74LS374

SN74LS373

AC WAVEFORMS

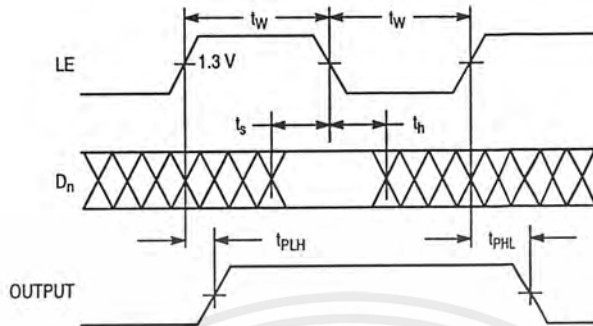


Figure 1.

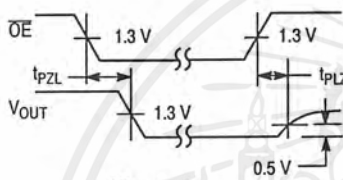


Figure 2.

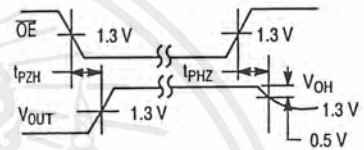
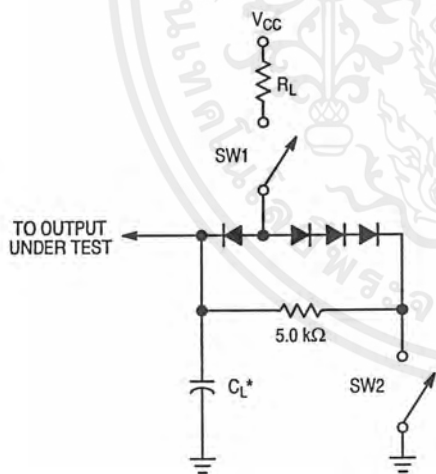


Figure 3.

AC LOAD CIRCUIT



* Includes Jig and Probe Capacitance.

Figure 4.

SWITCH POSITIONS

SYMBOL	SW1	SW2
tPZH	Open	Closed
tPZL	Closed	Open
tPLZ	Closed	Closed
tPHZ	Closed	Closed

SN74LS373, SN74LS374

SN74LS374

AC WAVEFORMS

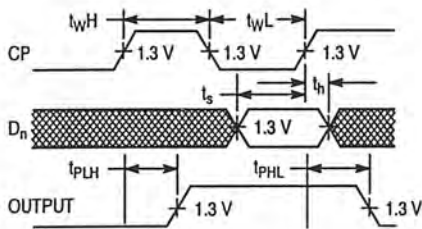


Figure 5.

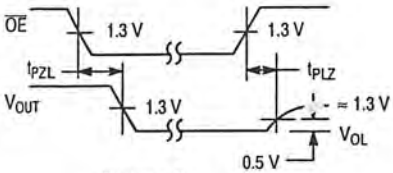


Figure 6.

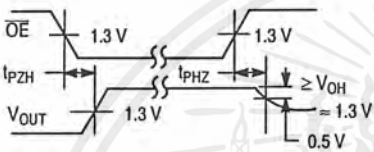
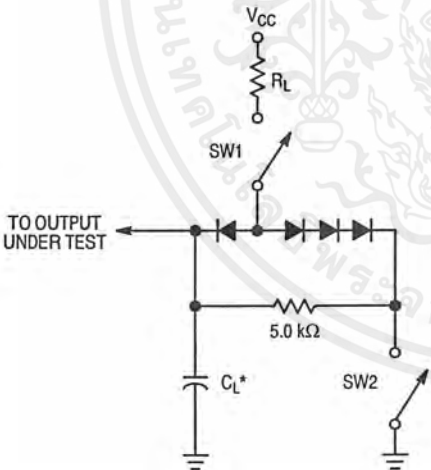


Figure 7.

AC LOAD CIRCUIT



* Includes Jig and Probe Capacitance.

SWITCH POSITIONS

SYMBOL	SW1	SW2
tPZH	Open	Closed
tPZL	Closed	Open
tPLZ	Closed	Closed
tPHZ	Closed	Closed

Figure 8.

SN74LS122 SN74LS123

Retriggerable Monostable Multivibrators

These dc triggered multivibrators feature pulse width control by three methods. The basic pulse width is programmed by selection of external resistance and capacitance values. The LS122 has an internal timing resistor that allows the circuits to be used with only an external capacitor. Once triggered, the basic pulse width may be extended by retriggering the gated low-level-active (A) or high-level-active (B) inputs, or be reduced by use of the overriding clear.

- Overriding Clear Terminates Output Pulse
- Compensated for V_{CC} and Temperature Variations
- DC Triggered from Active-High or Active-Low Gated Logic Inputs
- Retriggerable for Very Long Output Pulses, up to 100% Duty Cycle
- Internal Timing Resistors on LS122

GUARANTEED OPERATING RANGES

Symbol	Parameter	Min	Typ	Max	Unit
V_{CC}	Supply Voltage	4.75	5.0	5.25	V
T_A	Operating Ambient Temperature Range	0	25	70	°C
I_{OH}	Output Current – High			-0.4	mA
I_{OL}	Output Current – Low			8.0	mA
R_{ext}	External Timing Resistance	5.0		260	k Ω
C_{ext}	External Capacitance	No Restriction			
R_{ext}/C_{ext}	Wiring Capacitance at R_{ext}/C_{ext} Terminal			50	pF



ON Semiconductor

Formerly a Division of Motorola

<http://onsemi.com>

LOW POWER SCHOTTKY



PLASTIC
N SUFFIX
CASE 646



SOIC
D SUFFIX
CASE 751A



PLASTIC
N SUFFIX
CASE 648



SOIC
D SUFFIX
CASE 751B

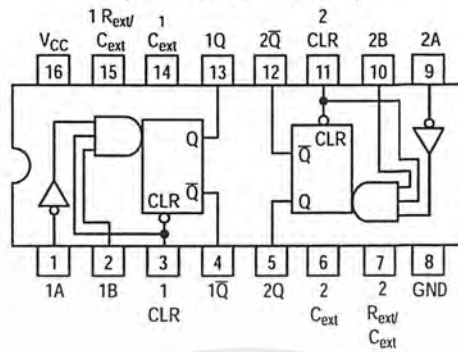
ORDERING INFORMATION

Device	Package	Shipping
SN74LS122N	14 Pin DIP	2000 Units/Box
SN74LS122D	14 Pin	2500/Tape & Reel
SN74LS123N	16 Pin DIP	2000 Units/Box
SN74LS123D	16 Pin	2500/Tape & Reel

SN74LS122 SN74LS123

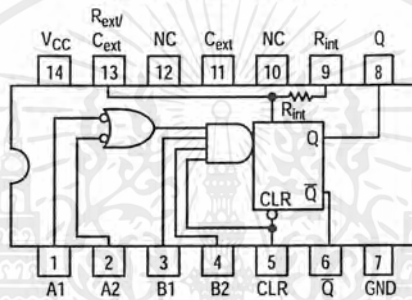
SN74LS123 (TOP VIEW)

(SEE NOTES 1 THRU 4)



SN74LS122 (TOP VIEW)

(SEE NOTES 1 THRU 4)



NC — NO INTERNAL CONNECTION.

NOTES:

1. An external timing capacitor may be connected between C_{ext} and R_{ext}/C_{ext} (positive).
2. To use the internal timing resistor of the LS122, connect R_{int} to V_{CC} .
3. For improved pulse width accuracy connect an external resistor between R_{ext}/C_{ext} and V_{CC} with R_{int} open-circuited.
4. To obtain variable pulse widths, connect an external variable resistance between R_{int}/C_{ext} and V_{CC} .

SN74LS122 SN74LS123

LS122 FUNCTIONAL TABLE

INPUTS					OUTPUTS	
CLEAR	A1	A2	B1	B2	Q	$\bar{Q}$
L	X	X	X	X	L	H
X	H	H	X	X	L	H
X	X	X	L	X	L	H
X	X	X	X	L	L	H
H	L	X	↑	H		
H	L	X	H	↑		
H	X	L	↑	H		
H	X	L	H	↑		
H	H	↓	H	H		
H	↓	↓	H	H		
H	↓	H	H	H		
↑	L	X	H	H		
↑	X	L	H	H		

LS123 FUNCTIONAL TABLE

INPUTS			OUTPUTS	
CLEAR	A	B	Q	$\bar{Q}$
L	X	X	L	H
X	H	X	L	H
X	X	L	L	H
H	L	↑		
H	↓	H		
↑	L	H		

TYPICAL APPLICATION DATA

The output pulse t_W is a function of the external components, C_{ext} and R_{ext} or C_{ext} and R_{int} on the LS122. For values of $C_{ext} \geq 1000$ pF, the output pulse at $V_{CC} = 5.0$ V and $V_{RC} = 5.0$ V (see Figures 1, 2, and 3) is given by

$$t_W = K R_{ext} C_{ext} \text{ where } K \text{ is nominally } 0.45$$

If C_{ext} is on pF and R_{ext} is in k Ω then t_W is in nanoseconds.

The C_{ext} terminal of the LS122 and LS123 is an internal connection to ground, however for the best system performance C_{ext} should be hard-wired to ground.

Care should be taken to keep R_{ext} and C_{ext} as close to the monostable as possible with a minimum amount of inductance between the R_{ext}/C_{ext} junction and the R_{ext}/C_{ext} pin. Good groundplane and adequate bypassing should be designed into the system for optimum performance to ensure that no false triggering occurs.

It should be noted that the C_{ext} pin is internally connected to ground on the LS122 and LS123, but not on the LS221. Therefore, if C_{ext} is hard-wired externally to ground, substitution of a LS221 onto a LS123 socket will cause the LS221 to become non-functional.

The switching diode is not needed for electrolytic capacitance application and should not be used on the LS122 and LS123.

To find the value of K for $C_{ext} \geq 1000$ pF, refer to Figure 4. Variations on V_{CC} or V_{RC} can cause the value of K to change, as can the temperature of the LS123, LS122.

Figures 5 and 6 show the behavior of the circuit shown in Figures 1 and 2 if separate power supplies are used for V_{CC} and V_{RC} . If V_{CC} is tied to V_{RC} , Figure 7 shows how K will vary with V_{CC} and temperature. Remember, the changes in R_{ext} and C_{ext} with temperature are not calculated and included in the graph.

As long as $C_{ext} \geq 1000$ pF and $5K \leq R_{ext} \leq 260K$, the change in K with respect to R_{ext} is negligible.

If $C_{ext} \leq 1000$ pF the graph shown on Figure 8 can be used to determine the output pulse width. Figure 9 shows how K will change for $C_{ext} \leq 1000$ pF if V_{CC} and V_{RC} are connected to the same power supply. The pulse width t_W in nanoseconds is approximated by

$$t_W = 6 + 0.05 C_{ext} (\text{pF}) + 0.45 R_{ext} (\text{k}\Omega) C_{ext} + 11.6 R_{ext}$$

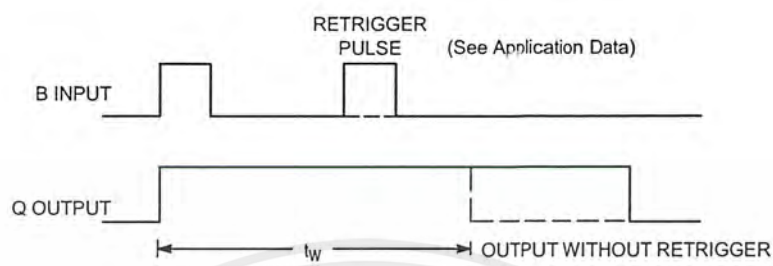
In order to trim the output pulse width, it is necessary to include a variable resistor between V_{CC} and the R_{ext}/C_{ext} pin or between V_{CC} and the R_{ext} pin of the LS122. Figure 10, 11, and 12 show how this can be done. R_{ext} remote should be kept as close to the monostable as possible.

Retriggering of the part, as shown in Figure 3, must not occur before C_{ext} is discharged or the retrigger pulse will not have any effect. The discharge time of C_{ext} in nanoseconds is guaranteed to be less than $0.22 C_{ext}$ (pF) and is typically $0.05 C_{ext}$ (pF).

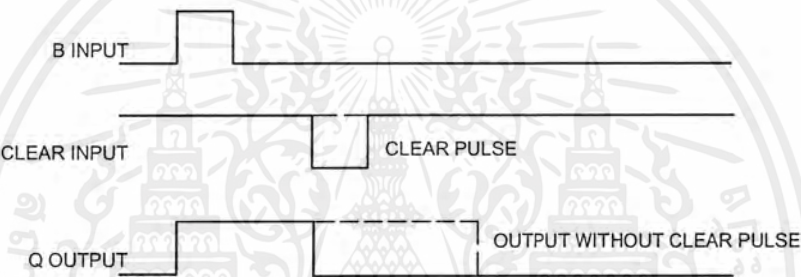
For the smallest possible deviation in output pulse widths from various devices, it is suggested that C_{ext} be kept ≥ 1000 pF.

SN74LS122 SN74LS123

WAVEFORMS



EXTENDING PULSE WIDTH



OVERRIDING THE OUTPUT PULSE

SN74LS122 SN74LS123

DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

Symbol	Parameter	Limits			Unit	Test Conditions
		Min	Typ	Max		
V_{IH}	Input HIGH Voltage	2.0			V	Guaranteed Input HIGH Voltage for All Inputs
V_{IL}	Input LOW Voltage			0.8	V	Guaranteed Input LOW Voltage for All Inputs
V_{IK}	Input Clamp Diode Voltage		-0.65	-1.5	V	$V_{CC} = \text{MIN}$, $I_{IN} = -18 \text{ mA}$
V_{OH}	Output HIGH Voltage	2.7	3.5		V	$V_{CC} = \text{MIN}$, $I_{OH} = \text{MAX}$, $V_{IN} = V_{IH}$ or V_{IL} per Truth Table
V_{OL}	Output LOW Voltage		0.25	0.4	V	$I_{OL} = 4.0 \text{ mA}$
			0.35	0.5	V	$I_{OL} = 8.0 \text{ mA}$
I_{IH}	Input HIGH Current			20	μA	$V_{CC} = \text{MAX}$, $V_{IN} = 2.7 \text{ V}$
				0.1	mA	$V_{CC} = \text{MAX}$, $V_{IN} = 7.0 \text{ V}$
I_{IL}	Input LOW Current			-0.4	mA	$V_{CC} = \text{MAX}$, $V_{IN} = 0.4 \text{ V}$
I_{OS}	Short Circuit Current (Note 1)	-20		-100	mA	$V_{CC} = \text{MAX}$
I_{CC}	Power Supply Current	LS122		11	mA	$V_{CC} = \text{MAX}$
		LS123		20		

Note 1: Not more than one output should be shorted at a time, nor for more than 1 second.

AC CHARACTERISTICS ($T_A = 25^\circ\text{C}$, $V_{CC} = 5.0 \text{ V}$)

Symbol	Parameter	Limits			Unit	Test Conditions
		Min	Typ	Max		
t_{PLH} t_{PHL}	Propagation Delay, A to Q Propagation Delay, A to $\overline{Q}$		23 32	33 45	ns	$C_{ext} = 0$ $C_L = 15 \text{ pF}$ $R_{ext} = 5.0 \text{ k}\Omega$ $R_L = 2.0 \text{ k}\Omega$
t_{PLH} t_{PHL}	Propagation Delay, B to Q Propagation Delay, B to $\overline{Q}$		23 34	44 56	ns	
t_{PLH} t_{PHL}	Propagation Delay, Clear to $\overline{Q}$ Propagation Delay, Clear to Q		28 20	45 27	ns	
$t_{W \min}$	A or B to Q		116	200	ns	
t_{WQ}	A to B to Q	4.0	4.5	5.0	μs	

AC SETUP REQUIREMENTS ($T_A = 25^\circ\text{C}$, $V_{CC} = 5.0 \text{ V}$)

Symbol	Parameter	Limits			Unit	Test Conditions
		Min	Typ	Max		
t_W	Pulse Width	40			ns	

SN74LS122 SN74LS123

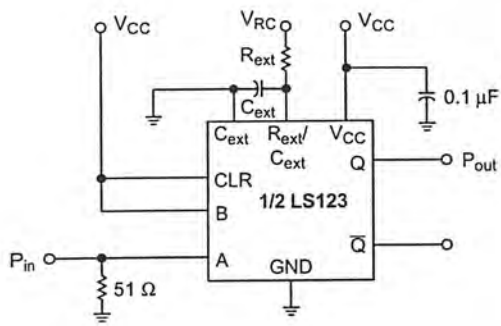


Figure 1.

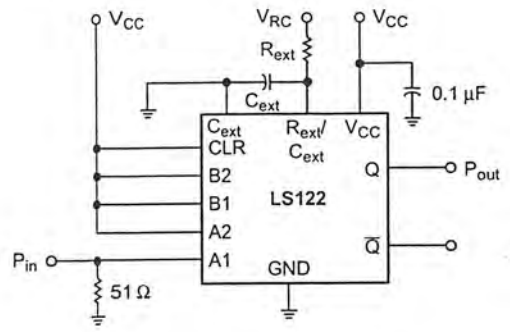


Figure 2.

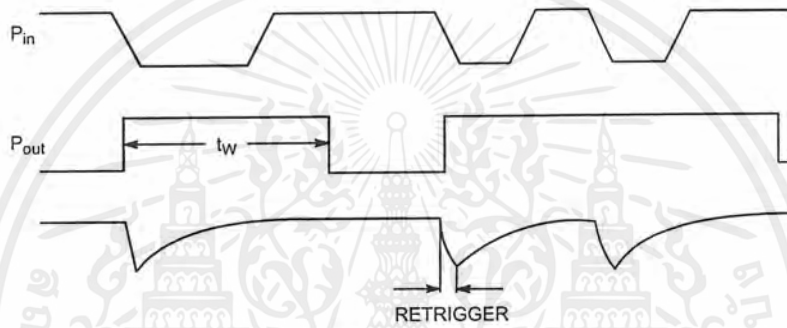


Figure 3.

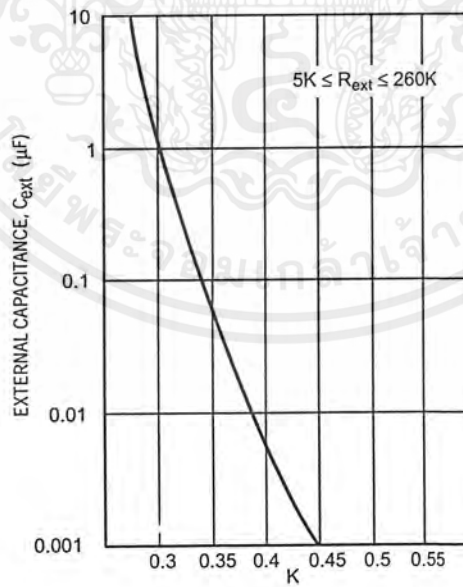


Figure 4.

SN74LS122 SN74LS123

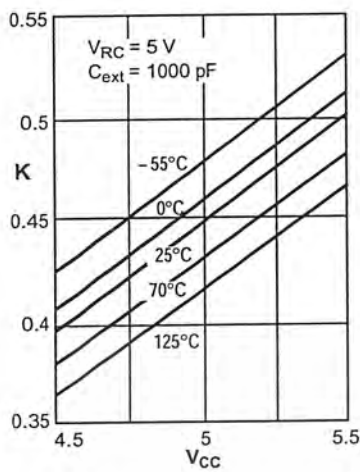


Figure 5. K versus V_{CC}

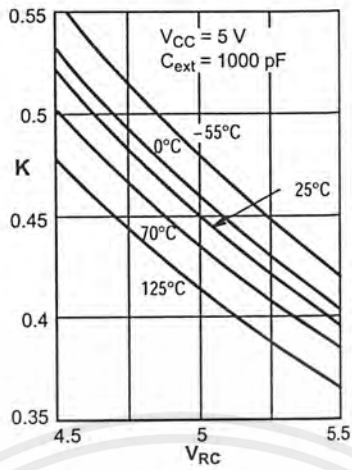


Figure 6. K versus V_{RC}

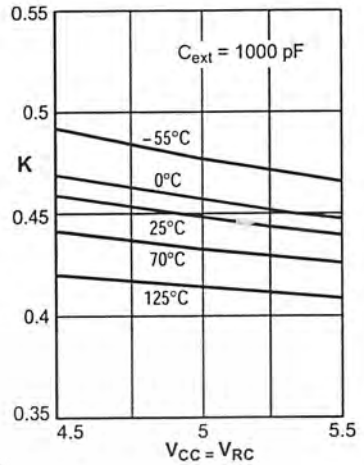


Figure 7. K versus V_{CC} and V_{RC}

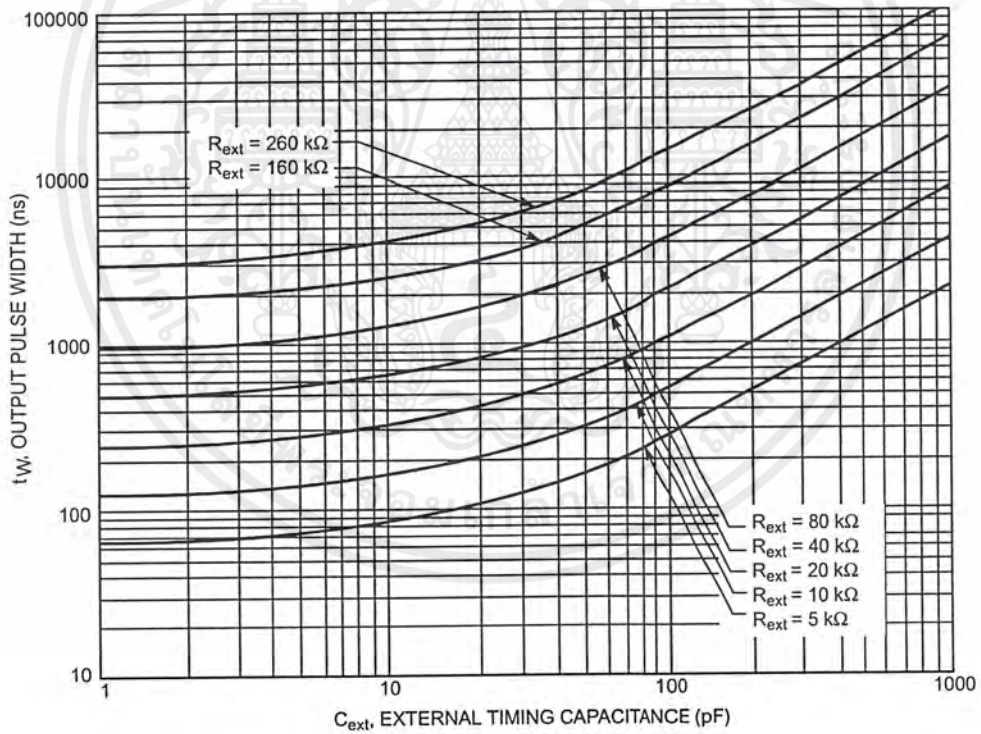


Figure 8.

SN74LS122 SN74LS123

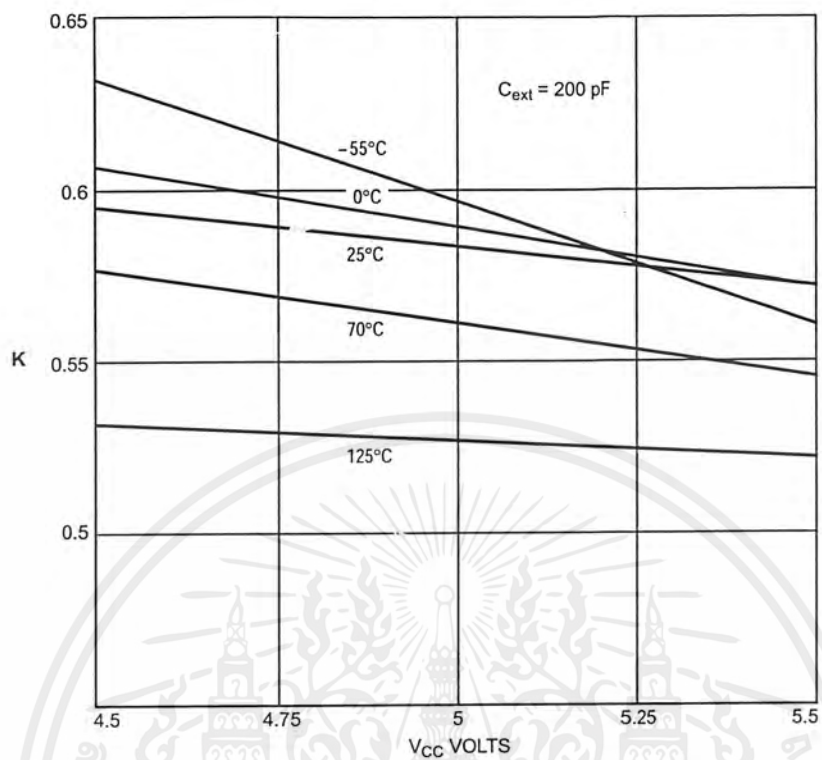


Figure 9.

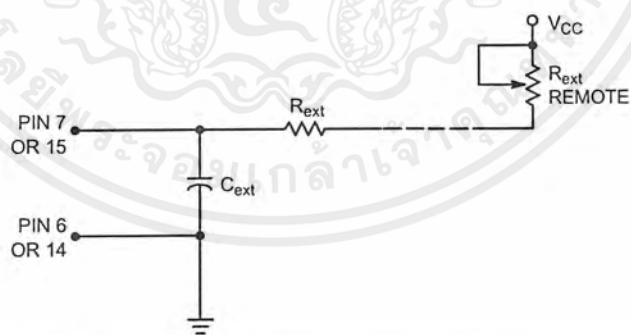


Figure 10. LS123 Remote Trimming Circuit

SN74LS122 SN74LS123

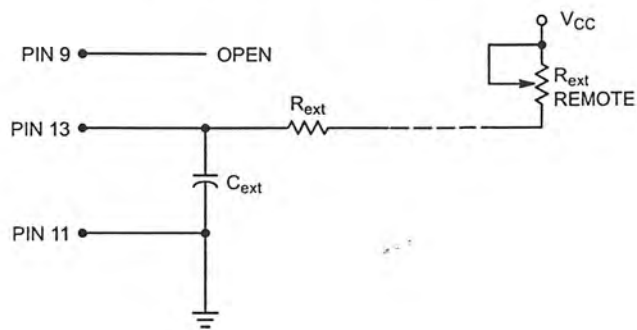


Figure 11. LS122 Remote Trimming Circuit Without R_{ext}

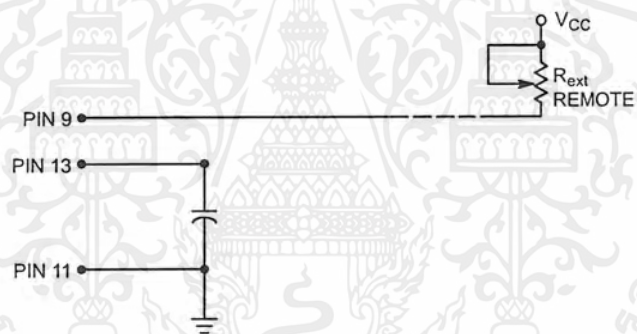


Figure 12. LS122 Remote Trimming Circuit with R_{int}

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สามารถจัดทำสำเร็จลุล่วงได้ด้วยดี ก็เพราะได้รับความแนะนำและคำปรึกษาจาก
ท่าน รศ.ดร. ถวิล พึ่งมา และท่าน ศ.มบุญ สุขเกษม คณะผู้จัดทำจึงขอขอบพระคุณเป็นอย่างสูง

คณะผู้จัดทำขอกราบขอบพระคุณ คุณพ่อ คุณแม่ ที่ให้กำลังใจและเป็นผู้ซึ่งให้โอกาสทางการศึกษา
และทุกสิ่งทุกอย่างมาโดยตลอด

ขอขอบคุณพี่ พิสิฐ บุญศรีเมือง , สายัณห์ พวงเงิน, นิวัติ เข่นกาย ที่คอยให้คำปรึกษา คำแนะนำ และ
ให้การช่วยเหลือแก้ปัญหาเกี่ยวกับปริญญานิพนธ์ฉบับนี้

ขอขอบคุณ พี่ๆ เพื่อนๆ และน้องๆ ทุกคนที่คอยเป็นกำลังใจในการทำปริญญานิพนธ์มาโดยตลอด

คณะผู้จัดทำ

นาย พิเชฐ เหล่าพัฒนา

นาย ไพฑูรย์ ทองจันทร์



หนังสืออ้างอิง

- [1] กมลมาส กำจรกิจการ, “คู่มือพัฒนาโปรแกรมด้วย Delphi 4”, บริษัทโปรวิชันจำกัด, พิมพ์ครั้งที่ 1, 2542
- [2] กฤษฎา ใจเย็น, “เรียนรู้และปฏิบัติการเชื่อมต่อคอมพิวเตอร์กับอุปกรณ์ภายนอกผ่านพอร์ตอนุกรม”, บริษัท อินโนเวทีฟอิเล็กทรอนิกส์เพอร์ซิเมนต์จำกัด
- [3] ชนนวนรณ ห้าวหาญ, “แผงตัวอักษรวิ่ง”, เซมิคอนดักเตอร์อิเล็กทรอนิกส์, ฉบับที่ 121, ตุลาคม, หน้า 10 – 18, 2535
- [4] สมยศ จุณณะปิยะ. “การใช้งานไมโครคอนโทรลเลอร์ตระกูล MCS-51”, คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง, กทม, พิมพ์ครั้งที่ 3, 2543

