หุ่นยนต์เคลื่อนที่อย่างชาญฉลาด

INTELLIGENT AUTONOMOUS ROBOT

โดย

นายปรนัย มุกด์ธนะอนันต์

นางสาวปิยณัฐ ทองสง

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

ภาควิชาวิศวกรรมสารสนเทศ

คณะวิศวกรรมศาสตร์

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2548

INTELLIGENT AUTONOMOUS ROBOT

BY

MR. PARANAI MUKTANA-ANAN

MISS PIYANAT THONGSONG

A THESIS SUBMITTED IN PARTAIL FULFILMENT OF

THE REQUIREMENT FOR THE DEGREE OF

BACHLOR IN DEPARTMENT OF ININFORMATION ENGINEERING

FACULTY OF ENGINEERING

KING MONGKUT'S INSTITUTE OF TECHNOLOGY LADKRABANG

2005

หัวข้อปริญญานิพนธ์	หุ่นยนต์เคลื่อนที่แบบชาญฉลาด
นักศึกษา	นายปรนัย มุกด์ธนะอนันต์ รหัสประจำตัว 45010435
	นางสาวปิยณัฐ ทองสง รหัสประจาตัว 45010468
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ.ดร.ปิติเขต สู้รักษา
	ผศ.กฤดากร กล่อมการ
ระดับการศึกษา	ปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2548

ปริญญานิพนธ์ฉบับนี้ได้รับการอนุมัติเป็นส่วนหนึ่งของการศึกษาตามหลักสูตร วิศวกรรมศาสตรบัณฑิต คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเหล้าเจ้าคุณทหารลาดกระบัง

.....

(รศ.ดร.ปิติเขต สู้รักษา)

อาจารย์ผู้ควบคุมวิทยานิพนธ์

.....

(ผศ.กฤดากร กล่อมการ)

อาจารย์ผู้ควบคุมวิทยานิพนธ์

หัวข้อปริญญานิพนธ์	หุ่นยนต์เลื่อนที่แบบชาญฉลาด
นักศึกษา	นายปรนัย มุกด์ธนะอนันต์ รหัสประจำตัว 45010435
	นางสาวปิยณัฐ ทองสง รหัสประจาตัว 45010468
อาจารย์ผู้ควบคุมวิทยานิพนธ์	รศ.ดร.ปิติเขต สู้รักษา
	ผศ.กฤดากร กล่อมการ
ระดับการศึกษา	ปริญญาวิศวกรรมศาสตรบัณฑิต
ภาควิชา	วิศวกรรมสารสนเทศ
ปีการศึกษา	2548

บทคัดย่อ

โครงงานนี้เป็นการศึกษาเพื่อพัฒนา ระบบของหุ่นยนต์ให้มีความฉลาดและสามารถตอบสนองความต้องการของผู้ใช้ได้มากขึ้น โดยเน้นให้หุ่นยนต์มีลักษณะการเคลื่อนที่แบบอลวน (Chaotic) ที่ทำให้หุ่นยนตเคลื่อนที่ได้ดีในพื้นที่ที่ไม่รู้จักภูมิประเทศได้ในเวลาที่จำกัดซึ่งทำให้ไม่ต้องเสียเวลาและทรัพยากรในการเรียนรู้ลักษณะภูมิประเทศนั้นๆ และยังสามารถใช้เสียงเป็นตัวสั่งงานเป็นภาษาอังกฤษทางไมโครโฟน ผ่านคอมพิวเตอร์แล้วส่งผ่านโมดูลไร้สายไปยังตัวหุ่นยนต์เพื่อเปลี่ยนลักษณะการเคลื่อนที่แบบอลวน ซึ่งระบบนี้สามารถนำไปพัฒนาต่อสร้างเป็นหุ่นยนต์อเนกประสงค์ เช่น หุ่นยนต์ดูดฝุ่น หุ่นยนต์เฝ้าบ้าน เป็นต้น เพื่อเป็นการอำนวยความสะดวกในการใช้ชีวิตประจำวันของมนุษย์มากขึ้น

THESIS	INTELLIGENT AUTONOMOUS ROBOT	
STUDENT	MR. PARANAI MUKTANA-ANAN	NO.45010435
	MISS PIYANAT THONGSONG	NO.45010468
ADVISOR	Assoc.Prof..Dr.PITIKHATE SOORAKSA	
	Asst.Prof.KITDAKORN KLOMKARN	
COURSE	BACHELOR OF INFORMATION ENGINEERING	
DEPARTMENT	INFORMATION ENGINEERING	
ACADEMIC YEAR	2005	

Abstract

This project is developed for improve the abilities of robot like Intelligence and can response for costumer and This project focus Chaotic movement , This system can movement in unknown area in limited time and do not spent time too much and Can command from Voice Recognition . That System can be developed build Multi - Purpose Robot such Cleaner Robot , Security Robot for make comport for human life better. This project developed education for Intelligent Autonomous Robot and to powerful response to user by Voice control input by microphone and sent to Wireless module pass to robot. A Chaotic Robot that is system will continuous develop for built Security Robot or Vacuum Cleaner Robot. All of this create for routine everyday of human.

กิตติกรรมประกาศ

จากความสำเร็จของโครงงานหุ่นยนต์เคลื่อนที่อย่างชาญฉลาด คณะผู้จัดทำขอขอบพระคุณ รศ.ดร.ปิติเขต สู้รักษา และ ผศ.กฤดากร กล่อมการ ที่ได้ให้คำปรึกษาและให้ความสนับสนุนในทุกๆด้าน และขอขอบคุณพี่แมว พี่เอ๋ สำหรับคำแนะนำในการทำโครงงาน รวมถึงเพื่อนๆที่ให้ความช่วยเหลือในด้านต่างๆตลอดมา และสุดท้ายขอขอบพระคุณ คุณพ่อ และ คุณแม่ที่ได้ให้การสนับสนุนในด้านต่างๆมาโดยตลอดจนโครงงานสำเร็จได้ด้วยดีรวมถึงกำลังใจที่มีให้ตลอดมา

นายปรนัย มุกด์ธนะอนันต์
นางสาวปิยณัฐ ทองสง

สารบัญ

สารบัญรูป

สารบัญตาราง

บทที่1
บทนำ

1.1 ที่มาของโครงงาน

ในปัจจุบัน หุ่นยนต์ได้เข้ามีบทบาทกับมนุษย์มากยิ่งขึ้น ทั้งในการอำนวยความสะดวกในชีวิตประจำวัน การเข้าทำหน้าที่ในภารกิจอันตรายต่างๆ ซึ่งหุ่นยนต์สามารถช่วยลดภาระและความเสี่ยงมนุษย์ได้เป็นอย่างดี องค์ประกอบที่สำคัญที่ทำให้หุ่นยนต์สามารถอำนวยความสะดวกให้กับมนุษย์ได้ก็คือ สมองกล ซึ่งเป็นส่วนประกอบที่เกิดจากการเขียนโปรแกรมให้หุ่นยนต์มีระบบของวิธีการคิดที่มีประสิทธิภาพ หรือที่เรียกว่าปัญญาประดิษฐ์ (Artificial Intelligence : AI) จึงทำให้หุ่นยนต์สามารถทำตามคำสั่งและตอบสนองความต้องการของมนุษย์ได้อย่างมีประสิทธิภาพ การทำงานของหุ่นยนต์โดยทั่วไปจะรอคำสั่งและทำตามคำสั่งที่มีมาให้เท่านั้น แต่ในยุคหลังๆหุ่นยนต์ได้มีการเริ่มที่จะคิด หรือ คาดเดาเหตุการณ์ล่วงหน้า และสามารถตัดสินใจตามโปรแกรมที่เขียนไว้ได้เป็นอย่างดี การคิดแบบไม่เป็นระเบียบ (Chaotic Robot) เป็นแนวความคิดที่มีพื้นฐานมาจากการสุ่มคิดและสุ่มเลือกเส้นทางการโปรแกรมที่เหมาะสมในตัวของมันเอง อาทิ เช่น หุ่นยนต์ตัดหญ้าในสนามหญ้า หรือ หุ่นยนต์ดูดฝุ่นในบ้าน การทำงานของหุ่นยนต์ประเภทนี้นั้นจะอาศัยหลักการของ Chaotic เข้ามาช่วยในเรื่องของเส้นการเดิน ในการตัดหญ้าหรือ ดูดฝุ่นโดยที่ไม่ยึดติดกับบริเวณการสามารถตัดหญ้า หรือ ดูดฝุ่นได้หมดทั่วทุกพื้นที่ ซึ่งเป็นประโยชน์ที่เกิดจากหุ่นยนต์ข้างต้น และเทคโนโลยีอีกอย่างหนึ่งที่ทำให้หุ่นยนต์มีความฉลาดพอที่จะเป็นเพื่อนของมนุษย์ มีความสามารถในการที่ใช้งาน และ โต้ตอบกับผู้ใช้ได้โดยไม่ต้องกดปุ่ม ได้แก่เทคโนโลยีการควบคุมหุ่นยนต์ด้วยเสียงพูด(Voice Control) ซึ่งเป็นเทคโนโลยีที่สามารถสร้างรูปแบบการใช้งาน (User Interface) ที่จะอำนวยความสะดวกในการสั่งงาน ที่สามารถพัฒนาไปถึงการคุยโต้ตอบกับหุ่นยนต์ได้ในอนาคต

1.2 วัตถุประสงค์ของโครงงาน

1.ต่อยอดองค์ความรู้จากโครงงานเดิมที่มีมา วิเคราะห์ข้อดีข้อเสียต่างๆและนำมาประยุกต์ใช้ในโครงงาน

2.ประมวลคำสั่งที่ใช้ในการสั่งงานด้วยเสียงและพัฒนาให้มีความชัดเจนมากยิ่งขึ้น

3.พัฒนาหุ่นยนต์ที่สามารถใช้การเคลื่อนที่แบบอลวนซึ่งเป็นการเพิ่มพื้นที่การทำงานและประสิทธิภาพของหุ่นยนต์

4.ประยุกต์การใช้งานมาเป็นระบบไร้สาย ในการสั่งงานผ่านคอมพิวเตอร์และควบคุมจากระยะ ไกล เพื่อเกิดความใช้งานได้หลายวัตถุประสงค์

5.โครงงานสามารถเป็นแนวทางในการนำไปพัฒนาต่อเป็นหุ่นยนต์ที่สามารถอำนวยความสะดวกและใช้งานได้จริงในอนาคต

1.3 ขอบเขตของโครงงาน

1.ตัวหุ่นยนต์สามารถเคลื่อนที่แบบอลวนได้ 3 รูปแบบ

2.หุ่นยนต์สามารถทำงานตามคำสั่งได้

4.การทำงานที่ถูกต้องจะมีการทำงานที่ตรงกับคำสั่งของผู้ใช้และแสดงผลได้อย่างถูกต้อง

5.รูปแบบการทำงานจะต้องง่ายและสามารถเข้าใจง่าย

6.ฮาร์ดแวร์สามารถทำงานได้อย่างถูกต้องและตรงกับคำสั่งที่ส่งผ่านคอมพิวเตอร์

1.4 ขั้นตอนและวิธีการดำเนินงาน

1.ศึกษาและหาข้อมูลจากแหล่งต่างๆที่เกี่ยวข้อง

2.กำหนดขอบเขตงานให้กระชับและตรงตามจุดประสงค์มากที่สุด

3.ศึกษาซอฟท์แวร์SDK เพื่อให้สามารถใช้งานได้เต็มประสิทธิภาพมากที่สุด

4.ศึกษาการลักษณะใช้อุปกรณ์ต่างๆและโปรแกรมที่จำเป็นต้องใช้กับฮาร์ดแวร์

5.ออกแบบตัวหุ่นยนต์และวงจรควบคุม

6.นำฮาร์ดแวร์มาทดลองเพื่อวิเคราะห์หาข้อผิดพลาด

7.นำซอฟท์แวร์มาทดลองใช้งานเพื่อหาเปอร์เซ็นต์ความผิดพลาดของระบบ

8.แก้ไขและปรับปรุงระบบให้มีความผิดพลาดที่น้อยที่สุด

9.บันทึกและสรุปผลการทดลอง

1.5 เครื่องมือที่ใช้ในการทดลอง

1.5.1 ซอฟท์แวร์

1.คอมพิวเตอร์ Pentium 4 – 2.4 GHz Ram 512 MB

2.Microsoft Visual Basic 6.0 Service Pack6

3.Microsoft SDK Speech 5.1

4.RS232-Program Burn Editor

5.MicroISP Version 3.01

1.5.2 ฮาร์ดแวร์

1.หุ่นยนต์ 2 ล้อ

2.ไมโครคอนโทรลเลอร์MCS51 เบอร์ P89C81RD2

3.วงจรควบคุม

4.RF-module

5.สายสัญญาณ RF232

บทที่2
ทฤษฎีและเอกสารงานวิจัยที่เกี่ยวข้อง

2.1 ทฤษฎีการเคลื่อนที่แบบอลวน (Chaotic Mobile)

หุ่นยนต์เคลื่อนที่แบบอลวนเป็นหุ่นยนต์ที่มีรูปแบบเส้นทางการเคลื่อนที่แบบอลวนที่แสดงให้เห็นว่าเป็นเส้นทางการเดินของหุ่นยนต์ที่มีพื้นที่การกระจายมากกว่าการเดินแบบสุ่ม ซึ่งการเคลื่อนที่แบบนี้ได้ทำให้เกิดประโยชน์หลายอย่าง เช่น นำไปใช้ในการเดินเรือ พฤติกรรมการเคลื่อนที่ของหุ่นยนต์ในภูมิประเทศที่ไม่รู้จัก เช่น การทำความสะอาด , การตรวจตราหรือการลาดตระเวณ เป็นต้น

ความอลวนได้นำเสนอรูปแบบหนึ่งของพฤติกรรมของการเคลื่อนที่แบบสมบูรณ์ ซึ่งถือว่าเป็นลักษณะเด่นอย่างหนึ่ง เช่นความไวต่อสภาพเริ่มต้นของการเปลี่ยนแปลงเพียงเล็กน้อย , การไม่สามารถคาดเดาเหตุการณ์ได้เป็นระยะเวลานาน , ความแน่นอนและความไม่แน่นอนของแนววิถีกระสุนหรือจรวดในอากาศ ตัวอย่างของระบบอลวน ได้แก่ การไหลเชี่ยวของของเหลว , การผสมรวมกันของของเหลว , ปฏิกิริยาทางเคมี , ระบบชีววิทยา เช่น ในสมองมนุษย์ , เสียงจังหวะการเต้นของหัวใจ

ความแตกต่างของระบบอลวนสามารถแสดงให้เห็นได้ด้วยส่วนประกอบสำคัญ 3 อย่างด้วยกัน ได้แก่

1. สมการเวลา
2. ค่าของตัวแปรที่บอกลักษณะของระบบ
3. สภาพเริ่มต้น

ในโครงงานนี้มีระบบอลวนทั้งหมด 3 ระบบ ที่นำมาใช้ในการกำหนดเส้นทางการเคลื่อนที่ของหุ่นยนต์ที่อาจเกิดขึ้นได้ ซึ่งรูปแบบของระบบอลวนที่ทำให้เกิดวิถีเส้นทางการเคลื่อนที่ของหุ่นยนต์ ได้แก่

1.สมการของเชน (Chen)

2.สมการของลอเลนซ์ (Lorenz)

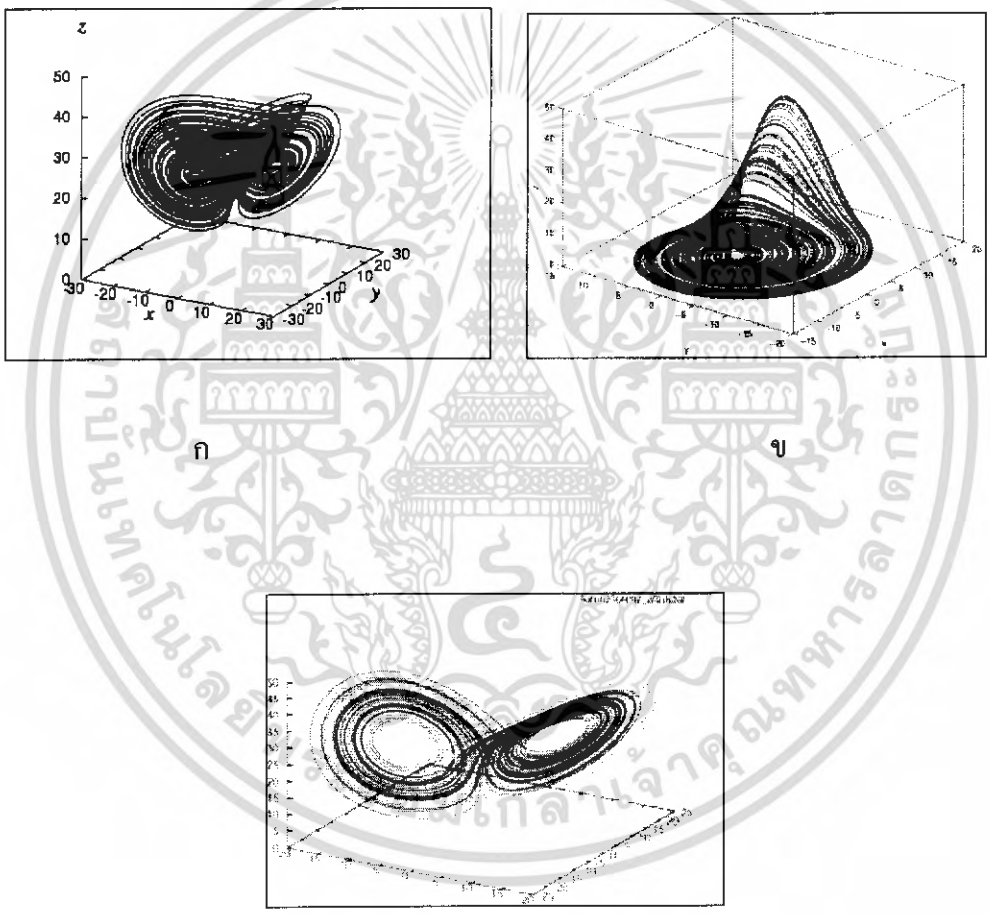
3. สมการของรอสเลอร์ (Rossler)

ซึ่งทั้งหมดเป็นระบบอลวนที่อยู่ในรูป 3 มิติ หลังจากนั้นจึงนำมาเปลี่ยนให้อยู่ในรูป 2 มิติ และทำเป็นหุ่นยนต์เพื่อแสดงเส้นทางเดินในภูมิประเทศจริงเพื่อนำมาเปรียบเทียบ

รูปแบบ 3 มิติของระบบอลวนสามารถอธิบายได้ด้วยรูปทั่วไปได้ดังนี้

$$x' = f(x,y,z)$$
$$y' = g(x,y,z) \qquad (1)$$
$$z' = h(x,y,z)$$

จากสมการข้างต้นสามารถแสดงเป็นโมเดลทางคณิตศาสตร์สำหรับระบบอลวนแต่ละระบบแบบ 3 มิติดังรูปที่ 2.1

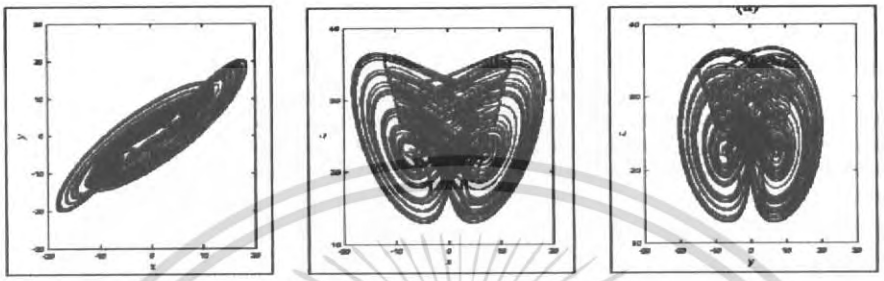


ก ข

ค

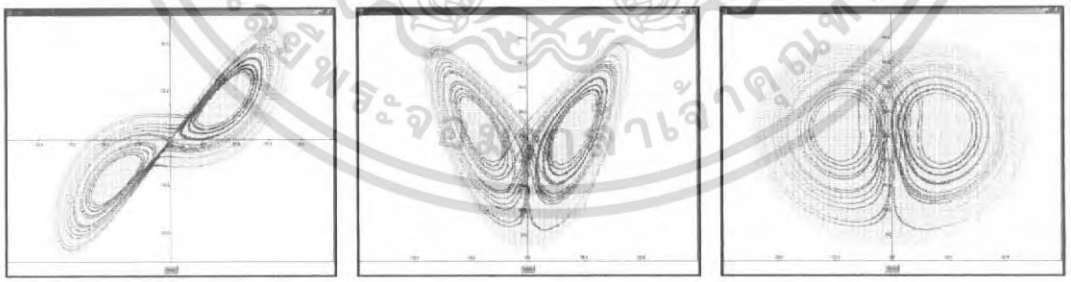
รูปที่ 2.1 แสดงโมเดลทางคณิตศาสตร์สำหรับระบบอลวนแบบ 3 มิติ ก. จากสมการของเชน , ข.จากสมการของรอสเลอร์ , ค.จากสมการของลอเรนซ์

และสามารถแปลงเป็น 2 มิติ ได้ดังรูปที่ 2.2



ก

ข



ค

รูปที่ 2.2 แสดงโมเดลทางคณิตศาสตร์แบบ 2 มิติ ก.จากสมการของเชน , ข.จากสมการของรอสเลอร์ , ค.จากสมการของลอเรนซ์

สำหรับเส้นทางการเคลื่อนที่แบบ 2 มิติสามารถอธิบายได้ด้วยสมการการเคลื่อนที่ของหุ่นยนต์ 2 ล้อ ได้ดังนี้

$$\begin{bmatrix} x' \\ y' \\ z' \end{bmatrix} = \begin{bmatrix} \cos\phi & 0 \\ \sin\phi & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} v \\ \varpi \end{bmatrix} \qquad (2)$$

โดย v เป็นความเร็วของหุ่นยนต์มีหน่วยเป็น เมตรต่อวินาที (m/s) และ

ϖ เป็นความเร็วเชิงมุมมีหน่วยเป็น เรเดียนต่อวินาที (rad/s) ซึ่งทั้งสองค่าเป็นอินพุตของระบบ

2.2 ทฤษฎีความอิสระ (Autonomy)

ความอิสระในที่นี้จะกล่าวถึงระบบที่สามารถทำงานในสภาพแวดล้อมจริง โดยปราศจากการควบคุมจากภายนอกเป็นช่วงเวลานาน ดังนั้นระบบของสิ่งมีชีวิตจึงเป็นต้นแบบของระบบที่เป็นอิสระในการควบคุมตัวเอง สามารถคงอยู่ในสภาพแวดล้อมจริงได้เป็นเวลานาน , สามารถรักษาสภาพโครงสร้างและกระบวนการทำงานภายในไว้ได้ , การใช้สภาพแวดล้อมหาที่ตั้งหรือข้อมูลเพื่อให้ระบบทำงานต่อไปได้ , และสามารถแสดงพฤติกรรมที่แตกต่างกันได้ เช่น การให้อาหาร , การเป็นผู้ช่วย เป็นต้น แต่ก็ยังมีข้อจำกัดในด้านความสามารถในการปรับตัวต่อสภาพแวดล้อมที่เปลี่ยนไป

ปัจจุบัน หุ่นยนต์ส่วนใหญ่ได้มีความอิสระทั้งหมด จึงทำให้ไม่สามารถคงอยู่หรือปฏิบัติภารกิจในสภาพแวดล้อมจริงได้เป็นเวลานานๆ ยกเว้นภายใต้เหตุการณ์ที่จัดไว้ แต่ก็ยังมีความหวังว่าในอนาคตอันใกล้หุ่นยนต์จะเพิ่มความสามารถ , ระดับของความอิสระจากการควบคุม และความฉลาด

2.2.1 ทฤษฎีความฉลาด (Intelligence)

หุ่นยนต์เป็นเครื่องจักรซึ่งมีความฉลาด , มีการคิดและการกระทำ คอมพิวเตอร์จะเป็นสมองให้กับหุ่นยนต์และจัดว่าเป็นส่วนประกอบที่สำคัญของระบบนี้ จึงมีความพยายามที่จะลดลงของขนาดและน้ำหนักของไมโครโพรเซสเซอร์แต่ให้มีหน่วยความจำและความเร็วเพิ่มขึ้น ซึ่งเป็นปัญหาหลักในการพัฒนาการเคลื่อนที่ของหุ่นยนต์ แต่เมื่อมีการใช้ชิปเดี่ยวที่มีความสามารถในการจ่ายพลังงานให้กับเมนเฟรมคอมพิวเตอร์ (mainframe computer) จึงทำให้ทุกวันนี้หุ่นยนต์ขนาดเล็กเต็มไปด้วยส่วนประกอบที่มีประสิทธิภาพระดับสูง ดังนั้นการกำหนดเกี่ยวกับความสามารถของ

กระบวนการคิดของหุ่นยนต์และความสามารถด้านการปฏิบัติภารกิจที่สมบูรณ์มากยิ่งขึ้น จึงต้องเป็นหน้าที่ของซอฟต์แวร์มากกว่าฮาร์ดแวร์

"ความฉลาด" ของหุ่นยนต์จะปรากฏในระบบดังนี้

1.กระบวนการการรับรู้ (Sensors Proceeding) ข้อมูลดิบที่ได้จากเซ็นเซอร์จะไม่สามารถนำมาใช้ประโยชน์ในการควบคุมพฤติกรรมได้ทันที ดังนั้น ความสามารถในการรับรู้ของหุ่นยนต์มักจะรวมซอฟต์แวร์ที่มีจุดประสงค์พิเศษในการหาตำแหน่งของพื้นที่ ในสภาพแวดล้อมที่แตกต่างกัน เช่น เพื่อตรวจสอบหาขอบเขต , เปรียบเทียบความแตกต่าง เป็นต้น

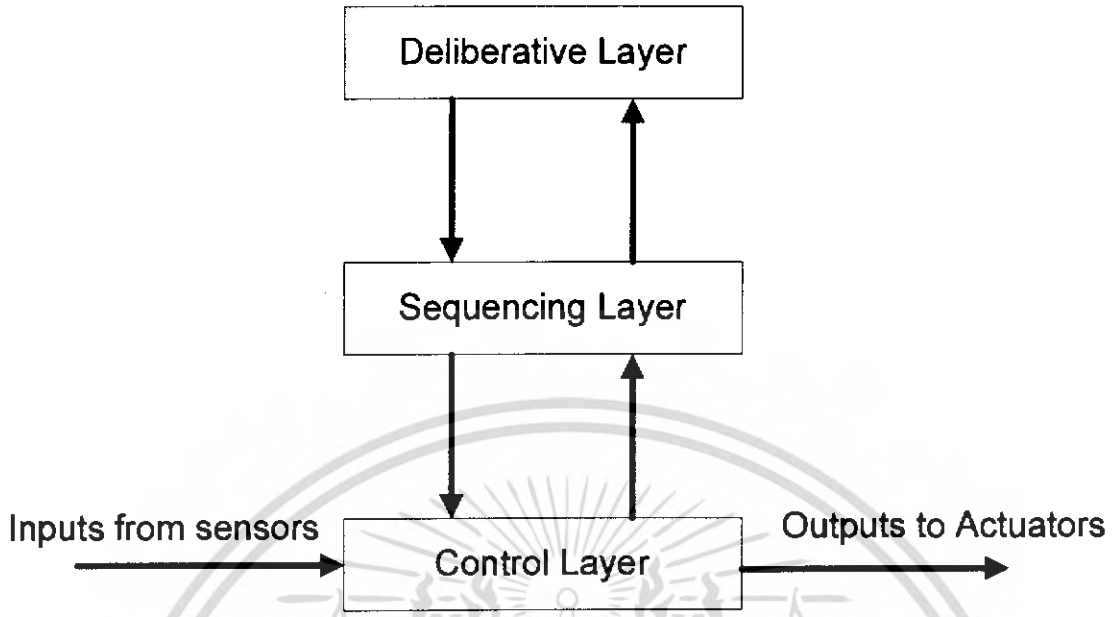
2.การกระทำโดยอัตโนมัติ (Reflex Behavior) ในระบบของหุ่นยนต์มีเส้นทางของปฏิกิริยาโต้ตอบอย่างรวดเร็วจากการรับรู้ จนกระทั่งมีการกระทำโดยอัตโนมัติ ซึ่งจะไม่เกี่ยวกับศูนย์กลางของระบบประสาทของหุ่นยนต์ ดังตัวอย่างการกระทำของมนุษย์ เช่น การดึงมือกลับเมื่อมีการสัมผัสกับของร้อน หรือการกระตุกของหัวเข่า สำหรับกรณีหลังจะเกิดเมื่อมีการเคาะเบาๆด้วยค้อนที่เส้นเอ็นให้กระดูกสะบ้าหัวเข่า อวัยวะการรับรู้ภายให้เส้นกล้ามเนื้อจะส่งผลการรับรู้ไปตามเส้นกล้ามเนื้อที่ต่อไปยังเส้นประสาทไขสันหลัง ผลลัพธ์ก็คือเกิดการหดตัวของกล้ามเนื้อทำให้เกิดการกระดุก หรือเมื่อมีการสะดุดก้อนหินในขณะที่กำลังเดิน ขาจะมีการถอยหลังกลับหรือพยายามที่จะหลีกเลี่ยงอุปสรรค

เมื่อรวมพฤติกรรมเหล่านี้ไว้มนกลไกการเดินของหุนยนต์ จึงปรากฏเป็นพฤติกรรมที่เราเรียกว่า " ความฉลาด " เกิดขึ้น

3.โปรแกรมสำหรับจุดประสงค์พิเศษ (Special-purpose Programming) เช่น การหาเส้นทาง , การหาตำแหน่งหรือการหลีกเลี่ยงอุปสรรค อาจจะถูกรวมกันไว้ในซอฟต์แวร์ของหุ่นยนต์

4.การทำงานของการะบวนการรับรู้ (Sensing Processing) การศึกษาในการสร้างความฉลาดให้กับหุ่นยนต์ได้ถูกพัฒนาอย่างต่อเนื่อง เพื่อใช้ในการทำงานของกระบวนการรับรู้ของหุ่นยนต์ ซึ่งรวมถึงการคิดอย่างมีเหตุผล , การเรียนรู้ และการวางแผน

เมื่อนำส่วนประกอบของซอฟต์แวร์ควบคุมหุ่นยนต์ที่กล่าวมาข้างต้นมารวมกัน จะเรียกว่าสถาปัตยกรรมด้านซอฟต์แวร์ของหุ่นยนต์ (Robot's software architecture) ซึ่งโดยทั่วไปแล้วจะมีการจัดโครงสร้างของระบบเป็นลำดับชั้น โดยส่วนประกอบที่แสดงปฏิกิริยาโต้ตอบจะอยู่ในชั้นล่างสุด และส่วนประกอบที่เกี่ยวกับการวางแผนและการเรียนรู้จะอยู่ในชั้นสูงสุด ดังตัวอย่างดังรูป



รูปที่ 2.3 แสดงสถาปัตยกรรมแบบ 3 ระดับของหุ่นยนต์เคลื่อนที่

2.3 ทฤษฎีไมโครคอนโทรลเลอร์

ไมโครคอนโทรลเลอร์ (Microcontroller) เป็นอุปกรณ์ไอซี (IC : Integrated Circuit) ที่สามารถโปรแกรมการทำงานได้หลายครั้ง รับข้อมูลในรูปสัญญาณดิจิตอลเข้าไปทำการประมวลผลแล้วส่งผลลัพธ์ข้อมูลดิจิตอลออกมาเพื่อนำไปใช้งานตามที่ต้องการได้

ไมโครคอนโทรลเลอร์หรืออาจจะเรียกอีกอย่างได้ว่าไมโครโพรเซสเซอร์ชิพเดี่ยว (Single-Chip Microprocessor) ซึ่งเป็นไมโครโพรเซสเซอร์ชนิดหนึ่ง เช่นเดียวกับหน่วยประมวลผลกลาง (CPU : Central Processing Unit) ที่ใช้ในคอมพิวเตอร์แต่ได้รับการพัฒนาแยกออกมาในภายหลังเพื่อนำไปใช้ในวงจรด้านงานควบคุม คือ แทนที่ในการใช้งานจะต้องต่อวงจรภายนอกต่างๆเพิ่มเติมเช่นเดียวกับไมโครโพรเซสเซอร์ ก็จะทำการรวมวงจรที่จำเป็น เช่น หน่วยความจำ ส่วนอินพุต/เอาท์พุตบางส่วนเข้าไปในตัวไอซีเดียวกันและเพิ่มวงจรบางอย่างเข้าไปด้วยเพื่อให้มีความสามารถเหมาะกับการใช้งานควบคุม เช่น วงจรตั้งเวลา วงจรการสื่อสารอนุกรม เป็นต้น ดังนั้นไมโครคอนโทรลเลอร์สามารถจะทำงานเสมือนกับเป็นคอมพิวเตอร์เล็กๆเครื่องหนึ่งกล่าวโดยสรุปคือ

Microcontroller = Microprocessor + Memory + I/O

ปัจจุบันไมโครคอนโทรลเลอร์ถูกนำไปใช้อย่างกว้างขวาง โดยมักจะเป็นการนำไปประยุกต์ใช้ในระบบร่วมกับอุปกรณ์อื่นๆ (Embeded Systems) เพื่อใช้ควบคุมการทำงานบางอย่าง

เช่นใช้ในรถยนต์ , เครื่องปรับอากาศ , เครื่องซักผ้าอัตโนมัติ เป็นต้นเพราะว่าไมโครคอนโทรลเลอร์มีข้อดีเหมาะสมต่อการใช้งานควบคุมหลายประการเช่น

1. ไอซีและระบบที่ได้มีขนาดเล็ก
2. ระบบที่ได้มีราคาถูกกว่าการใช้ชิพไมโครโพรเซสเซอร์
3. วงจรที่ได้จะมีความซับซ้อนน้อย ช่วยลดข้อผิดพลาดที่อาจจะเกิดขึ้นได้ในการต่อวงจร
4. มีคุณสมบัติเพิ่มเติมสำหรับงานควบคุมโดยเฉพาะซึ่งใช้งานได้ง่าย
5. ช่วยลดระยะเวลาในการพัฒนาระบบได้

ไมโครคอนโทรลเลอร์มีหลายยี่ห้อ หลายตระกูล และหลายเบอร์ด้วยกัน ซึ่งแต่ละเบอร์ก็จะมีโครงสร้างภายในและความสามารถในการทำงานที่แตกต่างกัน ทำให้เลือกใช้งานได้อย่างเหมาะสม

2.3.1 คุณสมบัติของไมโครคอนโทรลเลอร์ ตระกูล MCS-51

การเรียนรู้เพื่อพัฒนาไมโครคอนโทรลเลอร์MCS-51 ที่เหมาะสมควรจะมีขนาดของหน่วยความจำโปรแกรมใหญ่พอสมควร ซึ่งในที่นี้ใช้ MCS-51 เบอร์ P89C81RD2 ของ Phillips เนื่องจากมีหน่วยความจำโปรแกรมแบบแฟลชมากถึง 64 กิโลไบต์ มีพอร์ตให้ใช้งานถึง 4 พอร์ต พร้อมไทเมอร์ 3 ตัว มีโมดูล PCA สำหรับสร้าง PWM จำนวน 5 ช่อง และยังมีหน่วยความจำแรมพิเศษอีก 1 กิโลไบต์ และมีความสามารถเด่นคือ สามารถโปรแกรมหน่วยความจำผ่านพอร์ตอนุกรมในแบบ ISP (In-system Programming) ทำให้สามารถพัฒนาหรือแก้ไขโปรแกรมได้สะดวกขึ้น

2.3.1.1 คุณสมบัติทางเทคนิคของ P89C51RD2

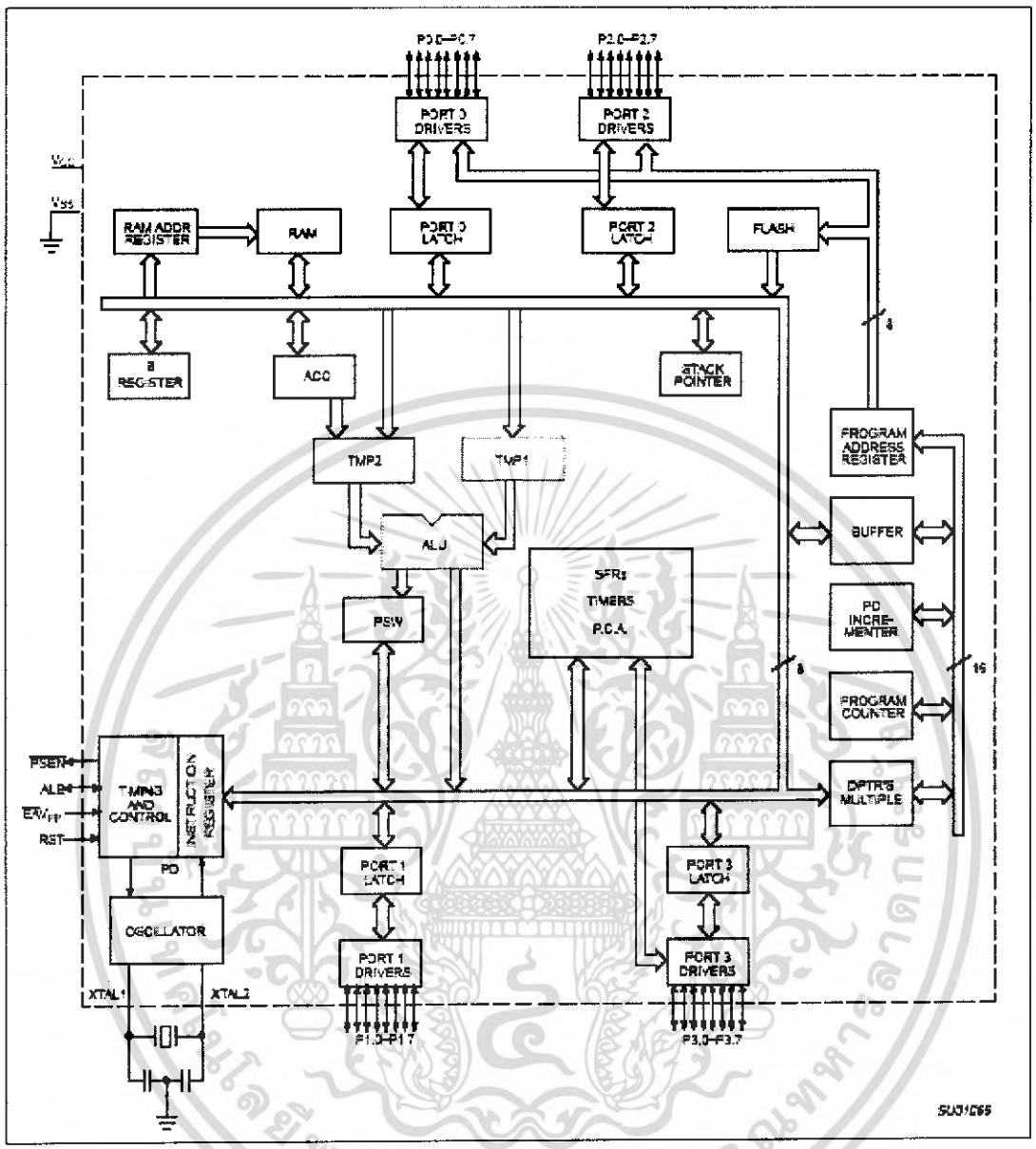
เป็นไมโครคอนโทรลเลอร์ 8 บิต ที่เข้ากันได้กับไมโครคอนโทรลเลอร์แบบพื้นฐานหน่วยความจำโปรแกรมภายในเป็นแบบแฟลช ทำให้สามารถลบและเขียนใหม่ได้นับหมื่นครั้งจึงสามารถใช้งานในรูปแบบไมโครคอนโทรลเลอร์ชิปเดี่ยว (single ship) โดยไม่ต้องใช้หน่วยความจำภายนอก ทำให้ใช้งานพอร์ตทั้งหมดได้อย่างครบถ้วน และขนาดของหน่วยความจำของ P89C51RD2 นี้มีขนาด 64 กิโลไบท์

- หน่วยความจำข้อมูลภายใน 1 กิโลไบต์
- สามารถโปรแกรมข้อมูลลงในหน่วยความจำโปรแกรมในแบบวงจรหรือในระบบ (ISP : In-system programming)
- ความถี่สัญญาณนาฬิกาสูงสุด 33 เมกะเฮิร์ซ ในกรณีทำงานด้วยสัญญาณนาฬิกาภายใน 12 ลูกต่อแมชชีนไซเกิลและ 20 เมกะเฮิร์ซ ในกรณีทำงานด้วยสัญญาณนาฬิกาภายใน 6 ลูก

ต่อแมชชีนไซเกิลP89C51RD2BN ได้รับการกำหนดให้ทำงานเบื้องต้นในโหมดสัญญาณนาฬิกา 12 ลูกค่อแมชชีนไซเกิล สามารถเลือกเปลี่ยนเป็น 6 สัญญาณนาฬิกาต่อแมชชีนไซเกิลได้ โดยสามารถเปลี่ยนกลับไปกลับมาได้ แตกต่างจาก P89C51RD2BN ที่ค่าเบื้องต้นเป็น 6 สัญญาณนาฬิกาต่อแมชชีนไซเกิลเมื่อเปลี่ยนเป็น 12 สัญญาณนาฬิกาแล้วจะเปลี่ยนกลับมาอีกไม่ได้

- ขาพอร์ต 8 บิต 4 พอร์ต แบบกึ่งสองทิศทาง (quasi-bidirectional) เป็นได้ทั้งอินพุตและเอาท์พุต
- อุปกรณ์เพอริเฟอรัลภายในไมโครคอนโทรลเลอร์สามารถทำงานด้วยความเร็ว 12 สัญญาณนาฬิกาต่อแมชชีนไซเกิลได้ แม้ว่าซีพียูจะทำงานด้วยความเร็ว 6 สัญญาณนาฬิกาต่อแมชชีนไซเกิล เป็นคุณสมบัติที่เพิ่มเติมเข้ามาในเบอร์ P89C51RD2BN
- มีวงจรสื่อสารอนุกรมแบบฟูลดูเพล็กซ์
- ไทเมอร์/เคาน์เตอร์ขนาด 16 บิต 3 ตัว (ไทเมอร์ 0,1 และ 2)
- มีรีจิสเตอร์ตัวชี้ตำแหน่งข้อมูลหรือ DPTR 2 ตัว
- สามารถรองรับแหล่งกำเนิดอินเตอร์รัปต์ได้ 7 ประเภท
- กำหนดนัยสำคัญของการตอบสนองอินเตอร์รัปต์ได้ 4 ระดับ
- สามารถติดต่อหน่วยความจำภายนอกได้สูงสุด 64 กิโลไบต์
- มีวอตช์ด็อกไทเมอร์
- มีโมดูลวงจรนับโปรแกรมได้ (PCA : Programmable Counter Array) ซึ่งบรรจุวงจรตรวจจับสัญญาณ (capture) , เปรียบเทียบสัญญาณ (compare) , วงจรมอดูเลชั่นทางความกว้างพัลส์(PWM) และวอตช์ด็อกไทเมอร์ (watchdog timer)

ในรูปที่ 2.4 เป็นโครงสร้างพื้นฐานของไมโครคอนโทรลเลอร์ MCS-51 เบอร์ P89C81RD2 จะเห็นได้ว่าจะมีโครงสร้างเหมือนกับไมโครคอนโทรลเลอร์ MCS-51 พื้นฐาน หากแต่มีข้อแตกต่างที่มีหน่วยความจำโปรแกรมภายในเป็นแบบแฟลชเพิ่มเข้ามา จึงทำให้สามารถลบและเขียนใหม่ได้



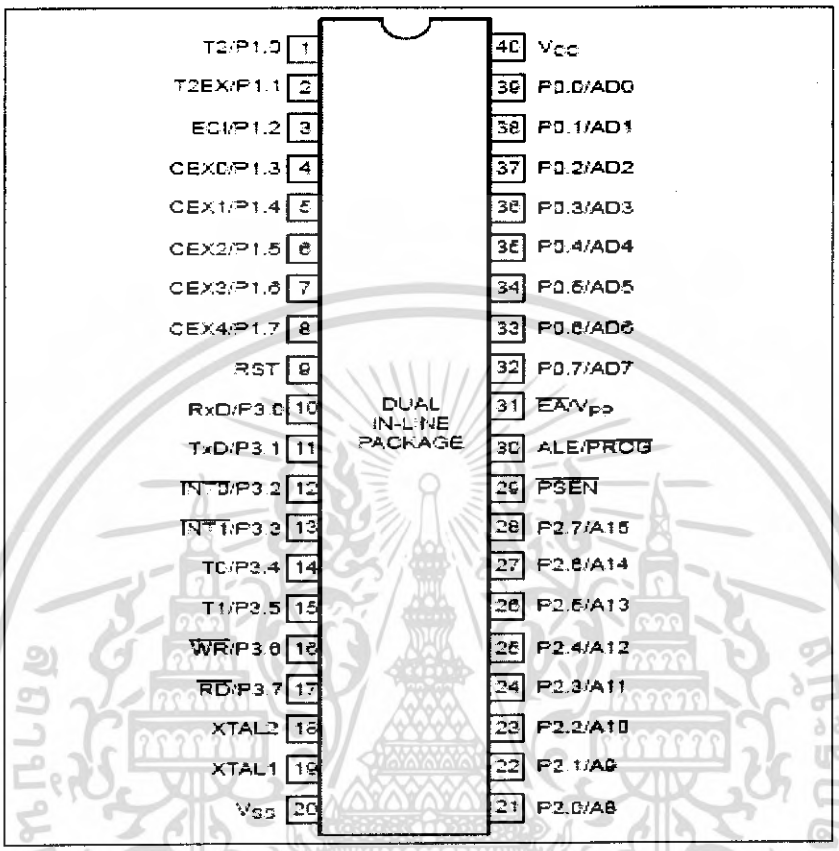
รูปที่ 2.4 รายละเอียดโครงสร้างหลักของไมโครคอนโทรลเลอร์ MCS-51 เบอร์ P89C81RD2 ของ Phillips

2.3.2 ลักษณะการจัดขาภายนอกของไมโครคอนโทรลเลอร์ MCS-51

ลักษณะการจัดขาภายนอกของไมโครคอนโทรลเลอร์ MCS-51 จะมีการแบ่งกลุ่มการจัดขาออกเป็น 4 กลุ่มด้วยกัน คือ

- กลุ่มขาแหล่งจ่ายไฟเลี้ยง และสัญญาณนาฬิกา
- กลุ่มขาสำหรับการอ้างแอดเดรสและรับส่งข้อมูล
- กลุ่มขาที่ใช้ในการควบคุม

- กลุ่มขาพอร์ตใช้งานแบบขนานและอนุกรม



รูปที่2.5 การจัดขาของไมโครคอนโทรลเลอร์ MCS-51

2.3.3 ขาที่สำคัญของไมโครคอนโทรลเลอร์ MCS-51

1.ขา Vcc เป็นขารับแรงดันไฟกระแสตรง +5 VDC

2.ขา GND เป็นขากราวด์

3.พอร์ต 0 (P0.0-P0.7) เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป

4.พอร์ต 1 (P1.0-P1.7) เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง บางขาสำหรับใช้งานพิเศษ

5.พอร์ต 2 (P2.0-P2.7) เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง สำหรับใช้งานทั่วไป

6.พอร์ต 3 (P3.0-3.7) เป็นพอร์ตอินพุตเอาต์พุตแบบ 2 ทิศทาง บางขาสำหรับใช้งานพิเศษ

7.ขารีเซต (RST) ใช้สำหรับการรีเซตการทำงานของไมโครคอนโทรลเลอร์ โดยการรีเซตจะต้องคงสภาวะ high อย่างน้อยนาน 2 แมชชีนไซเกิลในขณะที่ออสซิลเลเตอร์กำลังทำงานอยู่

8.ขา ALE/PROG เป็นขาสัญญาณเพื่อทำหน้าที่ควบคุมการแลตซ์ (Latch) ค่าตำแหน่งแอดเดรสไบต์ต่ำ (Address Latch Enable)

9.ขา PSEN (Progtam Store Enable) ทำหน้าที่เป็นสัญญาณสโตปเพื่ออ่านคำสั่งจากหน่วยความจำภายนอก เมื่อไมโครคอนโทรลเลอร์ประมวลผลคำสั่ง จากหน่วยความจำภายนอก ขานี้จะส่งสัญญาณสโตปจำนวน 2 ครั้งในแต่ละ Machine cycle แต่ในขณะที่ติดต่อกับหน่วยความจำข้อมูลภายนอกจะไม่มีการส่งสัญญาณสโตปแต่อย่างใด

10.ขา EA/Vcc (External Access Enable / Vcc) เป็นขาสำหรับการเลือกใช้หน่วยความจำโปรแกรมจากภายในหรือภายนอก โดยมีสถานะเป็น 0 และ 1 และขานี้ยังทำหน้าที่รับแรงดันไฟสำหรับโปรแกรม (Vcc) ขนาด 12 โวลต์ เพื่อใช้ในระหว่างการโปรแกรมหน่วยความจำโปรแกรม (EPROM)

11.ขา XTAL1 และขา XTAL2 เป็นขาใช้งานของวงจรอินเวอร์ติ้งออสซิลเลอร์เตอร์แอมปลิไฟเออร์ (Imverting Oscillator Amplifier) สำหรับใช้ต่อร่วมกับคริสตอลภายนอก

2.4 การอินเตอร์รัปต์ (Interrupt)

การอินเตอร์รัปต์เป็นชื่อเรียกกระบวนการที่เข้ามาขัดจังหวะการทำงานโดยปกติของไมโครคอนโทรลเลอร์

2.4.1 การจัดการอินเตอร์รัปต์ในไมโครคอนโทรลเลอร์ MCS-51

เมื่อมีการอินเตอร์รัปต์ในไมโครคอนโทรลเลอร์MCS-51 เกิดขึ้นและมีการอีนาเบิล (enable) การตอบสนองการอินเตอร์รัปต์ไว้ กระบวนการหลังจากนั้นซีพียูจะทำการกระโดดไปยังแอดเดรสในหน่วยความจำที่กำหนดไว้เรียกตำแหน่งแอดเดรสนี้ว่า แอดเดรสอินเตอร์รัปต์เวกเตอร์ (interropt vector address) ดังนั้นจะต้องมีการเขียนโปรแกรมย่อยการบริการอินเตอร์รัปต์ไว้ที่แอดเดรสอินเตอร์รัปต์เวกเตอร์นี้ โดยค่าของแอดเดรสอินเตอร์รัปต์เวกเตอร์จะแตกต่างกันไปในการอินเตอร์รัปต์แบบต่างๆ ดังมีรายละเอียดต่อไปนี้

การอินเตอร์รัปต์ภายนอกที่ขา INT0 มีค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 0003H
การอินเตอร์รัปต์ไทเมอร์ 0 มีค่าแอแดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 000BH
การอินเตอร์รัปต์ภายนอกที่ขา INT1 มีค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 0013H
การอินเตอร์รัปต์จากไทเมอร์ 1 มีค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 001BH
การอินเตอร์รัปต์จากพอร์ตอนุกรม มีค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 0023H

การอินเตอร์รัปต์จากไทเมอร์ 2 มีค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์อยู่ที่ 002BH

2.4.2 การเขียนโปรแกรมย่อยอินเตอร์รัปต์

มีหลักการโดยทั่วไปดังนี้

1.ต้องเริ่มต้นด้วยแอดเดรสอินเตอร์รัปต์เวกเตอร์เสมอ เพื่อให้การตรวจสอบการทำงานทำได้ง่ายและแยกส่วนของโปรแกรมย่อยนี้ออกจากโปรแกรมหลักหรือโปรแกรมย่อยอื่นๆอย่างชัดเจน ด้วยคำสั่ง ORGxxxxH (ค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์)

2.เมื่อเข้าสู่โปรแกรมย่อย ควรเก็บค่าของรีจิสเตอร์หรือแฟลกที่ใช้แสดงสถานะต่างๆที่ต้องมีการใช้งานในโปรแกรมย่อยบริการอินเตอร์รัปต์ไว้ในสแต็กเสียก่อน เพื่อป้องกันความผิดพลาดที่อาจเกิดขึ้นต่อการทำงานทั้งของโปรแกรมบริการอินเตอร์รัปต์นี้และโปรแกรมหลักด้วยคำสั่ง PUSH

3.เมื่อเขียนโปรแกรมบริการอินเตอร์รัปต์เรียบร้อยแล้ว ให้ทำการคืนค่าของรีจิสเตอร์ที่นำมาใช้ในโปรแกรมบริการอินเตอร์รัปต์ด้วยคำสั่ง POP ยกเว้นรีติสเตอร์ที่ต้องการนำผลการกระทำในโปรแกรมบริการอินเตอร์รัปต์นี้ไปใช้งาน ซึ่งในทางปฏิบัติจริง ไม่พบมากนักและไม่แนะนำให้เขียนโปรแกรมในลักษณะนี้

4.ปิดท้ายโปรแกรมย่อยบริการอินเตอร์รัปต์ด้วยคำสั่ง RETI เสมอ

2.4.3 รีจิสเตอร์ที่เกี่ยวข้องกับการอินเตอร์รัปต์ในไมโครคอนโทรลเลอร์ MCS-51

1.รีจิสเตอร์ IE (Interrupt Enable) ใช้ในการกำหนดว่าจะยอมให้มีการอินเตอร์รัปต์จากแหล่งใดได้บ้าง โดยมีรายละเอียดแต่ละบิตเป็นดังนี้

บิต 7							บิต 0
EA	X	ET2	ES	ET1	EX1	ET0	EX0

EA ถ้าเป็นลอจิก “1” หมายความว่า ให้อินเตอร์รัปต์ได้

ET2 ถ้าเป็นลอจิก “1” จะอีนาเบิลไทเมอร์ 2 (ใช้กับเบอร์ที่มีไทเมอร์ 2)

ES ถ้าเป็นลอจิก “1” จะอีนาเบิลอินเตอร์รัปต์จากพอร์ตอนุกรม

ET1 ถ้าเป็นลอจิก “1” จะอีนาเบิลไทเมอร์ 1

EX1 ถ้าเป็นลอจิก “1” จะอีนาเบิลสัญญาณอินเตอร์รัปต์จากภายนอกที่เขามาทางขา INT1

ET0 ถ้าเป็นลอจิก “1” จะอีนาเบิลไทเมอร์ 0

EX0 ถ้าเป็นลอจิก “1”จะอีนาเบิลสัญญาณอินเตอร์รัปต์จากภายนอกที่เขามาทางขา INT0

2.รีจิสเตอร์ IP (Interrupt Priority) ใช้กำหนดลำดับการอินเตอร์รัปต์ กรณีที่เกิดการอินเตอร์รัปต์จากหลายแหล่งพร้อมๆกัน

3.รีจิสเตอร์ TCON (Timer control) รีจิสเตอร์ตัวนี้นอกจากจะใช้ควบคุมไทเมอร์แล้วยังใช้ในการอินเตอร์รัปต์อีกด้วย โดยมีรายละเอียดดังนี้

บิต 7 บิต 0

TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0

TF1 เป็นบิตโอเวอร์โฟลว์ของไทเมอร์ 1 จะเป็นลอจิก “1”เมื่อไทเมอร์เกิดโอเวอร์โฟลว์ และบิตนี้สามารถอินเตอร์รัปต์ไมโครคอนโทรลเลอร์ได้ เมื่อทำโปรแกรมตอบสนองการอินเตอร์รัปต์จบบิต TF1 นี้จะกลับมาเป็นลอจิก “0”

TR1 ใช้ปิดเปิดไทเมอร์ 1

TF0 เหมือนกับ TF1 แต่ใช้กับไทเมอร์ 0

TR0 ใช้ปิดเปิดไทเมอร์ 0

IE1 เป็นบิตแสดงการอินเตอร์รัปต์ทางฮาร์ดแวร์ที่เข้ามาทาง INT1

IT1 ใช้ในการเลือกรูปแบบสัญญาณอินเตอร์รัปต์จากภายนอกที่เข้ามาทางขา INT1 ที่ต้องการตอบสนอง ถ้าเป็นลอจิก “0” หมายความว่าจะเกิดการอินเตอร์รัปต์เมื่อมีสัญญาณขอบขาลงเข้ามา ถ้าเป็นลอจิก “1” หมายความว่าตจะเกิดการอินเตอร์รัปต์เมื่อมีระดับลอจิกต่ำ

IE0 ใช้งานเหมือนกับ IE1 แต่จะใช้กับ INT0

IT0 ใช้งานเหมือนกับ IT1 แต่จะใช้กับ INT0

2.4.4 แหล่งกำเนิดสัญญาณอินเตอร์รัปต์จากไมโครคอนโทรลเลอร์ MCS-51

2.4.4.1 สัญญาณอินเตอร์รัปต์จากภายนอก

เป็นการตรวจสอบสัญญาณที่เข้ามายังขา INT0 และ INT1 หากตรงตามเงื่อนไขที่กำหนดก็จะทำให้เกิดการอินเตอร์รัปต์ขึ้น โดยการอีนาเบิลการอินเตอร์รัปต์แบบนี้สามารถกระทำได้โดยการกำหนดค่าในรีจิสเตอร์ IE ที่บิต EX0 สำหรับสัญญาณอินเตอร์รัปต์ที่ขา INT0 และบิต EX1 สำหรับสัญญาณอินเตอร์รัปต์ที่ขา INT1 และทำการเลือกเงื่อนไขของการตรวจสอบสัญญาณในรีจิสเตอร์ TCON ที่บิต IE0 สำหรับสัญญาณอินเตอร์รัปต์ที่ขา INT0 และบิต IE1 สำหรับสัญญาณอินเตอร์รัปต์ที่ขา INT1

เงื่อนไขการตรวจสอบสัญญาณอินเตอร์รัปต์ที่ขา INT0 และ INT1 มีด้วยกัน 2 ลักษณะ คือ

1.ตรวจสอบระดับลอจิก ถ้าบิต IEx ในรีจิสเตอร์ TCON เป็น “0” จะเกิดการอินเตอร์จากภายนอกที่ขา INT0 หรือ INT1 ได้เมื่อตรวจพบระดับลอจิกต่ำ หรือ “0” เมื่อเกิดการอินเตอร์รัปต์แล้วให้ดำเนินการทำให้สัญญาณที่ขานี้กลับสู่ระดับลอจิก “1” ก่อนที่การบริการอินเตอร์รัปต์เสร็จสิ้นเพื่อป้องกันการเกิดอินเตอร์รัปต์ซ้อน

2.ตรวจสอบขอบขาของสัญญาณ ถ้าหากบิต IEx ในรีจิสเตอร์ TCON เป็น “1” จะเกิดการอินเตอร์รัปต์จากภายนอกที่ขา INT0 หรือ INT1 ได้ต่อเมื่อตรวจพบการเปลี่ยนแปลงของสัญญาณที่ขา INT0 หรือ INT1 จาก “1” เป็น “0” หรือตรวจสอบพบขอบขาลงของสัญญาณที่ป้อนมายังขา INT0 หรือ INT1 และต้องมีการรักษาสถานะลอจิก “0” นี้เป็นเวลาอย่างน้อย 1 แมชชีนไซเคิล จึงถือว่าเกิดการอินเตอร์รัปต์อย่างสมบูรณ์

เมื่อเกิดการอินเตอร์รัปต์ขึ้น ซีพียูภายในไมโครคอนโทรลเลอร์จะกระโดดไปยังแอดเดรส 0003H สำหรับการอินเตอร์รัปต์ที่ขา INT0 และ 0013H สำหรับการอินเตอร์รัปต์ที่ขา INT1

2.4.4.2 การอินเตอร์รัปต์จากไทเมอร์/เคาน์เตอร์ 0 และ 1

แหล่งกำเนิดอินเตอร์รัปต์นี้จัดเป็นแหล่งกำเนิดอินเตอร์รัปต์ภายในแบบหนึ่ง โดยใช้การเกิดโอเวอร์โฟลว์ จากการนับค่าในไทเมอร์/เคาน์เตอร์ภายในไมโครคอนโทรลเลอร์ MCS-51 เมื่อไทเมอร์ 0 เกิดการโอเวอร์โฟลว์ ก็จะทำการเซตบิต TF0 ในรีจิสเตอร์ TCON และถ้าไทเมอร์ 1 เกิดการโอเวอร์โฟลว์ บิต TF1 ในรีจิสเตอร์ TCON จะได้รับการเซตค่าเช่นเดียวกัน

ค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์ของการอินเตอร์รัปต์แบบนี้อยู่ที่ 000BH สำหรับไทเมอร์ 0 และ001BH สำหรับไทเมอร์1

อย่างไรก็ตามการอินเตอร์รัปต์แบบนี้จะเกิดขึ้นหรือมีการตอบสนองก็ต่อเมื่อมีการอีนาเบิลการอินเตอร์รัปต์ โดยการเซตบิต EA , ET0 และ ET1 ในรีจิสเตอร์ IE

2.4.4.3 การอินเตอร์รัปต์จากพอร์ตอนุกรม

แหล่งกำเนิดอินเตอร์รัปต์นี้จัดเป็นแหล่งกำเนิดอินเตอร์รัปต์ภายในแบบหนึ่ง เมื่อวงจรพอร์ตอนุกรมส่งหรือรับข้อมูลเสร็จสมบูรณ์ก็จะกำเนิดสัญญาณอินเตอร์รัปต์ขึ้น โดยการเซตบิต RI ในกรณีรับข้อมูล และบิต TI ในกรณีส่งข้อมูล RI และ TI อยู่ในรีจิสเตอร์ SCON

ค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์ของการอินเตอร์รัปต์แบบนี้อยู่ที่ 0023H การอินเตอร์รัปต์แบบนี้สามารถแทนได้ด้วยการออร์กันของบิต RI และ TI

2.4.4.4 การอินเตอร์รัปต์จากไทเมอร์/เคาน์เตอร์ 2

แหล่งกำเนิดอินเตอร์รัปต์นี้จัดเป็นแหล่งกำเนิดอินเตอร์รัปต์ภายในแบบหนึ่ง โดยใช้การเกิดโอเวอร์โฟลว์ จากการนับค่าในไทเมอร์/เคาน์เตอร์ 2 หรือจากการแคปเจอร์หรือการตรวจจับสัญญาณที่ขา T2EX เมื่อเกิดการโอเวอร์โฟลว์ก็จะทำการเซตบิต TF2 ในรีจิสเตอร์ TCON และถ้าไทเมอร์ 2 สามารถตรวจจับการเปลี่ยนแปลงจากระดับ "1" เป็น "0" ที่ขา T2EX ได้ และบิต EXEN2 ในรีจิสเตอร์ T2CON ได้รับการเซตไว้ ก็จะทำให้บิต EXF2 ในรีจิสเตอร์ T2CON เซต เป็นการแจ้งว่าเกิดการอินเตอร์รัปต์เนื่องจากการแคปเจอร์ที่ไทเมอร์ 2 ค่าแอดเดรสอินเตอร์รัปต์เวกเตอร์ของการอินเตอร์รัปต์แบบนี้อยู่ที่ 002BH การอินเตอร์รัปต์แบบนี้สามารถแทนได้ด้วยการออร์กันของบิต TF2 และ EXF2

อย่างไรก็ตามการอินเตอร์รัปต์แบบนี้จะเกิดขึ้นหรือมีการตอบสนองก็ต่อเมื่อมีการอีนาเบิลการอินเตอร์รัปต์ โดยการเซตบิต EA ในรีจิสเตอร์ IE จะตอบสนองในกรณีเกิดโอเวอร์โฟลว์เมื่อบิต EXEN2 ในรีจิสเตอร์ T2CON เป็น "0" และตอบสนองในกรณีเกิดแคปเจอร์เมื่อบิต EXEN2 ในรีจิสเตอร์ T2CON เป็น "1"

2.5 อวัยวะที่ใช้ในการเปล่งเสียง (Articulation)

โดยทั่วไปแล้วมนุษย์เราเปล่งเสียงออกมาตามสำเนียงในระบบภาษาของตนเอง แม้ว่าคนที่อยู่ในสังคมเดียวกัน ใช้ภาษาเดียวกัน แต่เสียงของแต่ละคนที่เปล่งออกมานั้นจะมีลักษณะแตกต่างกัน ในบางครั้งแม้แต่คนๆเดียวกันเปล่งเสียงคำเดียวกันสองครั้งสัญญาณที่ได้ก็ยังมีความแตกต่างกัน แต่คนเราก็ยังสามารถแยกแยะความหมายของคำที่เกิดจากการพูดของแต่ละคนได้ การจดจำและแยกแยะความหมายของคำพูดจึงเป็นเรื่องง่ายสำหรับมนุษย์ แต่สำหรับคอมพิวเตอร์นั้น การที่จะให้จดจำหรือแยกแยะเสียงพูดนั้นเป็นเรื่องที่ยากมาก ขั้นตอนแรกในการที่จะทำให้คอมพิวเตอร์สามารถจำเสียงพูดนั้นได้ ต้องมีการศึกษาถึงลักษณะของเสียงนั้นเสียก่อน โดยเริ่มจากอวัยวะที่ใช้ในการเปล่งเสียง

อวัยวะที่ใช้ในการเปล่งเสียงแบ่งเป็น 3 พวกใหญ่ๆ คือ

1.อวัยวะที่ใช้ในการสร้างลม คือส่วนที่ทำให้เกิดการเคลื่อนไหวของลม

2.อวัยวะที่เคลื่อนที่ได้ (Active Articulator) หมายถึงอวัยวะที่ติดกับกระดูกคางส่วนล่าง ได้แก่ ริมฝี ปากและลิ้น

3.อวัยวะที่เคลื่อนที่ไม่ได้ (Passive Articulator) หมายถึงอวัยวะที่ติดกับกระดูกคางส่วนบนได้แก่ ริมฝีปากบน ฟันบน ปุ่มเหงือก เพดานแข้ง เพดานอ่อน ลิ้นไก่ และอวัยวะที่เป็นช่องว่าง ได้แก่ ช่องคอ ปาก ช่องจมูก

2.6 ลำดับการเกิดเสียง

ลำดับการเกิดเสียงนั้นเราแบ่งออกได้เป็น 3 ขั้นตอนใหญ่ๆ คือ

1.จุดเริ่มต้น (Initiation) อวัยวะที่ใช้ในขั้นตอนนี้คือปอด ที่ขั้นตอนนี้ลมจะถูกขับออกจากปอด

2.การดัดแปลงลมที่เส้นเสียง (Phonation) อวัยวะที่ใช้ในขั้นตอนนี้ก็คือ อวัยวะที่อยู่ต่อจากปอดขึ้นไปจนถึงกล่องเสียงเป็นขั้นตอนที่ลมจากปอดจะผ่านมาเข้าหลอดลมและกล่องเสียงซึ่งณ ที่กล่องเสียงนี้เส้นเสียงจะทำหน้าที่เป็นลิ้นเปิดและปิดทำให้เกิดเสียง 2 ชนิด คือ ถ้าเส้นเสียงเปิดตลอดเวลาที่ลมผ่าน ลมจะผ่านออกมาได้อย่างสะดวก ซึ่งจะทำให้เกิดเสียงไม่ก้อง แต่ถ้าเส้นเสียงปิดกั้นลมไว้ลมที่ผ่านออกมาจะเพิ่มแรงดันมากขึ้นจนเส้นเสียงเปิดและปิดสลับกันไปทำให้เกิดเสียงชนิดก้องและเรียกความถี่ในการเปิด-ปิดของเส้นเสียงว่า “ความถี่พื้นฐาน”

2.7 Voice Command Control

2.7.1 Properties

Voice Command control มีคุณสมบัติดังต่อไปนี้

AutoGainEnable

AwakeState

CountCommands

Device

Enabled

EnableMenu

hWnd

Initialized

LastError

MenuCreate

Microphone

Speaker

SRMode

SuppressException

Threshold

Properties ที่ใช้การเขียนโปรแกรม ได้แก่ CountCommands , Enabled , Initialized , MenuCreate

- CountCommand

รูปแบบคำสั่ง

CountCommand(Menu As Long) As Long

คำอธิบาย

เป็นคำสั่งที่ใช้ในการรีเทิร์นค่าของคำสั่งบนเมนู ซึ่งเป็นคำสั่งที่สามารถอ่านได้อย่างเดียว ไม่สามารถทำอย่างอื่นได้

- Enabled

รูปแบบคำสั่ง

Enabled As Long

คำอธิบาย

TRUE ถ้าสามารถทำการรู้จำเสียงหรือ FALSE ถ้าไม่สามารถทำได้ ซึ่งต้องทำการเซ็ตเป็น TRUE เพื่อที่จะสามารถฟังเสียงจากไมโครโฟนได้

เมื่อการรู้จำเสียงไม่สามารถทำได้ตัวรู้จำก็จะไม่สามารถทำการรู้จำคำสั่งใดๆ จากเมนูต่างๆ ได้ แอพลิเคชั่นหนึ่งๆ จะใช้คุณสมบัติ ENABLED ซึ่งจะอนุญาตใช้ ผู้ใช้ทำการปิดความสามารถในการรู้จำเสียงได้ เป็นการหยุดการรู้จำเสียงเป็นการชั่วคราว

ผู้ใช้สามารถใช้บัญชีรายชื่อ คำสั่งควบคุมในการเปิด-ปิดระบบเสียง เมื่อใดก็ตามที่เสียงถูกเปิดหรือปิด ข้อมูลข่าวสาร WM_SPEECHSTARTED หรือ WM_SPEECHENDED ทั้งหมดจะส่งไปที่ระบบวินโดว์ระดับบน แอพลิเคชั่นจะสามารถใช้ข้อมูลข่าวสารนี้ในการตัดสินใจเมื่อจะเปิดหรือปิดตัว VOICE COMMANDS หรือ VOICE TEXT

การเซ็ต ENABLED ให้เป็น FALSE นั้นจะอนุญาตใช้ ผู้ใช้ปิดการจำเสียง ดังนั้นจะไม่เกิดการรู้จำ สำหรับตัวอย่าง ผู้ใช้อาจจะต้องการปิดการรู้จำเสียงจากไมโครโฟนในระหว่างการประชุม ดังนั้นการรู้จำเสียงจะหยุดรอ หรือใครบางคนจะเผลอพูดคำสั่งให้ปิดการรู้จำในเมนู

ถ้าแอพลิเคชัน VOICE-NAVIGATION ถูกติดตั้งบนเครื่องคอมพิวเตอร์ของผู้ใช้ แอพลิเคชันจะไม่ต้องการเซ็ตสภาวะ ENABLED อย่างไรก็ตาม มันจะต้องเรียกใช้ฟังก์ชั่นเพื่อเริ่มที่จะทำการรู้จำเสียง สำหรับตัวอย่าง ถ้าแอพลิเคชันได้รับคำสั่ง " Do you want to print the document ?" มันจะการรู้จำเสียงและทำการตอบรับเสียงจากผู้ใช้

- Initialized

รูปแบบคำสั่ง

Initialized As Long

คำอธิบาย

จะเท่ากับ 1 ถ้าตัวคอนโทรลทำการเริ่มต้น และจะเป็น 0 ถ้าไม่ ผู้ใช้สามารถควบคุมตัวคอนโทรลให้มันเริ่มต้นโดยการเซ็ตให้เป็น 1 (ตัวจัดการเสียงจะสามารถโหลดได้ช้า ดังนั้นผู้ใช้อาจจะต้องการปรับคอนโทรลมากกว่าเมื่อทำการเริ่มต้น)

ส่วนมากการใช้ METHODS และ PROPERTIES จะทำการเริ่มต้นคอนโทรลโดยอัตโนมัติถ้ามันไม่ได้ถูกผู้ใช้เซ็ตให้เริ่มต้นอยู่ก่อนแล้ว

- MenuCreate

รูปแบบคำสั่ง

MenuCreata (Application As String , State As String , flags As Long) As Long

คำอธิบาย

จะทำการสร้างเมนูเสียงขึ้นมาใหม่ หรือตัวเมนูเสียงอาจจะมีอยู่แล้ว สำหรับแอพลิเคชันสามารถอ่านได้อย่างเดียว

พารามิเตอร์	คำบรรยาย
Application	เป็นชื่อของแอพลิเคชัน เช่น " Excel "
State	การกำหนดชื่อ เช่น " Main Memu " หรือ " File Open dialog box "
flags	Flags เป็นตัวบอกสถานะในการสร้างเมนูว่าสร้างลักษณะใด พารามิเตอร์นี้ สามารถเป็นหนึ่งในค่าต่างของตารางแสดงค่าข้างล่าง

ตารางที่ 2.1 แสดงความหมายของค่าพารามิเตอร์ต่างๆ ของ MenuCreate

ค่าของ flags	คำบรรยาย
VCMDMC_CREATE_ALWAYS	ทำการสร้างเมนูว่างๆขึ้นมาและตั้งชื่อ ถ้ามีเมนูอยู่แล้วในฐานข้อมูลของเมนูเสียง มันจะทำการลบออกก่อนแล้วทำการเก็บเมนูใหม่ลงไปในฐานข้อมูล เมื่อตัวเมนูเก่าถูกปล่อยออกจาก

	หน่วยความ
VCMDMC_CREATE_NEW	ทำการสร้างเมนูว่างๆขึ้นมาและตั้งชื่อ ถ้ามีเมนูอยู่แล้วในฐานข้อมูลของเมนูเสียง ฟังก์ชันจะทำการรีเทิร์นค่า Error เมนูที่สร้างขึ้นใหม่จะถูกเก็บลงในฐานข้อมูล เมื่อเมนูเก่าถูกปล่อยออกจากหน่วยความจำ
VCMDMC_CREATE_TEMP	ทำการสร้างเมนูว่างๆขึ้นมาและตั้งชื่อ ถ้ามีเมนูอยู่แล้วในฐานข้อมูลเสียง ฟังก์ชันจะทำการรีเทิร์นค่า Error เมนูใหม่นี้จะเป็นเมนูชั่วคราว และจะถูกยกเลิกเมื่อตัวเมนูถูกปล่อยออกจากหน่วยความจำ
VCMDMC_OPEN_ALWAYS	ทำการเปิดเมนูที่มีอยู่แล้วและตั้งชื่อ ถ้าไม่มีเมนูอยู่ฟังก์ชัน จะทำการสร้างเมนูว่างขึ้นมาใหม่ เมนูใหม่นี้จะถูกเก็บลงในฐานข้อมูล เมื่อตัวเมนูถูกปล่อยออกจากหน่วยความจำ
VCMDMC_OPEN_EXISTING	ทำการเปิดเมนูที่มีอยู่แล้ว ถ้าไม่มีเมนูอยู่ฟังก์ชันจะทำการรีเทิร์นค่า Error

ตารางที่ 2.2 แสดงค่าของ flags ของคำสั่ง MenuCreate

เมื่อแอพลิเคชันทำการสร้างเมนูเสียงขึ้นมาโดยใช้ Methods MenuCreate ซึ่งค่าของ VCMDMC_CREATE_NEW, VCMDMC_CREATE_ALWAYS , VCMDMC_OPEN_ALWAYS ตัวคำสั่งเสียงจะทำการเก็บเมนูใหม่ลงในฐานข้อมูล เมื่อแอพลิเคชันทำการเริ่มต้นใหม่มันจะสามารถรักษาเวลาในการโหลดเมนูจากฐานข้อมูลแทนที่จะสร้างเมนูขึ้นมาใหม่

แอพลิเคชันสามารถสร้างตัวเมนูสียง โดยการโหลดเมนูเสียงที่มีอยู่แล้วจากฐานข้อมูลของเมนูเสียงหรือทำการสร้างเมนูเสียงขึ้นมาใหม่ เมนูเสียงตัวหนึ่งจะไม่ต้องถูกเก็บลงในฐานข้อมูลแอพลิเคชันสามารถเมนูเสียงเป็นการชั่วคราวโดยการเซ็ต dwFlags กับ VCMDMC_CREATE_TEMP เมนูเสียงชั่วคราวนี้จะอยู่ไปเรื่อยๆ จนกว่าจะถูกปล่อยออกจากหน่วยความจำ

แอพลิเคชันสามารถสร้างตัวเมนูเสียงได้มากกว่าหนึ่ง โดยทำเหมือนเดิมสำหรับตัวอย่างหนึ่งแอพลิเคชันต้องทำการระบุเมนูของแอพลิเคชันอื่นๆ

2.7.2 Methods

Voice Command control มี Method ดังต่อไปนี้

Activate

AddCommand

CmdMimic

Deactivate

EnableItem

GeneralDlg

GetCommand

LexiconDlg

ListGet

ListSet

MenuDelete

ReleaseMenu

Remove

SetCommand

TrainGeneralDlg

TrainMenuDlg

TrainMicDlg

Methods ที่ใช้การเขียนโปรแกรม ได้แก่ Activate , AddCommand , Deactivate , GetCommand , ReleaseMenu

- Activate

รูปแบบคำสั่ง

Activate (Menu As Long)

คำอธิบาย

เมนูเสียงจะอยู่ในสถานะพร้อมที่จะรับคำสั่ง ดังนั้นมันจะสามารถรู้จักคำสั่งที่ป้อนเข้ามาได้ เมนูเสียงถูกสร้างเพียงครั้งเดียวและมีหลายๆ คำสั่ง มันจะต้องพร้อมก่อนที่จะทำการรู้จำคำสั่ง เมื่อเมนูเสียงพร้อม ตัวเช็คเมนูจะเช็คการแมทช์ของคำสั่งที่รับเข้ามา เมื่อไรก็ตามที่ตัวประมวลผลออกเสียงจากคำสั่งเสียงจะไม่สามารถเช็คเมนูได้เมื่อมันไม่พร้อม

พารามิเตอร์	คำบรรยาย
Menu	คำสั่งเสียง

ตารางที่2.3 แสดงความหมายของค่าพารามิเตอร์ต่างๆ ของ Activate

- AddCommand

รูปแบบคำสั่ง

AddCommand (Menu As Long , command As String , description As String , category As String , flags As Long , action As String)

คำอธิบาย

เป็นการเพิ่มคำสั่งกับเมนูที่ถูกสร้างโดย MenuCreate คำสั่งจะถูกเพิ่มเข้าไปต่อท้ายคำสั่งที่มีอยู่แล้วในเมนู

คำสั่งจะเป็นหมายเลขโดยเรียงลำดับจาก 1 ถึง n คำสั่งใหม่จะถูกเพิ่มเข้าไปในส่วนท้ายของเมนู ดังนั้น คำสั่งแรกที่ถูกเพิ่มเข้าไปจะมีหมายเลขเป็น 1+n คำสั่งเสียงจะมีการสนับสนุนการแสดงผลแบบลำดับรายการ

สำหรับผลที่ดีที่สุดนั้นจะต้องทำการให้เมนูเสียงอยู่สภาวะไม่พร้อมที่ปฏิบัติการก่อนที่เรียกใช้ AddCommand อีกอย่างหนึ่งก็คือ เมนูจะต้อง deactivate , recompiled , และ reactivated ก่อนที่จะ AddCommand ถ้าเมนูไม่พร้อมที่จะปฏิบัติการอยู่แล้วเมื่อ AddCommand ถูกเรียกตัวเมนูจะไม่ recompiled จนกว่าแอพลิเคชันจะอยู่ในสภาวะพร้อมที่จะปฏิบัติการอีกครั้ง

ถ้าคำสั่งข้อความประกอบด้วยลำดับรายชื่อ สามารถเรียกใช้ ListSet ทำการเซ็ตคำพูดที่ผู้ใช้สามารถใช้แทนรายชื่อเมื่อกำลังพูดคำสั่ง

พารามิเตอร์	คำบรรยาย
Menu	เมนูเสียง
Id	คำสั่งที่เหมือนกัน แอพลิเคชันสามารถระบุคำสั่งที่เหมือนกันแต่ละคำสั่งเสียงในเมนู คำสั่งที่เหมือนกันแอพลิเคชันจะกำหนดค่าพิเศษของคำสั่งภายในเมนู พารามิเตอร์นี้สามารถแก้ไขคำสั่งที่เหมือนกัน หรือมันสามารถถูกใช้สำหรับเหตุการณ์ เมื่อแอพลิเคชันรับเหตุการณ์แอพลิเคชันจะสามารถพิจารณาคำสั่งที่เหมือนกันในการตัดสินใจคำสั่งที่เป็นส่วนสำคัญของเหตุการณ์
Command	คำสั่งเสียงที่เป็นข้อความ ทุกๆคำสั่งเสียงจะต้องมีคำสั่งข้อความ เช่น " Open a

	file. " ในระหว่างกระบวนการเรียนรู้จำเสียงจะทำการแปลสัญญาณออดิโอที่รับเข้ามาไปเป็นข้อความและเปรียบเทียบมันกับคำสั่งข้อความในเมนูเสียงที่พร้อมที่จะทำงานภายในคำสั่งข้อความ จะมีสัญลักษณ์ที่มีความหมายพิเศษดังต่อไปนี้ < > แสดงรายชื่อของคำหรือถ้อยคำโดยสามารถพูดเป็นคำสั้นๆ สำหรับตัวอย่าง คำสั่งข้อความ " Send mail to < name > " บรรจุรายชื่อที่ถูกเรียก " name " ซึ่งการเพิ่มคำพูดในรายการจะใช้ Method Listset { } ถูกสำรองไว้ใช้ในอนาคต [ ] ถูกสำรองไว้ใช้ในอนาคต ตัวรู้จำคำสั่งจะสนับสนุนสัญลักษณ์ที่ใช้แทนคำสั่งข้อความ การใช้สัญลักษณ์จะอนุญาตให้ผู้ใช้พูดคำบางคำในระหว่างคำสำคัญในถ้อยคำนั้น สำหรับตัวอย่าง ถ้าแอพลิเคชันนิยามคำสั่งเสียงของ " {wild-card} mail {wild-card} name " , ผู้ใช้สามารถพูด , " Send mail to Fred " , " I want to mail Fred " หรือ " Mail , yes , I want it to go to Fred. "
Description	เป็นข้อความที่บอกถึงการกระทำของแอพลิเคชันที่ตอบสนองต่อคำสั่ง คำสั่งนี้จะบอกผู้ใช้ในจุดประสงค์ของคำสั่ง แอพลิเคชันจะเป็นตัวแสดงความหมายของคำสั่งข้อความ
category	ข้อความที่บอกึงลำดับชั้นซึ่งคำสั่งนี้จะเป็นส่วนหนึ่งของคำสั่งในเมนูเสียง จะทำการรวบรวมความแตกต่างของลำดับชั้น จะช่วยให้ผู้ใช้เลือกดลอกทั้งรายการของคำสั่งได้อย่างง่ายดาย ซึ่งมันจะมีแนวความคิดคล้ายกับเมนูวินโดว์ซึ่งจะรวมเอาคำสั่งต่างๆภายใต้ชื่อเมนู เช่น " File " , " Edit " , " View " และอื่นๆ สำหรับผลที่ดีที่สุดจะต้องใช้การแบ่งลำดับชั้น 20 หรือน้อยกว่านั้น
flags	flags จะแสดงข้อมูลข่าวสารเกี่ยวกับคำสั่ง
action	เป็นบล็อกของข้อมูลที่ถูกส่งไปให้แอพลิเคชันเมื่อพูดคำสั่ง ข้อมูลที่พร้อมจะให้แอพลิเคชันตีความหมาย ซึ่งข้อมูลจะรวมไปกับคำสั่งในเมนูเสียง ตัวจัดการจะส่งข้อมูลไปที่แอพลิเอชันเมื่อคำสั่งถูกการรู้จำเกิดขึ้น ข้อมูลที่พร้อมนี้จะพิเศษดังนั้นแอพลิเคชันต้องรู้ว่าจะทำอย่างไรกับคำสั่งเมื่อมีการสั่ง ข้อมูลจะถูกส่งผ่านไปคำสั่งโดยคอนโทรลจะไม่สามารถแปลความหมาย มันขึ้นอยู่กับการตัดสินใจของแอพลิเคชัน

ตารางที่ 2.4 แสดงความหมายของค่าพารามิเตอร์ต่างๆของ AddCommand

ค่าของ flags	คำบรรยาย
VCMDCMD_DISABLED_PERM	คำสั่งจะถูก DISABLED โดยจะใช้ตัวควบคุมของคำสั่งเสียง PROPERTEIS EnableMenu ทำให้คำสั่งเสียงไม่รู้จักมัน ดังนั้นคำสั่งจะไม่รวบรวมในเมนูคำสั่ง
VCMDCMD_DISABLED_TEMP	คำสั่งจะถูก DISABLED โดยจะใช้ตัวควบคุมของคำสั่งเสียง METHOD Setcommand คำสั่งจะถูกรวบรวมในเมนูเสียง อย่างไรก็ตามจะสามารถ ENABLED อีกครั้ง โดยปราศจากการรวบรวมของเมนู
VCMDCMD_VEIRFY	แอพลิเคชันจะต้องพร้อม โดยผู้ใช้จะทำหารพิสูจน์คำสั่งก่อนเอามันออก
VCMDCM_CANTRENAME	บอกถึงคำสั่งอัตโนมัติทั่วไปโดยไม่อนุญาตให้ผู้ใช้เปลี่ยนชื่อคำสั่ง

ตารางที่ 2.5 แสดงค่าของ flags ของคำสั่ง AddCommand

- Deactivate

รูปแบบคำสั่ง

Deactivate (Menu As Long)

คำอธิบาย

บอกให้ตัวรู้จำทำการหยุดฟังคำสั่งและคืนทรัพยากรไมโครโฟน ซาวด์การ์ด ถ้าผู้ใช้ยังออกคำสั่งและตัวรู้จำไม่ทำตามจะเกิด Error ขึ้น

พารามิเตอร์	คำบรรยาย
Menu	เมนูคำสั่ง

ตารางที่ 2.6 แสดงความหมายของค่าพารามิเตอร์ต่างๆของ Deactivate

- GetCommand

รูปแบบคำสั่ง

Getcommand (Menu As Long , index As Long , command Ax String , description As String , category As String , flags As Long ,action As String)

คำอธิบาย

ข้อมูลข่าวสารเกี่ยวกับคำสั่งที่ได้รับกลับมา

พารามิเตอร์	คำบรรยาย
Menu	ถ้ามีค่าเป็น TRUE จะเป็นการอ่านคำสั่งจากคำสั่งโกลบอลมันจะถูกใช้ในทุกๆ แอพลิเคชัน ถ้ามีค่าเป็น FALSE มันจะเข้าถึงข้อมูลคำสั่งสำหรับแอพลิเคชัน โดยทั่วไปแอพลิเคชันจะไม่ทำการเปลี่ยนแปลงแก้ไขคำสั่งโกลบอล
index	หมายเลขของคำสั่งที่ได้รับมา หมายเลขของคำสั่งจะเป็นลำดับจาก 1 ถึง n
Command	คำสั่งเสียงที่เป็นข้อความ ทุกๆคำสั่งเสียงจะต้องมีคำสั่งข้อความ เช่น " Open a file. " ในระหว่างกระบวนการเรียนรู้จำเสียงจะทำการแปลสัญญาณออดิโอที่รับเข้ามาไปเป็นข้อความและเปรียบเทียบมันกับคำสั่งข้อความในเมนูเสียงที่พร้อมที่จะทำงานภายในคำสั่งข้อความ จะมีสัญลักษณ์ที่มีความหมายพิเศษดังต่อไปนี้ < > แสดงรายชื่อของคำหรือถ้อยคำโดยสามารถพูดเป็นคำสั้นๆ สำหรับตัวอย่าง คำสั่งข้อความ " Send mail to < name > " บรรจุรายชื่อที่ถูกเรียก " name " ซึ่งการเพิ่มคำพูดในรายการจะใช้ Method Listset { } ถูกสำรองไว้ใช้ในอนาคต [ ] ถูกสำรองไว้ใช้ในอนาคต ตัวรู้จำคำสั่งจะสนับสนุนสัญลักษณ์ที่ใช้แทนคำสั่งข้อความ การใช้สัญลักษณ์จะอนุญาตให้ผู้ใช้พูดคำบางคำในระหว่างคำสำคัญในถ้อยคำนั้น สำหรับตัวอย่าง ถ้าแอพลิเคชันนิยามคำสั่งเสียงของ " {wild-card} mail {wild-card} name " , ผู้ใช้สามารถพูด , " Send mail to Fred " , " I want to mail Fred " หรือ " Mail , yes , I want it to go to Fred. "
Description	เป็นข้อความที่บอกถึงการกระทำของแอพลิเคชันที่ตอบสนองต่อคำสั่ง คำสั่งนี้จะบอกผู้ใช้ในจุดประสงค์ของคำสั่ง แอพลิเคชันจะเป็นตัวแสดงความหมายของคำสั่งข้อความ
category	ข้อความที่บอกึงลำดับชั้นซึ่งคำสั่งนี้จะเป็นส่วนหนึ่งของคำสั่งในเมนูเสียง จะ

	ทำการรวบรวมความแตกต่างของลำดับชั้น จะช่วยให้ผู้ใช้เลือกตลอกทั้งรายการของคำสั่งได้อย่างง่ายดาย ซึ่งมันจะมีแนวความคิดคล้ายกับเมนูวินโดว์ซึ่งจะรวมเอาคำสั่งต่างๆภายใต้ชื่อเมนู เช่น " File " , " Edit " , " View " และอื่นๆ สำหรับผลที่ดีที่สุดจะต้องใช้การแบ่งลำดับชั้น 20 หรือน้อยกว่านั้น
flags	flags จะแสดงข้อมูลข่าวสารเกี่ยวกับคำสั่ง
action	เป็นบล็อกของข้อมูลที่ถูกส่งไปให้แอพลิเคชันเมื่อพูดคำสั่ง ข้อมูลที่พร้อมจะให้แอพลิเคชันตีความหมาย ซึ่งข้อมูลจะรวมไปกับคำสั่งในเมนูเสียง ตัวจัดการจะส่งข้อมูลไปที่แอพลิเอชันเมื่อคำสั่งถูกการรู้จำเกิดขึ้น ข้อมูลที่พร้อมนี้จะพิเศษดังนั้นแอพลิเคชันต้องรู้ว่าจะทำอย่างไรกับคำสั่งเมื่อมีการสั่ง ข้อมูลจะถูกส่งผ่านไปคำสั่งโดยคอนโทรลจะไม่สามารถแปลความหมาย มันขึ้นอยู่กับการตัดสินใจของแอพลิเคชัน

ตารางที่2.7 แสดงความหมายของค่าพารามิเตอร์ต่างๆของ GetCommand

ค่าของ flags	คำบรรยาย
VCMDCMD_DISABLED_PERM	คำสั่งจะถูก DISABLED โดยจะใช้ตัวควบคุมของคำสั่งเสียง PROPERTEIS EnableMenu ทำให้คำสั่งเสียงไม่รู้จักมัน ดังนั้นคำสั่งจะไม่รวบรวมในเมนูคำสั่ง
VCMDCMD_DISABLED_TEMP	คำสั่งจะถูก DISABLED โดยจะใช้ตัวควบคุมของคำสั่งเสียง METHOD Setcommand คำสั่งจะถูกรวบรวมในเมนูเสียง อย่างไรก็ตามจะสามารถ ENABLED อีกครั้ง โดยปราศจากการรวบรวมของเมนู
VCMDCMD_VEIRFY	แอพลิเคชันจะต้องพร้อม โดยผู้ใช้จะทำหารพิสูจน์คำสั่งก่อนเอามันออก
VCMDCM_CANTRENAME	บอกถึงคำสั่งอัตโนมัติทั่วไปโดยไม่อนุญาตให้ผู้ใช้เปลี่ยนชื่อคำสั่ง

ตารางที่ 2.8 แสดงค่าของ flags ของคำสั่ง GetCommand

- ReleaseMenu

รูปแบบคำสั่ง

ReleaseMenu (Menu As long)

คำอธิบาย

ปล่อยเมนูเสียงออกจากหน่วยความจำ ซึ่งสามารถเรียกคำสั่ง ReleaseMenu สำหรับทุกๆ เมนูคำสั่งที่สร้างขึ้นได้

พารามิเตอร์	คำบรรยาย
Menu	เมนูคำสั่ง

ตารางที่ 2.9 แสดงความหมายของค่าพารามิเตอร์ต่างๆของ ReleaseMenu

2.7.3 Events

Voice Command control มี events ดังต่อไปนี้

AttribChanged

ClickIn

CommandOther

CommandRecognize

CommandStart

Interference

MenuActivate

UtteranceBegin

UtteranceEnd

VUMeter

Event ที่ใช้ในการเขียนโปรแกรมได้แก่ CommandRecognize

- CommandRecognize

รูปแบบคำสั่ง

CommandRecognize (ID As Long , CmdName As String , Flags As Long , Action As String , NumList As Long , ListValue As String , command As String)

คำอธิบาย

เหตุการณ์นี้จะเกิดขึ้นเมื่อคำสั่งที่พูดเข้ามาถูกรู้จำ พร้อมด้วยเหตุการณ์แอพลิเคชันจะรับข้อความของคำพูดและข้อมูล ซึ่งจะถูกป้อนให้แอพลิเคชัน

พารามิเตอร์	คำบรรยาย
ID	เช่นเดียวกับคำสั่งที่ถูกรู้จำ
CmdName	ชื่อของเมนูซึ่งบรรจุชื่อของคำวั่งที่ถูกรู้จำ
Flags	VCMDCMD_VERIFY ถ้าแอพลิเคชันต้องการพิสูจน์จาหผู้ใช้หรือไม่มีผลถ้าการพิสูจน์นั้นถูกปฏิเสธความต้องการในการพิสูจน์นั้น แอพลิเคชันจะแสดงบ็อกซ์ไดอะแกรมทขึ้นมา แอพลิเคชันจะทำการพิสูจน์สำหรับทำลายหรือกลับไม่ได้ของคำสั่ง เช่น " Format disk "
Action	ข้อความที่ถูกบรรจุข้อมูลมากับคำสั่งที่ถูกรู้จำ
NumList	หมายเลขของรายการที่ถูกรู้จำ ถ้าคำสั่งไม่ได้บรรจุรายการอะไรมาเลยพารามิเตอร์นี้จะเป็นศูนย์
ListValue	รายการของหนึ่งข้อความหรือมากกว่าที่ถูกแยกสำหรับตัวอย่าง ถ้าคำสั่งเป็น " Set the time to number AM or PM " พารามิเตอร์จะเป็น " Ten | PM "
command	คำสั่งข้อความสำหรับคำสั่งที่ถูกรู้จำ

ตารางที่ 2.10 แสดงความหมายของค่าพารามิเตอร์ต่างๆของ CommandRecognize

ผู้ใช้จะไม่ใช้ข้อความภายในของคำสั่งที่เหมือนกับคำสั่งที่ถูกรู้จำ แทนที่จะใช้ข้อมูลใน Action หรือข้อมูลที่เหมือนกันใน ID ในการตัดสินใจกับคำสั่งที่ถูกรู้จำ คำสั่งข้อความจะไม่บรรจุข้อความที่มีลักษณะเฉพาะ เพราะมีความเป็นไปได้สำหรับผู้ใช้ที่จะแก้ไขข้อความสำหรับคำสั่งสำหรับแอพลิเคชันที่ใช้ไมโครซอฟท์วอยซ์หรืออื่นๆ

ถ้ามีสองหรือมากกว่าเมนูคำสั่งที่บรรจุคำพูดและตัวรู้จำคำพูด ตัวจัดการจะทำการเรียก CommandRecognize สำหรับหนึ่งเมนูหรือ CommandOrder สำหรับอื่นๆตัวจัดการจะทำการตัดสินใจ ซึ่งเหตุการณ์จะเรียกเมนูอื่นๆ แอพลิเคชันจะไม่สมารถตัดสินใจเหตุการณ์ที่ถูกเรียก

สำหรับส่วนของ Properties , Methods และ Events ที่ไม่ได้กล่าวถึงสามารถดูได้ใน Help ของ Microsoft Speech SDK 4.0 ซึ่งจะมีคำบรรยายต่างๆ โดยละเอียด

การเปลี่ยนแปลงลักษณะเส้นเสียง (Articulation) อวัยวะที่ใช้ในขั้นตอนนี้ก็คือส่วนที่ต่อจากกล่องเสียงจนถึงริมฝีปากและช่องว่างใหญ่อีก 2 ช่อง คือ ช่องปากและช่องจมูก ขั้นตอนนี้ลมที่ออกจากกล่องเสียงจะถูกแปลงให้เกิดเสียงลักษณะต่างๆ ทั้งนี้เพราะช่องว่างภายในจะถูกเปลี่ยน

แปลงขนาดตลอดเวลาที่เปลี่ยนแปลงเสียง และช่องจมูกจะถูกควบคุมด้วยเพดานอ่อน (Velume) ดังนั้นถ้าช่องว่างเหล่านี้เปลี่ยน ขนาดไปจะทำให้เกิดความถี่กำทอน (Resonant Frequency) ของเสียงเปลี่ยนแปลงไปด้วย ซึ่งช่องว่างเหล่านี้เปรียบเสมือนท่อที่มีขนาดต่างๆกัน

2.8 หน่วยการจำเสียงพูด (Speed recognition Unit)

หน่วยการจำเสียงพูดมีวิธีการจำแนกตามลักษณะการพูดไว้ 4 แบบ

2.8.1 ผู้พูดคนเดิม (Speaker Dependent Recognition)

เป็นการที่ผู้พูดคนเดิมพูดเป็นพยางค์เพียงพยางค์เดียว เป็นคำ หรือเป็นประโยคโดยเราต้องสร้างหน่วยเปรียบเทียบที่เป็นเสียงของคนๆเดียว

2.8.2 ผู้พูดเป็นใครก็ได้ (Spaker Independent Reconition System)

เป็นการที่ใครก็ได้ที่พูดจะเป็นพยางค์เดียว เป็นคำหรือประโยคสอนให้เครื่องรู้จักคำในลักษณะของหน่วยเสียงหรืออย่างมากหน่วยพยางค์ เป็นการสอนให้เครื่องเข้าใจในเสียงที่มีการต่อเนื่องมากกว่าหนึ่งพยางค์ ต้นฉบับจะถูกเก็บไว้ในหน่วยเสียงหรือพยางค์นั้น

2.8.3 ผู้พูดสามารถพูดอย่างต่อเนื่องเหมือนการพูดปกติ (Globolization)

เครื่องจะต้องรู้จัก และจำแนกคำที่มีการออกเสียงเชื่อมกันการพูดแบบนี้ในมนุษย์เป็นเรื่องง่ายแต่สำหรับคอมพิวเตอร์เป็นเรื่องยากมาก เพราะการพูดแบบนี้ไม่สามารถหาเส้นแบ่งเขตระหว่างคำ หรือพยางค์ได้แม่นยำ ดังนั้นการรู้จักโดยการเปรียบเทียบกับต้นแบบจึงไม่ใช่ของง่าย วิธีการแบบนี้ใช้กับเครื่องมินิคอมพิวเตอร์ขึ้นไป

2.9 ActiveX

ActiveX เป็นชื่อเรียกเทคโนโลยีที่ไมโครซอฟท์คิดค้นขึ้น โดยมีวัตถุประสงค์หลักเพื่อสร้างระบบซอฟท์แวร์ที่มุ่งเน้นการใช้งานและพัฒนาสำหรับใช้บนอินเตอร์เน็ตเพื่อทำให้ซอฟท์แวร์ซึ่งอยู่ต่าง Platform สามารถทำงานร่วมกัน โดยเทคโนโลยี ActiveX มีพื้นฐานมาจากแนวความคิดของ COM (Component Object Model) ซึ่งเป็นสถาปัตยกรรมการออกแบบเชิงวัตถุที่มองซอฟท์แวร์เสมือนเป็นวัตถุที่ประกอบด้วยชิ้นส่วนย่อยรวมกันขึ้นมาเป็นซอฟท์แวร์ที่ต้องการ เสมือนว่าซอฟท์แวร์เป็นวัตถุจริงๆแบบเดียวกับฮาร์ดแวร์ เช่น คอมพิวเตอร์จะประกอบไปด้วยวัตถุย่อยๆ คือ CPU , ฮาร์ดดิสก์ , ซีดีรอม และส่วนประกอบอื่นๆที่ประกอบกันขึ้นมาเป็นคอมพิวเตอร์ แค่คอมพิวเตอร์จะมองว่าซอฟท์แวร์เสมือนประกอบด้วยส่วนย่อย โดยแต่ละส่วนอาจอยู่คนละ

Platform แยกจากกัน โดยซอฟท์แวร์ที่สร้างบนสถาปัตยกรรมคอมพิวเตอร์จะสามารถสื่อสารทำงานร่วมกันได้แม้ว่าจะถูกสร้างหรือทำงานบน Platform ที่ต่างกันก็ตาม

2.10 ActiveX Control

ActiveX Control หมายถึง คอนโทรลที่เพิ่มเติมจากออบเจ็คพื้นฐานที่ Visual Basic ได้เตรียมไว้ให้แล้วใน Toolbox โดย ActiveX Control เหล่านี้มีอยู่มากมายทั้งที่มาพร้อมกับ Visual Basic และที่พัฒนาออกมาขายโดยบริษัทต่างๆทั่วโลก ซึ่ง ActiveX Control เหล่านี้เองคือพลังผลักดันให้โปรแกรม Visual Basic และเครื่องมืออีกหลายตัวของไมโครซอฟท์เป็นที่นิยมอย่างแพร่หลายเนื่องจากผู้ใช้มองตัว ActiveX Contral เป็นเพียงกล่องดำหรือกล่องวัตถุ ซึ่งเพียงแค่นำคอนโทรลที่มีการทำงานตามที่ต้องการ มาแปะบนฟอร์มแล้วเรียกใช้งานเท่านั้น โดยไม่ต้องสนใจการทำงานภายในของ ActiveX Control เหล่านั้น เช่น ถ้าเราต้องการเขียนโปรแกรมเพื่อติดต่อกับอินเตอร์เน็ต (Internet) ถ้าเป็นในอดีตต้องศึกษาข้อมูลมากมาย เช่น Protocol การติดต่อสื่อสาร TCP/IP ซึ่งเป็นมาตรฐานของภาษาที่คอมพิวเตอร์ใช้สื่อสารกันในระบบอินเตอร์เน็ต แต่ถ้าใช้ ActiveX Control เพียงแค่นำ Internet ActiveX Control มาวางลงบนฟอร์มแล้วเรียนรู้วิธีใช้เท่านั้น ก็สามารถเขียนโปรแกรมที่ติดต่อกับเครือข่ายอินเตอร์เน็ตได้ในเวลาอันสั้น

2.11 Microsoft Speech SDK

Microsoft Speech SDK (Software Developer's Kit) เป็น ActiveX Control อีกอย่างหนึ่งซึ่งไม่มีในออบเจ็คพื้นฐานของ Visual Basic ต้องหาซอฟท์แวร์ตัดนี้มาทำการติดตั้งเข้าไปในคอมพิวเตอร์ที่ใช้จึงจะสามารถเรียกใช้ ActiveX Control ตัวนี้ได้

ใน SDK จะมี ActiveX Control มาให้ 6 คอนโทรลด้วยกัน ซึ่งสามารถใช้ในการรู้จำเสียง (Speech Recognition) , เขียนตามคำบอก (Dictation) , การสังเคราะห์เสียง (Speech Synthesis) และโทรศัพท์ (Telephony) ในสภาพแวดล้อมต่างๆ เช่น Visual Basic , Microsoft Office และ Internet Explorer ทั้ง 6 คอนโทรลนั้น ได้แก่

Voice Command Control

Voice Dictation Control

Voice Text Control

Voice Telephony Control

Direct Speech Recognition Control

Direct Speech Synthesis Control

ซึ่งทั้ง 6 คอนโทรลจะประกอบไปด้วย 2 คอนโทรลที่ทำหน้าที่รู้จำเสียง (Direct Speech Recognition และ Voice Command Control) มี 1 คอนโทรลทำหน้าที่เขียนตามคำบอก (Voice Dictation Control) มี 2 คอนโทรลทำหน้าที่สังเคราะห์เสียง (Direct Speech Synthesis Control และ Voice Text Control) และสุดท้ายอีก 1 คอนโทรลทำหน้าที่ติดต่อโทรศัพท์ (Voice Telephony Control)

ในคอนโทรล Direct Speech Recognition Control และ Direct Speech Synthesis Control นั้นตัวคอนโทรลได้จัดการการเข้าถึงข้อมูลในการติดต่อกับ Speech API (Application Programming Interface) อย่างเต็มที่ สำหรับการสังเคราะห์เสียงและการรู้จำมันจะติดต่อกับเครื่องมือเสียงในระดับต่ำทำให้ทำงานได้เร็วและควบคุมได้ง่าย

Voice Command Control และ Voice Text Control จะจัดการการเข้าถึงข้อมูลโดยการแชร์ (Share) Speech API ในระดับสูงในการเชื่อมต่อจะมีขีดจำกัดและช้ากว่าแบบแรกที่ติดต่อในระดับต่ำ แต่จะสะดวกกว่าโดยการจัดเตรียมทรัพยากรและการแชร์หน่วยความจำ (Memory) ระหว่างโปรแกรมที่ทำงานเกี่ยวกับเสียงจะเป็นไปโดยอัตโนมัติ

Voice Dictation Control เป็นการง่ายที่จะทำการเขียนตามที่ผู้ใช้บอกอย่างถูกต้องและแปลงเสียงเป็นข้อความตามปกติโดยอัตโนมัติ

Voice Telephony Control เป็นเครื่องมือในการติดต่อกับการสังเคราะห์เสียง , การรู้จำเสียง การสังเคราะห์เสียงแบบ Ware และ DTMF บน Single หรือ Multi-line voice ของอุปกรณ์โทรศัพท์

ในที่นี้จะกล่าวถึงเฉพาะ Voice Command Control ซึ่งเป็นคอนโทรลที่นำมาใช้เท่านั้นส่วนคอนโทรลตัวอื่นๆจะไม่ขอกล่าวถึงรายละเอียดมากนัก แต่จะมีเนื้อหาคร่าวๆในภาคผนวก

2.12 เทคนิคและทฤษฎีที่ใช้ในการดำเนินโครงการ

2.12.1 speech processing technology

การทำงานของคอมพิวเตอร์ที่นำ speech processing technology มาใช้สามารถแบ่งได้เป็น 2 ประเภท ดังนี้

1.Speech recognition หรือ Speech - to - Text เป็นเทคโนโลยีที่ใช้วิเคราะห์ และควบคุมการทำงานคอมพิวเตอร์ด้วยเสียง โดยเอนจิ้นจะพยายามทำความเข้าใจในชุดข้อมูลเสียงที่ได้รับ ด้วยการตรวจสอบโครงสร้างและรูปแบบของเสียงว่าถูกต้องตามที่กำหนดไว้หรือไม่ และจะดำเนินการตามเงื่อนไขของโปรแกรมที่ได้กำหนดไว้

Speech recognition หรือ Speech-to-Text จะเกี่ยวข้องกับการจับคลื่นเสียงและการแปลงคลื่นเสียงเป็นข้อมูล digital , การแปลงให้อยู่ใน phonemes , การสร้างคำจาก phonemes และการ

2. แปลภาษาเขียนให้อยู่ในสัญลักษณ์ Phonetic และ ฉันทลักษณ์ (Prosodic) และสร้างเสียงที่เป็นดิจิตอล (Digital Audio)

3. แปลและขยายสัญญาณผ่านทางลำโพง

4. Output ที่ได้จะได้ยินเสียงออกมาจากลำโพง

Synthesizer (Text-to-Speech) เป็น software driver ที่จัดการความซับซ้อนของการแปลงข้อความและสร้างคำพูด Text-to-Speech จะทำการสร้างเสียงที่คล้ายกับเสียงมนุษย์ และใช้ตัวกรองต่างๆ เพื่อจำลองเสียงที่ออกมาจากลำคอ, ช่องปาก, ลักษณะของริมฝีปาก และตำแหน่งของลิ้น Speech Synthesis หรือ Text-to-Speech (TTS) เป็นกระบวนการที่เกี่ยวข้องกับการแปลงข้อความเป็น digital audio แล้วส่งออกมาเป็นเสียงให้ผู้ใช้ได้ยิน ซึ่งสามารถแบ่งประเภทของ Text-to-Speech engine ได้จากวิธีที่ใช้ในการแปลงส่วนที่เล็กที่สุดของเสียง (phonemes) เป็นเสียงที่สามารถได้ยินได้ทั้งหมด 3 ประเภท ดังนี้

1. Concatenated Word ถึงแม้ว่าวิธีนี้จะไม่ใช่กระบวนการสังเคราะห์เสียงอย่างแท้จริง แต่ก็เป็นวิธีที่ใช้กันมากในระบบ Text - to - Speech ทั่วๆไป โดยเอนจินที่ใช้วิธีนี้ผู้ออกแบบโปรแกรมจะทำการบันทึกคำพูดและคำเฉพาะลงในฐานข้อมูล เมื่อมีการเรียกใช้โปรแกรมตัวเอนจินโปรแกรมจะทำการดึงเสียงที่ได้บันทึกไว้จากฐานข้อมูล แล้วนำมาต่อกันและออกเสียงเป็นประโยคหรือคำพูด ยกตัวอย่างเช่น โปรแกรม voice-mail เมื่อได้รับ e-mail ใหม่ เอนจินจะพูดว่า "[You have][three][new message]" ซึ่ง เอนจินนี้จะต้องมีการบันทึกคำว่า " You have", ตัวเลขทั้งหมด และ "new message"

2. Synthesis Text-to-Speech engine จะใช้กระบวนการสังเคราะห์เสียงเพื่อทำการสร้างเสียงที่คล้ายเสียงพูดของมนุษย์ และการใช้ตัวกรองเพื่อจำลองเสียงที่ออกมาจากลำคอ, ช่องปาก, ลักษณะของริมฝีปากและตำแหน่งของลิ้น เสียงที่ถูกสร้างขึ้นโดยเทคโนโลยีการสังเคราะห์เสียงปัจจุบันมีแนวโน้มที่จะได้เสียงใกล้เคียงกับเสียงมนุษย์น้อย กว่าเสียงที่ผลิตโดยวิธี Diphone Concatenation แต่ก็มีความเป็นไปได้ที่จะทำให้มีคุณภาพเสียงดีขึ้นโดยการเปลี่ยน parameter บางตัว

3. Diphone Concatenation Text-to-Speech engine จะใช้การปะติดปะต่อเชื่อมส่วนของ digital-audio สั้นๆ เข้าด้วยกันและทำให้รอยต่อระหว่างเสียงมีความราบรื่นเพื่อให้เสียงที่ได้มีความต่อเนื่องมากขึ้น การเตรียมส่วนต่างๆ ของเสียงจะได้มาจากการบันทึกเสียงของมนุษย์ในหลายรูปแบบและจะต้องพยายามระบุการเริ่มต้นและการสิ้นสุดของพยางค์เสียง ถึงแม้ว่าเทคนิคนี้สามารถสร้างเสียงที่มีความเหมือนจริงมากขึ้น แต่ก็ต้องคำนึงถึงจำนวนของงานเพื่อสร้างเสียงใหม่และต้องเป็นเสียงที่ไม่จำกัดเพราะว่าพยางค์เสียงจะขึ้นอยู่กับภาษาที่ผู้พูดใช้เท่านั้น

Text-to-Speech (TTS) มีการทำงานเป็นลำดับขั้นตอน ซึ่งการทำงานหลักๆ คือ การเปลี่ยนข้อความให้เป็นเสียงพูด โดยประกอบด้วยขั้นตอนต่างๆ ดังต่อไปนี้

1. Text Normalization เป็นขั้นตอนในการแยกข้อความที่ป้อนเข้าไปเป็นชุดของคำ ทำให้ข้อความมีความซับซ้อนน้อยลง

2. Homograph disambiguation วิเคราะห์คำที่เขียนเหมือนกัน แต่ออกเสียงต่างกัน โดยวิเคราะห์จาก ความหมายของข้อความและลักษณะเฉพาะของประโยค

3. Word pronunciation การวิเคราะห์และสร้างพยางค์เสียง (การแยกส่วนประกอบของคำศัพท์ แล้วทำการประมวลผลว่าตัวอักษรตัวใดจะออกเสียงอย่างไร)

4. Prosody หรือฉันทลักษณ์ คือระดับเสียงสูงต่ำ , ความเร็วในการออกเสียงรวมไปถึงความหนักเบาของการออกเสียงพยางค์

5. Concatenate wave segments ทำการเปลี่ยนข้อมูลของพยางค์เสียงและคุณสมบัติต่างๆ ของตัวพยางค์เสียงที่ได้มาจากขั้นตอนต่างๆ ให้อยู่ในรูป digital audio

บทที่ 3
การออกแบบ

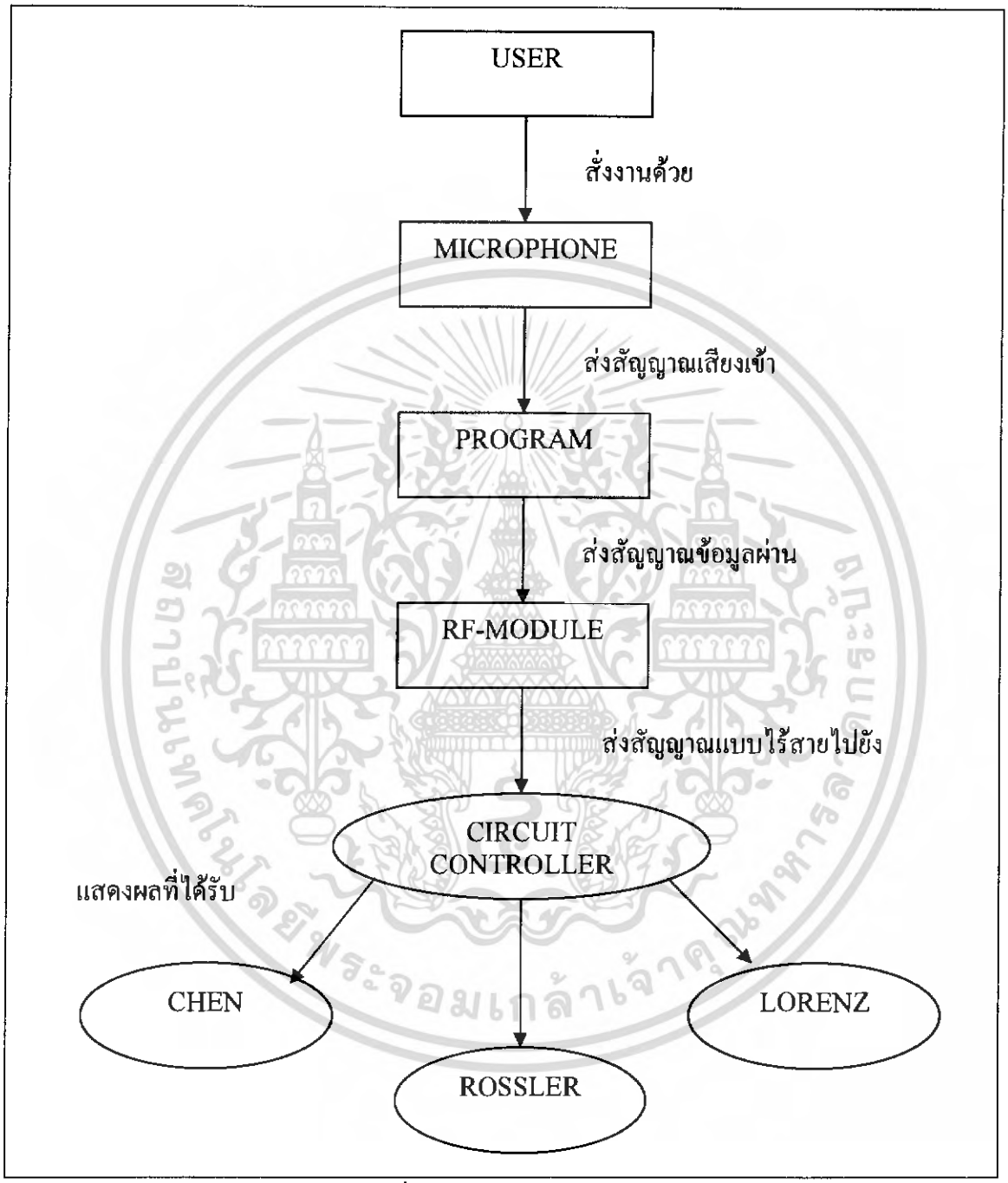
3.1 โครงสร้างโดยรวม

แนวความคิดในการออกแบบระบบ จะพัฒนาเป็น Application Program ที่มีผู้ใช้งานโปรแกรมสามารถสั่งงานในการทำงานในฟังก์ชั่นต่างๆของโปรแกรมได้ด้วยเสียงของผู้ใช้งานแทนการกดปุ่มที่แป้นพิมพ์หรือการคลิ๊กเมาท์ โดยการสั่งงานด้วยเสียงจะมีไมโครโฟนเป็นตัวรับสัญญาณ แล้วนำสัญญาณเสียงเข้าโปรแกรมในคอมพิวเตอร์เพื่อทำการประมวลผลว่าตรงกับคำสั่งให้ที่ได้กำหนดค่าไว้ก่อนหน้านี้ แล้วจึงทำการส่งข้อมูลออกผ่านมาทางโมดูลไร้สายแล้วข้อมูลที่ส่งออกมาจากเครื่องคอมพิวเตอร์จะเข้าไปในวงจรควบคุมที่ตัวหุ่นยนต์ เมื่อตัวหุ่นยนต์ได้รับสัญญาณที่ส่งมาก็จะนำไปประมวลผลที่ไมโครคอนโทรลเลอร์MCS-51โดยการแสดงผลการทำงานนั้น โปรแกรมจะทำการอ่านค่าข้อมูลจากพอร์ต ที่โปรแกรมในส่วนของการส่งค่าได้ส่งออกไป โดยในส่วนของโปรแกรมที่ใช้ในการอ่านค่าจะเป็นส่วนหนึ่งของโปรแกรมที่ไม่เกี่ยวข้องกับระบบการทำงานของทั้งโปรแกรม ซึ่งส่วนของการแสดงผลจะวนรอบการรับค่ามาแสดงตลอดเวลาโดยค่าที่ได้จากการอ่านพอร์ต จะแสดงบนหน้าจอติดต่อกับผู้ใช้และแสดงผลด้วยเสียงทันทีที่มีการเปลี่ยนแปลงการทำงานของอุปกรณ์บนวงจรควบคุม

ซึ่งโครงสร้างโดยรวมทั้งหมดที่กล่าวมากข้างต้นสามารถสรุปเป็นได้ดังรูปที่ 3.1

รูปที่ 3.1 โครงสร้างโดยรวมของโครงงาน

จากโครงสร้างโดยรวมเราสามารถนำมาแสดงเป็นโฟลชาร์ตได้ ดังนี้



รูปที่ 3.2 โครงสร้างโดยรวม

เมื่อพิจารณาโครงสร้างโดยรวมเราสามารถแบ่งโครงสร้างหลักออกเป็น 3 ส่วน ได้แก่

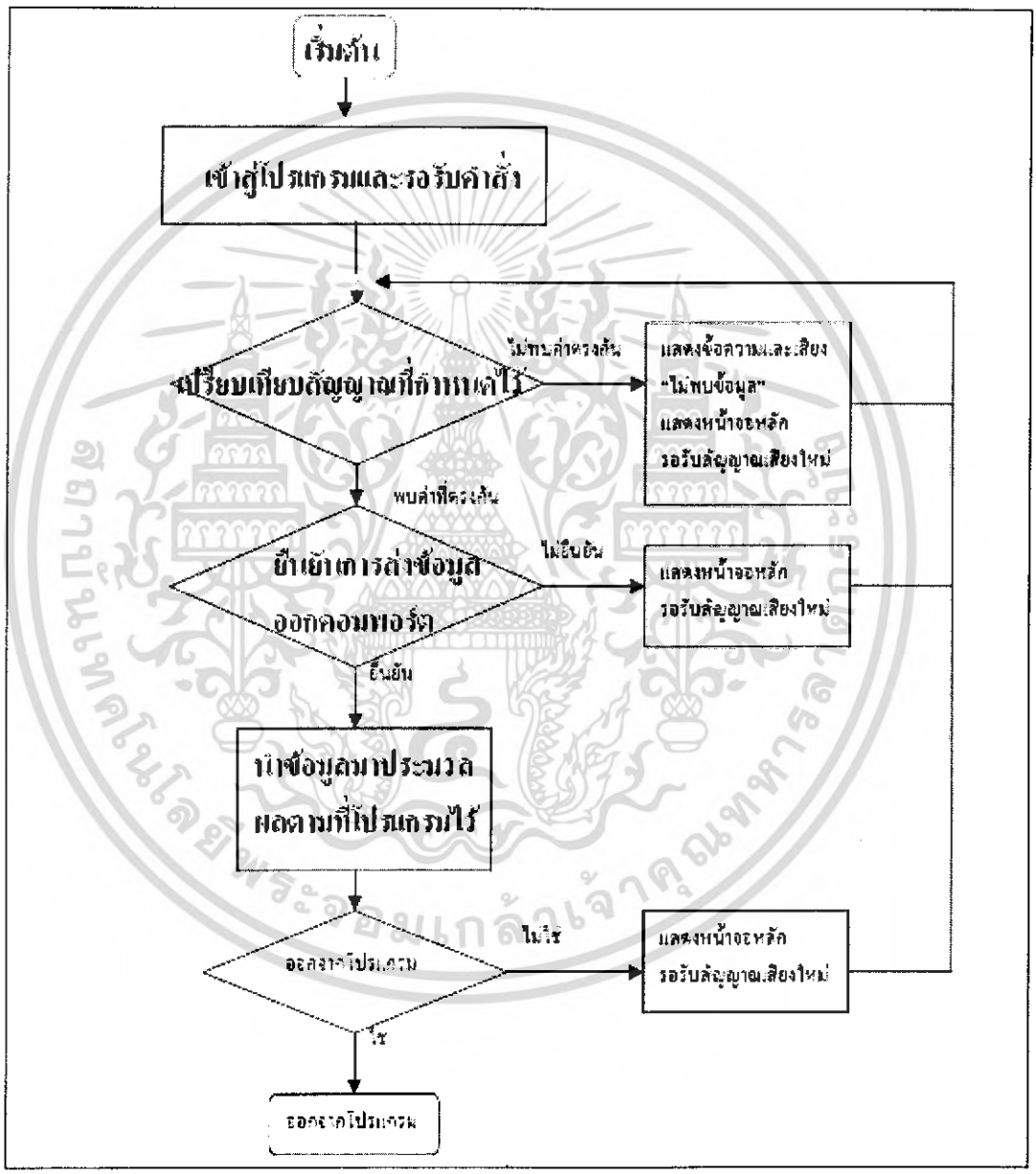
1.ส่วนของอินพุต

2.ส่วนของการควบคุม

3.ส่วนของเอาท์พุต

3.2 ส่วนของอินพุต

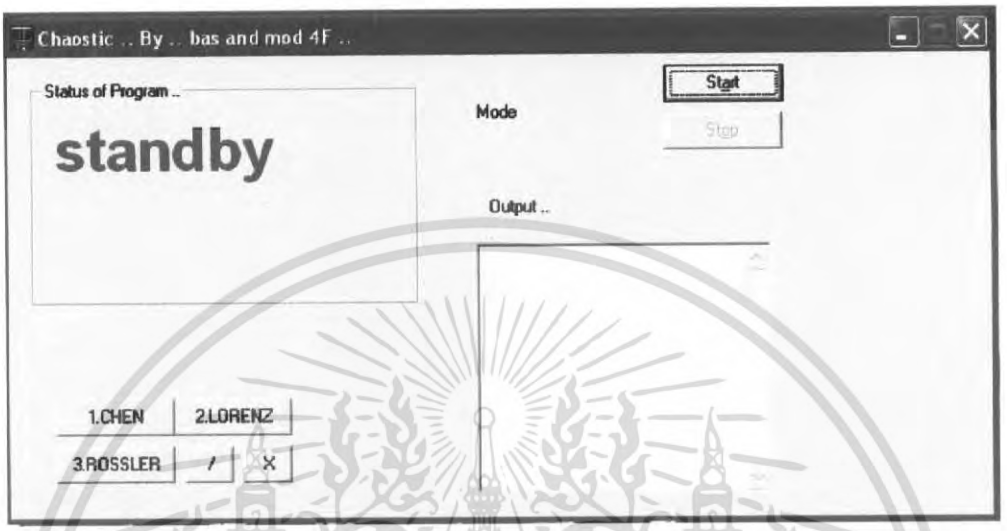
ผู้ใช้ที่ทำหน้าที่สั่งงานด้วยเสียงผ่านไมโครโฟน โปรแกรมหลักนั้นจะนำสัญญาณเสียงที่ได้มาประมวลผลเปรียบเทียบกับค่าที่ได้กำหนดไว้แล้ว แล้วจึงส่งข้อมูลที่ได้จากการประมวลผลออกไปทางโมดูลไร้สาย(RF-Module) ซึ่งสามารถแสดงกระบวนการทำงานได้ดังรูปที่ 3.2



รูปที่ 3.3 แสดงส่วนของการอินพุต

ซึ่งสำหรับในส่วนที่ใช้ในการติดต่อกับผู้ใช้โปรแกรมมีลักษณะดังนี้

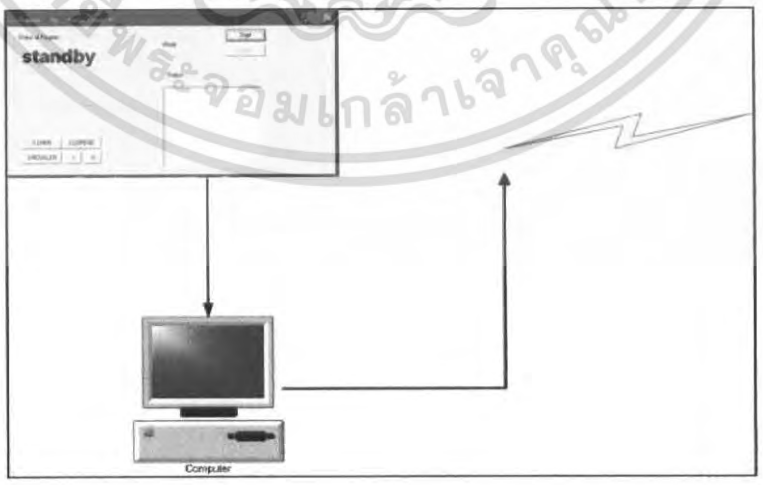
- ส่วนที่ใช้ในการควบคุมการทำงานในการติดต่อกับผู้ใช้



รูปที่ 3.4 Application User Control

ในส่วนนี้จะมีการออกแบบให้มีปุ่มที่สามารถใช้เม้าส์คลิกเลือกได้หรือเมื่อมีการใช้เสียงสั่งแทนการใช้เม้าส์คลิกให้เกิดการเคลื่อนที่ โดยก่อนที่จะมีกาใช้เม้าส์คลิกหรือใช้เสียงสั่งต้องมีการกดปุ่มเครื่องหมายถูก (/) เพื่อทำการเปิดพอร์ตก่อนที่จะส่งข้อมูล สำหรับช่องสีขาวจะมีไว้แสดงค่าที่โปรแกรมประมวลผลออกมาเพื่อส่งไปผ่านโมดูลไร้สายไปยังตัวหุ่นยนต์

-ส่วนของคอมพิวเตอร์และโมดูลเครื่องส่ง

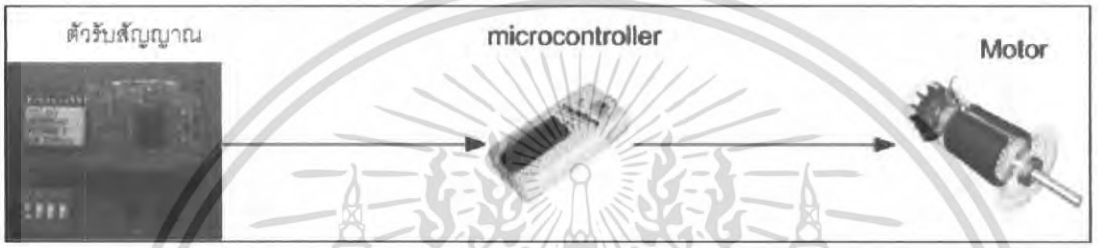


รูปที่ 3.5 Block User Control

การทำงานในส่วนนี้จะทำงานหลังจากที่ได้รับคำสั่งจากผู้ใช้ โดยจำนำคำสั่งไปประมวลผลในโปรแกรมที่ได้เขียนไว้ หลังจากนั้นจึงนำค่าที่ได้จากการประมวลผลส่งผ่านโมดูลไร้สายไปยังตัวรับสัญญาณบนตัวหุ่นยนต์เพื่อนำไปประมวลผลทำให้หุ่นยนต์เกิดการเคลื่อนที่ต่อไป

3.3 ส่วนของการควบคุม

ลักษณะการทำงานของส่วนควบคุมจะมีลักษณะ ดังนี้



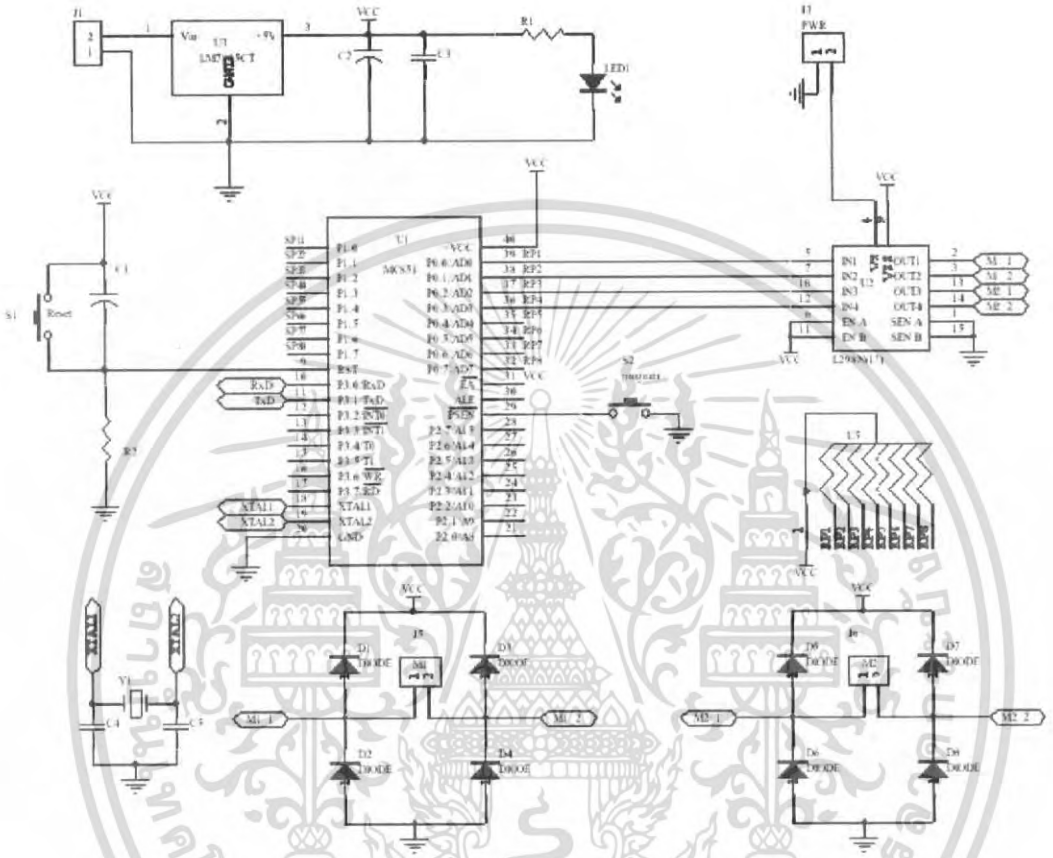
รูปที่ 3.6 ลักษณะการทำงานของส่วนควบคุม

ในการทำงานของส่วนนี้จะเริ่มหลังจากได้รับสัญญาณมาจากโปรแกรมบนคอมพิวเตอร์ เมื่อได้รับสัญญาณก็จะนำสัญญาณมาให้ไมโครคอนโทรลเลอร์ประมวลผลแล้วส่งข้อมูลไปให้มอร์เตอร์เป็นพัลส์วิธเพื่อควบคุมความเร็ว

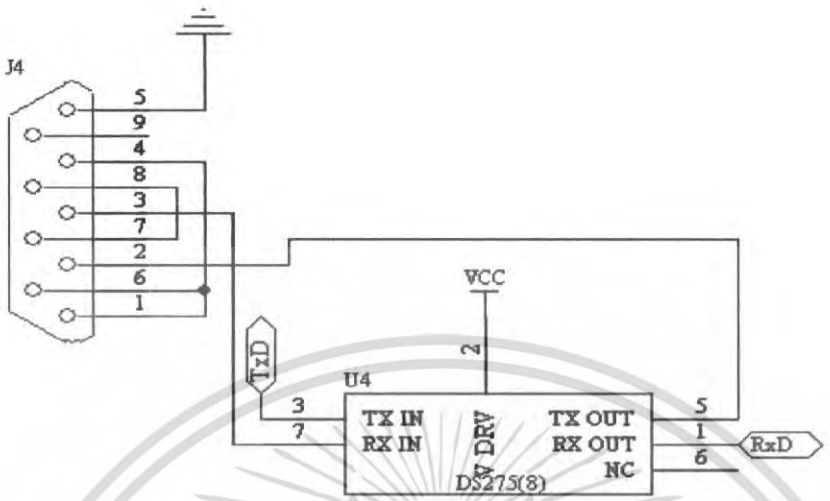
โดยส่วนของการควบคุมจะแบ่งออกเป็น 2 ส่วนคือส่วนของวงจรควบคุม(Controller) และส่วนของโปรแกรมที่ใช้ควบคุมการทำงาน

3.3.1 วงจรควบคุม

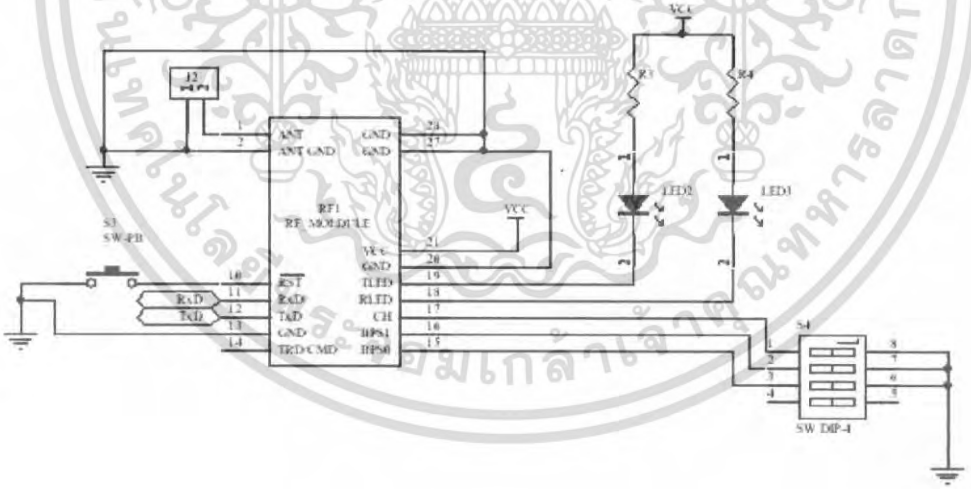
ประกอบด้วยไมโครคอนโทรลเลอร์ 1 ตัวและอุปกรณ์ต่างๆ ดังนี้



รูปที่ 3.7 วงจรควบคุมการทำงานของมอเตอร์



รูปที่ 3.8 วงจรรับ-ส่งข้อมูลผ่านพอร์ตอนุกรม

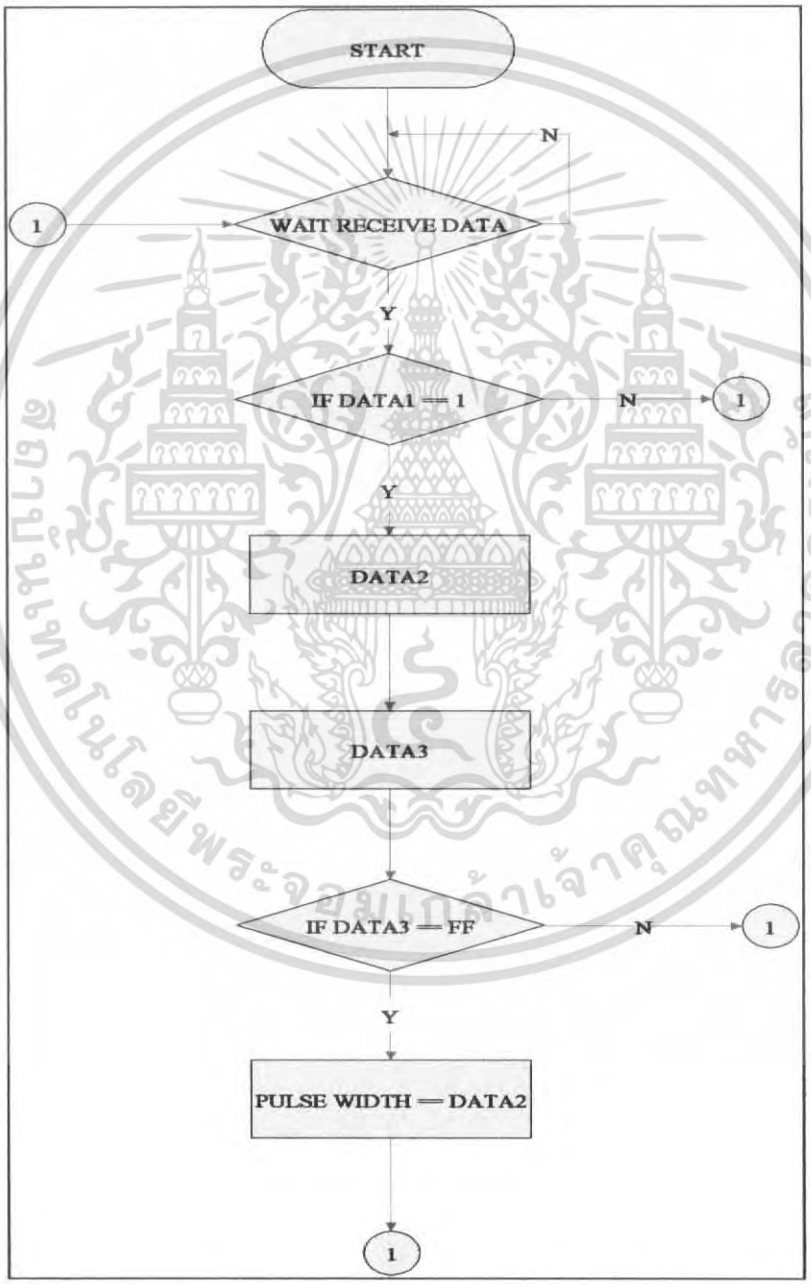


รูปที่ 3.9 วงจรรับ-ส่งข้อมูลผ่านอาร์เอฟโมดูล

จากรูปที่ 3.3 เป็นการแสดงการต่อวงจรควบคุมการทำงานซึ่งใช้ไมโครคอนโทรลเลอร์เบอร์ P89C51RD2 เพื่อใช้ควบคุมการทำงานของมอเตอร์ 2 ตัวที่ใช้ในการขับเคลื่อนตัวหุ่นยนต์ ส่วนในรูปที่ 3.4 เป็นการแสดงวงจรรับ-ส่งวงจรผ่านพอร์ตอนุกรม เพื่อใช้ในการทดสอบวงจร

ควบคุม และในรูปที่ 3.5 คือวงจรสำหรับการรับ-ส่งข้อมูลแบบไร้สายซึ่งเป็นชุดวงจรสำเร็จรูปซึ่งมีรูปแบบการต่อวงจรตามคู่มือ (datasheet) โดยสามารถเลือกความถี่ที่ใช้งานและอัตราความเร็วของการส่งข้อมูลได้

3.3.2 โปรแกรมควบคุมการทำงาน

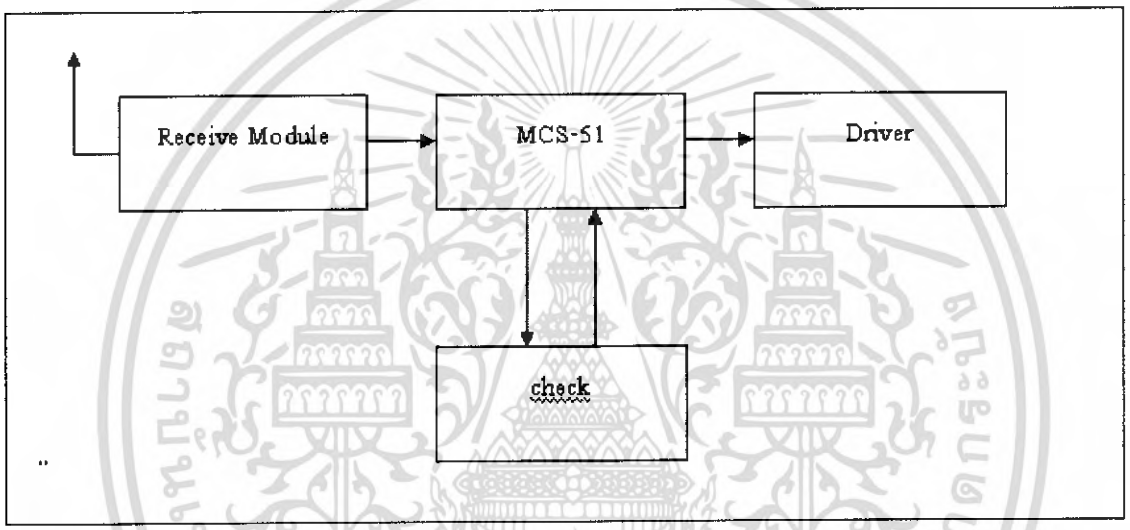


รูปที่ 3.10 Flow chart ควบคุมการทำงานแบบ manual

ส่วนของโปรแกรมควบคุมการเคลื่อนที่จะใช้การ Interrupt จาก I/O เป็นหลัก โดยในการควบคุม คอมพิวเตอร์จะส่งข้อมูลมาที่วงจรควบคุม จากนั้นไมโครคอนโทรลเลอร์จะนำค่าของข้อมูลที่เข้ามาเก็บไว้ในตัวแปรอินพุตแล้วทำการเปรียบเทียบค่าว่าต้องการให้มอเตอร์หมุนในทิศทางใด

3.3.3 ลักษณะการทำงานของวงจรควบคุม

ลักษณะการทำงานของวงจรควบคุมสามารถแสดงด้วยไดอะแกรมได้ดังนี้

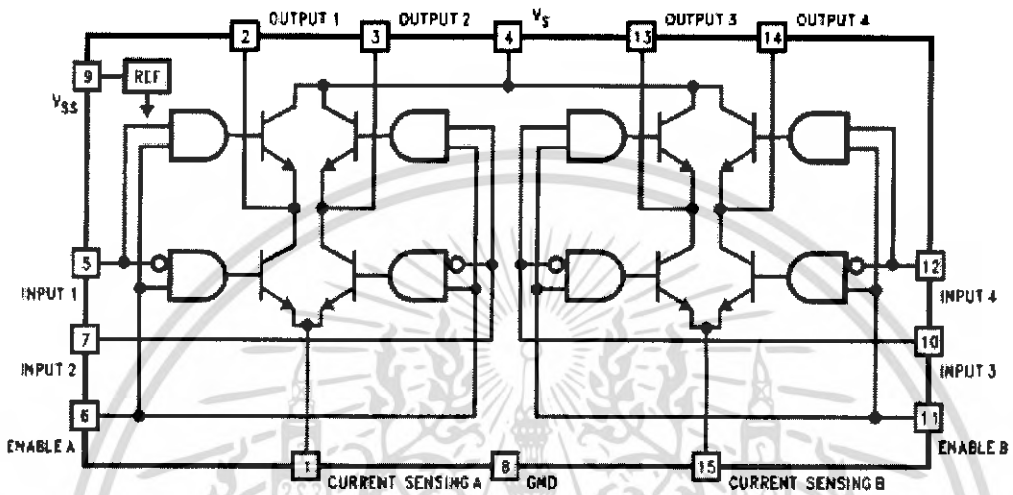


รูปที่ 3.11 ไดอะแกรมแสดงการทำงานของวงจรควบคุม

จากไดอะแกรมเราสามารถอธิบายการทำงานได้ดังนี้ เมื่อได้รับสัญญาณจากโมดูล ตัววงจรก็นำสัญญาณที่ได้มาแปลงและนำไปประมวลผลตัว MCS-51 จะเช็คว่าข้อมูลคำสั่งที่ได้เป็นคำสั่งอะไรโดยนำไปเปรียบเทียบกับโปรแกรมที่ได้เขียนไว้หลังจากนั้นมันก็ทำการขับมอเตอร์ให้หมุนตามคำสั่งที่ได้รับ

3.4 ส่วนของเอาท์พุท

ออกแบบวงจรขับเคลื่อนมอเตอร์ DC โดยใช้ LM298 เนื่องจากสามารถทนแรงดันได้สูงกว่าไอซีตัวอื่นๆโดยมีลักษณะภายในดังรูป



รูปที่ 3.12 วงจรขับเคลื่อนมอเตอร์ของ LM298

โดยมีการทำงานแบบH-Bridge หรือสะพานรูปตัว H ถ้าเราป้อนลอจิก เพื่อสั่งให้มอเตอร์ทำงานเป็น 01 หรือ 10 มอเตอร์จะหมุนเดินหน้าหรือถอยหลังแต่ถ้าเราสั่ง 00 หรือ 11 มอเตอร์จะหยุดหมุน แต่ถ้าต้องการให้มอเตอร์หมุนกลับทางก็ทำการสลับข้อมูลที่ส่งไปจาก 01 เป็น 10

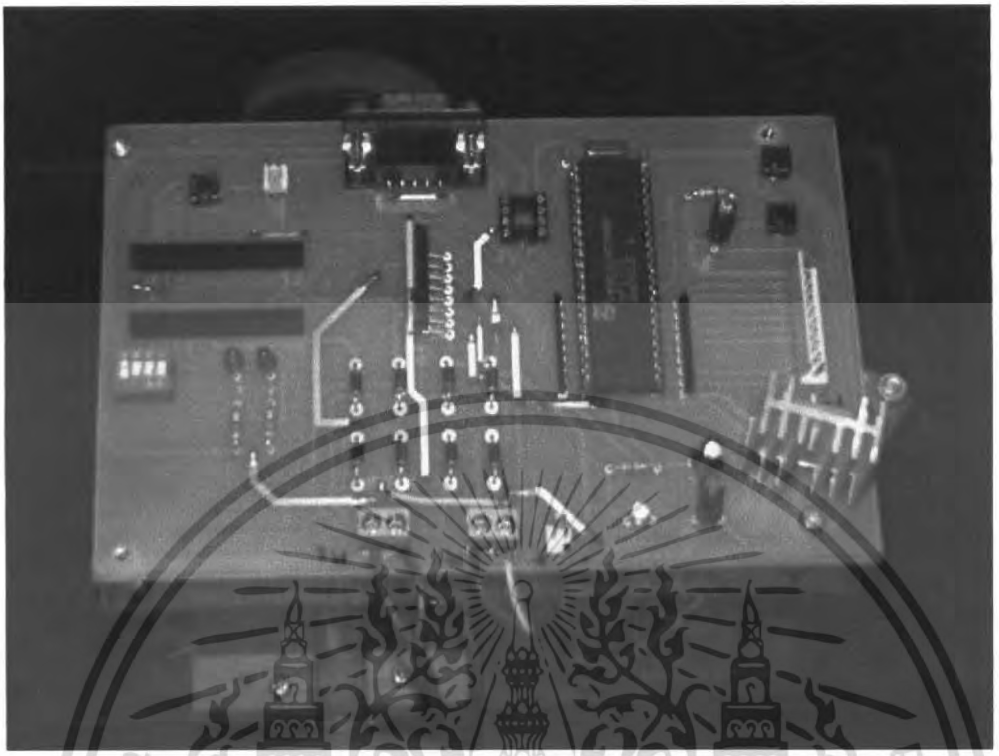
บทที่ 4

ผลการทดลอง

4.1 ผลจากการออกแบบ

จากการออกแบบวงจรควบคุมละนำไปติดตั้งบนตัวหุ่นยนต์จะได้ผลดังรูป

ก



ข

รูปที่ 4.1 ก และ ข แสดงการติดตั้งวงจรควบคุมบนตัวหุ่นยนต์ที่ใช้ในการทดลอง

ในการดำเนินงานโครงงานจะมีการแบ่งการทดลองออกเป็น 2 ส่วน คือ

1.ส่วนของการเคลื่อนที่แบบอลวน

2.ส่วนของการควบคุมหุ่นยนต์ด้วยเสียง

4.2 การทดลองการเคลื่อนที่แบบอลวน

ในส่วนนี้จะแบ่งการทดสอบเป็น 2 รูปแบบ คือ การทดสอบการเคลื่อนที่ของฮาร์ดแวร์และการทดสอบซอฟแวร์โดยการส่งค่าจากโปรแกรมบนเครื่องคอมพิวเตอร์มายังฮาร์ดแวร์เพื่อใช้ในการควบคุมการเคลื่อนที่

4.2.1 การทดลองทางฮาร์ดแวร์

ในการทดสอบฮาร์ดแวร์จะมีจุดประสงค์เพื่อ สังเกตการทำงานของตัวหุ่นยนต์ในการเคลื่อนที่และตรวจสอบโปรแกรมที่เขียนไว้เพื่อรับค่าที่จะส่งมาจากคอมพิวเตอร์ โดยอาศัยหลักการในเรื่องของความเร็วที่ต่างกันโดยการนำค่าการเคลื่อนที่ปบบอลวนที่ได้จากสมการมาแปลงค่าให้

อยู่ในช่วง 0 – 15 แล้วให้ไมโครโปรเซสเซอร์จ่ายความเร็วให้กับมอเตอร์ในรูปของพัลส์วิธ (Pulse width) มาช่วยทำให้เกิดการเปลี่ยนทิศทางของตัวหุ่นยนต์

การทดสอบการทำงานของหุ่นยนต์ มีวิธีการทดลองดังนี้

1.กำหนดให้ค่า X = ความเร็วของล้อซ้าย และ ค่า Y = ความเร็วของล้อขวา

2.กำหนดให้รับค่าความเร็วที่ได้อยู่ในช่วง 0-15

3.ทดสอบโดยการกำหนดค่า X และ Y แบ่งเป็น 3 กรณี ส่งผ่านพอร์ทอนุกรม ดังนี้

1. X = Y

2. X > Y

3. X < Y

4.ทดสอบโดยการส่งค่า X และ Y ทั้ง 3 กรณี ผ่านอาร์เอฟโมดูล

5.สังเกตการเคลื่อนที่ของหุ่นยนต์

6.ทดสอบโดยการวัดค่าพัลส์วิธผ่านออสซิลโลสโคป (Oscilloscope)

7.สังเกตค่าในออสซิลโลสโคปและบันทึกค่า

4.2.2 ผลการทดลองทางฮาร์ดแวร์

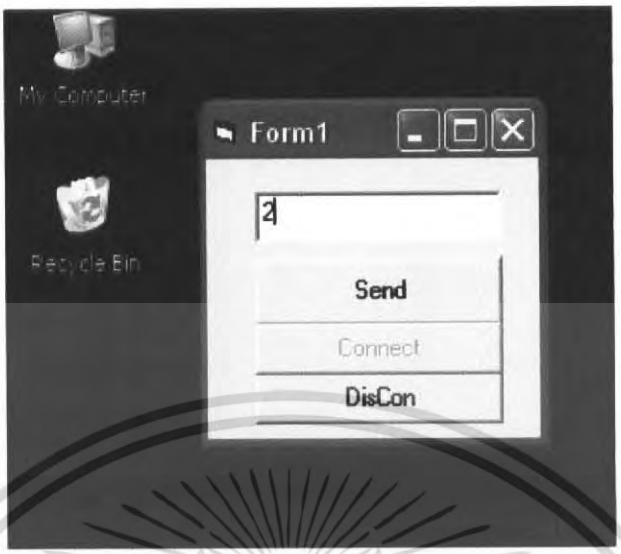
จากการกำหนดค่า X และ Y ผ่านพอร์ทอนุกรมและผ่านอาร์เอฟโมดูลทั้ง 3 กรณีจะได้ผลการทดลองดังนี้

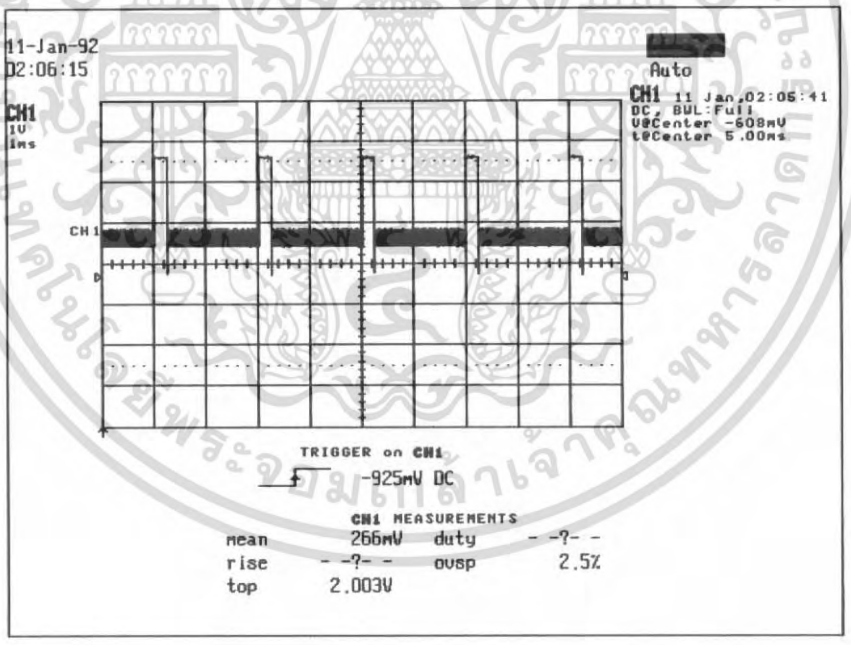
1. X = Y หุ่นยนต์จะเคลื่อนที่ตรงไปข้างหน้า

2. X > Y หุ่นยนต์จะมีการเคลื่อนที่ไปทางขวา

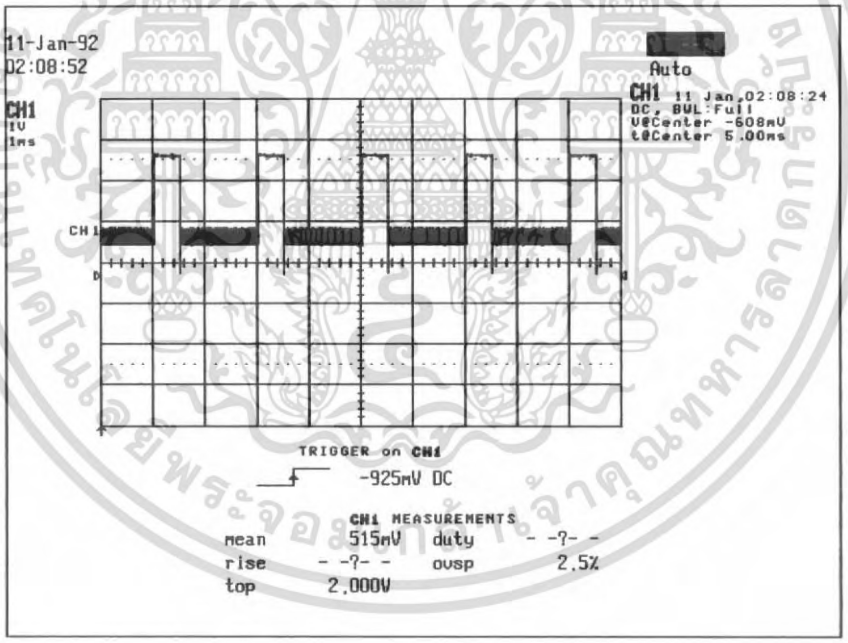
3. X < Y หุ่นยนต์จะมีการเคลื่อนที่ไปทางซ้าย

4.ค่าที่อ่านได้จากออสซิลโลสโคปเมื่อส่งค่าความเร็วที่แตกต่างกันได้ดังนี้

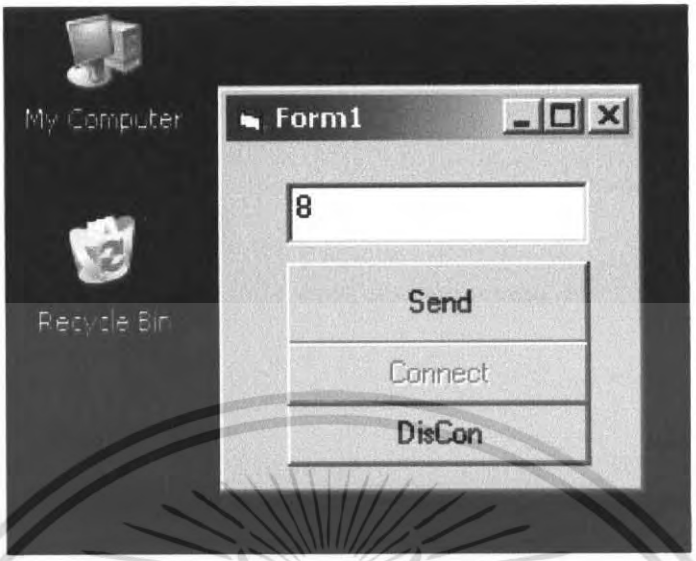


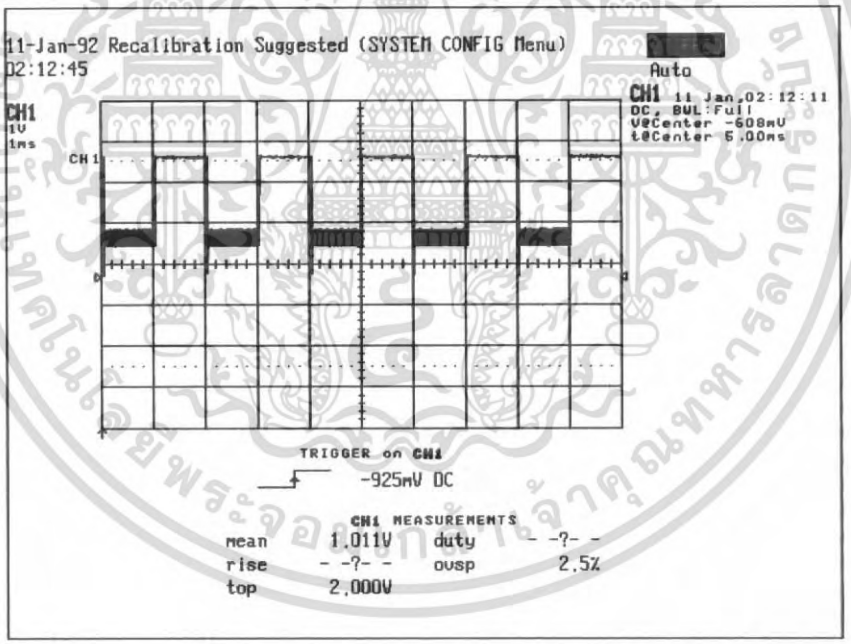


รูปที่ 4.2 แสดงค่าของพัลส์วิธเมื่อส่งค่าความเร็วเท่ากับ 2

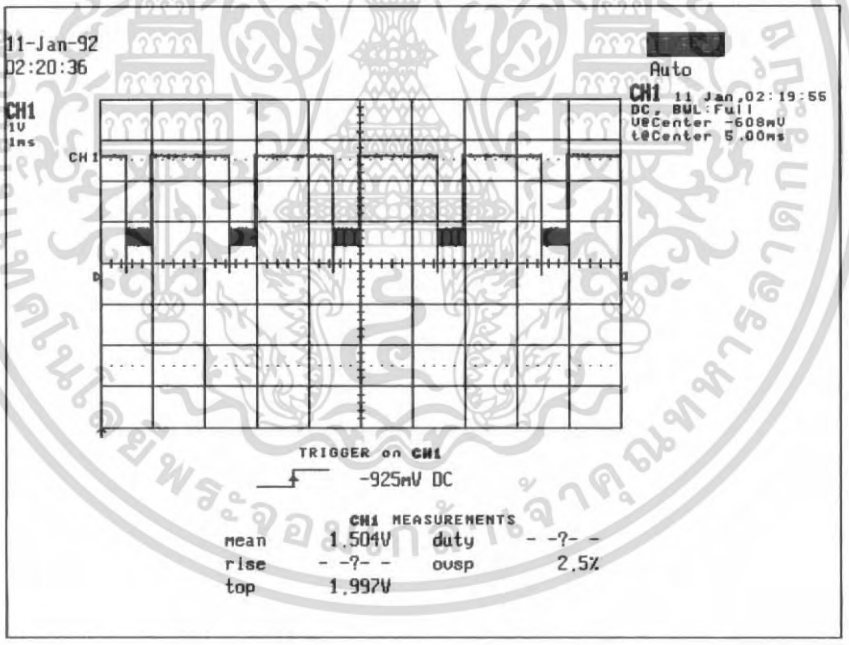


รูปที่ 4.3 แสดงค่าของพัลส์วิธเมื่อส่งค่าความเร็วเท่ากับ 4

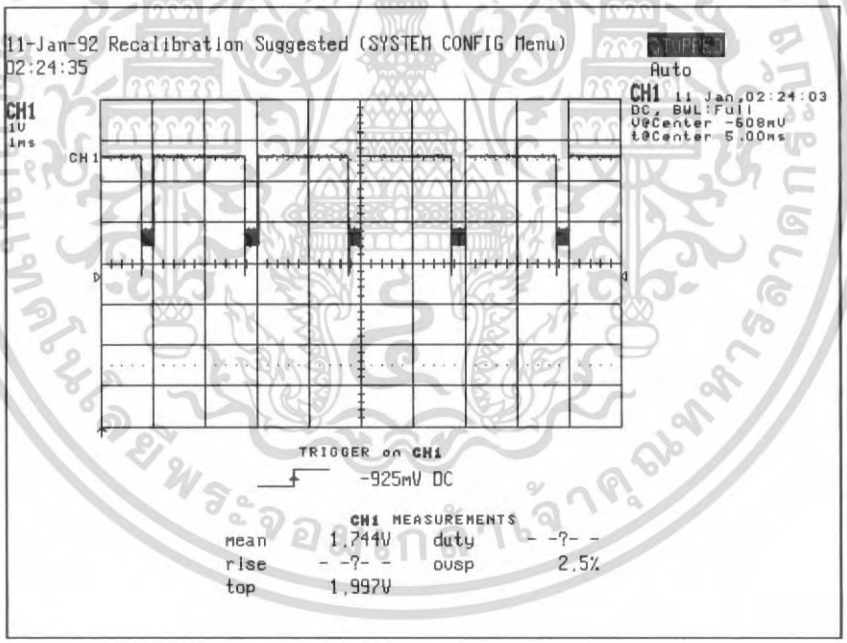




รูปที่ 4.4 แสดงค่าของพัลส์วิธเมื่อส่งค่าความเร็วเท่ากับ 8



รูปที่ 4.5 แสดงค่าของพัลส์วิธเมื่อส่งค่าความเร็วเท่ากับ 12



รูปที่ 4.6 แสดงค่าของพัลส์วิธเมื่อส่งค่าความเร็วเท่ากับ 14

4.2.3 การทดลองทางซอฟแวร์

การทดสอบซอฟแวร์เป็นการสังเกตการทำงานและตรวจสอบหาข้อผิดพลาด ซึ่งมีวิธีการทดลองดังนี้

1.เนื่องจากสมการที่ใช้ในการเคลื่อนที่แบบอลวนในโครงงานนี้มี 5 สมการจึงแทนค่าแต่ละสมการด้วยค่า 1 – 3 ดังนี้

1. แทนสมการของเชน

2.แทนสมการของรอสเลอร์

3.แทนสมการของลอเรนซ์

2.ป้อนค่าอินพุต 1ไปให้โปรแกรมทำการประมวลผลสมการของเชน แล้วทำการเขียนกราฟแสดงความสัมพันธ์ของค่า X , Y , Z ในรูปแบบของสมการ 2 มิติ

3.ป้อนค่าอินพุต 2 – 3 เพื่อให้โปรแกรมประมวลผลสมการตามที่กำหนดไว้แล้วนำมาทำการเขียนกราฟด้วยเช่นกัน

4. ป้อนค่าอินพุต 1ไปให้โปรแกรมทำการประมวลผลสมการของเชน แล้วทำการส่งค่า X และ Y ผ่านพอร์ทอนุกรมไปยังตัวหุ่นยนต์

5.ป้อนค่าอินพุต 2 – 3 เพื่อให้โปรแกรมประมวลผลสมการตามที่กำหนดไว้และทำการส่งค่า X และ Y ที่ได้ผ่านพอร์ทอนุกรมไปยังตัวหุ่นยนต์ ตามลำดับ

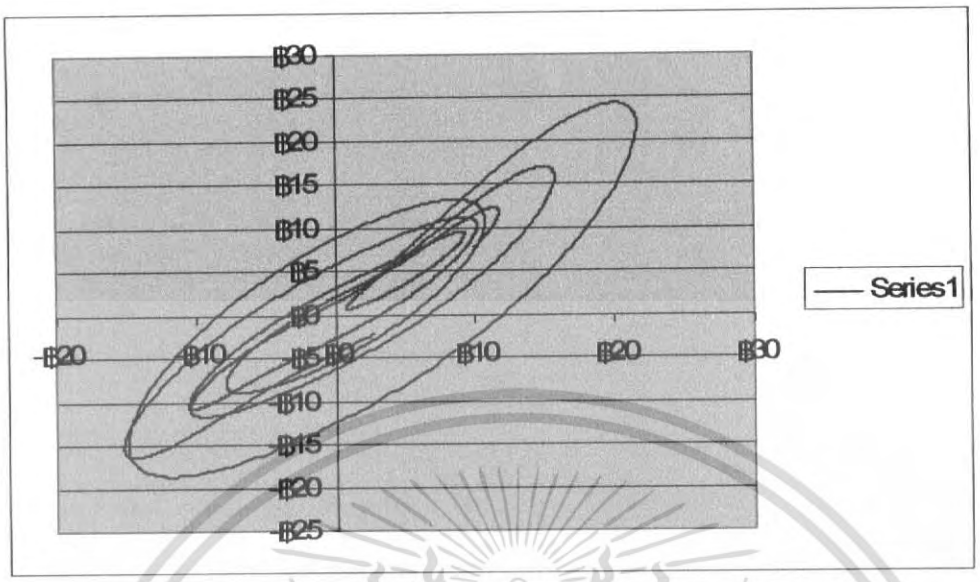
4.ทำการทดลองขั้นที่4 และ 5 ซ้ำอีกครั้งแต่ใช้การส่งค่าผ่านอาร์เอฟโมดูล

5.สังเกตการทำงาน

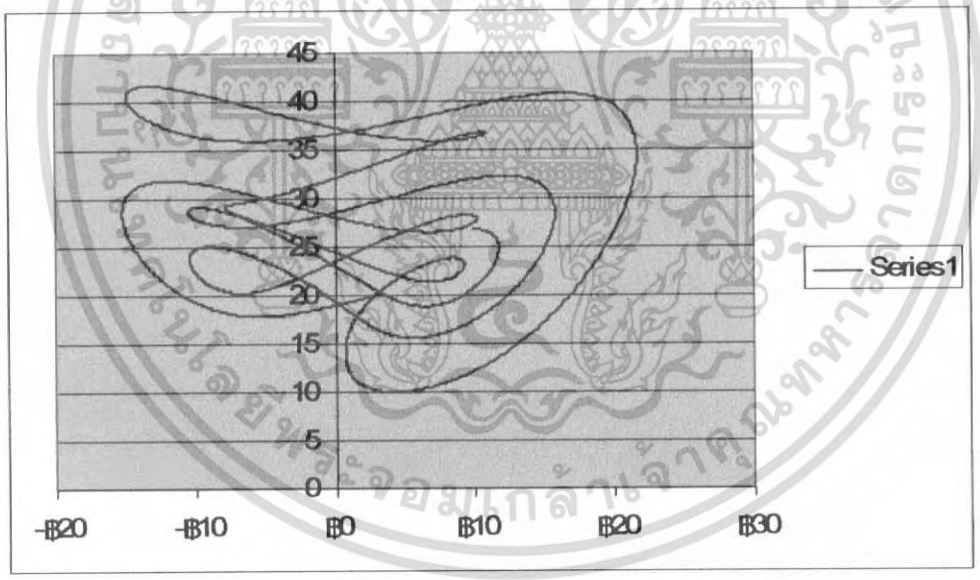
4.2.4 ผลการทดลองทางซอฟแวร์

เมื่อป้อนค่าอินพุต 1 - 3 ให้โปรแกรมทำการประมวลผลและทำการเขียนกราฟจะได้กราฟมีลักษณะดังนี้

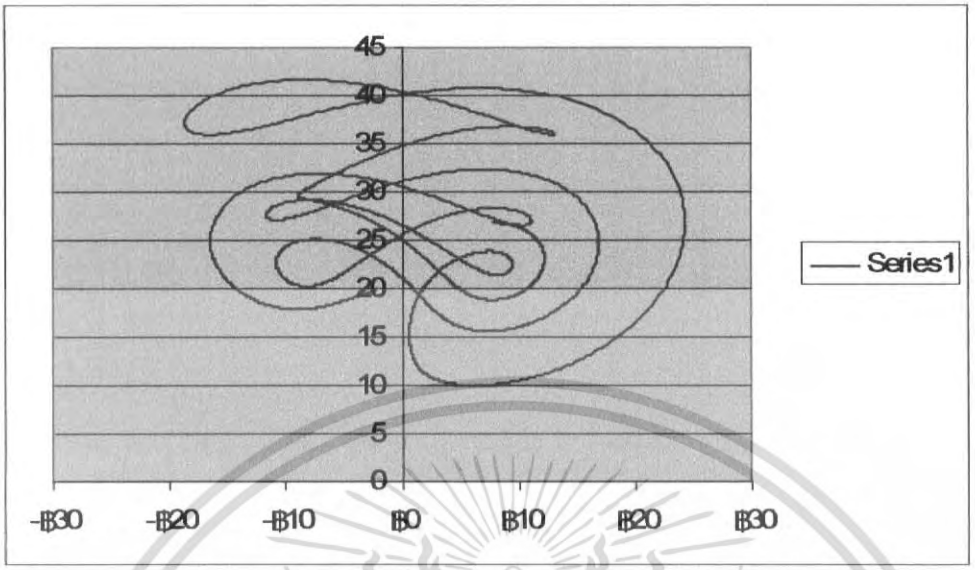
1.กราฟที่เขียนได้จากการประมวลผลสมการของเชน มีลักษณะดังนี้



รูปที่ 4.7 แสดงกราฟความสัมพันธ์ระหว่าง X และ Y ที่เขียนได้จากสมการของเชน

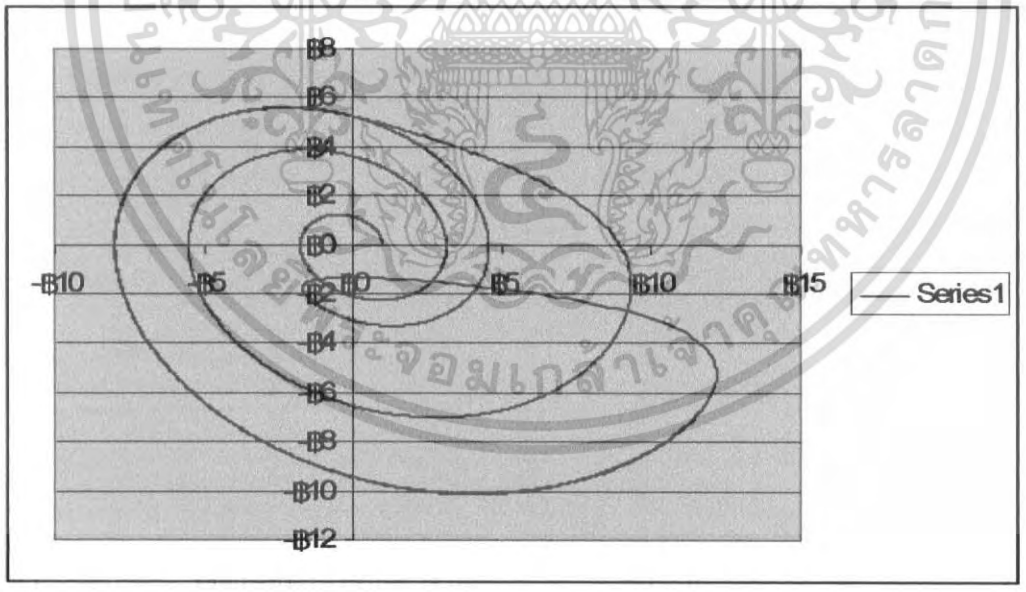


รูปที่ 4.8 แสดงกราฟความสัมพันธ์ระหว่าง X และ Z ที่เขียนได้จากสมการของเชน

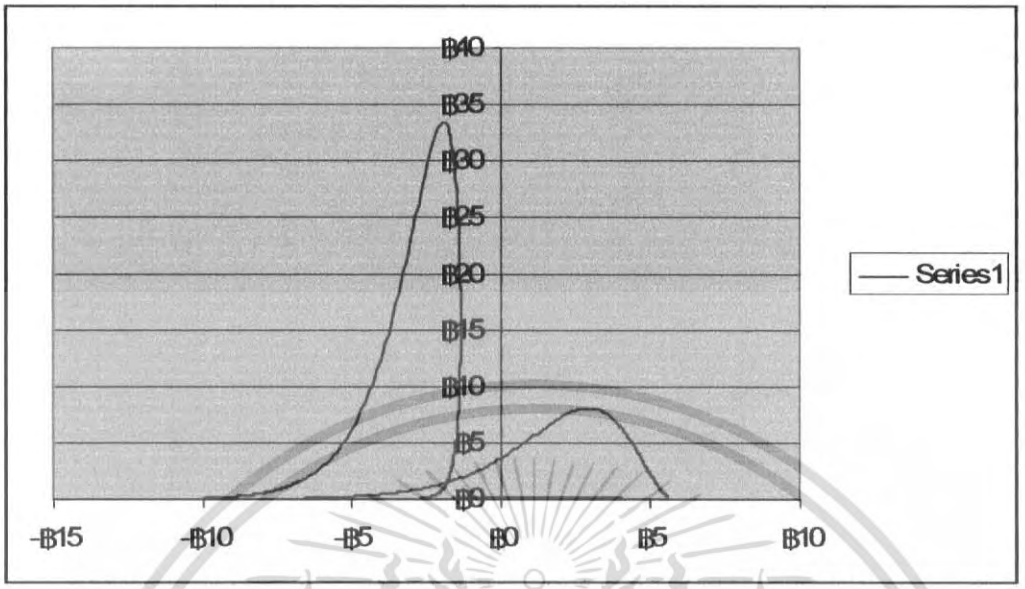


รูปที่ 4.9 แสดงกราฟความสัมพันธ์ระหว่าง Y และ Z ที่เขียนได้จากสมการของเชน

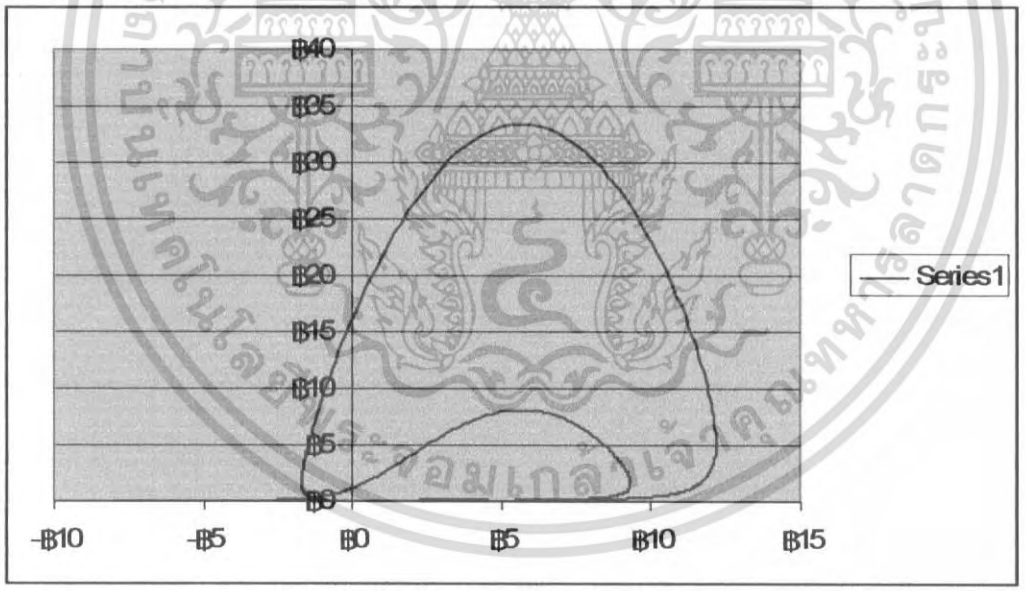
2.กราฟที่เขียนได้จากการประมวลผลสมการของรอสเลอร์ มีลักษณะดังนี้



รูปที่ 4.10 แสดงกราฟความสัมพันธ์ระหว่าง X และ Y ที่เขียนได้จากสมการของรอสเลอร์

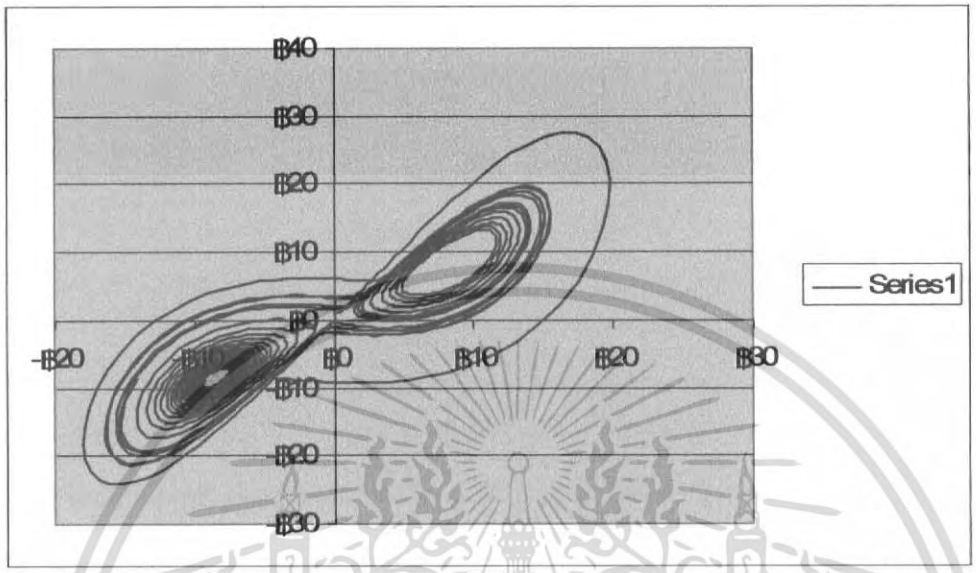


รูปที่ 4.11 แสดงกราฟความสัมพันธ์ระหว่าง Y และ Z ที่เขียนได้จากสมการของรอสเลอร์

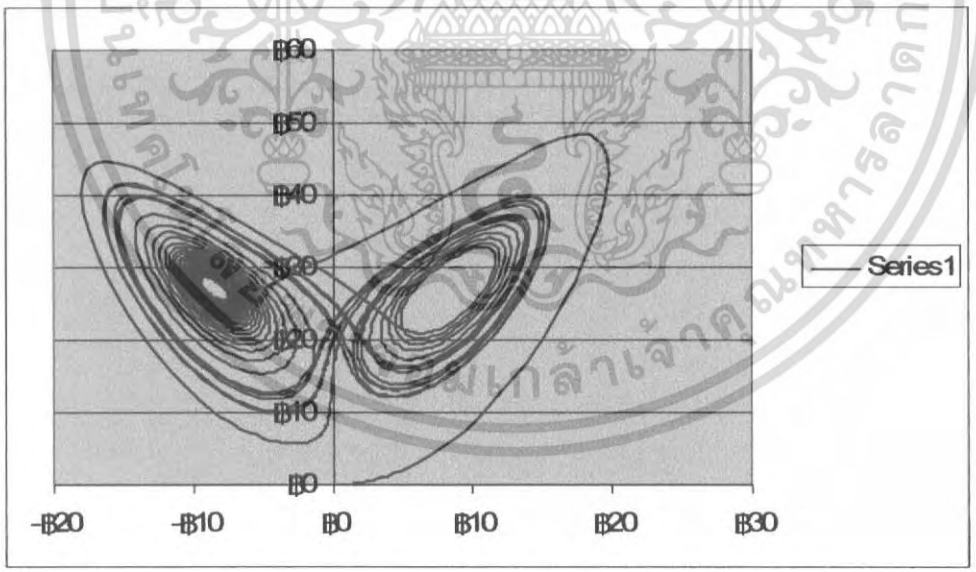


รูปที่ 4.12 แสดงกราฟความสัมพันธ์ระหว่าง X และ Z ที่เขียนได้จากสมการของรอสเลอร์

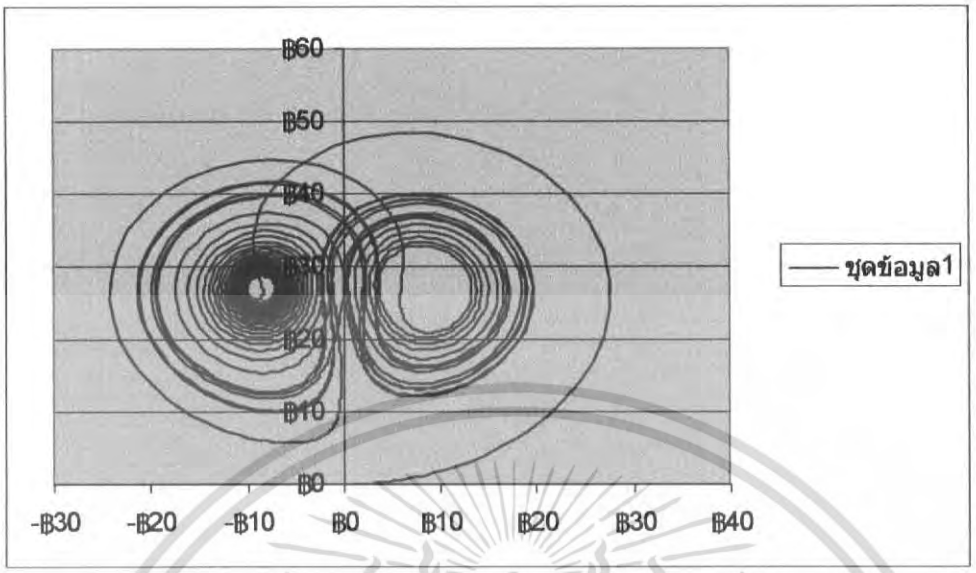
3.กราฟที่เขียนได้จากการประมวลผลสมการของลอเรนซ์ มีลักษณะดังนี้



รูปที่ 4.13 แสดงกราฟความสัมพันธ์ระหว่าง X และ Y ที่เขียนได้จากสมการของลอเรนซ์



รูปที่ 4.14 แสดงกราฟความสัมพันธ์ระหว่าง X และ Z ที่เขียนได้จากสมการของลอเรนซ์



รูปที่ 4.15 แสดงกราฟความสัมพันธ์ระหว่าง Y และ Z ที่เขียนได้จากสมการของลอเรนซ์

4.3 การทดลองการควบคุมหุ่นยนต์ด้วยเสียง

การทดลงในส่วนนี้จะเป็นส่วนของการสั่งให้มีการเคลื่อนที่แบบอลวนทั้ง 3 แบบเป็นภาษาอังกฤษ

4.3.1 การทดลองการควบคุมการเคลื่อนที่แบอลวนของหุ่นยนต์ด้วยเสียง

ในการทดลองนี้ใช้ผู้ทดลองจำนวน 10 คน เช่นเดียวกัน โดยมีขั้นตอนการทดลองดังนี้

1.ให้ผู้ทดลองคนที่ 1 สั่งให้หุ่นยนต์เคลื่อนที่แบบอลวนโดยใช้คำสั่ง Chen , Lorenz , Rossler เพื่อให้หุ่นยนต์เคลื่อนที่ในรูปแบบที่แตกต่างกันเพื่อบันทึกตัวอย่างความถี่เสียงเก็บไว้ในฐานข้อมูล

2.ให้ผู้ทดลองคนที่ 2 – 10 ออกคำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวนดังคำสั่งในข้อที่ 1

3.ให้ผู้ทดลองทั้ง 10 คน ทำการทดลองซ้ำอีกคนละ 5 ครั้ง

4.สังเกตและบันทึกผลการทดลอง

4.3.2 ผลการทดลองการควบคุมการเคลื่อนที่แบบอลวนของหุ่นยนต์ด้วยเสียง

การทดลองครั้งที่ 1

ผู้ทดลอง	Chen	Lorenz	Rossler
A	X	X	/
B	X	/	/
C	/	/	/
D	/	X	X
E	/	X	X
F	X	/	X
G	X	X	/
H	X	X	X
I	/	/	/
J	/	/	X

ตารางที่ 4.1 ตารางการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนในลักษณะต่างๆตามเสียงที่สั่งครั้งที่ 1

คำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวน	Chen	Lorenz	Rossler
ความถูกต้อง (%)	50	50	5o

ตารางที่ 4.2 ตารางแสดงเปอร์เซ็นต์ความถูกต้องในการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนตามเสียงที่สั่งครั้งที่ 1

การทดลองครั้งที่ 2

ผู้ทดลอง	Chen	Lorenz	Rossler
A	/	/	/
B	/	/	/
C	/	X	/
D	X	/	/
E	X	/	/
F	X	/	X
G	/	X	X
H	/	X	/
I	/	/	x
J	/	/	/

ตารางที่ 4.3 ตารางการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนในลักษณะต่างๆตามเสียงที่สั่งครั้งที่ 2

คำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวน	Chen	Lorenz	Rossler
ความถูกต้อง (%)	70	70	70

ตารางที่ 4.4 ตารางแสดงเปอร์เซ็นต์ความถูกต้องในการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนตามเสียงที่สั่งครั้งที่ 2

การทดลองครั้งที่ 3

ผู้ทดลอง	Chen	Lorenz	Rossler
A	/	/	/
B	/	/	/
C	/	X	/
D	/	/	/
E	X	X	X
F	X	/	/
G	/	X	/
H	X	X	/
I	/	/	/
J	/	/	X

ตารางที่ 4.5 ตารางการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนในลักษณะต่างๆตามเสียงที่สั่งครั้งที่ 3

คำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวน	Chen	Lorenz	Rossler
ความถูกต้อง (%)	70	60	80

ตารางที่ 4.6 ตารางแสดงเปอร์เซ็นต์ความถูกต้องในการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนตามเสียงที่สั่งครั้งที่ 3

การทดลองครั้งที่ 4

ผู้ทดลอง	Chen	Lorenz	Rossler
A	/	/	/
B	/	X	X
C	/	/	/
D	X	/	/
E	/	/	X
F	/	X	/
G	/	/	/
H	X	X	/
I	/	/	/
J	/	/	/

ตารางที่ 4.7 ตารางการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนในลักษณะต่างๆตามเสียงที่สั่งครั้งที่ 4

คำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวน	Chen	Lorenz	Rossler
ความถูกต้อง (%)	80	70	80

ตารางที่ 4.8ตารางแสดงเปอร์เซ็นต์ความถูกต้องในการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนตามเสียงที่สั่งครั้งที่ 4

การทดลองครั้งที่ 5

ผู้ทดลอง	Chen	Lorenz	Rossler
A	/	/	/
B	/	/	X
C	/	X	/
D	/	/	/
E	X	X	X
F	/	/	/
G	/	/	/
H	/	/	/
I	/	/	/
J	/	X	/

ตารางที่ 4.9 ตารางการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนในลักษณะต่างๆตามเสียงที่สั่งครั้งที่ 5

คำสั่งให้หุ่นยนต์เคลื่อนที่แบบอลวน	Chen	Lorenz	Rossler
ความถูกต้อง (%)	90	70	80

ตารางที่ 4.10 ตารางแสดงเปอร์เซ็นต์ความถูกต้องในการทดลองให้หุ่นยนต์เคลื่อนที่แบบอลวนตามเสียงที่สั่งครั้งที่ 5

บทที่ 5
สรุปผลการดำเนินงาน

5.1 สรุปผลการดำเนินงาน

โครงงานนี้ได้จัดทำขึ้นเพื่อศึกษาแนวทางในการพัฒนาหุ่นยนต์ให้มีประสิทธิภาพ โดยการนำความรู้เก่าที่มีมาในเรื่องการเคลื่อนที่ของหุ่นยนต์แบบอลวนนำมาศึกษาและทำการปรับปรุง โดยการเปลี่ยนระบบควบคุมการเคลื่อนที่แบบอลวนจากการใช้วงจรควบคุมมาเป็นการควบคุม โดยใช้โปรแกรมประมวลผลบนคอมพิวเตอร์แล้วส่งผ่านอาร์เอฟโมดูล ซึ่งใช้ในการส่งข้อมูลแบบไร้สายไปยังหุ่นยนต์ และนำการควบคุมการเคลื่อนที่ด้วยเสียงมาช่วยเพื่อให้หุ่นยนต์สามารถตอบสนองได้ตรงตามความต้องการมากยิ่งขึ้น

จากการดำเนินงานพบว่า หุ่นยนต์สามารถเคลื่อนที่ตามเสียงคำสั่งโดยการให้เปลี่ยนลักษณะการเคลื่อนที่แบบอลวนแบบต่างๆได้เป็นที่น่าพอใจซึ่งแสดงให้เห็นว่าหุ่นยนต์มีความสามารถในการเรียนรู้และจดจำคำสั่ง และในส่วนของการเคลื่อนที่แบบอลวนหุ่นยนต์สามารถเคลื่อนที่ได้ถูกต้องตรงตามรูปแบบเป็นที่น่าพอใจ และยังแสดงให้เห็นว่าหุ่นยนต์ยังขนาดพื้นที่ที่ใช้เป็นเส้นทางเคลื่อนที่ผ่านมากกว่าหุ่นยนต์ที่เคลื่อนที่แบบปกติ ทั้งยังทำให้เห็นถึงประโยชน์ของการเคลื่อนที่แบบอลวนที่จะสามารถนำมาพัฒนาต่อให้กับหุ่นยนต์ที่ต้องปฏิบัติภารกิจที่จำเป็นที่จะต้องมีความละเอียดในเรื่องของขนาดพื้นที่ เช่น หุ่นยนต์สำรวจกับระเบิด หุ่นยนต์ทำความสะอาด เป็นต้น

5.2 ปัญหาที่เกิดขึ้นกับโครงงาน

1.การใช้เสียงในการควบคุมการทำงานของหุ่นยนต์เป็นเรื่องที่กระทำได้ยากเนื่อง จากการควบคุมด้วยเสียงจะอาศัยหลักการของความถี่เสียง ซึ่งแต่ละบุคคลก็มีความถี่แตกต่างกันในการออกเสียงคำสั่งคำเดียวกัน จึงเป็นการยากที่โปรแกรมจะตรวจจับความถี่ได้ว่าเสียงที่สั่งออกมาเป็นคำสั่งอะไร ทำให้ผลการทำงานยังไม่เต็มประสิทธิภาพเท่าที่ควรและยังมีการจำเพาะบุคคล

2.ไมโครโฟนที่ใช้นั้นต้องมีความไวพอสมควรและไม่ก่อให้เกิดเสียงรบกวนมากเกินไป

3.ในการเคลื่อนที่แบบอลวนจำเป็นต้องใช้หุ่นยนต์ที่มีการแยกความเร็วในการหมุนของล้อซ้ายกับล้อขวาอย่างชัดเจน จึงต้องมีการใช้มอเตอร์ขับเคลื่อน 2 ตัวในการแยกการทำงานจึงทำหุ่นยนต์ที่ออกแบบไว้ในตอนแรกใช้ไม่ได้

5.3 ข้อจำกัดของโครงงาน

1.การใช้งานในส่วนของการควบคุมด้วยเสียงยังมีข้อจำกัดในเรื่องของจำนวนบุคคลที่สามารถออกคำสั่งให้หุ่นยนต์ตอบสนองยังทำงานได้ไม่เต็มประสิทธิภาพ

2.คำสั่งที่ใช้ในการควบคุมหุ่นยนต์มีน้อยคือมีเพียง 3 คำสั่งและเป็นภาษาอังกฤษ

3.การเคลื่อนที่แบบอลวนสำหรับหุ่นยนต์ที่พัฒนาขึ้นในโครงงานต้องใช้งานในพื้นที่โล่งเท่านั้น เนื่องจากไม่มีการติดเซ็นเซอร์เพื่อตรวจจับสิ่งกีดขวางที่ตัวหุ่นยนต์

5.4 แนวทางในการพัฒนาต่อ

1.สามารถพัฒนาต่อในส่วนของจำนวนคำสั่งที่สามารถพัฒนาเป็นภาษาไทยและเพิ่มปริมาณคำสั่งเพื่อให้หุ่นยนต์ตอบสนองได้ตรงกับความต้องการมากยิ่งขึ้น

2.สามารถนำระบบนี้ไปใช้ในเครื่องอำนวยความสะดวก เช่น เครื่องดูดฝุ่น ซึ่งทำให้เครื่องดูดฝุ่นสามารถทำงานได้ด้วยตัวเองและทำความสะอาดได้ทั่วถึงมากกว่าหุ่นยนต์ทั่วไป

3.สามารถนำไปดิดตัวเซ็นเซอร์เพื่อให้หุ่นยนต์มีความสามารถในด้านการหลบหลีกสิ่งกีดขวางได้

บรรณานุกรม

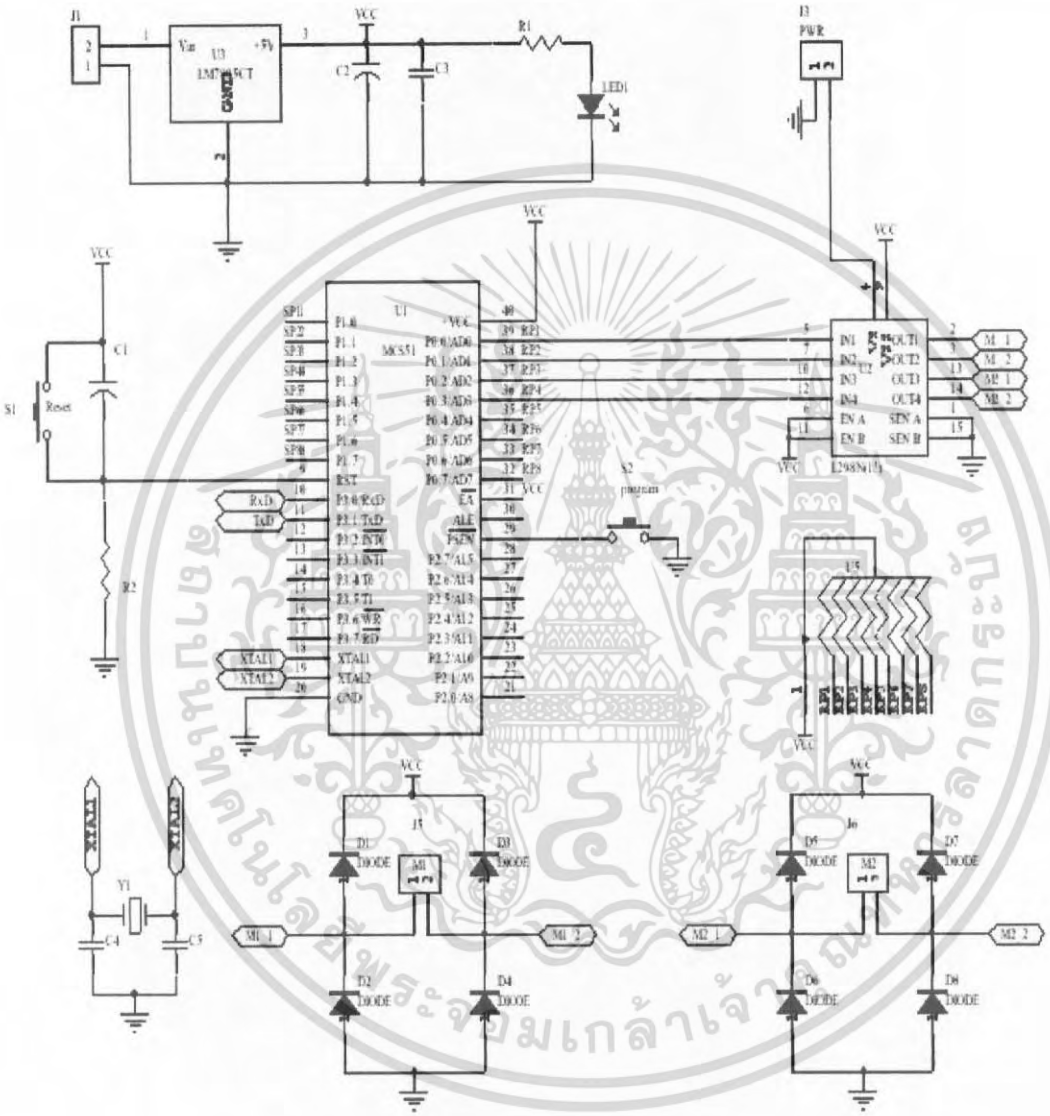
[1] Anurak Jansri , Kitdakorn Klomkarn and Pitikhate Sooraksa , **"On Comparison of Attractors for Chaotic Mobile Robots"** , King Mongkut's Institute of Technology Ladkrabang

[2] Bekey, George A , "**Autonomous robots : from biological inspiration to implementation and control** " , Cambridge, MA : Massachusetts Institute of Technology, c2005

[3] ธีรวัฒน์ ประกอบผล , **"การประยุกต์ใช้งานไมโครคอนโทรลเลอร์"** , สาคมส่งเสริมเทคโนโลยี (ไทย – ญี่ปุ่น) , 2543

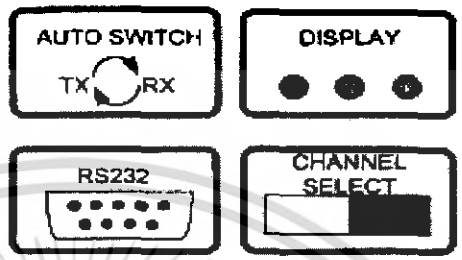
[4] ฉันททวุฒิ พืชผล , **"คู่มือเรียน Visual Basic 6"** , บริษัทโปรวิชั่น , 2547

ภาคผนวก



RDL401A Series

Information

Machine type	Function
 RDL401A	

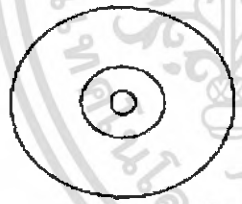
Standard Package

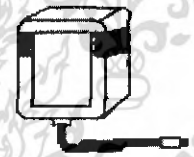

RDL401A


WAS-1297


CN5-OTT0006

Optional


DISK5146

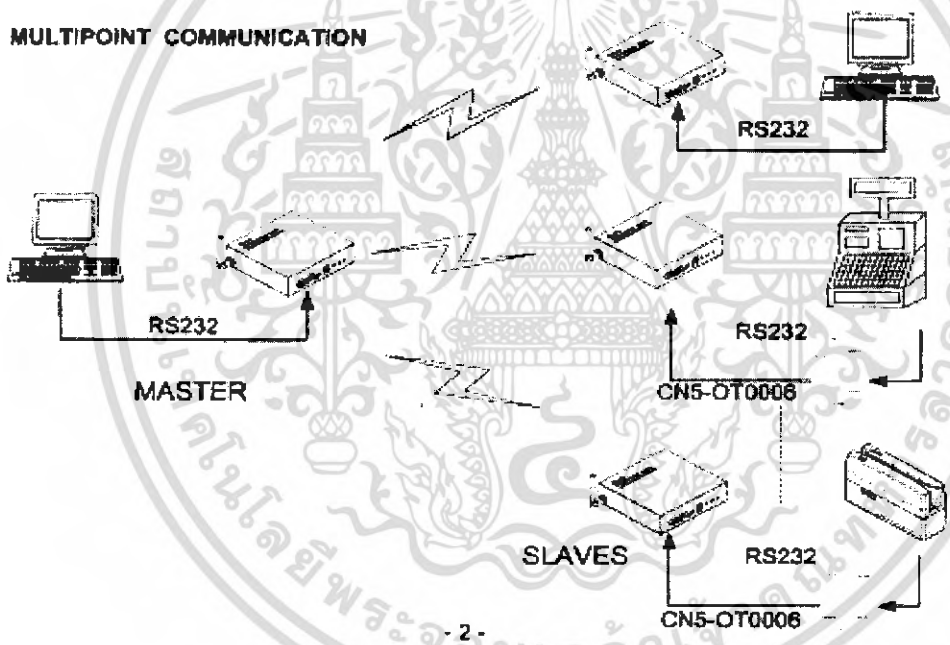

DC9V/120VAC
DC9V/230VAC

Application

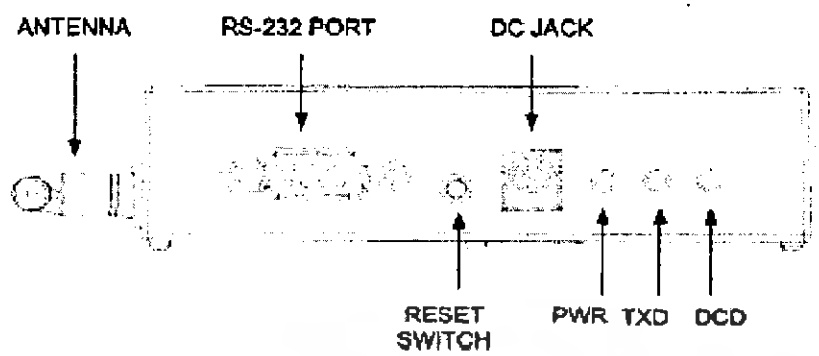
- To replace the RS-232 cable between Computer and Computer (Terminal) .
- Remote control
- Remote monitoring
- Data acquisition
- Computer aided integrated manufacturing
- Point-of-sale system
- Data transmission of movable station

POINT TO POINT COMMUNICATION

MULTIPOINT COMMUNICATION



Configuration



CHANNEL Select Switch :

Switch 1 : ON : 433.92MHz
OFF : 434.33MHz
Switch 2 : ON : 4800 BPS Disable
OFF : 4800 BPS Enable
Switch 3 : ON : 9600 BPS Enable
OFF : 19200 BPS Enable
Switch 4 : Reserved

RS-232 Port -- To be connected with the Computer (Terminal) .

REST Switch -- Press this REST switch to reset radio Channel to the new setting . Before pressing this switch , the radio Channel will not change although the CHANNEL Select Switch has been changed. (The CHANNEL Select Switch is located on the bottom of the RDL401A unit)

Power Jack -- For DC 9V power input .

PWR Power indicator -- Turn on when the power is applied .

TXD Indicator -- Green LED turns on when the data is sent from the connected Computer (Terminal) to RDL401A unit.

DCD Indicator -- Green LED turns on when it detects a radio transmission from the air (Data Carrier Detect).

CHANNEL Select Switch -- DIP switch located on the bottom of the RDL401A unit is for changing radio channel 433.92MHz or 434.33MHz. After you make Channel change , you must press RESET switch or power off and then reboot power . Then the RDL401A will be reset to the new radio Channel .

Installation

1. Install the RS-232 cable between the RDL401A and Computer or Terminal .
   (The pin connection is as PIN ASSIGNMENT)

2. Set up one pair of RDL401 , one for Master Computer (terminal) and the other for Remote Computer (terminal) . Set both RDL401A units at the same channel .

3. Connect the DC power supply to the DC Power Jack on RDL401A unit , the PWR LED will turn on.

4. The DCD LED must light green constantly for the receiving unit when detecting a radio transmission from the other unit . If not ,reinstall them to have a constant green DCD light for the receiving unit .
   Please note that you must not have both units become transmitting at the same time because RDL401A is for half-duplex communication only . At one time you must have one unit become transmitting and the other unit become receiving .

Note :

To get connection with RDL401A , you must set Computer (terminal) as the following parameters under the operation software you are using :

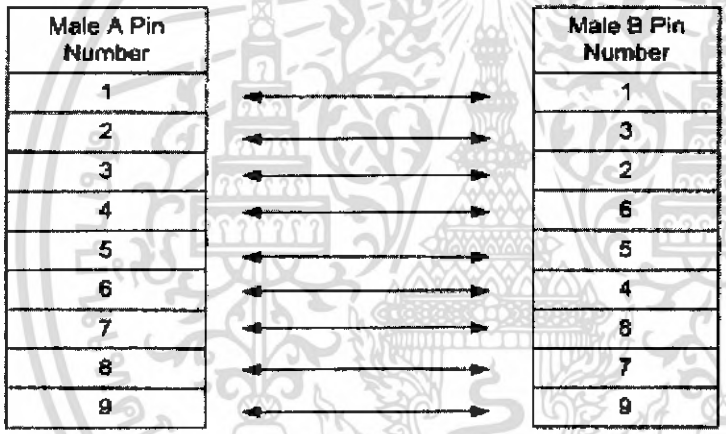
Standard :	Standard RS-232
Baud Rate :	4800/9600/19200 bps
Parity :	None
Data Bits :	8
Stop BIT :	1
Flow control :	None

PIN ASSIGNMENT

9 Pin D Female Connect

Pin Number	Signal
1	X
2	TXD (Out)
3	RXD (In)
4	X
5	Ground
6	X
7	X
8	RTS (Out)
9	X

CN5-OTT0006 Connect



Specification

Frequency :	433.92/434.33MHz
Antenna Impedance :	50 Ohms , unbalanced
Number of Channels :	2 (Frequency Synthesized]
Frequency Stability :	+/- 10 ppm (at 0~60 Deg. C)
Data Rate :	4800/9600/19200 Baud
Data format :	None Parity , 8 Data Bits , 1 Stop Bit , Half-duplex over a single channel Asynchronous , Serial
Communication Distance :	Approx. 50~150 meters (prospective)
Power Supply :	9 VDC
Environment :	Operating Temperature 0 ~+60 Deg. C Storage Temperature -10 ~+70 Deg. C Humidity : 10 % ~ 95 % , non-condensing
Dimensions :	W 107 * D 102 * H 26 mm
Weight :	APPROX. 0.38 KGs
Transmitter Output Power :	5mW maximum
Modulation Deviation :	35 +/- 5 KHz
PLL Programming Time :	5 m sec , maximum
Current Drain :	60 mA maximun at 9 VDC nominal
Sensitivty :	- 102 dBm
Selectivity :	85 KHz maximum
Receiver Current :	Typical 27 mA

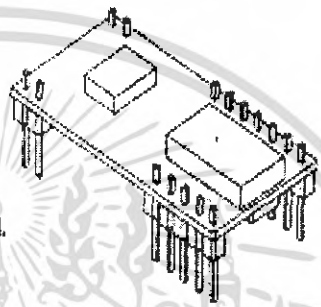
Radio Data Link RF Module with UART Interface

INTRODUCTIONS:

RDL200T (Radio Data Link) RF Module is designed for wireless RS232 serial data communication. It is suitable for most of your wireless data link applications. RDL200T RF Module can be built into your equipment through RS232 interface to replace the cable to become wireless communication. It is a low cost and high quality radio data communication solution.

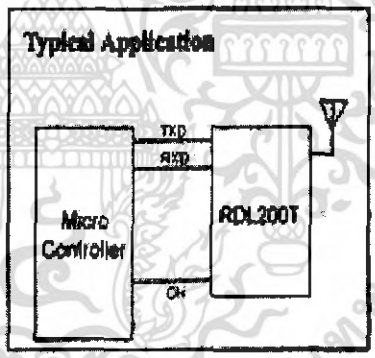
FEATURES:

1. ISM frequency 433.92MHz / 434.33MHz
2. Two channel selectable.
3. Low power consumption with 3.3Vdc
4. RS232 interface (Serial TTL level)
5. Speed up to 9600bps with automatic flow control.
6. Half-duplex with automatic Receiving/Transmitting control.
7. Small package size.
8. Distance: 50~150 meters with appropriate antenna



APPLICATION:

1. Hand-held terminals
2. Computer to computer
3. Computer to terminal
4. Data collection terminal
5. Mobile-around data communications
6. Remote control
7. Remote monitoring
8. POS systems.



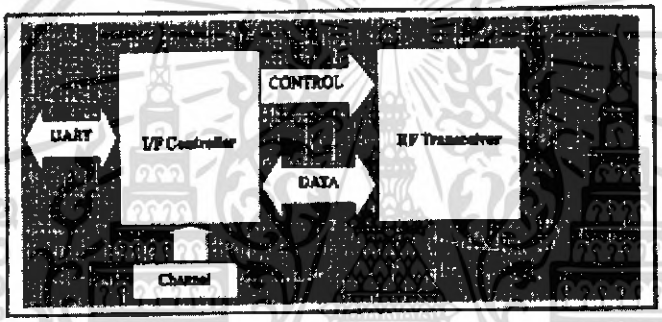
SPECIFICATIONS:

1. Frequency: CH1 433.92MHz / CH2 434.33MHz
2. Modulation: FSK
3. Output power: 7dbm (5mw)
4. Receiving Sensibility: -102dbm
5. Operating voltages: 2.7Vdc~3.3Vdc
6. Power consumption: Tx:80mA / Rx:30mA
7. Interface : RS232 (TTL Level: TXD/RXD)
8. Data Rate : 19200bps/9600bps / 4800bps , Half-duplex
9. Operating temperature: 0°C~55°C (Storage Temperature: -10°C ~60°C)

QUICK REFERENCE DATA

Parameter	Value	Unit
Frequency, Channel#1/Channel#2	433.92/434.33	MHz
Modulation	FSK	
Frequency deviation	+/-30	KHz
Max. RF output power	7	dBm
Sensitivity	-102	dBm
UART Baudrate	19200/9600/4800	bps
Supply voltage	2.7 – 3.3	Vdc
Max. Receive current	30	mA
Max. Transmit current	60	mA

BLOCK DIAGRAM



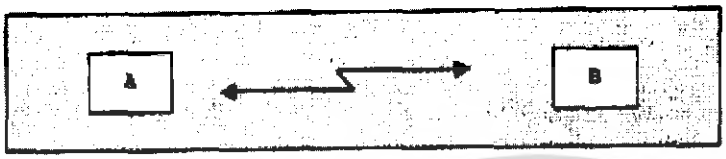
I/F Controller: UART and Radio interface unit

RF Transceiver: A long-range radio transceiver for wireless links operating in the globally available ISM band.

RDL200T APPLICATION

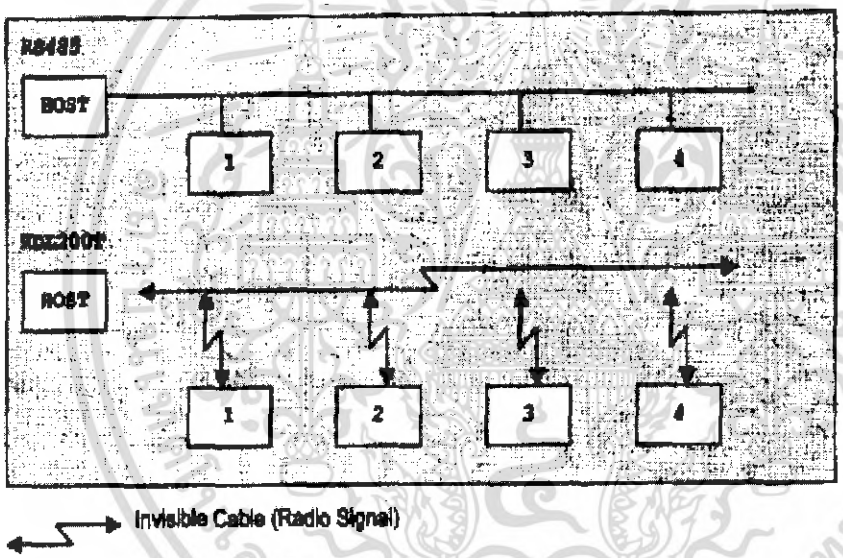
1. Point to Point Radio Data Link

   For point to point radio data link, just change your RS232 transceiver IC to RDL200T in your RS232 product. You don't need to change any firmware. RDL200T serves as a invisible RS232 cable in your product.

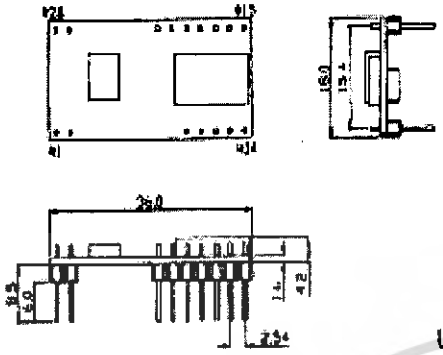


2. Multi-Station Radio Data Link

   For Multi-Station radio data link, you have to build a protocol for Multi-Station that's similar to RS485 to assign machine ID for each station.



PACKAGE OUTLINE (DIP28)



Unit: mm

PIN ASSIGNMENT

PIN	NAME	I/O	DESCRIPTION
1	ANT	I	*Antenna
2	ANTG	I	*Antenna Signal GND
3,4,5,6,7,8,9	NC		No Connect
10	/RST	I	Active Low Reset
11	RXD	I	UART Data Input
12	TXD	O	UART Data Output
13	GND	I	GND
14	TRD/CMD		Reserved
15	BPS0		4800bps 0:disable 1:enable
16	BPS1		0:9600bps 1:19200bps(*short data pack)
17	CH	I	0:433.92MHz 1:434.33MHz
18	RLED	O	RXD Status LED
19	TLED	O	TXD Status LED
20	GND		GND
21	VCC		Power Input +3.3Vdc~2.7Vdc
22,23,24,25,26	NC		No Connect
27	GND		GND
28	GND		GND

***Antenna Signal GND: Connect to antenna cable GND.**

***Antenna : 50 ohm / 430~435MHz / -25db.**

***19200bps for short data pack (max 70 bytes) only.**

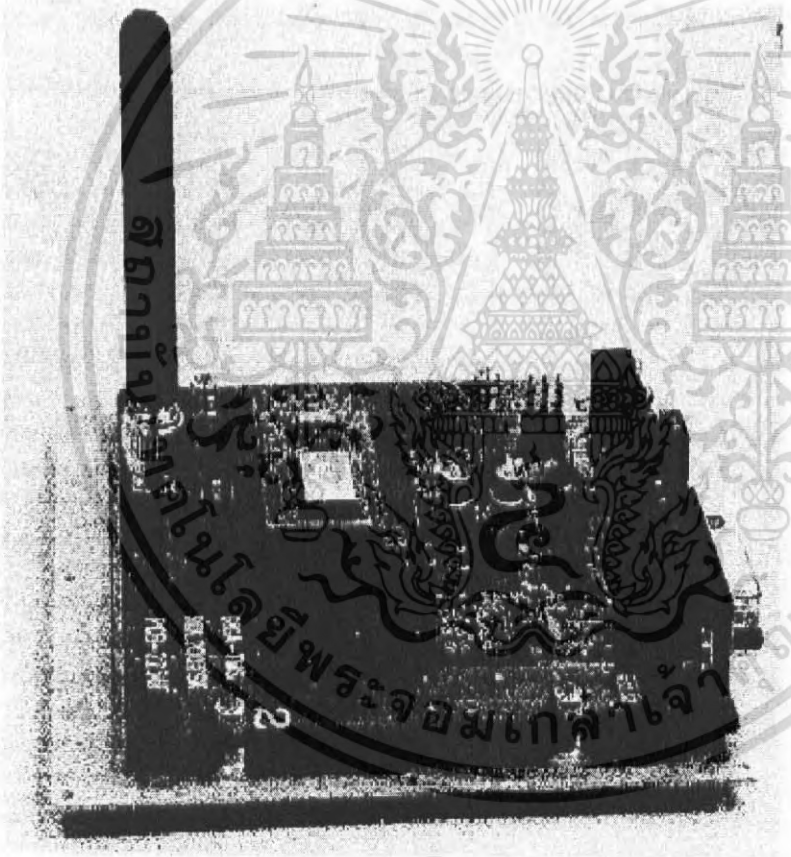
PROMAG

RDL200T

REV.B

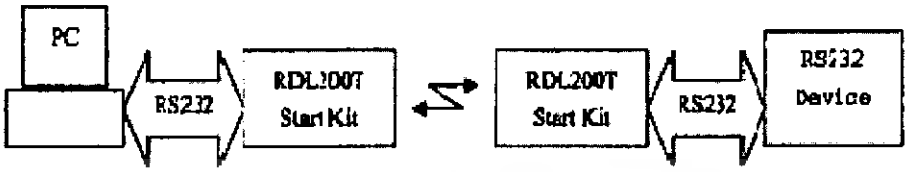
RDL200T START KIT

The RDL200TSK Start Kit includes the RDL200T module and RS232 Transceiver IC. You can easily learn and develop your product for wireless communication application.



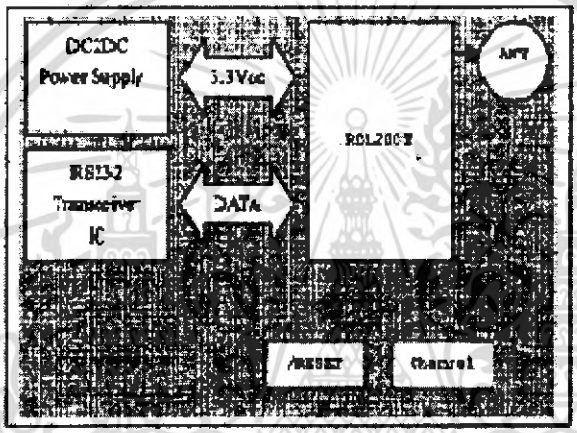
START KIT PACKAGE APPLICATION

Example: Point to Point radio data link

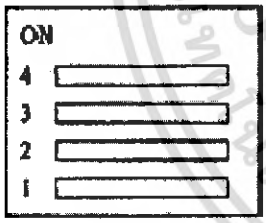


Keep your communication way and easy to upgrade your product to become wireless product with RDL200T

ASY-RDL200TSK BLOCK DIAGRAM

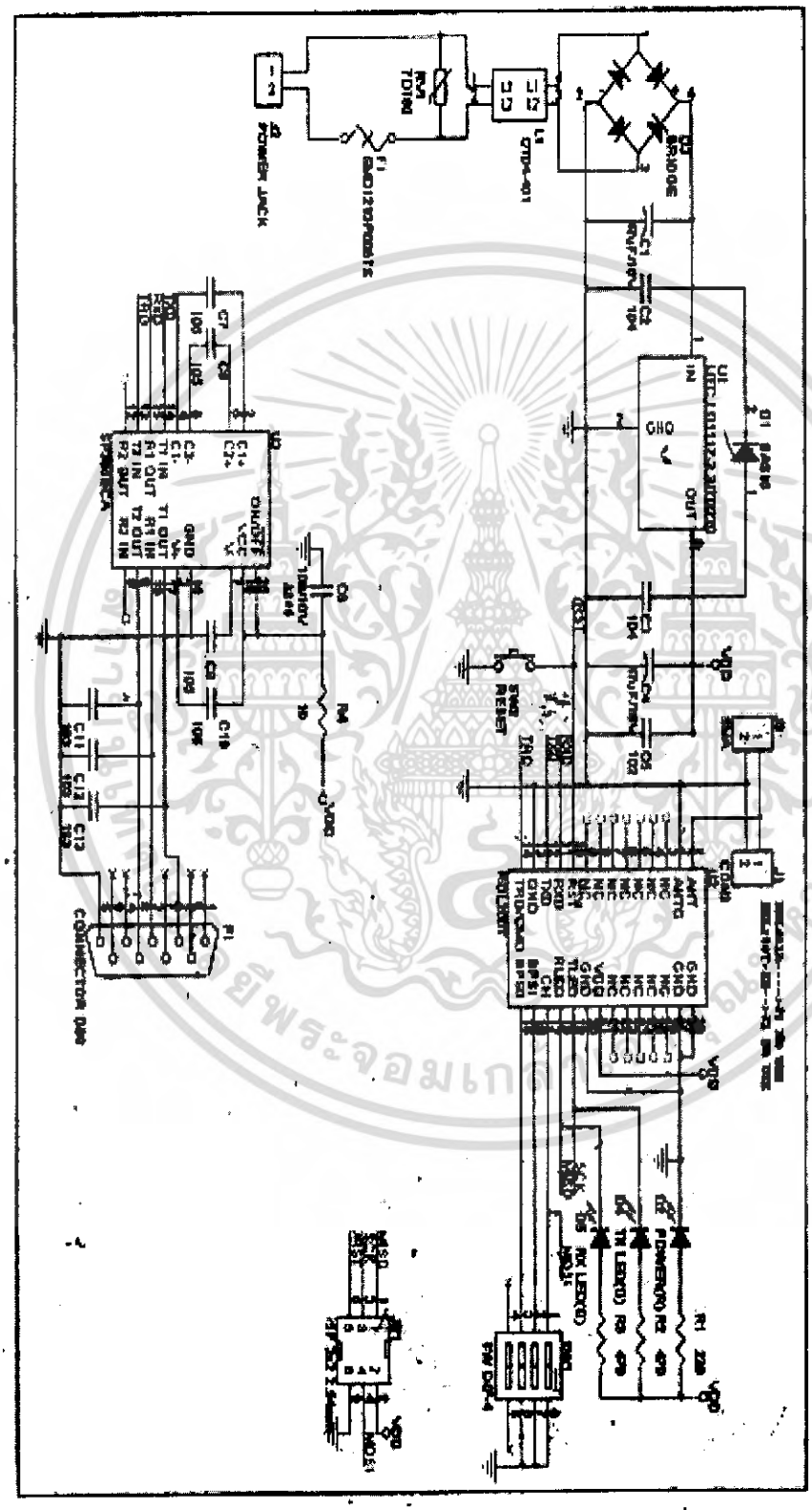


DIP SWITCH



1. ON : CH1(433.92MHz) / OFF:CH2(434.33MHz)
2. ON : 4800 bps DISABLE / OFF : 4800 bps ENABLE
3. ON : 9600 bps ENABLE / OFF : 19200 bps ENABLE
4. Reserved

ASY-RDL200TSK CIRCUIT DIAGRAM



ASY-RDL200T5K BILL OF MATERIALS

Part Number	Description	Location	QTY
UTCLD1117-3.3J	3.3Vdc DC2DC Power supply	U1	1
*[illegible]	[illegible]	U2	
SP385ACA	RS232 Transceiver IC	U3	1
BAS16	DIODE	D1	1
DBSJ04G	DIODE BRIDGE SMT DBS104G	D3	
VD180	Variable Resister	RV1	1
CEC45-476016W	47UF/16V	C1,C4	2
CMC5X1C2K50	1000PF/50V	C5	1
CMC5X1C3K50	0.01UF/50V	C11,C12,C13	3
C2012X7R1H104K	0.1UF/50V	C2,C3	2
CMC5X105KCH	1UF/25V	C7,C8,C9,C10	4
CMC6Y106Z16	10UF/25V	C6	1
0805-5-470R	470 ohm	R2,R3	2
0805-5-220R	220 ohm	R1	1
0805-5-10R	10 ohm	R4	1
*[illegible]	[illegible]	[illegible]	
*[illegible]	[illegible]		

*PROVIDED BY GIGA-TMS INC.

ORDERING INFORMATION

RDL200T-00: (Radio Data Link Module)
P/N: RDL200T-00

RDL200T Start Kit Package: (ASY-RDL200TSK x 2PCS)
P/N: RDL200T-SKE

E: Include 230VAC/9VDC Adaptor
U: Include 120VAC/9VDC Adaptor

RDL401A-00 (Stand-Alone Radio Data Link)
P/N: RDL401AE

E: Include 230VAC/9VDC Adaptor
U: Include 120VAC/9VDC Adaptor

OPTIONAL ANTENNA SOCKET AND ANTENNA

1. Antenna Socket

   P/N: **CN5-OTG0007** (Angle 180)

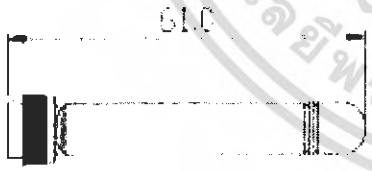
   P/N: **CN5-OTG0008** (Angle 90)

2. Antenna Extension Cable

   P/N: **WAS-1529**

3. Antenna

   P/N: **ANT-T001** (50 ohms, 434+-5 MHz)



DATA SHEET

89C51RB2/89C51RC2/89C51RD2

80C51 8-bit Flash microcontroller family

16KB/32KB/64KB ISP/IAP Flash with 512B/512B/1KB RAM

Preliminary specification

1999 Sep 23

IC28 Data Handbook

Philips
Semiconductors

PHILIPS

DESCRIPTION

The 89C51RB2/RC2/RD2 device contains a non-volatile 16kB/32kB/64kB Flash program memory that is both parallel programmable and serial In-System and In-Application Programmable. In-System Programming (ISP) allows the user to download new code while the microcontroller sits in the application. In-Application Programming (IAP) means that the microcontroller fetches new program code and reprograms itself while in the system. This allows for remote programming over a modem link. A default serial loader (boot loader) program in ROM allows serial In-System programming of the Flash memory via the UART without the need for a loader in the Flash code. For In-Application Programming, the user program erases and reprograms the Flash memory by use of standard routines contained in ROM.

This device executes one machine cycle in 6 clock cycles, hence providing twice the speed of a conventional 80C51. An OTP configuration bit lets the user select conventional 12 clock timing if desired.

This device is a Single-Chip 8-Bit Microcontroller manufactured in advanced CMOS process and is a derivative of the 80C51 microcontroller family. The instruction set is 100% compatible with the 80C51 instruction set.

The device also has four 8-bit I/O ports, three 16-bit timer/event counters, a multi-source, four-priority-level, nested interrupt structure, an enhanced UART and on-chip oscillator and timing circuits.

The added features of the P89C51RB2/RC2/RD2 makes it a powerful microcontroller for applications that require pulse width modulation, high-speed I/O and up/down counting capabilities such as motor control.

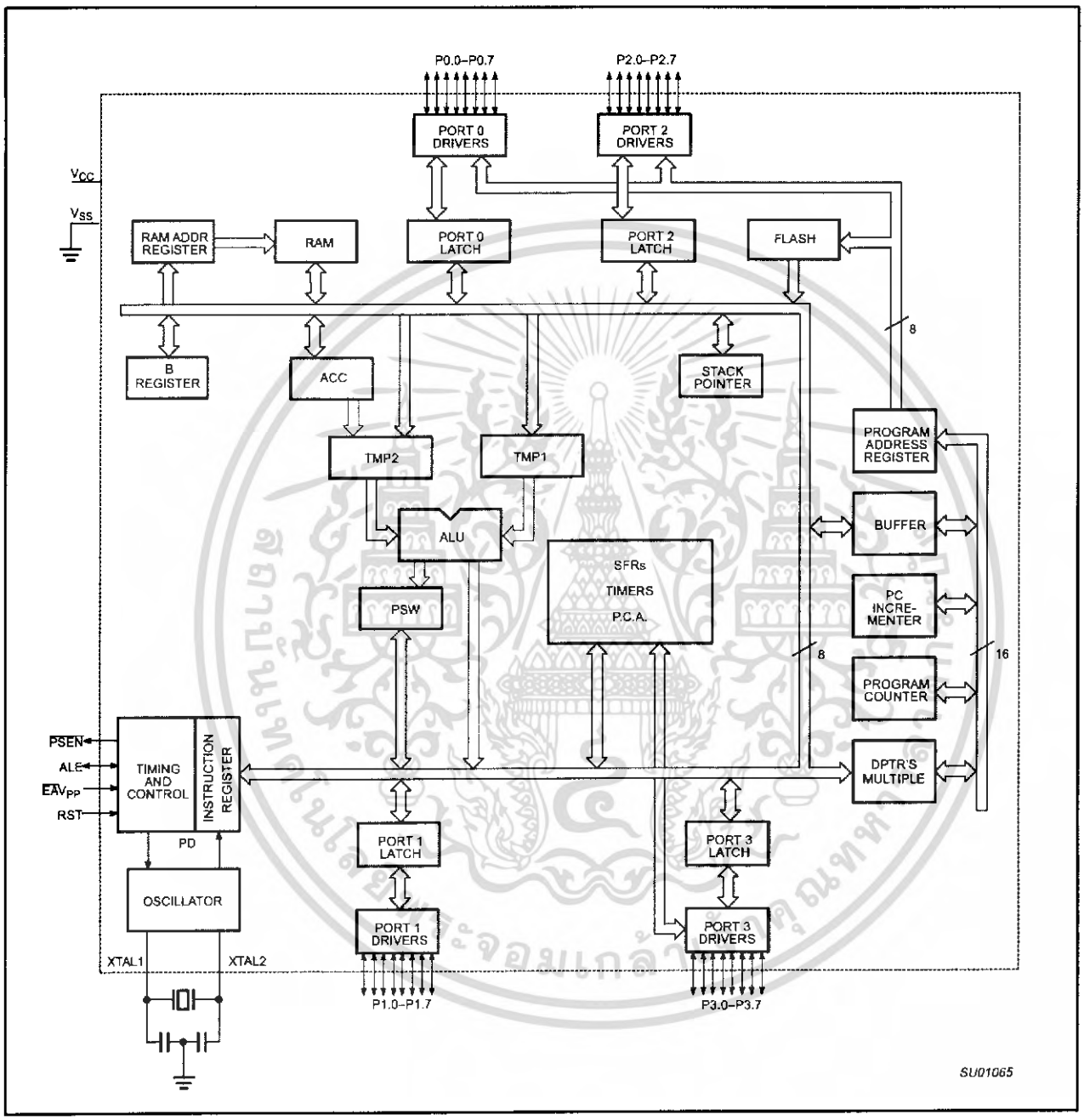
FEATURES

- 80C51 Central Processing Unit
- On-chip Flash Program Memory with In-System Programming (ISP) and In-Application Programming (IAP) capability
- Boot ROM contains low level Flash programming routines for downloading via the UART
- Can be programmed by the end-user application (IAP)
- 6 clocks per machine cycle operation (standard)
- 12 clocks per machine cycle operation (optional)
- Speed up to 20 MHz with 6 clock cycles per machine cycle (40 MHz equivalent performance); up to 33 MHz with 12 clocks per machine cycle
- Fully static operation
- RAM expandable externally to 64 kB
- 4 level priority interrupt
- 8 interrupt sources
- Four 8-bit I/O ports
- Full-duplex enhanced UART
  - Framing error detection
  - Automatic address recognition
- Power control modes
  - Clock can be stopped and resumed
  - Idle mode
  - Power down mode
- Programmable clock out
- Second DPTR register
- Asynchronous port reset
- Low EMI (inhibit ALE)
- Programmable Counter Array (PCA)
  - PWM
  - Capture/compare

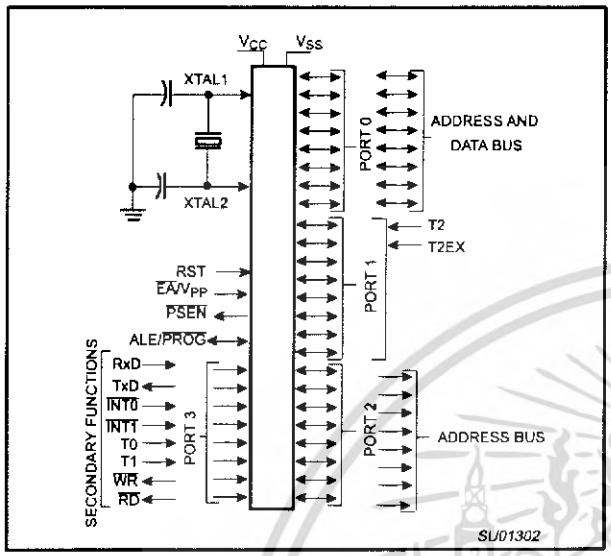
ORDERING INFORMATION

	PHILIPS (EXCEPT NORTH AMERICA) PART ORDER NUMBER PART MARKING	PHILIPS NORTH AMERICA PART ORDER NUMBER	MEMORY		TEMPERATURE RANGE (°C) AND PACKAGE	VOLTAGE RANGE	FREQUENCY (MHz)		DWG #
			FLASH	RAM			6 CLOCK MODE	12 CLOCK MODE	
1	P89C51RB2HBP	P89C51RB2BP	16 kB	512 B	0 to +70, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
2	P89C51RB2HFP	P89C51RB2FP	16 kB	512 B	–40 to +85, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
3	P89C51RB2HBA	P89C51RB2BA	16 kB	512 B	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
4	P89C51RB2HFA	P89C51RB2FA	16 kB	512 B	–40 to +85, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
5	P89C51RB2HBB	P89C51RB2BB	16 kB	512 B	0 to +70, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2
6	P89C51RB2HFB	P89C51RB2FB	16 kB	512 B	–40 to +85, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2
7	P89C51RC2HBP	P89C51RC2BP	32 kB	512 B	0 to +70, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
8	P89C51RC2HFP	P89C51RC2FP	32 kB	512 B	–40 to +85, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
9	P89C51RC2HBA	P89C51RC2BA	32 kB	512 B	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
10	P89C51RC2HFA	P89C51RC2FA	32 kB	512 B	–40 to +85, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
11	P89C51RC2HBB	P89C51RC2BB	32 kB	512 B	0 to +70, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2
12	P89C51RC2HFB	P89C51RC2FB	32 kB	512 B	–40 to +85, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2
13	P89C51RD2HBP	P89C51RD2BP	64 kB	1 kB	0 to +70, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
14	P89C51RD2HFP	P89C51RD2FP	64 kB	1 kB	–40 to +85, PDIP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT129-1
15	P89C51RD2HBA	P89C51RD2BA	64 kB	1 kB	0 to +70, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
16	P89C51RD2HFA	P89C51RD2FA	64 kB	1 kB	–40 to +85, PLCC	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT187-2
17	P89C51RD2HBB	P89C51RD2BB	64 kB	1 kB	0 to +70, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2
18	P89C51RD2HFB	P89C51RD2FB	64 kB	1 kB	–40 to +85, PQFP	4.5–5.5 V	0 to 20 MHz	0 to 33 MHz	SOT307-2

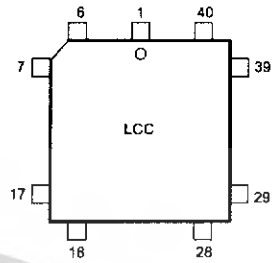
BLOCK DIAGRAM



LOGIC SYMBOL



Plastic Leaded Chip Carrier

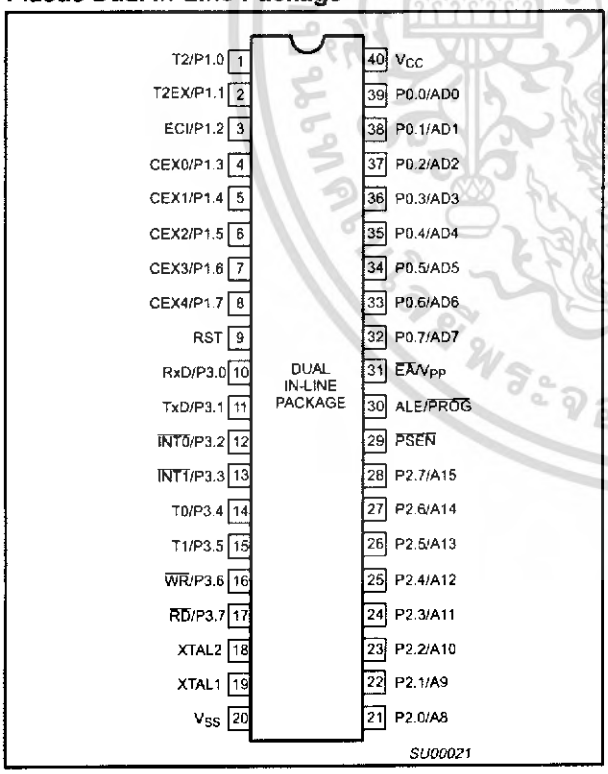


Pin	Function	Pin	Function	Pin	Function
1	NIC*	16	P3.4/T0	31	P2.7/A15
2	P1.0/T2	17	P3.5/T1	32	PSEN
3	P1.1/T2EX	18	P3.6/WR	33	ALE/PROG
4	P1.2/ECI	19	P3.7/RD	34	NIC*
5	P1.3/CEX0	20	XTAL2	35	EA/VPP
6	P1.4/CEX1	21	XTAL1	36	P0.7/AD7
7	P1.5/CEX2	22	VSS	37	P0.6/AD6
8	P1.6/CEX3	23	NIC*	38	P0.5/AD5
9	P1.7/CEX4	24	P2.0/A8	39	P0.4/AD4
10	RST	25	P2.1/A8	40	P0.3/AD3
11	P3.0/RxD	26	P2.2/A10	41	P0.2/AD2
12	NIC*	27	P2.3/A11	42	P0.1/AD1
13	P3.1/TxD	28	P2.4/A12	43	P0.0/AD0
14	P3.2/INT0	29	P2.5/A13	44	VCC
15	P3.3/INT1	30	P2.6/A14		

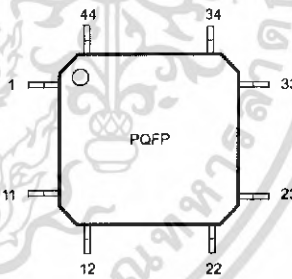
* NO INTERNAL CONNECTION SU00023

PINNING

Plastic Dual In-Line Package



Plastic Quad Flat Pack



Pin	Function	Pin	Function	Pin	Function
1	P1.5/CEX2	16	VSS	31	P0.6/AD6
2	P1.6/CEX3	17	NIC*	32	P0.5/AD5
3	P1.7/CEX4	18	P2.0/A8	33	P0.4/AD4
4	RST	19	P2.1/A9	34	P0.3/AD3
5	P3.0/RxD	20	P2.2/A10	35	P0.2/AD2
6	NIC*	21	P2.3/A11	36	P0.1/AD1
7	P3.1/TxD	22	P2.4/A12	37	P0.0/AD0
8	P3.2/INT0	23	P2.5/A13	38	VCC
9	P3.3/INT1	24	P2.6/A14	39	NIC*
10	P3.4/T0	25	P2.7/A15	40	P1.0/T2
11	P3.5/T1	26	PSEN	41	P1.1/T2EX
12	P3.6/WR	27	ALE/PROG	42	P1.2/ECI
13	P3.7/RD	28	NIC*	43	P1.3/CEX0
14	XTAL2	29	EA/VPP	44	P1.4/CEX1
15	XTAL1	30	P0.7/AD7		

* NO INTERNAL CONNECTION SU00024

PIN DESCRIPTIONS

MNEMONIC	PIN NUMBER			TYPE	NAME AND FUNCTION
	PDIP	PLCC	PQFP		
V_{SS}	20	22	16	I	**Ground:** 0 V reference.
V_{CC}	40	44	38	I	**Power Supply:** This is the power supply voltage for normal, idle, and power-down operation.
P0.0–0.7	39–32	43–36	37–30	I/O	**Port 0:** Port 0 is an open-drain, bidirectional I/O port. Port 0 pins that have 1s written to them float and can be used as high-impedance inputs. Port 0 is also the multiplexed low-order address and data bus during accesses to external program and data memory. In this application, it uses strong internal pull-ups when emitting 1s.
P1.0–P1.7	1–8	2–9	40–44, 1–3	I/O	**Port 1:** Port 1 is an 8-bit bidirectional I/O port with internal pull-ups on all pins except P1.6 and P1.7 which are open drain. Port 1 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 1 pins that are externally pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}). Alternate functions for 89C51RB2/RC2/RD2 Port 1 include:
	1	2	40	I/O	**T2 (P1.0):** Timer/Counter 2 external count input/Clockout (see Programmable Clock-Out)
	2	3	41	I	**T2EX (P1.1):** Timer/Counter 2 Reload/Capture/Direction Control
	3	4	42	I	**ECI (P1.2):** External Clock Input to the PCA
	4	5	43	I/O	**CEX0 (P1.3):** Capture/Compare External I/O for PCA module 0
	5	6	44	I/O	**CEX1 (P1.4):** Capture/Compare External I/O for PCA module 1
	6	7	1	I/O	**CEX2 (P1.5):** Capture/Compare External I/O for PCA module 2
	7	8	2	I/O	**CEX3 (P1.6):** Capture/Compare External I/O for PCA module 3
	8	9	3	I/O	**CEX4 (P1.7):** Capture/Compare External I/O for PCA module 4
P2.0–P2.7	21–28	24–31	18–25	I/O	**Port 2:** Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. Port 2 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 2 pins that are externally being pulled low will source current because of the internal pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @DPTR). In this application, it uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOV @Ri), port 2 emits the contents of the P2 special function register.
P3.0–P3.7	10–17	11, 13–19	5, 7–13	I/O	**Port 3:** Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. Port 3 pins that have 1s written to them are pulled high by the internal pull-ups and can be used as inputs. As inputs, port 3 pins that are externally being pulled low will source current because of the pull-ups. (See DC Electrical Characteristics: I_{IL}). Port 3 also serves the special features of the 89C51RB2/RC2/RD2, as listed below:
	10	11	5	I	**RxD (P3.0):** Serial input port
	11	13	7	O	**TxD (P3.1):** Serial output port
	12	14	8	I	**$\overline{INT0}$ (P3.2):** External interrupt
	13	15	9	I	**$\overline{INT1}$ (P3.3):** External interrupt
	14	16	10	I	**T0 (P3.4):** Timer 0 external input
	15	17	11	I	**T1 (P3.5):** Timer 1 external input
	16	18	12	O	**$\overline{WR}$ (P3.6):** External data memory write strobe
	17	19	13	O	**$\overline{RD}$ (P3.7):** External data memory read strobe
RST	9	10	4	I	**Reset:** A high on this pin for two machine cycles while the oscillator is running, resets the device. An internal diffused resistor to V_{SS} permits a power-on reset using only an external capacitor to V_{CC}.
ALE	30	33	27	O	**Address Latch Enable:** Output pulse for latching the low byte of the address during an access to external memory. In normal operation, ALE is emitted twice every machine cycle, and can be used for external timing or clocking. Note that one ALE pulse is skipped during each access to external data memory. ALE can be disabled by setting SFR auxiliary.0. With this bit set, ALE will be active only during a MOVX instruction.

MNEMONIC	PIN NUMBER			TYPE	NAME AND FUNCTION
	PDIP	PLCC	PQFP		
$\overline{PSEN}$	29	32	26	O	**Program Store Enable:** The read strobe to external program memory. When executing code from the external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory. $\overline{PSEN}$ is not activated during fetches from internal program memory.
$\overline{EA}/V_{PP}$	31	35	29	I	**External Access Enable/Programming Supply Voltage:** $\overline{EA}$ must be externally held low to enable the device to fetch code from external program memory locations. If $\overline{EA}$ is held high, the device executes from internal program memory. The value on the $\overline{EA}$ pin is latched when RST is released and any subsequent changes have no effect. This pin also receives the programming supply voltage (V_{PP}) during Flash programming.
XTAL1	19	21	15	I	**Crystal 1:** Input to the inverting oscillator amplifier and input to the internal clock generator circuits.
XTAL2	18	20	14	O	**Crystal 2:** Output from the inverting oscillator amplifier.

NOTE:
To avoid "latch-up" effect at power-on, the voltage on any pin (other than V_{PP}) must not be higher than V_{CC} + 0.5 V or less than V_{SS} – 0.5 V.

Table 1. Special Function Registers

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB							LSB	RESET VALUE
ACC*	Accumulator	E0H	E7	E6	E5	E4	E3	E2	E1	E0	00H
AUXR#	Auxiliary	8EH	–	–	–	–	–	–	EXTRAM	AO	xxxxxx10B
AUXR1#	Auxiliary 1	A2H	–	–	ENBOOT	–	GF2	0	–	DPS	xxxxxxx0B
B*	B register	F0H	F7	F6	F5	F4	F3	F2	F1	F0	00H
CCAP0H#	Module 0 Capture High	FAH									xxxxxxxxB
CCAP1H#	Module 1 Capture High	FBH									xxxxxxxxB
CCAP2H#	Module 2 Capture High	FCH									xxxxxxxxB
CCAP3H#	Module 3 Capture High	FDH									xxxxxxxxB
CCAP4H#	Module 4 Capture High	FEH									xxxxxxxxB
CCAP0L#	Module 0 Capture Low	EAH									xxxxxxxxB
CCAP1L#	Module 1 Capture Low	EBH									xxxxxxxxB
CCAP2L#	Module 2 Capture Low	ECH									xxxxxxxxB
CCAP3L#	Module 3 Capture Low	EDH									xxxxxxxxB
CCAP4L#	Module 4 Capture Low	EEH									xxxxxxxxB
CCAPM0#	Module 0 Mode	DAH	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM1#	Module 1 Mode	DBH	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM2#	Module 2 Mode	DCH	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM3#	Module 3 Mode	DDH	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
CCAPM4#	Module 4 Mode	DEH	–	ECOM	CAPP	CAPN	MAT	TOG	PWM	ECCF	x0000000B
			DF	DE	DD	DC	DB	DA	D9	D8	
CCON*#	PCA Counter Control	D8H	CF	CR	–	CCF4	CCF3	CCF2	CCF1	CCF0	00x00000B
CH#	PCA Counter High	F9H									00H
CL#	PCA Counter Low	E9H									00H
CMOD#	PCA Counter Mode	D9H	CIDL	WDTE	–	–	–	CPS1	CPS0	ECF	00xxx000B
DPTR:	Data Pointer (2 bytes)										
DPH	Data Pointer High	83H									00H
DPL	Data Pointer Low	82H									00H
			AF	AE	AD	AC	AB	AA	A9	A8	
IE*	Interrupt Enable 0	A8H	EA	EC	ET2	ES	ET1	EX1	ET0	EX0	00H
			BF	BE	BD	BC	BB	BA	B9	B8	
IP*	Interrupt Priority	B8H	–	PPC	PT2	PS	PT1	PX1	PT0	PX0	x0000000B
			B7	B6	B5	B4	B3	B2	B1	B0	
IPH#	Interrupt Priority High	B7H	–	PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H	x0000000B
			87	86	85	84	83	82	81	80	
P0*	Port 0	80H	AD7	AD6	AD5	AD4	AD3	AD2	AD1	AD0	FFH
			97	96	95	94	93	92	91	90	
P1*	Port 1	90H	CEX4	CEX3	CEX2	CEX1	CEX0	ECI	T2EX	T2	FFH
			A7	A6	A5	A4	A3	A2	A1	A0	
P2*	Port 2	A0H	AD15	AD14	AD13	AD12	AD11	AD10	AD9	AD8	FFH
			B7	B6	B5	B4	B3	B2	B1	B0	
P3*	Port 3	B0H	$\overline{RD}$	$\overline{WR}$	T1	T0	$\overline{INT1}$	$\overline{INT0}$	TxD	RxD	FFH
PCON#[1]	Power Control	87H	SMOD1	SMOD0	–	–	GF1	GF0	PD	IDL	00xxx000B

* SFRs are bit addressable.
\# SFRs are modified from or added to the 80C51 SFRs.
– Reserved bits.
1. Reset value depends on reset source.

Table 1. Special Function Registers (Continued)

SYMBOL	DESCRIPTION	DIRECT ADDRESS	BIT ADDRESS, SYMBOL, OR ALTERNATIVE PORT FUNCTION MSB LSB								RESET VALUE
			D7	D6	D5	D4	D3	D2	D1	D0	
PSW*	Program Status Word	D0H	CY	AC	F0	RS1	RS0	OV	F1	P	00000000B
RCAP2H#	Timer 2 Capture High	CBH									00H
RCAP2L#	Timer 2 Capture Low	CAH									00H
SADDR#	Slave Address	A9H									00H
SADEN#	Slave Address Mask	B9H									00H
SBUF	Serial Data Buffer	99H									xxxxxxxxB
			9F	9E	9D	9C	9B	9A	99	98	
SCON*	Serial Control	98H	SM0/FE	SM1	SM2	REN	TB8	RB8	TI	RI	00H
SP	Stack Pointer	81H									07H
			8F	8E	8D	8C	8B	8A	89	88	
TCON*	Timer Control	88H	TF1	TR1	TF0	TR0	IE1	IT1	IE0	IT0	00H
			CF	CE	CD	CC	CB	CA	C9	C8	
T2CON*	Timer 2 Control	C8H	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$	00H
T2MOD#	Timer 2 Mode Control	C9H	–	–	–	–	–	–	T2OE	DCEN	xxxxxx00B
TH0	Timer High 0	8CH									00H
TH1	Timer High 1	8DH									00H
TH2#	Timer High 2	CDH									00H
TL0	Timer Low 0	8AH									00H
TL1	Timer Low 1	8BH									00H
TL2#	Timer Low 2	CCH									00H
TMOD	Timer Mode	89H	GATE	C/$\overline{T}$	M1	M0	GATE	C/$\overline{T}$	M1	M0	00H
WDTRST	Watchdog Timer Reset	A6H									

* SFRs are bit addressable.
\# SFRs are modified from or added to the 80C51 SFRs.
– Reserved bits.

OSCILLATOR CHARACTERISTICS

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier. The pins can be configured for use as an on-chip oscillator.

To drive the device from an external clock source, XTAL1 should be driven while XTAL2 is left unconnected. Minimum and maximum high and low times specified in the data sheet must be observed.

This device is configured at the factory to operate using 6 clock periods per machine cycle, referred to in this datasheet as "6 clock mode". (This yields performance equivalent to twice that of standard 80C51 family devices). It may be optionally configured on commercially-available EPROM programming equipment to operate at 12 clocks per machine cycle, referred to in this datasheet as "12 clock mode". Once 12 clock mode has been configured, it cannot be changed back to 6 clock mode.

RESET

A reset is accomplished by holding the RST pin high for at least two machine cycles (12 oscillator periods in 6 clock mode, or 24 oscillator periods in 12 clock mode), while the oscillator is running. To ensure a good power-on reset, the RST pin must be high long enough to allow the oscillator time to start up (normally a few milliseconds) plus two machine cycles. At power-on, the voltage on V_{CC} and RST must come up at the same time for a proper start-up. Ports 1, 2, and 3 will asynchronously be driven to their reset condition when a voltage above V_{IH1} (min.) is applied to RESET.

The value on the $\overline{EA}$ pin is latched when RST is deasserted and has no further effect.

LOW POWER MODES

Stop Clock Mode

The static design enables the clock speed to be reduced down to 0 MHz (stopped). When the oscillator is stopped, the RAM and Special Function Registers retain their values. This mode allows step-by-step utilization and permits reduced system power consumption by lowering the clock frequency down to any value. For lowest power consumption the Power Down mode is suggested.

Idle Mode

In the idle mode (see Table 2), the CPU puts itself to sleep while all of the on-chip peripherals stay active. The instruction to invoke the idle mode is the last instruction executed in the normal operating mode before the idle mode is activated. The CPU contents, the on-chip RAM, and all of the special function registers remain intact during this mode. The idle mode can be terminated either by any enabled interrupt (at which time the process is picked up at the interrupt service routine and continued), or by a hardware reset which starts the processor in the same manner as a power-on reset.

Power-Down Mode

To save even more power, a Power Down mode (see Table 2) can be invoked by software. In this mode, the oscillator is stopped and the instruction that invoked Power Down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values down to 2.0 V and care must be taken to return V_{CC} to the minimum specified operating voltages before the Power Down Mode is terminated.

Either a hardware reset or external interrupt can be used to exit from Power Down. Reset redefines all the SFRs but does not change the on-chip RAM. An external interrupt allows both the SFRs and the on-chip RAM to retain their values.

To properly terminate Power Down, the reset or external interrupt should not be executed before V_{CC} is restored to its normal operating level and must be held active long enough for the oscillator to restart and stabilize (normally less than 10 ms).

With an external interrupt, INT0 and INT1 must be enabled and configured as level-sensitive. Holding the pin low restarts the oscillator but bringing the pin back high completes the exit. Once the interrupt is serviced, the next instruction to be executed after RETI will be the one following the instruction that put the device into Power Down.

POWER OFF FLAG

The Power Off Flag (POF) is set by on-chip circuitry when the V_{CC} level on the 89C51RB2/RC2/RD2 rises from 0 to 5 V. The POF bit can be set or cleared by software allowing a user to determine if the reset is the result of a power-on or a warm start after powerdown. The V_{CC} level must remain above 3 V for the POF to remain unaffected by the V_{CC} level.

Design Consideration

- When the idle mode is terminated by a hardware reset, the device normally resumes program execution, from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write when Idle is terminated by reset, the instruction following the one that invokes Idle should not be one that writes to a port pin or to external memory.

ONCE™ Mode

The ONCE ("On-Circuit Emulation") Mode facilitates testing and debugging of systems without the device having to be removed from the circuit. The ONCE Mode is invoked by:

1. Pull ALE low while the device is in reset and $\overline{PSEN}$ is high;
2. Hold ALE low as RST is deactivated.

While the device is in ONCE Mode, the Port 0 pins go into a float state, and the other port pins and ALE and $\overline{PSEN}$ are weakly pulled high. The oscillator circuit remains active. While the device is in this mode, an emulator or test CPU can be used to drive the circuit. Normal operation is restored when a normal reset is applied.

Programmable Clock-Out

A 50% duty cycle clock can be programmed to come out on P1.0. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed:

1. to input the external clock for Timer/Counter 2, or
2. to output a 50% duty cycle clock ranging from 122 Hz to 8 MHz at a 16 MHz operating frequency (61 Hz to 4 MHz in 12 clock mode).

To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (in T2CON) must be cleared and bit T20E in T2MOD must be set. Bit TR2 (T2CON.2) also must be set to start the timer.

The Clock-Out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L) as shown in this equation:

$$\frac{\text{Oscillator Frequency}}{n \times (65536 - \text{RCAP2H,RCAP2L})}$$

n = 2 in 6 clock mode
4 in 12 clock mode

Where (RCAP2H,RCAP2L) = the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

In the Clock-Out mode Timer 2 roll-overs will not generate an interrupt. This is similar to when it is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and the Clock-Out frequency will be the same.

Table 2. External Pin Status During Idle and Power-Down Mode

MODE	PROGRAM MEMORY	ALE	$\overline{PSEN}$	PORT 0	PORT 1	PORT 2	PORT 3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

TIMER 2 OPERATION

Timer 2

Timer 2 is a 16-bit Timer/Counter which can operate as either an event timer or an event counter, as selected by C/$\overline{T2}$* in the special function register T2CON (see Figure 1). Timer 2 has three operating modes: Capture, Auto-reload (up or down counting), and Baud Rate Generator, which are selected by bits in the T2CON as shown in Table 3.

Capture Mode

In the capture mode there are two options which are selected by bit EXEN2 in T2CON. If EXEN2=0, then timer 2 is a 16-bit timer or counter (as selected by C/$\overline{T2}$* in T2CON) which, upon overflowing sets bit TF2, the timer 2 overflow bit. This bit can be used to generate an interrupt (by enabling the Timer 2 interrupt bit in the IE register). If EXEN2= 1, Timer 2 operates as described above, but with the added feature that a 1- to -0 transition at external input T2EX causes the current value in the Timer 2 registers, TL2 and TH2, to be captured into registers RCAP2L and RCAP2H, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set, and EXF2 like TF2 can generate an interrupt (which vectors to the same location as Timer 2 overflow interrupt. The Timer 2 interrupt service routine can interrogate TF2 and EXF2 to determine which event caused the interrupt). The capture mode is illustrated in Figure 2 (There is no reload value for TL2 and TH2 in this mode. Even when a capture event occurs from T2EX, the counter keeps on counting T2EX pin transitions or osc/6 pulses (osc/12 in 12 clock mode).).

Auto-Reload Mode (Up or Down Counter)

In the 16-bit auto-reload mode, Timer 2 can be configured (as either a timer or counter [C/$\overline{T2}$* in T2CON]) then programmed to count up or down. The counting direction is determined by bit DCEN (Down Counter Enable) which is located in the T2MOD register (see Figure 3). When reset is applied the DCEN=0 which means Timer 2 will default to counting up. If DCEN bit is set, Timer 2 can count up or down depending on the value of the T2EX pin.

Figure 4 shows Timer 2 which will count up automatically since DCEN=0. In this mode there are two options selected by bit EXEN2 in T2CON register. If EXEN2=0, then Timer 2 counts up to 0FFFFH and sets the TF2 (Overflow Flag) bit upon overflow. This causes the Timer 2 registers to be reloaded with the 16-bit value in RCAP2L and RCAP2H. The values in RCAP2L and RCAP2H are preset by software means.

If EXEN2=1, then a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at input T2EX. This transition also sets the EXF2 bit. The Timer 2 interrupt, if enabled, can be generated when either TF2 or EXF2 are 1.

In Figure 5 DCEN=1 which enables Timer 2 to count up or down. This mode allows pin T2EX to control the direction of count. When a logic 1 is applied at pin T2EX Timer 2 will count up. Timer 2 will overflow at 0FFFFH and set the TF2 flag, which can then generate an interrupt, if the interrupt is enabled. This timer overflow also causes the 16-bit value in RCAP2L and RCAP2H to be reloaded into the timer registers TL2 and TH2.

When a logic 0 is applied at pin T2EX this causes Timer 2 to count down. The timer will underflow when TL2 and TH2 become equal to the value stored in RCAP2L and RCAP2H. Timer 2 underflow sets the TF2 flag and causes 0FFFFH to be reloaded into the timer registers TL2 and TH2.

The external flag EXF2 toggles when Timer 2 underflows or overflows. This EXF2 bit can be used as a 17th bit of resolution if needed. The EXF2 flag does not generate an interrupt in this mode of operation.

(MSB)							(LSB)
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$

Symbol	Position	Name and Significance
TF2	T2CON.7	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK or TCLK = 1.
EXF2	T2CON.6	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	T2CON.5	Receive clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	T2CON.4	Transmit clock flag. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	T2CON.3	Timer 2 external enable flag. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	T2CON.2	Start/stop control for Timer 2. A logic 1 starts the timer.
C/$\overline{T2}$	T2CON.1	Timer or counter select. (Timer 2) 0 = Internal timer (OSC/6 in 6 clock mode or OSC/12 in 12 clock mode) 1 = External event counter (falling edge triggered).
CP/$\overline{RL2}$	T2CON.0	Capture/Reload flag. When set, captures will occur on negative transitions at T2EX if EXEN2 = 1. When cleared, auto-reloads will occur either with Timer 2 overflows or negative transitions at T2EX when EXEN2 = 1. When either RCLK = 1 or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

SU01251

Figure 1. Timer/Counter 2 (T2CON) Control Register

Table 3. Timer 2 Operating Modes

RCLK + TCLK	CP/$\overline{RL2}$	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud rate generator
X	X	0	(off)

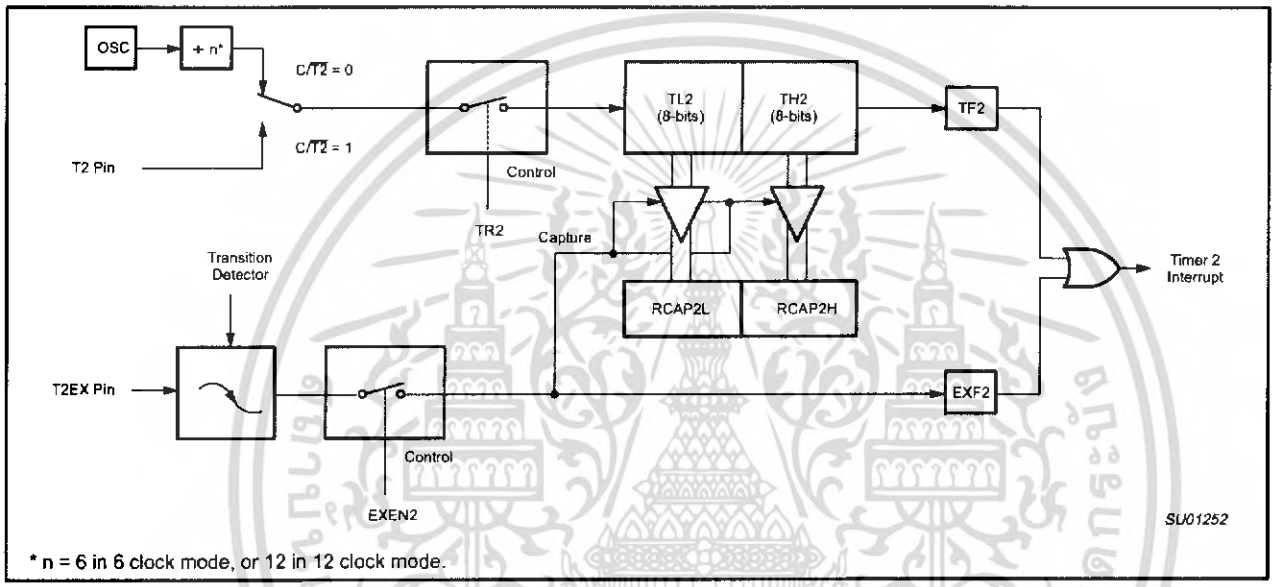


Figure 2. Timer 2 in Capture Mode

T2MOD Address = 0C9H Reset Value = XXXX XX00B

Not Bit Addressable

—	—	—	—	—	—	T2OE	DCEN
Bit 7	6	5	4	3	2	1	0

Symbol	Function
—	Not implemented, reserved for future use.*
T2OE	Timer 2 Output Enable bit.
DCEN	Down Count Enable bit. When set, this allows Timer 2 to be configured as an up/down counter.

* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU00729

Figure 3. Timer 2 Mode (T2MOD) Control Register

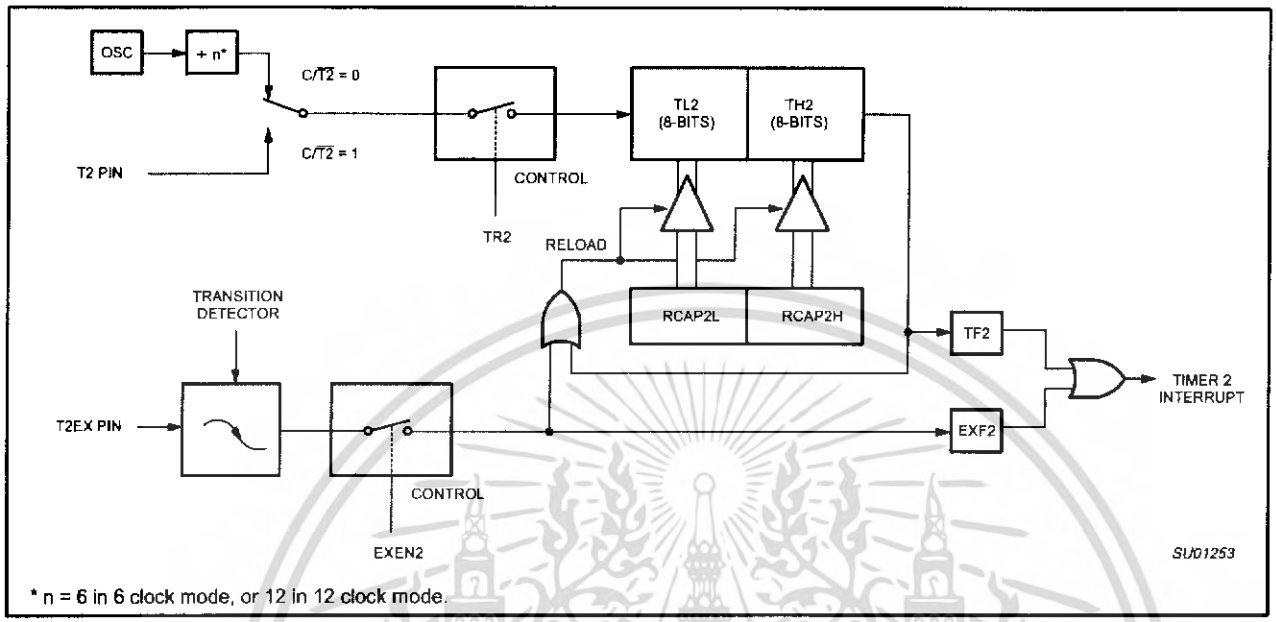


Figure 4. Timer 2 in Auto-Reload Mode (DCEN = 0)

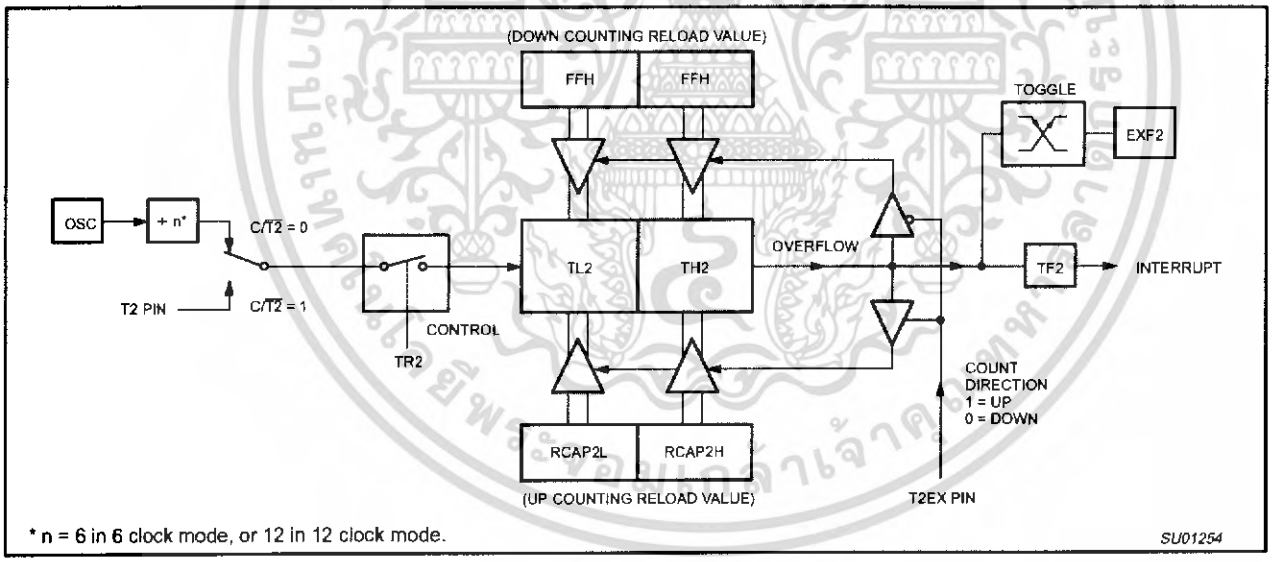


Figure 5. Timer 2 Auto Reload Mode (DCEN = 1)

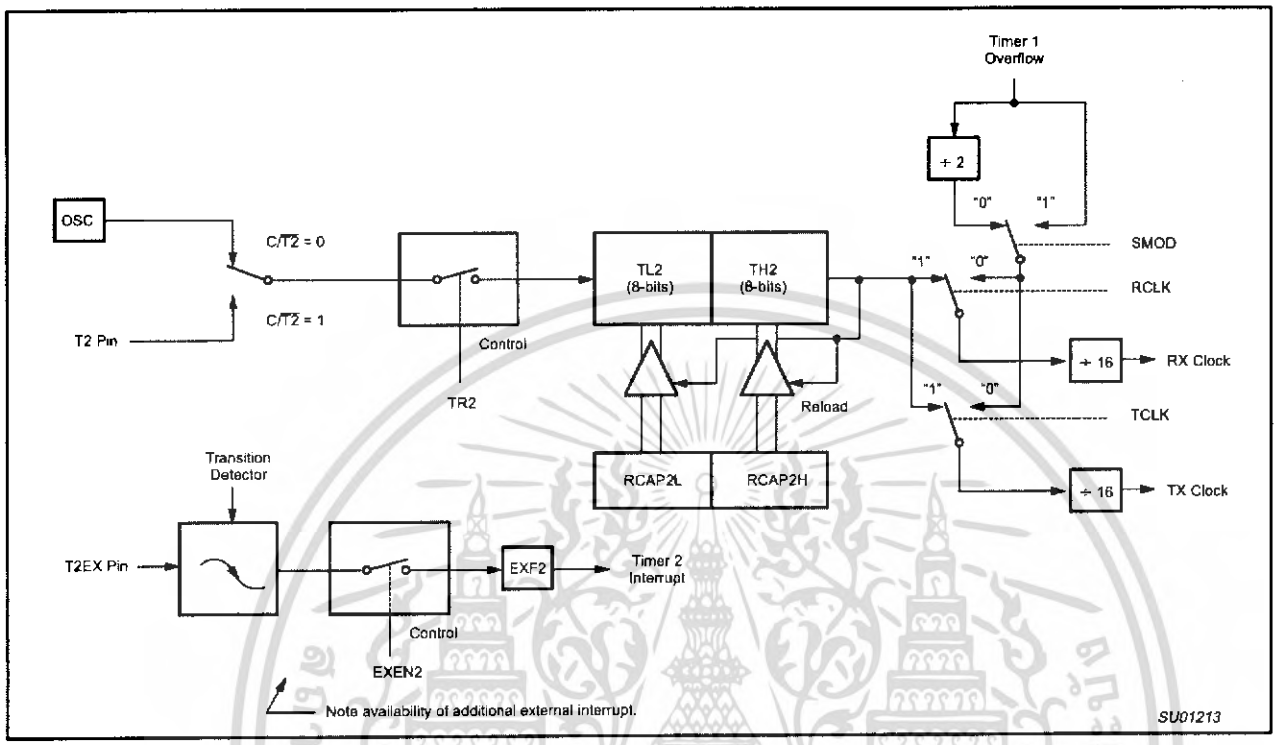


Figure 6. Timer 2 in Baud Rate Generator Mode

Table 4. Timer 2 Generated Commonly Used Baud Rates

Baud Rate		Osc Freq	Timer 2	
12 clock mode	6 clock mode		RCAP2H	RCAP2L
375 k	750 k	12 MHz	FF	FF
9.6 k	19.2 k	12 MHz	FF	D9
2.8 k	5.6 k	12 MHz	FF	B2
2.4 k	4.8 k	12 MHz	FF	64
1.2 k	2.4 k	12 MHz	FE	C8
300	600	12 MHz	FB	1E
110	220	12 MHz	F2	AF
300	600	6 MHz	FD	8F
110	220	6 MHz	F9	57

Baud Rate Generator Mode

Bits TCLK and/or RCLK in T2CON (Table 4) allow the serial port transmit and receive baud rates to be derived from either Timer 1 or Timer 2. When TCLK= 0, Timer 1 is used as the serial port transmit baud rate generator. When TCLK= 1, Timer 2 is used as the serial port transmit baud rate generator. RCLK has the same effect for the serial port receive baud rate. With these two bits, the serial port can have different receive and transmit baud rates – one generated by Timer 1, the other by Timer 2.

Figure 6 shows the Timer 2 in baud rate generation mode. The baud rate generation mode is like the auto-reload mode,in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in modes 1 and 3 are determined by Timer 2's overflow rate given below:

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

The timer can be configured for either "timer" or "counter" operation. In many applications, it is configured for "timer" operation (C/T2*=0). Timer operation is different for Timer 2 when it is being used as a baud rate generator.

Usually, as a timer it would increment every machine cycle (i.e., $^1/_6$ the oscillator frequency in 6 clock mode, $^1/_{12}$ the oscillator frequency in 12 clock mode). As a baud rate generator, it increments at the oscillator frequency in 6 clock mode ($^{OSC}/_2$ in 12 clock mode). Thus the baud rate formula is as follows:

$$\text{Modes 1 and 3 Baud Rates} = \frac{\text{Oscillator Frequency}}{[\,n^* \times [65536 - (\text{RCAP2H, RCAP2L})]]}$$

$^*n$ = 16 in 6 clock mode
32 in 12 clock mode

Where: (RCAP2H, RCAP2L)= The content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

The Timer 2 as a baud rate generator mode shown in Figure 6, is valid only if RCLK and/or TCLK = 1 in T2CON register. Note that a rollover in TH2 does not set TF2, and will not generate an interrupt. Thus, the Timer 2 interrupt does not have to be disabled when Timer 2 is in the baud rate generator mode. Also if the EXEN2 (T2 external enable flag) is set, a 1-to-0 transition in T2EX (Timer/counter 2 trigger input) will set EXF2 (T2 external flag) but will not cause a reload from (RCAP2H, RCAP2L) to (TH2,TL2). Therefore when Timer 2 is in use as a baud rate generator, T2EX can be used as an additional external interrupt, if needed.

When Timer 2 is in the baud rate generator mode, one should not try to read or write TH2 and TL2. As a baud rate generator, Timer 2 is incremented every state time (osc/2) or asynchronously from pin T2; under these conditions, a read or write of TH2 or TL2 may not be accurate. The RCAP2 registers may be read, but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

Table 4 shows commonly used baud rates and how they can be obtained from Timer 2.

Summary of Baud Rate Equations

Timer 2 is in baud rate generating mode. If Timer 2 is being clocked through pin T2(P1.0) the baud rate is:

$$\text{Baud Rate} = \frac{\text{Timer 2 Overflow Rate}}{16}$$

If Timer 2 is being clocked internally, the baud rate is:

$$\text{Baud Rate} = \frac{f_{osc}}{[\,n^* \times [65536 - (\text{RCAP2H, RCAP2L})]]}$$

* n = 16 in 6 clock mode
32 in 12 clock mode

Where f_{OSC}= Oscillator Frequency

To obtain the reload value for RCAP2H and RCAP2L, the above equation can be rewritten as:

$$\text{RCAP2H, RCAP2L} = 65536 - \left(\frac{f_{osc}}{n^* \times \text{Baud Rate}}\right)$$

Timer/Counter 2 Set-up

Except for the baud rate generator mode, the values given for T2CON do not include the setting of the TR2 bit. Therefore, bit TR2 must be set, separately, to turn the timer on. see Table 5 for set-up of Timer 2 as a timer. Also see Table 6 for set-up of Timer 2 as a counter.

Table 5. Timer 2 as a Timer

MODE	T2CON	
	INTERNAL CONTROL (Note 1)	EXTERNAL CONTROL (Note 2)
16-bit Auto-Reload	00H	08H
16-bit Capture	01H	09H
Baud rate generator receive and transmit same baud rate	34H	36H
Receive only	24H	26H
Transmit only	14H	16H

Table 6. Timer 2 as a Counter

MODE	TMOD	
	INTERNAL CONTROL (Note 1)	EXTERNAL CONTROL (Note 2)
16-bit	02H	0AH
Auto-Reload	03H	0BH

NOTES:
1. Capture/reload occurs only on timer/counter overflow.
2. Capture/reload occurs on timer/counter overflow and a 1-to-0 transition on T2EX (P1.1) pin except when Timer 2 is used in the baud rate generator mode.

Enhanced UART

The UART operates in all of the usual modes that are described in the first section of *Data Handbook IC20, 80C51-Based 8-Bit Microcontrollers*. In addition the UART can perform framing error detect by looking for missing stop bits, and automatic address recognition. The UART also fully supports multiprocessor communication as does the standard 80C51 UART.

When used for framing error detect the UART looks for missing stop bits in the communication. A missing bit will set the FE bit in the SCON register. The FE bit shares the SCON.7 bit with SM0 and the function of SCON.7 is determined by PCON.6 (SMOD0) (see Figure 7). If SMOD0 is set then SCON.7 functions as FE. SCON.7 functions as SM0 when SMOD0 is cleared. When used as FE SCON.7 can only be cleared by software. Refer to Figure 8.

Automatic Address Recognition

Automatic Address Recognition is a feature which allows the UART to recognize certain addresses in the serial bit stream by using hardware to make the comparisons. This feature saves a great deal of software overhead by eliminating the need for the software to examine every serial address which passes by the serial port. This feature is enabled by setting the SM2 bit in SCON. In the 9 bit UART modes, mode 2 and mode 3, the Receive Interrupt flag (RI) will be automatically set when the received byte contains either the "Given" address or the "Broadcast" address. The 9-bit mode requires that the 9th information bit is a 1 to indicate that the received information is an address and not data. Automatic address recognition is shown in Figure 9.

The 8 bit mode is called Mode 1. In this mode the RI flag will be set if SM2 is enabled and the information received has a valid stop bit following the 8 address bits and the information is either a Given or Broadcast address.

Mode 0 is the Shift Register mode and SM2 is ignored.

Using the Automatic Address Recognition feature allows a master to selectively communicate with one or more slaves by invoking the Given slave address or addresses. All of the slaves may be contacted by using the Broadcast address. Two special Function Registers are used to define the slave's address, SADDR, and the address mask, SADEN. SADEN is used to define which bits in the SADDR are to b used and which bits are "don't care". The SADEN mask can be logically ANDed with the SADDR to create the "Given" address which the master will use for addressing each of the slaves. Use of the Given address allows multiple slaves to be recognized while excluding others. The following examples will help to show the versatility of this scheme:

Slave 0	SADDR	=	1100 0000
	SADEN	=	1111 1101
	Given	=	1100 00X0
Slave 1	SADDR	=	1100 0000
	SADEN	=	1111 1110
	Given	=	1100 000X

In the above example SADDR is the same and the SADEN data is used to differentiate between the two slaves. Slave 0 requires a 0 in bit 0 and it ignores bit 1. Slave 1 requires a 0 in bit 1 and bit 0 is ignored. A unique address for Slave 0 would be 1100 0010 since slave 1 requires a 0 in bit 1. A unique address for slave 1 would be 1100 0001 since a 1 in bit 0 will exclude slave 0. Both slaves can be selected at the same time by an address which has bit 0 = 0 (for slave 0) and bit 1 = 0 (for slave 1). Thus, both could be addressed with 1100 0000.

In a more complex system the following could be used to select slaves 1 and 2 while excluding slave 0:

Slave 0	SADDR	=	1100 0000
	SADEN	=	1111 1001
	Given	=	1100 0XX0
Slave 1	SADDR	=	1110 0000
	SADEN	=	1111 1010
	Given	=	1110 0X0X
Slave 2	SADDR	=	1110 0000
	SADEN	=	1111 1100
	Given	=	1110 00XX

In the above example the differentiation among the 3 slaves is in the lower 3 address bits. Slave 0 requires that bit 0 = 0 and it can be uniquely addressed by 1110 0110. Slave 1 requires that bit 1 = 0 and it can be uniquely addressed by 1110 and 0101. Slave 2 requires that bit 2 = 0 and its unique address is 1110 0011. To select Slaves 0 and 1 and exclude Slave 2 use address 1110 0100, since it is necessary to make bit 2 = 1 to exclude slave 2.

The Broadcast Address for each slave is created by taking the logical OR of SADDR and SADEN. Zeros in this result are trended as don't-cares. In most cases, interpreting the don't-cares as ones, the broadcast address will be FF hexadecimal.

Upon reset SADDR (SFR address 0A9H) and SADEN (SFR address 0B9H) are leaded with 0s. This produces a given address of all "don't cares" as well as a Broadcast address of all "don't cares". This effectively disables the Automatic Addressing mode and allows the microcontroller to use standard 80C51 type UART drivers which do not make use of this feature.

SCON Address = 98H Reset Value = 0000 0000B

Bit Addressable

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

(SMOD0 = 0/1)*

Symbol	Function
FE	Framing Error bit. This bit is set by the receiver when an invalid stop bit is detected. The FE bit is not cleared by valid frames but should be cleared by software. The SMOD0 bit must be set to enable access to the FE bit.
SM0	Serial Port Mode Bit 0, (SMOD0 must = 0 to access bit SM0)
SM1	Serial Port Mode Bit 1

SM0	SM1	Mode	Description	Baud Rate**
0	0	0	shift register	$f_{OSC}/6$ (6 clock mode) or $f_{OSC}/12$ (12 clock mode)
0	1	1	8-bit UART	variable
1	0	2	9-bit UART	$f_{OSC}/32$ or $f_{OSC}/16$ (6 clock mode) or $f_{OSC}/64$ or $f_{OSC}/32$ (12 clock mode)
1	1	3	9-bit UART	variable

SM2	Enables the Automatic Address Recognition feature in Modes 2 or 3. If SM2 = 1 then RI will not be set unless the received 9th data bit (RB8) is 1, indicating an address, and the received byte is a Given or Broadcast Address. In Mode 1, if SM2 = 1 then RI will not be activated unless a valid stop bit was received, and the received byte is a Given or Broadcast Address. In Mode 0, SM2 should be 0.
REN	Enables serial reception. Set by software to enable reception. Clear by software to disable reception.
TB8	The 9th data bit that will be transmitted in Modes 2 and 3. Set or clear by software as desired.
RB8	In modes 2 and 3, the 9th data bit that was received. In Mode 1, if SM2 = 0, RB8 is the stop bit that was received. In Mode 0, RB8 is not used.
TI	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.
RI	Receive interrupt flag. Set by hardware at the end of the 8th bit time in Mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

NOTE:
*SMOD0 is located at PCON6.
**f_{OSC} = oscillator frequency

SU01255

Figure 7. SCON: Serial Port Control Register

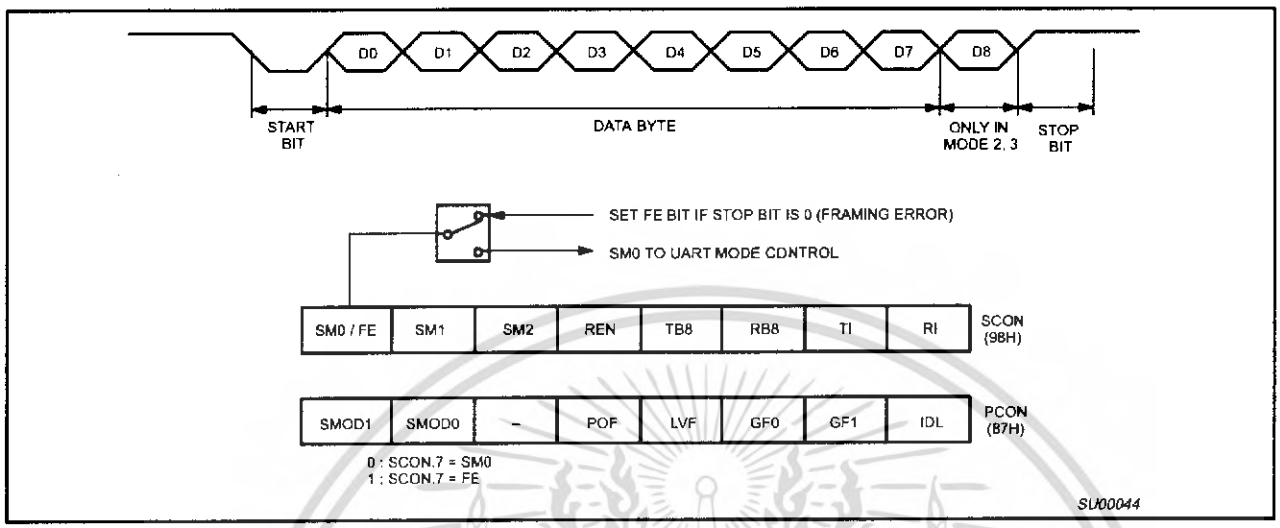


Figure 8. UART Framing Error Detection

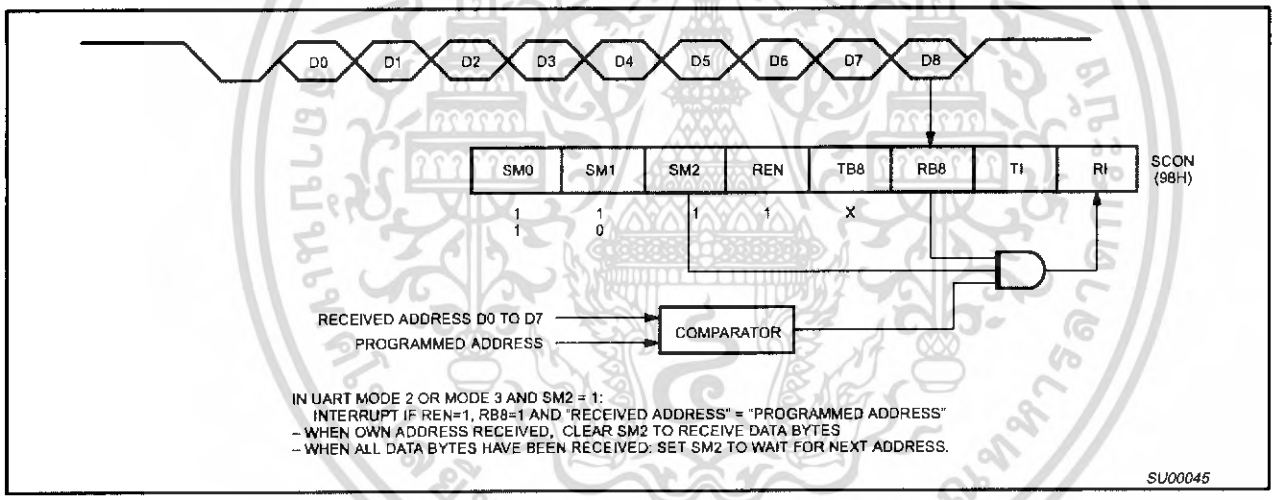


Figure 9. UART Multiprocessor Communication, Automatic Address Recognition

Interrupt Priority Structure

The 89C51RB2/RC2/RD2 has an 8 source four-level interrupt structure (see Table 7).

There are 3 SFRs associated with the four-level interrupt. They are the IE, IP, and IPH. (See Figures 10, 11, and 12.) The IPH (Interrupt Priority High) register makes the four-level interrupt structure possible. The IPH is located at SFR address B7H. The structure of the IPH register and a description of its bits is shown in Figure 12.

The function of the IPH SFR is simple and when combined with the IP SFR determines the priority of each interrupt. The priority of each interrupt is determined as shown in the following table:

PRIORITY BITS		INTERRUPT PRIORITY LEVEL
IPH.x	IP.x	
0	0	Level 0 (lowest priority)
0	1	Level 1
1	0	Level 2
1	1	Level 3 (highest priority)

The priority scheme for servicing the interrupts is the same as that for the 80C51, except there are four interrupt levels rather than two as on the 80C51. An interrupt will be serviced as long as an interrupt of equal or higher priority is not already being serviced. If an interrupt of equal or higher level priority is being serviced, the new interrupt will wait until it is finished before being serviced. If a lower priority level interrupt is being serviced, it will be stopped and the new interrupt serviced. When the new interrupt is finished, the lower priority level interrupt that was stopped will be completed.

Table 7. Interrupt Table

SOURCE	POLLING PRIORITY	REQUEST BITS	HARDWARE CLEAR?	VECTOR ADDRESS
X0	1	IE0	N (L)[1] Y (T)[2]	03H
T0	2	TP0	Y	0BH
X1	3	IE1	N (L) Y (T)	13H
T1	4	TF1	Y	1BH
PCA	5	CF, CCFn n = 0–4	N	33H
SP	6	RI, TI	N	23H
T2	7	TF2, EXF2	N	2BH

NOTES:
1. L = Level activated
2. T = Transition activated

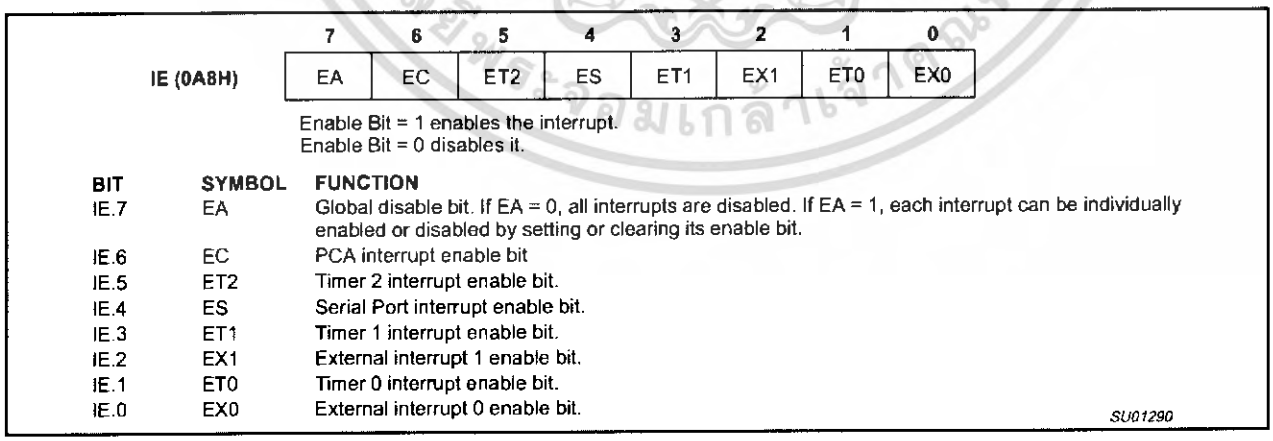


Figure 10. IE Registers

	7	6	5	4	3	2	1	0
IP (0B8H)	–	PPC	PT2	PS	PT1	PX1	PT0	PX0

Priority Bit = 1 assigns high priority
Priority Bit = 0 assigns low priority

BIT	SYMBOL	FUNCTION
IP.7	–	–
IP.6	PPC	PCA interrupt priority bit
IP.5	PT2	Timer 2 interrupt priority bit.
IP.4	PS	Serial Port interrupt priority bit.
IP.3	PT1	Timer 1 interrupt priority bit.
IP.2	PX1	External interrupt 1 priority bit.
IP.1	PT0	Timer 0 interrupt priority bit.
IP.0	PX0	External interrupt 0 priority bit.

SU01291

Figure 11. IP Registers

	7	6	5	4	3	2	1	0
IPH (B7H)	–	PPCH	PT2H	PSH	PT1H	PX1H	PT0H	PX0H

Priority Bit = 1 assigns higher priority
Priority Bit = 0 assigns lower priority

BIT	SYMBOL	FUNCTION
IPH.7	–	–
IPH.6	PPCH	PCA interrupt priority bit
IPH.5	PT2H	Timer 2 interrupt priority bit high.
IPH.4	PSH	Serial Port interrupt priority bit high.
IPH.3	PT1H	Timer 1 interrupt priority bit high.
IPH.2	PX1H	External interrupt 1 priority bit high.
IPH.1	PT0H	Timer 0 interrupt priority bit high.
IPH.0	PX0H	External interrupt 0 priority bit high.

SU01292

Figure 12. IPH Registers

Reduced EMI Mode

The AO bit (AUXR.0) in the AUXR register when set disables the ALE output.

Reduced EMI Mode

AUXR (8EH)

7	6	5	4	3	2	1	0
–	–	–	–	–	–	EXTRAM	AO

AUXR.1	EXTRAM	
AUXR.0	AO	Turns off ALE output.

Dual DPTR

The dual DPTR structure (see Figure 13) is a way by which the chip will specify the address of an external data memory location. There are two 16-bit DPTR registers that address the external memory, and a single bit called DPS = AUXR1/bit0 that allows the program code to switch between them.

- New Register Name: AUXR1#
- SFR Address: A2H
- Reset Value: xxxxxxx0B

AUXR1 (A2H)

7	6	5	4	3	2	1	0
–	–	ENBOOT	–	GF2	0	–	DPS

Where:
DPS = AUXR1/bit0 = Switches between DPTR0 and DPTR1.

Select Reg	DPS
DPTR0	0
DPTR1	1

The DPS bit status should be saved by software when switching between DPTR0 and DPTR1.

The GF2 bit is a general purpose user-defined flag. Note that bit 2 is not writable and is always read as a zero. This allows the DPS bit to be quickly toggled simply by executing an INC AUXR1 instruction without affecting the GF2 bit.

The ENBOOT bit determines whether the BOOTROM is enabled or disabled. This bit will automatically be set if the status byte is non zero during reset or $\overline{PSEN}$ is pulled low, ALE floats high, and EA > V_{iH} on the falling edge of reset. Otherwise, this bit will be cleared during reset.

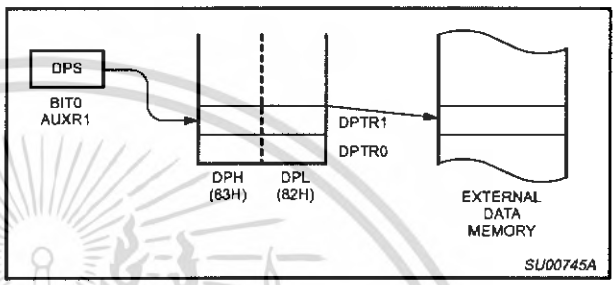


Figure 13.

DPTR Instructions

The instructions that refer to DPTR refer to the data pointer that is currently selected using the AUXR1/bit 0 register. The six instructions that use the DPTR are as follows:

INC DPTR	Increments the data pointer by 1
MOV DPTR, #data16	Loads the DPTR with a 16-bit constant
MOV A, @ A+DPTR	Move code byte relative to DPTR to ACC
MOVX A, @ DPTR	Move external RAM (16-bit address) to ACC
MOVX @ DPTR , A	Move ACC to external RAM (16-bit address)
JMP @ A + DPTR	Jump indirect relative to DPTR

The data pointer can be accessed on a byte-by-byte basis by specifying the low or high byte in an instruction which accesses the SFRs. See *Application Note AN458* for more details.

Programmable Counter Array (PCA)

The Programmable Counter Array available on the 89C51RB2/RC2/RD2 is a special 16-bit Timer that has five 16-bit capture/compare modules associated with it. Each of the modules can be programmed to operate in one of four modes: rising and/or falling edge capture, software timer, high-speed output, or pulse width modulator. Each module has a pin associated with it in port 1. Module 0 is connected to P1.3(CEX0), module 1 to P1.4(CEX1), etc. The basic PCA configuration is shown in Figure 14.

The PCA timer is a common time base for all five modules and can be programmed to run at: 1/6 the oscillator frequency, 1/2 the oscillator frequency, the Timer 0 overflow, or the input on the ECI pin (P1.2). The timer count source is determined from the CPS1 and CPS0 bits in the CMOD SFR as follows (see Figure 17):

CPS1	CPS0	PCA Timer Count Source
0	0	1/6 oscillator frequency (6 clock mode); 1/12 oscillator frequency (12 clock mode)
0	1	1/2 oscillator frequency (6 clock mode); 1/4 oscillator frequency (12 clock mode)
1	0	Timer 0 overflow
1	1	External input at ECI pin

In the CMOD SFR are three additional bits associated with the PCA. They are CIDL which allows the PCA to stop during idle mode, WDTE which enables or disables the watchdog function on module 4, and ECF which when set causes an interrupt and the PCA overflow flag CF (in the CCON SFR) to be set when the PCA timer overflows. These functions are shown in Figure 15.

The watchdog timer function is implemented in module 4 (see Figure 24).

The CCON SFR contains the run control bit for the PCA and the flags for the PCA timer (CF) and each module (refer to Figure 18). To run the PCA the CR bit (CCON.6) must be set by software. The PCA is shut off by clearing this bit. The CF bit (CCON.7) is set when the PCA counter overflows and an interrupt will be generated if the ECF bit in the CMOD register is set, The CF bit can only be cleared by software. Bits 0 through 4 of the CCON register are the flags for the modules (bit 0 for module 0, bit 1 for module 1, etc.) and are set by hardware when either a match or a capture occurs. These flags also can only be cleared by software. The PCA interrupt system shown in Figure 16.

Each module in the PCA has a special function register associated with it. These registers are: CCAPM0 for module 0, CCAPM1 for module 1, etc. (see Figure 19). The registers contain the bits that control the mode that each module will operate in. The ECCF bit (CCAPMn.0 where n=0, 1, 2, 3, or 4 depending on the module) enables the CCF flag in the CCON SFR to generate an interrupt when a match or compare occurs in the associated module. PWM (CCAPMn.1) enables the pulse width modulation mode. The TOG bit (CCAPMn.2) when set causes the CEX output associated with the module to toggle when there is a match between the PCA counter and the module's capture/compare register. The match bit MAT (CCAPMn.3) when set will cause the CCFn bit in the CCON register to be set when there is a match between the PCA counter and the module's capture/compare register.

The next two bits CAPN (CCAPMn.4) and CAPP (CCAPMn.5) determine the edge that a capture input will be active on. The CAPN bit enables the negative edge, and the CAPP bit enables the positive edge. If both bits are set both edges will be enabled and a capture will occur for either transition. The last bit in the register ECOM (CCAPMn.6) when set enables the comparator function. Figure 20 shows the CCAPMn settings for the various PCA functions.

There are two additional registers associated with each of the PCA modules. They are CCAPnH and CCAPnL and these are the registers that store the 16-bit count when a capture occurs or a compare should occur. When a module is used in the PWM mode these registers are used to control the duty cycle of the output.

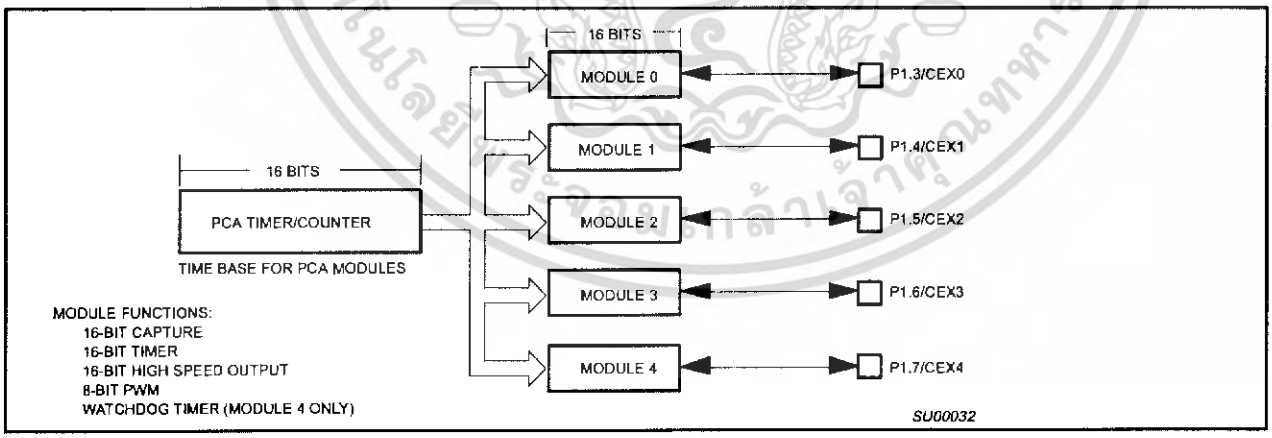


Figure 14. Programmable Counter Array (PCA)

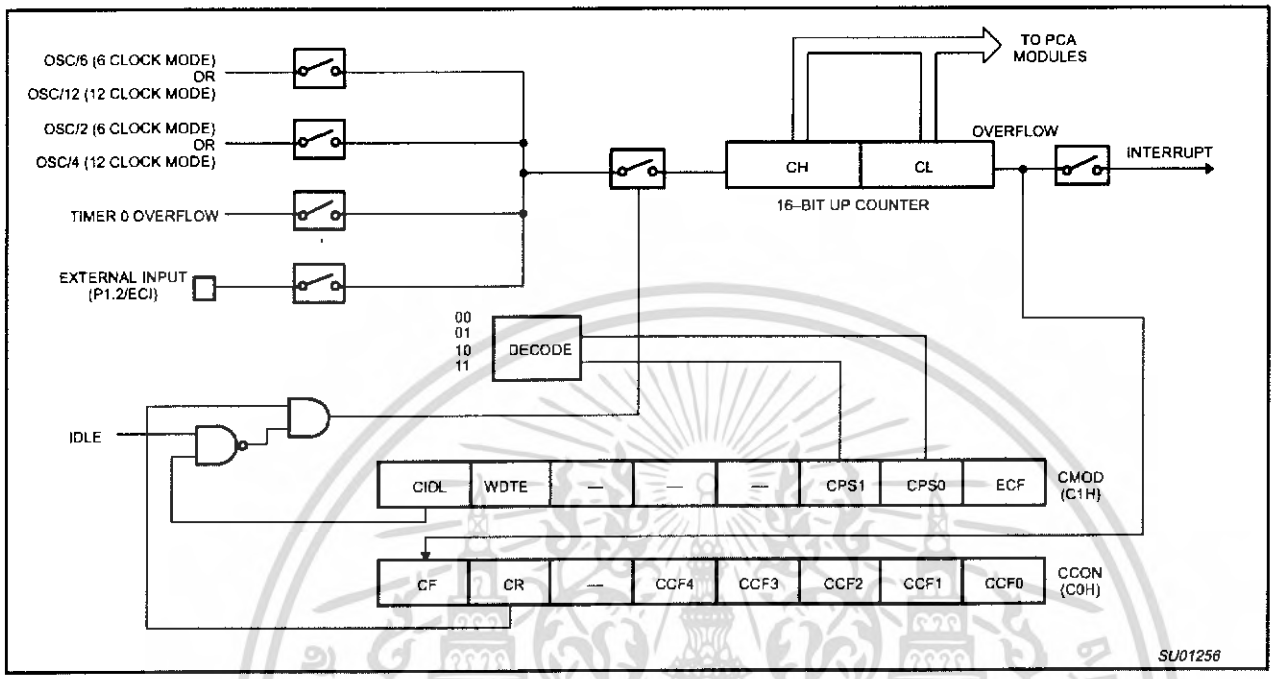


Figure 15. PCA Timer/Counter

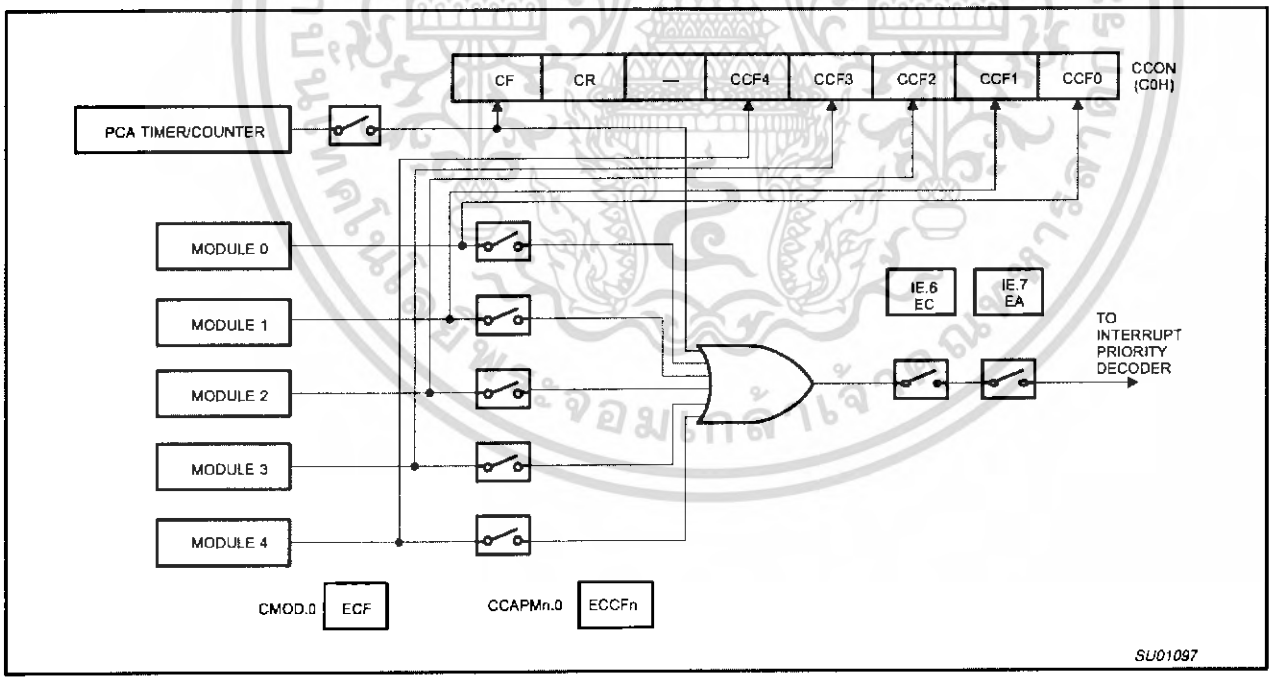


Figure 16. PCA Interrupt System

CMOD Address = C1H Reset Value = 00XX X000B

Bit:	7	6	5	4	3	2	1	0
	CIDL	WDTE	–	–	–	CPS1	CPS0	ECF

Symbol	Function
CIDL	Counter Idle control: CIDL = 0 programs the PCA Counter to continue functioning during idle Mode. CIDL = 1 programs it to be gated off during idle.
WDTE	Watchdog Timer Enable: WDTE = 0 disables Watchdog Timer function on PCA Module 4. WDTE = 1 enables it.
–	Not implemented, reserved for future use.*
CPS1	PCA Count Pulse Select bit 1.
CPS0	PCA Count Pulse Select bit 0.

CPS1	CPS0	Selected PCA Input**
0	0	0 Internal clock, f_{OSC}/6 in 6 clock mode (f_{OSC}/12 in 12 clock mode)
0	1	1 Internal clock, f_{OSC}/2 in 6 clock mode (f_{OSC}/4 in 12 clock mode)
1	0	2 Timer 0 overflow
1	1	3 External clock at ECI/P1.2 pin (max. rate = f_{OSC}/4 in 6 clock mode, f_{OSC}/8 in 12 clock mode)

ECF	PCA Enable Counter Overflow interrupt: ECF = 1 enables CF bit in CCON to generate an interrupt. ECF = 0 disables that function of CF.

NOTE:
* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.
** f_{OSC} = oscillator frequency

SU01257

Figure 17. CMOD: PCA Counter Mode Register

CCON Address = 0C0H Reset Value = 00X0 0000B

Bit Addressable

Bit:	7	6	5	4	3	2	1	0
	CF	CR	–	CCF4	CCF3	CCF2	CCF1	CCF0

Symbol	Function
CF	PCA Counter Overflow flag. Set by hardware when the counter rolls over. CF flags an interrupt if bit ECF in CMOD is set. CF may be set by either hardware or software but can only be cleared by software.
CR	PCA Counter Run control bit. Set by software to turn the PCA counter on. Must be cleared by software to turn the PCA counter off.
–	Not implemented, reserved for future use*.
CCF4	PCA Module 4 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF3	PCA Module 3 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF2	PCA Module 2 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF1	PCA Module 1 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.
CCF0	PCA Module 0 interrupt flag. Set by hardware when a match or capture occurs. Must be cleared by software.

NOTE:
* User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01099

Figure 18. CCON: PCA Counter Control Register

CCAPMn Address CCAPM0 0C2H Reset Value = X000 0000B
CCAPM1 0C3H
CCAPM2 0C4H
CCAPM3 0C5H
CCAPM4 0C6H

Not Bit Addressable

	–	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn
Bit:	7	6	5	4	3	2	1	0

Symbol	Function
–	Not implemented, reserved for future use*.
ECOMn	Enable Comparator. ECOMn = 1 enables the comparator function.
CAPPn	Capture Positive, CAPPn = 1 enables positive edge capture.
CAPNn	Capture Negative, CAPNn = 1 enables negative edge capture.
MATn	Match. When MATn = 1, a match of the PCA counter with this module's compare/capture register causes the CCFn bit in CCON to be set, flagging an interrupt.
TOGn	Toggle. When TOGn = 1, a match of the PCA counter with this module's compare/capture register causes the CEXn pin to toggle.
PWMn	Pulse Width Modulation Mode. PWMn = 1 enables the CEXn pin to be used as a pulse width modulated output.
ECCFn	Enable CCF interrupt. Enables compare/capture flag CCFn in the CCON register to generate an interrupt.

NOTE:
*User software should not write 1s to reserved bits. These bits may be used in future 8051 family products to invoke new features. In that case, the reset or inactive value of the new bit will be 0, and its active value will be 1. The value read from a reserved bit is indeterminate.

SU01100

Figure 19. CCAPMn: PCA Modules Compare/Capture Registers

–	ECOMn	CAPPn	CAPNn	MATn	TOGn	PWMn	ECCFn	MODULE FUNCTION
X	0	0	0	0	0	0	0	No operation
X	X	1	0	0	0	0	X	16-bit capture by a positive-edge trigger on CEXn
X	X	0	1	0	0	0	X	16-bit capture by a negative trigger on CEXn
X	X	1	1	0	0	0	X	16-bit capture by a transition on CEXn
X	1	0	0	1	0	0	X	16-bit Software Timer
X	1	0	0	1	1	0	X	16-bit High Speed Output
X	1	0	0	0	0	1	0	8-bit PWM
X	1	0	0	1	X	0	X	Watchdog Timer

Figure 20. PCA Module Modes (CCAPMn Register)

PCA Capture Mode
To use one of the PCA modules in the capture mode either one or both of the CCAPM bits CAPN and CAPP for that module must be set. The external CEX input for the module (on port 1) is sampled for a transition. When a valid transition occurs the PCA hardware loads the value of the PCA counter registers (CH and CL) into the module's capture registers (CCAPnL and CCAPnH). If the CCFn bit for the module in the CCON SFR and the ECCFn bit in the CCAPMn SFR are set then an interrupt will be generated. Refer to Figure 21.

16-bit Software Timer Mode
The PCA modules can be used as software timers by setting both the ECOM and MAT bits in the modules CCAPMn register. The PCA timer will be compared to the module's capture registers and when a match occurs an interrupt will occur if the CCFn (CCON SFR) and the ECCFn (CCAPMn SFR) bits for the module are both set (see Figure 22).

High Speed Output Mode
In this mode the CEX output (on port 1) associated with the PCA module will toggle each time a match occurs between the PCA counter and the module's capture registers. To activate this mode the TOG, MAT, and ECOM bits in the module's CCAPMn SFR must be set (see Figure 23).

Pulse Width Modulator Mode
All of the PCA modules can be used as PWM outputs. Figure 24 shows the PWM function. The frequency of the output depends on the source for the PCA timer. All of the modules will have the same frequency of output because they all share the PCA timer. The duty cycle of each module is independently variable using the module's capture register CCAPLn. When the value of the PCA CL SFR is less than the value in the module's CCAPLn SFR the output will be low, when it is equal to or greater than the output will be high. When CL overflows from FF to 00, CCAPLn is reloaded with the value in CCAPHn. the allows updating the PWM without glitches. The PWM and ECOM bits in the module's CCAPMn register must be set to enable the PWM mode.

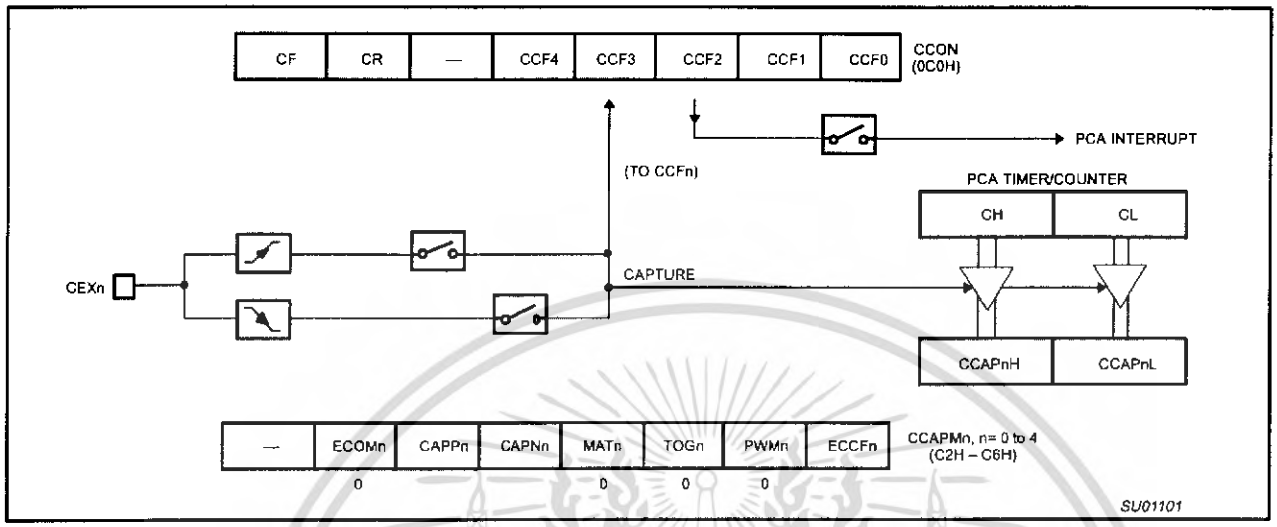


Figure 21. PCA Capture Mode

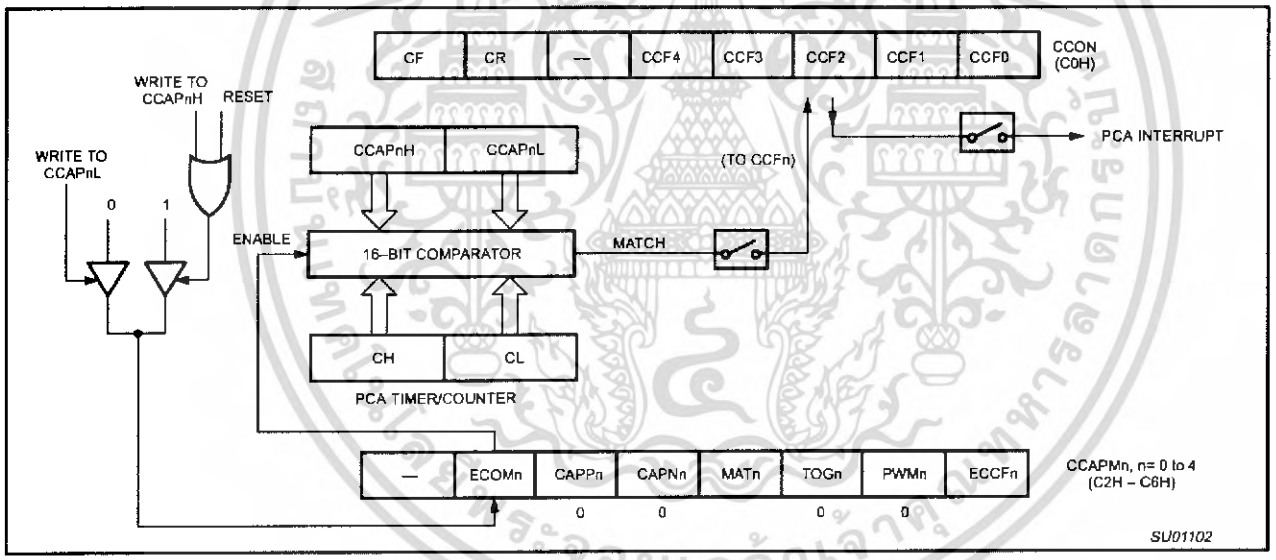


Figure 22. PCA Compare Mode

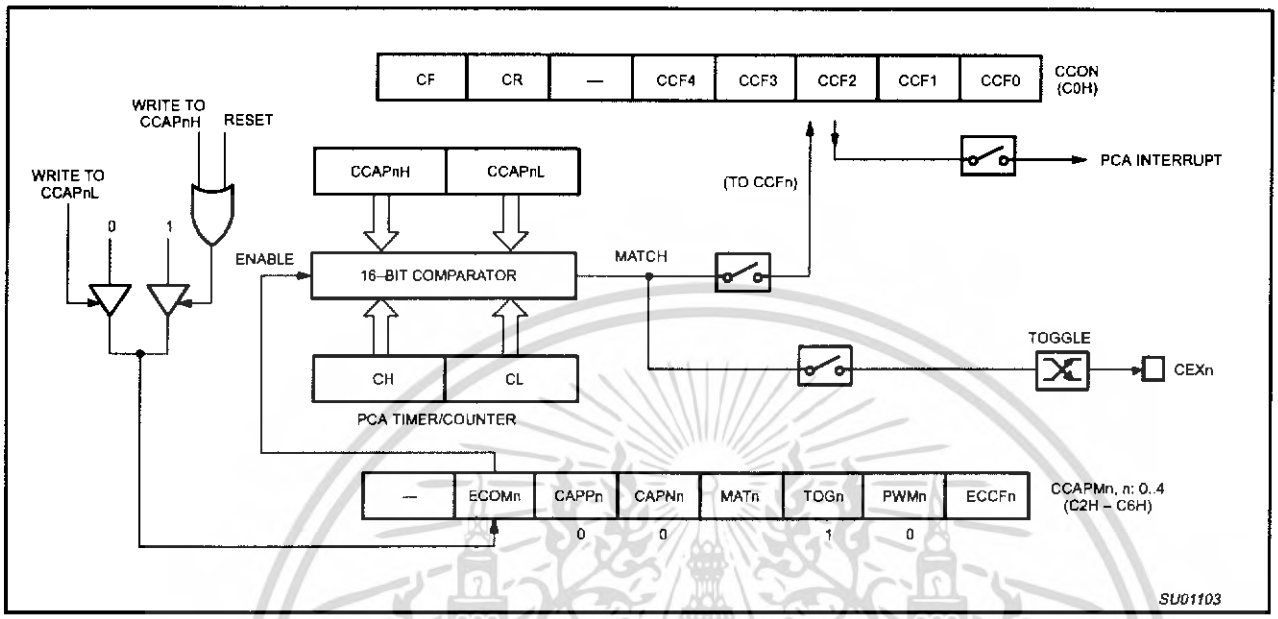


Figure 23. PCA High Speed Output Mode

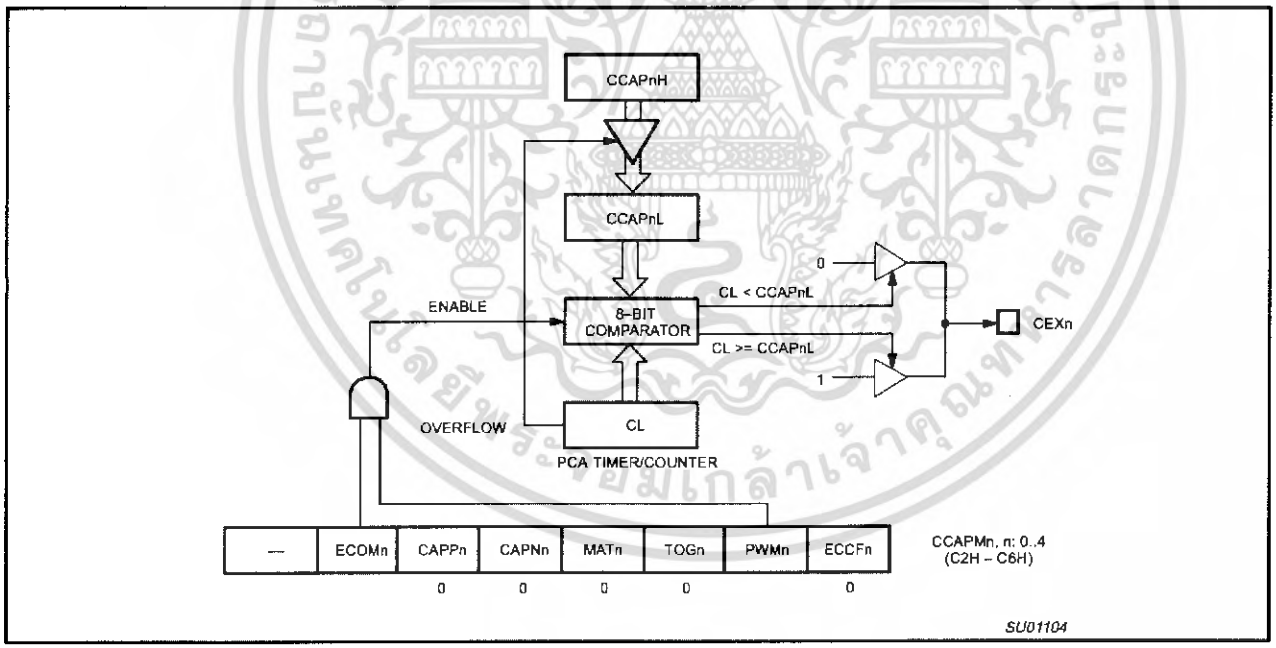


Figure 24. PCA PWM Mode

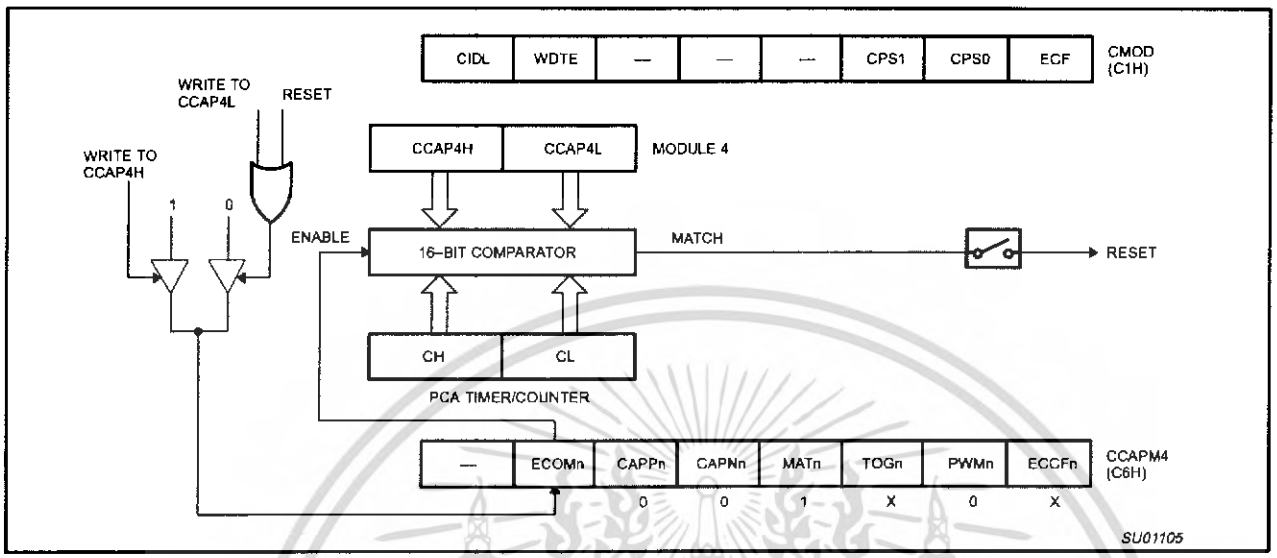


Figure 25. PCA Watchdog Timer m(Module 4 only)

PCA Watchdog Timer

An on-board watchdog timer is available with the PCA to improve the reliability of the system without increasing chip count. Watchdog timers are useful for systems that are susceptible to noise, power glitches, or electrostatic discharge. Module 4 is the only PCA module that can be programmed as a watchdog. However, this module can still be used for other modes if the watchdog is not needed.

Figure 25 shows a diagram of how the watchdog works. The user pre-loads a 16-bit value in the compare registers. Just like the other compare modes, this 16-bit value is compared to the PCA timer value. If a match is allowed to occur, an internal reset will be generated. This will not cause the RST pin to be driven high.

In order to hold off the reset, the user has three options:

1. periodically change the compare value so it will never match the PCA timer,
2. periodically change the PCA timer value so it will never match the compare values, or
3. disable the watchdog by clearing the WDTE bit before a match occurs and then re-enable it.

The first two options are more reliable because the watchdog timer is never disabled as in option #3. If the program counter ever goes astray, a match will eventually occur and cause an internal reset. The second option is also not recommended if other PCA modules are being used. Remember, the PCA timer is the time base for **all** modules; changing the time base for other modules would not be a good idea. Thus, in most applications the first solution is the best option.

Figure 26 shows the code for initializing the watchdog timer. Module 4 can be configured in either compare mode, and the WDTE bit in CMOD must also be set. The user's software then must periodically change (CCAP4H,CCAP4L) to keep a match from occurring with the PCA timer (CH,CL). This code is given in the WATCHDOG routine in Figure 26.

This routine should not be part of an interrupt service routine, because if the program counter goes astray and gets stuck in an infinite loop, interrupts will still be serviced and the watchdog will keep getting reset. Thus, the purpose of the watchdog would be defeated. Instead, call this subroutine from the main program within 2^{16} count of the PCA timer.

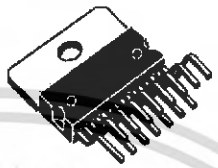
```
INIT_WATCHDOG:
   MOV CCAPM4, #4CH          ; Module 4 in compare mode
   MOV CCAP4L, #0FFH         ; Write to low byte first
   MOV CCAP4H, #0FFH         ; Before PCA timer counts up to
                             ; FFFF Hex, these compare values
                             ; must be changed
   ORL CMOD, #40H            ; Set the WDTE bit to enable the
                             ; watchdog timer without changing
                             ; the other bits in CMOD
;
;*******************************************************************
;
; Main program goes here, but CALL WATCHDOG periodically.
;
;*******************************************************************
;
WATCHDOG:
   CLR EA                    ; Hold off interrupts
   MOV CCAP4L, #00           ; Next compare value is within
   MOV CCAP4H, CH            ; 255 counts of the current PCA
   SETB EA                   ; timer value
   RET
```

Figure 26. PCA Watchdog Timer Initialization Code

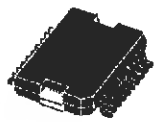
L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15



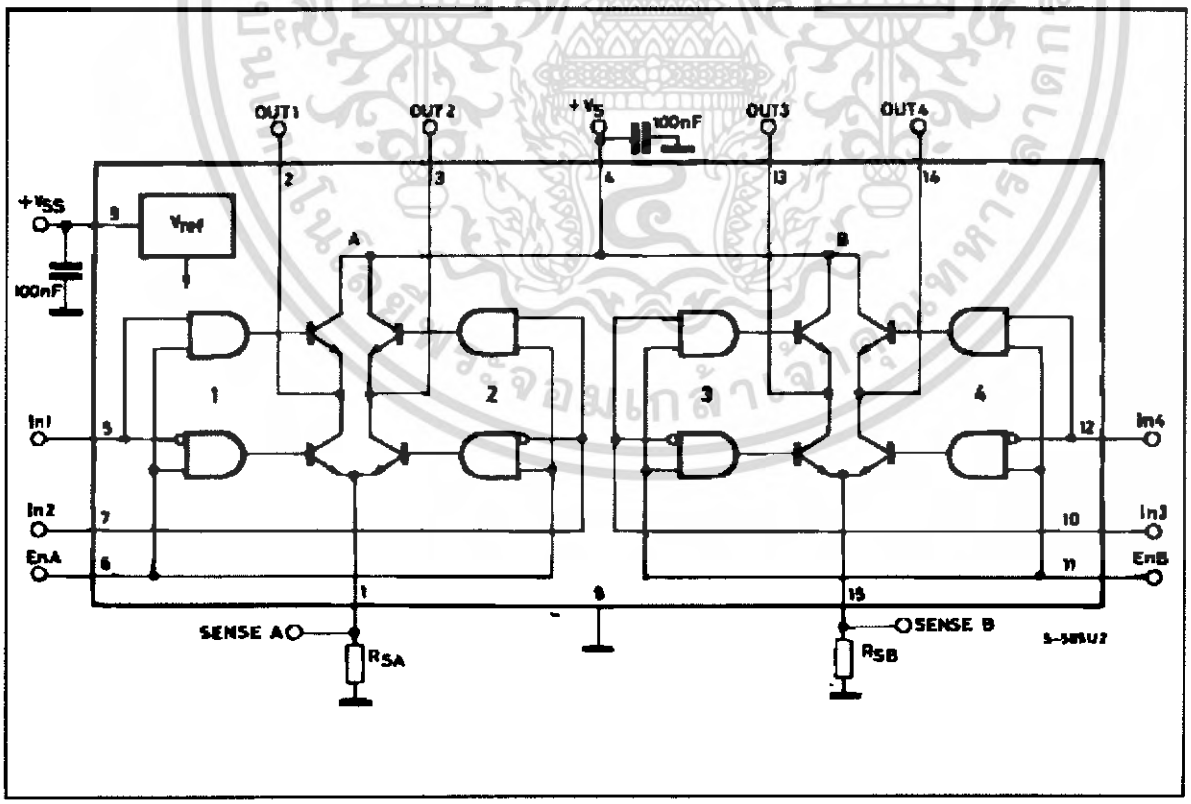
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

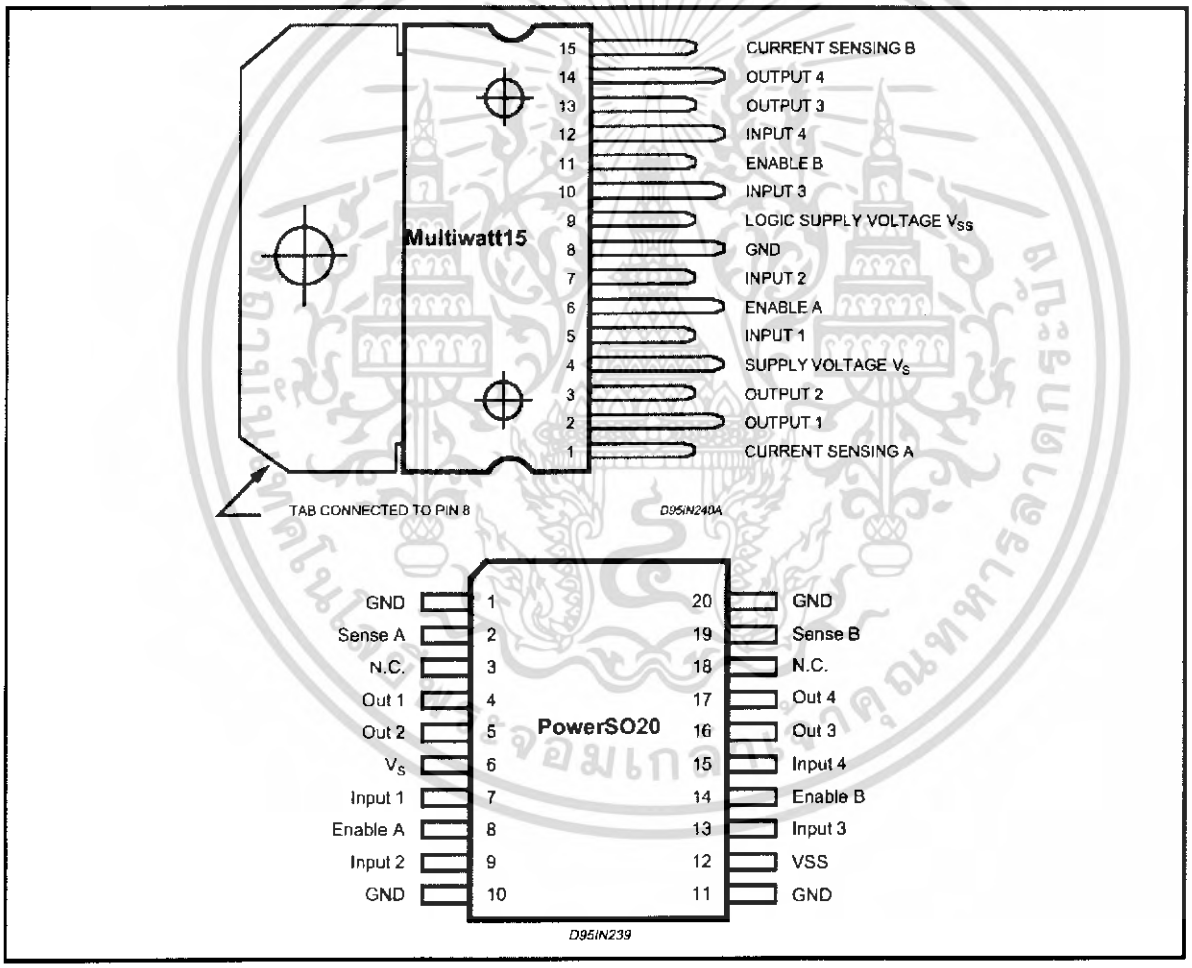
BLOCK DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i, V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)



THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V_i)	Source Current Turn-off Delay	0.5 V_i to 0.9 I_L (2); (4)		1.5		µs
T_2 (V_i)	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		0.2		µs
T_3 (V_i)	Source Current Turn-on Delay	0.5 V_i to 0.1 I_L (2); (4)		2		µs
T_4 (V_i)	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.7		µs
T_5 (V_i)	Sink Current Turn-off Delay	0.5 V_i to 0.9 I_L (3); (4)		0.7		µs
T_6 (V_i)	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.25		µs
T_7 (V_i)	Sink Current Turn-on Delay	0.5 V_i to 0.9 I_L (3); (4)		1.6		µs
T_8 (V_i)	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.2		µs
fc (V_i)	Commutation Frequency	I_L = 2A		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (2); (4)		3		µs
T_2 (V_{en})	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		1		µs
T_3 (V_{en})	Source Current Turn-on Delay	0.5 V_{en} to 0.1 I_L (2); (4)		0.3		µs
T_4 (V_{en})	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.4		µs
T_5 (V_{en})	Sink Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (3); (4)		2.2		µs
T_6 (V_{en})	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.35		µs
T_7 (V_{en})	Sink Current Turn-on Delay	0.5 V_{en} to 0.9 I_L (3); (4)		0.25		µs
T_8 (V_{en})	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.1		µs

1) 1)Sensing voltage can be –1 V for t ≤ 50 µsec; in steady state V_{sens} min ≥ –0.5 V.
2) See fig. 2.
3) See fig. 4.
4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

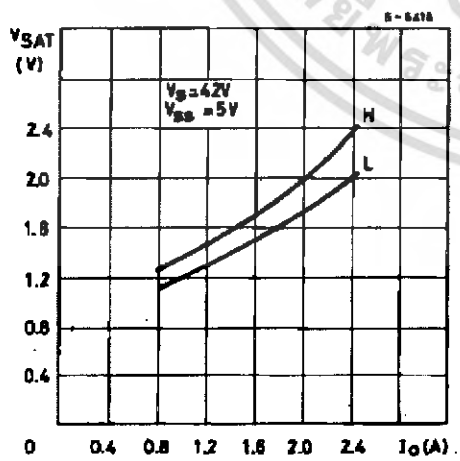
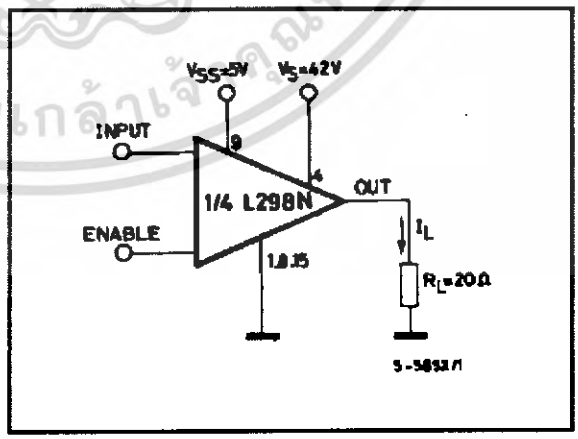


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.

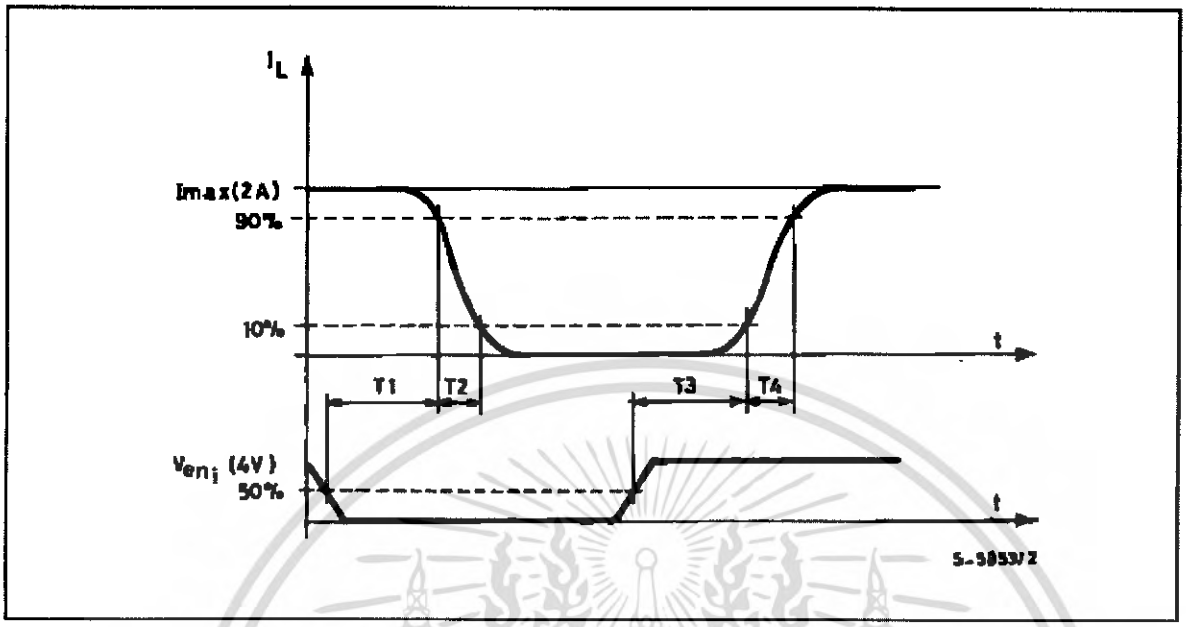
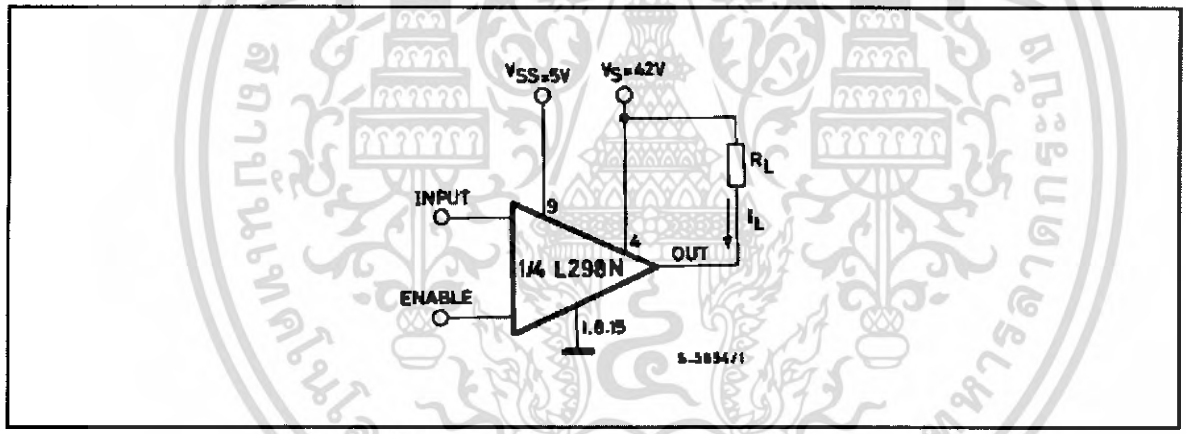


Figure 4 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

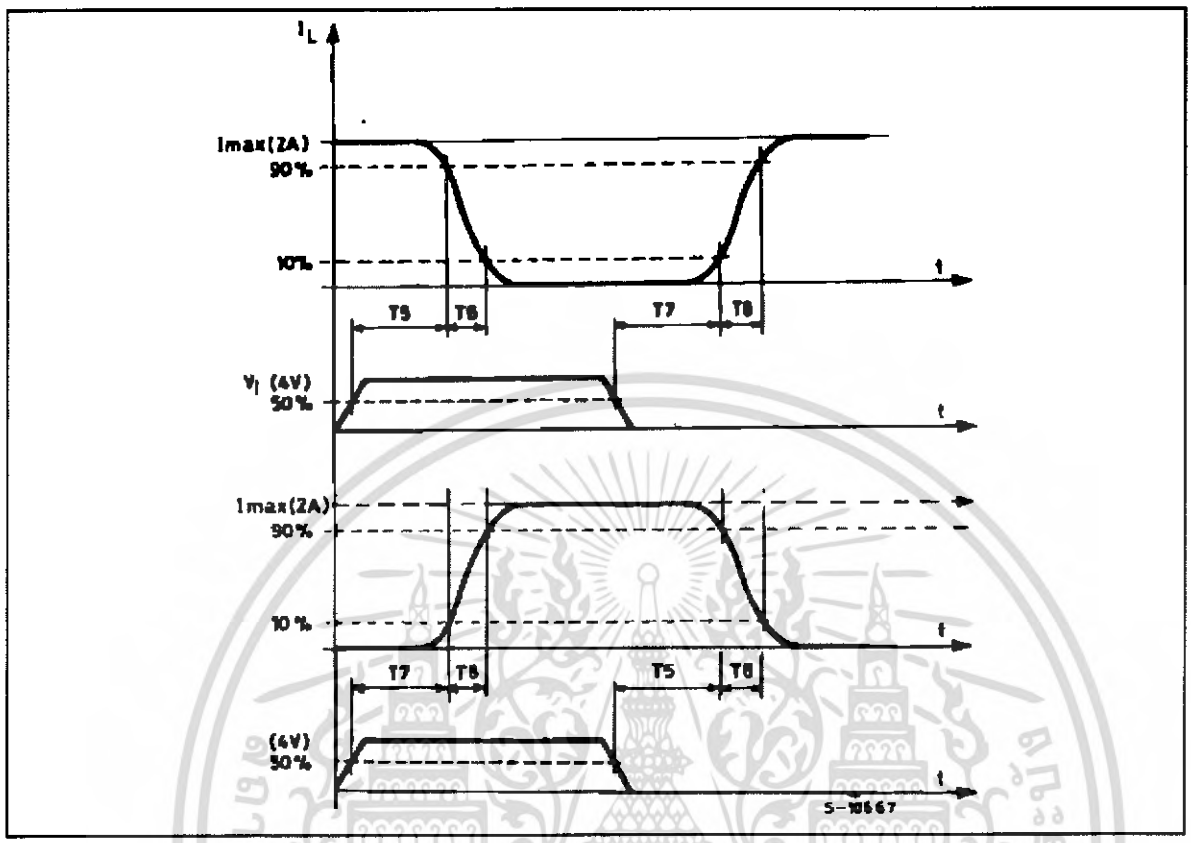
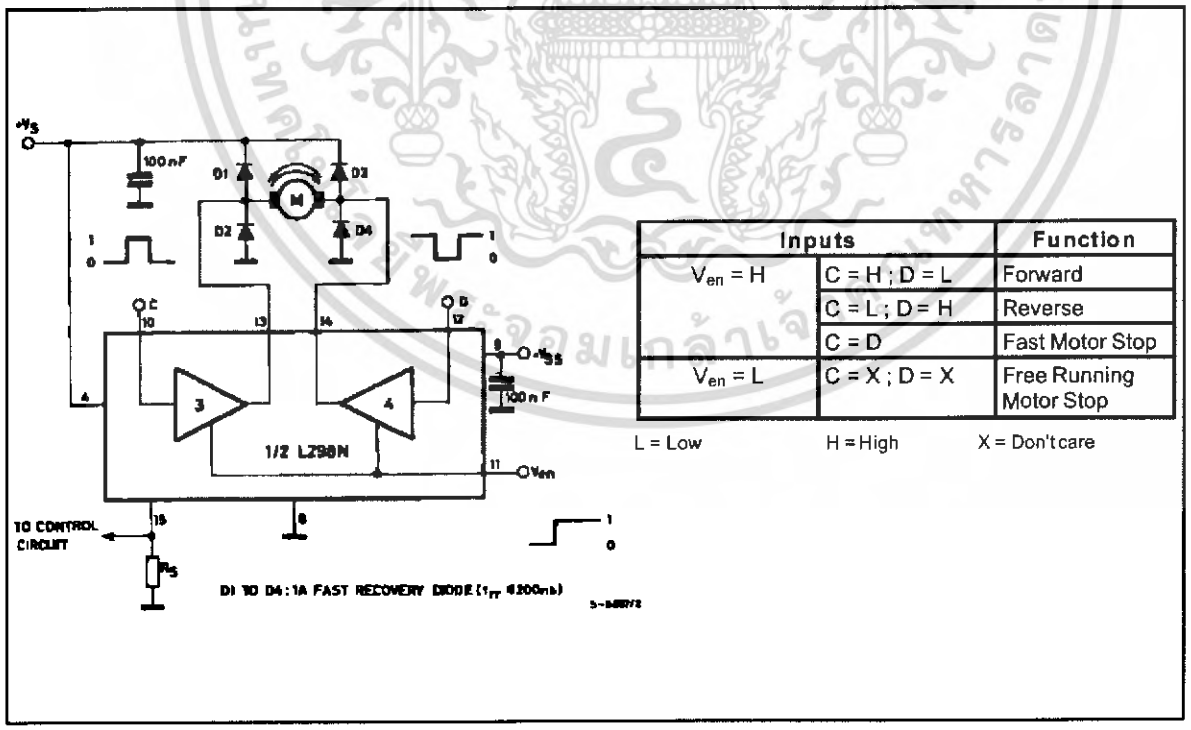


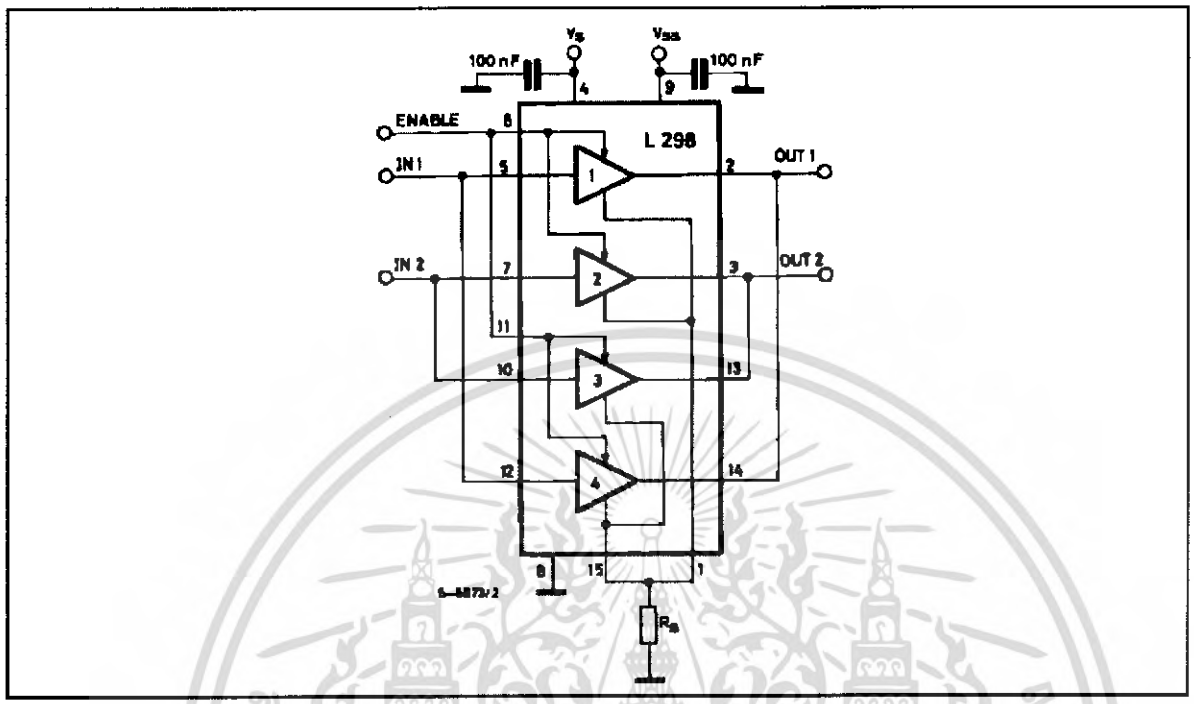
Figure 6 : Bidirectional DC Motor Control.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A ; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output : an external resistor (R_{SA} ; R_{SB}.) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ($t_{rr} \leq 200$ nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped ; Shottky diodes would be preferred.

This solution can drive until 3 Amps In DC operation and until 3.5 Amps of a repetitive peak current.

On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.

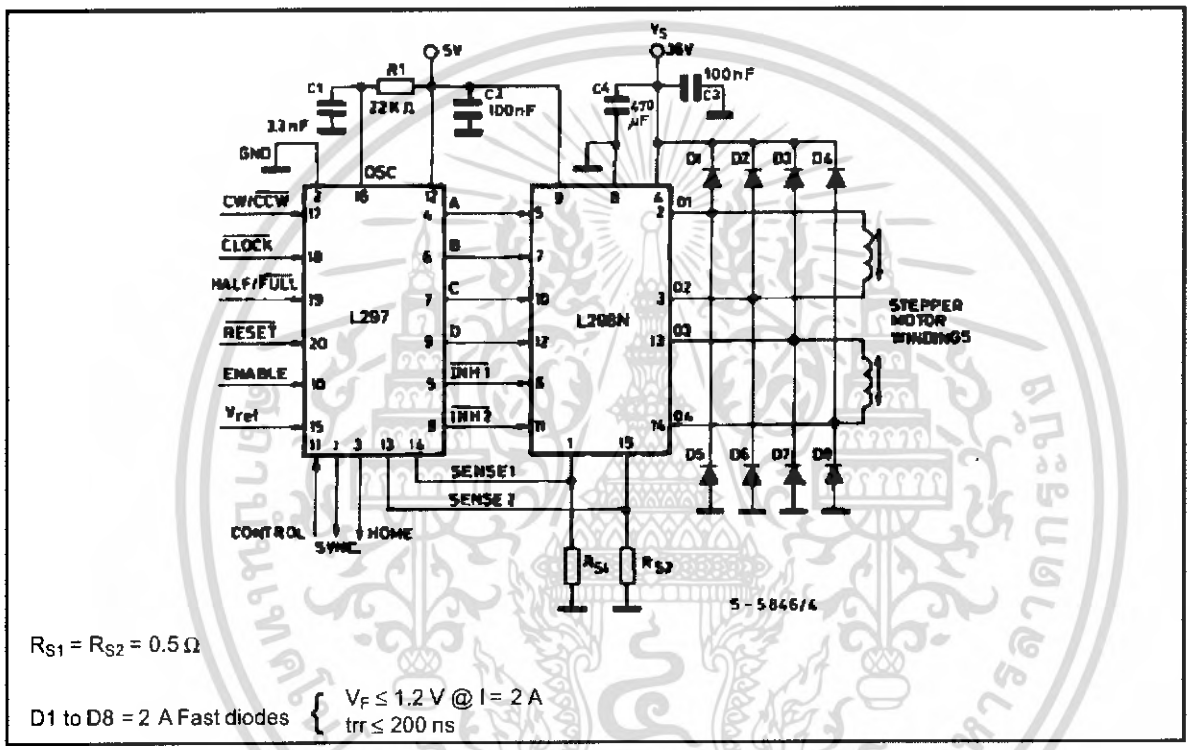


Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

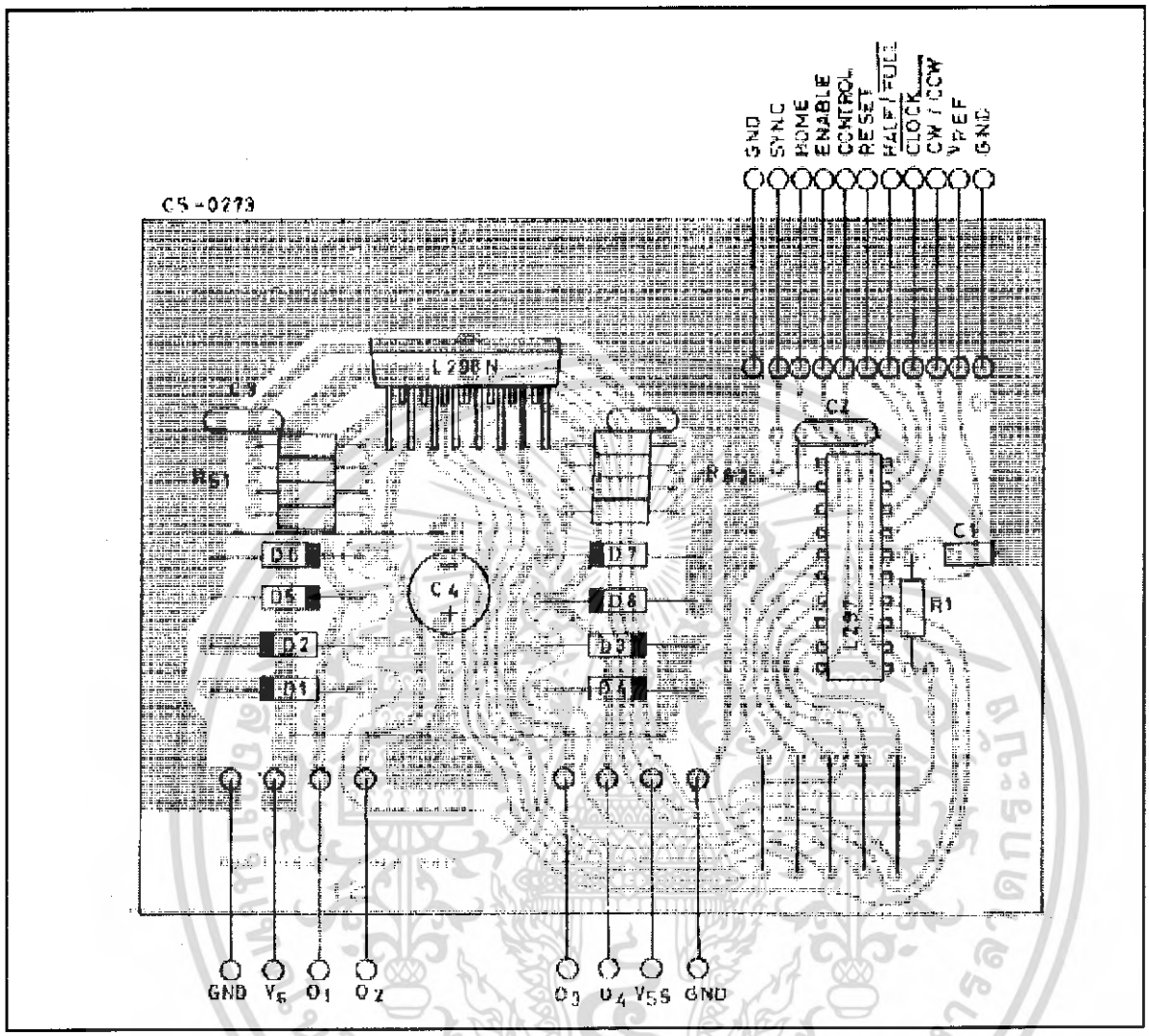
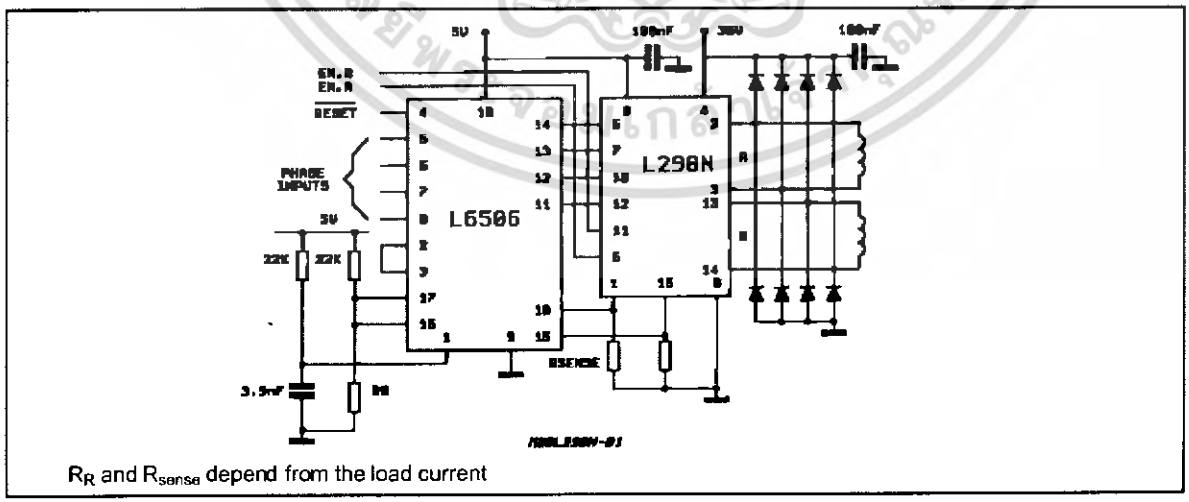


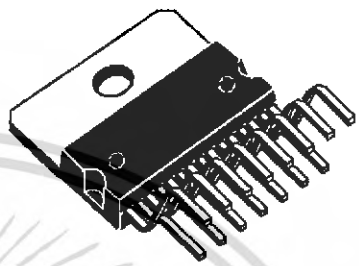
Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.



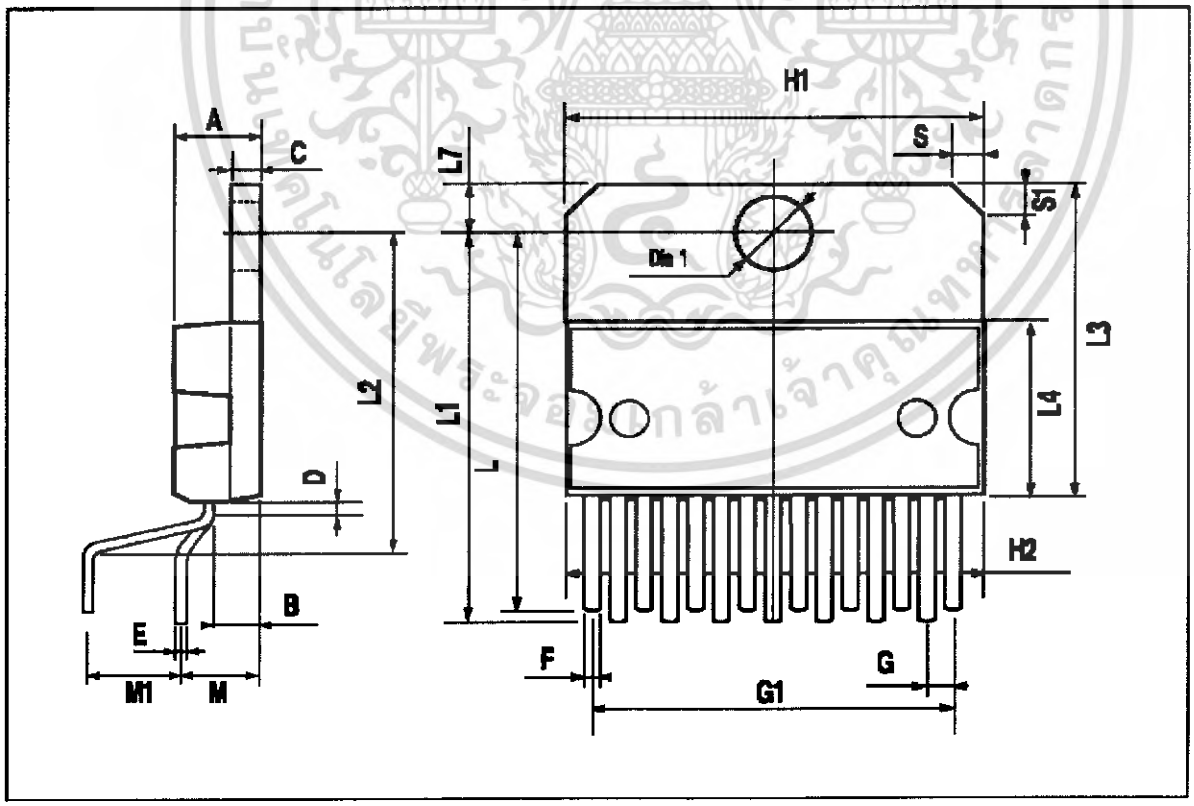
R_R and R_{sense} depend from the load current

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA

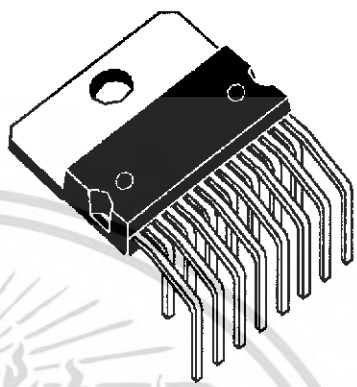


Multiwatt15 V

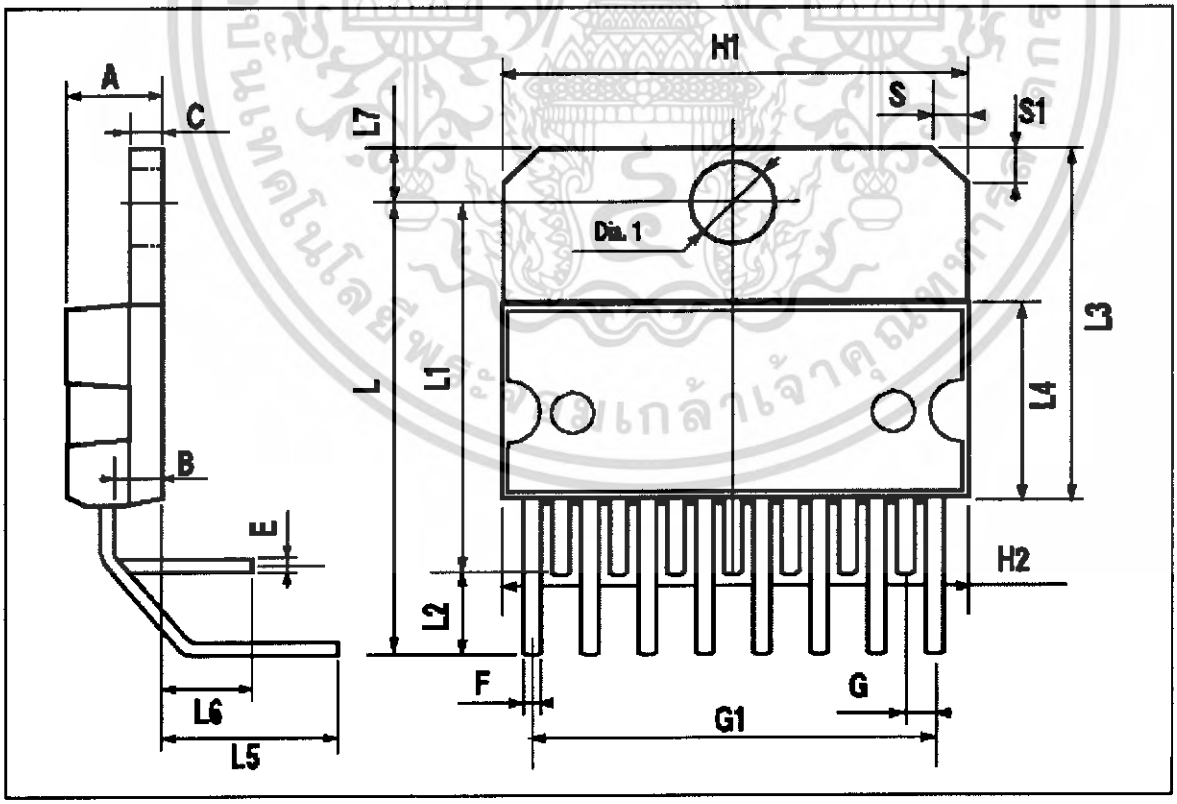


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA



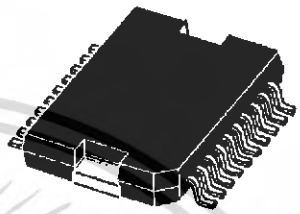
Multiwatt15 H



DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

(1) "D and F" do not include mold flash or protrusions.
- Mold flash or protrusions shall not exceed 0.15 mm (0.006").
- Critical dimensions: "E", "G" and "a3"

OUTLINE AND MECHANICAL DATA



JEDEC MO-166

PowerSO20