

การต่อภาพจากเศษภาพ

JIGSAW COMPLETING MACHINE



โดย

นางสาวกัญช์

อมตวรกุล

นายพิพัฒน์

ทรงเกียรติศักดิ์

นางสาวสมฤดี

แช่เล่า

นายสันหกิจ

โสมณอุดมศักดิ์

พ.พ.

7/3/2543

2543

เลขหมู่.....

เลขทะเบียน.....42337

วัน, เดือน, ปี 7 พ.ค. 2545

b.....

i.....

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมระบบควบคุม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2543

ปริญญานิพนธ์ปีการศึกษา 2543

ภาควิชาวิศวกรรมระบบควบคุม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง การต่อภาพจากเสขภาพ

ผู้จัดทำ

- | | |
|----------------|------------------|
| 1. นางสาวกัญช์ | อมตวรกุล |
| 2. นายพิพัฒน์ | ทรงเกียรติศักดิ์ |
| 3. นางสาวสมฤดี | แซ่เล่า |
| 4. นายสันหกิจ | โสภณอุดมศักดิ์ |

..... อาจารย์ที่ปรึกษา
(รศ.ดร.โยธิน เปรมปรางค์รัตน์)

..... อาจารย์ที่ปรึกษา
(อ.สุมิตร พนาอุดมทรัพย์)

การต่อภาพจากเศษภาพ

นางสาวกัญช์ อมตวรกุล
นายพิพัฒน์ ทรงเกียรติศักดิ์
นางสาวสมฤดี แซ่เล่า
นายสันหกิจ โสภณอุดมศักดิ์

รศ.ดร.โยธิน เปรมปราชญ์ อาจารย์ที่ปรึกษา
อ.สุมิตร พนาอุดมทรัพย์ อาจารย์ที่ปรึกษา
ปีการศึกษา 2543

บทคัดย่อ

เนื่องจากในปัจจุบันการนำเอาความรู้ด้านการควบคุมไปใช้กับงานหลายประเภท เนื่องจากงานด้านการคอนโทรล (control) เป็นงานที่สามารถใช้ร่วมกับซอฟต์แวร์ (software) และ ฮาร์ดแวร์ (hardware) ในหลายส่วน เช่น งานควบคุมโรบอต (robot) โดยมีซอฟต์แวร์ เข้ามาใช้ในการประมวลผล ในโครงการปริญญานิพนธ์ฉบับนี้ จะเป็นการนำเสนอผลงานที่ได้พัฒนาความรู้ด้านการควบคุมมาใช้ร่วมกับงานด้านซอฟต์แวร์ ทางด้านฐานข้อมูลและการจัดเก็บข้อมูล และทำการควบคุมฮาร์ดแวร์ โดยใช้ไมโครคอนโทรลเลอร์ (Microcontroller) เป็นตัวส่งผ่านสัญญาณควบคุม คือ “การต่อภาพจากเศษภาพ” โดยการนำข้อมูลของเศษภาพแต่ละชิ้นจากกล้องวงจรปิด นำมาประมวลผลและส่งสัญญาณไปควบคุมแกนแกนให้ทำการต่อเศษภาพให้สมบูรณ์

Abstract

Presently, The knowledge of control mostly used in variety field. Because, it's able to work with the part of software and hardware such as robot controller by using software in processing. This thesis provides the development of controlling and using software in database system and controlling hardware by Microcontroller that transfers the signal. The Jigsaw Completing Machine works by getting data of picture from VDO camera to be processed by software (Delphi), the send the signal to control machine to complete the picture.

สารบัญ

	หน้า
บทคัดย่อ	I
สารบัญ	III
สารบัญภาพ	IV
สารบัญตาราง	VI
บทที่1 บทนำ	1
บทที่2 ทฤษฎีและหลักการ	2
2.1ทฤษฎีและการหักเหของสัญญาณภาพ	2
2.2โปรแกรมเตลไฟล์	7
2.3ไมโครคอนโทรลเลอร์และการส่งข้อมูลผ่านพอร์ตอนุกรม	13
2.4ชนิดของแขนกลและส่วนประกอบต่างๆของฮาร์ดแวร์	26
บทที่3 การคำนวณและการสร้าง	36
3.1การสร้างและการออกแบบ	36
3.2การคำนวณ	42
บทที่4 การทดลองและผลการทดลอง	48
4.1การทดลองและผลการทดลองในส่วนของซอฟต์แวร์	48
4.2การทดลองและผลการทดลองในส่วนของฮาร์ดแวร์	49
บทที่5 บทวิจารณ์และสรุป	51
กิตติกรรมประกาศ	52
หนังสืออ้างอิง	53
ภาคผนวก	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

	หน้า
รูปที่ 2.1 ภาพการสแกนเส้นภาพ	2
2.2 ภาพแสดงลักษณะสัญญาณภาพ	4
2.3 ภาพแสดงการลดความยุ่งยากในการเขียนโปรแกรมเซลล์ไฟล์	8
2.4 ภาพแสดงการตรวจสอบข้อผิดพลาดและการแก้ไข ของโปรแกรมเซลล์ไฟล์	9
2.5 ภาพองค์ประกอบการทำงานของเซลล์ไฟล์	10
2.6 ภาพ Object Inspector	10
2.7 ภาพ Code Editor และ Code Explorer	11
2.8 ภาพกลุ่มของ Component Palette	12
2.9 ภาพกลุ่มของ Standard Component	12
2.10 ภาพแสดงโครงสร้างภายในของไมโครคอนโทรลเลอร์MCS-51	14
2.11 ภาพบล็อกแสดงหน่วยการทำงานพื้นฐานของMCS-51	14
2.12 (ก) ภาพโครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์	15
(ข) ภาพแสดงสัญญาณควบคุมและสัญญาณข้อมูล	15
2.13 ภาพแสดงสถานะใน1รอบคำสั่ง	16
2.14 ภาพแสดงเวิร์คข้อมูลขนาด8บิต กับบิตสตาร์ทและบิตสตอป ที่ใช้ในการโอนย้ายข้อมูลแบบอนุกรม	21
2.15 ภาพบล็อกไดอะแกรมของUART	22
2.16 ภาพสัญญาณค่าตัวอินพุตโทรบ	22
2.17 บิตต่างๆในรีจิสเตอร์สถานะของ UART	23
2.18 รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม	24
2.19 ภาพแกนแขนชนิด Cartesisn robot	26
2.20 ภาพแกนแขนชนิด Cylindrical robot	27
2.21 ภาพแกนแขนชนิด Spherical robot	27
2.22 ภาพแกนแขนชนิด SCARA robot	27
2.23 ภาพแกนแขนชนิด Articulated robot	28
2.24 วงจรเรกยูเลเตอร์พื้นฐาน โดยใช้ซีเนอรี่ไดโอด	28
2.25 วงจรเรกยูเลเตอร์แบบขนาน โดยใช้ทรานซิสเตอร์	30

2.26 วงจรเรกจูเลเตอร์แบบอนุกรมโดยใช้ตัวต้านทานปรับค่าได้	30
2.27 ภาพแสดงรูปแบบโฟโต้เซลล์	30
2.28 ภาพแสดงสัญลักษณ์โฟโต้เซลล์	31
2.29 ภาพโครงสร้างทางเทคนิคของโซลาร์เซลล์	31
2.30 ภาพสัญลักษณ์โซลาร์เซลล์	31
2.31 ภาพแสดงพี-เอ็น จังก์ชันโฟโต้ไดโอด	32
2.32 ภาพโครงสร้างภายนอกของโฟโต้ไดโอด	32
2.33 ภาพแสดงสัญลักษณ์ของโฟโต้ไดโอด	32
2.34 ภาพวงจรใช้งานของโฟโต้ไดโอด	33
2.35 ภาพการทำงานเบื้องต้นของมอเตอร์	34
2.36 ภาพวงจรขับมอเตอร์	35
3.1 ภาพแสดงบล็อกไดอะแกรมของไดโอด L293D	38
3.2 ภาพConection Diagram ของวงจร Coparator	38
3.3 ภาพSensor Opto	39
4.1 ภาพการใส่ออบเจกต์เพื่อเพิ่มความถี่	49

สารบัญตาราง

หน้า

ตารางที่ 2.1 ตัวเชื่อมต่อที่นิยมใช้กับสายส่งสัญญาณอนุกรม

แบบมาตรฐาน RS-232

19

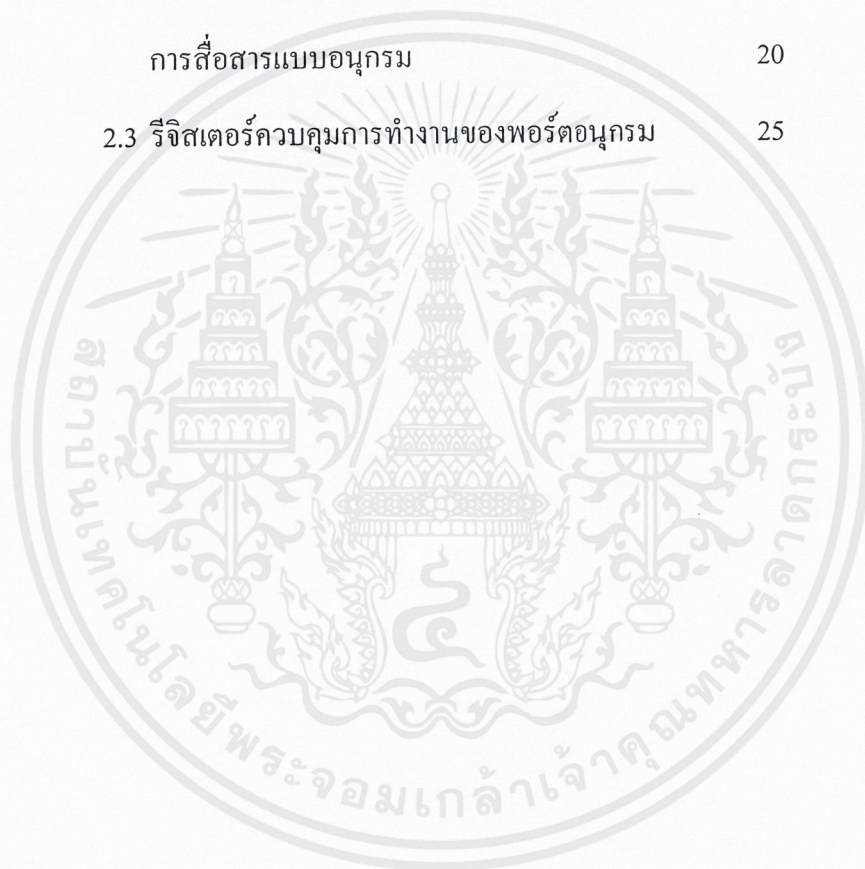
2.2 อัตราบอดและช่วงเวลาของแต่ละบิตข้อมูลใน

การสื่อสารแบบอนุกรม

20

2.3 รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม

25



บทที่ 1

บทนำ

ในปัจจุบัน การนำเอาความรู้ด้านสัญญาณภาพหรือทางด้านอิมเมจโปรเซสเซอร์ (Image processing) มาใช้ร่วมกับงานทางด้านการควบคุมด้านฮาร์ดแวร์ (Hardware) มีเพิ่มมากขึ้นในโรงงานอุตสาหกรรม หรืออาจเรียกได้ว่าอยู่ในช่วงการพัฒนาให้เกิดประสิทธิภาพมากที่สุดกับงานในส่วนต่างๆ โดยเฉพาะงานที่เป็นอันตรายหรืองานที่ต้องการความละเอียดสูง

สำหรับโครงการเรื่อง เครื่องต่อเศษภาพ ได้ทำการจัดแบ่งเป็น 2 ส่วน คือ ส่วนของซอฟต์แวร์ (Software) และ ฮาร์ดแวร์ โดยแต่ละส่วนมีขอบเขตของการทำงาน ดังนี้ ส่วนของซอฟต์แวร์ เป็นส่วนที่ใช้ในการเก็บข้อมูลภาพใช้ในการประมวลผลของข้อมูลภาพว่า เศษภาพชิ้นใดควรต่อกับชิ้นใดเพื่อให้ประกอบเป็นภาพที่สมบูรณ์ นอกจากนั้นในส่วนซอฟต์แวร์ ยังต้องทำการติดต่อกับ ไมโครคอนโทรลเลอร์ (Microcontroller) เพื่อส่งสัญญาณไปควบคุมแกนแขนในการหยิบและต่อชิ้นภาพ ดังนั้นส่วนของซอฟต์แวร์ สามารถแบ่งได้เป็น 2 ส่วน คือ ส่วนของการเก็บข้อมูลภาพ และส่วนของการประมวลผล โดยใช้โปรแกรมเดลไฟล์ 5.0 (Program Delphi 5.0) ซึ่งเป็นซอฟต์แวร์ ที่ทำงานเกี่ยวกับฐานข้อมูล และส่วนของการติดต่อกับไมโครคอนโทรลเลอร์ ในการควบคุมแกนแขนซึ่งใช้ภาษาแอสเซมบลี (Assembly) ในการเขียนโปรแกรม

ในส่วนของฮาร์ดแวร์ จะเป็นการศึกษาในส่วนของกล่อง แกนแขนในการหยิบชิ้นเศษภาพ และวงจรต่างๆที่จำเป็นเช่น วงจรขับเคลื่อนมอเตอร์

โครงการนี้แม้จะยังไม่สามารถนำไปใช้ในงานด้านอุตสาหกรรมได้จริง แต่ก็พื้นฐานในการพัฒนาและค้นคว้าต่อไป ซึ่งในอนาคตอาจจะปรับปรุงจนสามารถนำไปใช้ทดแทนเครื่องจักรบางชนิดหรือทดแทนแรงงานในงานด้านต่างๆก็ได้

บทที่ 2

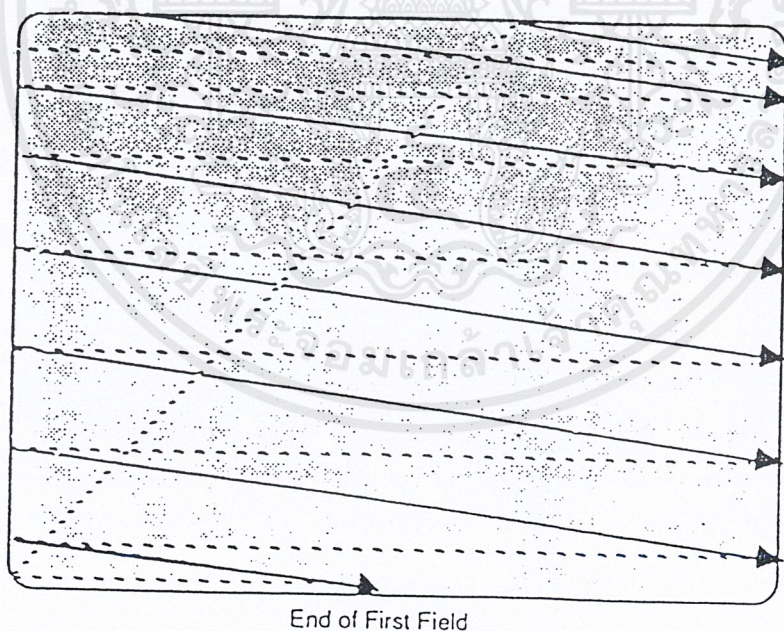
ทฤษฎีและหลักการ

2.1 ทฤษฎีและหลักการของสัญญาณภาพ

2.1.1 การแสดงภาพบนหน้าจอ

โดยปกติการแสดงภาพในหน้าจอหนึ่ง จะประกอบไปด้วยการแสดงผลภาพจำนวน 625 เส้น โดยจะมีการไล่สแกนจากเส้นที่ 1 ไปยังเส้นสุดท้ายด้วยความเร็วสูงจนเกิดเป็นภาพขึ้น แต่การสแกนแบบนี้ต้องอาศัยความเร็วสูงมาก และภาพอาจเกิดการกระพริบได้ จึงมีการใช้การสแกนแบบ

อินเตอร์ลอสแกน (Interlace scan) คือแบ่งภาพออกเป็น 2 ส่วน คือ field คู่ และ field คี่ และทำการสแกน 2 ครั้งต่อ 1 ภาพ ครั้งละ 312.5 เส้น การแสดงผลภาพเพื่อให้เกิดความต่อเนื่อง โดยปกติจะใช้ความถี่ 25 Hz คือ 25 ภาพต่อ 1 วินาที หรือมีจำนวน field 50 field เกิดขึ้นใน 1 วินาที



รูปที่ 2.1 แสดงการสแกนเส้นภาพ

2.1.2 สัญญาณวิดีโอ

เมื่อใช้กล้องวิดีโอจับภาพ กล้องจะส่งสัญญาณภาพออกมาอย่างต่อเนื่อง โดยสัญญาณภาพที่ส่งออกมาแบ่งเป็น 2 ชนิด คือ

สัญญาณอนาล็อกวิดีโอ

1. สัญญาณคอมโพสิตวิดีโอ - รวมสัญญาณซิงค์, ส่องสว่างและสี
2. สัญญาณ S – VHS หรือ Y/C - แยกสัญญาณส่องสว่างกับสีออกจากกัน
3. สัญญาณคอมโพเนนท์ y,u,v - แยกสัญญาณส่องสว่างกับความต่างสี
4. สัญญาณ RGB - แยกสัญญาณแม่สีทั้ง 3 ออกจากกัน

สัญญาณดิจิทัลวิดีโอ

1. สัญญาณคอมโพสิตดิจิทัลวิดีโอ

1.1 Parallel Composite digital ทั่วไปรู้จักกันสำหรับเทปฟอร์แมต D2 ระบบ NTSC ใช้มาตรฐาน SMPTE244M

1.2 Serial Composite digital จะใช้ชื่อมาตรฐาน SMPTE259M อัตราส่งข้อมูล 143 Mb/s ในระบบ NTSC และ 177 Mb/s ในระบบ PAL

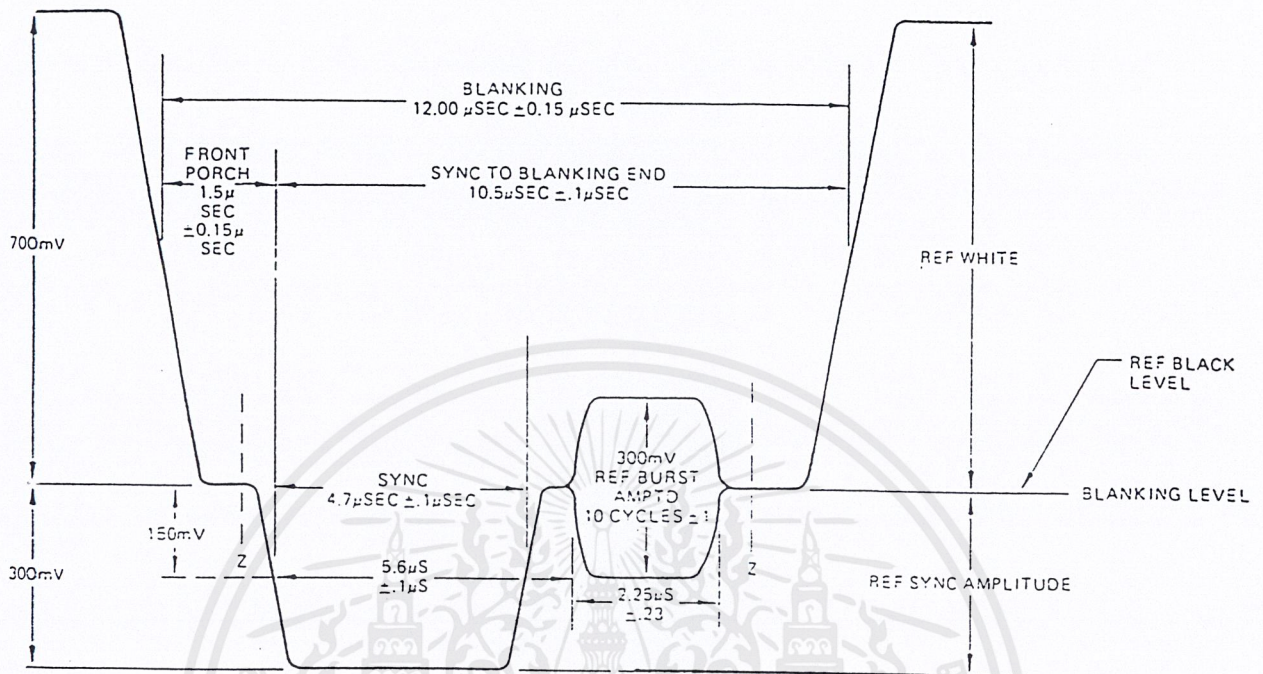
2. สัญญาณคอมโพเนนท์ดิจิทัลวิดีโอ

2.1 Parallel Component digital เป็นดิจิทัลอินพุตมาตรฐานของ NTL ENCODER และทั่วไปใช้ชื่อมาตรฐานว่า CCIR656

2.2 Serial Component digital หรือมาตรฐาน CCIR601 ใช้อัตราการส่งข้อมูล 270 Mb/s หรือเรียกว่าฟอร์แมต D1 เป็นมาตรฐานที่ดีที่สุดในปัจจุบัน

2.1.3 สัญญาณคอมโพสิตวิดีโอ

ประกอบด้วยสัญญาณภาพในระบบ PAL โดยสแกนแบบอินเตอร์เลซเพื่อลดการกระพริบของภาพ และสัญญาณซิงค์ ซึ่งประกอบด้วย Horizontal Sync, Vertical Sync, Burst (สัญญาณซิงค์สี)



รูปที่ 2.2 แสดงลักษณะสัญญาณภาพ

2.1.4 รูปแบบของไฟล์รูปภาพ

- 1 BMP(bitmap) เป็นรูปแบบไฟล์มาตรฐานที่ใช้กันทั่วไปกับwindow ,dos ไฟล์รูปแบบนี้รองรับโหมดสีแบบ RGB , index-color , grayscale และ bitmap แต่ไม่รองรับ Alpha channels สำหรับรูปภาพใน windows ที่มีการเก็บข้อมูล 4 bit และ 8 bit ต่อ 1 pixel file แบบนี้สนับสนุนการบีบอัดแบบ RLE อีกด้วย
- 2 DCS(desktop color separations) เป็นรูปแบบมาตรฐานของ EPS ซึ่งถูกพัฒนาโดย Quark file แบบ DCS 2.0 จะรองรับโหมด Multichannel, CMYK ที่มี single alpha channel และมี multiple spot channel ส่วน file แบบ DCS 1.0 จะรองรับโหมด CMYK ที่ไม่มี alpha channel ทั้ง file แบบ DCS1.0 และ DCS 2.0 รองรับการใช้ Clipping paths ทั้งคู่
- 3 TIFF(Tagged-image file format) ถูกใช้ในการแลกเปลี่ยนfileระหว่างprogram และแพลตฟอร์มของเครื่องรุ่นต่างๆ ไฟล์ TIFF รองรับโหมดสีแบบCMYK,RGB,Grayscaleซึ่งมีAlpha channels, และLab,indexed-color,Bitmapซึ่งไม่มี Alpha channels

4 GIF(Graphics interchange Format)เป็นไฟล์ที่ใช้กันมากในการแสดงภาพใน mode indexed Color และรูปภาพที่ไฟล์เอกสารแบบ HTML(hypertext markup language) ซึ่งใช้กันมากในเว็บไซด์และ บริการออนไลน์ต่างไฟล์แบบนี้จะใช้การวิธแบบบีบอัดแบบLZW ทำให้ไฟล์มีขนาดเล็กและสามารถส่งผ่านระบบเครือข่ายได้รวดเร็ว GIF รองรับโหมดสีแบบ RGB,Indexed-color,Grayscale

5 JPEG (Joint PhotographicExpert Group) เป็นไฟล์ที่ใช้กันมากเพื่อแสดงภาพแบบ Index-color และรูปภาพที่ไฟล์เอกสารแบบ HTML ซึ่งใช้กันมากในเว็บไซด์และบริการออนไลน์แบบต่างๆ ไฟล์แบบJPEGนี้รองรับโหมดสีแบบ CMYK,RGB,Grayscale-color แต่ไม่รองรับ Alpha channels ข้อแตกต่างระหว่าง GIF และ JPEG คือJPEG จะเก็บข้อมูลของไฟล์รูปภาพในโหมด RGB โดยการเลือกข้อมูลที่สำคัญและทิ้งข้อมูลที่ไม่จำเป็น

6 PDF(Portable Document Format) ถูกใช้โดยโปรแกรม Adobe Acrobat ซึ่งเป็น softwareที่ช่วยให้คุณสามารถอ่านเอกสารและคู่มือที่มีอยู่ในinternet ได้ง่ายขึ้น ไฟล์ PDF รองรับโหมดสีแบบ RGB , Indexed-color , CMYK , Grayscale , Bitmap , Lab แต่ไม่รองรับการใช้ Alpha Channels

7 PICT เป็นไฟล์ที่ใช้กันแพร่หลายกันในกราฟฟิกของเครื่องแมคอินทอชและโปรแกรมPagemaker ไฟล์PICT รองรับโหมดสีแบบIndexed-colors ,Grayscale,Bitmapที่ไม่มี Alpha Channels และRGBที่มีsingle Alpha channels

2.1.5 Image processing

1 Shape analysis

เมื่อเก็บภาพเป็น pixel โดยเก็บค่าเป็น binary คือเลข 0 และ 1 โดยมีขนาด 192*144 pixel ในการวิเคราะห์ขอบภาพได้นำทฤษฎี chain code มาทำการวิเคราะห์ขอบภาพโดยจะพิจารณาคำแทนของเลข 1 ที่สัมพันธ์กับตำแหน่งข้างเคียงว่าเป็นเท่าใด โดยเริ่มไล่จากเลข 0 ในทิศทางตามเข็มนาฬิกา ดังรูปที่

1	2	3
0	P	4
7	6	5

รูปที่ 2.3 ทฤษฎี Chain code

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

และจะแสดงค่าขอบภาพอยู่ในรูปของ chain code ตัวอย่างเช่น

0	0	0	1	0	0	0
0	0	1	1	1	0	0
0	1	1	1	1	0	0
0	1	1	1	1	0	0
0	0	1	1	1	0	0
0	0	0	0	0	0	0
0	0	0	0	0	0	0

เลข 1 คือ ภาพเมื่อแปลงเป็น pixel แล้ว พิจารณาเลข 1 ตัวบน ซ้ายสุด ทำการวนรอบจุดนี้โดยเริ่มจาก 0 จะพบว่าเลข 1 ตัวแรกที่พบมีความสัมพันธ์เป็นเลข 3 ตามรูปที่ และทำดังนี้ต่อไปจนวนมาถึง 1 ตัวแรก โดยพิจารณา เลข 1 ที่ ขอบภาพเท่านั้น

$$a = [3\ 3\ 5\ 6\ 6\ 6\ 0\ 0\ 1\ 2]^T$$

จะเห็นได้ว่า โดยหลักการ chain code นี้ จะสามารถวิเคราะห์หาขอบภาพในแต่ละด้านได้และนำไปประยุกต์ในการต่อภาพต่อไป

2 Area descriptor

ให้ (x, y) เป็น แถว และ คอลัมน์ ของ pixel ในขอบเขตของ R โดยหา moment ของ R จาก

$$m_{kj} = \sum_{(x,y) \in R} x^k y^j \quad k \geq 0, j \geq 0$$

จะมี centroid ของ R ที่ X_c, Y_c โดยหาจาก

$$A = m_{00} \quad X_c = m_{10} / m_{00} \quad Y_c = m_{01} / m_{00}$$

Zeroth-order moment (m_{00}) จะเป็นขนาดพื้นที่ของขอบเขต R

First-order moment $m_{10}/m_{00}, m_{01}/m_{00}$ ตำแหน่งของขอบเขต R

Example

		1	y →							10
x ↓	1	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	1	1	1	0	0	0	0	0	0
	0	0	0	1	1	1	0	0	0	0
	0	0	0	0	0	1	1	1	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	0	0	0	0	0	0	0	0	0	0
	8	0	0	0	0	0	0	0	0	0

$$m_{00} = 9$$

$$m_{01} = 45$$

$$m_{10} = 36$$

$$A = m_{00} = 9$$

$$X_c = m_{10} / m_{00} = 4$$

$$Y_c = m_{01} / m_{00} = 5$$

จะเห็นได้ว่า ตำแหน่ง (4,5) เป็นจุด centroid ของภาพ

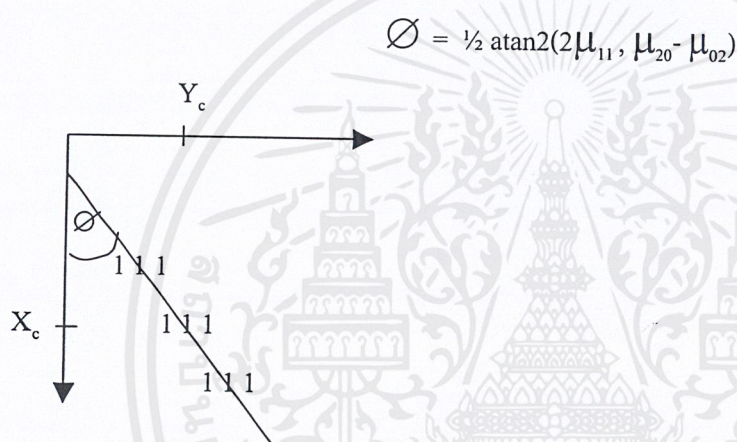
Central Moment

ให้ X_c, Y_c เป็นจุด centroid ของรูปภาพ central moment ของขอบเขต R คือ

$$\mu_{kj} = \sum_{(x,y) \in R} (X - X_c)^k (Y - Y_c)^j \quad k \geq 0, j \geq 0$$

3 Principle Angle

ในกรณีที่มีการหมุนของขอบเขต R จึงมีทฤษฎีบทขึ้นมาใช้ในการหามุมการหมุน เรียกว่า Principle Angle

$$\theta = \frac{1}{2} \text{atan2}(2\mu_{11}, \mu_{20} - \mu_{02})$$


ทั้ง Area descriptors , Central moment และ Principle Angle นำไปประยุกต์ในการจับตัวชิ้นงานในการต่อและการหมุนมุมให้เหมาะสมในการต่อภาพโดยผนการจับตัวภาพต้องจับที่จุด Centroid (X_c, Y_c)

2.2 โปรแกรมเดลไฟล์

เนื่องจากเดลไฟล์นั้น ได้ชื่อว่าเป็นหนึ่งในชุดเครื่องมือสำหรับพัฒนา Application บน วินโดวส์(windows) ที่ได้รับการยอมรับอย่างสูง จากนัก Application ทั่วโลก ง่ายต่อการ Interface มี ฟังก์ชันรองรับมากมาย อีกทั้งเดลไฟล์นั้น ใช้ภาษาปาสคาล(Pascal)ซึ่งได้ชื่อว่าเป็นภาษา คอมพิวเตอร์ที่มีรูปแบบของภาษาที่ดีที่สุดภาษาหนึ่ง และผนวกเอาชุดเครื่องมืออื่น ๆ ที่เกี่ยวข้องมาไว้ด้วยกัน ทำให้เดลไฟล์สร้าง Application มีขนาดเล็ก ทำงานรวดเร็ว นำใช้งาน และเมื่อมองถึง โครงงานแขนงกลต่อเศษภาพนั้นซอฟต์แวร์ที่เขียนขึ้นนั้น ก็ต้องการให้ทำงานบน Windows เพื่อ สะดวกในการทำงานและพัฒนาต่อไป

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.1 จุดเด่นของ Delphi 5

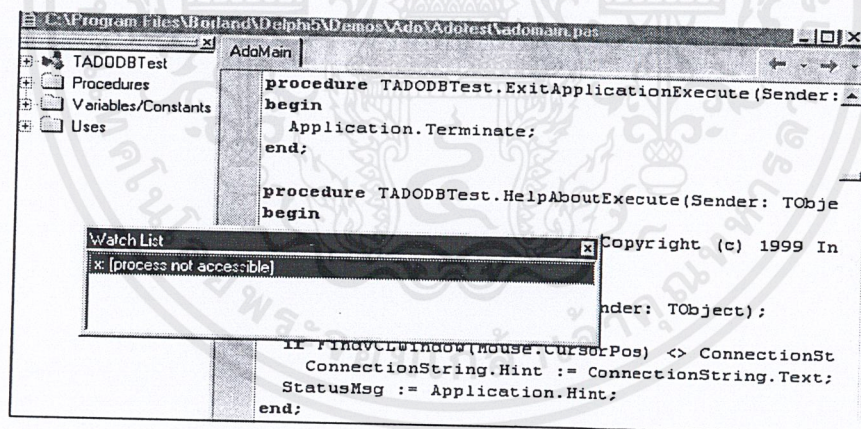
Delphi 5.0 มีความสามารถมาก โดยเฉพาะความสามารถด้านการสร้าง Application สำหรับ Internet อีกทั้งจุดเด่นของการปรับปรุงประสิทธิภาพของเครื่องมือเครื่องมือ ทำให้เป็นโปรแกรมที่น่าสนใจ

2.2.2 ความเร็วของ Application ที่สร้างจาก Delphi 5

Delphi นั้นได้ชื่อว่าเป็น คอมไพเลอร์โดยสมบูรณ์ในตัว ช่วยให้ Application ที่สร้างขึ้นเป็นไฟล์ .EXE ที่ใช้งานได้ทันที และใช้งานได้ในทุกเครื่อง โดยไม่จำเป็นต้องมีไฟล์พิเศษเพิ่มเติมเลย ทำให้ Application ที่สร้างจาก Delphi มีขนาดเล็ก กินทรัพยากรของระบบน้อย จึงทำงานได้รวดเร็วมาก

2.2.3 ความสามารถในการเขียนโปรแกรม

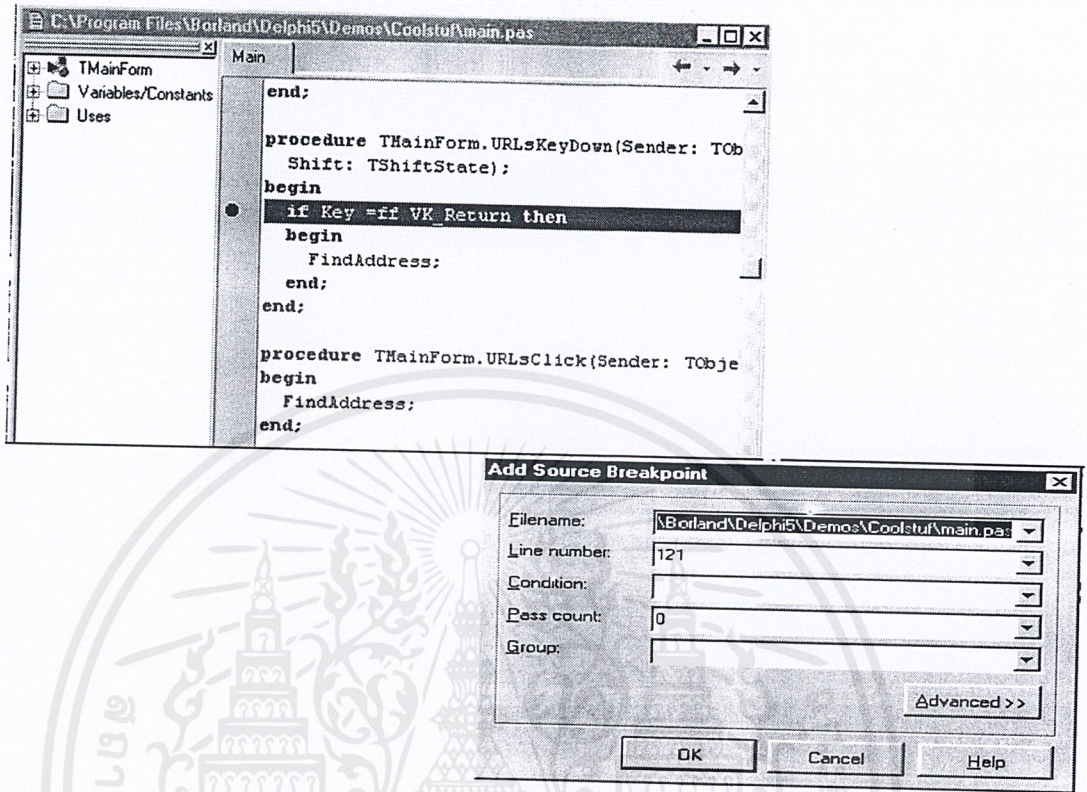
การเขียนโปรแกรมนั้นเป็นเรื่องของความมีระบบระเบียบ โดยธรรมชาติอยู่แล้ว แต่เมื่อความซับซ้อนของโปรแกรมมีมากขึ้นทำให้การเขียนโปรแกรมเป็นเรื่องยุ่งยากอย่างหลีกเลี่ยงไม่ได้ ทำให้มีหลายๆ วิธีที่คิดขึ้นมาเพื่อลดความซับซ้อนลงซึ่ง Delphi ก็มีหลากหลายวิธีที่ช่วยให้เราเขียน และทดสอบโปรแกรมที่เขียนได้อย่างสะดวกสบาย



รูปที่ 2.4 แสดงการลดความยุ่งยากในการเขียนโปรแกรมใน Delphi 5.0

2.2.4 ความสามารถในการตรวจสอบโปรแกรม

เมื่อความซับซ้อนของโปรแกรมมีมากขึ้น ความผิดพลาดย่อมเพิ่มมากขึ้นตามตัว ทำให้เราต้องมีวิธีการที่ช่วยให้การเขียนโปรแกรมง่ายขึ้นโดยการเอาข้อผิดพลาดออกไปให้มากที่สุด เพื่อการทำงานที่ถูกต้องของแอปพลิเคชัน ซึ่ง Delphi 5.0 มีความสามารถนี้อย่างเต็มเปี่ยม

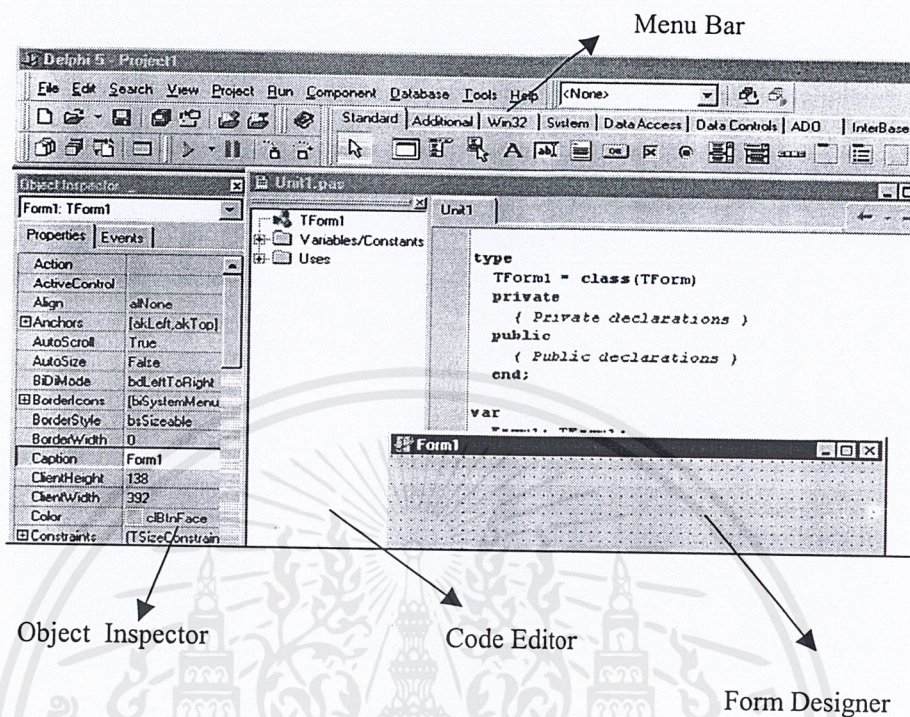


รูปที่ 2.5 การตรวจสอบข้อผิดพลาดและการแก้ไข

โปรแกรมใน Delphi 5.0

2.2.5 ส่วนประกอบการทำงานของ Delphi

การทำงานเพื่อสร้างแอปพลิเคชันกับ Delphi 5.0 นั้นเราจะอยู่ภายใต้สภาวะแวดล้อมการพัฒนาแบบครบครัน นั่น คือ ประกอบด้วยเครื่องมือชนิดต่างๆ ที่นำมาสร้าง และทดสอบแอปพลิเคชัน

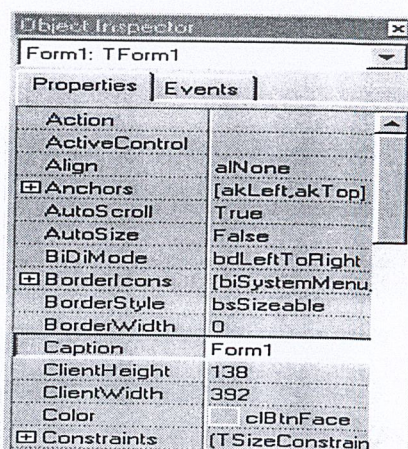


รูปที่ 2.6 องค์ประกอบการทำงานของ Delphi 5

Menu Bar เป็นส่วนที่เก็บคำสั่งเพื่อเรียกใช้งาน Delphi 5.0 ในรูปแบบ ซึ่งอยู่บนสุดของการแสดงผล

Form Designer เป็นส่วนที่เราออกแบบหน้าตาของฟอร์ม ซึ่งถือเป็นพื้นที่แสดงผลหลัก ที่เราสามารถนำเอาออบเจกต์ต่างๆมาวางไว้ภายในให้เป็นแอปพลิเคชันตามที่เรต้องการได้

Object Inspector เราสามารถกำหนดค่าพร็อพเพอร์ตี้ของออบเจกต์ต่างๆ ผ่านทาง Object Inspector นอกจากนี้เรายังเริ่มต้นจัดการกับอีเวนต์ต่างๆ ของออบเจกต์ผ่านทาง Object Inspector เช่นเดียวกัน



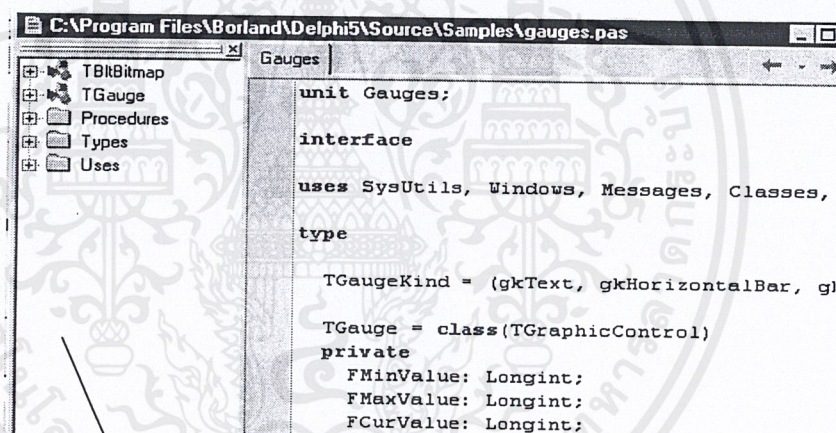
รูปที่ 2.7 Object Inspector

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Component Palette คือส่วนที่เก็บคอมโพเนนต์ต่างๆ แยกเป็นหมวดหมู่เอาไว้ พร้อมให้เรานำไปประกอบในแอปพลิเคชันที่จะสร้างขึ้น และยังสามารถสร้างคอมโพเนนต์มาใช้งานเองได้ บรรจุไว้ใน Component Palette ได้ด้วย

Code Editor เป็นส่วนที่เราใช้ในการเขียนคำสั่งต่างๆ เข้าไปเพื่อกำกับการทำงานของแอปพลิเคชันให้สอดคล้องกันซึ่งคำสั่งต่างๆ ในภาษาปาสคาลที่อยู่ใน Code Editor จะประกอบไปด้วยส่วนที่ Delphi เตรียมไว้ให้ และส่วนที่เราต้องเขียนเพิ่มเติมเข้าไป

ใน Code Editor เรามักจะเห็น Code Explorer ควบคู่ไปด้วย ซึ่ง Code Explorer จะช่วยให้เราเห็นตัวแปร , โปรแกรมย่อย , ออบเจกต์ , พร็อพเพอร์ตี้ เมธอดของออบเจกต์ต่างๆ ที่ประกอบกันเป็นแอปพลิเคชัน



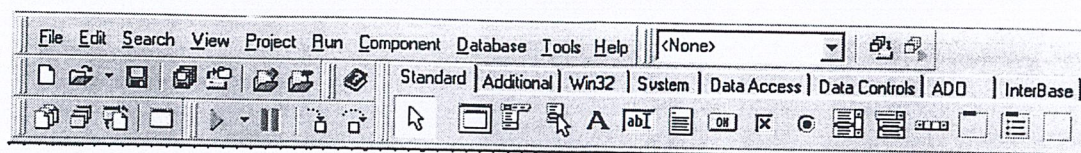
Code Explorer

เขียนโปรแกรม Code Editor

รูปที่ 2.8 Code Editor และ Code Explorer

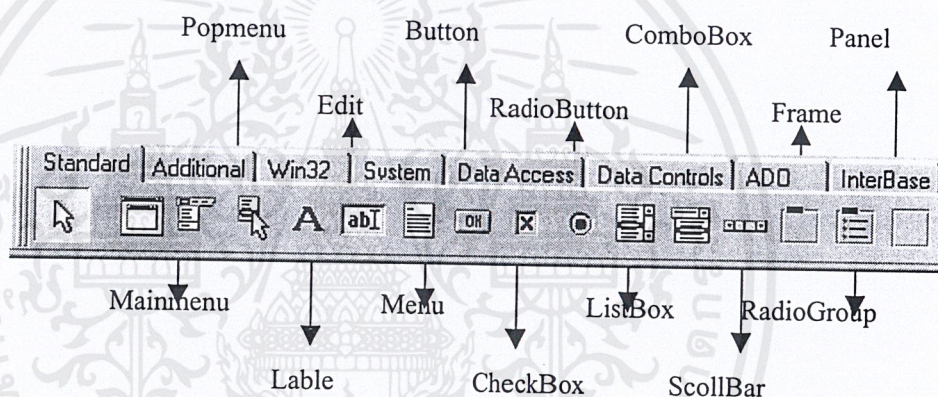
2.2.6 การใช้งานคอมโพเนนต์พื้นฐานของ Delphi

คอมโพเนนต์ต่างๆ ถูกเก็บไว้ใน Component Palette และมีการแยกคอมโพเนนต์เหล่านั้นออกเป็น 19 กลุ่มตามลักษณะการเลือกใช้



รูปที่2.9 Component Palette

กลุ่มของคอมโพเนนต์ที่ใช้มากในโครงงานนี้จะเป็นกลุ่มของStandard ซึ่งเป็นกลุ่มของคอมโพเนนต์ที่มีรูปร่างหน้าตาเหมือนกับการใช้งานทั่วไปของWindows



รูปที่2.10 กลุ่มของStandard Component

คอมโพเนนต์Label จะทำหน้าที่แสดงข้อความตามที่เรต้องการ ซึ่งข้อความดังกล่าวมักจะเป็นข้อความที่ไม่มีการเปลี่ยนแปลงแล้ว

คอมโพเนนต์Button เป็นคอมโพเนนต์ที่มีการนำมาใช้มากที่สุด ทำหน้าที่รับคำสั่งจากผู้ใช้งานในลักษณะการคลิกที่ปุ่มนั้น

คอมโพเนนต์Edit ทำหน้าที่รับข้อความจากผู้ใช้งานที่ป้อนเข้ามา นอกจากนี้ยังใช้แสดงข้อความได้อีกด้วย

คอมโพเนนต์Memo มีความสามารถเช่นเดียวกับEdit แต่แตกต่างกันที่Memo สามารถแสดงข้อความได้มากกว่า 1บรรทัด

คอมโพเนนต์CheckBox มีไว้เพื่อสร้างตัวเลือกในแอปพลิเคชัน ซึ่งตัวเลือกนี้สามารถเลือกได้พร้อมๆกันหลายตัวเลือก

เอกสารนี้เป็นเอกสารลิขสิทธิ์ของมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
คอมโพเนนต์RadioButton มีหน้าที่เหมือนCheckBox แต่สามารถเลือกได้ตัวเดียวเท่านั้น
 ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

คอมโพเนนต์ListBox เป็นอีกวิธีที่ใช้ในการให้ผู้เลือกใช้เลือกตัวเลือกจากที่มีให้ โดยภายใน ListBox จะมีตัวเลือกต่างๆ ให้เลือก ถ้ามีจำนวนมากก็จะแสดง ScrollBox ให้เราเลื่อนขึ้นลงเพื่อเลือกได้

คอมโพเนนต์ComboBox เป็นรายการตัวเลือกได้เหมือนกับ ListBox แต่มีข้อพิเศษกว่า คือ นอกจากจะเลือกจากตัวเลือกในลักษณะ DropDown เช่นเดียวกับ ListBox แล้วยังสามารถเลือกโดยการพิมพ์ข้อความเข้าไปในลักษณะเดียวกับคอมโพเนนต์ Edit ก็คือ ComboBox เป็นเหมือนการผสมกันระหว่างคอมโพเนนต์ Edit กับ คอมโพเนนต์ ListBox

คอมโพเนนต์ScrollBar เป็นแถบเลื่อน(ซึ่งเราใช้กันบ่อยใน Windows) ซึ่ง ScrollBar จะเลื่อนตำแหน่งภายใน Windows

คอมโพเนนต์GroupBox เป็นคอมโพเนนต์ที่ใช้ในการจัดกลุ่มคอมโพเนนต์ที่วางอยู่บนฟอร์ม โดยเราจะเลือกคอมโพเนนต์ที่สัมพันธ์กันมาวางไว้ใน GroupBox เดียวกัน โดยที่ GroupBox จะมคุณสมบัติเป็นเหมือน Container คือสามารถสามารถบรรจุเอาคอมโพเนนต์ต่างๆ ไว้ภายในซึ่งจะไม่เกี่ยวข้องกับคอมโพเนนต์อื่นใด ที่อยู่นอก Container

คอมโพเนนต์RadioGroup ทำหน้าที่คล้าย GroupBox แต่มีความพิเศษคือใช้แยกกลุ่มของ RadioButton เข้าด้วยกัน ซึ่งจะให้ผลเหมือนกับการทำงานของ GroupBox แต่มีวิธีการที่ใช้แตกต่างกัน

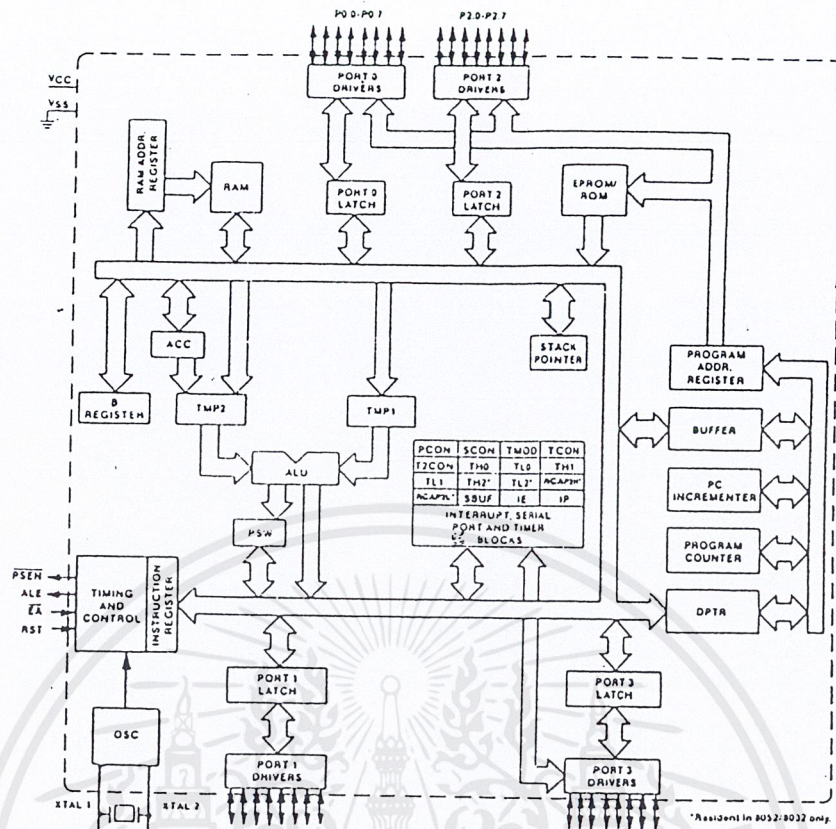
คอมโพเนนต์Panel เป็นคอมโพเนนต์อีกประเภทที่มีคุณสมบัติเป็น Container คือสามารถบรรจุเอาคอมโพเนนต์อื่นๆ เข้าไปได้ แต่จะช่วยให้รูปร่างหน้าต่างที่สวยงามไปอีกแบบ

2.3 ไมโครคอนโทรลเลอร์ และการส่งข้อมูลผ่านพอร์ตอนุกรม

2.3.1 ไมโครคอนโทรลเลอร์ 8051

1) โครงสร้างภายในของ 8051

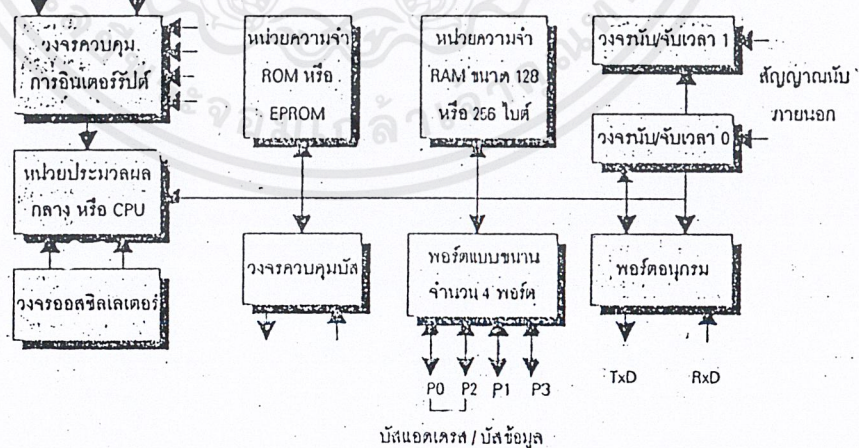
ภายใน 8051 ประกอบด้วยเกทต่าง ๆ ซึ่งจะถูกนำมาออกแบบให้มีหน้าที่ในการทำงานต่าง ๆ เช่น วงจรถอดรหัสคำสั่ง, วงจรสร้างสัญญาณนาฬิกา โครงสร้างภายในของ 8051 จะประกอบด้วยส่วนย่อย ๆ ดังไดอะแกรมในรูปที่ 2.11



รูปที่ 2.11 แสดงโครงสร้างภายในของไมโครคอนโทรลเลอร์ MCS-51

2) คุณลักษณะพื้นฐานของ 8051

สัญญาณอินพุต/เอาต์พุตภายนอก



รูปที่ 2.12 แสดงภาพบล็อกแสดงหน่วยการทำงานพื้นฐานของ MCS-51

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

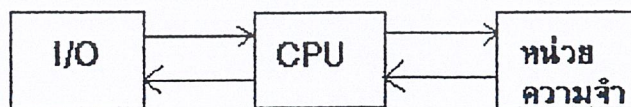
หน่วยการทำงานพื้นฐานของไมโครคอนโทรลเลอร์เบอร์ต่าง ๆ ที่จัดอยู่ภายใน
ตระกูล 8051 นี้

ประกอบด้วย

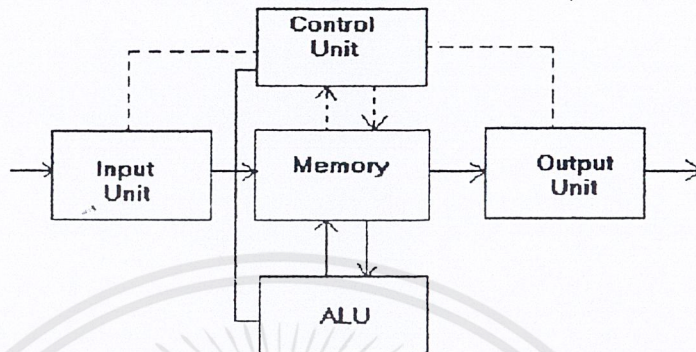
- 1) หน่วยประมวลผลกลางขนาด 8 บิต
- 2) หน่วยประมวลผลสำหรับข้อมูลแบบบิต (Boolean Processor)
- 3) ความสามารถในการอ้างตำแหน่งของหน่วยความจำโปรแกรม 64 กิโลไบต์
- 4) ความสามารถในการอ้างตำแหน่งของหน่วยความจำข้อมูล 64 กิโลไบต์
- 5) หน่วยความจำโปรแกรมภายในขนาด 4 กิโลไบต์ แบบ EPROM (เบอร์ 8751) หรือแบบ ROM (เบอร์ 8051)
- 6) หน่วยความจำแบบ RAM ภายในจำนวน 128 ไบต์
- 7) พอร์ตอินพุต / เอาต์พุต แบบขนานจำนวน 32 เส้น ซึ่งสามารถแยกการทำงานได้อย่างอิสระ
- 8) วงจรรนับ / จับเวลาขนาด 16 บิต จำนวนสองวงจร
- 9) วงจรสื่อสารแบบอนุกรมแบบฟูลดูเพล็กซ์ (Full Duplex)
- 10) วงจรควบคุมการอินเตอร์รัปต์จากแหล่งกำเนิดสัญญาณ 6 ประเภท พร้อมการกำหนดลำดับความสำคัญได้สองระดับ
- 11) วงจรออสซิลเลเตอร์ภายใน

3) ส่วนประกอบของไมโครคอนโทรลเลอร์ 8051

8051 ประกอบด้วย 3 ส่วนหลัก คือ



(ก)



(ข)

รูปที่ 2.13(ก) โครงสร้างพื้นฐานของไมโครคอมพิวเตอร์

(ข) เส้นประแสดงสัญญาณควบคุมและสัญญาณเลือก
เส้นทึบแสดงสัญญาณข้อมูล

1 ตัวประมวลผล (CPU-Central Processing Unit)

วงจรส่วนนี้ทำหน้าที่สร้างสัญญาณควบคุมในการติดต่อกับส่วนอื่น ๆ เรียกว่า วงจรควบคุม(Control Unit) สัญญาณที่สร้างจากวงจรควบคุม ได้แก่ สัญญาณสำหรับการติดต่อกับหน่วยความจำ อุปกรณ์รับข้อมูลเข้าหรือส่งข้อมูลออกจากตัว 8051 ซึ่งส่วนควบคุมการขัดจังหวะ (Interrupt Control) และส่วนควบคุมบัส (Bus Control) ก็เป็นส่วนหนึ่งของวงจรควบคุมด้วยการสร้างสัญญาณควบคุมจาก ส่วนซีพียูนี้ จะทำการสร้างสัญญาณโดยการถอดรหัสจากคำสั่งตามที่มีการกำหนดได้ และสัญญาณที่ สร้างขึ้นมาจะนำมาอ้างอิงกับสัญญาณนาฬิกา ที่สร้างจากวงจรออสซิลเลเตอร์เพื่อให้ทุก ๆ ส่วนในวงจรทำงานประสานกันอย่างถูกต้อง

ส่วน CPU แบ่งออกเป็น 2 ส่วน คือ หน่วยคำนวณ (Arithmetic unit , ALU) และหน่วยควบคุม (Control unit)

หน่วยคำนวณ จะทำหน้าที่ในการคำนวณทางคณิตศาสตร์และลอจิกโดยตรง ฟังก์ชันพื้นฐานของหน่วยคำนวณคือ บวก,ลบและฟังก์ชันทางลอจิก การบวก,ลบ ตัวเลขแต่ละครั้งจะได้ผลลัพธ์ซึ่งจะถูกเก็บไว้ในหน่วยความจำชั่วคราว การแสดงผลจากการคำนวณจะแสดงออกทาง บิทแฟล็ก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน่วยคำนวณ จะส่งสัญญาณควบคุมให้จังหวะแก่หน่วยอื่นๆ การทำงานของหน่วยควบคุมจะเริ่มจากการรับคำสั่งจากหน่วยความจำแล้วแปรคำสั่งนั้น และส่งสัญญาณต่างๆ ไปควบคุมหน่วยอื่นให้ทำงานตามคำสั่ง

การกระทำตั้งแต่หน่วยควบคุมเริ่มไปอ่านคำสั่งจากหน่วยความจำ นำมาถอดรหัส จนนำคำสั่งส่งลงไป เราเรียกว่าหนึ่งรอบคำสั่ง แต่ละรอบคำสั่งประกอบด้วยขั้นตอนย่อยๆ คือ สภาวะเฟตช์ (ขบวนการที่หน่วยควบคุมไปนำคำสั่งมาจากหน่วยความจำแล้วถอดรหัสคำสั่งนั้น) และ สภาวะเอ็กซ์คิว (ขบวนการตั้งแต่หน่วยควบคุมเริ่มทำตามคำสั่งนั้นจนเสร็จคำสั่ง)



รูปที่ 2.14 สภาวะใน 1 รอบคำสั่ง

2 หน่วยความจำ (Memory)

ในระบบของไมโครคอนโทรลเลอร์ 8051 จำเป็นต้องมีหน่วยความจำซึ่งประกอบด้วย

1) หน่วยความจำโปรแกรม

หน่วยความจำโปรแกรมของ 8051 เป็นบริเวณหน่วยความจำ สำหรับเก็บข้อมูลและคำสั่งใช้งานต่างๆ ซึ่งแม้ว่าจะไม่มีการจ่ายกระแสไฟฟ้าให้กับระบบ ข้อมูลเหล่านี้ก็ยังคงไม่สูญหาย

2) หน่วยความจำข้อมูล

หน่วยความจำข้อมูลมีหน้าที่สำหรับเก็บข้อมูล หรือตัวแปรที่เกิดขึ้นในขณะที่กำลังประมวลผลโปรแกรมไว้เป็นการชั่วคราว โดยพื้นฐานแล้วหน่วยความจำข้อมูล จัดเป็นหน่วยความจำ RAM แบบสแตติก ดังนั้นเมื่อไม่มีการจ่ายไฟให้กับระบบ ก็จะมีผลทำให้ข้อมูลที่จัดเก็บไว้ภายในหน่วยความจำนี้สูญหายไป ข้อมูลที่เก็บในหน่วยความจำจะถูกเก็บในรูปของเลขฐานสองหรือเลขไบนารี เลขไบนารี 1 หลักเราเรียกว่า “บิต” เลขไบนารีหลายๆบิตที่คอมพิวเตอร์ถือเป็นคำสั่งเราเรียกว่า “คำ” คำที่ประกอบด้วยเลขไบนารี 8 หลักเราเรียกว่า “1 ไบท์” ตำแหน่งต่างๆในหน่วยความจำที่เราสามารถอ้างอิงได้เราเรียกว่า “ที่อยู่”

3. พอร์ตอินพุต / เอาต์พุต ของ 8051

เป็นส่วนที่ใช้ส่งข้อมูลเข้าหรือออกจาก 8051 ทำให้ 8051 ติดต่อกับภายนอกได้ อุปกรณ์อินพุต/เอาต์พุต ได้แก่ 4 อินพุต / เอาต์พุต พอร์ต ทำหน้าที่เป็นส่วนที่รับ-ส่งข้อมูลผ่านทางบัสข้อมูล ซึ่งเป็น

สัญญาณดิจิทัลเข้าหรือออกจากตัว MCS-51 โดยมีทั้งหมด 4 พอร์ต แต่ละพอร์ตจะรับ-ส่งข้อมูลได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

8 บิต มี P0, P1, P2 และ P3 บางพอร์ตจะทำงานได้มากกว่า 1 อย่างได้ แต่จะใช้วิธีการทำงานตามลำดับโดยการควบคุมยก สัญญาณควบคุมที่ถอดรหัสมาจากแต่ละคำสั่งที่ให้คอมพิวเตอร์ทำงาน และสัญญาณทั้งหมดจะอ้างอิงกับสัญญาณนาฬิกา

อินพุทพอร์ต คือแหล่งที่มีข้อมูลให้แก่ไมโครโปรเซสเซอร์ เมื่อไมโครโปรเซสเซอร์ทำคำสั่งอินพุท

เอาต์พุทพอร์ต คือแหล่งที่รับข้อมูลจากไมโครโปรเซสเซอร์ เมื่อไมโครโปรเซสเซอร์ทำคำสั่งเอาต์พุท

1) ไทม์เมอร์ 0 และ ไทม์เมอร์ 1

เป็นวงจรนับที่สามารถกำหนดให้ทำการนับจำนวนไซเคิลของสัญญาณ ที่ต่อจากภายนอก 8051 หรือจำนวนไซเคิลของสัญญาณนาฬิกาภายใน 8051 ก็ได้ค่าจากการนับจะถูกอ่านหรือตั้งค่าเริ่มต้นของการนับได้โดยซีพียู

2) พอร์ตอนุกรม (Serial Port)

ซีพียู จะอ่านและเขียนข้อมูลกับพอร์ตอนุกรมเป็นแบบ 8 บิต แต่ข้อมูลจะถูกส่งออกจาก 8051 เรียงไปที่ละบิตออกจากขา TXD และมีการรับข้อมูลเข้าก็จะรับเข้ามาที่ละบิตทางขา RXD แล้วจัดเรียงใหม่เป็น 8 บิต เพื่อให้ซีพียูอ่านไปใช้งานต่อไป

8051 มีพอร์ตให้ใช้งานได้หลายแบบ ทำให้สะดวกแก่การนำไปใช้งานต่าง ๆ โดยจะต้องเขียนโปรแกรมขึ้นมาควบคุม

4) ระบบอินเตอร์รัปต์ของ 8051

การติดต่อระหว่างไมโครคอนโทรลเลอร์กับอุปกรณ์ภายนอกมักจะทำได้โดยการตรวจสอบสถานะของสัญญาณติดต่อระหว่างกัน การอินเตอร์รัปต์เป็นวิธีการหนึ่งที่นิยมนำมาใช้กับไมโครคอนโทรลเลอร์เพื่อสามารถจัดการตอบรับหรือบริการกับอุปกรณ์ต่าง ๆ ให้เป็นไปได้อย่างรวดเร็ว

ไมโครคอนโทรลเลอร์ 8051 สามารถเกิดการอินเตอร์รัปต์ โดยการจำแนกตามแหล่งที่มาของสัญญาณ (Signal Source) ของสัญญาณอินเตอร์รัปต์นั้น ๆ ได้แก่

1) สัญญาณอินเตอร์รัปต์ภายนอก (External Interrupt)

การตรวจสอบสัญญาณที่เข้ามาอินเตอร์รัปต์นี้ จะสารณกำหนดให้มีการตรวจสอบในลักษณะเมื่อได้มีการเปลี่ยนแปลงสัญญาณไปแล้ว หรือในช่วงเวลาขณะเริ่มมีการเปลี่ยนแปลงสัญญาณจากลอจิกสูงไปต่ำ

2) สัญญาณอินเตอร์รัปต์ภายใน (Internal Interrupt)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แหล่งกำเนิดสัญญาณนี้จะป็นวงจรภายในของไมโครคอนโทรลเลอร์เองเช่น วงจรนับ/วงจรจับเวลา วงจรเชื่อมต่อสัญญาณอนุกรม เป็นต้น

2.3.2 การสื่อสารข้อมูลแบบอนุกรม

การเชื่อมต่อระหว่างพอร์ตอนุกรมแต่ละพอร์ตจะใช้เส้นสัญญาณอนุกรมแบบมาตรฐานอยู่ 3 แบบ คือ มาตรฐาน EIA RD-422, ระบบวงรอบกระแส และมาตรฐาน EIA RS-232 โดยทั่วไปเราจะใช้เส้นส่งสัญญาณอนุกรมแบบมาตรฐาน EIA RS-232 มากที่สุด ซึ่งเราจะเรียกว่า RS-232 สายส่ง

1) มาตรฐาน EIA RS-232

มาตรฐาน EIA RS-232 ซึ่งเราจะเรียก RS-232 ได้กำหนดให้ค่าสัญญาณไฟฟ้าที่มีระดับศักดาไฟฟ้าเท่ากับ 3 โวลต์หรือสูงกว่า มีค่าทางตรรกะเป็น 0 วงจรไอซีที่สร้างสัญญาณเหล่านี้ต้องการแหล่งจ่ายไฟขนาด +12 โวลต์ RS-232 จะใช้สาย 1 เส้นสำหรับส่งข้อมูล และใช้สายอีก 1 เส้นสำหรับข้อมูลโดยสัญญาณในแต่ละสายนี้จะถูกอ้างอิงเทียบกับกราวด์ (ขาเบอร์ 7) มาตรฐาน RS-232 นี้ยังได้กำหนดสัญญาณตอบรับเพื่อใช้ในการควบคุมการรับ/ส่งข้อมูลด้วย ในตารางที่ 2.1 แสดงตัวเชื่อมต่อแบบ D ชนิด 9 ขา (9 pin D-connector)

มาตรฐาน RS-232 จะสามารถใช้ส่งข้อมูลได้ไกลสุด 50 เมตร ด้วยอัตรา 9600 บอด แต่ถ้าเราต้องการให้ส่งได้ไกลกว่านี้ เราก็ต้องส่งข้อมูลด้วยอัตราส่งข้อมูลที่ช้ากว่านี้ และถ้าเราต้องการส่งข้อมูลในระยะที่ใกล้กว่านี้ เราก็สามารถส่งข้อมูลด้วยอัตราที่มากกว่าอัตรา 9600 บอด

9 Pin	EIA ES-232 Circuit	CCIT V.24 Circuit	RS-232 Description	Signal Type & Direction
5	AB	102	Signal ground/common return	Ground/common
2	BB	104	Received data	Data from DCE
3	BA	103	Transmitted data	Data to DCE

ตารางที่ 2.1 ตัวเชื่อมต่อที่นิยมใช้กับสายส่งสัญญาณอนุกรมแบบมาตรฐาน RS-232

2) รูปแบบของข้อมูลอนุกรมและอัตราบอดในการสื่อสารข้อมูลอนุกรม

อัตราบอด (baud rate) คือ ความเร็วในการรับส่งข้อมูลอนุกรม มีหน่วยเป็นบิตต่อวินาที เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

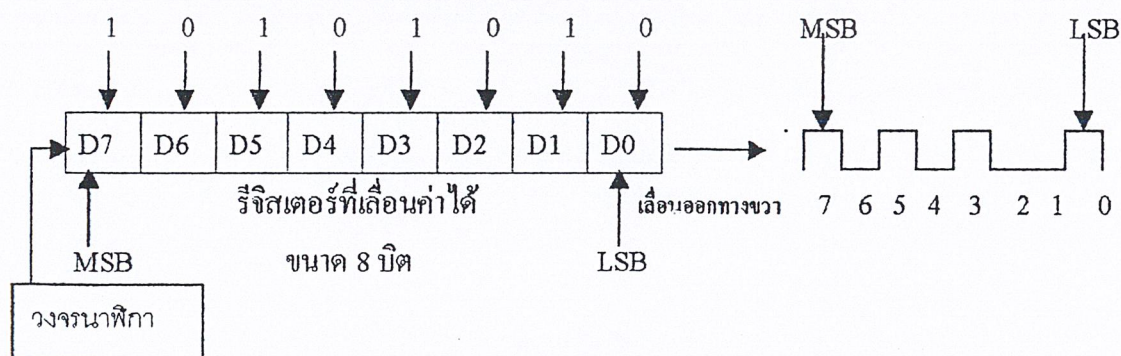
อัตราบอด	ช่วงเวลาของแต่ละบิต
110	9.91 ms
150	6.67ms
300	3.33ms
600	1.67ms
1200	0.833ms
2400	0.417ms
4800	0.208ms
9600	0.104ms
19200	0.052ms

ตารางที่ 2.2 อัตราบอดและช่วงเวลาของแต่ละบิตข้อมูลในการสื่อสารข้อมูลอนุกรม

3) การเชื่อมต่อแบบอนุกรมและ UART

ในรูปที่ 2.13 แสดงการแปลงข้อมูลแบบขนานมาเป็นข้อมูลแบบอนุกรม โดยเริ่มแรกข้อมูลแบบขนานจะถูกนำไปเก็บไว้ในรีจิสเตอร์ที่เลื่อนค่าได้ (shift register) จากนั้นเราจะใช้สัญญาณนาฬิกาในการเลื่อนค่าในรีจิสเตอร์ออกมาทีละบิต (โดยจะเลื่อนค่าไปทางขวามือ) โดยบิตแรกที่ถูกเลื่อนออกมาคือ บิต LSB ของข้อมูลและบิตทั้งสองที่ถูกเลื่อนออกมาก็คือ บิตที่อยู่ถัดจากบิต LSB และบิตต่อ ๆ ไป สำหรับบิตสุดท้ายที่ถูกเลื่อนออกมาก็คือ บิต MSB ของข้อมูล

ข้อมูลที่เข้ามาแบบขนาน



รูปที่ 2.14 การแปลงข้อมูลแบบขนานเป็นข้อมูลอนุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การแปลงข้อมูลแบบอนุกรมไปเป็นข้อมูลแบบขนานนั้นจะมีขั้นตอนตรงกันข้ามกับที่กล่าวมา นั่นคือข้อมูลแบบอนุกรมจะถูกเลื่อนไปเก็บในรีจิสเตอร์ที่เลื่อนค่าได้ โดยใช้สัญญาณนาฬิกาเป็นตัวควบคุม และหลังจากที่ได้มีการเลื่อนข้อมูลทุกบิตเข้าไปในรีจิสเตอร์ที่เลื่อนค่าได้หมดแล้ว ข้อมูลในรีจิสเตอร์นี้ก็จะถูกนำออกมาแบบขนานเพื่อนำไปให้กับไมโครโปรเซสเซอร์

อุปกรณ์ที่ทำหน้าที่แปลงข้อมูลแบบอนุกรมเป็นข้อมูลแบบขนานและแปลงข้อมูลแบบขนานเป็นข้อมูลแบบอนุกรม เราเรียกว่า UART (universal asynchroniud receiver-transmitter) UART เป็นวงจรร LSI ซึ่งนอกจากจะมีหน้าที่ในการแปลงข้อมูลแล้วมันยังมีหน่วยควบคุมและหน่วยตรวจสอบการทำงานอีกด้วย

ในการส่งข้อมูลขนาด 8 บิตแบบอนุกรมนี้อาจต้องมีบิตสตาร์ท (start bit) และบิตสตอป (stop bit) เพิ่มขึ้นมา ซึ่งจะทำให้ข้อมูลที่ส่งไปจริงๆ นั้นมีขนาด 10 บิต ในรูปที่ 13.8 แสดงเวิร์คข้อมูลขนาด 8 บิต และบิตสตอป 1 บิตเพิ่มเข้ามา โดยบิตสตาร์ท (มีค่าเป็น 0 เสมอ) จะบอก UART ที่รับข้อมูลให้รู้ว่าเริ่มมีข้อมูลกำลังเข้ามา และบิตสตอป (มีค่าเป็น 1 เสมอ) จะบอก UART ที่รับข้อมูลให้รู้ว่าการส่งข้อมูลได้เสร็จสิ้นลงแล้ว

stop	D7	D6	D5	D4	D3	D2	D1	D0	start
------	----	----	----	----	----	----	----	----	-------

รูปที่ 2.15 เวิร์คข้อมูลขนาด 8 บิต กับบิตสตาร์ทและบิตสตอปที่ใช้ในการโอนย้ายข้อมูลแบบอนุกรม

เมื่อเรานำบิตที่ 8 ของข้อมูลมาใช้ในการตรวจสอบความผิดพลาดในการสื่อสารข้อมูล ซึ่งเราเรียกบิตนี้ว่า บิตพาริตี (parity bit) เราจะใช้พาริตีคี่หรือพาริตีคู่ในการตรวจสอบความผิดพลาดในการส่งข้อมูล เมื่อ UART ได้รับข้อมูลมันก็จะทำการทดสอบว่าข้อมูลนั้นเป็นพาริตีคี่หรือพาริตีคู่ ถ้า UART พบว่าข้อมูลที่มันทดสอบมีค่าพาริตีไม่ตรงตามที่กำหนด บิตพาริตีในรีจิสเตอร์สถานะของ UART ก็จะถูกเซตเพื่อแสดงว่าข้อมูลที่รับเข้ามานั้นมีข้อผิดพลาด และโปรแกรมที่ทำการรับข้อมูลนั้นก็จะขอให้มีการส่งข้อมูลให้กับ UART ใหม่อีกครั้ง

จากรูปเราจะพบว่าเราสามารถแบ่ง UART ออกเป็น 4 ส่วนได้แก่ ส่วนที่ทำหน้าที่ส่งข้อมูล ส่วนที่ทำหน้าที่รับข้อมูล ส่วนที่กำหนดสถานะ และส่วนที่เป็นวงจรควบคุม

ข้อมูลในรีจิสเตอร์สถานะของ UART สามารถถูกนำไปเก็บในบัฟเฟอร์ข้อมูลได้ ซึ่งทำได้โดยการส่งสัญญาณควบคุมสเตตัสเอาต์พุตสโตรบ (status output strobe) ซึ่งขั้นตอนดังกล่าวจะทำให้ไมโครโปรเซสเซอร์สามารถอ่านข้อมูลจากรีจิสเตอร์สถานะได้

รีจิสเตอร์สถานะของ UART จะมีบิตบอกสถานะดังต่อไปนี้

บิต OR (overrun) : บิตนี้จะมีค่าเป็น 1 เมื่อข้อมูลชิ้นใหม่เข้ามาทับข้อมูลชิ้นเดิมที่เก็บอยู่ ซึ่งข้อมูลชิ้นเดิมนี้อย่างไม่ได้ถูกนำไปเก็บในบัฟเฟอร์รับข้อมูล

บิต FE (framing error) : บิตนี้จะมีค่าเป็น 1 ถ้า UART ไม่พบบิตสตอป ซึ่งอาจเป็นเพราะว่า UART ไม่ได้อ่านบิตสตาร์ตจากตำแหน่งที่ถูกต้อง ซึ่งก็หมายความว่ามันเริ่มทำงานกับบิตที่ไม่ใช่บิตสตาร์ต

บิต PE (parity error) : บิตนี้จะมีค่าเป็น 1 เมื่อ UART ได้ตรวจสอบพบว่าข้อมูลที่มันได้รับมีค่าพาริตีไม่ตรงกับค่าในบิตพาริตีของข้อมูล

เราจะกำหนดให้ UART ทำการทดสอบค่าแบบพาริตีคู่หรือพาริตีคี่หรือพาริตีคี่ได้โดยระบุไว้ที่วงจรควบคุมก่อนที่จะนำ UART นี้มาทำการรับข้อมูล

บิต TBE (transmit buffer empty) : บิตนี้จะมีค่าเป็น 1 เมื่อ UART ได้ทำการส่งข้อมูลออกไปแล้ว ซึ่งทำให้เราสามารถนำข้อมูลชิ้นต่อไปที่จะส่งเก็บลงในบัฟเฟอร์

บิต DA (Data available) : บิตนี้จะมีค่าเป็น 1 เมื่อ UART ได้รับข้อมูลชิ้นใหม่เข้ามาและไมโครโปรเซสเซอร์สามารถทำการอ่านข้อมูลนี้ได้โดยส่งสัญญาณคาล์เอาต์พุตสโตรบ

			Frame	Parity	Trans	Data	
0	0	Overrun	Error	error	Buffer Empty	available	0

รูปที่ 2.18 ความหมายของบิตต่างๆในรีจิสเตอร์สถานะของ UART

4) พอร์ตอนุกรมของ 8051

คุณสมบัติที่สำคัญข้อหนึ่งของ 8051 คือ พอร์ตแบบอนุกรมที่มีอยู่ในตัวชิป 8051 ในไมโครโปรเซสเซอร์ทั่วไป การเพิ่มพอร์ตอนุกรมจะต้องเพิ่ม UART และวงจรควบคุมหรือไอซี ที่สนับสนุนการเชื่อมต่อแบบอนุกรม ซึ่งอุปกรณ์เหล่านี้มีความซับซ้อนและมีราคาค่อนข้างสูง ต่อไปเราจะกล่าวถึงพอร์ตอนุกรมใน 8051

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเชื่อมต่อแบบอนุกรมของ 8051 นั้นเป็นแบบสื่อสารสองทางเต็มอัตรา (full duplex) ซึ่งหมายความว่ามันสามารถทำการรับและส่งข้อมูลได้ในเวลาเดียวกัน การเชื่อมต่อแบบอนุกรมนี้อาจมีการพักข้อมูลที่รับเข้ามา (receive-buffered) ซึ่งหมายความว่าก่อนที่จะส่งข้อมูลขึ้นแรกจะถูกส่งไปยังไมโครโปรเซสเซอร์ พอร์ตอนุกรมจะสามารถรับข้อมูลอนุกรมขึ้นที่สองได้ อย่างไรก็ตามข้อมูลขึ้นแรกจะต้องถูกส่งไปยังไมโครโปรเซสเซอร์ก่อนที่จะส่งข้อมูลที่สองที่รับเข้ามาจะถูกนำไปเก็บในแลตช์ข้อมูล มิฉะนั้นข้อมูลขึ้นแรกที่อยู่ในนั้นจะถูกเขียนทับ

การเชื่อมต่อแบบอนุกรมนี้อาจมีการใช้รีจิสเตอร์ 2 ตัว ที่ทำหน้าที่รับและส่งข้อมูล เราติดต่อกับรีจิสเตอร์ทั้งสองตัวนี้ได้โดยการอ้างอิงรีจิสเตอร์พิเศษตัวหนึ่งที่มีชื่อว่า SBUF ถ้าเราทำการเขียนข้อมูลลง SBUF แสดงว่าเราได้เขียนข้อมูลลงรีจิสเตอร์ส่งข้อมูล แต่ถ้าเราทำการอ่านข้อมูลจาก SBUF แสดงว่าเราอ่านข้อมูลที่อยู่ในรีจิสเตอร์รับข้อมูล จะเห็นได้ว่ารีจิสเตอร์ที่ทำหน้าที่รับและส่งข้อมูลทั้งสองนี้มีค่าแอดเดรสค่าเดียวกัน (เนื่องจากเราสามารถอ้างอิงรีจิสเตอร์ทั้งสองนี้ได้โดยใช้ค่าแอดเดรสที่เป็นตำแหน่งของ SBUF เพียงค่าเดียว)

รีจิสเตอร์ควบคุมพอร์ตอนุกรม (serial port control register : SCON) จะเก็บข้อมูลที่ควบคุมการทำงานของพอร์ตอนุกรม รูปที่ 2.18 แสดงความหมายของบิตแต่ละบิตในรีจิสเตอร์ควบคุมพอร์ตอนุกรม

SM0	SM1	SM2	REN	TB8	RB8	TI	RI
-----	-----	-----	-----	-----	-----	----	----

รูปที่ 2.19 รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม (serial port control register : SCON)

SM0 SCON.7 กำหนดโหมดการทำงานของพอร์ตอนุกรม

SM1 SCON.6 กำหนดโหมดการทำงานของพอร์ตอนุกรม

SM2 SCON.5 บิตนี้จะนำมาใช้ในการสื่อสารระหว่างไมโครโปรเซสเซอร์หลายตัวในโหมดการทำงานที่ 1 และ 3 สำหรับโหมดที่ 2 และ 3 นี้ ถ้าบิต SM2 มีค่าเป็น 1 บิต RI จะไม่เกิดการเปลี่ยนแปลง หากค่าในบิตที่ 9 ที่รับเข้ามามีค่าเป็น 0 สำหรับในโหมด 1 ถ้าบิต SM2 มีค่าเป็น 1 บิต RI จะไม่เปลี่ยนแปลงถ้าเกิดการรับบิตสตอปผิด สำหรับในโหมด 0 บิต SM2 จะมีค่าเป็น 0

REN SCON.4 บิตนี้จะถูกเซตหรือเคลียร์โดยโปรแกรมเพื่อทำการรับ/ไม่รับข้อมูล

TB8 SCON.3 บิตที่ 9 ในข้อมูลอนุกรมที่จะถูกส่งไป

RB8 SCON.2 บิตนี้จะมีค่าเท่ากับบิตที่ 9 ในข้อมูลอนุกรมที่รับเข้ามาในโหมดที่ 2 และโหมดที่ 3 สำหรับในโหมดที่ 1 ถ้าบิต SM2 มีค่าเป็น 0 บิต RB8 จะมีค่าเท่ากับค่าในบิตสตอปของข้อมูลที่รับเข้ามา สำหรับในโหมด 0 บิต RB8 จะไม่ถูกนำมาใช้งาน

TI SCON.1 แฟล็ก transmit interrupt จะถูกเซตโดยฮาร์ดแวร์ หลังจากได้มีการเลื่อนข้อมูลขนาด 8 บิตออกในโหมด 0 หรือเมื่อได้ทำการส่งข้อมูลจนพบบิตสตอปในโหมดอื่นหลังจากส่งข้อมูลแบบอนุกรมค่าในบิตนี้จะถูกเคลียร์โดยซอฟต์แวร์ เราสามารถทำการอ้างอิงข้อมูลในรีจิสเตอร์นี้ทีละบิตได้

RI SCON.0 แฟล็ก receive interrupt จะถูกเซตโดยฮาร์ดแวร์หลังจากที่ได้มีการเลื่อนข้อมูลขนาด 8 บิต เข้ามาในโหมด 0 หรือที่จุดครึ่งทางของช่องบิตสตอปในโหมดอื่น (ยกเว้นในกรณี SM2 มีค่าเป็น 1) หลังการรับข้อมูลแบบอนุกรม ค่าในบิตนี้จะถูกเคลียร์โดยซอฟต์แวร์

ตารางการทำงานโหมด

SM0	SM1	โหมดการทำงาน ของพอร์ตอนุกรม	รายละเอียด	อัตราบอด
0	0	0	รีจิสเตอร์ที่เลื่อนค่าได้	ความถี่ออสซิลเลเตอร์ 1/12
0	1	1	UART ขนาด 8 บิต	ความถี่ออสซิลเลเตอร์ 1/64
1	0	2	UART ขนาด 9 บิต	ความถี่ออสซิลเลเตอร์ 1/32
1	1	3	UART ขนาด 9 บิต	เปลี่ยนแปลงได้

ตารางที่ 2.3 รีจิสเตอร์ควบคุมการทำงานของพอร์ตอนุกรม (serial port control register : SCON)

เราสามารถทำการอ้างอิงข้อมูลในรีจิสเตอร์นี้ทีละบิตได้

บิต SM0 และ SM1 จะใช้ในการกำหนดโหมดการทำงานของพอร์ตอนุกรม (ซึ่งมีทั้งหมด 4 โหมดคือ โหมด 0, 1, 2 และ 3) ต่อไปเราจะกล่าวถึงการทำงานของพอร์ตอนุกรมในแต่ละโหมด

ในโหมด 0 พอร์ตอนุกรมจะเป็นเหมือนรีจิสเตอร์ที่เลื่อนค่าได้ (shift register) ซึ่งจะเลื่อนค่าตามสัญญาณนาฬิกาที่มีอัตราความถี่เท่ากับ 1/12 ของความถี่ออสซิลเลเตอร์ใน 8051 สัญญาณนาฬิกานี้เป็นตัวกำหนดอัตราความเร็วในการเคลื่อนย้ายข้อมูลแบบอนุกรมที่เราเรียกว่า อัตราบอด (baud rate)

ในโหมดที่ 1 พอร์ตอนุกรมจะรับและส่งข้อมูลขนาด 10 บิต ซึ่งข้อมูลขนาด 10 บิตนี้ประกอบด้วยบิตสตาร์ท บิตข้อมูลขนาด 8 บิต และบิตสตอป 1 บิต ในโหมดการทำงานนี้ ไทเมอร์

(timer) ตัวหนึ่งใน 8051 จะถูกนำมาใช้กำหนดอัตราเร็วในการเคลื่อนย้ายข้อมูลของพอร์ตอนุกรม (หรือกำหนดอัตราบอด)

ในโหมดที่ 2 พอร์ตอนุกรมจะทำการรับและส่งข้อมูลทีละ 11 บิต ซึ่งข้อมูลขนาด 11 บิตนี้ ประกอบด้วยข้อมูลขนาด 8 บิต บิตที่ 9 (เป็นบิตที่เราสามารถกำหนดค่าได้) บิตสตาร์ทและบิตสตอป โดยบิตที่ 9 จะมีค่าตรงกับบิต TB8 ของรีจิสเตอร์ควบคุมพอร์ตแบบอนุกรม (SCON) ในโหมดการทำงานนี้เราสามารถเลือกอัตราบอดให้มีค่าเป็น $1/32$ หรือ $1/64$ ของความถี่ออสซิลเลเตอร์ใน 8051 ได้

การทำงานของพอร์ตอนุกรมในโหมดที่ 3 จะทำการรับและส่งข้อมูลทีละ 11 บิต และมีบิตที่ 9 ที่กำหนดค่าได้ โดยบิตนี้จะมีค่าตรงกับบิต RB8 ในรีจิสเตอร์ควบคุมพอร์ตแบบอนุกรม (SCON) โหมดที่ 3 นี้จะทำหน้าที่เช่นเดียวกับโหมดที่ 2 ยกเว้นแต่อัตราเร็วในการเคลื่อนย้ายข้อมูลจะสามารถเปลี่ยนแปลงได้

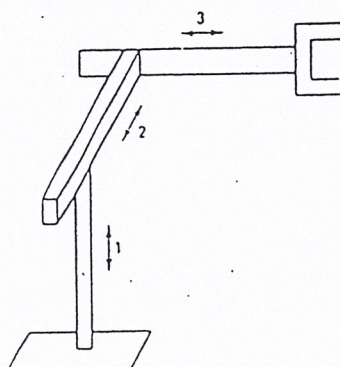
2.4 ชนิดของแขนกลและส่วนประกอบต่างๆของฮาร์ดแวร์

2.4.1 ชนิดของแกนแขน

แขนกลที่มีใช้ในปัจจุบันมีอยู่มากมายหลายชนิด เพื่อให้เหมาะสมตามลักษณะต่างๆ การแบ่งแขนกลออกเป็นประเภทต่างๆ จะแตกต่างกันไปตามส่วนที่พิจารณา ในที่นี้จะแบ่งตามลักษณะการเคลื่อนที่ของท่อนแขน ซึ่งแบ่งได้เป็น 5 ประเภทคือ

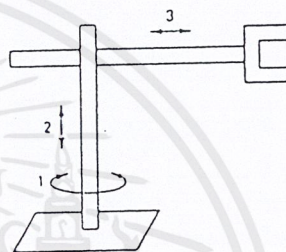
1 Cartesian robot

มีลักษณะการเคลื่อนที่เป็นแบบ 3 แกน คือท่อนแขนแต่ละท่อน เคลื่อนที่ตามแนวแกน XYZ ดังรูป มีความสามารถในการเข้าถึงจุดต่างๆได้อย่างแม่นยำและง่ายต่อการควบคุม ขอบเขตการเคลื่อนที่เป็นกล่องสี่เหลี่ยม



2 Cylindrical robot

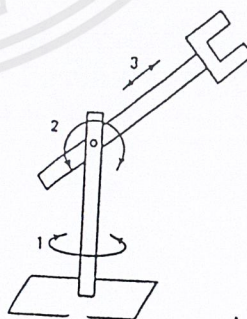
ท่อนแขนท่อนล่างหมุนรอบแกนได้ พร้อมทั้งมีการเคลื่อนที่ตามแนวแกน ส่วนท่อนถัดมาจะเคลื่อนที่ตามแนวแกนซึ่งตั้งฉากกับท่อนแรก ขอบเขตการเคลื่อนที่เป็นทรงกระบอก



รูปที่ 2.21 แกนแขนชนิด Cylindrical robot

3 Spherical robot

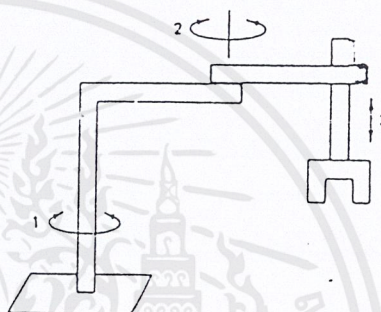
ท่อนแขนท่อนล่างหมุนรอบแกน ข้อต่อระหว่างท่อนแรกและท่อนที่สองจะเป็นจุดหมุนให้ท่อนที่สอง โดยที่ท่อนที่สองสามารถเคลื่อนที่ตามแนวแกนได้ด้วย มีขอบเขตการเคลื่อนที่เป็นทรงกลม



รูปที่ 2.22 แกนแขนชนิด Spherical robot

4 SCARA robot

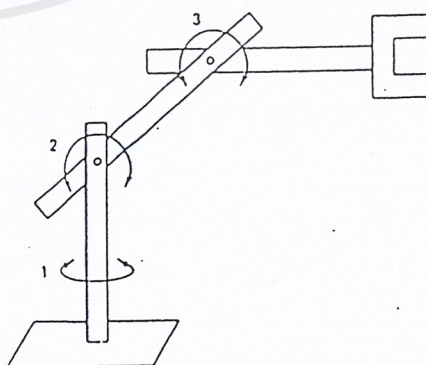
ท่อนแขนท่อนแรกเป็นรูปฉากหมุนรอบแกน ท่อนที่สองหมุนรอบข้อต่อระหว่างท่อนแรกกับท่อนที่สอง และท่อนที่สามเคลื่อนที่ตามแนวแกน ความสามารถในการเข้าถึงจุดต่างๆ ขึ้นอยู่กับขีดจำกัดของแขนสองท่อนแรกเป็นสำคัญดังรูป



รูปที่ 2.23 แกนแขนชนิด SCARA robot

5 Articulated robot

ท่อนแขนท่อนแรกหมุนรอบแกน โดยมีข้อต่อกับท่อนที่สองเป็นจุดหมุนให้ท่อนที่สอง และข้อต่อของท่อนที่สองกับท่อนที่สามเป็นจุดหมุนให้ท่อนที่สาม แขนกลชนิดนี้มีการเคลื่อนที่ซับซ้อน และใกล้เคียงกับแขนของมนุษย์มากที่สุด



รูปที่ 2.24 แกนแขนชนิด Articulated robot

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การเลือกใช้แกนกลั่นนั้นจะต้องดูความเหมาะสมของงาน ทั้งทางด้านขอบเขตของการเข้าถึงจุดต่างๆ และลักษณะการเคลื่อนที่ว่าเป็นแบบจุดต่อจุด ซึ่งจะให้ความสำคัญที่ความแม่นยำในการไปถึงจุดเป้าหมาย หรือการเคลื่อนที่แบบต่อเนื่อง ซึ่งจะให้ความสำคัญที่เส้นทางการเคลื่อนที่ให้เป็นไปอย่างต่อเนื่อง โดยอาศัยการคำนวณอย่างละเอียด ดังนั้นการเลือกแกนกลั่นที่ถูกต้องจะช่วยให้การทำงานเป็นไปอย่างมีประสิทธิภาพมากขึ้น

2.4.2 วงจรโวลต์เตจ เรกกูเลเตอร์ (เปลี่ยนกระแส supply จาก 12 โวลต์ เป็น 5 โวลต์)

โวลต์เตจ เรกกูเลเตอร์ สามารถแบ่งเป็น 2 ชนิดพื้นฐาน คือ ชั๊นท์ เรกกูเลเตอร์ หรือ เรกกูเลเตอร์แบบขนาน และเรกกูเลเตอร์แบบอนุกรม โดยชั๊นท์ได้จากลักษณะการต่อของเรกกูเลเตอร์เข้ากับวงจรเรกกูเลเตอร์แบบอนุกรมมีการนำมาใช้มากกว่าเรกกูเลเตอร์แบบขนาน ทั้งนี้เพราะมีประสิทธิภาพสูงกว่าและมีการสูญเสียกำลังน้อยกว่า โดยมากแล้วเรกกูเลเตอร์แบบขนานมักใช้เป็นอุปกรณ์ควบคุม เป็นตัวเรกกูเลเตอร์ป้องกันในกรณีโวลต์เกิดการช็อก



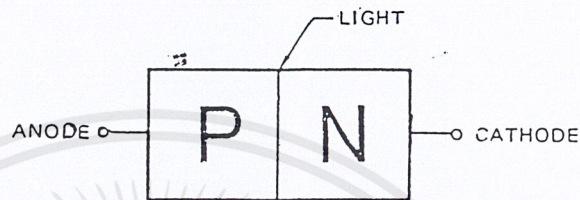
รูปที่ 2.25 แสดงวงจรเรกกูเลเตอร์พื้นฐาน โดยใช้ซีเนอร์ไดโอด

วงจรเรกกูเลเตอร์พื้นฐานโดยใช้ซีเนอร์ไดโอดซึ่งจัดเป็นเรกกูเลเตอร์แบบขนานโดยที่ซีเนอร์ไดโอดจะต่ออนุกรมอยู่กับตัวต้านทาน แรงดันอินพุต (DC) ซึ่งยังไม่ได้ควบคุม จะถูกจ่ายมาตกร่อมตัวต้านทานและซีเนอร์ไดโอด จะทำให้เกิดรีเวิร์กไบอัสแก่ซีเนอร์ไดโอด และตัวต้านทานจะยอมให้กระแสปริมาณน้อยไหลเพื่อรักษาระดับแรงดันซีเนอร์ไว้ไม่ให้เลยจุดเบรคดาวน์ โดยที่แรงดันอินพุตต้องมีระดับแรงดันที่สูงกว่าแรงดันซีเนอร์ และแรงดันซึ่งตกคร่อมซีเนอร์จะเท่ากับแรงดันของตัวซีเนอร์ ส่วนแรงดันที่ตกคร่อมตัวต้านทานจะเท่ากับผลต่างระหว่างแรงดันซีเนอร์กับแรงดันอินพุต

2.4.3 อุปกรณ์ตรวจจับแสง (Opto devices)

อุปกรณ์ตรวจจับแสง (Light-sensitive devices) แบ่งออกได้หลายชนิด แต่ในโครงการนี้เลือกใช้โฟโต้ไดโอดเป็นอุปกรณ์ตรวจจับแสง

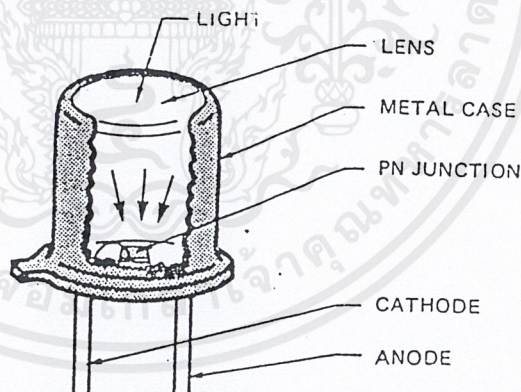
โฟโตไดโอด (photodiode) คืออุปกรณ์สารกึ่งตัวนำที่ทำจากซิลิคอน และมันสามารถถูกสร้างได้ 2 วิธี ซึ่งวิธีแรกคือจะให้เป็นแบบต่อร่วมกันของสารพี-เอ็น ดังในรูปที่ 2.31 และอีกวิธีหนึ่งคือการใส่ชั้นอินเทนซิก (intinsic (ไม่มีการโดป) layer) ระหว่างสารพีกับสารเอ็น ดังในรูปที่ 2.32 ซึ่งจะอยู่ในรูปของพีไอเอ็น โฟโตไดโอด (PIN photodiode)



รูปที่ 2.26 แสดงพี-เอ็น จังก์ชันโฟโตไดโอด

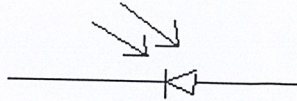
โฟโตไดโอดแบบรอยต่อ พี-เอ็น (PN junction photodiode) จะมีหลักการทำงานเหมือนกับโซลาร์เซลล์ แต่มันถูกใช้โดยตรงที่มันถูกใช้ในการควบคุมการไหลของกระแสไม่ได้

ข้อดีของโฟโตไดโอด คือมีความไวในการตอบสนองต่อการเปลี่ยนแปลงของแสงและยังเป็นอุปกรณ์ตรวจจับแสงที่ไวที่สุดด้วย

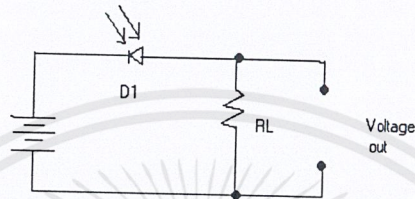


รูปที่ 2.27 แสดงโครงสร้างภายนอกของโฟโตไดโอด

ข้อเสียของโฟโตไดโอด คือจะให้เอาที่พูดค่าเมื่อเทียบกับอุปกรณ์ตรวจจับแสงอื่น ๆ จากในรูปที่ 2.27 จะแสดงโครงสร้างภายนอกของโฟโตไดโอด และสัญลักษณ์ของมันจะแสดงในรูปที่ 2.28 และวงจรการต่อใช้งานแสดงในรูปที่ 2.29



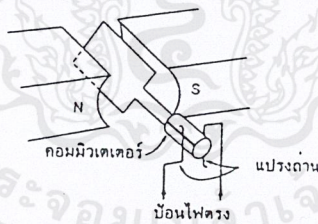
รูปที่ 2.28 แสดงสัญลักษณ์ของโฟโตไดโอด



รูปที่ 2.29 แสดงวงจรใช้งานของโฟโตไดโอด

2.4.4 มอเตอร์กระแสตรง

มอเตอร์จะมีลักษณะการทำงานเป็นการเปลี่ยนไฟฟ้าเป็นพลังงานกล มอเตอร์จะหมุนต่อเมื่อได้รับแรงดันไฟฟ้า โครงสร้างของมอเตอร์แสดงดังรูปที่ 2.35 เมื่อป้อนแรงดันไฟฟ้าเข้าที่แปรงถ่าน จะเกิดกระแสไหลผ่านขดลวดตัดสนามแม่เหล็ก ทำให้เกิดแรงผลักให้ขดลวดหมุนตามความสัณพันธ์



รูปที่ 2.30 การทำงานเบื้องต้นของมอเตอร์

$$E \propto WB$$

โดยที่ W คือ ความเร็วในการหมุน

B คือ ความเข้มของสนามแม่เหล็ก

หากกระแส i มีค่าสูงสุด ความเร็วจะกลายเป็น 0 แต่ถ้ากระแสไหลลดลง ความเร็วจะเพิ่มขึ้น แรงบิดที่เกิดขึ้นในมอเตอร์จะขึ้นกับกระแสไฟฟ้าที่ไหลในขดลวดของมอเตอร์ตามความสัณพันธ์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

$$T = iB$$

โดยที่ T คือ แรงบิดที่เกิดขึ้นในมอเตอร์

อัตราความเร็วของมอเตอร์จะเพิ่มขึ้นจนกระทั่งแรงบิดสมดุลกับโหลดที่ต่อเข้ามอเตอร์

อัตราความเร็วของมอเตอร์ไฟตรงสามารถควบคุมโดยการเปลี่ยนแปลงความเข้มของสนามแม่เหล็ก หรือปรับแรงดันอาร์มเจอร์ ส่วนการสร้างสนามแม่เหล็กสามารถใช้แม่เหล็กถาวรหรือต่อขดลวดเพื่อสร้างสนามแม่เหล็กก่อนก็ได้

2.4.7 วงจรเปรียบเทียบแรงดัน (Comparator)

วงจรเปรียบเทียบแรงดัน (Comparator) คือวงจรที่ทำการแบ่งแยกแรงดันออกจากกันให้ชัดเจนยิ่งขึ้นซึ่งในที่นี้จะทำการรับค่ามาจากตัว Sensor เนื่องจากแรงดันที่ได้จากตัว Sensor ยังไม่เป็นลอจิกที่น่าเชื่อถือพอที่จะส่งไปยัง PC (Personal Computer) ได้จึงนำสัญญาณที่ได้ไปผ่านวงจร Comparator ก่อน เพื่อให้ได้ลอจิก “0” และ “1” ที่น่าเชื่อถือออกมา วงจรจะทำการเปรียบเทียบแรงดันของขา + และขา - ของ ออปแอมป์

ถ้า โวลต์เตจขา + มากกว่าโวลต์เตจขา - จะให้โวลต์เตจที่ Output เป็น 5 โวลต์
(ลอจิก 1)

ถ้า โวลต์เตจขา + น้อยกว่าโวลต์เตจขา - จะให้โวลต์เตจที่ Output เป็น 0 โวลต์
(ลอจิก 0)

บทที่ 3

การคำนวณและการสร้าง

3.1 การสร้างและออกแบบ

การทำงานของเซนกลต่อเศษภาพ ต้องอาศัยการทำงานจากส่วนต่างๆ หลายส่วน ซึ่งได้แสดงรายละเอียดในแต่ละส่วนดังนี้

3.1.1 ส่วนรับข้อมูลภาพ

เนื่องจากการต่อเศษภาพในโครงการนี้ อาศัยการพิจารณารูปร่างของเศษภาพแต่ละชิ้น จึงต้องการเพียงข้อมูลภาพแบบขาวดำ จึงเลือกใช้กล้อง B/W CCD CAMERA ซึ่งให้ข้อมูลภาพเป็นแบบขาวดำ และมีการส่งสัญญาณภาพโดยใช้มาตรฐาน CCIR

สัญญาณภาพที่ได้จากกล้องวิดีโอ จะถูกนำไปเก็บใน CPU โดยผ่านวงจรแปลงสัญญาณอนาลอกเป็นดิจิทัล แต่เนื่องจากการศึกษาสัญญาณวิดีโอที่ผ่านมา จะพบว่าวงจรส่วนนี้ต้องใช้ chip ที่มีความเร็วสูงมาก และยังมีความซับซ้อนมากเกินไป จึงได้มีการนำเอา TV JUNER มาใช้ในการรับภาพ แสดงออกหน้าจอคอมพิวเตอร์และ Capture เก็บลงในหน่วยความจำแทน ลักษณะการเก็บข้อมูลจะเก็บเป็น File โดยใช้ 1 File ต่อ 1 ภาพ และเป็น File ชนิด bmp แบบ 24 bit RGB (3 byte ต่อ 1 จุด) ขนาด 192*144 bitmap ซึ่งง่ายต่อการนำไปวิเคราะห์หารูปร่างเพื่อใช้ในการต่อเศษภาพ

3.1.2 ส่วนประมวลผล

ในการต่อเศษภาพ จะใช้หลักการเดียวกับหลักการคิดของคนทั่วไป โดยจะตัดบางขั้นตอนที่มีความยุ่งยากมากออกไปเช่นการเทียบสี แต่จะเน้นไปที่รูปร่างของเศษภาพแต่ละชิ้น เริ่มจากการหามุมทั้ง 4 ต่อขอบ และต่อเข้าไปเรื่อยๆจนถึงขั้นในสุด (ส่วนรายละเอียดต่างๆ จะกล่าวถึงในขั้นต่อไป) เนื่องจากข้อมูลภาพแต่ละภาพมีขนาดใหญ่มาก และเศษภาพมีจำนวนหลายชิ้น จึงไม่เหมาะที่จะนำไปประมวลผลบน Single Board จึงได้ทำการเก็บข้อมูลรวมทั้ง Program ในการประมวลผลไว้ใน CPU เพื่อให้ทำการต่อเศษภาพให้เสร็จก่อน แล้วจึงส่งข้อมูลผ่านส่วน Interface ไปยังเซนกลเพื่อทำการหยิบต่อเศษภาพอีกที จากการศึกษา Program ที่ผ่านมาจะพบว่า Program Delphi มีความเหมาะสมต่อโครงการนี้ เนื่องจากเป็นการนำภาษา Pascal มาประยุกต์ง่ายต่อการทำความเข้าใจ และมี Function ให้ใช้งานที่รองรับการจัดการข้อมูลขนาดใหญ่ ทั้งทางด้าน Image Process และการ Interface เป็นอย่างดี

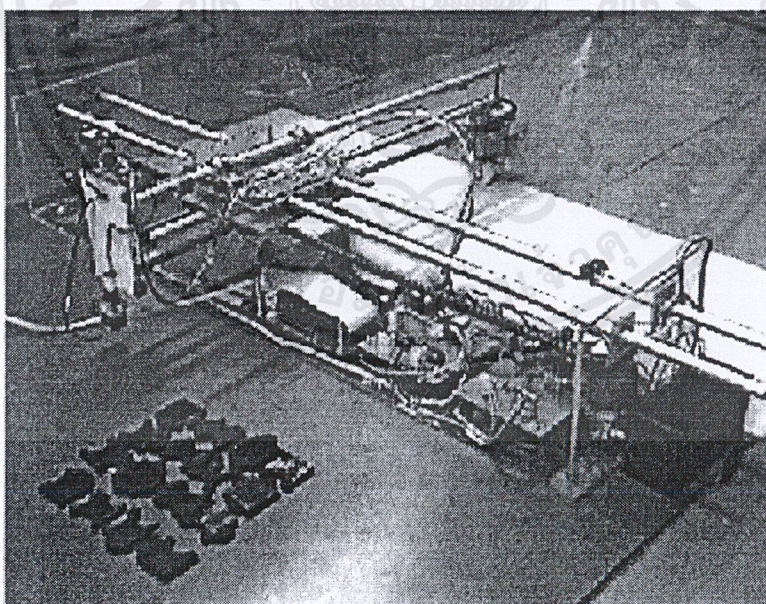
Program ต่อเศษภาพนั้นจะต้องบอกตำแหน่งของภาพแต่ละชิ้น โดยอาศัย Chain Code ในการพิจารณารูปร่าง และบอกตำแหน่งในการจับพร้อมทั้งองศาในการหมุนภาพให้กับแขนกล โดยอาศัย Moments หาจุดศูนย์กลางภาพ และ Central Moments ช่วยพิจารณาองศาในการหมุน

3.1.3 ส่วน Interface

การส่งข้อมูลจาก CPU ไปยังแขนกล จะส่งผ่านทาง Serial Port โดยมี Program ควบคุมการ Interface เพื่อควบคุมการรับส่งข้อมูลให้เป็นไปอย่างเหมาะสมและถูกต้อง ซึ่งจะถูกเขียนขึ้นบน Delphi

3.1.4 แขนกล

เนื่องจากการทำงานของแขนกลต่อเศษภาพเป็นแบบจุดต่อจุด และมีขอบเขตการเคลื่อนที่ อยู่ในกรอบสี่เหลี่ยม ดังนั้นจึงเลือกใช้แขนกลแบบ Cartesian robot หรือแบบ 3 แกน (XYZ) เพื่อให้ง่ายต่อการควบคุม และเข้าถึงจุดต่างๆ ได้อย่างแม่นยำ โดยใช้มอเตอร์กระแสตรงในการขับเคลื่อน ทางด้านโปรแกรมควบคุมการเคลื่อนที่ของแขนกล จะใช้ภาษา Assembly โปรแกรมลงใน Microcontroller ที่เป็น Single Board ควบคุมแขนกลผ่านวงจร drive



รูปที่ 3.1 เครื่องต่อภาพจากเศษภาพ

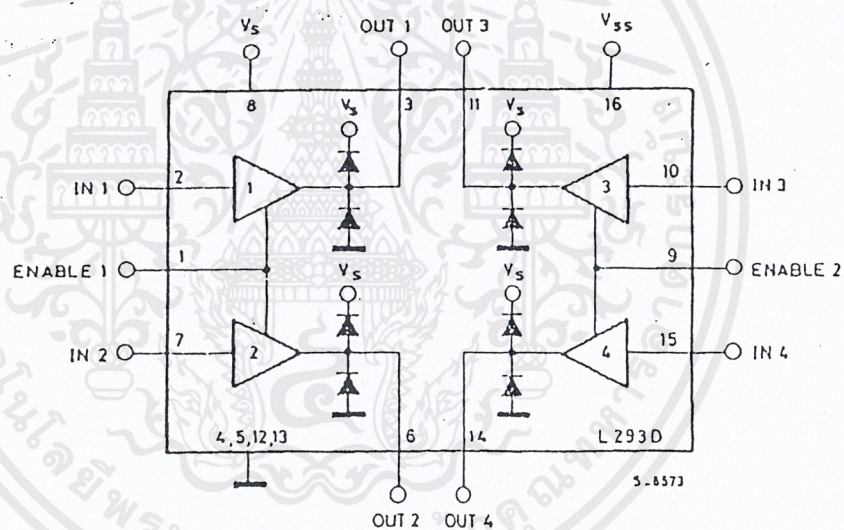
3.1.5 ส่วนวงจรต่างๆที่ประกอบในHardware

วงจรขับเคลื่อนมอเตอร์ ในโครงงานนี้ได้เลือกใช้ “ไดโอดเบอร์ L293D ในการขับเคลื่อนมอเตอร์

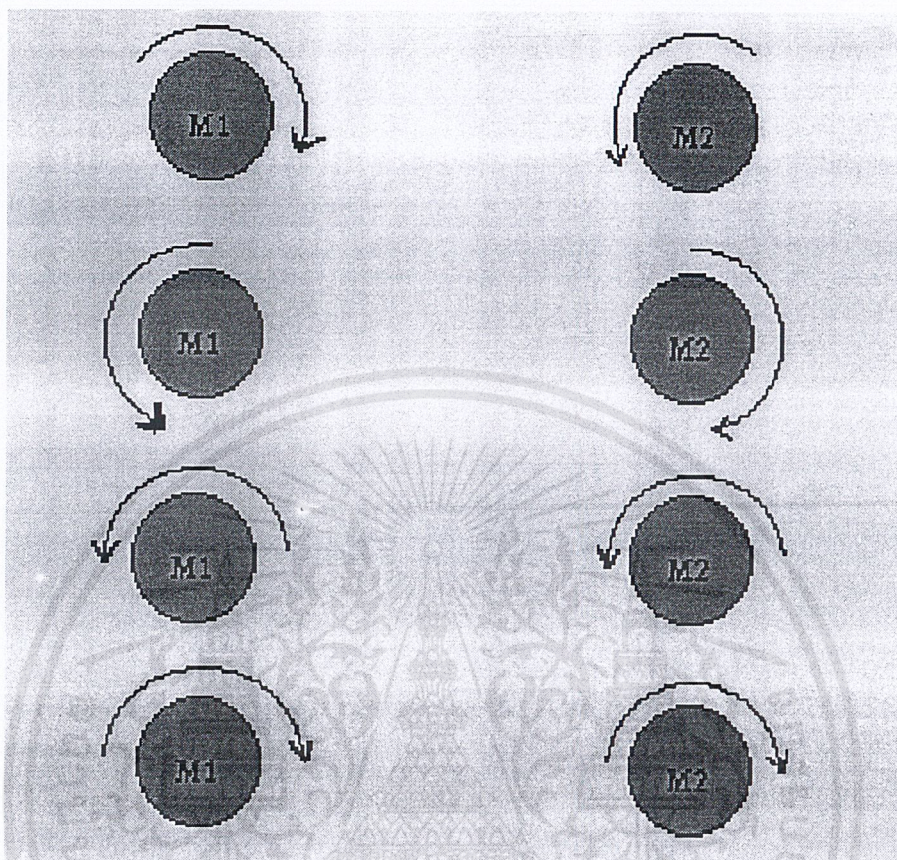
กระแสตรง

คุณสมบัติของไดโอด L293D มีดังนี้

- กระแสเอาต์พุตสูงสุด 1.2A
- สะดวกและง่ายในการนำไปใช้
- อินพุตพัลส์เวดจ์ 0-1.5V
- การเชื่อมต่อระหว่างขาต่างๆอยู่ในตัวไดโอด



รูปที่3.2 Block Diagram ของไดโอดL293D



รูปที่ 3.3 แสดงทิศทางการหมุนของมอเตอร์ควบคุมการเคลื่อนที่แกน X และ Y

จากรูปที่ 3.3

ในทิศทางการหมุนมอเตอร์แบบแรก จะทำให้แกนกลเคลื่อนที่ในแนวแกน Y ทิศขึ้น

ในทิศทางการหมุนมอเตอร์แบบที่2 จะทำให้แกนกลเคลื่อนที่ในแนวแกน Y ทิศลง

ในทิศทางการหมุนมอเตอร์แบบที่3 จะทำให้แกนกลเคลื่อนที่ในแนวแกน X ทิศซ้ายมือ

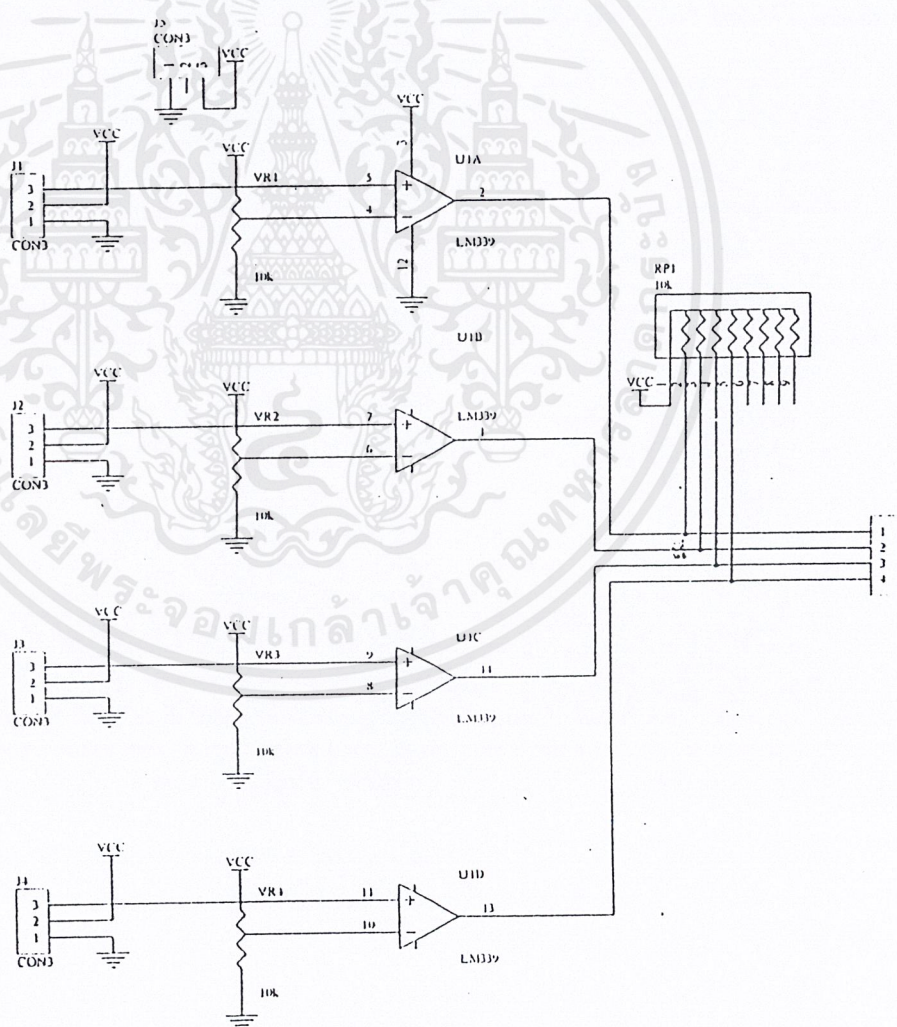
ในทิศทางการหมุนมอเตอร์แบบที่4 จะทำให้แกนกลเคลื่อนที่ในแนวแกน X ทิศขวามือ

ดังนี้

วงจรComparator

เลือกใช้Voltage comparator เบอร์ LM339 ซึ่งมีข้อดีและคุณสมบัติ

- มีความแม่นยำสูง
- กระแสไบอัสต่ำสุดด้านอินพุต = 25nA
- กระแสชดเชยต่ำสุดด้านอินพุต = 5nA
- โวลต์เดจชดเชยต่ำสุดด้านอินพุต = 3mV

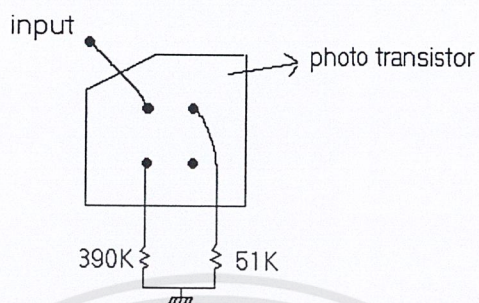


รูปที่3.4 Conection Diagram ของวงจร Comparator

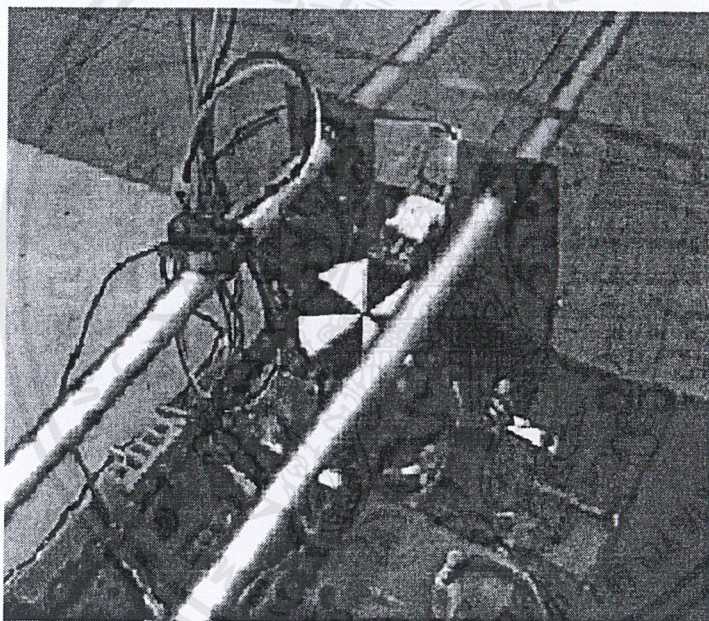
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

อุปกรณ์ตรวจจับแสง(Opto Devices)

เลือกใช้โฟโตทรานซิสเตอร์

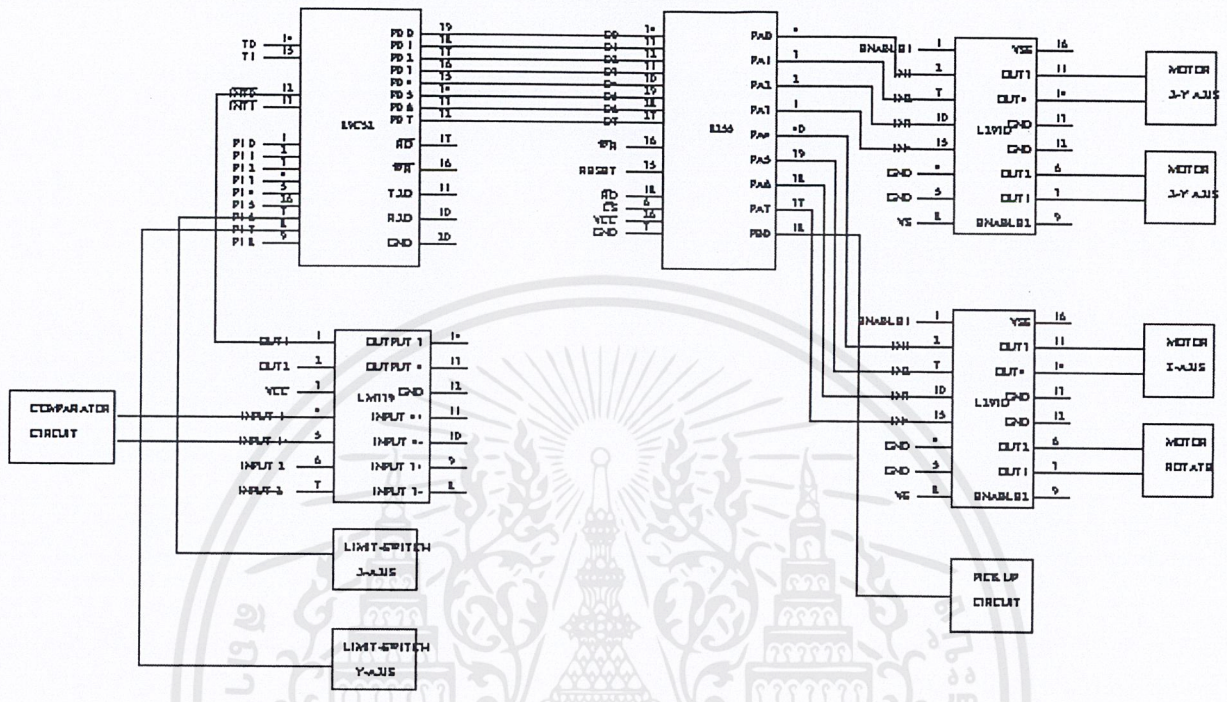


รูปที่3.5 วงจรSensor opto



รูปที่ 3.6 แสดงส่วนประกอบของอุปกรณ์ตรวจจับแสง

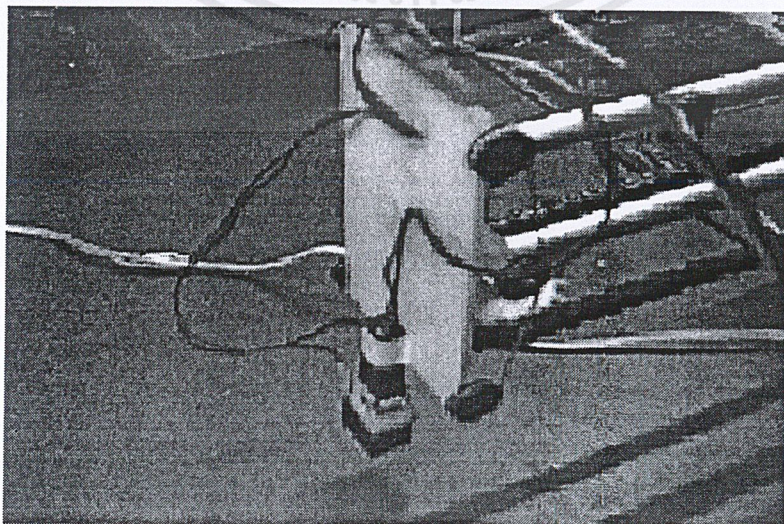
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.7 วงจรรวมของแขนกลต่อเศษภาพ

ส่วนจับชิ้นภาพ

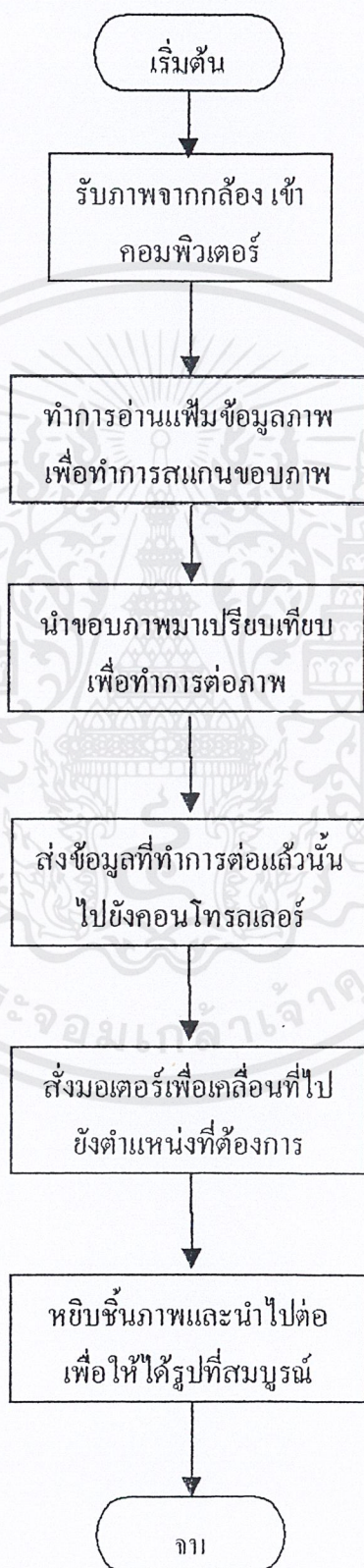
ใช้หลักการของสนามแม่เหล็ก โดยการปล่อยกระแสไฟฟ้าผ่านขดลวดที่พันรอบแกนเหล็กซึ่งจะทำให้เกิดการเหนี่ยวนำของสนามแม่เหล็ก ส่งผลให้ส่วนจับชิ้นภาพมีคุณลักษณะเป็นสนามแม่เหล็ก ทำให้สามารถดูดชิ้นภาพที่มีแผ่นเหล็กขนาดเล็กติดอยู่ได้



รูปที่ 3.8 ที่จับชิ้นภาพ

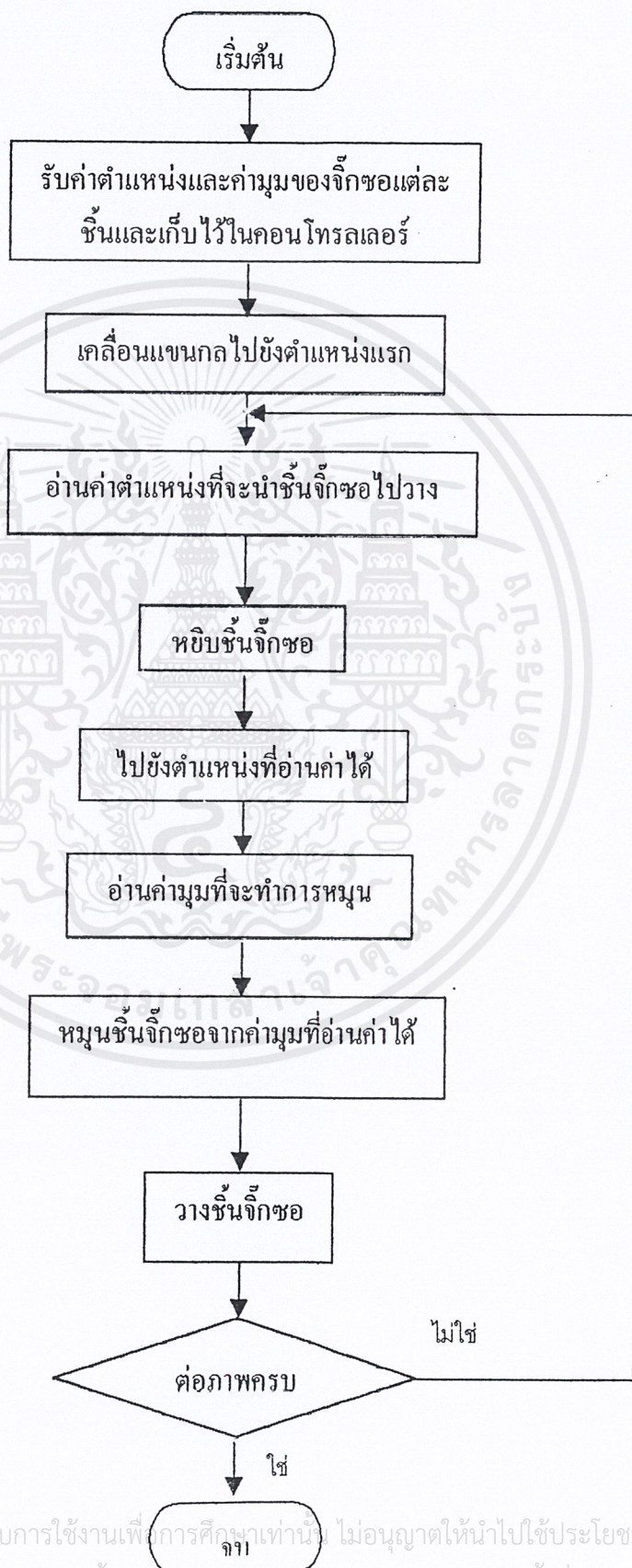
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนผังลำดับงานโดยรวม



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนผังลำดับงานของคอนโทรลเลอร์

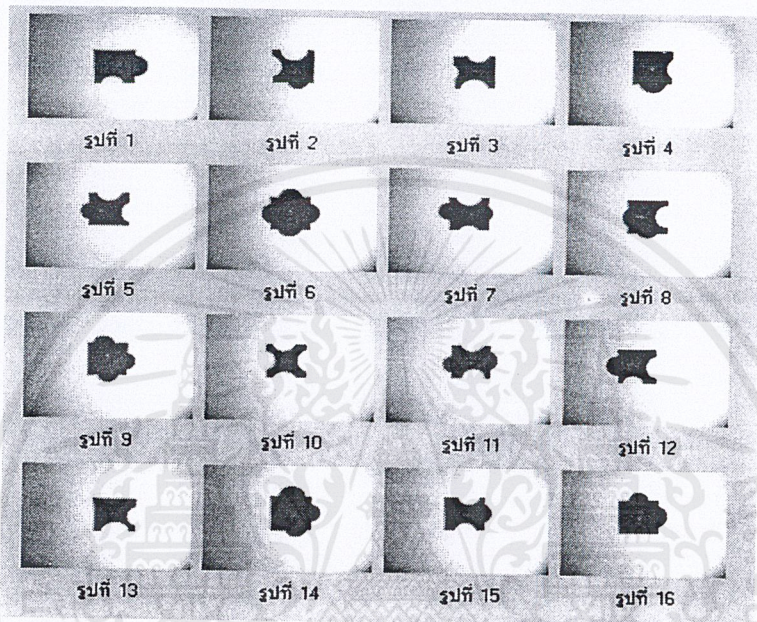


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

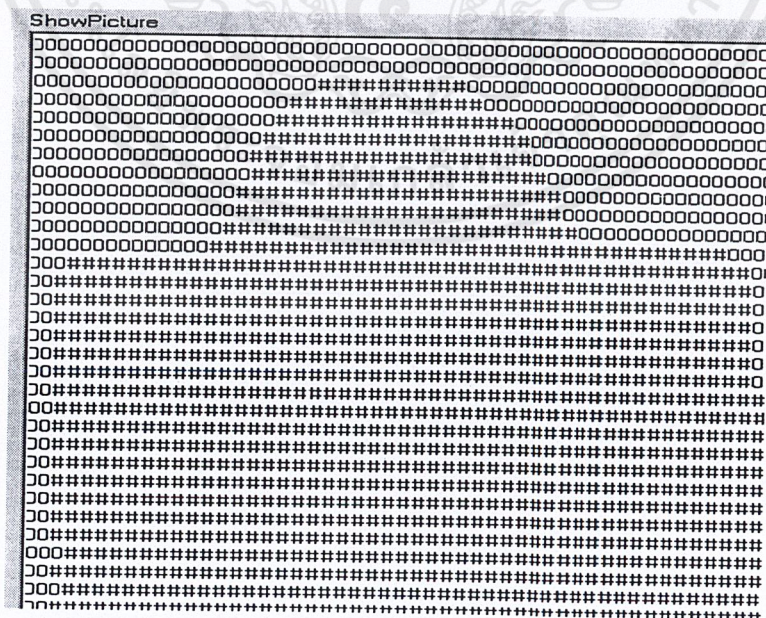
3.2 การคำนวณ

การแบ่งส่วนในการเขียนโปรแกรมแบ่งออกเป็น 4 ส่วน ดังนี้

1. รับไฟล์ภาพที่ถ่ายจากกล้อง แล้วทำการหา Matrix ขอบ รวมทั้งบอกพิกัดของขอบต่าง ๆ ด้วย

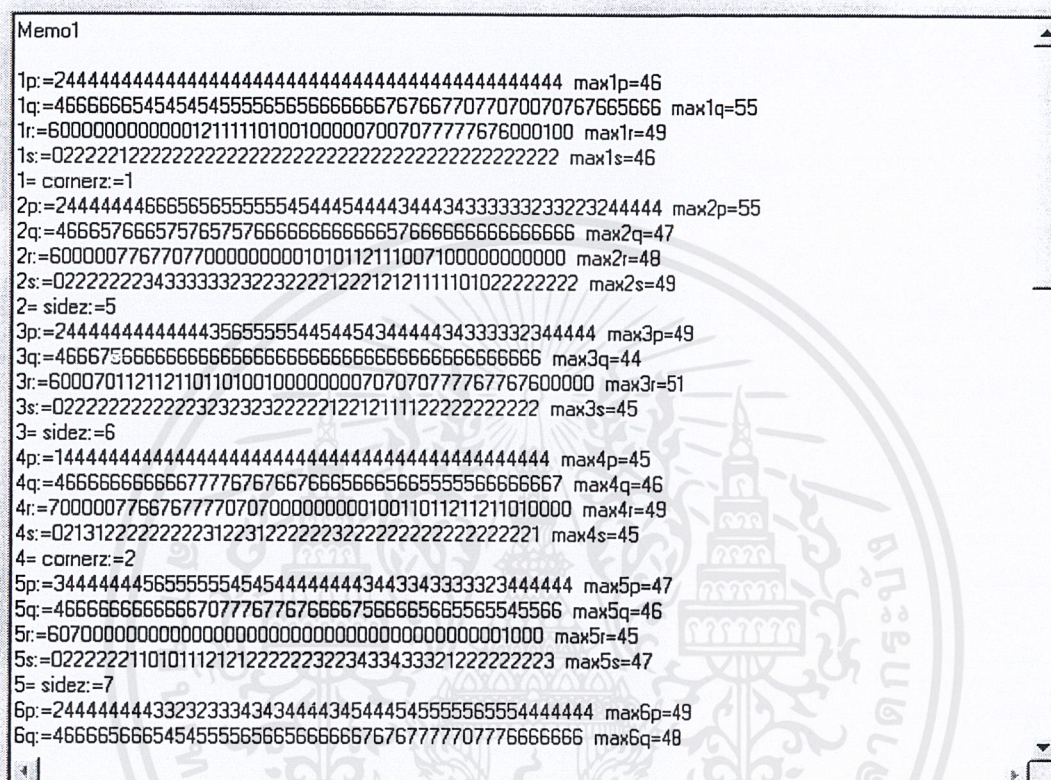


รูปที่ 3.9 แสดงชิ้นส่วนภาพที่นำมาต่อ



เอกสารนี้เป็นเอกสารที่รูปที่ 3.10 บกแสดงการแปลงสัญญาณวิดีโอเป็นรูปภาพให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. นำ Matrix ขอบมาแบ่งเป็น 4 ด้าน ว่าแต่ละด้านมี Matrix เป็นอย่างไร ทั้งสรุปด้วยว่า
 เศษภาพที่นำมาต่อ นั้น เป็นด้าน ขอบ มุม หรือตรงกลาง



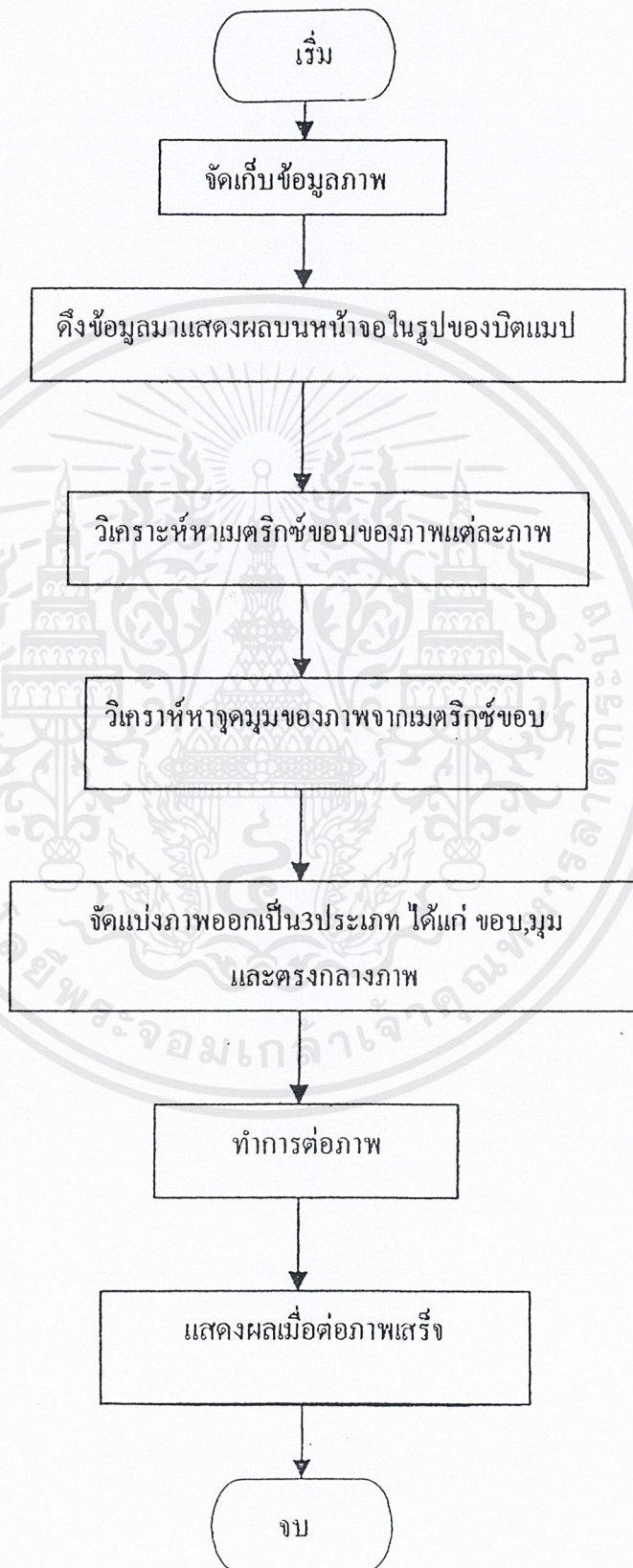
รูปที่ 3.11 แสดงการตัดขอบภาพ

3. นำ Matrix ขอบที่ได้แต่ละรูป มาต่อด้านขอบ และ มุม ให้เรียบร้อย

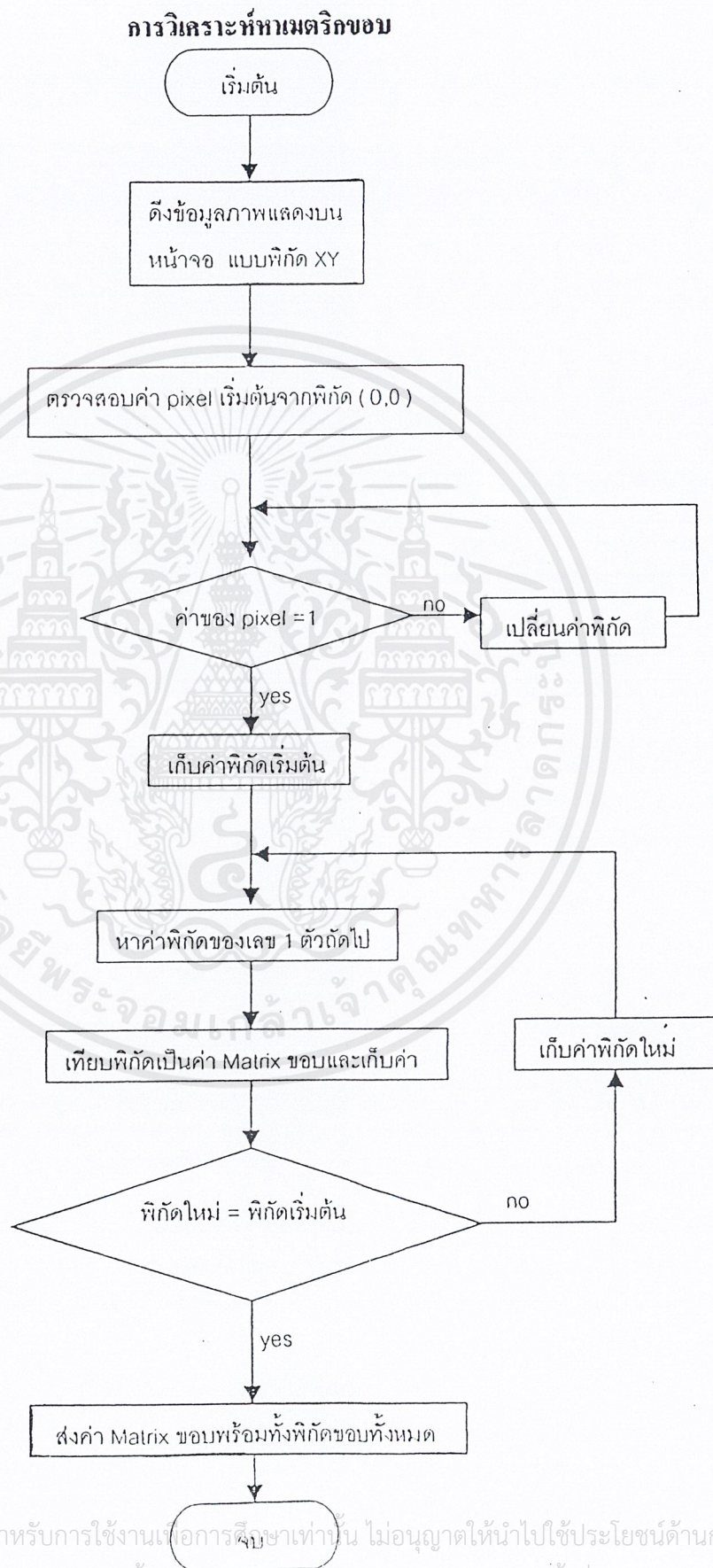
4. นำเศษภาพที่เหลือมาต่อกันให้ครบ

และเมื่อเขียนโปรแกรมแต่ละส่วนเรียบร้อยแล้ว ก็จะได้โปรแกรมแต่ละส่วนมารวมกัน
 เพื่อนำไปใช้ควบคุมแขนกลต่อไป

Flow Chart โดยรวมของโปรแกรม

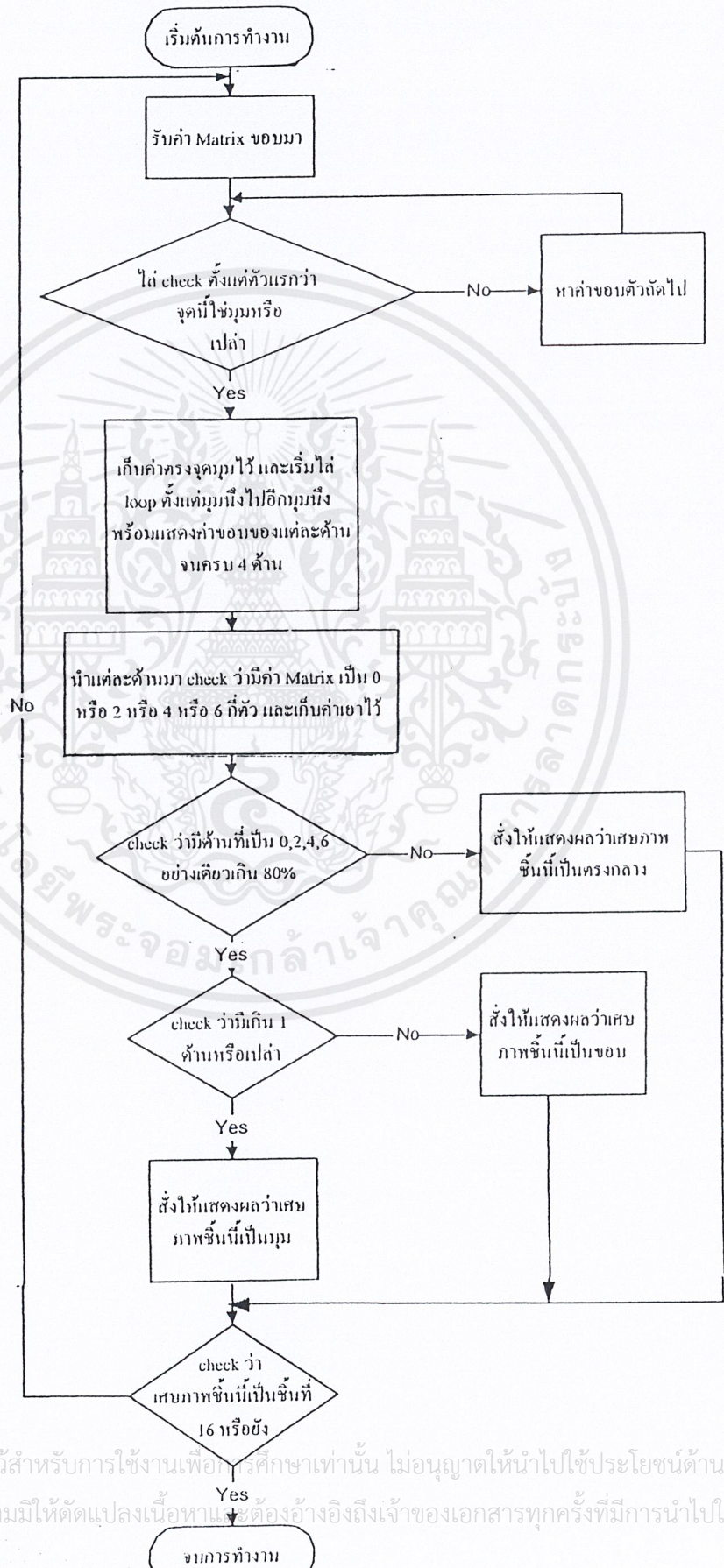


เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



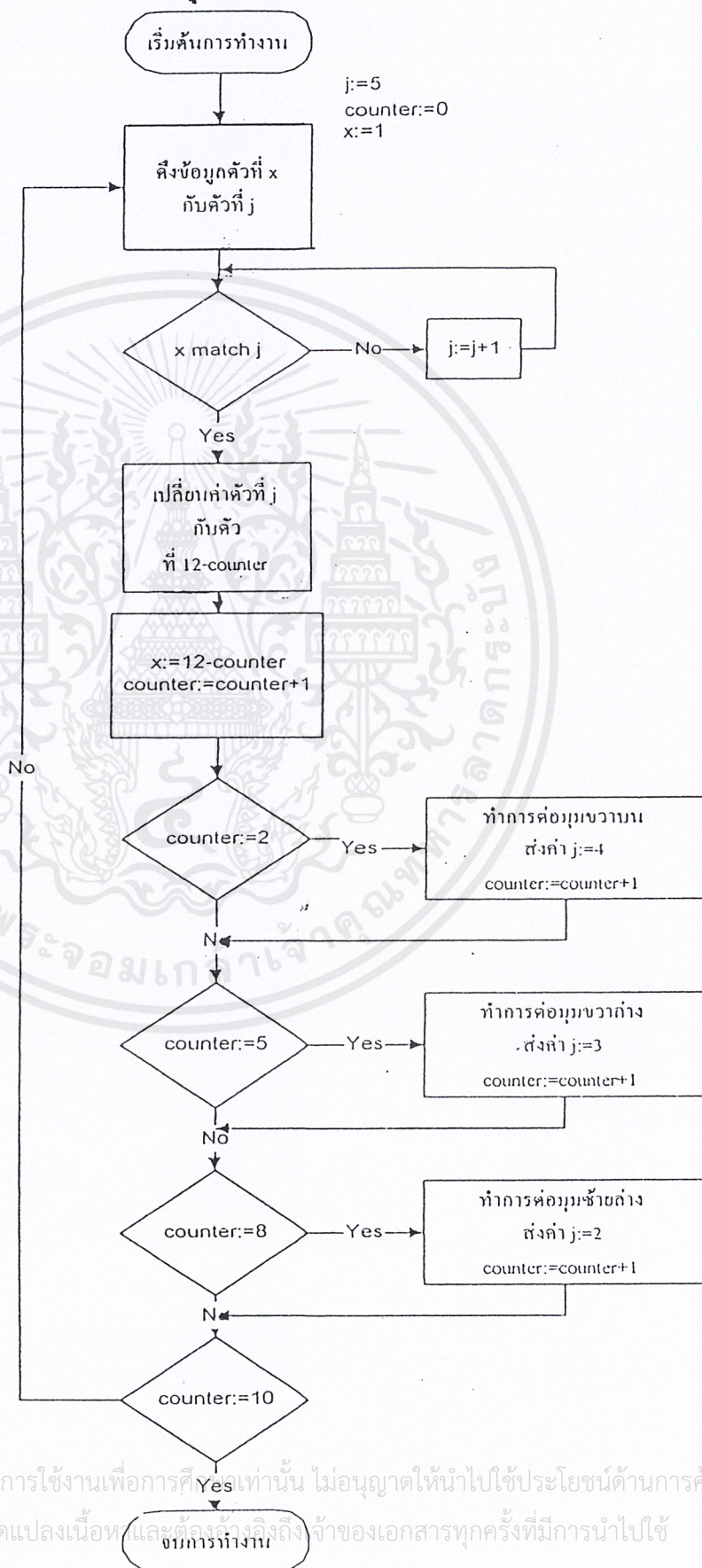
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การจัดแบ่งด้านและการหาเมตริกในแต่ละด้าน



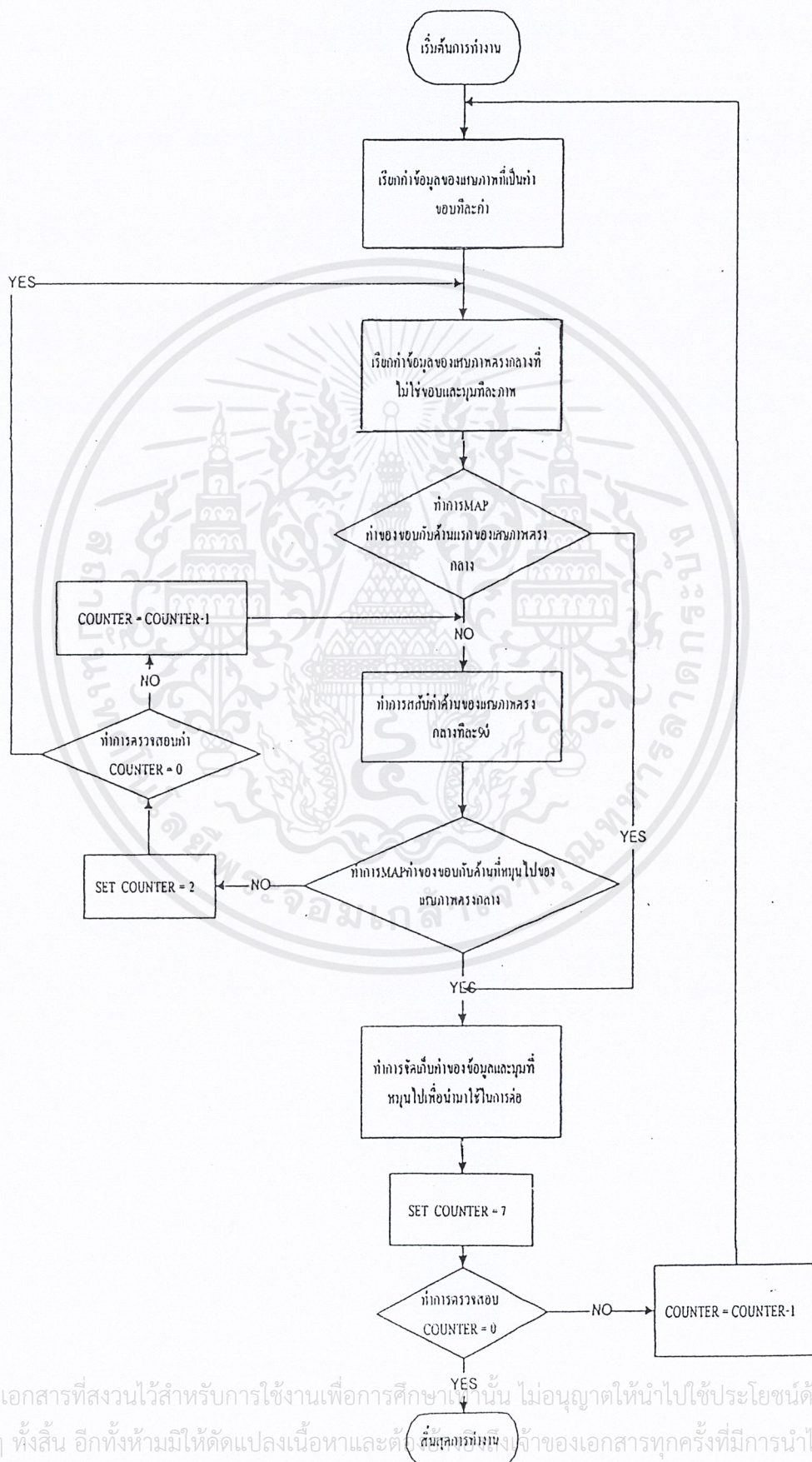
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การต่อมูและขอบภาพ



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษานั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

การต่อภาพตรงกลาง



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและข้อมูลของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลองและผลการทดลอง

จากหัวข้อการคำนวณและการสร้างที่กล่าวมาข้างต้นแสดงให้เห็นว่าการแบ่งทำงานในโครงการนี้แบ่งออกเป็นส่วนใหญ่ๆได้ 2 ส่วน คือ ส่วนของซอฟต์แวร์และฮาร์ดแวร์ เราแบ่งการทดลองและผลการทดลองเป็น 2 ส่วน ดังนี้

4.1 การทดลองในส่วนซอฟต์แวร์

แบ่งออกเป็น 3 ส่วนหลัก

โปรแกรม Delphi ที่ใช้ในการประมวลผลภาพ

การทดลอง ในส่วนนี้เราได้แบ่งโปรแกรมออกเป็นส่วนเพื่อให้ง่ายต่อการเขียน เช่น ส่วนของการแบ่งขอบภาพ หรือส่วนของการหาชิ้นส่วนภาพที่เป็นขอบและมุม ดังนั้นในขั้นตอนแรกเราจึงต้องทำการทดลองโปรแกรมในแต่ละส่วนก่อนเมื่อโปรแกรมในแต่ละส่วนทำงานได้ตามจุดประสงค์แล้ว ขั้นตอนต่อไปจึงนำโปรแกรมมารวมกันและทำการทดลองในส่วนของการโปรแกรมทั้งหมด

ผลการทดลอง ในการทดลองส่วนแรกในแต่ละโปรแกรม ผลที่ได้ไม่เป็นไปตามที่ต้องการเนื่องจากการวางกล้องมีการกำหนดระยะและมุมของกล้องที่ไม่แน่นอนพอ จึงทำให้ข้อมูลของภาพที่ได้จากกล้องวงจรปิดมีความผิดพลาดไม่ตรงกับข้อมูลภาพจริง ทำให้เกิดปัญหาในการประมวลผลของโปรแกรมส่งผลให้ข้อมูลที่ได้จากการประมวลผลผิดพลาด จึงต้องทำการปรับเก็บข้อมูลของภาพใหม่โดยให้มีระยะและมุมของกล้องที่แน่นอน

ในขั้นตอนของการทดลองโปรแกรมทั้งหมดหลังจากการนำมารวมกันแล้ว ผลในการทดลองครั้งแรกคือโปรแกรมในแต่ละส่วนไม่สามารถส่งข้อมูลระหว่างกันได้ เนื่องจากได้มีการตั้งชื่อตัวแปรซ้ำกันในแต่ละส่วนทำให้โปรแกรม Delphi สับสนและเกิดความผิดพลาดของข้อมูล จึงต้องทำการเปลี่ยนชื่อของตัวแปรให้ไม่เกิดการซ้ำกัน

โปรแกรม Delphi ในส่วนของการส่งข้อมูลออกพอร์ตอนุกรม

การทดลอง ทำการทดลองเพื่อตรวจสอบว่าข้อมูลส่งออกพอร์ตอนุกรมได้จริงและถูกต้องหรือไม่

ผลการทดลอง ในการทดลองครั้งแรกมีปัญหาเกิดขึ้นมากเนื่องจากการเลือกใช้ Component ในการส่งออกพอร์ตอนุกรมผิด ทำให้เกิดความยุ่งยากในการใช้และข้อมูลที่ส่งออกเกิดความผิดพลาด

ในโปรแกรมDelphi ไม่มีComponentที่รองรับพอร์ตอนุกรม จึงต้องทำการDownload จากอินเทอร์เน็ต(Internet) ซึ่งมีComponentหลายตัวให้เลือกใช้ ทำให้เกิดความยุ่งยากในการพิจารณา ถ้าเลือกใช้Component ที่ไม่เหมาะสมอาจทำให้เกิดปัญหาดังที่กล่าวมาข้างต้น

เมื่อทำการพิจารณาและเปลี่ยนComponentใหม่แล้วทำให้ลดความยุ่งยากและความสับสนในการใช้ไปได้มากและทำให้ข้อมูลที่ส่งผ่านพอร์ตมีความถูกต้อง ในโครงการนี้ทางคณะผู้จัดทำได้เลือกใช้ Component Library Version 2.60 ในการส่งข้อมูลออกพอร์ตอนุกรม

โปรแกรมแอสเซมบลีที่ใช้ประมวลผลในไมโครคอนโทรลเลอร์

การทดลอง ในขั้นตอนนี้เกิดความยุ่งยากในการทดลองมาก ต้องทำการตรวจสอบว่าส่วนของฮาร์ดแวร์ทำงานตามคำสั่งที่ออกจากไมโครคอนโทรลเลอร์ได้ถูกต้องหรือไม่ โดยสังเกตทิศทางการหมุนของมอเตอร์ทั้ง3ตัวและระยะที่เลื่อนไปของแกนแกน เป็นต้น

ผลการทดลอง ในการทดลองครั้งแรกเกิดความสับสนระหว่างส่วนของไมโครคอนโทรลเลอร์กับฮาร์ดแวร์ คือเมื่อไมโครคอนโทรลเลอร์ส่งสัญญาณมาที่ขาT1 เพื่อทำการInterupเพื่อให้มอเตอร์หยุดหมุน จะเกิดความคลาดเคลื่อนของระยะเวลาการทำงานของ2ส่วน ทำให้ระยะทางของแกนแกนที่เลื่อนผิดพลาด และปัญหาอีกประการหนึ่งคือ สัญญาณของSensor (Opto Device) ไม่ละเอียดพอ

4.2การทดลองในส่วนของฮาร์ดแวร์

การทดลองในส่วนของฮาร์ดแวร์พบปัญหาในหลายๆส่วนเช่น ความฝืดที่เกิดขึ้นทำให้มอเตอร์มีกำลังไม่พอ ความฝืดที่เกิดขึ้นเกิดจากหลายสาเหตุ คือ ความฝืดในส่วนของแกนอลูมิเนียม และความฝืดในการที่ดึงโซ่ตึงเกินไป โซ่หลุดจากเฟืองเกิดจากการที่ติดตั้งเฟือง ไม่ได้ระดับ และโซ่หย่อนเกินไป เกิดความผิดพลาดกับการทำงานของออปโตแสงซึ่งเกิดจากการที่ระยะระหว่างSensor กับตัวEncoderไม่เหมาะสม

4.3ผลการทดลอง

ในโครงการเครื่องต่อภาพจากเศษภาพนี้ได้แสดงผลการทดลองออกเป็น 3 ส่วน คือ ผลการทดลองในด้านระยะทาง , มุม และจำนวนชิ้นภาพที่หีบได้

ผลการทดลองจำนวน 30 ครั้ง แสดงเป็นตารางดังนี้

* การทดลองในแต่ละครั้งเป็นการทดลองต่อภาพ 16 ภาพ

** ความแม่นยำในตำแหน่ง คือ การที่ขึ้นภาพวางซ้ำที่เดิม

***ความถูกต้องในตำแหน่ง คือ การที่ขึ้นภาพวางในตำแหน่งที่ถูกต้อง

ผลการทดลองครั้งที่	จำนวนขึ้นภาพที่หมุนมุมถูกต้อง	จำนวนขึ้นภาพที่มีความแม่นยำในตำแหน่ง	จำนวนขึ้นภาพที่มีความถูกต้องในตำแหน่ง	จำนวนขึ้นภาพที่ทำการหยิบได้	หมายเหตุ
1	13	10	8	9	
2	11	11	7	12	
3	10	13	9	13	
4	11	11	7	12	
5	10	14	9	13	
6	9	9	6	7	*แกนแขนเกิดการสะดุด
7	9	14	8	11	
8	10	10	10	9	
9	12	12	8	12	
10	11	15	9	11	
11	12	13	8	12	
12	14	13	9	14	
13	13	12	10	13	
14	13	11	8	12	
15	12	12	9	13	
16	14	11	8	12	

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ผลการ ทดลอง ครั้งที่	จำนวนชิ้น ภาพที่หมุน มุมถูกต้อง	จำนวนชิ้นภาพ มีความแม่นยำ ในตำแหน่ง	จำนวนชิ้นภาพ ที่มีความถูกต้อง ในตำแหน่ง	จำนวนชิ้น ภาพที่ทำ การหยิบได้	หมายเหตุ
17	13	10	9	12	
18	14	11	8	11	
19	12	10	8	5	*เกิดการขัดข้อง ของวงจรที่ขับเคลื่อน เตอร์ในแกนZ
20	12	11	9	13	
21	14	12	10	12	
22	13	13	10	12	
23	14	12	8	13	
24	15	13	10	14	
25	14	12	8	11	
26	12	12	7	10	
27	13	11	10	12	
28	14	10	10	13	
29	12	11	7	12	
30	13	12	8	11	

ตารางที่4.1 ผลการทดลอง

เอกสารนี้เป็นเอกสารที่จัดทำขึ้นเพื่อการวิจัยและพัฒนาเท่านั้น ไม่ควรนำข้อมูลไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้คัดลอกเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความถูกต้องในการหมุนภาพ = 66.2%

ความถูกต้องทางตำแหน่ง = 50.6%

ความแม่นยำทางตำแหน่ง = 74.38%

เปอร์เซ็นต์ในการหิบบนภาพได้ในการต่อแต่ละครั้ง = 68.12%

4.4 ปัญหาและการแก้ไข

ความผิดพลาดในการเคลื่อนที่แนวแกน X และแกน y

1. การใช้ Counter นับจำนวนรอบ



แก้ไขโดยทำให้เกิดสัญญาณเข้าที่ขา INTO โดยให้เกิดสัญญาณ Interrupt ที่ขอบขาลง เพื่อเขียน โปรแกรมนับจำนวนรอบแทน

2. สัญญาณที่ตรวจจำนวนรอบจาก Opto

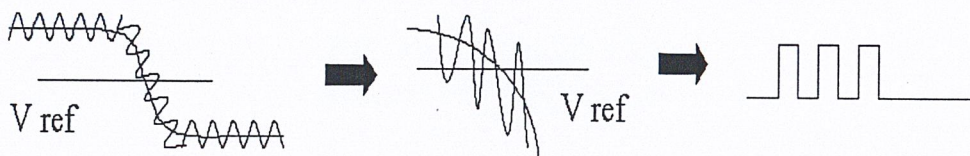
ทางด้าน Mechanic

- ระยะห่างระหว่าง Opto กับ Encoder ไม่เหมาะสม
- การสะท้อนแสงที่ขอบสีที่ Encoder

การแก้ไขทางด้าน Mechanic ก็ต้องเขียน โปรแกรมนับจำนวน Pulse ที่ได้จาก Opto ต่อ 1 Strp มาทดสอบ และ ที่แถบสีดำต้องไม่มีการสะท้อนแสงแต่จากการใช้สติกเกอร์ทำให้เกิดการสะท้อนแสง จึงต้องเปลี่ยนเป็นใช้กระดาษสีดำแทน

ทางด้าน Electronic

- เกิด noise จากการเปลี่ยนแปลงของแสงจากภายนอกที่ตกกระทบกับ Encoder
- เกิด noise จากการรวมวงจรขับเคลื่อนมอเตอร์ กับไมโครคอนโทรลเลอร์

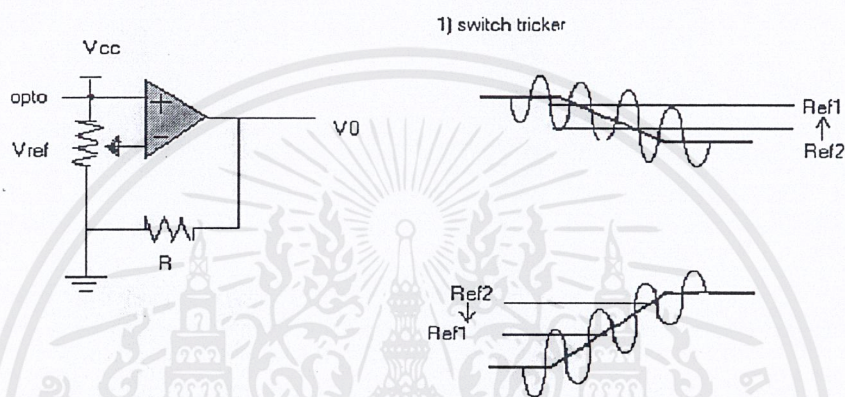


รูปที่ 4.1 ลักษณะเออร์พุทของ comparator เมื่อมีสัญญาณรบกวน

การแก้ไขแบ่งออกได้เป็น 2 ทางได้แก่

1. ทางด้านวงจร Electronic

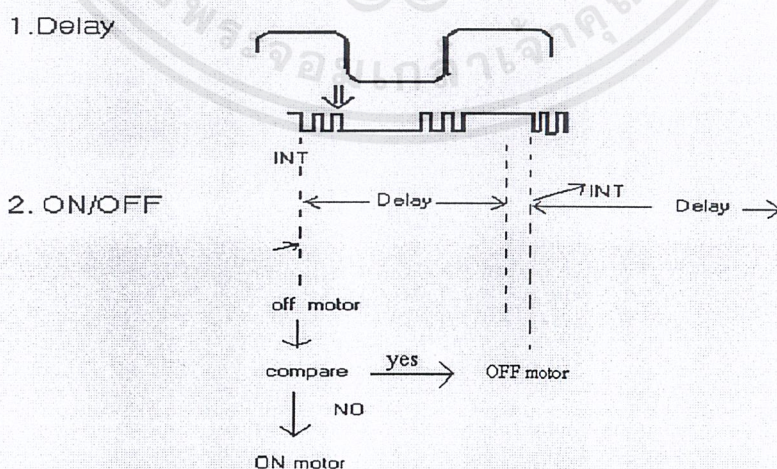
SWITCH TRICKER



รูปที่ 4.2 วงจร Switch Tricker

2. ทางด้านโปรแกรม

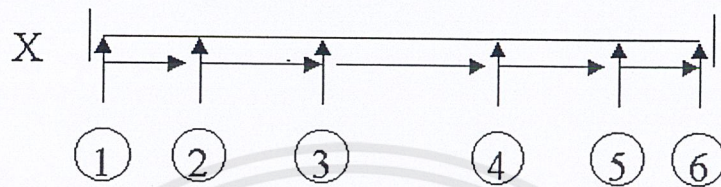
การแก้ไขโดยใช้โปรแกรม



รูปที่ 4.3 การปรับปรุงสัญญาณรบกวนทางด้านซอฟต์แวร์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

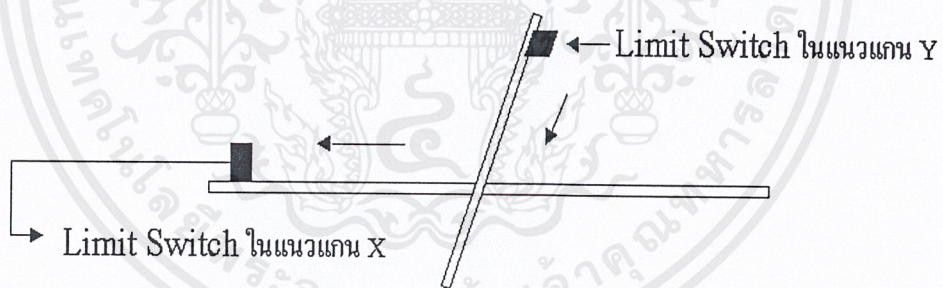
3. ความผิดที่ตำแหน่งต่างๆของแขนกลมีไม่เท่ากัน



รูปที่ 4.4 ความผิด ณ ตำแหน่งต่างๆของแขนกล

- เมื่อมีการสั่ง ON/OFF มอเตอร์จะมีความผิดพลาดสะสมมากขึ้นเรื่อยๆ
- Loop Delay เดียวกันทำให้ระยะไม่เท่ากันที่ตำแหน่งต่างกัน

แก้ไขโดยให้มีการติด Limit Switch ไว้ที่จุดเริ่มต้น และให้มีการกลับไปจุดเริ่มต้นทุกๆ 4 ภาพ



รูปที่ 4.5 ลักษณะการติด Limit Switch ในแกน X และ Y

บทที่ 5

บทวิจารณ์และสรุป

จากการทำโครงที่ผ่านมามีความรู้ในเรื่อง B/W CCD CAMERA การนำสัญญาณภาพมาประยุกต์ใช้งาน การอ่านไฟล์ภาพจากคอมพิวเตอร์มาแสดงเป็น BIT MAP การสแกนขอบภาพเพื่อการต่อภาพ การส่งข้อมูลผ่านพอร์ทอนุกรมและการควบคุมมอเตอร์ให้เคลื่อนที่ไปยังตำแหน่งที่ต้องการ

จากการทำงาน ผู้จัดทำได้พบปัญหาต่าง ๆ มากมาย ไม่ว่าจะเป็นปัญหาด้าน ฮาร์ดแวร์ และซอฟต์แวร์ ผู้จัดทำได้พยายาม วิเคราะห์แนวคิดในการแก้ปัญหาต่าง ๆ ดังแสดงไว้แล้วข้างต้น โครงงานที่สามารถการอ่านไฟล์ภาพที่ถ่ายจากกล้อง แล้วนำมาวิเคราะห์เพื่อทำการต่อภาพให้สมบูรณ์ และ การควบคุมมอเตอร์ในการหยิบชิ้นภาพเพื่อต่อภาพให้ได้ภาพที่สมบูรณ์ แต่ยังมีข้อผิดพลาดอยู่บ้างในบางส่วน เช่น ความแม่นยำในการต่อภาพ

จากที่ได้กล่าวมาแล้ว โครงงานนี้ยังต้องมีการพัฒนาและปรับปรุงในส่วนต่าง ๆ อีก เช่น

- การปรับลักษณะของสัญญาณภาพให้มีความละเอียดมากขึ้น
- การควบคุมแขนกล และ ส่วน HARD WARE ให้มีความแม่นยำและเที่ยงตรงมากขึ้น รวมไปถึงการแก้ไขเรื่องความผิดในชิ้นส่วนต่าง ๆ
- การปรับความละเอียดของ ENCODER และ SENSOR
- การแก้ไขโปรแกรมส่วนประมวลผลให้มีความยืดหยุ่นในการทำงานมากขึ้น

กิตติกรรมประกาศ

ปริญญานิพนธ์ฉบับนี้สำเร็จได้ด้วยดีจากความช่วยเหลือจากอาจารย์และบุคคลหลายท่าน
ขอขอบคุณอาจารย์สุมิตร พนาอุดมทรัพย์ อาจารย์ที่ปรึกษาและ รศ.ดร.โยธิน เปรมปราณี
รัชต์ ที่ได้กรุณาให้คำปรึกษาชี้แจงแนวทางในการแก้ไขปัญหาดังกล่าวอย่างทันท่วงที ตลอดจนสถานที่
และอุปกรณ์ที่ใช้ในโครงการและปริญญานิพนธ์ และความเอาใจใส่อย่างดีเสมอมา

ขอขอบคุณภาควิชาระบบควบคุม อาจารย์ และเจ้าหน้าที่ทุกท่านที่ช่วยอำนวยความสะดวก
เอื้อเพื่ออุปกรณ์และให้คำปรึกษาที่ดีเสมอมา

ขอขอบคุณเพื่อนๆห้อง5 (ก๊วก, โน้ต, เต๋า, หยก, เมย์) ที่ช่วยเหลือด้านอุปกรณ์ ให้ความช่วย
เหลือและให้กำลังใจตลอดการทำงาน

ขอขอบคุณ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบังสำหรับประสบการณ์
ชีวิต ความรู้ และทุกสิ่งทุกอย่างตลอดระยะเวลา 4 ปี

ขอขอบพระคุณสำหรับบุญคุณอันยิ่งใหญ่ที่สุดคือ คุณพ่อ คุณแม่ และครอบครัวของพวกเรา
ทั้งสี่ ที่คอยห่วงใยและให้กำลังใจ ตลอดจนให้ความช่วยเหลือในทุกๆเรื่องแก่พวกเราเสมอมาไม่เคย
เปลี่ยนแปลง

นางสาวกัญช์ อมตวรกุล

นายพิพัฒน์ ทระเกียรติศักดิ์

นางสาวสมฤดี แซ่เล่า

นายสันติ หกกิจ โสณอุดมศักดิ์

หนังสืออ้างอิง

1. ชัยวัฒน์ ลิ้มพรวิตรวิไล ,” คู่มืออิเล็กทรอนิกส์ “,บ.ซีเอ็ดยูเคชั่น จำกัด ,หน้า128 ,2539
2. ค.ร. พีรศักดิ์ วรสุนทรโรสก ,” เครื่องจักรไฟฟ้า1 วงจรแม่เหล็กและเครื่องจักรไฟฟ้ากระแสตรง” , สถาบันเทคโนโลยีพระจอมเกล้า วิทยาเขตพระนครเหนือ , หน้า 121-157 , 2519
3. ดร. สิทธิชัย โกไลยอุดม ,”ทฤษฎีและการคำนวณวงจรอิเล็กทรอนิกส์ “,บ.ซีเอ็ดยูเคชั่น จำกัด , หน้า133.-334 ,2521
4. นุกูล กระจ่าย ,” เทอร์โบปาสคาล “ ,บ.ซีเอ็ดยูเคชั่น จำกัด , 500 หน้า , 2541
5. บ.อีทีที จำกัด ,” คู่มือไอซี ไมโครโปรเซสเซอร์ “ ,บ.อีทีที จำกัด , หน้า 145-180 , 2540
6. พันจันทร์ ธนวัฒนเสถียร ,” ไมโครโปรเซสเซอร์ ทฤษฎีและการประยุกต์ใช้งาน “ ,บ.ซีเอ็ดยูเคชั่น จำกัด , หน้า 557-573 , 2537
7. ยืน ภูสุวรรณ ,” ไมโครโปรเซสเซอร์ ไมโครคอมพิวเตอร์ “ ,บ.ซีเอ็ดยูเคชั่น , หน้า 95-148,2533
8. สัจจะ จรัสรุ่งเรือง ,” Delphi 5.0 “ ,บ.ด้านสุทธาการพิมพ์ จำกัด , 531 หน้า , 2540
9. Charles M. Gilmore ,” Microprocessor Principles and Applications “ , 540 P ,1995

ภาคผนวก ก

```
unit before;

interface

uses
  Windows, Messages, SysUtils, Classes, Graphics,
  Controls, Forms, Dialogs,
  StdCtrls, ExtCtrls;

type
  TForm1 = class(TForm)
    Image1: TImage;
    Image2: TImage;
    Image3: TImage;
    Image4: TImage;
    Image5: TImage;
    Image6: TImage;
    Image7: TImage;
    Image8: TImage;
    Image9: TImage;
    Image10: TImage;
    Image11: TImage;
    Image12: TImage;
    Image13: TImage;
    Image14: TImage;
    Image15: TImage;
    Image16: TImage;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;

    Label10: TLabel;
    Label11: TLabel;
    Label12: TLabel;
    Label13: TLabel;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label34: TLabel;
    Label35: TLabel;
    Label36: TLabel;
    Label37: TLabel;
    Label38: TLabel;
    Label39: TLabel;
    Label40: TLabel;
    Label41: TLabel;
    Label42: TLabel;
    Label43: TLabel;
    Label44: TLabel;
    Button1: TButton;
    Label28: TLabel;
    Label29: TLabel;
    Label30: TLabel;
    Label31: TLabel;
    Label25: TLabel;
    Button2: TButton;

    procedure Button1Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);

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

var
  Form1: TForm1;

implementation

uses findmat;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

{$R *.DFM}

procedure TForm1.Button1Click(Sender: TObject);
begin
    Image1.Picture.Bitmap.LoadFromFile('c:\Ks001.bmp');
    Image2.Picture.Bitmap.LoadFromFile('c:\Ks002.bmp');
    Image3.Picture.Bitmap.LoadFromFile('c:\Ks003.bmp');
    Image4.Picture.Bitmap.LoadFromFile('c:\Ks004.bmp');
    Image5.Picture.Bitmap.LoadFromFile('c:\Ks005.bmp');
    Image6.Picture.Bitmap.LoadFromFile('c:\Ks006.bmp');
    Image7.Picture.Bitmap.LoadFromFile('c:\Ks007.bmp');
    Image8.Picture.Bitmap.LoadFromFile('c:\Ks008.bmp');
    Image9.Picture.Bitmap.LoadFromFile('c:\Ks009.bmp');
    Image10.Picture.Bitmap.LoadFromFile('c:\Ks010.bmp');
    Image11.Picture.Bitmap.LoadFromFile('c:\Ks011.bmp');
    Image12.Picture.Bitmap.LoadFromFile('c:\Ks012.bmp');
    Image13.Picture.Bitmap.LoadFromFile('c:\Ks013.bmp');
    Image14.Picture.Bitmap.LoadFromFile('c:\Ks014.bmp');
    Image15.Picture.Bitmap.LoadFromFile('c:\Ks015.bmp');
    Image16.Picture.Bitmap.LoadFromFile('c:\Ks016.bmp');
end;

procedure TForm1.Button2Click(Sender: TObject);
begin
    form2.ShowModal
end;

end.

unit display;

interface

uses
    Windows, Messages, pat, SysUtils, Classes, Graphics,
    Controls, Forms, Dialogs,
    StdCtrls, ExtCtrls, CPort;

type
    TForm5 = class(TForm)
        Image1: TImage;
        Image2: TImage;
        Image3: TImage;
        Image4: TImage;
        Image5: TImage;
        Image6: TImage;
        Image7: TImage;
        Image8: TImage;
        Image9: TImage;
        Image10: TImage;
        Image11: TImage;
        Image12: TImage;
        Image13: TImage;
        Image14: TImage;
        Image15: TImage;
        Image16: TImage;
        Label1: TLabel;
        Label2: TLabel;
        Label3: TLabel;
        Label4: TLabel;
        Label5: TLabel;
        Label6: TLabel;
        Label7: TLabel;
        Label8: TLabel;
        Label9: TLabel;
        Label10: TLabel;
        Label11: TLabel;
        Label12: TLabel;
        Label13: TLabel;
        Label14: TLabel;
        Label15: TLabel;
        Label16: TLabel;
        Label17: TLabel;
        Label18: TLabel;
        Label19: TLabel;
        Label20: TLabel;
        Label21: TLabel;
        Label22: TLabel;
        Label23: TLabel;
        Label24: TLabel;
        Label25: TLabel;
        Label26: TLabel;
        Label27: TLabel;
        Label28: TLabel;
        Label29: TLabel;
    end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Label30: TLabel;	
Label31: TLabel;	private
Label32: TLabel;	{ Private declarations }
Label33: TLabel;	public
Label34: TLabel;	{ Public declarations }
Label35: TLabel;	end;
Label36: TLabel;	
Label37: TLabel;	var
Label38: TLabel;	Form5: TForm5;
Label39: TLabel;	function name1(x:integer):string;
Label40: TLabel;	function send(x:integer):string;
Label41: TLabel;	function send1(x:integer):string;
Label42: TLabel;	
Label43: TLabel;	implementation
Label44: TLabel;	{SR *.DFM}
Label45: TLabel;	
Label46: TLabel;	function name1(x:integer):string;
Label47: TLabel;	begin
Label48: TLabel;	case x of
Label49: TLabel;	1 : name1 := 'c:\Ks001.bmp';
Label50: TLabel;	2 : name1 := 'c:\Ks002.bmp';
Label51: TLabel;	3 : name1 := 'c:\Ks003.bmp';
Label52: TLabel;	4 : name1 := 'c:\Ks004.bmp';
Label53: TLabel;	5 : name1 := 'c:\Ks005.bmp';
Label54: TLabel;	6 : name1 := 'c:\Ks006.bmp';
Label55: TLabel;	7 : name1 := 'c:\Ks007.bmp';
Label56: TLabel;	8 : name1 := 'c:\Ks008.bmp';
Label57: TLabel;	9 : name1 := 'c:\Ks009.bmp';
Label58: TLabel;	10 : name1 := 'c:\Ks010.bmp';
Label59: TLabel;	11 : name1 := 'c:\Ks011.bmp';
Label60: TLabel;	12 : name1 := 'c:\Ks012.bmp';
Label61: TLabel;	13 : name1 := 'c:\Ks013.bmp';
Label62: TLabel;	14 : name1 := 'c:\Ks014.bmp';
Label63: TLabel;	15 : name1 := 'c:\Ks015.bmp';
Label64: TLabel;	16 : name1 := 'c:\Ks016.bmp';
Button1: TButton;	end;
Button2: TButton;	end;
ComPort1: TComPort;	
Button3: TButton;	
procedure Button1Click(Sender: TObject);	function send(x:integer):string;
procedure Button3Click(Sender: TObject);	begin
procedure Button2Click(Sender: TObject);	case x of
	1 : send := '1';

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

2 : send := '2';
3 : send := '3';
4 : send := '4';
5 : send := '5';
6 : send := '6';
7 : send := '7';
8 : send := '8';
9 : send := '9';
10 : send := 'a';
11 : send := 'b';
12 : send := 'c';
13 : send := 'd';
14 : send := 'e';
15 : send := 'f';
16 : send := '0';
end;
end;

function send1(x:integer):string;
begin
    case x of
        0 : send1 := '0';
        90 : send1 := '1';
        180 : send1 := '2';
        270 : send1 := '3';
    end;
end;

procedure TForm5.Button1Click(Sender: TObject);
begin
    Image1.Picture.Bitmap.LoadFromFile(name1(show1
[1]));
    Image2.Picture.Bitmap.LoadFromFile(name1(show1
[2]));
    Image3.Picture.Bitmap.LoadFromFile(name1(show1
[3]));
    Image4.Picture.Bitmap.LoadFromFile(name1(show1
[4]));
    Image5.Picture.Bitmap.LoadFromFile(name1(show1
[12]));
    Image6.Picture.Bitmap.LoadFromFile(name1(complete
[1]));
    Image7.Picture.Bitmap.LoadFromFile(name1(complete
[2]));
    Image8.Picture.Bitmap.LoadFromFile(name1(show1
[5]));
    Image9.Picture.Bitmap.LoadFromFile(name1(show1
[11]));
    Image10.Picture.Bitmap.LoadFromFile(name1(complete
[4]));
    Image11.Picture.Bitmap.LoadFromFile(name1(complete
[3]));
    Image12.Picture.Bitmap.LoadFromFile(name1(show1
[6]));
    Image13.Picture.Bitmap.LoadFromFile(name1(show1
[10]));
    Image14.Picture.Bitmap.LoadFromFile(name1(show1
[9]));
    Image15.Picture.Bitmap.LoadFromFile(name1(show1
[8]));
    Image16.Picture.Bitmap.LoadFromFile(name1(show1
[7]));
    Label1.Caption := IntToStr(show1[1]);
    Label2.Caption := IntToStr(show1[2]);
    Label3.Caption := IntToStr(show1[3]);
    Label4.Caption := IntToStr(show1[4]);
    Label5.Caption := IntToStr(show1[12]);
    Label6.Caption := IntToStr(show1[13]);
    Label7.Caption := IntToStr(show1[14]);
    Label8.Caption := IntToStr(show1[5]);
    Label9.Caption := IntToStr(show1[11]);
    Label10.Caption := IntToStr(show1[16]);
    Label11.Caption := IntToStr(show1[15]);
    Label12.Caption := IntToStr(show1[6]);
    Label13.Caption := IntToStr(show1[10]);
    Label14.Caption := IntToStr(show1[9]);
    Label15.Caption := IntToStr(show1[8]);
    Label16.Caption := IntToStr(show1[7]);
    Label17.Caption := IntToStr(corer[1]);
    Label18.Caption := IntToStr(corer[2]);
    Label19.Caption := IntToStr(corer[3]);
    Label20.Caption := IntToStr(corer[4]);
    Label21.Caption := IntToStr(corer[12]);
    Label22.Caption := IntToStr(com[1]);
    Label23.Caption := IntToStr(com[2]);

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

Label24.Caption := IntToStr(corn[5]);
Label25.Caption := IntToStr(corn[11]);
Label26.Caption := IntToStr(corn[4]);
Label27.Caption := IntToStr(corn[3]);
Label28.Caption := IntToStr(corn[6]);
Label29.Caption := IntToStr(corn[10]);
Label30.Caption := IntToStr(corn[9]);
Label31.Caption := IntToStr(corn[8]);
Label32.Caption := IntToStr(corn[7]);
end;

procedure TForm5.Button3Click(Sender: TObject);
begin
    ComPort1.ShowSetupDialog;
end;

procedure TForm5.Button2Click(Sender: TObject);
var
    Str: String;
    i: integer;
begin
    if not ComPort1.Connected then
        ComPort1.Open;
    for i:= 1 to 16 do
    begin
        Str := send(show1[i]);
        ComPort1.WriteStr(Str);
        showmessage('pic'+IntToStr(i));
    end;
    for i:= 1 to 16 do
    begin
        Str := send1(corn[i]);
        ComPort1.WriteStr(Str);
        showmessage('corner of pic'+IntToStr(i));
    end;
    ComPort1.Close;
end;

end.

unit findmat;

interface

uses
    Windows, Messages, SysUtils, Classes, Graphics,
    Controls, Forms, Dialogs,
    StdCtrls;

type
    TForm2 = class(TForm)
        Memo1: TMemo;
        Memo2: TMemo;
        Open: TButton;
        OpenFileDialog1: TOpenDialog;
        name: TEdit;
        Memo3: TMemo;
        Label1: TLabel;
        Next: TButton;
        ShowMatrix: TLabel;
        ShowPicture: TLabel;
        procedure OpenClick(Sender: TObject);
        procedure NextClick(Sender: TObject);
    private
        { Private declarations }
    public
        { Public declarations }
    end;

const
    Maxlengthx = 192;
    Maxlengthy = 144;
    NumPic = 16;

type
    Tmatrix = 0..7;
    ArrayData = array [1..Maxlengthx, 1..Maxlengthy] of char;

var
    Form2: TForm2;
    da : ArrayData;
    Mat : Array [1..NumPic, 0..400] of Tmatrix;
    Long : Array [1..NumPic] of integer;
    Center : Array [1..Numpic, 1..2] of real;
    function findmatrix(da: ArrayData; var x,y: integer;
        var a: Tmatrix): Tmatrix;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

procedure Nextpos(a: Tmatrix; var x,y: integer);
procedure matrix( var k: integer; var Memo1: TMemo; var
Memo2: TMemo; var Memo3: TMemo;
var OpenFileDialog1: TOpenDialog; var fname: TEdit);

```

implementation

uses mee;

{SR *.DFM}

Procedure Nextpos(a: Tmatrix; var x,y: integer);

begin

case a of

0 : x := x-1;

1 : begin x := x-1; y := y-1; end;

2 : y := y-1;

3 : begin x := x+1; y := y-1; end;

4 : x := x+1;

5 : begin x := x+1; y := y+1; end;

6 : y := y+1;

7 : begin x := x-1; y := y+1; end;

end;

end;

Function findmatrix(da: ArrayData; var x,y: integer;

var a: Tmatrix): Tmatrix;

var

i,j : integer;

n : integer;

begin

{find org for check}

if a < 3 then a := a+5

else a := a-3;

{find next data}

i := x; j := y;

Nextpos(a,i,j);

n := 0;

{loop check next data}

while da[i,j] <> '#' do

begin

if a = 7 then a := 0

else a := a+1;

i := x; j := y;

Nextpos(a,i,j);

{check error}

n := n+1;

if n = 9 then ShowMessage('can not find next data');

end;

findmatrix := a;

x := i; y := j;

end;

Procedure matrix(var k: integer; var Memo1: TMemo; var

Memo2: TMemo; var Memo3: TMemo;

var OpenFileDialog1: TOpenDialog; var fname: TEdit);

var

f : File;

d : Array[1..(18432)66816]of char; { data }

buf : string[192];

b : integer; { bytread }

s : longint; { filesize for seek }

x : integer;

y : integer;

e : longint; { Exsample }

n,j : integer;

Xc : real;

orgx,orgy : integer;

a : Tmatrix; {matrix}

l,i : integer; {Long,matrix}

Yc : real;

begin

{ --- OPEN FILE ---}

with OpenFileDialog1 do

begin

//execute;

OpenDialog1.FileName := fname.text{'a:bw.bmp'};

end;(with)

AssignFile(f,OpenDialog1.FileName);

Reset(f,1);

s := filesize(f)-1;

Memo1.text := IntToStr(s) + #13#10;

s := s-15*576;{8640}

Memo1.text := Memo1.text + IntToStr(s) + #13#10;

x:=1;

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

y:=1;

{ ---Read File--- }

s := s{size-15line} - 66815{116*576};
seek(f,s);
BlockRead(f,d,{576*116}66816,b);
if b = 0 then
begin
  ShowMessage('error');
end {then} {check error}

{ ---Write File In Array--- }

else
begin
  j := 66815;
  n := j;
  x := 192;
  while j > 0 do
  begin
    if d[j] > 'Z' then d[j] := '0' {white}
    else d[j] := '#' {back};
    da[x,y] := d[j];
    x:=x-1;
    j:=j-3;
    if j = n-576 then
    begin
      x := 192;
      y := y + 1;
      n := n - 576;
    end;
  end;{while} {loop write data}
end;{else} {not error and write}

{ ---Show Picture--- }

y := 1;
while y <> 117 do
begin
  x := 1;
  buf := "";
  while x <> 193 do
begin
  buf := buf + da[x,y];
  x := x + 1;
end;
  Memo1.text := Memo1.text + buf + #13#10;
  y := y + 1;
end;
  CloseFile(f);

{ --- FIND MATRIX --- }

{begin}
  x := {6}17;
  y := 6;
  {find origin}
  while (da[x,y] <> '#') and (x <> Maxlengthx) do
  begin
    if y <> Maxlengthy-5-13 then y := y+1
    else begin y := 6; x := x+1; end;
    if x = Maxlengthx-5 then
    begin
      ShowMessage('can not find org');
      x := Maxlengthx;
    end;
  end;
  {keep origin}
  orgx := x;
  orgy := y;
  Memo2.text := IntToStr(k) + ' ' + fname.text + ' ' + '(' +
    +IntToStr(x) + ',' + IntToStr(y) + ')';
  {find next 2}
  a := 3;
  Mat[k,0] := findmatrix(da,x,y,a);
  i := 0;
  {find matrix}
  while (x <> orgx) or (y <> orgy) do
  begin
    l := 400;
    if i = l then ShowMessage('length error');
    i := i+1;
    Mat[k,i] := findmatrix(da,x,y,a);
    Long[k] := i;
  end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

{show matrix}
    Memo2.Text := Memo2.Text + ' Length of Matrix='
        + IntToStr(i) + #13#10 + 'Matrix is'
        + #13#10 + '[';
    l := i;
    for i := 1 to l do
    begin
        Memo2.Text := Memo2.Text + ',' + IntToStr(Mat[k,i]);
        if (i mod 35 = 0) then
            Memo2.Text := Memo2.Text + #13#10;
    end;
    Memo2.Text := Memo2.Text + ']' + #13#10;
    Memo3.Text := Memo3.Text + Memo2.Text;
    k := k + 1; { lev of pic }

{ ---Find Center--- }
    l := 0;{m00}
    e := 0;{m10 find Xc}
    s := 0;{m01 find Yc}
    for y := 1 to 116 do
    for x := 1 to 192 do
    if da[x,y] = '#' then
    begin
        l := l + 1;
        e := e + x;
        s := s + y;
    end;
    Xc := ((e*100)div l)/100 ;{Xc}
    Yc := ((s*100)div l)/100;{Yc}
    {ShowMessage('m00 = ' + IntToStr(l)+#13#10+
        'm10 = ' + IntToStr(e)+#13#10+
        'm01 = ' + IntToStr(s)+#13#10+}
    Memo3.Text := Memo3.Text + 'Center is [ ' +
        FloatToStr(Xc) + ' , ' +
        FloatToStr(Yc) + ' ]' + #13#10 + #13#10;
{ keep center }
    Center[k-1,1] := Xc;
    Center[k-1,2] := Yc;
end;

procedure TForm2.OpenClick(Sender: TObject);
var k : integer; { lev of pic }
begin
    Memo3.Text := '';
    k := 1;
    {1}
        name.Text := 'c:\Ks001.bmp';
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {2}
        name.Text := 'c:\Ks002.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {3}
        name.Text := 'c:\Ks003.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {4}
        name.Text := 'c:\Ks004.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {5}
        name.Text := 'c:\Ks005.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {6}
        name.Text := 'c:\Ks006.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {7}
        name.Text := 'c:\Ks007.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {8}
        name.Text := 'c:\Ks008.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {9}
        name.Text := 'c:\Ks009.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {10}
        name.Text := 'c:\Ks010.bmp';
        ShowMessage( 'Finish Picture'+IntToStr(k-1));
        matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
    {11}
        name.Text := 'c:\Ks011.bmp';

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{12}
name.Text := 'c:\Ks012.bmp';
ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{13}
name.Text := 'c:\Ks013.bmp';
ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{14}
name.Text := 'c:\Ks014.bmp';
ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{15}
name.Text := 'c:\Ks015.bmp';
ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{16}
name.Text := 'c:\Ks016.bmp';
ShowMessage( 'Finish Picture'+IntToStr(k-1));
matrix( k,Memo1,Memo2,Memo3,OpenDialog1,name);
{end}
name.text := 'None';
ShowMessage( 'Number of Picture = '+IntToStr(k-1)
              +'#10#13 +'Finish All Picture');
end;

procedure TForm2.NextClick(Sender: TObject);
begin
    Form3.ShowModal
end;

end.

unit mee;

interface

uses
    Windows, Messages, findmat, SysUtils, Classes,
    Graphics, Controls, Forms, Dialogs,
    StdCtrls;

type
    TForm3 = class(TForm)
        Button1: TButton;
        Memo1: TMemo;
        Button2: TButton;
        procedure Button1Click(Sender: TObject);
        procedure Button2Click(Sender: TObject);
        // procedure Button2Click(Sender: TObject);
        {procedure Button2Click(Sender: TObject);}
    private
        { Private declarations }
    public
        { Public declarations }
    end;

type da = record
    p : array [1..90] of integer;
    q : array [1..90] of integer;
    r : array [1..90] of integer;
    s : array [1..90] of integer;
    max : array [1..4] of integer;
    z : array [1..3] of integer;
end;

var
    Form3: TForm3;
    pic :array [1..16] of da;
    l,ll,nn,mm,numpic,numside,ss : integer;

implementation

uses pat;

{$R *.DFM}

procedure TForm3.Button1Click(Sender: TObject);
var g : array [1..16] of da;
    x : array [1..350] of integer;
    n,m,o,p,l : integer;
    y : array [1..16,1..4] of integer;
    z : array [1..16,1..4] of integer;
    a : array [1..16,1..4] of integer;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

b : array [1..16,1..4] of integer;
ya,za,aa,t,ba,j,k,i,d,ii,jj : integer;

begin
  ll:=13; mm:=5; nn:=1;numpic:=1;numside:=60;
  for l:=1 to 16 do
    begin
      m:=0;n:=0;o:=0;p:=0;
      for i:=1 to long[l] do
        begin
          x[i]:=mat[l,i];
          end;
          for i:=1 to long[l] do
            begin
              if (i=1) and (x[i]=4) and (x[i+1]=4) then m:=i else
                if ((x[i]=2) or (x[i]=3)) and (x[i+1]=4) then
                  begin
                    ii:=0;
                    for jj:=i+1 to i+11 do
                      begin
                        if x[jj]=4 then
                          ii:=ii+1;
                        end;
                        if ((x[i-1]=2) or (x[i-1]=3) or (x[i-1]=1) or (x[i-1]=0))
                          and ((x[i-2]=2) or (x[i-2]=3) or (x[i-2]=0))
                          and (ii>3) and (i<50) and (m<>1) and (i<28)
                        then m:=i;
                      end;
                    end;
                    if ((x[i]=0) or (x[i]=1)) and (x[i+1]=2) and (p=0)
                      then
                        begin
                          ii:=0;
                          for jj:=i+1 to i+11 do
                            begin
                              if x[jj]=2 then
                                ii:=ii+1;
                              end;
                              if ((x[i-1]=0) or (x[i-1]=1)) and ((x[i-2]=0) or (x[i-2]=1))
                                and (ii > 5) and (i>(m+(long[l]/4)-10))
                                and (i<(m+(long[l]/4)+10))) then n:=i;
                            end;
                          end;
                          if ((x[i]=6) or (x[i]=7)) and (x[i+1]=0) and (o=0)
                            then
                              begin
                                ii:=0;
                                for jj:=i+1 to i+11 do
                                  begin
                                    if x[jj]=0 then
                                      ii:=ii+1;
                                    end;
                                    if ((x[i-1]=6) or (x[i-1]=7) or (x[i-1]=5)) and ((x[i-2]=6) or (x[i-2]=7) or (x[i-2]=5))
                                      and (ii > 2) and (i>(n+(long[l]/4)-10))
                                      and (i<(n+(long[l]/4)+10))
                                    then o:=i;
                                  end;
                                end;
                              end;
                            end;
                          end;
                        end;
                      end;
                    end;
                  end;
                end;
              end;
            end;
          end;
        end;
      end;
    end;
  end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

        p:=i;
    end;
end;

if l=11 then
    begin
        m:=26;
        n:=70;
        o:=115;
        p:=167;

    end;

i:=1;
memo1.text:=memo1.text+#13+#10+intToStr
(l)+'p:=';
ya:=0;za:=0;aa:=0;ba:=0;
for k:=m to n do
    begin
        if k=1 then
            begin
                g[l].p[1]:=mat[l, long[l]];
                memo1.text:=memo1.text+intToStr(g[l].p[i]);
                i:=i+1;
            end;
        g[l].p[i]:=x[k];
        memo1.text:=memo1.text+intToStr(g[l].p[i]);
        if g[l].p[i]=0 then ya:=ya+1;
        if g[l].p[i]=2 then za:=za+1;
        if g[l].p[i]=4 then aa:=aa+1;
        if g[l].p[i]=6 then ba:=ba+1;
        i:=i+1;
    end;
    g[l].max[1]:=i-1;
    memo1.text:= memo1.text+' max'+intToStr
(l)+'p='+intToStr(g[l].max[1]);
    t:=g[l].max[1];
    memo1.text:=memo1.text+#13+#10+intToStr
(l)+'q:=';
    i:=1;
    y[l,1]:=ya; z[l,1]:=za; a[l,1]:=aa; b[l,1]:=ba;
    ya:=0; za:=0; aa:=0; ba:=0;

```

```

for k := n to o do
    begin
        g[l].q[i]:=x[k];
        memo1.text:=memo1.text+intToStr(g[l].q[i]);
        if g[l].q[i]=0 then ya:=ya+1;
        if g[l].q[i]=2 then za:=za+1;
        if g[l].q[i]=4 then aa:=aa+1;
        if g[l].q[i]=6 then ba:=ba+1;
        i:=i+1;
    end;
    g[l].max[2]:=i-1;
    memo1.text:= memo1.text+' max'+intToStr
(l)+'q='+intToStr(g[l].max[2]);
    y[l,2]:=ya; z[l,2]:=za; a[l,2]:=aa; b[l,2]:=ba;
    ya:=0; za:=0; aa:=0; ba:=0;
    memo1.text:=memo1.text+#13+#10+intToStr
(l)+'r:=';
    i:=1;
    for k:=o to p do
        begin
            g[l].r[i]:=x[k];
            memo1.text:=memo1.text+intToStr(g[l].r[i]);
            if g[l].r[i]=0 then ya:=ya+1;
            if g[l].r[i]=2 then za:=za+1;
            if g[l].r[i]=4 then aa:=aa+1;
            if g[l].r[i]=6 then ba:=ba+1;
            i:=i+1;
        end;
        g[l].max[3]:=i-1;
        memo1.text:= memo1.text+' max'+intToStr
(l)+'r='+intToStr(g[l].max[3]);
        y[l,3]:=ya; z[l,3]:=za; a[l,3]:=aa; b[l,3]:=ba;
        ya:=0; za:=0; aa:=0; ba:=0;
        memo1.text:=memo1.text+#13+#10+intToStr
(l)+'s:=';
        i:=1;
        for k:=p to long[l] do
            begin
                g[l].s[i]:=x[k];
                memo1.text:=memo1.text+intToStr(g[l].s[i]);
                if g[l].s[i]=0 then ya:=ya+1;
                if g[l].s[i]=2 then za:=za+1;
                if g[l].s[i]=4 then aa:=aa+1;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

if g[l].s[i]=6 then ba:=ba+1;
i:=i+1;
end;
if m<>1 then
begin
for k:=1 to m do
begin
g[l].s[i]:=x[k];
memo1.text:=memo1.text+intTostr(g[l].s[i]);
if g[l].s[i]=0 then ya:=ya+1;
if g[l].s[i]=2 then za:=za+1;
if g[l].s[i]=4 then aa:=aa+1;
if g[l].s[i]=6 then ba:=ba+1;
i:=i+1;
end;
end;
g[l].max[4]:=i-1;
memo1.text:= memo1.text+' max'+intTostr
(l)+'s'+intTostr(g[l].max[4]);
g[l].max[1]:=t;
y[l,4]:=ya; z[l,4]:=za; a[l,4]:=aa; b[l,4]:=ba;
d:=0;
if ((b[l,1]/(n-m+1))>0.72) or ((y[l,1]/(n-m+1))>0.72)
or ((z[l,1]/(n-m+1))>0.72) or ((a[l,1]/(n-m+1))>0.72)
then d:=d+1;
if ((b[l,2]/(o-n+1))>0.72) or ((y[l,2]/(o-n+1))>0.72)
or ((z[l,2]/(o-n+1))>0.72) or ((a[l,2]/(o-n+1))>0.72)
then d:=d+1;
if ((b[l,3]/(p-o+1))>0.72) or ((y[l,3]/(p-o+1))>0.72)
or ((z[l,3]/(p-o+1))>0.72) or ((a[l,3]/(p-o+1))>0.72)
then d:=d+1;
if ((b[l,4]/(n-m+1))>0.72) or ((y[l,4]/(n-m+1))>0.72)
or ((z[l,4]/(n-m+1))>0.72) or ((a[l,4]/(n-m+1))>0.72)
then d:=d+1;
if d=0 then
begin
memo1.text:=memo1.text+#13+#10+intTostr(l)+'=
center';
for i:=1 to g[l].max[1] do
pic[l].p[i]:=g[l].p[i];
for i:=1 to g[l].max[2] do
begin
pic[l].q[i]:=g[l].q[i];
pic[l].r[i]:=g[l].r[i];
end;
for i:=1 to g[l].max[3] do
begin
pic[l].z[i]:=g[l].z[i];
end;
for i:=1 to g[l].max[4] do
begin
pic[l].s[i]:=g[l].s[i];
end;
pic[l].max[1]:=g[l].max[1];
pic[l].max[2]:=g[l].max[2];
pic[l].max[3]:=g[l].max[3];
pic[l].max[4]:=g[l].max[4];
pic[l].z[1]:=i;
end;
for i:=1 to g[l].max[3] do
begin
pic[l].z[1]:=i;
end;
memo1.text:=memo1.text+#13+#10+intTostr(l)+'=
side';
for i:=1 to g[l].max[1] do
begin
pic[mm].p[i]:=g[l].p[i];
end;
for i:=1 to g[l].max[2] do
begin
pic[mm].q[i]:=g[l].q[i];
end;
for i:=1 to g[l].max[3] do
begin
pic[mm].r[i]:=g[l].r[i];
end;
for i:=1 to g[l].max[4] do
begin
pic[mm].s[i]:=g[l].s[i];
end;
pic[mm].max[1]:=g[l].max[1];
pic[mm].max[2]:=g[l].max[2];
pic[mm].max[3]:=g[l].max[3];
pic[mm].max[4]:=g[l].max[4];
pic[mm].z[1]:=i;
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

memo1.text:=memo1.text+'z:='+inttostr(mm);
pic[mm].z[2]:=0;
mm:=mm+1;
end;
if d=2 then
begin
memo1.text:=memo1.text+#13+#10+intTostr
(l)+'= corner';
for i:=1 to g[l].max[1] do
begin
pic[nn].p[i]:=g[l].p[i];
end;
for i:=1 to g[l].max[2] do
begin
pic[nn].q[i]:=g[l].q[i];
end;
for i:=1 to g[l].max[3] do
begin
pic[nn].r[i]:=g[l].r[i];
end;
for i:=1 to g[l].max[4] do
begin
pic[nn].s[i]:=g[l].s[i];
end;
pic[nn].max[1]:=g[l].max[1];
pic[nn].max[2]:=g[l].max[2];
pic[nn].max[3]:=g[l].max[3];
pic[nn].max[4]:=g[l].max[4];
pic[nn].z[1]:=l;
memo1.text:=memo1.text+'z:='+inttostr(nn);
pic[nn].z[2]:=0;
nn:=nn+1;

end;

end;

end;

```

```

procedure TForm3.Button2Click(Sender: TObject);
begin
Form4.ShowModal
end;

end.

unit pat;

interface

uses
Windows, Messages, mee, Controls, Classes, SysUtils,
Graphics, Forms, Dialogs,
ExtCtrls, StdCtrls;

type
TForm4 = class(TForm)
Label1: TLabel;
Label2: TLabel;
Button1: TButton;
Button2: TButton;
Button3: TButton;
Button4: TButton;
Label3: TLabel;
Label4: TLabel;

procedure Button1Click(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button4Click(Sender: TObject);

private
{ Private declarations }

public
{ Public declarations }

end;

type
fg=record

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

r1:array[1..60] of integer;
max:integer ;
end;
{da = record
    p : array [1..60] of integer;
    q : array [1..90] of integer;
    r : array [1..90] of integer;
    s : array [1..90] of integer;
    max : array [1..4] of integer;
    z : array [1..3] of integer;
end;
}
var
Form4: TForm4;
z:integer;
pic1:array [1..16] of da;
gun:array[1..8] of fg;
show1,show3:array[1..16] of integer;
corner:array[1..16] of integer;
complete:array[1..4] of integer;
com:array[1..4] of integer;
aa,bb,cc,dd:boolean;
display1,display2:string;
function chkcorner(input1:array of
integer;max:integer):boolean;
function rotcorner(input2:integer;m:integer):integer;
function compare(qside:array of
integer;max1:integer;sside:array of
integer;max2:integer):boolean;
function forgun(ip:integer;m:integer):integer;
function rtop(a:array of
integer;max:integer;n:integer):integer;
function switchqtop (a:array of integer;n:integer): integer ;
function switchrtop (a:array of integer;n:integer): integer ;
function switchstop (a:array of integer;n:integer): integer ;
implementation

uses display;

{$R *.DFM}

```

```

procedure TForm4.Button1Click(Sender: TObject);

```

```

var n,rim,rim2:integer; display:string;

```

```

begin

```

```

for n:=1 to 16 do

```

```

pic1[n]:=pic[n];

```

```

for n:=2 to 47 do

```

```

pic1[5].q[n]:=6;

```

```

for n:=2 to 45 do

```

```

pic1[2].s[n]:=2;

```

```

for n:=1 to 16 do

```

```

begin

```

```

pic[n].z[3]:=n;

```

```

pic1[n].z[3]:=n;

```

```

end;

```

```

pic1[10]:=pic[11];

```

```

pic1[11]:=pic[10];

```

```

pic1[10].z[3]:=11;

```

```

pic1[11].z[3]:=10;

```

```

aa:=chkcorner(pic1[1].p,pic1[1].max[1]);

```

```

bb:=chkcorner(pic1[1].q,pic1[1].max[2]);

```

```

cc:=chkcorner(pic1[1].r,pic1[1].max[3]);

```

```

dd:=chkcorner(pic1[1].s,pic1[1].max[4]);

```

```

//=====
=====

```

```

if aa and bb then

```

```

begin

```

```

rim:=pic[1].r[1];

```

```

rim2:=pic[1].s[1];

```

```

display:="";

```

```

for n:=2 to pic1[1].max[3] do

```

```

pic1[1].q[n]:=rotcorner(pic[1].r[n],rim);

```

```

for n:=2 to pic1[1].max[4] do

```

```

pic1[1].s[n]:=rotcorner(pic[1].s[n],rim2);

```

```

pic1[1].q[1]:=4;

```

```

pic1[1].s[1]:=0;

```

```

showmessage('เริ่มต่อได้ :');

```

```

pic1[1].z[2]:=270;

```

```

end

```

```

else

```

```

if bb and cc then

```

```

begin

```

```

rim:=pic[1].s[1];

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

rim2:=pic[1].p[1];
display:="";
for n:=2 to pic1[1].max[4] do
pic1[1].q[n]:=rotcorner(pic[1].s[n],rim);
for n:=2 to pic1[1].max[1] do
pic1[1].s[n]:=rotcorner(pic[1].p[n],rim2);
pic1[1].q[1]:=4;
pic1[1].s[1]:=0;
showmessage(' ');
pic1[1].z[2]:=180;
end
else

if cc and dd then
begin
rim:=pic[1].p[1];
rim2:=pic[1].q[1];
display:="";
for n:=2 to pic1[1].max[1] do
pic1[1].q[n]:=rotcorner(pic[1].p[n],rim);
for n:=2 to pic1[1].max[2] do
pic1[1].s[n]:=rotcorner(pic[1].q[n],rim2);
pic1[1].q[1]:=4;
pic1[1].s[1]:=0;
showmessage('เริ่มต่อได้ ');
pic1[1].z[2]:=90;
end
else

if aa and dd then
begin
rim:=pic[1].r[1];
display:="";
for n:=2 to pic1[1].max[3] do
begin
pic1[1].s[n]:=rotcorner(pic[1].r[n],rim);
pic1[1].q[1]:=4;
pic1[1].s[1]:=0;
end;
showmessage('เริ่มต่อได้ ');
pic1[1].z[2]:=0;
end;
end;

////////////////////////////////////

function chkcorner(input1:array of
integer;max:integer):boolean;
var u:integer; index1:integer;
begin
u:=1;
index1:=0;
repeat
if input1[u]=input1[u+1] then
begin
index1:=index1+1;
u:=u+1;
end
else
u:=u+1;
until (u=max+1);
if index1>=0.8*max then result:=true
else result:=false;
end;
////////////////////////////////////

function rotcorner(input2:integer; m:integer):integer;
begin
if (m=6)or(m=7) then
begin
if input2=0 then
input2:=6
else
if input2=1 then
input2:=7
else
if input2=7 then
input2:=5
else
if input2=6 then
input2:=4
else
if input2=2 then
input2:=0
else
if input2=5 then

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไมอนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ตัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

input2:=3
else
if input2=3 then
input2:=1;
result:=input2;
end

else

if (m=0) or (m=1) then
begin
if input2=2 then
input2:=6
else
if input2=1 then
input2:=5
else
if input2=3 then
input2:=7
else
if input2=0 then
input2:=4
else
if input2=4 then
input2:=0
else
if input2=7 then
input2:=3
else
if input2=5 then
input2:=1;
result:=input2;
end
else

if (m=2) or (m=3) then
begin
if input2=4 then
input2:=6
else
if input2=5 then
input2:=7
else

```

```

if input2=3 then
input2:=5
else
if input2=2 then
input2:=4
else
if input2=6 then
input2:=0
else
if input2=1 then
input2:=3
else
if input2=7 then
input2:=1;
result:=input2;
end

else
if (m=4) or (m=5) then
result:=input2
else
result:=input2;
end;

////////////////////
function forgun(ip:integer;m:integer):integer;
begin
if (m=0) or (m=1) then
begin
if ip=2 then
ip:=0
else
if ip=3 then
ip:=1
else
if ip=0 then
ip:=6
else
if ip=7 then
ip:=5
else
if ip=5 then

```



```

ip:=3
else
if ip=4 then
ip:=2
else
if ip=1 then
ip:=7;
result:=ip;
end
else

```

```

if (m=2) or (m=3) then

```

```

begin
if ip=2 then
ip:=6
else
if ip=3 then
ip:=7
else
if ip=4 then
ip:=0
else
if ip=7 then
ip:=3
else
if ip=5 then
ip:=1
else
if ip=6 then
ip:=2
else
if ip=1 then
ip:=5;
result:=ip;
end
else

```

```

if (m=4) or (m=5) then
begin
if ip=6 then
ip:=0
else

```

```

if (m=4) or (m=5) then

```

```

begin
if ip=6 then
ip:=0
else

```

```

if ip=3 then

```

```

ip:=5
else
if ip=0 then
ip:=2
else
if ip=7 then
ip:=1
else
if ip=5 then
ip:=7
else

```

```

if ip=4 then

```

```

ip:=6
else
if ip=1 then
ip:=3;
result:=ip;
end

```

```

else
if (m=6) or (m=7) then
result:=ip
else
result:=ip;
end;

```

```

//=====

```

```

=====>

```

```

function compare(qside:array of
integer;max1:integer;sside:array of
integer;max2:integer):boolean;
var m,ms,ax1,ax2:integer; index2,index1,index3:integer;
begin
m:=1;
index2:=0;
index1:=0;
ax1:=max1;
ax2:=max2;
if max1>max2 then ms:=max2
else ms:=max1;

```

```

repeat

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```
if qside[m]=sside[m] then
```

```
begin
```

```
index2:=index2+1;
```

```
m:=m+1;
```

```
end
```

```
else m:=m+1;
```

```
until (m=ms) ;
```

```
repeat
```

```
if qside[ax1]=sside[ax2] then
```

```
begin
```

```
index1:=index1+1;
```

```
ax1:=ax1-1;
```

```
ax2:=ax2-1;
```

```
end
```

```
else
```

```
begin
```

```
ax1:=ax1-1;
```

```
ax2:=ax2-1;
```

```
end
```

```
until (ax1=0) or (ax2=0) ;
```

```
if index2>index1 then index3:=index2
```

```
else index3:=index1;
```

```
index3:=index3+ abs(max1-max2);
```

```
if index3>=(0.6*ms) then result:=true
```

```
else result:=false;
```

```
end;
```

```
//=====
```

```
=====>
```

```
procedure TForm4.Button4Click(Sender: TObject);
```

```
begin
```

```
Form5.ShowModal
```

```
end;
```

```
procedure TForm4.Button2Click(Sender: TObject);
```

```
var counter,n,j,m,g,k,x,w,e,h,l:integer; a,b,c,d,ans:boolean;
```

```
temp,temp2,temp3,temp4,temp5:array[1..90]of integer;
```

```
a2,b2,c2,d2:boolean; rim,rim2,tm:integer;
```

```
display,display1:string;
```

```
map:integer; id:array[1..4] of integer;
```

```
temp6:array[1..3] of integer;
```

```
temp7:array[1..4] of integer;
```

```
begin
```

```
if aa and bb then l:=3
```

```
else
```

```
if bb and cc then l:=4
```

```
else
```

```
if cc and dd then l:=1
```

```
else
```

```
if aa and dd then l:=2;
```

```
id[1]:=1;
```

```
for n:=2 to 4 do id[n]:=0;
```

```
show1[1]:=1;
```

```
corner[1]:=pic1[1].z[2];
```

```
counter:=0;
```

```
map:=0;
```

```
j:=5;
```

```
x:=1;
```

```
repeat
```

```
a:=chkcorner(pic1[j].p,pic1[j].max[1]);
```

```
b:=chkcorner(pic1[j].q,pic1[j].max[2]);
```

```
c:=chkcorner(pic1[j].r,pic1[j].max[3]);
```

```
d:=chkcorner(pic1[j].s,pic1[j].max[4]);
```

```
if a then
```

```
begin
```

```
for n:=2 to pic1[j].max[4] do
```

```
temp[n]:=pic1[j].s[n];
```

```
for n:=2 to pic1[j].max[4] do
```

```
pic1[j].s[n]:=temp[pic1[j].max[4]+2-n];
```

```
//--
```

```
for n:=2 to pic1[j].max[4] do
```

```
pic1[j].s[n]:=rotcorner(pic1[j].s[n],pic1[j].s[1]);
```

```
pic1[j].s[1]:=4;
```

```
//--
```

```
ans:=compare(pic1[x].q,pic1[x].max[l],pic1[j].s,pic1[j].max[4]); //=====> compare
```

```
if ans then
```

```
begin
```

```
showmessage('Next');
```

```
counter:=counter+1;
```

```
l:=2;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

show1[counter+1]:=pic1[j].z[1];
if (counter=1) or (counter=2) then pic1[j].z[2]:=0;
if (counter=4) or (counter=5) then pic1[j].z[2]:=90;
if (counter=7) or (counter=8) then pic1[j].z[2]:=180;
if (counter=10) or (counter=11) then pic1[j].z
[2]:=270;
corner[counter+1]:=pic1[j].z[2];
end
else
begin
pic1[j]:=pic[pic1[j].z[3]]; //-----not sure
j:=j+1;
end
end
else
if b then
begin
for n:=2 to pic1[j].max[1] do
temp[n]:=pic1[j].p[n];
for n:=2 to pic1[j].max[1] do {arrange value}
pic1[j].p[n]:=temp[pic1[j].max[1]+2-n];
//---
for n:=2 to pic1[j].max[1] do
pic1[j].p[n]:=rotcorner(pic1[j].p[n],pic1[j].p[1]); {set
value to the form for match}
pic1[j].p[1]:=4;
//---
ans:=compare(pic1[x].q,pic1[x].max[1],pic1[j].p,pic1
[j].max[1]);
if ans then
begin
showmessage('Next');
for n:=2 to pic1[j].max[3] do
pic1[j].q[n]:=rotcorner(pic1[j].r[n],pic1[j].r[1]);
//next q side for map
for n:=2 to pic1[j].max[4] do
pic1[j].r[n]:=forgun(pic1[j].s[n],pic1[j].s[1]);
pic1[j].q[1]:=4;
pic1[j].r[1]:=6;
counter:=counter+1;
l:=3;
show1[counter+1]:=pic1[j].z[1];

```

```

if (counter=1) or (counter=2) then pic1[j].z[2]:=270;
if (counter=4) or (counter=5) then pic1[j].z[2]:=0;
if (counter=7) or (counter=8) then pic1[j].z[2]:=90;
if (counter=10) or (counter=11) then pic1[j].z
[2]:=180;
corner[counter+1]:=pic1[j].z[2];
end
else
begin
pic1[j]:=pic[pic1[j].z[3]]; //----- not sure
j:=j+1;
end
end
else
if c then
begin
for n:=2 to pic1[j].max[2] do
temp[n]:=pic1[j].q[n];
for n:=2 to pic1[j].max[2] do
pic1[j].q[n]:=temp[pic1[j].max[2]+2-n];
pic1[j].q[1]:=4;
ans:=compare(pic1[x].q,pic1[x].max[1],pic1[j].q,pic1
[j].max[2]);
if ans then
begin
showmessage('Next');
for n:=2 to pic1[j].max[4] do
pic1[j].q[n]:=rotcorner(pic1[j].s[n],pic1[j].s[1]);
for n:=2 to pic1[j].max[1] do
pic1[j].r[n]:=forgun(pic1[j].p[n],pic1[j].p[1]);
//=====> obposit edge
pic1[j].r[1]:=6;
counter:=counter+1;
l:=4;
show1[counter+1]:=pic1[j].z[1];
if (counter=1) or (counter=2) then pic1[j].z[2]:=180;
if (counter=4) or (counter=5) then pic1[j].z[2]:=270;
if (counter=7) or (counter=8) then pic1[j].z[2]:=0;
if (counter=10) or (counter=11) then pic1[j].z[2]:=90;
corner[counter+1]:=pic1[j].z[2];
end
else

```



```

begin
pic1[j]:=pic[pic1[j].z[3]]; //-----not sure
j:=j+1;
end
end
else
if d then
begin
for n:=1 to pic1[j].max[3] do
temp[n]:=pic1[j].r[n];
for n:=1 to pic1[j].max[3] do
pic1[j].r[n]:=temp[pic1[j].max[3]+2-n];
pic1[j].r[1]:=6;
for n:=2 to pic1[j].max[3] do
pic1[j].r[n]:=rotcorner(pic1[j].r[n],pic1[j].r[1]);
pic1[j].r[1]:=4;
ans:=compare(pic1[x].q,pic1[x].max[l],pic1[j].r,pic1
[j].max[3]);
if ans then
begin
showmessage('Next');
for n:=2 to pic1[j].max[2] do
pic1[j].r[n]:=forgun(pic1[j].q[n],pic1[j].q[1]);
for n:=2 to pic1[j].max[1] do
pic1[j].q[n]:=rotcorner(pic1[j].p[n],pic1[j].p[1]);
pic1[j].r[1]:=6;
counter:=counter+1;
l:=1;
show1[counter+1]:=pic1[j].z[1];
if (counter=1) or (counter=2) then pic1[j].z[2]:=90;
if (counter=4) or (counter=5) then pic1[j].z[2]:=180;
if (counter=7) or (counter=8) then pic1[j].z[2]:=270;
if (counter=10) or (counter=11) then pic1[j].z[2]:=0;
corner[counter+1]:=pic1[j].z[2];
end
else
begin
pic1[j]:=pic[pic1[j].z[3]]; //----- not sure *****
j:=j+1;
end
end;

```

if ans then

```

begin
if b then
begin
for m:=1 to pic1[j].max[1] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].p[m];
for m:=1 to pic1[12-map].max[1] do
pic1[j].p[m]:=pic1[12-map].p[m];
for m:=1 to pic1[j].max[1] do
pic1[12-map].p[m]:=temp2[m];
for m:=1 to pic1[j].max[2] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].q[m];
for m:=1 to pic1[12-map].max[2] do
pic1[j].q[m]:=pic1[12-map].q[m];
for m:=1 to pic1[j].max[2] do
pic1[12-map].q[m]:=temp2[m];
for m:=1 to pic1[j].max[3] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].r[m];
for m:=1 to pic1[12-map].max[3] do
pic1[j].r[m]:=pic1[12-map].r[m];
for m:=1 to pic1[j].max[3] do
pic1[12-map].r[m]:=temp2[m];
for m:=1 to pic1[j].max[4] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].s[m];
for m:=1 to pic1[12-map].max[4] do
pic1[j].s[m]:=pic1[12-map].s[m];
for m:=1 to pic1[j].max[4] do
pic1[12-map].s[m]:=temp2[m];
end
else
if a then
begin
for m:=1 to pic1[j].max[1] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].p[m];
for m:=1 to pic1[12-map].max[1] do
pic1[j].p[m]:=pic1[12-map].p[m];

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```
for m:=1 to pic1[j].max[1] do
pic1[12-map].p[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[2] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].q[m];
for m:=1 to pic1[12-map].max[2] do
pic1[j].q[m]:=pic1[12-map].q[m];
for m:=1 to pic1[j].max[2] do
pic1[12-map].q[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[3] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].r[m];
for m:=1 to pic1[12-map].max[3] do
pic1[j].r[m]:=pic1[12-map].r[m];
for m:=1 to pic1[j].max[3] do
pic1[12-map].r[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[4] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].s[m];
for m:=1 to pic1[12-map].max[4] do
pic1[j].s[m]:=pic1[12-map].s[m];
for m:=1 to pic1[j].max[4] do
pic1[12-map].s[m]:=temp2[m];
end
else
if c then
begin
```

```
for m:=1 to pic1[j].max[1] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].p[m];
for m:=1 to pic1[12-map].max[1] do
pic1[j].p[m]:=pic1[12-map].p[m];
for m:=1 to pic1[j].max[1] do
pic1[12-map].p[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[2] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].q[m];
for m:=1 to pic1[12-map].max[2] do
pic1[j].q[m]:=pic1[12-map].q[m];
```

```
for m:=1 to pic1[j].max[2] do
pic1[12-map].q[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[3] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].r[m];
for m:=1 to pic1[12-map].max[3] do
pic1[j].r[m]:=pic1[12-map].r[m];
for m:=1 to pic1[j].max[3] do
pic1[12-map].r[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[4] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].s[m];
for m:=1 to pic1[12-map].max[4] do
pic1[j].s[m]:=pic1[12-map].s[m];
for m:=1 to pic1[j].max[4] do
pic1[12-map].s[m]:=temp2[m];
end
```

```
else
if d then
begin
for m:=1 to pic1[j].max[1] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].p[m];
for m:=1 to pic1[12-map].max[1] do
pic1[j].p[m]:=pic1[12-map].p[m];
for m:=1 to pic1[j].max[1] do
pic1[12-map].p[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[2] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].q[m];
for m:=1 to pic1[12-map].max[2] do
pic1[j].q[m]:=pic1[12-map].q[m];
for m:=1 to pic1[j].max[2] do
pic1[12-map].q[m]:=temp2[m];
```

```
for m:=1 to pic1[j].max[3] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].r[m];
for m:=1 to pic1[12-map].max[3] do
pic1[j].r[m]:=pic1[12-map].r[m];
```



```

for m:=1 to pic1[j].max[3] do
pic1[12-map].r[m]:=temp2[m];

```

```

for m:=1 to pic1[j].max[4] do //if it match then change it
to the pic[12-map]
temp2[m]:=pic1[j].s[m];
for m:=1 to pic1[12-map].max[4] do
pic1[j].s[m]:=pic1[12-map].s[m];
for m:=1 to pic1[j].max[4] do
pic1[12-map].s[m]:=temp2[m];
end;

```

```

for m:=1 to 3 do
temp6[m]:=pic1[j].z[m];
for m:=1 to 3 do
pic1[j].z[m]:=pic1[12-map].z[m];
for m:=1 to 3 do
pic1[12-map].z[m]:=temp6[m];

```

```

for m:=1 to 4 do
temp7[m]:=pic1[j].max[m];
for m:=1 to 4 do
pic1[j].max[m]:=pic1[12-map].max[m];
for m:=1 to 4 do
pic1[12-map].max[m]:=temp7[m];

```

```

map:=map+1;
x:=13-map;
j:=5;
end;

```

```

if (counter=2) or (counter=5) or (counter=8) then
//corner
begin
k:=2;
w:=x;
repeat
repeat
if id[k] <> 0 then k:=k+1
until (k=4) or (id[k]=0);

```

```

a2:=chkcorner(pic1[k].p,pic1[k].max[1]);

```

```

b2:=chkcorner(pic1[k].q,pic1[k].max[2]);

```

```

c2:=chkcorner(pic1[k].r,pic1[k].max[3]);
d2:=chkcorner(pic1[k].s,pic1[k].max[4]);

```

```

if a2 and b2 then
begin
rim:=pic1[k].r[1];
rim2:=pic1[k].s[1];
display:="";
for g:=2 to pic1[k].max[4] do
pic1[k].s[g]:=rotcorner(pic1[k].s[g],rim2); //---> map
for g:=2 to pic1[k].max[4] do //=====> reverse
value

```

```

temp[g]:=pic1[k].s[g];
for g:=2 to pic1[k].max[4] do
pic1[k].s[g]:=temp[pic1[k].max[4]+2-g];
pic1[k].s[1]:=4;
ans:=compare(pic1[w].q,pic1[w].max[1],pic1[k].s,pic1
[k].max[4]);

```

```

if ans then
begin
for g:=2 to pic1[k].max[3] do
pic1[k].q[g]:=rotcorner(pic1[k].r[g],rim); //---> return
value for map

```

```

pic1[k].q[1]:=4; // with next pic
if counter=2 then pic1[k].z[2]:=0;
if counter=5 then pic1[k].z[2]:=90;
if counter=8 then pic1[k].z[2]:=180;
counter:=counter+1;
x:=k;
show1[counter+1]:=pic1[k].z[1];
corner[counter+1]:=pic1[k].z[2];

```

```

end

```

```

else

```

```

begin

```

```

pic1[k]:=pic[k];

```

```

k:=k+1;

```

```

end

```

```

end

```

```

else

```

```

if b2 and c2 then

```

```

begin

```

```

rim:=pic1[k].s[1];

```



```

rim2:=pic1[k].p[1];
display:="";
for g:=2 to pic1[k].max[1] do
pic1[k].s[g]:=rotcorner(pic1[k].p[g],rim2);
for g:=2 to pic1[k].max[1] do //=====>
reverse value
temp[g]:=pic1[k].s[g];
for g:=2 to pic1[k].max[1] do
pic1[k].s[g]:=temp[pic1[k].max[1]+2-g];
pic1[k].s[1]:=4;
ans:=compare(pic1[w].q,pic1[w].max[l],pic1[k].s,pic1
[k].max[4]);
if ans then
begin
for g:=2 to pic[pic1[k].z[1]].max[4] do
pic1[k].q[g]:=rotcorner(pic[pic1[k].z[1]].s[g],rim); //--->
return value
pic1[k].q[1]:=4; //for map with next
pic
if counter=2 then pic1[k].z[2]:=270;
if counter=5 then pic1[k].z[2]:=0;
if counter=8 then pic1[k].z[2]:=90;
counter:=counter+1;
x:=k;
show1[counter+1]:=pic1[k].z[1];
comer[counter+1]:=pic1[k].z[2];
end
else
begin
pic1[k]:=pic[k];
k:=k+1;
end
end
else
begin
pic1[k]:=pic[k];
k:=k+1;
end
end
else
if c2 and d2 then
begin
rim:=pic1[k].p[1];
rim2:=pic1[k].q[1];
display:="";
for g:=2 to pic1[k].max[2] do
pic1[k].s[g]:=rotcorner(pic1[k].q[g],rim2);

```

```

for g:=2 to pic1[k].max[2] do //=====> reverse
value
temp[g]:=pic1[k].s[g];
for g:=2 to pic1[k].max[2] do
pic1[k].s[g]:=temp[pic1[k].max[2]+2-g];
pic1[k].s[1]:=4;
ans:=compare(pic1[w].q,pic1[w].max[l],pic1[k].s,pic1
[k].max[4]);
if ans then
begin
for g:=2 to pic1[k].max[1] do
pic1[k].q[g]:=rotcorner(pic1[k].p[g],rim); //---> return
value for
pic1[k].q[1]:=4; //map with next pic
if counter=2 then pic1[k].z[2]:=180;
if counter=5 then pic1[k].z[2]:=270;
if counter=8 then pic1[k].z[2]:=0;
counter:=counter+1;
x:=k;
show1[counter+1]:=pic1[k].z[1];;
comer[counter+1]:=pic1[k].z[2];
end
else
begin
pic1[k]:=pic[k];
k:=k+1;
end
end
else
if a2 and d2 then
begin
rim:=pic1[k].r[1];
display:="";
for g:=2 to pic1[k].max[3] do
pic1[k].r[g]:=rotcorner(pic1[k].r[g],rim);
for g:=2 to pic1[k].max[3] do
//=====> reverse value
temp[g]:=pic1[k].r[g];
for g:=2 to pic1[k].max[3] do
pic1[k].r[g]:=temp[pic1[k].max[3]+2-g];
pic1[k].r[1]:=4;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

ans:=compare(pic1[w].q,pic1[w].max[l],pic1[k].r,pic1
[k].max[3]); //----???
if ans then
begin
pic1[k].q[1]:=4;
if counter=2 then pic1[k].z[2]:=90;
if counter=5 then pic1[k].z[2]:=180;
if counter=8 then pic1[k].z[2]:=270;
counter:=counter+1;
x:=k;
show1[counter+1]:=pic1[k].z[1];
comer[counter+1]:=pic1[k].z[2];
end
else
begin
pic1[k]:=pic[k];
k:=k+1;
end
end;
//-----
if ans then
begin
id[k]:=1;
j:=5;
end;

until ans or (k=5);
end;
until (counter=11);

for n:=2 to pic1[5].max[1] do
pic1[5].r[n]:=forgun(pic1[5].p[n],pic1[5].p[1]);
//=====> obposit edge
pic1[j].r[1]:=6;

```

```
for n:=1 to 60 do
```

```
begin
```

```

gun[1].r1[n]:=pic1[12].r[n];
gun[2].r1[n]:=pic1[11].r[n];
gun[3].r1[n]:=pic1[10].r[n];
gun[4].r1[n]:=pic1[9].r[n];
gun[5].r1[n]:=pic1[8].r[n];
gun[6].r1[n]:=pic1[7].r[n];
gun[7].r1[n]:=pic1[6].r[n];
gun[8].r1[n]:=pic1[5].r[n];
end;
gun[1].max:=pic[2].max[4];
gun[2].max:=pic[3].max[4];
gun[3].max:=pic[8].max[3];
gun[4].max:=pic[12].max[3];
gun[5].max:=pic[15].max[2];
gun[6].max:=pic[14].max[2];
gun[7].max:=pic[9].max[2];
gun[8].max:=pic[5].max[1];
end;

```

```
procedure TForm4.Button3Click(Sender:TObject);
```

```
type
```

```
mm=record
```

```
p:array[1..60] of integer ;
```

```
max:integer
```

```
end;
```

```
var i,j,k,g,count,T,a,b,f,l,n,u,z:integer;
```

```
comp,comp1:real;
```

```
midpic:array[1..4] of da;
```

```
pbound1,pbound2:array[1..4] of mm;
```

```
pfq,pfr,pfs,pfp,side2:array[1..60] of integer;
```

```
buffer:array[1..4,1..4] of boolean;
```

```
display,display1:string;
```

```
begin
```

```
midpic[1]:=pic[6];
```

```
midpic[2]:=pic[7];
```

```
midpic[3]:=pic[10];
```

```
midpic[4]:=pic[11];
```

```
pbound1[1].max:=gun[1].max;
```

```
pbound1[2].max:=gun[2].max;
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

pbound1[3].max:=gun[5].max;
pbound1[4].max:=gun[6].max;
pbound2[1].max:=gun[8].max;
pbound2[2].max:=gun[2].max;
pbound2[3].max:=gun[4].max;
pbound2[4].max:=gun[7].max;

```

```

n:=0 ;
for g:=1 to 60 do
begin
n:=n+1;
pbound1[1].p[g]:=rtop(gun[1].r1,gun[1].max,n);
pbound1[2].p[g]:=rtop(gun[2].r1,gun[2].max,n);
pbound1[3].p[g]:=rtop(gun[5].r1,gun[5].max,n);
pbound1[4].p[g]:=rtop(gun[6].r1,gun[6].max,n);
pbound2[1].p[g]:=rtop(gun[8].r1,gun[8].max,n);
pbound2[2].p[g]:=rtop(gun[3].r1,gun[2].max,n);
pbound2[3].p[g]:=rtop(gun[4].r1,gun[4].max,n);
pbound2[4].p[g]:=rtop(gun[7].r1,gun[7].max,n);
end;
for i:=1 to 4 do
for j:=1 to 4 do
buffer[i][j]:= false;
// end;
for i := 1 to 4 do(pbound1)
begin
for j := 1 to 4 do(midpic)
begin

```

```

n:=0;
for f:= 1 to 60 do
begin
n:=n+1;
pfp[f] := midpic[j].p[f];
pfq[f] := switchqtop(midpic[j].q,n);
pfr[f] := switchrtop(midpic[j].r,n);
pfs[f] := switchstop(midpic[j].s,n);
end;
{side}

```

```

count := 0;
u:= pbound1 [j].max;

```

```

for k := 1 to u do

```

```

if pbound1[i].p[k] = pfp[k]
then count := count+1;
comp :=(count/u)*100;
if (comp>26.5)and (comp<43.5)
then buffer[j][1] := true
else buffer[j][1]:=false;

```

```

count := 0;
for k := 1 to u do
if pbound1[i].p[k] = pfq[k]
then count := count+1;

comp := (count/u)*100;
if (comp>26.5)and (comp<43.5)
then buffer[j][2] := true
else buffer[j][2] := false;

```

```

count := 0;
for k := 1 to u do
if pbound1[i].p[k] = pfr[k]
then count := count+1;

comp := (count/u)*100;
if (comp>26.5)and (comp<43.5)
then buffer[j][3] := true
else buffer[j][3] := false;

```

```

count := 0;
for k := 1 to u do

if pbound1[i].p[k] = pfs[k]
then count := count+1;

comp := (count/u)*100;
if (comp>26.5)and (comp<43.5)
then buffer[j][4] := true
else buffer[j][4] := false;

```

```

end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

////////////////////////////////////

////////////////////////////////////

```
if i=1 then
begin
for a := 1 to 4 do
begin
for b := 1 to 4 do
begin
n:=0;
if buffer[a][b]=true then
begin
n:=n+1;
case b of
1: begin
for f:= 1 to 60 do
side2[f]:= switchstop(midpic[i].s,n);
count := 0;
u:=pbound2[i].max;
for T := 1 to u do
begin
if pbound2[i].p[T] = side2[T] then
count := count+1;
end;
comp1 := (count/u)*100;
if comp1 >= 80 then
buffer[a][b] := true else
buffer[a][b] := false;
end;
2: begin
for T:=1 to 60 do
side2[T]:= midpic[i].p[T];
count := 0;

for T := 1 to u do
begin
if pbound2[i].p[T] = side2[T] then
count := count+1;
end;
comp1 := (count/u)*100;
```

```
if comp1 >= 80 then
buffer[a][b] := true else
buffer[a][b] := false;
end;
3: begin
for f:= 1 to 60 do
side2[f]:= switchqtop(midpic[i].q,n);
count := 0;
for T := 1 to u do
begin
if pbound2[i].p[T] = side2[T] then
count := count+1;
end;
comp1 := (count/u)*100;
if comp1 >= 80 then
buffer[a][b] := true else
buffer[a][b] := false;
end;
4: begin
for f:= 1 to 60 do
side2[f]:= switchrtop(midpic[i].r,n);
count := 0;
for T := 1 to u do
begin
if pbound2[i].p[T] = side2[T] then
count := count+1;
end;
comp1 := (count/u)*100;
if comp1 >= 80 then
buffer[a][b] := true else
buffer[a][b] := false;
end;
end;
end;
end;
count:=0;
for a:=1 to 4 do
for b:=1 to 4 do
if buffer[a][b]= true
then count:=count+1;
if count<>1 then begin complete[1]:=7;
com[1]:=90;
```



```

end
else for b:= 1 to 4 do
  for a:= 1 to 4 do
    begin
      if buffer[a][b]=true then
        case a of
          1:complete[1]:=6;
          2:complete[1]:=7;
          3:complete[1]:=10;
          4:complete[1]:=11;
        end;
        case b of
          1:com[1]:=0;
          2:com[1]:=270;
          3:com[1]:=180;
          4:com[1]:=90;
        end;
      end;
    end;
  end;
end;

//-----//
if i=2 then
begin
  for a := 1 to 4 do
    begin
      for b := 1 to 4 do
        begin
          n:=0;
          if buffer[a][b]=true then
            begin
              n:=n+1;
              u:=pbound2[i].max;
              case b of
                1: begin
                  for f:= 1 to 60 do
                    side2[f]:= switchqtop(midpic[i].q,n);
                  count := 0;
                  for T := 1 to u do
                    begin
                      if pbound2[i].p[T] = side2[T] then
                        count := count+1;
                      end;
                    end;
                  comp1 := (count/u)*100;
                end
              if comp1 >= 80 then
                buffer[a][b] := true else
                buffer[a][b] := false;
              end;
            end
          2: begin
            for f:= 1 to 60 do
              side2[f]:= switchrtop(midpic[i].r,n);
            count := 0;
            for T := 1 to u do
              begin
                if pbound2[i].p[T] = side2[T] then
                  count := count+1;
                end;
              end;
            comp1 := (count/u)*100;
            if comp1 >= 80 then
              buffer[a][b] := true else
              buffer[a][b] := false;
            end;
          end
          3: begin
            for f:= 1 to 60 do
              side2[f]:= switchstop(midpic[i].s,n);
            count := 0;
            for T := 1 to u do
              begin
                if pbound2[i].p[k] = side2[k] then
                  count := count+1;
                end;
              end;
            comp1 := (count/u)*100;
            if comp1 >= 80 then
              buffer[a][b] := true else
              buffer[a][b] := false;
            end;
          end
          4: begin
            for T:= 1 to 60 do
              side2[T]:= midpic[i].p[T];
            count := 0;
            for T := 1 to u do
              begin
                if pbound2[i].p[k] = side2[k] then
                  count := count+1;
                end;
              end;
            comp1 := (count/u)*100;
            if comp1 >= 80 then
              buffer[a][b] := true else
              buffer[a][b] := false;
            end;
          end
        end
      end
    end
  end
end

```



```

        buffer[a][b] := true else
        buffer[a][b] := false;
    end;
end;
end;
end;
count:=0;
for a:=1 to 4 do
    for b:= 1 to 4 do
        if buffer[a][b]= true
        then count:=count+1;
    if count<>1 then begin complete[2]:=6;
        com[2]:=180;
        end
    else for b:=1 to 4 do
        for a:= 1 to 4 do
            begin
                if buffer[a][b]=true then
                    case a of
                        1:complete[2]:=6;
                        2:complete[2]:=7;
                        3:complete[2]:=10;
                        4:complete[2]:=11;
                    end;
                    case b of
                        1:com[2]:=0;
                        2:com[2]:=270;
                        3:com[2]:=180;
                        4:com[2]:=90;
                    end;
                end;
            end;
        end;
    end;
end;
//-----//
if i=3 then
begin
for a := 1 to 4 do
    begin
        for b := 1 to 4 do
            begin
                n:=0;
                if buffer[a][b]=true then
                    begin

```

```

n:=n+1;
u:=pbound2[i].max;
case b of
1: begin
    for f:= 1 to 60 do
        side2[f]:= switchstop(midpic[i].s,n);
        count := 0;
        for T := 1 to u do
            begin
                if pbound2[i].p[T] = side2[T] then
                    count := count+1;
            end;
        comp1 := (count/u)*100;
        if comp1 >= 80 then
            buffer[a][b] := true else
            buffer[a][b] := false;
        end;
2: begin
    for T:= 1 to 60 do
        side2[T]:= midpic[i].p[T];
        count := 0;
        for T := 1 to u do
            begin
                if pbound2[i].p[T] = side2[T] then
                    count := count+1;
            end;
        comp1 := (count/u)*100;
        if comp1 >= 80 then
            buffer[a][b] := true else
            buffer[a][b] := false;
        end;
3: begin
    for f:= 1 to 60 do
        side2[f]:= switchqtop(midpic[i].q,n);
        count := 0;
        for T := 1 to u do
            begin
                if pbound2[i].p[T] = side2[T] then
                    count := count+1;
            end;
        comp1 := (count/u)*100;
        if comp1 >= 80 then
            buffer[a][b] := true else

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

buffer[a][b] := false;
end;
4: begin
  for f:= 1 to 60 do
    side2[f]:= switchrtop(midpic[i].r,n);
    count := 0;
    for T := 1 to u do
      begin
        if pbound2[i].p[T] = side2[T] then
          count := count+1;
        end;
        comp1 := (count/u)*100;
        if comp1 >= 80 then
          buffer[a][b] := true else
            buffer[a][b] := false;
        end;
      end;
    end;
    count:=0;
    for a:=1 to 4 do
      for b:= 1 to 4 do
        if buffer[a][b]= true
          then count:=count+1;
        if count<>1 then begin complete[3] :=11;
          com[3]:=0;
          end
        else for b:=1 to 4 do
          for a:= 1 to 4 do
            begin
              if buffer[a][b]=true then
                case a of
                  1:complete[3]:=6;
                  2:complete[3]:=7;
                  3:complete[3]:=10;
                  4:complete[3]:=11;
                end;
              case b of
                1:com[3]:=180;
                2:com[3]:=90;
                3:com[3]:=0;
                4:com[3]:=270;
              end;
            end;
          end;
        end;
      end;
    end;
  end;
  //-----//
  if i=4 then
    begin
      for a := 1 to 4 do
        begin
          for b := 1 to 4 do
            begin
              n:=0;
              if buffer[a][b]=true then
                begin
                  n:=n+1;
                  u:=pbound2[i].max;
                  case b of
                    1: begin
                      for f:= 1 to 60 do
                        side2[f]:= switchqtop(midpic[i].q,n);
                        count := 0;
                        for T := 1 to u do
                          begin
                            if pbound2[i].p[T] = side2[T] then
                              count := count+1;
                            end;
                            comp1 := (count/u)*100;
                            if comp1 >= 80 then
                              buffer[a][b] := true else
                                buffer[a][b] := false;
                            end;
                          2: begin
                            for f:= 1 to 60 do
                              side2[f]:= switchrtop(midpic[i].r,n);
                              count := 0;
                              for T := 1 to u do
                                begin
                                  if pbound2[i].p[T] = side2[T] then
                                    count := count+1;
                                  end;
                                  comp1 := (count/u)*100;
                                  if comp1 >= 80 then
                                    buffer[a][b] := true else
                                      buffer[a][b] := false;
                                  end;
                                end;
                              end;
                            end;
                          end;
                        end;
                      end;
                    end;
                  end;
                end;
              end;
            end;
          end;
        end;
      end;
    end;
  end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

end;
3: begin
    for f:= 1 to 60 do
        side2[f]:= switchstop(midpic[i].s,n);
    count := 0;
    for T := 1 to u do
        begin
            if pbound2[i].p[T] = side2[T] then
                count := count+1;
            end;
            comp1 := (count/u)*100;
            if comp1 >= 80 then
                buffer[a][b] := true else
                buffer[a][b] := false;
            end;
        end;
    4: begin
        for T:= 1 to 60 do
            side2[T]:= midpic[i].p[T];
            count := 0;
            for T := 1 to u do
                begin
                    if pbound2[i].p[k] = side2[k] then
                        count := count+1;
                    end ;
                    comp1 := (count/u)*100;
                    if comp1 >= 80 then
                        buffer[a][b] := true else
                        buffer[a][b] := false;
                    end;
                end;
            end;
        end;
    end;
    end;
    count:=0;
    for a:=1 to 4 do
        for b:= 1 to 4 do
            if buffer[a][b]= true
            then count:=count+1;
            if count<>1 then begin complete[4]:=10;
                com[4]:=0;
            end
            else for b:=1 to 4 do
                for a:=1 to 4 do
                    begin
                        if buffer[a][b]=true then
                            case a of
                                1:complete[4]:=6;
                                2:complete[4]:=7;
                                3:complete[4]:=10;
                                4:complete[4]:=11;
                            end;
                            case b of
                                1:com[4]:=180;
                                2:com[4]:=90;
                                3:com[4]:=0;
                                4:com[4]:=270;
                            end;
                        end;
                        end;
                        //////////////////////////////////////
                    end;
                    show1[13]:=complete[2];
                    show1[14]:=complete[1];
                    show1[15]:=complete[3];
                    show1[16]:=complete[4];
                    corner[13]:=com[1];
                    corner[14]:=com[2];
                    corner[15]:=com[3];
                    corner[16]:=com[4];
                //end;
                {for z:= 1 to 16 do
                    begin
                        case show1[z] of
                            1:show3[z]:=13;
                            2:show3[z]:=10;
                            3:show3[z]:=7;
                            4:show3[z]:=6;
                            5:show3[z]:=16;
                            6:show3[z]:=14;
                            7:show3[z]:=9;
                            8:show3[z]:=11;
                            9:show3[z]:=15;
                            10:show3[z]:=5;
                            11:show3[z]:=1;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

12:show3[z]:=10;
13:show3[z]:=2;
14:show3[z]:=4;
15:show3[z]:=3;
16:show3[z]:=8;
end;}
display:=' ';
display1:=' ';
for n:=1 to 16 do
begin
display:=display+inttostr(show1[n])+ ' ';
label1.caption:=display;
display1:=display1+inttostr(comer[n])+ ' ';
label2.caption:=display1;
end;
end;

////////////////////
function rtop (a:array of
integer;max:integer;n:integer):integer ;
var i,j:integer;
out2:array[1..60] of integer;
l:integer;
b,ab,out,out1:array[1..60] of string;
r:integer;
begin
//do switching//
for i:=1 to 59 do
ab[i]:= inttostr(a[i]);
b[1]:='6';
j:=2;
for i:=max-1 downto 1 do
begin
b[j]:=ab[i];
j:=j+1;
end;
for i:= max to 59 do
begin
b[j]:=ab[i];
j:=j+1;
end;
for l:= 1 to 60 do
out[l]:= b[l];
end;
//do changing//
for r:= 1 to 60 do
begin
if out[r]='0' then out1[r]:='4'
else if out[r]='1' then out1[r]:='5'
else if out[r]='2' then out1[r]:='6'
else if out[r]='3' then out1[r]:='7'
else if out[r]='4' then out1[r]:='0'
else if out[r]='5' then out1[r]:='1'
else if out[r]='6' then out1[r]:='2'
else if out[r]='7' then out1[r]:='3' ;
out2[r]:=strtoint(out1[r]);
end;
result:= out2[n];
end;
//-----//
function switchqtop (a:array of integer;n:integer):integer ;
var i:integer;
out2:array[1..60] of integer;
l:integer;
b,out,out1:array[1..60] of string;
r:integer;
begin
//do str to array//
b[1]:='4';
for i:=2 to 60 do
b[i]:=inttostr(a[i]);
for l:= 1 to 60 do

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

begin
out[i]:= b[i];
end;

//do changing//

for r:= 1 to 60 do
begin
if out[r]='0' then out1[r]:='6'
else if out[r]='1' then out1[r]:='7'
else if out[r]='2' then out1[r]:='0'
else if out[r]='3' then out1[r]:='1'
else if out[r]='4' then out1[r]:='2'
else if out[r]='5' then out1[r]:='3'
else if out[r]='6' then out1[r]:='4'
else if out[r]='7' then out1[r]:='5' ;

out2[r]:= strtoint(out1[r]);
end;
result:= out2[n];
end;

function switchrtop (a:array of integer;n:integer):integer ;
var i:integer;
out2:array[1..60] of integer;
l:integer;
b,out,out1:array[1..60] of string;
r:integer;
begin
//do str to array//
b[1]:='6';
for i:= 2 to 60 do
b[i]:=inttostr(a[i]);
for l:= 1 to 60 do
begin
out[l]:= b[l];
end;

//do changing//

for r:= 1 to 60 do
begin
if out[r]='0' then out1[r]:='2'
else if out[r]='1' then out1[r]:='3'
else if out[r]='2' then out1[r]:='4'
else if out[r]='3' then out1[r]:='5'
else if out[r]='4' then out1[r]:='6'
else if out[r]='5' then out1[r]:='7'
else if out[r]='6' then out1[r]:='0'
else if out[r]='7' then out1[r]:='1' ;

out2[r]:= strtoint(out1[r]);
end;
result:= out2[n];
end;

function switchstop (a:array of integer;n:integer): integer ;
var i:integer;
out2:array[1..60] of integer;
l:integer;
out,b,out1:array[1..60] of string;
r:integer;
begin
//do str to array//
b[1]:='0';
for i:= 2 to 60 do
b[i]:=inttostr(a[i]);
for l:= 1 to 60 do
begin
out[l]:= b[l];
end;

//do changing//

for r:= 1 to 60 do
begin
if out[r]='0' then out1[r]:='4'
else if out[r]='1' then out1[r]:='5'
else if out[r]='2' then out1[r]:='6'
else if out[r]='3' then out1[r]:='7'
else if out[r]='4' then out1[r]:='0'
else if out[r]='5' then out1[r]:='1'
else if out[r]='6' then out1[r]:='2'
else if out[r]='7' then out1[r]:='3' ;

out2[r]:= strtoint(out1[r]);
end;
result:= out2[n];
end;

function switchtop (a:array of integer;n:integer):integer ;
var i:integer;
out2:array[1..60] of integer;
l:integer;
b,out,out1:array[1..60] of string;
r:integer;
begin
//do str to array//
b[1]:='6';
for i:= 2 to 60 do
b[i]:=inttostr(a[i]);
for l:= 1 to 60 do
begin
out[l]:= b[l];
end;

//do changing//

for r:= 1 to 60 do
begin
if out[r]='0' then out1[r]:='2'
else if out[r]='1' then out1[r]:='3'
else if out[r]='2' then out1[r]:='4'
else if out[r]='3' then out1[r]:='5'
else if out[r]='4' then out1[r]:='6'
else if out[r]='5' then out1[r]:='7'
else if out[r]='6' then out1[r]:='0'
else if out[r]='7' then out1[r]:='1' ;

out2[r]:= strtoint(out1[r]);
end;
result:= out2[n];
end;

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

else if out[r]='5' then out1[r]:='7'
else if out[r]='6' then out1[r]:='0'
else if out[r]='7' then out1[r]:='1' ;

out2[r]:= strtoint(out1[r]);
end;
result:= out2[n];
end;

//-----//

end.

```



โปรแกรม Assembly

```
org      0000h
        jmp      setport
org      0003h
        jmp      count
```

```
;===== init
```

```
=====
```

```
;===== setport =====
```

```
setport: porta    equ   0F800h
          portb    equ   0F801h
          portcon   equ   0F803h
```

```
        mov      a,#89h          ;A=out B=out
```

C=in

```
        mov      dptr,#portcon
```

```
        movx     @dptr,a
```

```
;===== init register =====
```

```
init:    mov      SCON,#0F0h      ;Set serial
```

mode3 multiprocessor mode

```
        mov      PCON,#00h        ;smod=0
```

```
        mov      TMOD,#26h        ;Timer1
```

mode2 Counter0 mode2

```
        mov      TH1,#0FDh        ;BAUD 9.6 K
```

```
        setb      tr1              ;Start Timer1
```

```
        mov      TH0,#0F8h;8 times ---> overflow
```

(auto reload)

```
        mov      IE,#81h          ;enable
```

interrup(ea),enable int0(ex0),dis serial(es)

```
        setb      it0              ;edge low
```

interrup(TCON)

```
        clr       ri               ;for recieve
```

data

```
;===== set coordinate =====
```

```
        left      equ      05h      ;left
```

M1=M2=01 ----> 0101h

```
        right     equ      0Ah      ;right
```

M1=M2=10 ----> 1010h

```
        front     equ      09h      ;front M1<>
```

M2 ----> 1001h

```
        back      equ      06h      ;back M1<>
```

M2 ----> 0110h

```
up       equ      80h              ;up M3 = 01 --
```

```
--> 1000 0000h
```

```
down     equ      40h              ;down M3 =
```

```
10 ----> 0100 0000h
```

```
turn_r   equ      20h              ;turn right M4
```

```
= 01 ----> 0001 0000h
```

```
turn_l   equ      10h              ;turn left M4 =
```

```
10 ----> 0010 0000h
```

```
;===== recieve data =====
```

```
        mov      dptr,#8000h      ;point RAM for
```

data

```
recieve: jnb      ri,$              ;Wait for data
```

```
        mov      a,sbuf            ;read data
```

```
        clr       ri
```

```
        movx     @dptr,a            ;keep data
```

```
        inc       dpl
```

```
        mov      a,dpl
```

```
        cjne     a,#20h,recieve    ;16 data
```

position + 16 degree

```
        clr       tr1              ;stop timer1
```

```
;===== control arm
```

```
=====
```

```
; r1 = exis x (next) , delay(z)  r2 = exis y (next) , delay(z)
```

```
; r4 = exis x (now)              r5 = exis y (now)
```

```
; r3 = direct (robot)           r6 = address keep
```

data (pos,degree)

```
; r7 = data (degree) , compare position(xy) , delay error
```

(xy)

```
; r0 = delay error(xy)
```

```
        mov      r4,#00h
```

```
        mov      r5,#00h
```

```
        mov      dptr,#portb
```

```
        mov      a,#00h
```

```
        movx     @dptr,a
```

```
        mov      r1,#01h            ;1pls
```

```
        mov      r2,#04h            ;4pls
```

```
        mov      r6,#00h            ;pic1
```

```
        call     process
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

			call	process	
mov	r1,#02h	;2pls			
mov	r2,#04h	;4pls	mov	r1,#01h	;1pls
mov	r6,#01h	;pic2	mov	r2,#01h	;1pls
call	process		mov	r6,#09h	;pic10
			call	process	
mov	r1,#03h	;3pls			
mov	r2,#04h	;4pls	mov	r1,#01h	;1pls
mov	r6,#02	;pic3	mov	r2,#02h	;2pls
call	process		mov	r6,#0Ah	;pic11
			call	process	
mov	r1,#04h	;4pls			
mov	r2,#04h	;4pls	mov	r1,#01h	;1pls
mov	r6,#03	;pic4	mov	r2,#03h	;3pls
call	process		mov	r6,#0Bh	;pic12
			call	process	
lcall	home		lcall	home	
mov	r1,#04h	;4pls			
mov	r2,#03h	;3pls	mov	r1,#02h	;2pls
mov	r6,#04h	;pic5	mov	r2,#03h	;3pls
call	process		mov	r6,#0Ch	;pic13
			call	process	
mov	r1,#04h	;4pls			
mov	r2,#02h	;2pls	mov	r1,#03h	;3pls
mov	r6,#05h	;pic6	mov	r2,#03h	;3pls
call	process		mov	r6,#0Dh	;pic14
			call	process	
mov	r1,#04h	;4pls			
mov	r2,#01h	;1pls	mov	r1,#03h	;3pls
mov	r6,#06h	;pic7	mov	r2,#02h	;2pls
call	process		mov	r6,#0Eh	;pic15
			call	process	
mov	r1,#03h	;3pls			
mov	r2,#01h	;1pls	mov	r1,#02h	;2pls
mov	r6,#07h	;pic8	mov	r2,#02h	;2pls
call	process		mov	r6,#0Fh	;pic16
			call	process	
lcall	home		lcall	home	
			jmp	end	
mov	r1,#02h	;2pls			
mov	r2,#01h	;1pls			
mov	r6,#08h	;pic9			

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

===== process
===== (sub control arm)
process: call control
        lcall z_con ;for take
jigsaw   left
        mov dph,#80h
        mov dpl,r6 ;set for read
data position
        call zone
        lcall z_zone ;for put jigsaw
        ret

===== home
===== (sub control arm)
home:    clr ea
        mov dptr,#porta
home1:   mov a,#back
        movx @dptr,a
        setb p1.7
        jb p1.7,home1
        mov a,#00h
        movx @dptr,a
        mov dptr,#porta
home2:   mov a,#left
        movx @dptr,a
        setb p1.6
        jb p1.6,home2
        mov a,#00h
        movx @dptr,a
        mov r4,#00h
        mov r5,#00h
        setb ea
        ret

===== zone
===== (sub process)
zone:    movx a,@dptr ;load data
        lcall compare
        call control
        ret

===== control
===== (sub process)
control: mov a,r4

subb     a,r1 ;now(x)-next
        jc right1
        call m_left ;now > next =
        jmp exisy
right1:  clr c
        call m_right ;now < next =
        right
exisy:   mov a,r5
        subb a,r2 ;now(y)-next
        jc front1
        call m_back ;now > next =
        jmp ret
front1:  clr c
        call m_front ;now < next =
        ret1: ret

===== move xy ===== (sub control)
m_right: mov r3,#right
        call gox
        ret
m_left:  mov r3,#left
        call gox
        ret
m_front: mov r3,#front
        call goy
        ret
m_back:  mov r3,#back
        call goy
        ret

===== (sub move xy)
gox:     mov a,r4
        xrl a,r1
        cjne a,#00h,go1 ;now <> new
        jmp ret_g
goy:     mov a,r5

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

        xrl        a,r2
        cjne      a,#00h,go1      ;now <> new      ;=====
        jmp       ret_g            ;=====
                                     =====(sub move z)
go1:      mov     dptr,#porta      ;control motor
        (a.0,a.1=M2 a.2,a.3=M1)
        mov     a,r3
        movx    @dptr,a          ;ena M1,M2
        mov     r7,a
        cjne    r7,#00h,$        ;now = new ?
        mov     a,#00h
        movx    @dptr,a          ;dis M1,M2
ret_g:    ret

;=====move
z=====
z_con:    call    godown          ;down
        mov     a,#01h
        mov     dptr,#portb
        movx    @dptr,a          ;take
        call    delay2
        call    goup            ;up
        ret

z_zone:   mov     a,r6            ;go to read
degree
        add     a,#10h
        mov     dpl,a
        mov     dph,#80h
        movx    a,@dptr          ;read degree
        mov     r7,a            ;r7 = degree
        mov     a,#turn_r        ;turn right
        call    turn

        call    godown          ;down
        mov     a,#00h
        mov     dptr,#portb
        movx    @dptr,a          ;put
        call    delay2
        call    goup            ;up
        mov     a,#turn_l        ;turn left
        call    turn
        ret

godown:   mov     a,#down        ;down
        mov     dptr,#porta
        movx    @dptr,a          ;ena motor
        call    delay1
        mov     a,#00h
        movx    @dptr,a          ;dis motor
        ret

goup:     mov     a,#up          ;up
        mov     dptr,#porta
        movx    @dptr,a          ;ena motor
        call    delay3
        mov     a,#00h
        movx    @dptr,a          ;dis motor
        ret

;=====turn=====
turn=====
turn:     mov     dptr,#porta    ;ena motor
        movx    @dptr,a
        mov     a,r7
        call    compare_d
turn3:    cjne    a,#00h,turn1
        jmp     turn2
turn1:    call    delay2
        dec     a
        jmp     turn3
turn2:    movx    @dptr,a        ;dis motor
        ret

;=====delay2 for
turn=====
delay2:   mov     r0,#03h
loop4:    dec     r0
        mov     r1,#0AFh
loop3:    dec     r1
        mov     r2,#0FFh
        djnz    r2,$
        cjne    r1,#00h,loop3
        cjne    r0,#00h,loop4

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

ret
;=====delay for
down=====
delay1:  mov    r0,#04h
loop2:   dec     r0
        mov     r1,#0FFh
loop1:   dec     r1
        mov     r2,#0FFh
        djnz    r2,$
        cjne    r1,#00h,loop1
        cjne    r0,#00h,loop2
        ret

;=====delay for
up=====
delay3:  mov    r0,#04h
loop6:   dec     r0
        mov     r1,#0FFh
loop5:   dec     r1
        mov     r2,#0FFh
        djnz    r2,$
        cjne    r1,#00h,loop5
        cjne    r0,#00h,loop6
        ret

;=====interrupt=====
count:   clr     ea           ;dis interrupt
        cjne    r3,#left,chk1
        dec     r4           ;left = dec
now(x)   mov     a,r4
        xrl     a,r1         ;compare
now,new  mov     r7,a
        cjne    r7,#00h,next3
        jmp     next4
chk1:    cjne    r3,#right,chk2
        inc     r4           ;right = inc
now(x)   mov     a,r4
        xrl     a,r1         ;compare
now,new  mov     r7,a
        cjne    r7,#00h,next3
        jmp     next4
chk2:    cjne    r3,#front,chk3
        inc     r5           ;front = inc
now(y)   mov     a,r5
        xrl     a,r2         ;compare
now,new  mov     r7,a
        cjne    r7,#00h,next3
        jmp     next4
chk3:    dec     r5           ;back = dec
now(y)   mov     a,r5
        xrl     a,r2         ;compare
now,new  mov     r7,a
        cjne    r7,#00h,next3
        jmp     next4
next3:   call    delay_err
        jmp     ret2
next4:   mov     a,#00h
        movx    @dptr,a       ;dis motor
ret2:    setb    ea
        reti

;=====delay
error=====
delay_err: mov    r0,#0ffh
        djnz    r0,$
        ret

;===== compare
=====
compare:  cjne    a,#'1',pic2
        mov     r1,#06h       ;exis x = 6
        mov     r2,#04h       ;exis y = 4
        jmp     ret
pic2:    cjne    a,#'2',pic3
        mov     r1,#07h       ;exis x = 7
        mov     r2,#04h       ;exis y = 4
        jmp     ret
pic3:    cjne    a,#'3',pic4

```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้


```

mov    r1,#08h           ;exis x = 8
mov    r2,#04h           ;exis y = 4
jmp    ret
pic4:  cjne    a,#'4',pic5
mov    r1,#09h           ;exis x = 9
mov    r2,#04h           ;exis y = 4
jmp    ret
pic5:  cjne    a,#'5',pic6
mov    r1,#06h           ;exis x = 6
mov    r2,#03h           ;exis y = 3
jmp    ret
pic6:  cjne    a,#'6',pic7
mov    r1,#07h           ;exis x = 7
mov    r2,#03h           ;exis y = 3
jmp    ret
pic7:  cjne    a,#'7',pic8
mov    r1,#09h           ;exis x = 8
mov    r2,#03h           ;exis y = 3
jmp    ret
pic8:  cjne    a,#'8',pic9
mov    r1,#08h           ;exis x = 9
mov    r2,#03h           ;exis y = 3
jmp    ret
pic9:  cjne    a,#'9',pic10
mov    r1,#06h           ;exis x = 6
mov    r2,#02h           ;exis y = 2
jmp    ret
pic10: cjne    a,#'a',pic11
mov    r1,#07h           ;exis x = 7
mov    r2,#02h           ;exis y = 2
jmp    ret
pic11: cjne    a,#'b',pic12
mov    r1,#08h           ;exis x = 8
mov    r2,#02h           ;exis y = 2
jmp    ret
pic12: cjne    a,#'c',pic13
mov    r1,#09h           ;exis x = 9
mov    r2,#02h           ;exis y = 2
jmp    ret
pic13: cjne    a,#'d',pic14
mov    r1,#06h           ;exis x = 6
mov    r2,#01h           ;exis y = 1
jmp    ret

pic14: cjne    a,#'e',pic15
mov    r1,#07h           ;exis x = 7
mov    r2,#01h           ;exis y = 1
jmp    ret
pic15: cjne    a,#'f',pic16
mov    r1,#08h           ;exis x = 8
mov    r2,#01h           ;exis y = 1
jmp    ret
pic16: cjne    a,#'0',error
mov    r1,#09h           ;exis x = 9
mov    r2,#01h           ;exis y = 1
jmp    ret
error: mov    r1,#0FFh     ;for error
mov    r2,#0FFh
jmp    ret
ret:   ret

compare_d: cjne    a,#'0',c_d1
mov    a,#00h
jmp    ret_d
c_d1:  cjne    a,#'1',c_d2
mov    a,#01h
jmp    ret_d
c_d2:  cjne    a,#'2',c_d3
mov    a,#02h
jmp    ret_d
c_d3:  cjne    a,#'3',c_derr
mov    a,#03h
jmp    ret_d
c_derr: mov    a,#10h
ret_d:  ret

end:   jmp     end
end

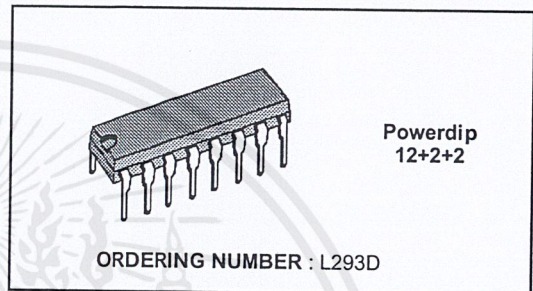
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PUSH-PULL FOUR CHANNEL DRIVER WITH DIODES

PRELIMINARY DATA

- 600mA. OUTPUT CURRENT CAPABILITY PER CHANNEL
- 1.2A PEAK OUTPUT CURRENT (NON REPETITIVE) PER CHANNEL
- ENABLE FACILITY
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5V (HIGH NOISE IMMUNITY)
- INTERNAL CLAMPS DIODES



DESCRIPTION

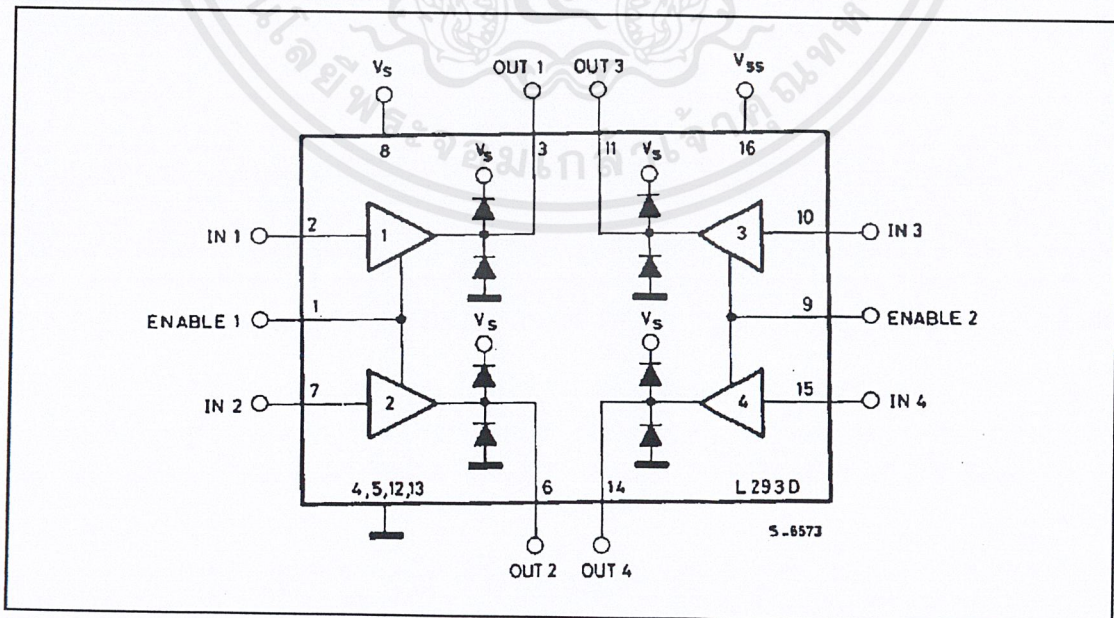
The L293D is a monolithic integrated high voltage, high current four channel driver designed to accept standard DTL or TTL logic levels and drive inductive loads (such as relays solenoids, DC and stepping motors) and switching power transistors.

To simplify use as two bridges is pair of channels is equipped with an enable input. A separate supply input is provided for the logic, allowing operation at a low voltage and internal clamp diodes are included.

This device is suitable for use in switching applications at frequencies up to 5 KHz.

The L293D is assembled in a 16 lead plastic package which has 4 center pins connected together and used for heatsinking.

BLOCK DIAGRAM

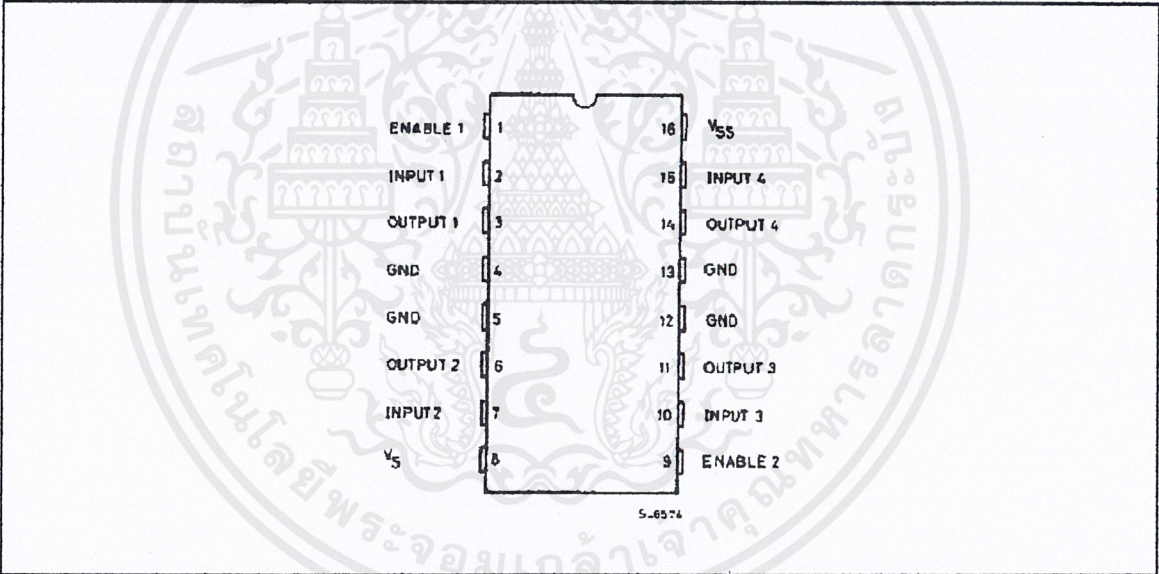


L293D

ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V _s	Supply Voltage	36	V
V _{ss}	Logic Supply voltage	36	V
V _i	Input voltage	7	V
V _{en}	Enable voltage	7	V
I _o	Peak output current (100μs non repetitive)	1.2	A
P _{tot}	Total power dissipation at T _{ground-pins} = 80°C	5	W
T _{stg} , T _j	Storage and junction temperature	-40 to 150	°C

CONNECTION DIAGRAM



THERMAL DATA

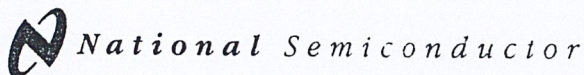
Symbol	Parameter	Value	Unit
R _{th-j-case}	Thermal resistance junction-case	max 14	°C/W
R _{th j-case}	Thermal resistance junction-ambient	max 80	°C/W

ELECTRICAL CHARACTERISTICS (For each channel, $V_s = 24V$, $V_{ss} = 5V$, $T_{amb} = 25^\circ C$, unless otherwise specified)

Symbol	Parameter	Test conditions	Min.	Typ.	Max.	Unit
V_s	Supply voltage		V_{ss}		36	V
V_{ss}	Logic supply voltage (pin 16)		4.5		36	V
I_s	Total quiescent supply current (pin 8)	$V_i = L \quad I_o = 0 \quad V_{en} = H$		2	6	mA
		$V_i = H \quad I_o = 0 \quad V_{en} = H$		16	24	
		$V_{en} = L$			4	
I_{ss}	Total quiescent logic supply current (pin 16)	$V_i = L \quad I_o = 0 \quad V_{en} = H$		44	60	mA
		$V_i = H \quad I_o = 0 \quad V_{en} = H$		16	22	
		$V_{en} = L$		16	24	
V_{IL}	Input low voltage (pin 2, 7, 10, 15)		-0.3		1.5	V
V_{IH}	Input high voltage (pin 2, 7, 10, 15)	$V_{ss} \leq 7V$	2.3		V_{ss}	V
		$V_{ss} > 7V$	2.3		7	
I_{IL}	Low voltage input current (pin 2, 7, 10, 15)	$V_{IL} = 1.5V$			-10	μA
I_{IH}	High voltage input current (pin 2, 7, 10, 15)	$2.3 \leq V_{IH} \leq V_{ss} - 0.6V$		30	100	μA
V_{enL}	Enable low voltage (pin 1, 9)		-0.3		1.5	V
V_{enH}	Enable high voltage (pin 1, 9)	$V_{ss} \leq 7V$	2.3		V_{ss}	V
		$V_{ss} > 7V$	2.3		7	
I_{enL}	Low voltage enable current (pin 1, 9)	$V_{enL} = 1.5V$		-30	-100	μA
I_{enH}	High voltage enable current (pin 1, 9)	$2.3V \leq V_{enH} \leq V_{ss} - 0.6V$			± 10	μA
V_{CEsatH}	Source output saturation voltage (pin 3, 6, 11, 14)	$I_o = -0.6A$		1.4	1.8	V
V_{CEsatL}	Sink output saturation voltage (pins 3, 6, 11, 14)	$I_o = 0.6A$			1.2	1.8
V_F	Clamp diode forward voltage	$I_o = 600 mA$		1.3		V
t_r	Rise time (*)	0.1 to 0.9 V_o		250		ns
t_f	Fall time (*)	0.9 to 0.1 V_o		250		ns
t_{on}	Turn-on delay (*)	0.5 V_i to 0.5 V_o		750		ns
t_{off}	Turn-off delay (*)	0.5 V_i to 0.5 V_o		200		ns

(*) See fig.1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหาและต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



LM139/LM239/LM339/LM2901/LM3302 Low Power Low Offset Voltage Quad Comparators

General Description

The LM139 series consists of four independent precision voltage comparators with an offset voltage specification as low as 2 mV max for all four comparators. These were designed specifically to operate from a single power supply over a wide range of voltages. Operation from split power supplies is also possible and the low power supply current drain is independent of the magnitude of the power supply voltage. These comparators also have a unique characteristic in that the input common-mode voltage range includes ground, even though operated from a single power supply voltage.

Application areas include limit comparators, simple analog to digital converters; pulse, squarewave and time delay generators; wide range VCO; MOS clock timers; multivibrators and high voltage digital logic gates. The LM139 series was designed to directly interface with TTL and CMOS. When operated from both plus and minus power supplies, they will directly interface with MOS logic—where the low power drain of the LM339 is a distinct advantage over standard comparators.

Advantages

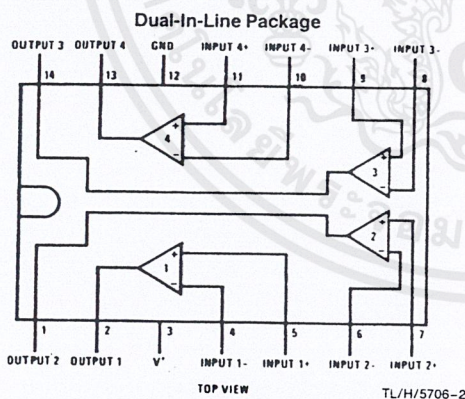
- High precision comparators
- Reduced V_{OS} drift over temperature

- Eliminates need for dual supplies
- Allows sensing near GND
- Compatible with all forms of logic
- Power drain suitable for battery operation

Features

- Wide supply voltage range
LM139 series, $2 V_{DC}$ to $36 V_{DC}$ or $\pm 1 V_{DC}$ to $\pm 18 V_{DC}$
LM139A series, LM2901 $2 V_{DC}$ to $28 V_{DC}$
LM3302 or $\pm 1 V_{DC}$ to $\pm 14 V_{DC}$
- Very low supply current drain (0.8 mA) — independent of supply voltage
- Low input biasing current $25 nA$
- Low input offset current $\pm 5 nA$ and offset voltage $\pm 3 mV$
- Input common-mode voltage range includes GND
- Differential input voltage range equal to the power supply voltage
- Low output saturation voltage 250 mV at 4 mA
- Output voltage compatible with TTL, DTL, ECL, MOS and CMOS logic systems

Connection Diagrams



Order Number LM139J, LM139J/883*, LM139AJ, LM139AJ/883**, LM239J, LM239AJ, LM339J, See NS Package Number J14A

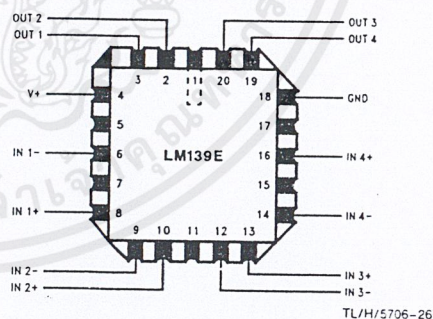
Order Number LM339AM, LM339M or LM2901M See NS Package Number M14A

Order Number LM339N, LM339AN, LM2901N or LM3302N

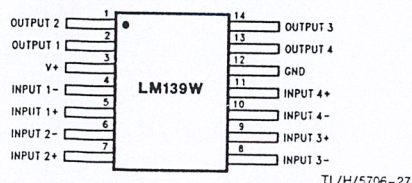
See NS Package Number N14A

*Available per JM38510/11201

**Available per SMD # 5962-8873901



Order Number LM139AE/883 or LM139E/883
See NS Package Number E20A



Order Number LM139AW/883 or LM139W/883*
See NS Package Number W14B

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications. (Note 10)

LM139/LM239/LM339
LM139A/LM239A/LM339A
LM2901
LM3302
LM2901

Supply Voltage, V^+	36 VDC or ± 18 VDC	28 VDC or ± 14 VDC	Operating Temperature Range	LM339/LM239/LM339	LM3302	LM2901
Differential Input Voltage (Note 8)	36 VDC	28 VDC	LM339/LM239A	LM3302	LM2901	LM3302
Input Voltage	-0.3 VDC to $+36$ VDC	-0.3 VDC to $+28$ VDC	LM239/LM239A	LM3302	LM2901	LM3302
Input Current ($V_{IN} < -0.3$ VDC, Note 3)	50 mA	50 mA	LM2901	LM3302	LM2901	LM3302
Power Dissipation (Note 1)	1050 mW	1050 mW	LM139/LM139A	LM3302	LM2901	LM3302
Molded DIP	1190 mW	1050 mW	Soldering Information	LM3302	LM2901	LM3302
Cavity DIP	760 mW	1050 mW	Dual-In-Line Package	LM3302	LM2901	LM3302
Small Outline Package			Soldering (10 seconds)	LM3302	LM2901	LM3302
Output Short-Circuit to GND, (Note 2)	Continuous	Continuous	Small Outline Package	LM3302	LM2901	LM3302
Storage Temperature Range	-65°C to $+150^\circ\text{C}$	-65°C to $+150^\circ\text{C}$	Vapor Phase (60 seconds)	LM3302	LM2901	LM3302
Lead Temperature (Soldering, 10 seconds)	260°C	260°C	Infrared (15 seconds)	LM3302	LM2901	LM3302
			See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.	LM3302	LM2901	LM3302
			ESD rating (1.5 kV in series with 100 pF)	LM3302	LM2901	LM3302

Electrical Characteristics

($V^+ = 5$ VDC, $T_A = 25^\circ\text{C}$, unless otherwise stated)

Parameter	Conditions	LM139A	LM239A, LM339A	LM139	LM239, LM339	LM2901	LM3302	Units
Input Offset Voltage (Note 9)		Min Typ Max	Min Typ Max	Min Typ Max	Min Typ Max	Min Typ Max	Min Typ Max	
Input Bias Current	$I_{IN}(+)$ or $I_{IN}(-)$ with Output in Linear Range, (Note 5), $V_{CM} = 0$ V	1.0 2.0 25	1.0 2.0 25	2.0 5.0 25	2.0 5.0 25	2.0 7.0 25	3 20 20	mVDC
Input Offset Current	$I_{IN}(+) - I_{IN}(-)$, $V_{CM} = 0$ V	3.0 25	5.0 50	3.0 25	5.0 50	5 50	3 100	nADC
Input Common-Mode Voltage Range	$V^+ = 30$ VDC (LM3302, $V^+ = 28$ VDC) (Note 6)	0 $V^+ - 1.5$	0 $V^+ - 1.5$	0 $V^+ - 1.5$	0 $V^+ - 1.5$	0 $V^+ - 1.5$	0 $V^+ - 1.5$	VDC
Supply Current	$R_L = \infty$ on all Comparators, $R_L = \infty$, $V^+ = 36$ V, (LM3302, $V^+ = 28$ VDC)	0.8 2.0	0.8 2.0 1.0 2.5	0.8 2.0 1.0 2.5	0.8 2.0 1.0 2.5	0.8 2.0 1.0 2.5	0.8 2.0 1.0 2.5	mADC
Voltage Gain	$R_L \geq 15$ k Ω , $V^+ = 15$ VDC, $V_O = 1$ VDC to 11 VDC	50 200	50 200	50 200	50 200	25 100	2 30	V/mV
Large Signal Response Time	$V_{IN} = \text{TTL Logic Swing}$, $V_{REF} = 1.4$ VDC, $V_{RL} = 5$ VDC, $R_L = 5.1$ k Ω , $V_{RL} = 5$ VDC, $R_L = 5.1$ k Ω , (Note 7)	300	300	300	300	300	300	ns
Response Time		1.3	1.3	1.3	1.3	1.3	1.3	μ s
Output Sink Current	$V_{IN}(-) = 1$ VDC, $V_{IN}(+) = 0$, $V_O \leq 1.5$ VDC	6.0 16	6.0 16	6.0 16	6.0 16	6.0 16	6.0 16	mADC

LM139/LM239/LM339/LM2901/LM3302

LM78XX Series Voltage Regulators

General Description

The LM78XX series of three terminal regulators is available with several fixed output voltages making them useful in a wide range of applications. One of these is local on card regulation, eliminating the distribution problems associated with single point regulation. The voltages available allow these regulators to be used in logic systems, instrumentation, HiFi, and other solid state electronic equipment. Although designed primarily as fixed voltage regulators these devices can be used with external components to obtain adjustable voltages and currents.

The LM78XX series is available in an aluminum TO-3 package which will allow over 1.0A load current if adequate heat sinking is provided. Current limiting is included to limit the peak output current to a safe value. Safe area protection for the output transistor is provided to limit internal power dissipation. If internal power dissipation becomes too high for the heat sinking provided, the thermal shutdown circuit takes over preventing the IC from overheating.

Considerable effort was expended to make the LM78XX series of regulators easy to use and minimize the number

of external components. It is not necessary to bypass the output, although this does improve transient response. Input bypassing is needed only if the regulator is located far from the filter capacitor of the power supply.

For output voltage other than 5V, 12V and 15V the LM117 series provides an output voltage range from 1.2V to 57V.

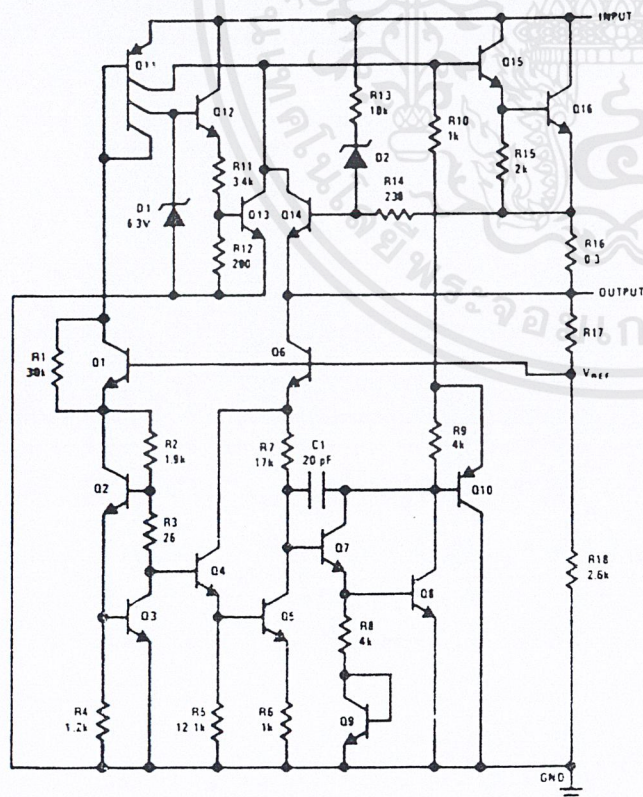
Features

- Output current in excess of 1A
- Internal thermal overload protection
- No external components required
- Output transistor safe area protection
- Internal short circuit current limit
- Available in the aluminum TO-3 package

Voltage Range

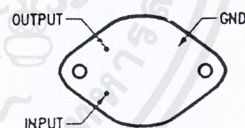
LM7805C	5V
LM7812C	12V
LM7815C	15V

Schematic and Connection Diagrams



TL/H/7746-1

**Metal Can Package
TO-3 (K)
Aluminum**

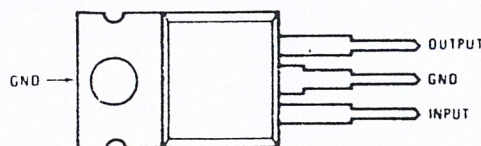


TL/H/7746-2

Bottom View

Order Number LM7805CK,
LM7812CK or LM7815CK
See NS Package Number KC02A

**Plastic Package
TO-220 (T)**



TL/H/7746-3

Top View

Order Number LM7805CT,
LM7812CT or LM7815CT
See NS Package Number T03B