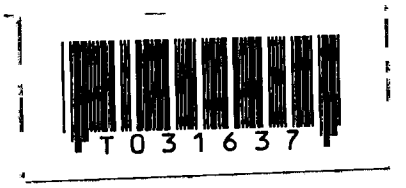
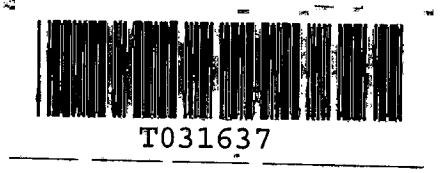
การควบคุมเครื่องใช้ไฟฟ้าบนเวลาจริงผ่านการเชื่อมต่ออินฟราเรด



นางสาวประภาพรรณ วิภาตวิทย์
นางสาวปวีณา ศรีสุวรรณ



โครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต
ภาควิชาฟิสิกส์ประยุกต์
คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
ปีการศึกษา 2540

REAL-TIME CONTROLLER WITH IR LINK

Miss. Praphaphan Wipatawit

Miss. Paweena Srisuwan

A Special Project Submitted in Partial Fulfillment of the Requirement for

the Degree of Bachelor of Science

Department of Applied Physics Faculty of Science

King Mongkut's Institute of Technology Ladkrabang 1997

หัวข้อโครงงานพิเศษ	การควบคุมเครื่องใช้ไฟฟ้าบนเวลาจริงผ่านการเชื่อมต่ออินฟราเรด
โดย	นางสาวประภาพรรณ วิภาตวิทย์
	นางสาวปวีณา ศรีสุวรรณ
อาจารย์ที่ปรึกษา	ผศ. วิชิต ศิริโชติ
	ผศ. ดร. ศิริศักดิ์ เตชะทวีกุล

ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง
อนุมัติให้นับโครงงานพิเศษนี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรวิทยาศาสตรบัณฑิต

---------- หัวหน้าภาควิชาฟิสิกส์ประยุกต์

(รองศาสตราจารย์ สุรพล รักวิจัย)

คณะกรรมการสอบโครงงานพิเศษ

---------- กรรมการ

(ผู้ช่วยศาสตราจารย์ วิชิต ศิริโชติ)

---------- กรรมการ

(ผู้ช่วยศาสตราจารย์ ดร. ศิริศักดิ์ เตชะทวีกุล)

---------- กรรมการ

(อาจารย์ ปิติพร ถนอมงาม)

---------- กรรมการ

(อาจารย์ ภัทรียา กิตติเดชาชาญ)

ลิขสิทธิ์ภาควิชาฟิสิกส์ประยุกต์ คณะวิทยาศาสตร์
สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

หัวข้อโครงงานพิเศษ	การควบคุมเครื่องใช้ไฟฟ้าบนเวลาจริงผ่านการเชื่อมต่ออินฟราเรด
ชื่อนักศึกษา	นางสาวประภาพรรณ วิภาตวิทย์ นางสาวปวีณา ศรีสุวรรณ
อาจารย์ที่ปรึกษา	ผู้ช่วยศาสตราจารย์ วิชิต ศิริโชติ ผู้ช่วยศาสตราจารย์ ดร. ศิริศักดิ์ เตชะทวีกุล
ภาควิชา	ฟิสิกส์ประยุกต์
ปีการศึกษา	2540

บทคัดย่อ

ตัวควบคุมเวลาจริงผ่านการเชื่อมต่ออินฟราเรดได้ถูกออกแบบและสร้างขึ้นเพื่อให้ตัวบังคับนี้ไปควบคุมเครื่องใช้ไฟฟ้าตามเวลาที่กำหนดไว้เฉพาะ โดยมีส่วนประกอบหลักดังนี้ 1) ไมโครคอนโทรลเลอร์แบบชิพเดี่ยวเบอร์ 89C2051 ที่ทำหน้าที่เป็นนาฬิกาโดยให้ข้อมูลออกมาเป็น 8 ช่องสัญญาณเพื่อควบคุม 2) ตัวส่งสัญญาณอินฟราเรดเบอร์ MC145027 สัญญาณควบคุมที่ออกมาจาก 89C2051 จะถูกเข้ารหัส เพื่อเปลี่ยนเป็นสัญญาณแบบพัลส์อนุกรม ข้อมูลแบบอนุกรมนี้จะถูกมอดูเลตโดยพาหะที่มีความถี่ 50 กิโลเฮิร์ต จากนั้นนำสัญญาณที่ได้นี้ไปขับส่งโดยใช้ LED อินฟราเรดไปยังตัวรับ ตัวรับอินฟราเรดจะทำการถอดรหัสขาเข้าที่เป็นแบบอนุกรม แล้วนำออกมาเป็นสัญญาณ 4 บิต เพื่อใช้ควบคุมรีเลย์แบบโซลิดสเตท การสื่อสารแบบไร้สายหรือการสื่อสารโดยใช้อินฟราเรดจะทำให้อุปกรณ์ของเราติดตั้งง่ายและมีราคาถูก

Special Project Title	Real-Time Controller with IR link
Name	Miss Praphaphan Wipatawit
	Miss Paweena Srisuwan
Special Project Advisors	Asst. Prof. Vichit Sirichot
	Asst. Prof. Dr. Sirisak Tachataveekul
Department	Applied Physics
Academic Year	1997

Abstract

The real-time controller with wireless link employing infrared communication has been designed and built. The controller can be used to control electrical appliances at a specified time. Main circuit consists of 1) a single chip microcomputer, 89C2051 emulating real-time clock with 8 channels control output 2) an infrared transmitter, MC145026 and 3) an infrared receiver, MC145027. The output control bit from 89C2051 enables the encoder chip to convert the input settings to a serial pulse data. The serial data is then modulated by a 50 kHz carrier frequency. This signal was used to drive an infrared LED sending the data to receivers. The infrared receiver decodes serial data to receivers. The infrared receiver decodes serial data and provides 4-Bits output used to control solid-state relays. The wireless link via infrared communication makes the controller installation easy and low-cost

กิตติกรรมประกาศ

โครงงานพิเศษนี้สามารถเสร็จสมบูรณ์ได้ด้วยความช่วยเหลือจากบุคคลต่างๆ ดังนี้

บุพการี	ที่ให้ความอุปการะในทุกๆ ด้านจนสำเร็จการศึกษา
ผศ. วิชิต ศิริโชติ	ที่ให้คำแนะนำต่างๆ และให้ความอนุเคราะห์อุปกรณ์ต่างๆ ในการทำโครงงานพิเศษนี้
ผศ. ดร. ศิริศักดิ์ เตชะทวีกุล	ที่ให้ความอนุเคราะห์ทางด้านซอฟแวร์
คณะกรรมการทุกท่าน	ที่ตรวจรายงานโครงงานพิเศษฉบับนี้
ภาควิชาฟิสิกส์ประยุกต์	ที่ให้ความอนุเคราะห์เครื่องมือและอุปกรณ์ต่างๆ
เพื่อนทุกๆ คน	ที่คอยให้กำลังใจและให้ความช่วยเหลือต่างๆ

สารบัญ

สารบัญตาราง

สารบัญรูปภาพ

หน้า

หน้า

บทที่ 1
บทนำ

1.1 คำนำ

การดำรงชีวิตมนุษย์ในปัจจุบันล้วนยึดติดกับสิ่งอำนวยความสะดวก ของเครื่องใช้ไฟฟ้าที่ทันสมัย มากมายโดยไม่คำนึงถึงพลังงานและทรัพยากรอันมีอยู่จำกัดในเวลานี้ ดังนั้นเราจึงควรจะทำการประหยัดพลังงานให้ได้มากที่สุด บวกเข้ากับวิวัฒนาการอันทันสมัย คงหลีกไม่พ้นการควบคุมระยะไกลที่สามารถตั้งเวลาเปิด-ปิดได้และใช้การสื่อสารเป็นแบบไร้สายเพื่อเป็นการประหยัดทรัพยากรธรรมชาติให้มากที่สุด โดยเราจะเริ่มจากระบบง่าย ๆ ที่ใช้ในงานกันอยู่ทั่วไป คือ การใช้คำสั่งงานจริงผ่านการประมวลผลของไมโครคอนโทรลเลอร์แบบชิพเดี่ยว และอาศัยหลักการสื่อสารย่านแสงอินฟราเรดที่นิยมใช้กันอย่างแพร่หลายและไม่เป็นอันตรายในการสื่อสารเป็นตัวนำพาข้อมูลไป โครงงานพิเศษชุดนี้ใช้หลักการควบคุมแบบเวลาจริงซึ่งสามารถที่จะทำการเลือกเวลาเปิด-ปิดได้ตามเวลาที่ต้องการและเราสามารถที่จะควบคุมอุปกรณ์ไฟฟ้าต่าง ๆ ได้จากการสั่งผ่านคอมพิวเตอร์ที่มีคำสั่งในการใช้งานแบบ GUI ทำให้เกิดความสวยงามและง่ายต่อการใช้และความเข้าใจของผู้ใช้เอง จะเห็นว่าโครงงานพิเศษชุดนี้สามารถช่วยให้เราประหยัดพลังงานที่เกิด จากการใช้พลังงานไฟฟ้าอย่างสิ้นเปลืองลงได้

1.2 วัตถุประสงค์

1. เพื่อศึกษาการใช้หลักการทำงานควบคุมระยะไกล แบบมัลติพอยต์ (multi-point control system)
2. เพื่อศึกษาการใช้หลักการทำงานโดยใช้ลำแสงอินฟราเรด
3. เพื่อใช้คอมพิวเตอร์เป็นตัวควบคุมและแสดงผลระบบการควบคุมแบบเวลาจริง
4. เพื่อนำหลักการทางฟิสิกส์ไปประยุกต์ใช้งานสร้างเครื่องมือเพื่อความสะดวกในชีวิตประจำวัน
5. เพื่อเป็นแนวทางในการพัฒนาเครือข่ายอินฟราเรดให้มีประสิทธิภาพมากยิ่งขึ้น

1.3 ขอบเขตของการทำโครงงานพิเศษ

1. เราจะต้องทำการตั้งค่าวันและเวลาให้กับไมโครคอนโทรลเลอร์
2. ตั้งค่าวันและเวลาที่ต้องการให้อุปกรณ์ไฟฟ้าภายในห้องเปิด-ปิด
3. สามารถเลือกอ่านโปรแกรมที่มีอยู่ในไมโครลคอนโทรลเลอร์ก่อนแล้วได้
4. ทำการเลือกห้องและตั้งค่าเปิด-ปิดให้กับอุปกรณ์ไฟฟ้า
5. ทำการส่งข้อมูลให้กับไมโครคอนโทรลเลอร์
6. เมื่อถึงวันและเวลาดังกล่าวแล้วอุปกรณ์ไฟฟ้าจะเปิด-ปิดตามคำสั่ง

1.4 วิธีที่ใช้ในการดำเนินการ

1. ส่วนของวงจร Real-Time Controller จะควบคุมโดยการเขียนภาษาแอสแซมบลี
2. เราจะได้จำนวนสัญญาณที่ออกมานำไปเข้ารหัสเพื่อต่อเข้ากับการส่งสัญญาณผ่านตัวกลางที่เป็นแสงอินฟราเรดที่ภาคส่ง
3. สัญญาณอินฟราเรดที่ตรวจจับได้โดยภาครับ เราก็จะนำสัญญาณทางแสงมาแปลงให้เป็นสัญญาณทางไฟฟ้าแล้วนำเอาสัญญาณมาควบคุมอุปกรณ์ไฟฟ้า
4. ทำการประยุกต์ในการใช้งานให้ง่ายขึ้นโดยใช้การทำงานแบบ GUI โดยการประยุกต์ด้วยภาษา Visual Basic เพื่อให้ผู้ใช้สามารถใช้งานได้ง่าย

บทที่ 2

ระบบ REAL-TIME และการเชื่อมต่อแบบอินฟราเรด

2.1 ประวัติความเป็นมา

ในค.ศ 1800 เซอร์ วิลเลี่ยม เฮอร์เชล ได้ค้นพบ การแผ่รังสีนอกเหนือจากย่านสเปกตรัมที่ตามองเห็นโดยหากนำเทอร์โมมิเตอร์ไปวางไว้เหนือย่านสีแดงของสเปกตรัมที่มองเห็นได้ ซึ่งมาจากปรีซึม แล้ว เฮอร์เชลได้แสดงว่าสามารถตรวจพบค่าพลังงานที่เป็นความร้อนเกิดขึ้น ซึ่งเราสามารถเห็นแสงนั้นได้

เขาได้ปรับปรุงเนื้อหาของการแผ่รังสีแบบนี้ แล้วพบว่า แสง อินฟราเรดจะประพฤติเหมือนกฎของแสงย่านมองเห็นได้ อย่างไรก็ตามในปี 1830 ได้พัฒนาเครื่องตรวจจับการแผ่รังสีชนิดนี้ได้ โดยใช้รากฐานจากเทอร์โมคัปเปิล และเรียกว่า เทอร์โมพาย

โซโลมิเตอร์ ซึ่งอาศัยโลหะที่มีคุณสมบัติค่าความต้านทานเปลี่ยนไปตามอุณหภูมิ ซึ่งทำขึ้นในปี 1880 และได้ปรับปรุงให้เป็นเครื่องตรวจจับอินฟราเรด

ในระหว่างปี 1870 และ 1920 มีการพัฒนาเทคโนโลยีมากขึ้น ทำให้เกิดการพัฒนาเครื่องตรวจจับควอนตัมอันดับที่ 1 ซึ่งมีรากฐานมาจากทำปฏิกิริยาระหว่างการแผ่รังสีและอาศัยการแปลงโดยตรงจากการแผ่รังสีไปเป็นสัญญาณไฟฟ้า

ตัว Photoconductivity หรือ Photovoltaic detectors พบว่าสามารถตอบสนองได้เร็วกว่าและมีความไวสูงกว่า และต่อไปนี้คือบัญชีที่มีการค้นพบภายหลัง

1930-1944 พัฒนาตัวตรวจนับแบบตะกั่วซัลไฟด์ (Pbs) โดยเฉพาะทางการทหาร ตัวตรวจจับเหล่านี้มีความไวในช่วง 1.3-3 μm

1940-1950 ขยายช่วงสเปกตรัมไปอยู่ที่อินฟราเรด3-5 μm โดยใช้ indium antimonide (InSb)

1960 มีการสำรวจอินฟราเรดที่ 8-14 μm โดยใช้ตัวตรวจจับแบบ Mercury - tellurium-cadmium detectors (Hg Te cd)

ตัวตรวจจับแบบหลังสุดนี้ต้องการการหล่อเย็น เพราะความที่มีความไวสูงและตอบสนองในเวลาสั้น ตัวตรวจจับนี้จะนำไปสู่การพัฒนา ระบบ thermal inoying ซึ่งนำไปทำการตรวจจับการแผ่รังสีอินฟราเรดที่แผ่มาโดย อนุภาคในช่วง 2-15 μm

ความยาวคลื่น ของรังสีที่แผ่ออกมาจะแปรผันตรงข้ามกับการเคลื่อนที่ของพลังงาน ซึ่งเป็นข้อสำคัญที่ต้องจดจำในเรื่องอินฟราเรด ที่ความยาวคลื่นยาวจะมีพลังงานการแผ่รังสีต่ำ ซึ่งจะช่วยในการตรวจจับการแผ่รังสีอินฟราเรด

ถ้าวัสดุยอมให้เกิดการเคลื่อนย้าย (การกระตุ้นโดยอุณหภูมิของโมเลกุล) แต่ละอะตอมจะหาจำนวนของพลังงานการแผ่รังสีที่แน่นอนซึ่งสถิตย์อยู่ได้จากค่าที่เป็นไปได้ การกระจายของความยาวคลื่นเป็นแบบเดียวรวมทั้งการแผ่รังสี จึงเรียกว่ามีสเปกตรัมต่อเนื่อง

โดยหลักใหญ่จะพูดถึงการเคลื่อนย้ายแบบ Well- defined (การเคลื่อนย้ายควอนตัมของอิเลคตรอนในอะตอม) การแผ่รังสีจะออกมาในแบบที่ไม่ต่อเนื่อง และพูดได้ว่ามีเส้นสเปกตรัม

การดูดกลืนของการแผ่รังสีโดยวัสดุจะเกิดจากระบวนการตรงข้าม ซึ่งจะมีค่าการดูดกลืนมากหรือน้อยขึ้นกับความยาวคลื่นและวัตถุ สเปกตรัมของการแผ่รังสีจะถูกหารไปเป็นจำนวนขอบเขต ซึ่งตอบสนองช่วงการทำงานของแหล่งกำเนิดและตัวตรวจจับ

ตารางข้างล่างแสดงสเปกตรัมคลื่นแม่เหล็กไฟฟ้า

cosmic ray	Gamma ray	X-ray	UV	visible	Infrared	Radio						
								UHF	VHF	HF	MF	LF	

ตารางที่ 2.1 แสดงเส้นสเปกตรัมคลื่นแม่เหล็กไฟฟ้า

สเปกตรัมของอินฟราเรดจะถูกแผ่ที่อุณหภูมิปกติที่พบบนผิวโลก ที่ทุกตัวมีการแผ่รังสีที่อุณหภูมิดังกล่าว อินฟราเรดจะไม่ออกมาจากวัตถุถ้ามันมีอุณหภูมิต่ำ เพื่อลดจำนวนการแผ่รังสีที่ให้ความยาวคลื่น 1 μm โดยวัตถุที่อุณหภูมิ 0°C ลง 1 % อุณหภูมิจะต้องลดลงโดย 100 องศา

สเปกตรัมของอินฟราเรดจะแบ่งได้ เป็น 3 เขตใหญ่ๆ ตามตัวตรวจจับที่ใช้จับที่ใกล้ infrared จะถูกจับโดย Photographid emulsion โดย photoemission cell และโดย Photoconductive และ Photovoltaic detection และการแผ่รังสีในย่านอินฟราเรด จะวัดโดย Thermal detectors

visible	Near infrared	Middle infrared	Far infrared
0.35 um	0.75 um	20 um	1000 um

ตารางที่ 2.2 ความยาวคลื่นในย่านอินฟาเรด

ความยาวคลื่นอินฟราเรด ปกติจะวัดในหน่วยไมครอน จำนวนคลื่นที่สังเกตได้ในสเปกโตรสโคปจะได้จาก

$$\sigma = 1/\lambda$$

λ เป็นเมตรหรือ

$$\lambda = 10/\lambda$$

λ ในหน่วยไมครอน

um	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
cm^{-1}	10000	5000	3333	2500	2000	1667	1429	1250	1111	1000	909	833	769	714	668

ตารางที่ 2.3 แสดงค่าความยาวคลื่นและส่วนกลับค่าต่างๆ

2.2 แหล่งกำเนิดอินฟราเรดชนิดต่าง ๆ

2.2.1 ระบบของสัญญาณควบคุม

ปัจจุบันนี้การแบ่งประเภทการควบคุมของระบบรีโมตคอนโทรล อาจจำแนกออกได้เป็น 2 ประเภท ตามลักษณะของการส่งผ่านสัญญาณ หรือตัวกลางในการเชื่อมโยงสัญญาณ

<u>ระบบใช้สาย</u>

เป็นระบบควบคุมที่ต้องมีอุปกรณ์นำสัญญาณจากตัวส่ง (สถานีส่ง) ไปยังตัวรับ(สถานีรับ) อุปกรณ์การนำสัญญาณนั้นขึ้นอยู่กับชนิดของสัญญาณพาหะ ซึ่งอาจได้แก่สัญญาณไฟฟ้าและสัญญาณแสง เป็นต้น

ในกรณีของสัญญาณควบคุมที่เป็นสัญญาณไฟฟ้า อุปกรณ์นำสัญญาณอาจได้แก่ สายไฟ , สายโทรศัพท์ , สายโคแอกเชียล เป็นต้น แต่ถ้าในกรณีของสัญญาณควบคุมที่เป็นแสง อุปกรณ์นำสัญญาณจะเป็นเส้นใยแก้วนำแสง หรือ ไฟเบอร์ออปติก

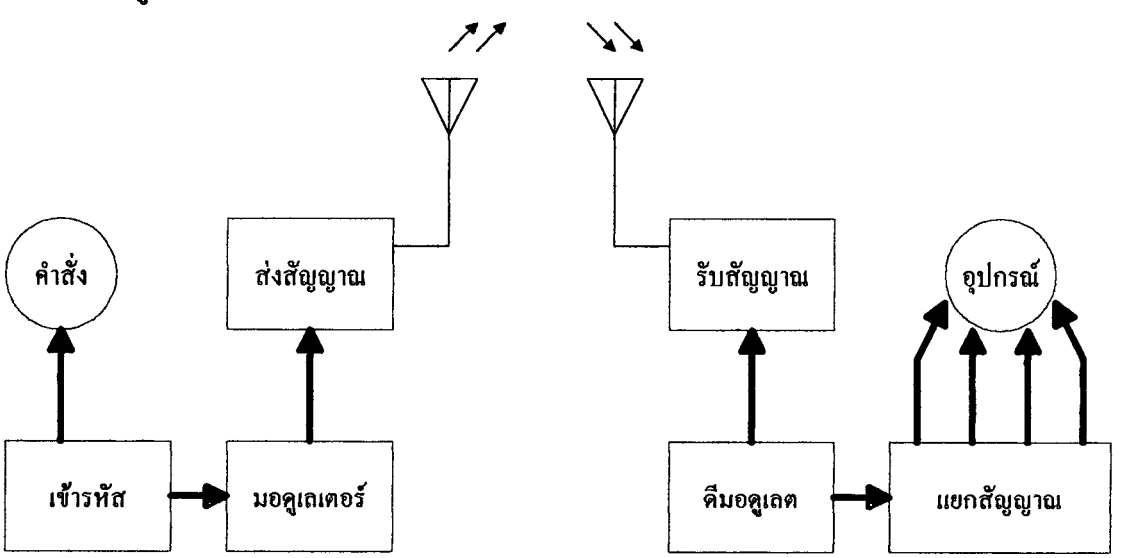
ตัวอย่างของรีโมตคอนโทรลแบบใช้สายไฟ จะเห็นได้จากเครื่องใช้ไฟฟ้ารุ่นเก่า เช่น เครื่องเล่นวีดีโอ แอร์ ที่มีสายไฟเสียบต่อกับรีโมตคอนโทรลออกมาภายนอกเครื่อง ทำให้ไม่มีความสะดวกในการใช้งาน ปัจจุบันจึงไม่ค่อยเป็นที่นิยมอีกแล้ว จะมีก็แต่งานบางประเภทเท่านั้นในส่วนของรีโมตคอนโทรลที่ใช้ไฟเบอร์ออปติก อาจจะไม่ค่อยพบเห็นกันบ่อยนัก เนื่องจากว่าไม่ได้ใช้งานควบคุม แต่ใช้ในการส่งสัญญาณในลักษณะการสื่อสารข้อมูลที่ไม่เกี่ยวกับการควบคุมในทางปลายทางแทนสายส่งสัญญาณแบบธรรมดา เพื่อหลีกเลี่ยงปัญหาที่เกิดขึ้นกับระบบสายไฟธรรมดา ในหลาย ๆด้าน

ระบบไร้สาย

เป็นระบบควบคุมที่ไม่ต้องมีอุปกรณ์ใด ๆ เป็นตัวนำสัญญาณ โดยสัญญาณควบคุมจะเดินทางผ่านไปในอากาศ ชนิดของสัญญาณควบคุมที่เดินทางผ่านไปในอากาศได้ อาจอยู่ในรูปของสัญญาณเสียง สัญญาณแสง และคลื่นวิทยุ

การใช้สัญญาณเสียงเป็นตัวส่งสัญญาณควบคุม ได้กล่าวไว้แล้วในตอนแรก ในส่วนของการใช้สัญญาณแสงกำลังเป็นที่นิยมกัน ในการนำมาเพื่อใช้ควบคุมเครื่องใช้ไฟฟ้าตามบ้านเรือน ดังนั้นเนื้อหาส่วนใหญ่ของบทความนี้จะเน้นที่ระบบนี้เป็นหลัก สำหรับกรณีของการควบคุมด้วยการใช้คลื่นวิทยุนั้น นิยมใช้กับเครื่องเล่นประเภทวิทยุบังคับ เช่น เครื่องบินเล็ก , เรือเร็ว , รถเด็กเล่น เป็นต้น ทั้งนี้ก็เพราะรัศมีทำการของระบบวิทยุบังคับ มีรัศมีการควบคุมที่ไกลมาก ขึ้นอยู่กับกำลังการส่งสัญญาณออกอากาศ และใช้ได้แม้ในพื้นที่คับแคบ, คดเคี้ยววนเวียน แต่รัศมีทำการอาจจะแคบเข้ามาด้วย เพราะคลื่นวิทยุสามารถทะลุผ่านสิ่งกีดขวางได้ สัญญาณควบคุมของระบบวิทยุอาจถูกกำหนดและเข้ารหัสด้วยวิธีเดียวกับระบบรีโมตคอนโทรลทั่วไป เพียงแต่จะถูกนำมามอดูเลตกับคลื่นวิยุที่ภาคส่งก่อน การมอดูเลตสัญญาณใช้หลักการของการสื่อสารทั่วไป เช่น การมอดูเลตแบบ AM FM เป็นต้น

ส่วนของวงจรภาครับ เมื่อได้รับสัญญาณวิทยุก็จะทำการดีมอดูเลต เพื่อแยกชนิดของสัญญาณควบคุมออกมาจากความถี่วิทยุ สำหรับใช้สั่งงานอุปกรณ์ส่วนอื่นต่อไป ดังบล็อกไดอะแกรมรูปที่ 2.1



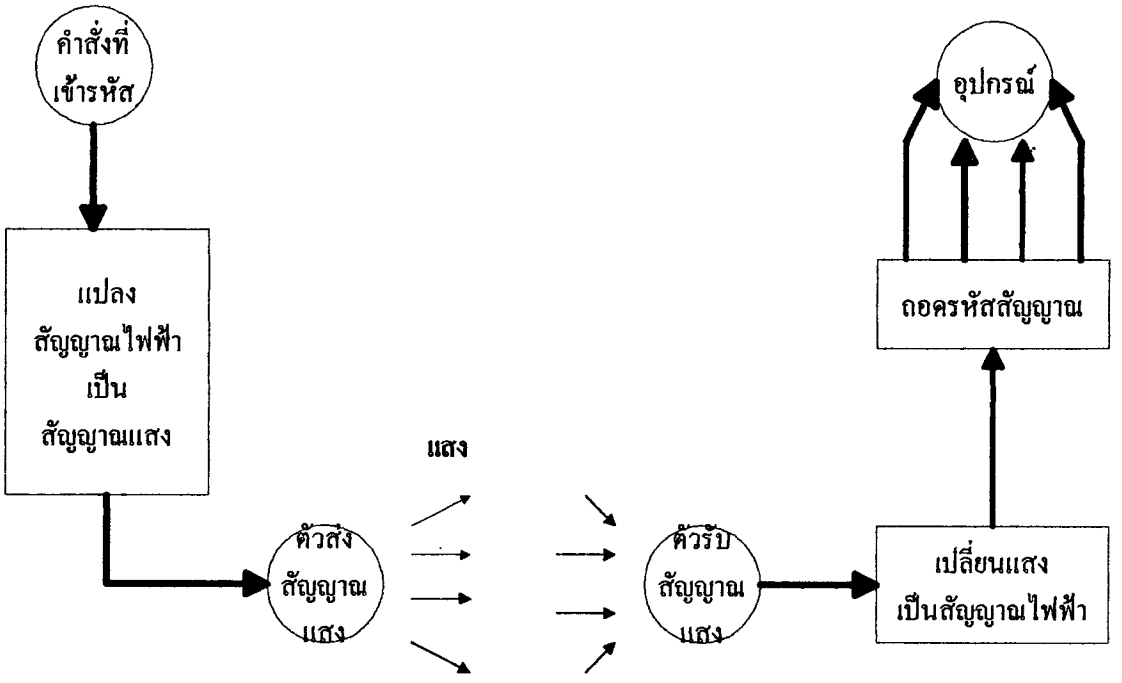
รูปที่ 2.1 บล็อคไดอะแกรมของระบบวิทยุบังคับ

หากทำการเปรียบเทียบระบบรีโมตคอนโทรลชนิดไร้สายแบบใช้แสงกับใช้คลื่นวิทยุแล้ว พอจะสรุปได้ดังตารางที่ 2.4 ซึ่งพอจะเห็นได้ว่าทำไมระบบควบคุมด้วยแสงจึงเป็นที่นิยมใช้กันอย่างแพร่หลายในเครื่องไฟฟ้าที่เกี่ยวข้องกับชีวิตประจำวันของเรา

ข้อเปรียบเทียบ	ควบคุมด้วยแสง	ควบคุมด้วยคลื่นวิทยุ
ส่วนของวงจร	วงจรไม่ซับซ้อนออกแบบง่าย	วงจรค่อนข้างซับซ้อน การออกแบบวงจรค่อนข้างพิถีพิถัน
รัศมีทำการควบคุม	ใกล้ แต่เหมาะกับห้องที่มีฝาผนังเพราะมีการสะท้อนได้ดี	ไกลตามคำสั่งส่ง มีอำนาจทะลุทะลวงสิ่งกีดขวาง เหมาะกับการใช้งานกลางแจ้ง
ปัญหาสัญญาณรบกวน	ไม่มีหรือน้อยมาก	อาจสร้างสัญญาณรบกวนได้กับเครื่องไฟฟ้าอื่นได้ง่ายและถูกรบกวนได้ง่ายเช่นกัน
ปัญหาด้านกฎหมาย	ไม่มีกฎหมายควบคุม	ต้องขออนุญาตจากทางการเพื่อขอใช้คลื่นวิทยุ
ราคา	ราคาถูก-ปานกลาง	ราคาปานกลาง-แพง
ขนาดรูปร่าง	สามารถปรับปรุงให้มีขนาดเล็กลงได้	ไม่สามารถลดขนาดให้เล็กได้มากเท่าที่ควรเนื่องจากเหตุผลด้านกำลังส่งและอุปกรณ์ประกอบ

ตารางที่ 2.4 ตารางเปรียบเทียบระบบควบคุมแบบไร้สายโดยการควบคุมด้วยแสง กับใช้คลื่นวิทยุ

ในการควบคุมด้วยสัญญาณแสง สัญญาณควบคุมที่เป็นสัญญาณไฟฟ้าจะถูกแปลงเป็นสัญญาณแสงก่อน แล้วจึงถูกส่งออกไปยังตัวรับ ซึ่งก็จะต้องมีอุปกรณ์พิเศษทำหน้าที่รับสัญญาณแสงแล้วแปลงกลับให้เป็นสัญญาณไฟฟ้า จากนั้นจึงทำการแยกชนิดของสัญญาณว่าเป็นสัญญาณควบคุมที่สอดคล้องกับคำสั่งของตัวส่งอย่างไร ดังบล็อกไดอะแกรมที่แสดงดังรูปที่ 2.2 ลักษณะของแสงที่ใช้ส่งสัญญาณแบ่งเป็น 2 ประเภทตามค่าความถี่ของแสง คือ ประเภทแสงที่มองเห็นได้ และ ประเภทแสงที่มองไม่เห็น ซึ่งมักได้แก่แสงในย่ายของความถี่อินฟราเรดหรือใต้แสงสีแดง



รูปที่ 2.2 แสดงบล็อคไดอะแกรมโครงสร้างของระบบควบคุมด้วยแสงแบบไร้สาย

2.2.2 คุณสมบัติของ LED อินฟราเรด

แรงดันตกคร่อมที่รอยต่อ PN ของไดโอด ต้องมีค่ามากกว่าแรงดัน Threshold จึงจะสามารถทำให้ไดโอดนำกระแสได้ สำหรับซิลิกอนไดโอดแรงดันทำงานมีค่าประมาณ 0.6V ส่วน LED ที่ให้แสงในย่านมองเห็นได้ ถ้าทำจากสาร GaP ซึ่งให้แสงสีเขียวจะมีค่าแรงดันทำงานประมาณ 2.1-2.8V ถ้าเป็น LED ที่ทำจาก AlGaAs ให้แสงสีแดงมีแรงดันทำงาน 1.75-2.5V ส่วน LED ที่ให้แสงใกล้ย่านอินฟราเรดทำจากสาร GaAs มีแรงดันทำงาน 1.5V โดยให้แสงที่มีความยาวคลื่น 940 นาโนเมตร และถ้าทำจาก AlGaAs จะได้แสงความยาวคลื่น 880 นาโนเมตร ที่แรงดัน 1.75 โวลต์

พลังงานที่ได้จากการเปล่งแสงของ LED หาได้จากกระแสไบอัสตรงของไดโอดและต้องระมัดระวังไม่ให้กระแสส่วนนี้มี ค่าสูงจนเกิดความร้อนอันจะทำอันตรายต่อชิ้นไดโอด

สิ่งสำคัญที่สุดของ LED อินฟราเรดกำลังสูงคือ ชิ้นสาร AlGaAs ที่ให้ความยาวคลื่น 880 นาโนเมตร และสาร GaAs ซิลิกอนไดโอดที่ให้แสงความยาวคลื่น 940 นาโนเมตร ดังที่แสดงไว้ในรูปที่ 2 คือกราฟสเปกตรัมที่เปรียบเทียบความยาวคลื่นกับการเปล่งแสงของ LED ทั้ง 2 ชนิด

ซิลิกอน LED ที่ทำจาก GaAs ให้กำลังงานประมาณ 5 มิลลิวัตต์ ที่กระแสไบอัสตรง 100 มิลลิแอมป์ LED ที่ทำจาก AlGaAs จะให้กำลังงานเป็น 2 เท่าเมื่อให้กระแสตรงค่าเดียวกันข้อที่ดีกว่าอีกประการหนึ่งของ LED ชนิด AlGaAs คือมี rise time และ fall time ที่เร็วกว่าคือประมาณ 0.5 ไมโครวินาที ในขณะที่ GaAs ซิลิกอนไดโอดมีค่า 1.5 ไมโครต่อวินาที ข้อดีอีกประการหนึ่งคือ

การเปล่งแสงของ LED ที่ความยาวคลื่น 880 นาโนเมตร (AlGaAs) จะใกล้เคียงกับความยาวคลื่นที่ซิลิกอนโฟโตทรานซิสเตอร์มีความไวสูงสุดจึงเป็นการเหมาะที่จะใช้ LED ที่ความยาวคลื่น 940 นาโนเมตร

นอกจากนั้น LED ที่ให้ความยาวคลื่นแสง 880 นาโนเมตร ยังไม่ถูกดูดกลืนโดยละอองน้ำเหมือน LED ที่ให้ความคลื่น 940 นาโนเมตร จึงสามารถนำไปใช้ในการตรวจจับไอน้ำในอากาศ LED ชนิด 940 นาโนเมตร ไม่เหมาะกับการสื่อสารด้วยแสงภายนอกเพราะจุดอ่อนเรื่องการถูกดูดกลืนด้วยไอน้ำในอากาศนั่นเอง ส่วน LED ชนิดซิลิกอนที่ทำจาก GaAs มักจะใช้เป็นแหล่งกำเนิดแสงย่านอินฟราเรด

การแผ่รังสีจาก LED อินฟราเรด

การใช้งาน LED อินฟราเรดกำลังสูง อาจพบความยุ่งยากที่ไม่อาจเห็นการแผ่รังสีย่านที่ตาสามารถมองเห็นได้ของอุปกรณ์ชนิดนี้ จึงเป็นการยากที่จะจัดเลนส์โฟกัส และยากต่อการทำรูปแบบการกระจายของรังสี

ด้วยเหตุนี้จึงได้มีการนำการ์ดเคลือบฟอสเฟอร์ชนิดพิเศษมาใช้ในการเปลี่ยนรังสีย่านใกล้อินฟราเรดเป็นรูปแบบของแสงที่มองเห็นได้

ตอนแรกจะนำการ์ดนี้ไปถูกแสงสว่างก่อน เช่น แสงในห้อง ก็จะทำให้พื้นผิวของการ์ดกลายเป็นสีเข้ม เมื่อการ์ดรับแสงจาก LED อินฟราเรด ก็จะเปลี่ยนเป็นสีส้มหรือสีเขียว

การเปล่งแสงที่มีความยาวคลื่น 880 นาโนเมตร ของ LED อินฟราเรด กำลังสูงมองเห็นเป็นแสงสีแดงมัว ๆ เพราะการรับรู้ด้วยตาของมนุษย์จำกัดไว้ที่ 880 นาโนเมตรพอดี แต่ดังที่แสดงไว้ในรูปที่ 2 LED อินฟราเรด AlGaAs มีการเปล่งแสงที่ความยาวคลื่นต่ำลง เช่น 750 นาโนเมตร ไดโอดที่ให้แสงสีแดงส่วนใหญ่อาจมาจากไดโอดที่ให้แสงความยาวคลื่นต่ำนี้ได้

LED อินฟราเรดสามารถนำไปประยุกต์งานอย่างกว้างขวาง ทั้งในโรงงานอุตสาหกรรม และระบบตรวจจับสมัยใหม่นับเป็นเซนเซอร์ที่เก่งอีกตัวหนึ่ง

2.2.3 LED กำลังสูง ที่ให้แสงย่านใกล้อินฟราเรด

รอยต่อ PN ของ LED เป็นแหล่งกำเนิดโฟตอนยอดเยี่ยมมาก สมัยแรก ๆ ได้มีการค้นพบว่ารอยต่อของแกลเลียมอาร์เซไนด์ ซึ่งให้แสงย่านใกล้อินฟราเรด ทำให้เกิดโฟตอน 88 ตัว ต่ออิเล็กตรอน 100 ตัว เป็นประสิทธิภาพควอนตัมที่น่าทึ่งไม่น้อยทีเดียว

แต่เนื่องจากโฟตอนที่ถูกปล่อยมาจากรอยต่อ ส่วนหนึ่งถูกสกัดกั้นโดยกรอบของชิ้นสารเอาท์พุทจริง ๆ จึงน้อยกว่า 88 เปอร์เซ็นต์

อีกส่วนหนึ่งของการแผ่รังสีจากรอยต่อ ก็ถูกดูดกลืนไปโดยตัวไดโอดเองและยังมีการแผ่รังสีบางส่วนที่กระทบผิวไดโอดด้วยมุมที่มากกว่ามุมวิกฤติ (critical angle) จึงถูกสะท้อนกลับไปยังไดโอดทำให้เกิดการสูญเสียเช่นกัน

ปัญหาเรื่องมุมวิกฤต เป็นปัญหาสำคัญประการหนึ่ง ซึ่งอธิบายได้ดังนี้ อากาศมีดัชนีของการหักเหเท่ากับ 1 (หรือ 1.0003เมื่อเทียบกับสูญญากาศ) ดัชนีการหักเหของเพชรคือ 2.42 ในขณะที่ GaAs มีดัชนีหักเหของเพชรคือ 2.42 ในขณะที่ GaAs มีดัชนีหักเหของแสงเท่ากับ 3.5 ซึ่งเป็นหนึ่งในสสารเพียงไม่กี่ชนิดที่มีดัชนีของการหักเหมากกว่าเพชร ความแตกต่างของดัชนีการหักเหนี้ ทำให้การแผ่รังสีของแสงที่เกิดขึ้นภายในชิ้นสาร GaAs เมื่อกระทบผิวที่ติดต่อกับอากาศภายนอกมีมุมวิกฤต 16 องศา หากมุมตกกระทบที่รอยต่อของผิวมากกว่า 16 องศา แสงนั้นจะถูกสะท้อนกลับมายังชิ้นสาร ปรากฏการณ์เช่นนี้เรียกว่า ปรากฏการณ์สะท้อนกลับภายใน (total internal reflection)

วิธีลดความไม่สมดุลระหว่างดัชนีการหักเหของแสงมีอยู่หลายวิธี วิธีหนึ่งเป็นของเท็กซัสอินสตรูเมนต์คือ ทำผิวหน้าของชิ้นไดโอดเป็นรูปโดม ทำให้แสงจากภายในมาถึงผิวต่อระหว่างอากาศและ GaAs ด้วยมุมที่ไม่เกิน 16 องศา วิธีนี้สามารถเพิ่มประสิทธิภาพให้สูงกว่าชิ้นไดโอดแบบแบนราบประมาณ 10เท่า

การผลิตไดโอดแบบนี้มีราคาค่อนข้างแพง แล้วยังต้องการสาร GaAs มากกว่าแบบแบนราบด้วย จึงมีการใช้อีพ็อกซี่เคลือบไดโอด เพื่อทำให้ดัชนีการหักเหของแสงสมดุลดัชนีการหักเหของสารเคลือบนี้มีค่าตั้งแต่ 1.4 ถึง 1.8 แม้ว่าจะไม่เท่ากับดัชนีการหักเหของ GaAs ก็ตาม แต่ก็มีค่าใกล้เคียงมากกว่าอากาศ

ส่วน LED ที่ทำจากแกลเลียมฟอสไฟด์ ที่ให้แสงสีเขียวและมีมุมวิกฤต7.7 องศา การเคลือบไดโอดด้วยอีพ็อกซี่ที่มีดัชนีหักเห 1.66 ทำให้เพิ่มมุมวิกฤติเป็น 30.3 องศา และทำให้การเปล่งแสงจากไดโอดเพิ่มขึ้นอีก 2.5 เท่า

พิจารณถึงแสงที่ปล่อยออกมาจากขอบของสารที่ใช้ในการสร้าง LEDในสมัยแรกๆ นั้น แสงส่วนนี้จะสูญเสียไปถ้าชิ้นสารถูกวางไว้ในตัวสะท้อน (reflector) ตัวเล็ก ๆ ดังรูปที่ 1 ตัวสะท้อนนี้จะสะท้อนแสงที่เปล่งจากขอบของชิ้นสารออกมาสู่ภายนอก

เทคโนโลยี เฮตเทอโรจังก์ชัน

ในระยะต่อการของการพัฒนา LED ได้มีการค้นพบว่าสามารถสร้าง LED ที่ให้แสงย่านใกล้อินฟราเรดที่มีประสิทธิภาพสูงได้โดยใช้เทคโนโลยีที่เรียกว่า เฮตเทอโรจังก์ชัน (heterojunction)

เทคโนโลยีแบบนี้ไดโอดจะมีสภาพเหมือนแซนวิชของสารกึ่งตัวนำที่มีคุณสมบัติทางแสงและทางไฟฟ้าต่างกันอยู่เล็กน้อย การทำชั้นสารหลาย ๆ ชั้นในรูปแบบของเฮตเทอโรจังก์ชันเป็นการเพิ่มประสิทธิภาพการเปล่งแสงของไดโอด เนื่องจากการใช้เทคโนโลยีแบบนี้ทำให้ลดพื้นที่บริเวณรอยต่อ PN ลง ชั้นนอกสุดของเฮตเทอโรจังก์ชันมีความโปร่งใสที่ให้แสงผ่านได้มากกว่ารอยต่อ PN ของไดโอดที่ใช้เทคโนโลยีแบบโฮโมจังก์ชัน (homojunction) ทำให้พลังงานแสงที่ได้จึงมากกว่า 2 เท่า

ด้วยเทคโนโลยีแบบเฮตเทอโรจังก์ชันทำให้สามารถผลิตเลเซอร์ไดโอดที่ทำงานได้อย่างต่อเนื่อง ณ อุณหภูมิ ห้องและผลิตเลเซอร์ไดโอดที่ให้แสงในย่านที่ตามนุษย์สามารถมองเห็นได้ ต่อมาเทคโนโลยีแบบเฮตเทอโรจังก์ชั่นก็ได้นำมาใช้ในการผลิต LED ที่ให้แสงในย่านที่มองเห็นได้ และนำอะลูมิเนียมแกลเลียมอาร์เซไนด์ มาใช้ผลิต LED แสงสีแดงที่มีความสว่างสูง

แบบของวัสดุ

1. แก้ว

แก้วสามารถที่จะตัดความยาวคลื่นที่ต้องการ 2.7 μm เพราะมีการดูดกลืนที่แรงโดย OH^- สำหรับแก้วควอทซ์หรือแก้วซิลิกาจะผ่านได้ถึง 5 μm ซึ่งเป็นที่น่าสนใจ การดูดกลืนที่เกินขีดจำกัดนี้เนื่องจากการสั่นสเปกตรัมของพันธะ Si-O เราสามารถที่จะจำกัดช่องว่างของน้ำระหว่างการสร้างเนื่องมาจากขอบการดูดซับใกล้ 2.8 μm แก้วนี้สามารถสร้างผลิตภายใต้สูญญากาศ โดยเฉพาะพิเศษที่ความยาวคลื่นที่ยาวกว่า เรารวมถึง

- Calcium aluminate เช่น Bousch - Lomb IR 10-11-12
- แก้ว Kodak Irtron

- Irtron 1 : เกิดจากความร้อนและการอัด แมกนีเซียม ฟลูออไรด์ (MgF_2)
- Irtron 2 : เกิดจากความร้อนและการจัด ซิงค์ ซัลไฟด์ (ZnS)
- Irtron 3 : แก้ว แคลเซียม ฟลูออไรด์ (CuF_2)
- Irtron 4 : แก้ว ซิงค์ ซัลไฟด์ (ZnS)
- Irtron 5 : เกิดจากความร้อนและการวัดแมกนีเซียม ออกไซด์ (MgO)
- Irtron 6 : แก้ว แคดเมียม เทลลูไรด์ (Cd Te)

- แก้ว Chalcoyenides จะมีธาตุหนัก เช่น arsenic , ตะกั่ว
- แก้วที่มี arsenic trisulphide (As_2S_3)
- แก้วที่มี germanuim oxide

2. ผลึก

อาจเป็น Monocrystalline หรือ Polycrystalline โดยทั่วไปจะเป็นสารไดอิเล็กตริกหรือสารกึ่งตัวนำ ความโปร่งแสงของมันขึ้นกับอุณหภูมิ เมื่ออุณหภูมิจะไปลดการผ่านได้ของแสงในตัวกลางสารกึ่งตัวนำอย่างแรง เพราะการเกิดของระดับพลังงานอะตอมใหม่ที่ยอมให้มีการดูดโฟตอน โดยทั่วไปใช้คริสตอล

3. พลาสติก

วัสดุชนิดนี้ไม่ค่อยจะเหมาะสมเพราะไม่มีคุณสมบัติทางกลและยังมีสภาพที่ไม่คงที่ทางอุณหภูมิอย่างไรก็ตาม ก็ยังสามารถนำมาประดิษฐ์ได้โดยการขด ซึ่งจะได้ผลิตภัณฑ์เป็นแบบ aspherical opticalสำหรับอินฟราเรดยิ่งกว่านั้นฟิล์มpdythene และ perpex (polymethylmethacrytate) สามารถนำไปใช้เป็นหน้าต่างของอุปกรณ์ และยังมีการส่งผ่านอินฟราเรดดีมากแม้จะมีการดูดกลืนบางแถบ

4. โลหะ

โลหะบางอย่างมีคุณสมบัติการสะท้อนที่ดีและด้วยเหตุนี้จึงนำไปใช้ทำกระจก

ฟิล์มสะท้อนที่ทำจากโลหะจะสร้างโดยพัดพาตะกอนในสูญญากาศลงบนฐานรองซึ่งทำให้เกิดรูปแบบของแก้ว , ควอทซ์ , pyrex หรือ ceramic ที่มีสัมประสิทธิ์การขยายตัวมาก

โดยทั่วไปจะใช้โลหะเช่น อลูมิเนียม เงิน ทอง และ ทองแดง ผิวของกระจกแบบนี้มักจะป้องกันโดยชั้นบางของซิลิกอนออกไซด์

2.3 การทำงานแบบ Real-time

ทฤษฎีและหลักการทำงานของ 89C2051

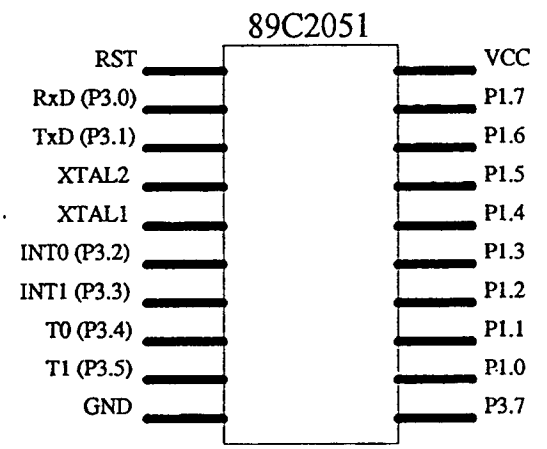
AT 89C2051 เป็นชิพไมโครคอมพิวเตอร์ขนาดจิ๋ว ผลิตโดย ATMEL ชุดคำสั่งและสถาปัตยกรรมภายในจะเหมือนกับไมโครคอมพิวเตอร์ตระกูล MCS - 51 ซึ่งผลิตโดยอินเทลมีหน่วยความจำภายในเป็น PROM หรือเรียกว่า Flash memory ขนาด 2 KB สามารถเขียนและลบใหม่ได้ไม่น้อยกว่า 1000 ครั้ง โปรแกรมที่บันทึกไว้ภายในชิพสามารถเก็บไว้ได้นานถึง 10 ปี ซึ่งเป็นระยะเวลาที่ใกล้เคียงกับชิพ 8751 ของตระกูล MCS - 51 ซึ่งหน่วยความจำโปรแกรมเป็นแบบ EPROM จะใช้แสง UV ในการล้างข้อมูล จุดเด่นของชิพ 89C2051 มีดังต่อไปนี้

1. สามารถใช้แทนไมโครคอมพิวเตอร์ ตระกูล MCS-51
2. หน่วยความจำโปรแกรมเป็น Flash memory ขนาด 2 KB
3. ใช้ไฟเลี้ยงได้ตั้งแต่ 2.7 - 6 V
4. ออสซิลเลเตอร์สามารถป้อนได้ตั้งแต่ 0-24 MHz
5. สามารถล็อคโปรแกรมได้สองระบบ
6. มีอินพุทและเอาท์พุทพอท 15 บิต
7. หน่วยความจำข้อมูล128 บิต
8. มีตัวนับและตั้งเวลาขนาด 16 บิต 2 ตัว
9. แหล่งอินเทอรัพมี 5 แหล่ง
10. พอทอนุกรมแบบ UARTโปรแกรมความเร็วในการส่งข้อมูลได้
11. เอาท์พุทพอทสามารถขับ LED ได้โดยตรง ด้วยกระแส sink 20 mA
12. มีอนาลอกคอมพาราเตอร์
13. มีโหมด Idea และ Power

ชิพ 89c2051 สร้างด้วยเทคโนโลยี Atmel's High Density Nonvolalatile Memory Technology หน่วยความจำโปรแกรมภายใน ขนาด 2KB ดังกล่าวเป็นแบบ Elash Progrsmmable And Erasable Read Only Memory (PEROM) จากความสามารถและจุดเด่น กล่าวคือ เมื่อเปรียบเทียบกับชิพไมโครคอมพิวเตอร์จิ๋วของ Philips เช่นเบอร์87c750 มีโหรแกรมภายในเป็นแบบEPROM ในแง่การออกแบบและพัฒนาโปรแกรมขนาดสำหรับงานควบคุมขนาดเล็กเราอสจใช้ชีป 89c2051 -ได้สะดวกกว่าเพราะขณะพัฒนาโหรแกรมเราสามารถล้างโปรแกรมด้วยสัญญาณไฟฟ้าจะกินเวลาไม่เกิน 5 วินาที ในขณะที่ ชิป 87c750 เป็น EPROM เวลาล้างโปรแกรมจะใช้เวลามากกว่า 5 วินาที และที่สำคัญการพัฒนากระทำได้ลำบาก

ตัวถังและขาสัญญาณของชิป AT89c2051

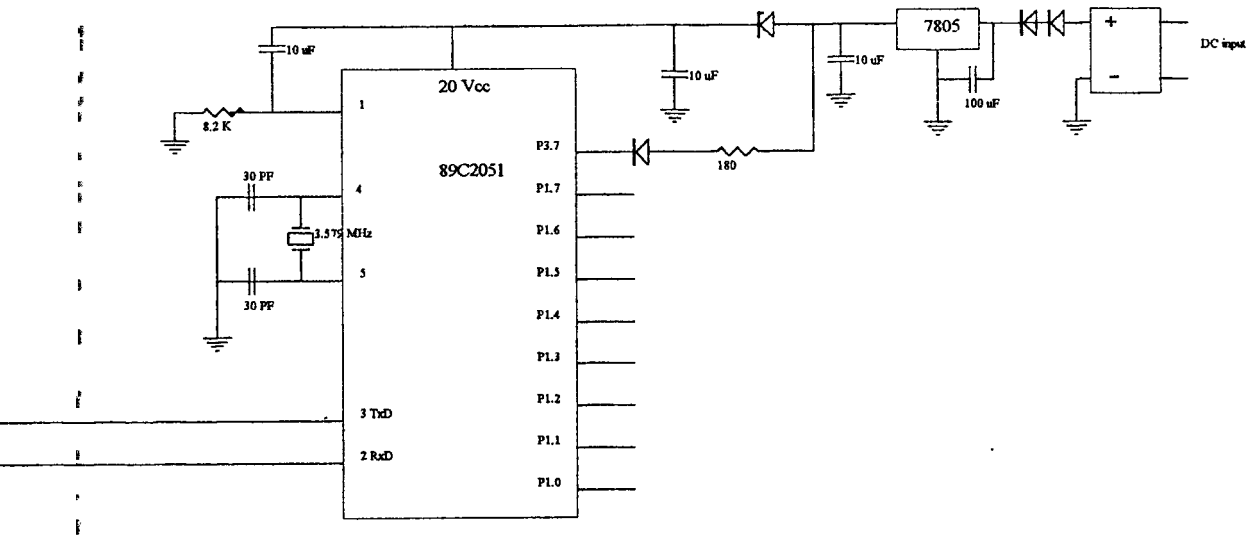
ลักษณะตัวถังเป็นแบบPDIP (Plastic Dual-Inline-Package) ขนาด 20 ขา เปรียบเทียบตัวถังของ 87c51 แบบเซรามิก 40 ขา กับชิป 89c2051 หากเราจำได้กับระบบ 8051จะทำความเข้าใจได้ไม่ยาก ขาที่แตกต่างสำหคัญที่เราสังเกตได้ชัดเจนคือ I/O พอท มีเพียงพอท 1 จำนวน 8 บิต ระบุด้วย p1.0-p1.7 ข้อสังเกตจะพบว่า p1.0และ p1.1 มีฟังก์ชันเสริมคือ AINO และ AINI เป็นอนาลอกอินพุทของวงจรคอมพาราเตแร์ แสดงผังวงจรภายในดังรูป



รูปที่ 2.3แสดงตัวถังและขาสัญญาณของชิป AT89c2051

จะเห็นว่าเราสามารถใช้วงจรเปรียบเทียบศักดาป้อนเข้ามาที่ขา p1.0 และ p1.1ได้โดยเอาท์พุทจะต่อกับ p3.6 ซึ่งสามารถอ่านค่าด้วยโปรแกรม ดังนั้นพอท p3.6 จึงมิได้ต่อกับขาภายนอก พอท 3 จึงมีเพียง p3.1-p3.5 สำหรับฟังก์ชันเผื่อเลือก ยังคงใช้งานได้เหมือนกับชิปตระกูล 8051 ของอินเทล

วงจรเบื้องต้นไมโครคอมพิวเตอร์ AT89c2051



รูปที่ 2.4แสดงวงจรเบื้องต้นไมโครคอมพิวเตอร์ AT89c2051

จะเห็นว่าเดี๋ยวนี้เราสามารถไมโครคอมพิวเตอร์ได้ง่ายกว่าต่อวงจร555 เสียอีก ในขณะที่ในอดีตเราใช้วงจรแยกส่วน (discrete logic) จำนวนมากต่อกันเป็นวงจรต่าง ๆ ขณะนี้เราสามารถใช้วิธีการเขียนโปรแกรมแทน เราสามารถสร้างสัญญาณแทนการต่อวงจรในอดีตเกือบทั้งหมดของวงจรโดยเฉพาะวงจรดิจิตอล ไม่ว่าจะเป็นวงจรลอจิคหรือวงจรฟลิปฟลอป วงจรนับและตั้งเวลา อย่างไรก็ตามวงจรแอพพลิเคชัน ที่จำเป็นต้องเชื่อมต่อกับสัญญาณอนาลอก อาจจะต้องใช้ชิพเพิ่มขึ้น

2.4 ขั้นตอนการพัฒนาโปรแกรม

ปกติแล้วชิปที่เราซื้อมาจะได้รับการล้างโปรแกรมภายในเรียบร้อยแล้ว กล่าวคือภายในหน่วยความจำโปรแกรมเป็นFFH ทั้งหมดการเขียนโปรแกรมคือ การทำให้บิตที่ต้องการโปรแกรมมีค่าเป็นลอจิค 0 อัลกอลิทึม ที่ใช้สำหรับการโปรแกรมศึกษาเพิ่มเติมรายละเอียดได้ในภาคผนวก เนื่องจากกระบวนการเขียนโปรแกรมลงบนชิป อาจทำได้ลำบาก สำหรับการทดลองนี้ ดังนั้นเราจะใช้เครื่องโปรแกรมสำเร็จรูปของศิลา PRO-100 V1.0 สามารถโปรแกรมชิปในตระกูลของ ATMEL ได้หลายเบอร์เช่น 89c2051

89c2051 89c51 89c52 ขั้นตอนการพัฒนาโปรแกรมด้วยชิป 89c2051 มีดังต่อไปนี้

1. เขียนโปรแกรมภาษาแอสเซมบลี้ text file ด้วยeditor มีสกุลเป็น .ASM เช่น mono.asm
2. ใช้แอสเซมเบลอร์แปลง source file ให้เป็นมาตราฐาน intel hex fileสกุล.HEX เช่นmono.hex
3. Power on PRO-100 เลือกชิป 89c2051 ด้วยคำสั่ง TYPE เสียบชิป 89c2051 ลงในZIFชิดด้านบน
4. Download Hex file ลงไปบนหน่วยความจำบัฟเฟอร์ของPRO-100 ด้วยคำสั่งLOAD ขนาดความเร็ว 9600bps 8N1
5. เขียนโปรแกรมลงบนชิป89c2051 ด้วยคำสั่งwrite
6. นำชิปออกจาก ZIF (Zero Insertion Force)ไปเสียบบนวงจรแอพพลิเคชัน

หากโปรแกรมไม่ถูกต้อง ต้องการเขียนโปรแกรมเขียนโปรแกรมใหม่ให้ทำการล้างโปรแกรมด้วยคำสั่ง Erase ก่อนจึงทำการโปรแกรมใหม่ได้

ข้อควรระวัง

1. ZIFของPRO-100 เป็นแบบ 40 ขาชิป 89c2051 มีเพียง 20 ขา เวลาเสียบลงบนPRO-100 ต้องเสียบชิปด้านบน

2.ก่อนเสียบชิปลงบนZIFต้องแน่ใจว่า กระเดื่องล็อกตั้งขึ้น

3.ตำแหน่งชิปขาหนึ่งต้องอยู่ชิดด้านบนโดยสังเกตรอยปุ่มกลม

4.เมื่อแน่ใจว่าตำแหน่งก่อนเสียบถูกต้องจึงค่อยใส่ลงในZIF socket และกดกระเดื่องลงเพื่อล็อกชิปที่จะโปรแกรมให้แน่นหนา

บทที่ 3

การติดต่อกับ USER โดยใช้ GUI

3.1 การเขียนโปรแกรมติดต่อกับ USER โดยใช้ GUI แทนการใช้ DOS MODE

โดยปกติแล้วคอมพิวเตอร์สามารถเข้าใจได้แต่รหัสไบนารี (binary code) ซึ่งมีแค่เลขศูนย์และเลขหนึ่งเท่านั้น รหัสไบนารีรู้จักกันในอีกชื่อหนึ่งว่า รหัสเครื่อง (machine code) ซึ่งจัดอยู่ในภาษาสำหรับเขียนโปรแกรมระดับต่ำ (low - level programming language) แต่การเขียนโปรแกรมที่ประกอบด้วยเลขศูนย์และเลขหนึ่งนั้นเป็นเรื่องยากและเสียเวลา เพื่อให้เขียนโปรแกรมได้ง่ายขึ้น นักวิทยาศาสตร์จึงประดิษฐ์ภาษาขึ้นเป็นภาษาแอสเซมบลี (Assembly language) และมีข้อเสียคือคอมพิวเตอร์ไม่สามารถเข้าใจภาษาแอสเซมบลีได้มีแต่ user เท่านั้นที่รู้ ดังนั้นก่อนที่โปรแกรมที่เขียนขึ้นด้วยภาษาแอสเซมบลีจะสามารถรันได้ ก็จำเป็นที่จะต้องแปลโปรแกรมนั้นเป็นโปรแกรมรหัสเครื่องเสียก่อน เพราะการแปลในขั้นตอนนี้ไม่ได้ให้ประสิทธิภาพได้ครบทั้งร้อยเปอร์เซ็นต์ โปรแกรมเดียวกันที่เขียนด้วยภาษาแอสเซมบลีจะช้ากว่า และใหญ่กว่าโปรแกรมแบบเดียวกันที่เขียนโดยใช้รหัสเครื่องด้วยความพยายามที่จะทำให้การเขียนโปรแกรมได้ง่ายขึ้น นักวิทยาศาสตร์จึงได้ประดิษฐ์ภาษาใหม่ ๆ ขึ้นมา เริ่มจาก A , B, C และท้ายสุดก็คือ C++ ซึ่งทั้งหมดนี้ถูกจัดอยู่ในภาษาระดับกลาง (mid - level language) ภาษาระดับกลางนี้เริ่มรวมเอาคำในภาษาอังกฤษกับสัญลักษณ์เข้าด้วยกันเช่น a++ เป็นคำสั่งที่เข้าใจได้ในภาษา C++ การเขียนโปรแกรมด้วยภาษา C++ นั้นง่ายกว่ากว่าการเขียนด้วยภาษาแอสเซมบลีมาก อย่างไรก็ตามโปรแกรมที่เขียนด้วยภาษา C++ ก็ต้องแปลให้เป็นรหัสเครื่องก่อน เพราะว่าภาษา C++ แตกต่างจากรหัสเครื่อง ดังนั้นการแปลให้เป็นรหัสเครื่องจึงทำให้ประสิทธิภาพลดลงโปรแกรมที่เขียนด้วย C++ จะรันได้ช้ากว่าโปรแกรมเดียวกันที่เขียนโดยใช้รหัสเครื่องอยู่เล็กน้อย

จนกระทั่งมีการใช้ภาษาระดับสูง (high - level language) อย่างเช่น COBAL , PASCAL และ BASIC ใช้คำในภาษาอังกฤษ และตัวย่อ ทำให้เราสามารถอ่านและเขียนได้ง่ายขึ้น ซึ่งในโครงงานพิเศษนี้ผู้จัดทำได้เลือกใช้ โปรแกรม Visual Basic 4.0

3.2 ความเป็นมาของ GUI (Graphical User Interface)

ในยุคแรกคอมพิวเตอร์มีระบบปฏิบัติการเป็น MS DOS ที่สามารถใช้ทรัพยากรได้อย่างจำกัด เนื่องจากข้อจำกัดหลาย ๆ ด้าน เช่น หน่วยความจำหลักในการประมวลผล ความเร็วในการประมวลผลของ CPU 8088 , 2 MHz และ หน้าจอเป็นสีเขียวเพียงสีเดียว ทำให้การเขียนโปรแกรมในสมัยก่อนจำเป็นต้องคำนึงถึงทรัพยากรที่มีอยู่อย่างจำกัด ในไม่กี่ปีมานี้เกิดการพัฒนาทางด้านอิเลคทรอนิคส์ ทำให้ประสิทธิภาพของคอมพิวเตอร์ในปัจจุบันมีการพัฒนามากกว่าในสมัยก่อนทั้งด้านความเร็ว และการแสดงผลในรูปของสีสันที่สวยงาม ทำให้มี operating ตัวใหม่ๆ ออกมาสู่ท้องตลาดมากขึ้น เช่น Windows 3.11 , Window 95 ซึ่งเป็นโปรแกรมที่ออกแบบมาให้ผู้ใช้เข้าใจได้ง่าย และติดต่อกับผู้ใช้ในรูปแบบของรูปภาพ (Graphic) ที่สวยงาม ซึ่งในปัจจุบันเป็นที่นิยมอย่างยิ่ง เรียกว่า GUI (Graphical User Interface)

3.2.1 ส่วนประกอบพื้นฐานของ GUI

แม้ว่าโปรแกรมที่ใช้จะแตกต่างกัน แต่ลักษณะของ GUI ก็ยังดูเหมือน ๆ กัน เพราะใช้ส่วนประกอบต่าง ๆ ที่คล้ายคลึงกัน

GUI จะแสดงข้อมูลหรือข่าวสารในกรอบที่เรียกว่า หน้าต่าง (window) โดยหน้าต่างที่ว่านี้อาจปรากฏเป็นบางส่วนก็ได้ ในขณะเดียวกัน หน้าต่างสามารถแสดงบนจอภาพได้หลาย ๆ บานพร้อมกัน และอาจมีการเหลื่อมซ้อนกัน หรือเรียงต่อกันก็ได้ ใน Visual Basic จะเรียกหน้าต่างนี้ว่า ฟอร์ม (form)

Visual Basic ที่ถูกเรียกขึ้นมาครั้งแรก จะพบฟอร์มว่าง ๆ อันหนึ่งซึ่งผู้ใช้ต้องจัดวางออบเจ็กต์ (object) ลงบนฟอร์มเพื่อสร้างสรรค์สิ่งที่มีประโยชน์ ซึ่งออบเจ็กต์อาจเป็นได้ทั้งรูปภาพ กรอบ หรือปุ่มต่าง ๆ Visual Basic เรียกออบเจ็กต์เหล่านี้ว่า คอนโทรล (control)

ออบเจ็กต์ช่วยให้ฟอร์มสามารถรับการกระทำอย่างใดอย่างหนึ่งจากผู้ใช้ เพื่อนำมาทำให้เกิดประโยชน์อย่างหนึ่งอย่างใด

ปุ่มสี่เหลี่ยมสีเทา มีชื่อเรียกว่า คอมมานด์บัตทอน (command button) ซึ่งเป็นออบเจ็กต์ธรรมดา ๆ แต่สามารถสร้างเป็นสิ่งต่างๆ มากมาย ตัวของออบเจ็กต์นั้นไม่สามารถทำอะไรได้ ผู้ใช้จะต้องเขียนโค้ด BASIC เพื่อบอกให้รู้ว่าจะต้องทำอะไรบ้าง

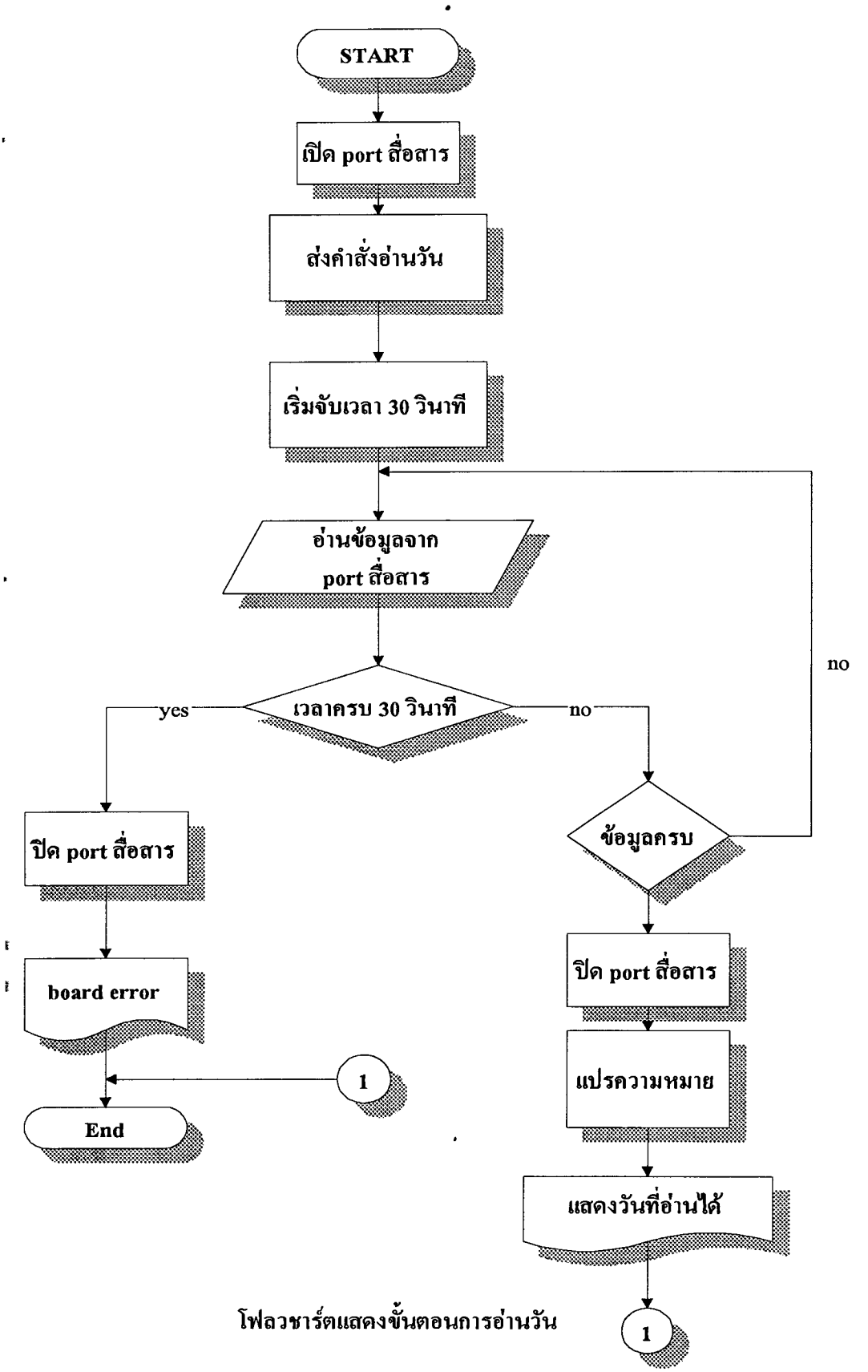
Visual Basic มีส่วนประกอบของ GUI ที่เป็นรูปแบบพื้นฐานของวินโดว์ทั่ว ๆไปไว้ให้พร้อม programmer จะต้องสร้าง GUI ที่สามารถทำให้บุคคลทั่วไปสามารถเข้าใจได้ง่ายที่สุด

3.3 การพัฒนาโปรแกรมในการสั่งงานบอร์ดแบบ Real-time ที่สร้างขึ้นใน โครงงานพิเศษ

ทางผู้จัดทำเลือกใช้อุปกรณ์พัฒนาโปรแกรม Visual Basic 4.0 Enterprise Edition (32bits) เพื่อความสวยงามและความสะดวกในการใช้งาน ทำให้การสื่อสารระหว่างผู้ใช้กับบอร์ดมีประสิทธิภาพมากยิ่งขึ้น

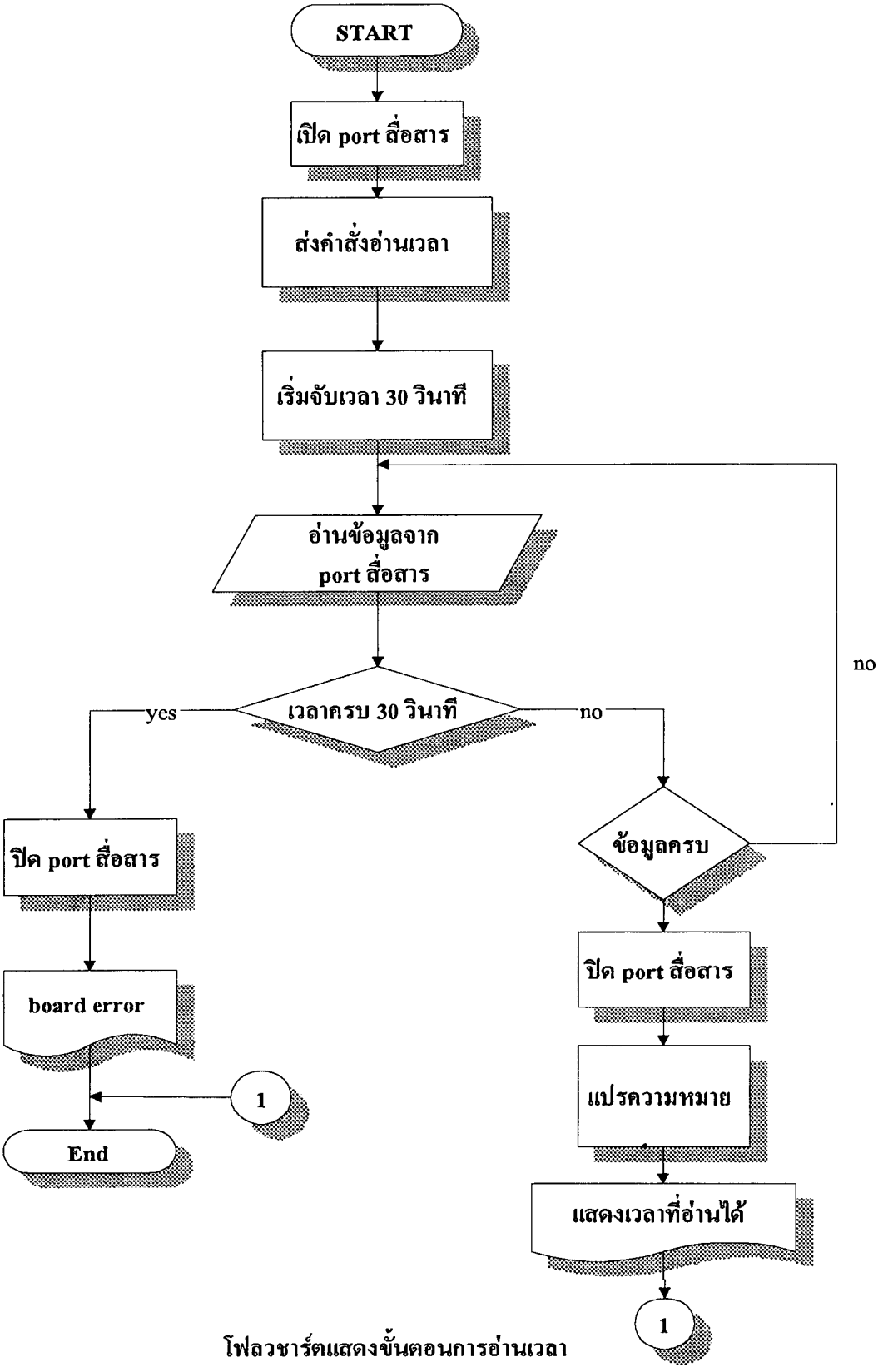
3.4 ขั้นตอนการทำงานของโปรแกรม

แนวความคิด(โฟลว์ชาร์ต)ต่าง ๆในการเขียนโปรแกรม



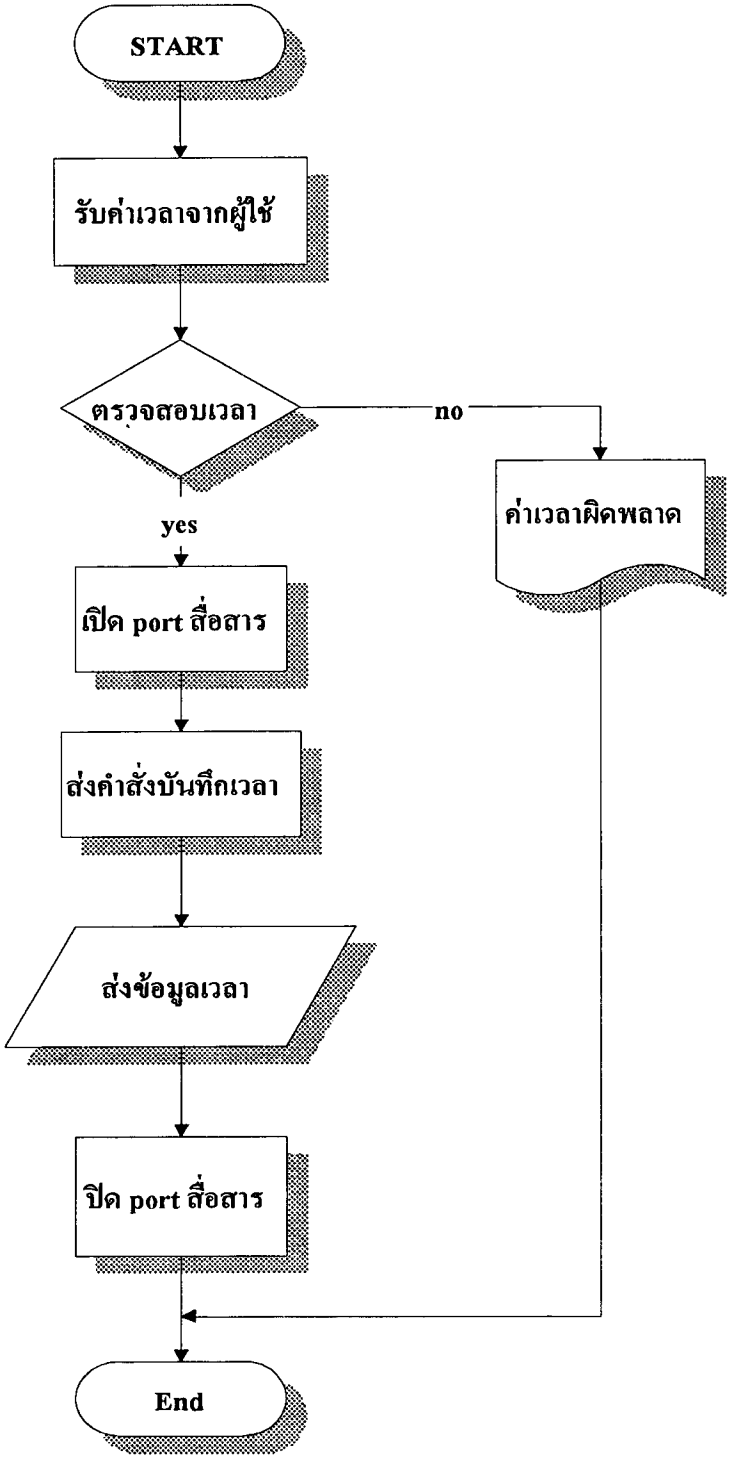
โฟลวชาร์ตแสดงขั้นตอนการอ่านวัน

รูปที่ 3.1



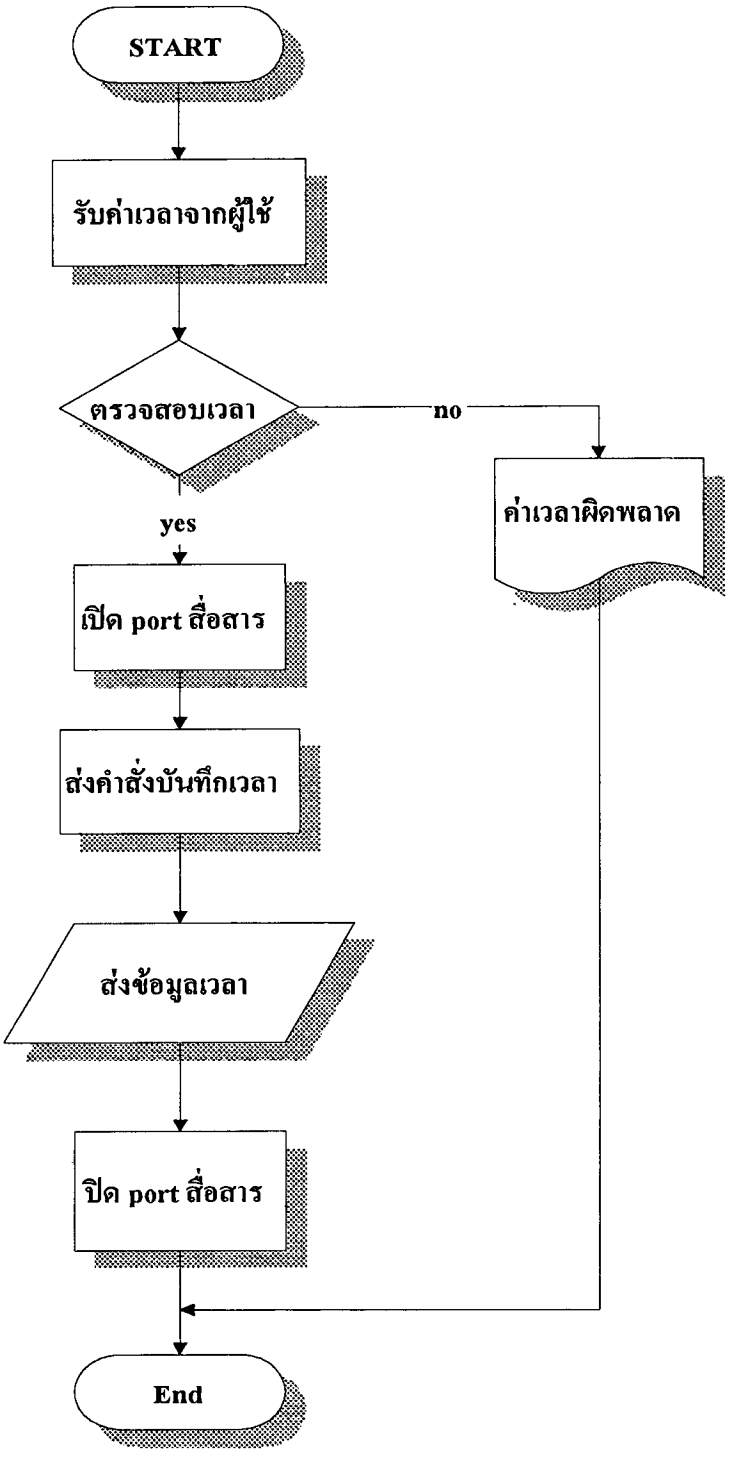
โฟลวชาร์ตแสดงขั้นตอนการอ่านเวลา

รูปที่ 3.2



โฟลวชาร์ตแสดงขั้นตอนการบันทึกวัน

รูปที่3.3



โฟลวชาร์ตแสดงขั้นตอนการบันทึกวัน

รูปที่ 3.4

3.4.1 ประโยชน์ของการติดต่อกับบอร์ดโดยใช้โปรแกรมจาก GUI

เนื่องจากโปรแกรมที่เลือกใช้คือ Visual Basic นี้ถูกเขียนขึ้นเพื่อใช้ติดต่อกับบอร์ด ทำให้การออกแบบเกี่ยวกับลักษณะการออกคำสั่งระหว่างคอมพิวเตอร์กับบอร์ด สามารถกำหนดได้จากผู้เขียนโปรแกรมโดยตรงโดยผู้ใช้จะทำเพียงออกคำสั่งที่รัดกุม และบอร์ดสามารถเข้าใจได้เท่านั้น ทำให้การติดต่อระหว่างบอร์ดกับผู้ใช้เป็นไปอย่างสะดวกสบาย และเกิดข้อผิดพลาดจากการออกคำสั่งน้อยลงจากการให้ผู้ใช้ออกคำสั่งโดยตรงกับบอร์ด

3.4.2 ลักษณะการรับส่งข้อมูลที่ใช้ควบคุม

การติดต่อระหว่างบอร์ดกับคอมพิวเตอร์ผ่านทาง serial port (com2) โดยใช้รูปแบบการส่งมาตรฐานที่ให้มากับ Visual Basic (custom control) ใช้อัตราการรับส่ง 9600 bps ซึ่งเป็นการส่งแบบปกติผ่านทาง RS 232

สัญญาณควบคุมเครื่องใช้ไฟฟ้าจากบอร์ด

รหัสควบคุมเครื่องใช้ไฟฟ้า 4 บิตหลัง คือ A8 , A7 , A6 , A5

สามารถกำหนดค่าได้โดยถ้าต้องการให้อุปกรณ์ไฟฟ้าเปิดแทนด้วยบิต 1 และถ้าต้องการให้อุปกรณ์ไฟฟ้าปิดแทนด้วยบิต 0 หรือ ให้อุปกรณ์ไฟฟ้าเปิดแทนด้วยบิต 0 และให้อุปกรณ์ไฟฟ้าปิดแทนด้วยบิต 1 ก็ได้แล้วแต่ความสะดวกของผู้จัดทำ

รหัสบอกตำแหน่งของห้องต่างๆ 4 บิตหลัง คือ A4 , A3 , A2 , A1 ซึ่งสามารถกำหนดค่าขึ้นเองได้เช่น

รหัส 0000 แทนห้องที่ 1

รหัส 0001 แทนห้องที่ 2

รหัส 0010 แทนห้องที่ 3

รหัส 0011 แทนห้องที่ 4 เป็นต้น

คำสั่งภายในระหว่างบอร์ดกับคอมพิวเตอร์

O	return prompt
T	send time
Z	adj. time base
E	enter current time
D	enter current date
S	stroke setting
P	program timer
R	read program
C	clear I / O
/	send time once

? help command

3.4.3 การส่งโปรแกรมและบันทึกวันและเวลา

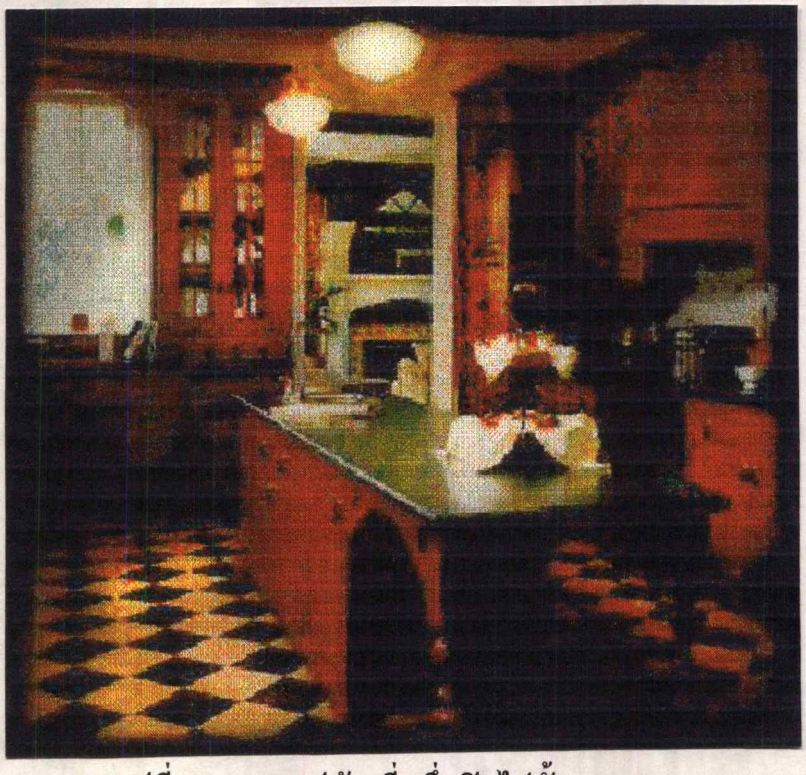
เนื่องจากในการใช้โปรแกรมทุกครั้งจะต้องมีการตรวจสอบวันเวลาที่แน่นอน เพื่อประโยชน์ในการสั่งควบคุมเครื่องใช้ไฟฟ้า จึงต้องมีโปรแกรมการอ่านวันและเวลาจากบอร์ดเพื่อเป็นการตรวจสอบวันและเวลาจากบอร์ดและคอมพิวเตอร์ให้ตรงกันโดยเราสารถที่จะส่งค่าวันเวลาที่ต้องการตั้งค่าลงไปยังบ อร์ดได้พร้อมกับการส่งโปรแกรมการควบคุมเครื่องใช้ไฟฟ้าลงไปได้ด้วย

3.4.4 การอ่านโปรแกรมและการแสดงค่าที่อ่านได้

การอ่านโปรแกรมจากบอร์ดมีความจำเป็นเนื่องจากบอร์ด real-time ในโครงงานพิเศษนี้มีการเก็บข้อมูลลงใน RAM บนไมโครโปรเซสเซอร์ 89C2051 ทุกครั้งในการใช้งานผู้ใช้สามารถอ่านข้อมูลโปรแกรมจากบอร์ดได้

3.5 รูปแบบของ GUI ที่นำมาใช้งานในโครงงานพิเศษ

โปรแกรม GUI ที่นำมาใช้งานในโครงงานพิเศษนี้มีการออกแบบให้ผู้ใช้สามารถใช้งานได้อย่างมีประสิทธิภาพ และมีรูปแบบที่สวยงาม โดยการควบคุมเครื่องใช้ไฟฟ้านั้นสามารถทำได้ง่ายเพียงเลือกกำหนดสถานะเปิด - ปิดของอุปกรณ์ไฟฟ้าจากห้องต่าง ๆ ที่ทำการเขียนโปรแกรมจำลองขึ้นมาเพื่อความสะดวกของผู้ใช้โดยที่ผู้ใช้ต้องการให้ไฟในห้องต่าง ๆ เปิดหรือปิด ก็สามารถที่จะใช้เมาส์คลิกไปยังดวงไฟได้ทันที ซึ่งในโครงงานพิเศษนี้มีห้องต่างๆให้ผู้ใ ช้ สามารถเลือกได้ดังต่อไปนี้



รูปที่ 3.5 แสดงรูปห้องที่1 ซึ่งเปิดไฟทั้งสองดวง

รูปที่ 3.6 แสดงรูปห้องที่ 1 ซึ่งปิดไฟ 1 ดวง

รูปที่ 3.7 แสดงรูปห้องที่ 1 ซึ่งปิดไฟ 1 ดวง

รูปที่ 3.8 แสดงรูปห้องที่ 1 ซึ่งปิดไฟทั้งสองดวง

รูปที่ 3.9 แสดงรูปห้องที่ 2

รูปที่ 3.10 แสดงรูปห้องที่ 3

รูปที่ 3.11 แสดงรูปห้องที่ 4

บทที่ 4
ทฤษฎีและหลักการที่เกี่ยวข้อง

4.1 Real-time Controller

ไมโครคอนโทรลเลอร์แบบชิพเดี่ยว (Single chip Microcontroller) คือไมโครคอมพิวเตอร์แบบที่มีขนาดเล็กโดยบรรจุไว้ในแผงวงจรรวม (Integrated circuit) เพียงชิพเดี่ยวเหมาะสำหรับงานควบคุมอุปกรณ์อื่น ๆ แบบอัตโนมัติ เพราะผู้ใช้สามารถเขียนโปรแกรมควบคุมการทำงานได้ตามต้องการ ซึ่งจะมีรีจิสเตอร์ฟังก์ชันพิเศษที่เรานำมาใช้ในการประยุกต์เข้ากับโครงงานพิเศษชิ้นนี้คือ

4..1.1 ไทเมอร์/เคาน์เตอร์

ไทเมอร์คือการต่ออนุกรมกันไปของฟลิปฟอลปที่ทำหน้าที่หารสอง โดยสัญญาณนาฬิกาจากแหล่งกำเนิดป้อนเป็นอินพุทของฟลิปฟลอปตัวแรก ฟลิปฟลอปตัวแรกจะทำหน้าที่หารสัญญาณนาฬิกาสองเอาท์พุทที่ได้จากฟลิปฟลอปตัวแรกจะถูกป้อนเป็นสัญญาณอินพุทของฟลิปฟลอปตัวที่สอง ฟลิปฟลอปตัวที่สองจะทำการหารสอง และป้อนเอาท์พุทไปยังตัวที่สามเป็นเช่นนี้เรื่อย ๆ ไป ดังนั้นถ้ามีฟลิปฟลอป N ตัว สัญญาณนาฬิกาจากแหล่งกำเนิดก็จะถูกหารด้วย 2^n เอาท์พุทของฟลิปฟลอปตัวสุดท้ายจะถูกป้อนเข้าเป็นอินพุทของฟลิปฟลอปอีกตัวหนึ่งเรียกฟลิปฟลอปตัวนี้ว่า โอเวอร์โฟล์วฟลิปฟลอป หรือ โอเวอร์โฟล์วแฟลก ซึ่งสามารถตรวจสอบได้โดยใช้ซอร์ฟแวร์หรือกระตุ้นให้เกิดการอินเทอร์รัพได้

Application ที่ใช้ไมโครคอนโทรลเลอร์จำนวนมากต้องการนับเหตุการณ์ภายนอกเช่น ความถี่พัลส์ หรือการสร้างความแน่นอนของการหน่วงเวลาภายใน ระหว่างการทำงานของคอมพิวเตอร์ทั้ง 2 อย่างนี้สามารถทำให้สำเร็จโดยใช้เทคนิคซอฟต์แวร์

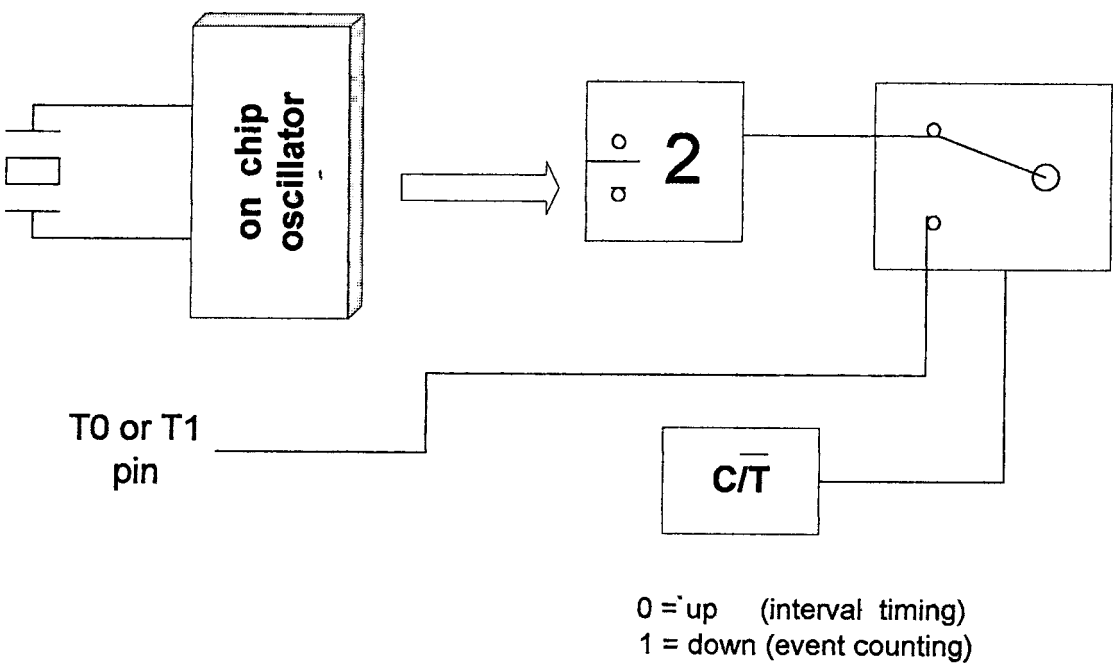
การใช้เป็นตัวจับเวลา (Timer)

ถ้า C/T ใน TMOD เป็นลอจิก "0" จะเป็นการเลือกให้ Timer นำ Clock มาจากวงจร Oscillator ในชิพ ซึ่งสัญญาณนาฬิกาจะเข้ามาทุก ๆ Machine Cycle หรือ อาจกล่าวได้ว่าค่าใน THx และ TLx จะมีคาเพิ่มขึ้นด้วยอัตราการนับแต่ละครั้งใช้เวลาเท่ากับ 1/12 ของความถี่ของสัญญาณนาฬิกาที่ใช้บนชิพ ดังแสดงในรูปที่ 5.3 ถ้า MCS-51 ใช้สัญญาณนาฬิกา 12 MHz การนับจะมีความถี่เท่ากับ 1 MHz

การใช้เป็นตัวนับ (Counter)

ถ้าบิต C/ T เป็น "1" ตัว Timer จะนำ Clock มาจากภายนอกโดยใช้ขา P3.4 หรือ T0 เป็นขา Input Clock ให้กับ Timer 0 และใช้ขา P3.5 หรือ T1 เป็น Input Clock ให้กับ Timer 1 ดังรูปที่ 5.3 หรืออาจมองว่า ถ้าจะให้นับอะไรสัญญาณที่จะนับให้ต่อกับขา T0 และ T1 ในการใช้เป็น Counter สัญญาณที่เข้ามามีการเปลี่ยนแปลงจาก "1" เป็น "0" จะทำให้วงจรนับ TLx มีค่าเพิ่มขึ้น 1 ภายใน MCS -51 นี้จะตรวจสอบขาอินพุท T0 และ T1 ในช่วงเวลาเฟส 2 ของ State 5 (S5P2) ลอจิกอินพุทเปลี่ยนเป็น "0" จะทำให้ค่าใน Timerเพิ่มขึ้น 1 ดังนั้น จะเห็นได้ว่าการนับ 1 ครั้งจะต้องใช้เวลา 2 Machine Cycles ดังนั้นความถี่สูงสุดที่จะให้ Timer ทำงานเป็น Counter นับได้จะมีค่ามากที่สุด 500 kHz ถ้า MCS-51 ทำงานที่ความถี่สัญญาณนาฬิกา 12 MHz

โดยเราจะนำโหมด Timer และ counter นี้มาประยุกต์ให้ได้สัญญาณนาฬิกาที่เราต้องการดังรูปต่อไปนี้



รูปที่ 4.1 ความถี่ของสัญญาณนาฬิกาที่จะเข้า Timer

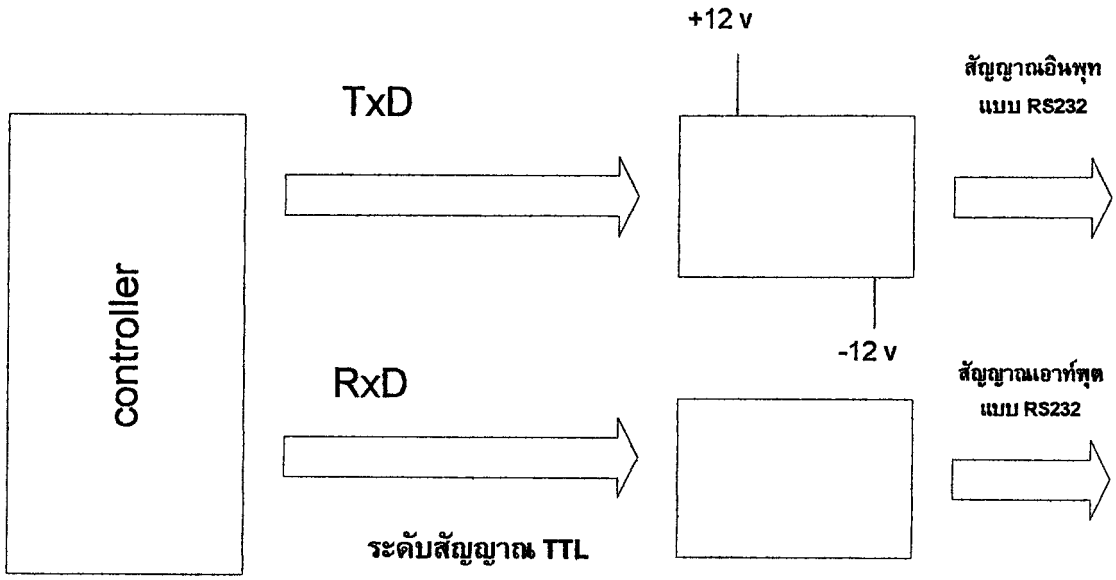
โดยเราจะใช้ในโหมด timer ซึ่งจะใช้ crystal ความถี่ 3.579 MHz แล้วทำการคำนวณ

4.1.2 การติดต่อสื่อสารข้อมูลอนุกรม

พอร์ทอนุกรมที่สามารถสั่งงานให้ทำงานได้หลายโหมดอยู่ภายในชิพ การทำงานเป็นแบบฟลูดูเพลก (full duplex) หมายถึงสามารถรับและส่งข้อมูลในช่วงเวลาเดียวกันได้ มีรีจิสเตอร์ตัวหนึ่งที่ทำหนาที่เป็นบัฟเฟอร์ ทางด้านรับข้อมูลช่วยให้ตัวอักษรที่รับได้เรียบร้อยแล้วถูกนำมาเก็บไว้ในบัฟเฟอร์ในขณะที่กำลังรับตัวอักษรตัวที่สอง ด้วยเหตุนี้ถ้า ซีพียูอ่านตัวอักษรตัวแรกออกไปก่อนที่การรรับตัวอักษณตัวที่สองจะรับเสร็จข้อมูลจะไม่สูญหาย บัฟเฟอร์อีกตัวหนึ่งใช้สำหรับเก็บข้อมูลก่อนที่จะส่งออกไป

การเชื่อมต่อแบบมาตรฐาน RS-232C

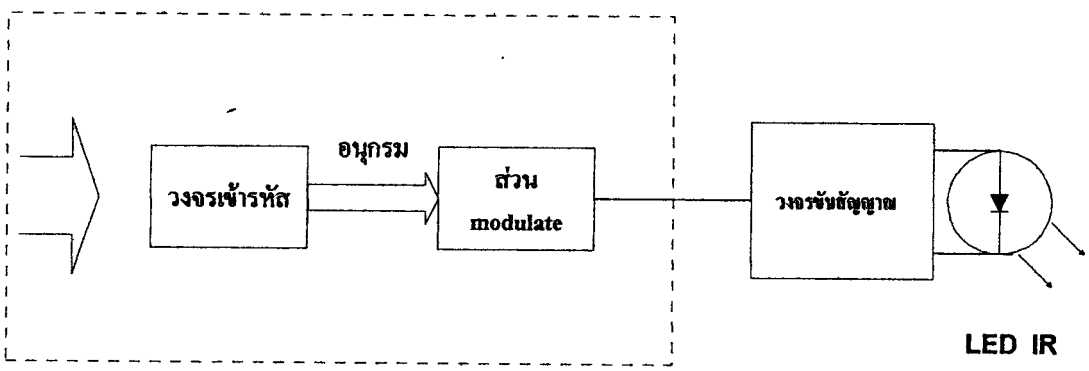
ในการเชื่อมต่อแบบอนุกรมเข้ากับอุปกรณ์คอมพิวเตอร์ต่าง ๆ เช่น คอมพิวเตอร์ เทเลกซ์ หรือ โทรพิมพ์ เป็นต้น มักจะกำหนดใช้การเชื่อมต่อตามมาตรฐาน RS - 232 C ทั้งนี้เพื่อให้มีการใช้งานเส้นสัญญาณหรือรูปแบบของตัวเชื่อมที่สอดคล้องกัน จะได้ลดปัญหาการเข้ากันไม่ได้ระหว่างสัญญาณของอุปกรณ์ที่มาเชื่อมต่อกันทั้งสองด้านให้น้อยลง เนื่องจากระดับโวลเตจที่ใช้และการแทนความหมายของระดับลอจิกตามมาตรฐานนี้แตกต่างไปจากที่ใช้งานกันในระบบดิจิตอลทั่วไป โดยระดับสัญญาณของ RS-232C เป็นแบบไบโพลาร์ (Bipolar) ระดับโวลเตจทางด้านลบช่วง -3 V ถึง -20V แทนค่าลอจิก 1 และโวลเตจทางด้านบวกช่วง +3 V ถึง +20 V แทนค่าลอจิก 0 ดังนั้น จะเห็นว่ามีความจำเป็นต้องเพิ่มเติมอุปกรณ์หรือวงจรพิเศษเข้าไป เพื่อเปลี่ยนระดับโวลเตจจากระบบ 0V ถึง +5V จากขาสัญญาณของ 8051 เป็นระดับโวลเตจที่สูงกว่าค่า +3.0 V หรือต่ำกว่า -3.0 V ดังในรูปที่ 8.17 ซึ่งแสดงให้เห็นว่าระดับสัญญาณแบบ TTL จากขาสัญญาณ TxD และRxD ของ 8051 จะต้องถูกปรับเปลี่ยนไปเป็นระดับสัญญาณ RS-232C ก่อน ที่จะทำการส่งออกไปในสายนำสัญญาณต่อไป



รูปที่ 4.2 แผนภาพแสดงให้เป็นถึงแนวการเปลี่ยนแปลงระดับสัญญาณแบบ TTL จากไมโครคอนโทรลเลอร์ 89C2051 ไปเป็นระดับสัญญาณแบบ RS232 และการแปลงระดับสัญญาณอินพุทแบบ RS232 ไปเป็นระดับสัญญาณแบบ TLL ก่อนที่จะได้เชื่อมต่อกับขาสัญญาณของไมโครคอนโทรลเลอร์ 89C2051

4.2 ภาคการส่งอินฟราเรด

จากรูปเป็นการแสดงบล็อคไดอะแกรมของวงจรในภาคส่งซึ่งประกอบไปด้วยวงจรเข้ารหัสที่รับสัญญาณ input เป็นแบบขนาน เมื่อผ่านตัววงจรเข้ารหัสแล้วก็จะเข้ารหัสออกมาเป็นแบบสัญญาณอนุกรมจากนั้นจะทำการมอดูเลท เพื่อให้เกิดสัญญาณพาห์ขึ้น แล้วทำการส่งต่อไปยังภาควงจรขับสัญญาณเพื่อเป็นการป้อนสัญญาณพร้อม ๆ กับทำการขยายสัญญาณไปด้วยในตัวกระตุ้นให้เกิดการแปล่งแสงของ LED อินฟราเรดส่งออกไป



รูปที่ 4.3 บล็อคไดอะแกรมภาคส่งอินฟาเรด

การแปลงรหัส (The Code conversion function)

ในวงจรแบบดิจิตอลนั้นเราจะใช้ความต่างศักย์ 2 ค่า คือ 0 กับ 5 โวลต์ ซึ่งเราแทนให้เป็นเลข 0 และ 1 ตามลำดับ ในการใช้งานที่หลากหลายจำเป็นต้องใช้ ค่า 2 ค่านี้ไปแทนความหมาย จึงทำให้เราต้องใช้รหัส

รหัสเป็นกลุ่มของบิตที่จัดเตรียมขึ้น เป็นรูปแบบหนึ่ง รูปแบบใดซึ่งนำไปใช้ในการแทนข้อมูลเฉพาะอย่าง การใช้งานรหัสนั้นขึ้นอยู่กับผู้ตั้งรหัสจะเป็นผู้กำหนดความหมายว่าแต่ละรหัสนั้นหมายถึงอะไร หรืออาจมีรหัสที่เป็นกลางที่มีความหมายให้คนจำนวนมากเข้าใจและนำไปใช้ได้ ตัวอย่างเช่นในงานทางดิจิตอลก็จะมีการเรียงบิตในรูปแบบไบนารีโค้ด หรือ แบบรหัสเกรย์ เป็นต้น

ในการใช้งานนั้นเราจะมีเครื่องมือที่จะทำการแปลงรหัสรูปแบบหนึ่งไส่รหัสอีกรูปแบบหนึ่ง ซึ่งเป็นวิธีพื้นฐานในการส่งรหัสติดต่อกับอุปกรณ์ต่าง ๆได้ การเปลี่ยนรหัส พอยกตัวอย่างได้เช่น รหัสแบบ BCD และรหัสเกรย์

4.2.1 การเข้ารหัส (The Encoding Function)

การเข้ารหัสจะทำได้โดยวงจรอย่างหนึ่งให้ชื่อว่าวงจรเข้ารหัส การเข้ารหัสง่ายๆ ที่เราพอเข้าใจก็เช่น การเปลี่ยนตัวเลขฐานสิบหรือ รูปตัวอักษรต่าง ๆให้อยู่ในรูปแบบรหัส ซึ่งรหัสนี้เราอาจจะคุ้นเคย เช่น รหัสแอสกี ในคอมพิวเตอร์

การเข้ารหัสนั้นจะขึ้นอยู่กับการออกแบบวงจรว่าจะทำการเข้ารหัสที่มีข้อมูลขาเข้าเป็นอะไรและจะให้ข้อมูลขาออกเป็นอย่างไร อย่างเช่นการเปลี่ยนเลขฐานสิบตั้งแต่ 0-9 ไปเป็นรหัสไบนารี (เลขฐานสอง) ซึ่งการเข้ารหัสแบบนี้มีวงจรที่อยู่ในรูปชิพ มีจำหน่ายอยู่มากมายตามท้องตลาด นอกจากนี้วงจรเข้ารหัสยังสามารถใช้ร่วมกับการส่งข้อมูลทางคลื่นวิทยุได้อีกด้วย ซึ่งบางวงจรที่ได้ออกแบบก็ได้มีการกำหนดเฉพาะสัญญาณขาเข้าและข้อมูลขาออกเป็นแบบใด ซึ่งเวลาจะเลือกใช้จำเป็นจะต้องอ่านคู่มือของวงจรที่ให้มาก่อน เพื่อเลือกวงจรที่เหมาะสมกับงานที่เราต้องการ

สำหรับการเข้ารหัสที่เราได้นำมาประยุกต์ใช้กับโครงงานพิเศษชุดนี้คือ การเข้ารหัสแบบขนาน ให้เป็นแบบอนุกรมซึ่งจะมีสัญญาณข้อมูลเฉพาะแบบที่นำมาใช้ได้

Basic shift Register function

ส่วนประกอบภายในชิพรีจิสเตอร์ จะประกอบไปด้วยการจัดเรียงตัวของฟลิปฟลอป ซึ่งมีความสำคัญในการนำไปใช้เกี่ยวข้องกับการเก็บและการเคลื่อนย้าย ข้อมูลในระบบดิจิตอลพื้นฐาน ความแตกต่างระหว่างรีจิสเตอร์และตัวนับ(counter) นั่นคือ รีจิสเตอร์จะไม่มีลำดับเฉพาะในตัวเอง ยกเว้นจะนำไปใช้ในงานที่พิเศษ ตัวรีจิสเตอร์โดยทั่วไปจะถูกใช้เพียงเก็บและเลื่อนข้อมูล (1และ 0) ให้เข้าไปสู่ตัวมันจากแหล่งข้อมูลภายนอกและไม่มีสถานะลำดับภายใน

รีจิสเตอร์เป็นวงจรดิจิตอลซึ่งมีหน้าทที่หลัก 2 อย่างคือ เก็บข้อมูลและเลื่อนข้อมูลความจุของการเก็บของรีจิสเตอร์นั้นคือเป็นสิ่งที่สำคัญขง อุปกรณ์หน่วยความจำ การเก็บข้อมูล 1 หรือ 0 ในฟลิปฟลอป แบบ D ฟลิปฟลอป เมื่อมีข้อมูล 1 เป็นข้อมูลขาเข้าให้กับอุปกรณ์ดังรูป และสัญญาณนาฬิกาเมื่อเข้ามาแล้ว ข้อมูล 1 นี้จะถูกทำการเซตในฟลิปฟลอป และเมื่อข้อมูล 1 ที่ด้านขาเข้าย้ายไป ตัวฟลิปฟลอปก็ยังคงอยู่ในสภาวะเซตเป็นลำดับ กระบวนการเก็บข้อมูล 1 แบบนี้จะมีกระบวนการแบบเดียวกัน สำหรับเก็บข้อมูล 0 ด้วย

ค่าความจุของการเก็บของรีจิสเตอร์ จะเป็นจำนวนของบิต (1 และ 0) ของข้อมูลดิจิตอลในแต่ละสเตทของฟลิปฟลอปภายในรีจิสเตอร์จะแทน 1 บิตของความจุการเก็บ ด้วยเหตุนี้จำนวนของสเตทในรีจิสเตอร์จะตัดสินค่าความจุสุทธิได้ ตัวรีจิสเตอร์จะประกอบด้วยฟลิปฟลอป หรืออุปกรณ์อย่างอื่นที่มีคุณสมบัติแบบเดียวกัน ความสามารถในการเลื่อนข้อมูลของรีจิสเตอร์จากสเตทหนึ่งไปสู่สเตทหนึ่ง ภายในรีจิสเตอร์หรือเข้ามาหรือออกจากรีจิสเตอร์นั้นขึ้นอยู่กับสัญญาณนาฬิกาที่เข้ามา รูปแบบของการเลื่อนไปของข้อมูลในชิพรีจิสสเตอร์ ในบล็อคจะแทนรีจิสเตอร์แบบ 4 บิต และลูกศรจะแสดงทิศทางของการเคลื่อนที่ของข้อมูล คุณสมบัติของรีจิสเตอร์นั้นถูกนำไปเป็นส่วนหนึ่งของอุปกรณ์หลายตัว เช่น อุปกรณ์หน่วยความจำ , คอนโทรลเลอร์ต่าง ๆ เป็นต้น คุณสมบัติของมันสามารถทำให้เราส่งข้อมูลออกเข้าหรือเก็บข้อมูลไว้ตรวจสอบเงื่อนไขต่าง ๆ จึงถือว่าเป็นส่วนสำคัญที่จะต้องใช้รวมกับอุปกรณ์ประมวลผลแบบต่าง ๆ ไม่น่าจะเป็นคอมพิวเตอร์ , ไมโครโปรเซลเซอร์หรือไมโครโปสเซลเซอร์หรือ ไมโครคอนโทรเลอร์ ในงานด้านการส่งผ่านข้อมูลต่าง ๆ จึงจำเป็นต้องใช้อุปกรณ์รีจิสเตอร์ในการใช้งานชิพต์รีจิสเตอร์ แบบเข้าอนุกรมออกแบบขนาน (Serial in - Parallel out shift register)

4.2.2 รีจิสเตอร์ข้อมูลเข้าแบบขนานออกแบบอนุกรม

ลักษณะการทำงานของรีจิสเตอร์แบบข้อมูลเป็นขนานและข้อมูลออกเป็นแบบอนุกรม อธิบายได้ดังนี้ เมื่อมีข้อมูลส่งเข้ามาในแต่ละสเตทโดยตรง โดยที่แต่ละสเตทจะเรียงตัวแบบอนุกรม เมื่อข้อมูลเข้ามาในแต่ละสเตทนั้น เมื่อสัญญาณนาฬิกาจะไปทำให้ข้อมูลส่งไปสู่ แต่ละสเตทเรียงกันไปเป็นการส่งข้อมูลแบบอนุกรมนั่นเอง

รีจิสเตอร์ชนิดนี้จะต้องอาศัยอุปกรณ์อย่างอื่นมาช่วย เช่น เกท นั่นหมายความว่าวงจรจะมีความซับซ้อนกว่าแบบที่ผ่านมา

ส่วนใหญ่การใช้งานรีจิสเตอร์มักใช้ในภาคส่งสัญญาณระยะไกลในระบบสื่อสารดิจิตอล ไม่ว่าจะใช้กับวิยุสื่อสาร หรือ ใช้กับรีโมตคอนโทรลซึ่งมีนิยมใช้กับรีโมต มากที่สุด ส่วนในงานด้านอื่นๆ ก็มักจะเกี่ยวข้องกับระบบภายในของไมโครคอมพิวเตอร์ทั้งหลาย ๆ แบบ ซึ่งการรับส่งข้อมูลในไมโครโปรเซสเซอร์มีหลายแบบอยู่แล้ว หากไม่ใช้กับการสื่อสารใช้สาย ก็ยังใช้กับการสื่อสารแบบใช้สายทองแดงก็ได้ ซึ่งจะมีประโยชน์ในการประหยัดสายสัญญาณ โดยในการทำงานลักษณะนี้จะใช้คู่กับรีจิสเตอร์แบบข้อมูลเข้าแบบอนุกรมออกแบบขนาน โดยจะแบ่งกันไปทำงานในภาคส่งสัญญาณ รับสัญญาณตามลำดับ จึงนับว่าได้อุปกรณ์รีจิสเตอร์เป็นส่วนประกอบสำคัญในวงจรอิเล็กทรอนิกส์แทบจะขาดไม่ได้เลยก็ว่าได้

4.2.3 Modulation

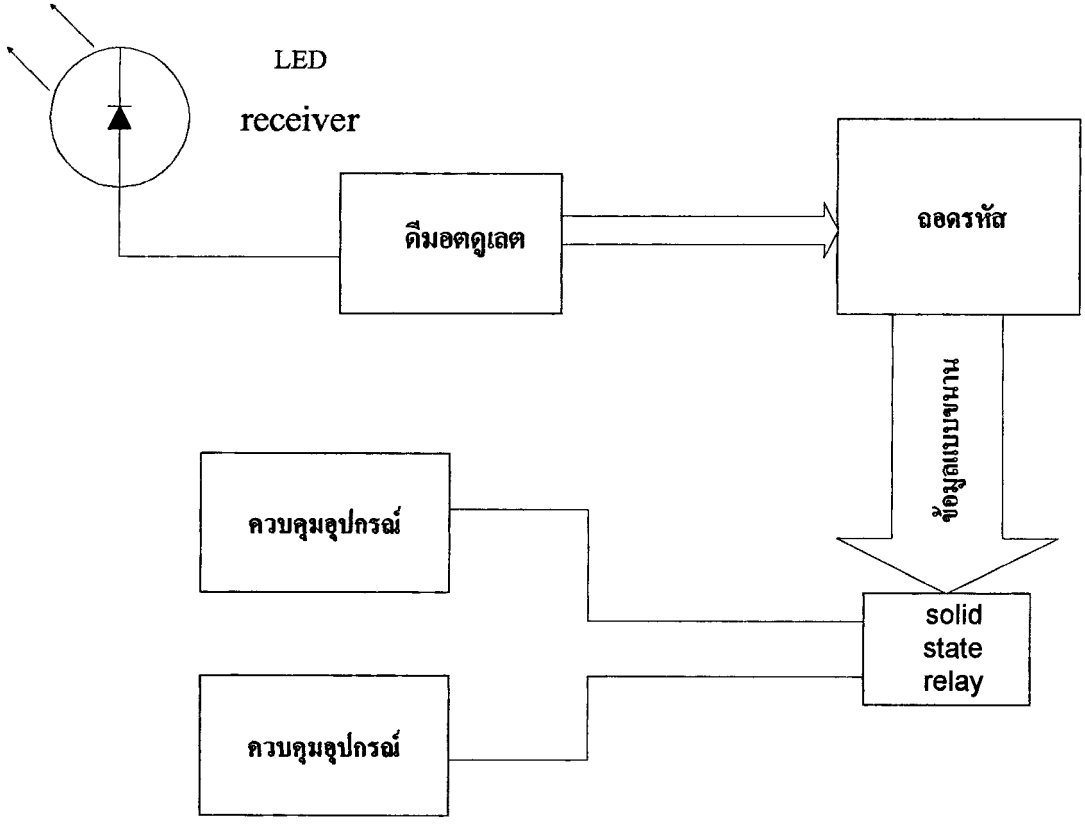
เทคนิคอีกวิธีหนึ่งที่เราใช้ในการสื่อสารก็คือการมอดูเลต คือกระบวนการของข้อมูลที่ถูกส่งออกไปที่สัญญาณความถี่สูงเพื่อจุดประสงค์ของการส่งของข้อมูลที่ใดที่หนึ่งในแถบสนามแม่เหล็กไฟฟ้า ตัวอย่างเช่น วิทยุ , สายไฟ , เส้นใยนำแสง ถ้าปราศจากการมอดูเลทแล้วการสื่อสารก็จะไม่ได้รู้จักกันจนทุกวันนี้

สัญญาณข้อมูลเช่นเสียง , วีดีโอ , หรือข้อมูลแบบไบนารีที่ถูกส่งออกไปโดยตรงจากจุดหนึ่งไปยังอีกจุดหนึ่งโดยอาศัยตัวกลาง ตัวอย่างเช่น สัญญาณเสียงที่ถูกส่งออกไปทางสายไฟฟ้าในระบบโทรศัพท์ , สายโคแอกเซียลที่ทำการพาสัญญาณวีดีโอระหว่างจุดสองจุด อย่างไรก็ตามเมื่อเราทำการส่งออกไปนั้น ในระยะทางไกล ๆ จะทำให้ได้ผลคลาดเคลื่อนจากของเดิม การสื่อสารแบบวิทยุที่ใช้กันในปัจจุบันนี้สามารถที่จะส่งไปได้ในระยะทางที่ไกลมากขึ้น สัญญาณความถี่สูงจะนำออกมาใช้ ด้วยเหตุผลนี้เองมันควรจะใช้สัญญาณข้อมูลที่มีความสูงกว่า ในความถี่สนามแม่เหล็กไฟฟ้า ในกระบวนการมอดูเลตนี้จะสร้างสัญญาณความถี่สูง บรรจุเข้ากับสัญญาณความถี่เดิม

การมอดูเลทที่สามารถที่จะปรับปรุงคุณลักษณะของสัญญาณให้สามรถเข้ากันได้ ซึ่งส่วนใหญ่สัญญาณข้อมูลเสียง , วีดีโอ , ไบนารี หรือสัญญาณข้อมูลอื่นต่าง ๆ ในทางปกติเราจะใช้สัญญาณความถี่ที่สูงกว่าซึ่งเราเรียกกันว่า “คลื่นพาห์” และสัญญาณข้อมูลที่เราใช้อยู่เรียกว่า “สัญญาณมอดูเลท” ในทางปฏิบัติส่วนใหญ่เราจะพิจารณาให้คลื่นพาห์มีค่าสูงกว่าค่าสัญญาณที่สูงที่สุดของความถี่ข้อมูลที่ถูกส่งออกไป

4.3 ภาครับอินฟราเรด

บล็อคไดอะแกรมจะแสดงภาครับสัญญาณ โดยเริ่มจากโฟโต้ไดโอดที่ทำหน้าที่เปลี่ยนแปลงสัญญาณแสงให้เป็นสัญญาณไฟฟ้า เมื่อได้สัญญาณไฟฟ้าออกมาแล้ว เราก็จะทำการดีมอดูเลทออกก่อนเพื่อเป็นการแยกสัญญาณต่าง ๆที่เป็นคลื่นความถี่สูงหรือคลื่นพาหะที่เราทำการมอดูเลทเข้ามา และเข้าสู่วงจรถอดรหัสให้ทำการแปลงสัญญาณกลับออกมาและได้ผลลัพธ์ออกมาเป็นแบบขนานเพื่อทำการนำสัญญาณขาออกต่าง ๆ ไปควบคุมอุปกรณ์ไฟฟ้าผ่านการติดต่อกับรีเลย์ต่อไป



รูปที่ 4.4 บล็อคไดอะแกรมภาครับอินฟาเรด

4.3.1 การถอดรหัส (The Decoding function)

การถอดรหัส เราจะได้จากวงจรที่เรียกว่า วงจรถอดรหัส ซึ่งเปลี่ยนจากรหัสข้อมูล เช่น เลขฐานสิบ

ในการทำงานควบคู่กันระหว่างไมโครคอมพิวเตอร์ , ไมโครโปเซสเซอร์ หรือ ไมโครคอนโทรลเลอร์ที่มีการต่อกับอุปกรณ์ภายนอก เช่น Key board , จอมอนิเตอร์ , เมาส์ และอื่น ๆ ถ้าไมโครคอนโทรลเลอร์จะทำการส่งสัญญาณสั่งงานมาในรูปของรหัสส่วนใหญ่เช่นเลขฐานสอง , ฐานสิบหก , หรือ แอสกี้ ดังนั้นวงจรถอดรหัสจึงเป็นวงจรที่จำเป็นตลอดเวลาสำหรับงานเชื่อมต่อกับไมโครคอนโทรลเลอร์ หรือแบบชนิดอื่นที่ได้กล่าวมาแล้ว

ตัวอย่างของการทำงาน เช่น รูปแบบของตัวถอดรหัสที่ทำการเปลี่ยนรหัส 4 บิตไปเป็นดิจิตแบบฐานสิบที่เหมาะสม

ตัวถอดรหัสที่ใช้ในการขับ 7 เซกเมนต์ แต่ละส่วนของ 7 เซกเมนต์จะต่อกับเส้น output ของตัวถอดรหัส เมื่อมีรหัสไบนารีโค้ดที่ต้องการเข้ามา ที่ขาเข้าของตัวถอดรหัส จะทำให้ที่ขา output จะไปขับให้จอแสดงผลแบบ 7 เซกเมนต์สว่างซึ่งจะแสดงในรูปฐานสิบ

4.3.2 รีจิสเตอร์แบบข้อมูลเข้าอนุกรมออกแบบขนาน (SIPO)

การทำงานในลักษณะข้อมูลเข้าให้เป็นแบบอนุกรมและข้อมูลออกเป็นแบบขนานนั้นพอจะอธิบายได้ดังนี้ เมื่อข้อมูลเข้ามาในลักษณะอนุกรมนั้นที่ตัวฟลิปฟลอป ที่อยู่ภายในรีจิสเตอร์ก็จะรับข้อมูลไว้ตามจังหวะสัญญาณนาฬิกา และก็จะมีการเลื่อนข้อมูลไปสู่ฟลิปฟลอปตัวต่อไปเรื่อย ๆ เหมือนกับรีจิสเตอร์ที่ผ่านมา แต่จะมีข้อแตกต่างคือที่ด้านขาออกจะอยู่ที่ฟลิปฟลอปแต่ละตัวไปเลย จึงทำให้ข้อมูลออกมาในแบบขนาน ดังนั้นเมื่อมีข้อมูลเข้ามาแค่ตัวหนึ่ง จะมีข้อมูลอกมาทีเดียวทั้ง 4 บิต ดังรูป 9-8 ดังนั้นการส่งผ่านโปรแกรมลักษณะนี้จะทำให้ได้ข้อมูลออกอย่างรวดเร็ว

ในการประยุกต์ใช้งานรีจิสเตอร์แบบนี้ เรามักใช้ในวงจรถอดรหัสเป็นตัวเสริมคือเราจะสมมุติให้ ขาข้อมูลที่ออกแบบเป็นแบบขนาน ใช้กับอุปกรณ์เป็นจอแสดงผล LED เมื่อเราส่งข้อมูลมาในแบบอนุกรมผ่านการถอดรหัสจนได้สัญญาณมาถึงตัวรีจิสเตอร์นี้ ตัวรีจิสเตอร์นี้ก็จะส่งข้อมูลออกมา ปรากฏที่ LED เลย ซึ่งจะทำให้เราเห็นไฟวิ่งในแถว LED เป็นต้น

อุปกรณ์ในคุณสมบัติมักใช้ในงานดิจิตอลทั่วไป อย่างกว้างขวางโดยเฉพาะใส่ร่วมกับภาครับของวิทยุอื่น ๆ มีอุปกรณ์ให้ใช้มากเป็นเบอร์ 74164 เป็นต้น

ดังนั้นการส่งข้อมูลในลักษณะนี้จะทำให้เราเห็นข้อมูลที่ผ่านฟลิปฟลอปทีละตัวด้วยตาโดยตรง ซึ่งจะทำให้การส่งผ่านข้อมูลมีความเร็วกว่าแบบข้อมูลเข้าอนุกรมออกอนุกรมซึ่งจะช้าที่สุด

4.4 รีเลย์ (Relay)

เป็นอุปกรณ์ที่ใช้กระแสต่ำเพื่อควบคุมสวิทช์ให้ติดต่อโหลดที่มีกระแสสูง ๆ การทำงานของรีเลย์คือเมื่อมีแรงดันตกคร่อมขดลวดจะทำให้เกิดกระแสผ่านขดลวด ซึ่งจะทำให้หน้าสัมผัสเคลื่อนที่ (moving contact) เกิดสนามแม่เหล็กไฟฟ้าดูดหน้าสัมผัส NO (ปกติเปิดวงจร normally open) ให้ต่อวงจร และเมื่อปลดแรงดันออก สนามแม่เหล็กก็จะหมดลง หน้าสัมผัสเคลื่อนที่ก็จะดีดกลับมายังหน้าสัมผัส NC (ปกติต่อวงจร normally close)

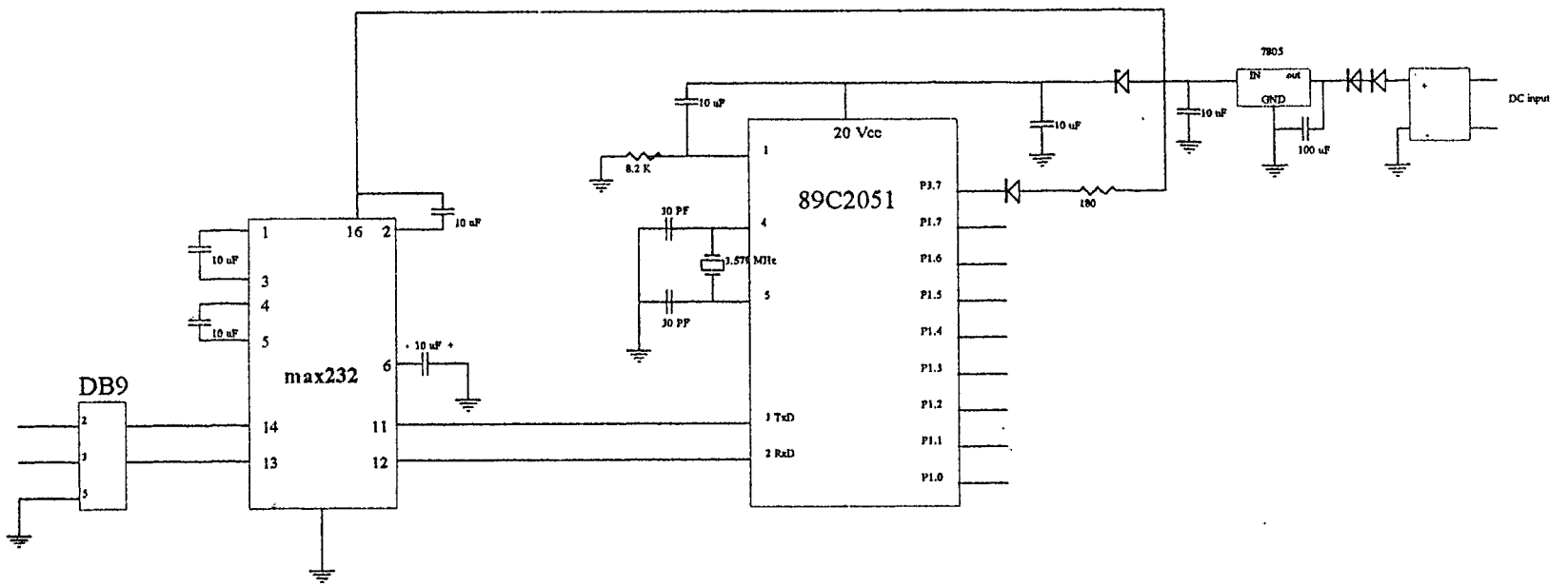
รีเลย์เป็นอุปกรณ์ที่มีความเร็วในการทำงานต่ำ ดังนั้นรีเลย์ชนิดแรงดันต่ำจะใช้เวลาในการทำงานราว 15-30 มิลลิวินาที ซึ่งได้มีการปรับปรุงเป็นการใช้ Solid State relay แทนซึ่งจะช่วยลดปัญหาหน้าสัมผัสลงได้ เป็นอย่างมากและเมื่ออัตราการ Switch ที่เร็วขึ้นด้วย

บทที่ 5

การดำเนินงานวิจัย

5.1 วงจร Real - time controller

จากรูปที่ 5.1 จะแสดงถึงวงจรที่เราใช้ทำในการทำงานของไมโครคอนโทรลเลอร์แบบชิพเดี่ยวเพื่อเป็นศูนย์การประมวลผลของเรา โดยที่เราจะทำการใช้ไมโครคอนโทรลเลอร์ในกาติดต่อเขียนโปรแกรมภาษาแอสเซมบลีบนคอมพิวเตอร์ผ่านตัว RS 232 ซึ่งเป็นตัวรับสัญญาณจากพอร์ทอนุกรมออกมาติดต่อกับไมโครคอนโทรลเลอร์เบอร์ 89C2051 ซึ่งจะมีขาการติดต่อจาก RS 232 คือ ขา 3 ที่เป็นตัวส่งผ่านข้อมูลกับ RS 232 และตัวรับข้อมูลทีขา 2 โดย โปรแกรมที่เราเขียนนี้จะใช้หลักของตัวไทเมอร์และเคาท์เตอร์เสียเป็นส่วนใหญ่ในการเลือกโหมดไทเมอร์ เราจะอาศัย clock สัญญาณนาฬิกาจากคริสตอล 3.579 Mhz และนำไปเขียนโปรแกรมเพื่อทำการับเวลาให้เป็น 1 วินาที และทำการนับเพิ่มค่าขึ้นไปจนเหมือนกับเกิดสัญญาณเป็นนาฬิกาเพื่อบอกถึงวันและเวลาให้เราทราบ หลังจากนั้นเราก็จะทำการเขียนเอาท์พุทออกที่พอร์ทข้อมูลแบบขนานที่พอร์ท 1 ซึ่งจะหมายถึงข้อมูลที่เราจะนำไปใช้ต่อไป ในการใช้งานวงจรนี้ทั้งหมดจะใช้แรงดันไฟฟ้าในการต่อเข้ากับทั้งตัวไมโครคอนโทลเลอร์และตัว MAX 232 ประมาณ 5 โวลต์ โดยเราจะทำการเรกกูเลเตอร์โดยการใช้ 7805 ซึ่งจะสามารถแปลงแรงดันไฟฟ้า DC จากโวลต์สูงให้เป็นโวลต์ต่ำลงได้ โดยแสดงไว้ดังรูปที่ 5.1



รูปที่ 5.1 วงจร REAL-TIME CONTROLLER

5.2 วงจรภาคส่งอินฟราเรด

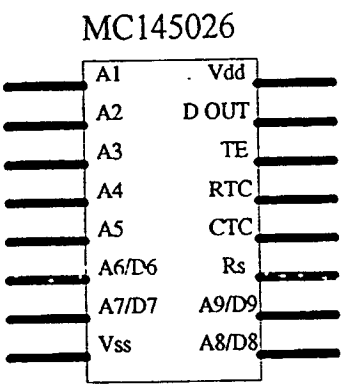
5.2.1 หลักการทำงานของ MC 145026

ตัว MC 145026 นี้จะถูกออกแบบมเพื่อใช้งานเป็นตัวเข้ารหัสที่ใช้การประยุกต์ตัวงานแบบการควบคุมระยะไกล

ตัว MC 145026 นี้ทำการเข้ารหัสแบบ 9 สายของสัญญาณข้อมูล และส่งสัญญาณออกมาเป็น แบบอนุกรม (1 สาย) เมื่อได้รับสัญญาณการส่งที่ขา TE หรือ transmitt enable สัญญาณข้อมูลที่ 9 สายนี้สามารถทำการเข้ารหัสด้วยลักษณะข้อมูลแบบ 3 สถานะ (High , Low , open) หรือ แบบ 2 สถานะ (high,Low) มีรายละเอียดของอุปกรณ์ดังนี้

- ทำงานในช่วงอุณหภูมิ -40 °C - 85° C
- ใช้กระแสในการเข้ารหัสที่ต่ำมาก คือ 300 mA ที่ 25 °C
- สามารถเชื่อมต่อเข้ากับ RF , อัลตราโซนิค , อินฟราเรด
- ใช้วงจร RC Oscillator ไม่ต้องการคริสตอลในการกำเนิดสัญญาณ

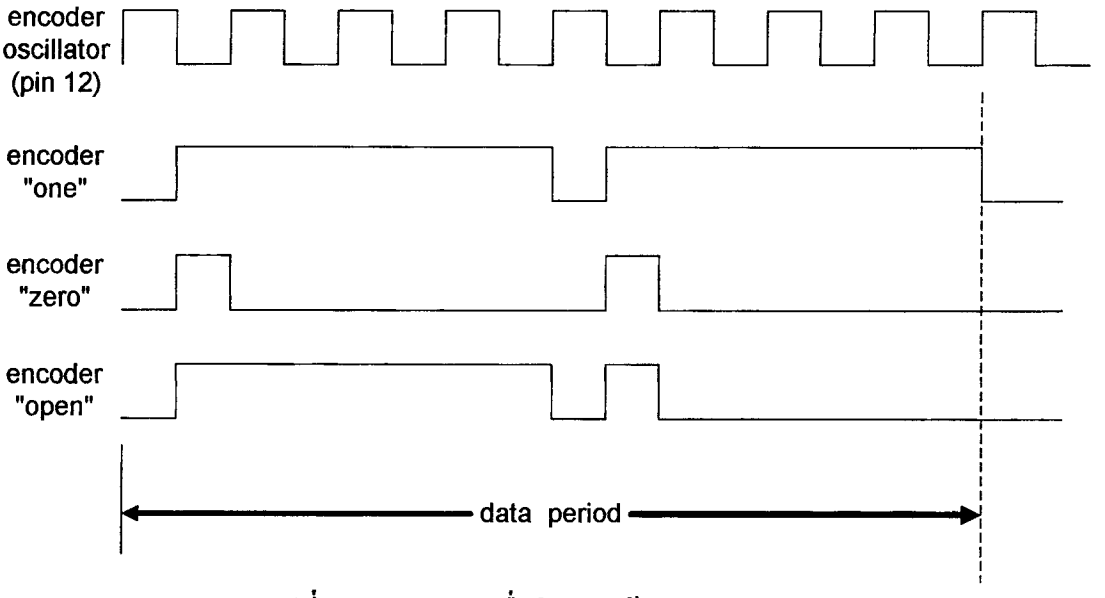
รายละเอียดของขาต่าง ๆ เป็นดังรูป



รูปที่ 5.2 แสดงตัวถังของ MC145026

ตัวเข้ารหัสที่จะส่งออกมาเป็นแบบอนุกรมสถานะตั้งแต่ A1-A5 และ A6/D6 - A9/D9 ซึ่งขาสัญญาณเหล่านี้ทำให้เราสามารถที่จะเข้ารหัสได้ทั้งหมด 19683 รหัส และจะส่งออกมาเมื่อขา TE เป็นสถานะ low และจะส่งออกมาเรื่อย ๆ เมื่อขา TE ยังคงเป็น low อยู่ แต่ละสถานะการส่งออกไปที่เป็นแบบอนุกรมนั้น จะแสดงให้เห็นดังรูปที่ 5.3 ถ้าเป็นลอจิก 0 ก็จะทำการเข้ารหัสเป็น 2 ลูก

คลื่นสั้น ๆต่อกัน ลอจิก High ก็จะเป็น 2 ลูกคลื่นยาว ๆต่อกัน แต่ ถ้าเป็นในสถานะเปิดก็จะเป็นลูกคลื่นแบบยาวและสั้นต่อกันออกมาอย่างละ 2 ลูกคลื่น



รูปที่ 5.3 แสดงลูกคลื่นในการเข้ารหัส

รายละเอียดของตัว MC145026

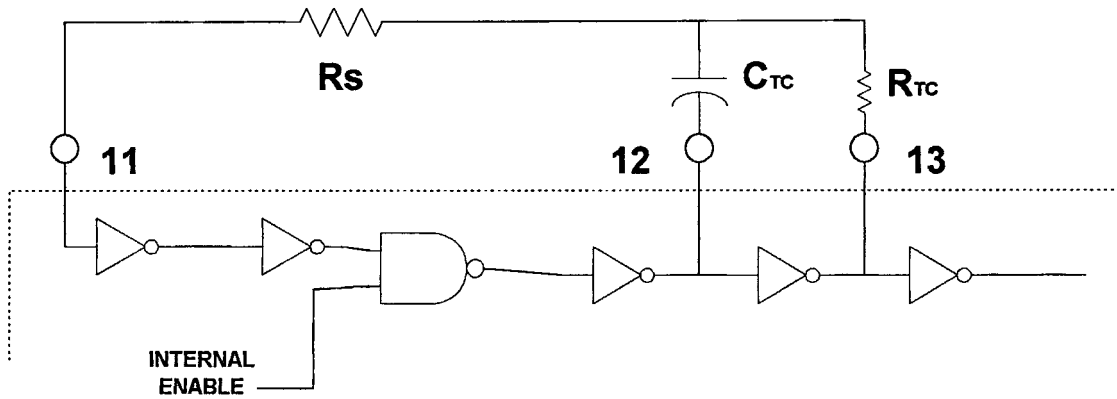
A1-A5 และ A6/D6 - A9/D9

Address , Address / Data input (ขา 1-7 , 9 และ 10)

Input address / Data เหล่านี้จะถูกเข้ารหัส และข้อมูลที่ถูกส่งออกไปจากตัวเข้ารหัสทางขา D output

R_S , C_{TC} , R_{TC} ขา (11,12,13)

ขาเหล่านี้จะเป็นส่วนของการกำเนิดสัญญาณความถี่ของตัวเข้ารหัสดังรูป



รูปที่ 5.4 แสดงข้อมูลในการกำเนิดความถี่ในการเข้ารหัส

ถ้าเราเลือกที่จะใช้สัญญาณจากภายนอกแทนการกำเนิดจากภายใน ก็ทำการต่อเข้าที่ขา R_S และที่ R_{TC} และ C_{TC} ก็จะปล่อยให้เป็น open circuit

ค่าของ R_S ที่เลือกควรจะมีค่าเท่ากับ 2 เท่าของ R_{TC} ซึ่งในช่วงที่เลือกนี้กระแสที่ผ่าน R_S จะมีค่าใช้ได้เมื่อเทียบกับกระแสที่ผ่าน R_{TC} หรือค่าสูงสุดของ R_S ที่สามารถใช้ได้คือ $R_S \times 5PF$ (ค่าความชุขาเข้า) จะน้อยกว่าเมื่อเทียบกับค่า $R_{TC} \times C_{TC}$

TE

Transmit enable (ขา14)

ขานี้จะทำงานเมื่อมีสถานะเป็น low ซึ่งจะมีอุปกรณ์ภายในทำการ pull-up ให้เป็นสถานะ high เมื่ออยู่ในสถานะปกติ

Dout

Data output (ขา 15)

เป็นขาที่ output ตัวเข้ารหัสซึ่งจะเป็นแบบอนุกรม

Vss

Negative power supply(ขา 8)

ขานี้โดยปกติจะต่อลง GND

Vdd

Positive Power Supply (ขา 16)

ส่วนใหญ่จะเป็นขาบวกของเครื่องจ่ายไฟ

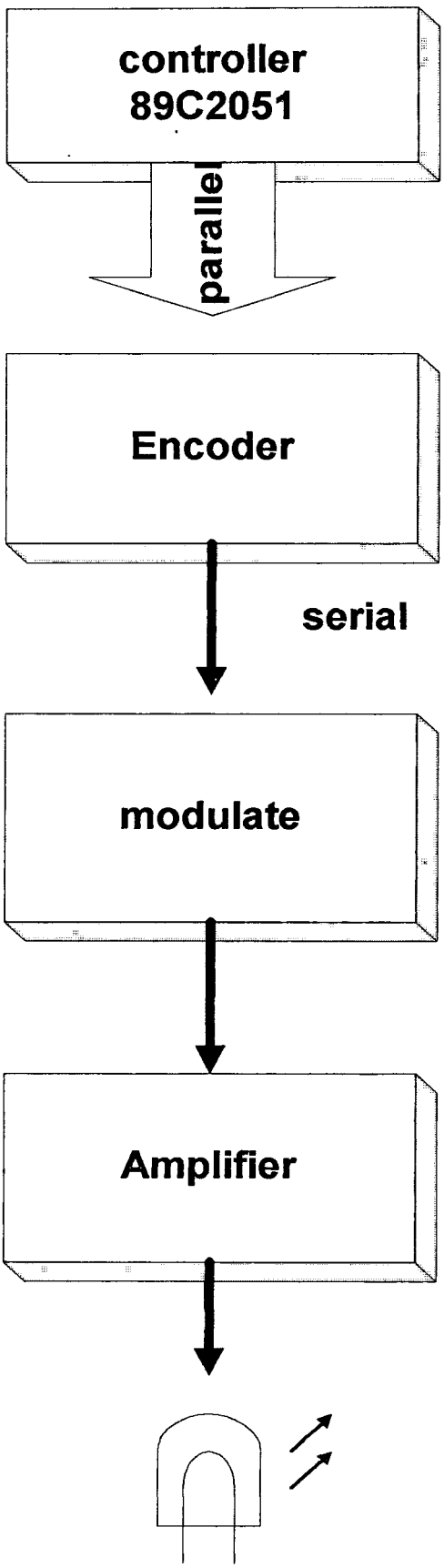
5.2.2 หลักการทำงานของ MC145026

การทำงานของวงจรเราสามารถแสดงได้อย่างงาย ๆ ตามบล็อคไดอะแกรมตามรูปที่ 5.5 และในการต่อวงจรที่ใช้งานจริงเป็นดังรูปวงจรที่ 5.6 เราจะเห็นว่าอุปกรณ์ MC 145026 ที่เราเลือกใช้เป็นตัวเข้ารหัสจากสัญญาณข้อมูลขนาน ให้กลายเป็นสัญญาณข้อมูลแบบอนุกรมซึ่งจะมีรูปแบบการเข้ารหัสเป็นสัญญาณเฉพาะแบบที่อธิบายไปในตอนที่แล้ว โดยเราจะเลือกสัญญาณความถี่ในการส่งเป็น 4 KHz ใช้วงจรกำเนิดสัญญาณความถี่จาก RC Oscillator ที่ต่อเข้ากับขา R_3 (ขา11) R_{TC} (ขา12) และ C_{TC} (ขา 13) ตามลำดับโดยเราสามารถเลือกความถี่ได้โดยการต่อความต้านทานและตัวเก็บประจุแต่ละค่าดังต่อไปนี้

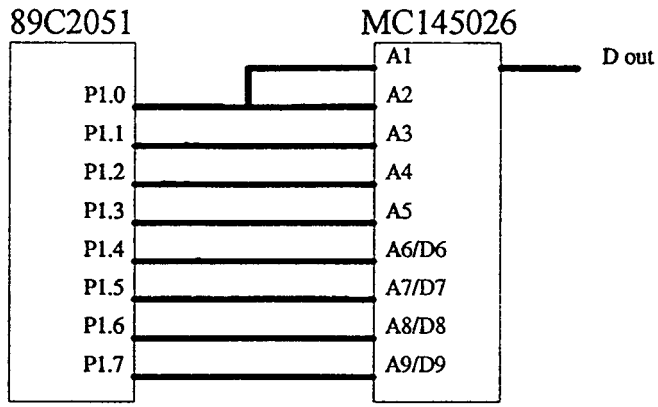
$$R_S = 220\ K\Omega$$

$$C_{TC} = 1000\ PF$$

$$R_T = 100\ K\Omega$$



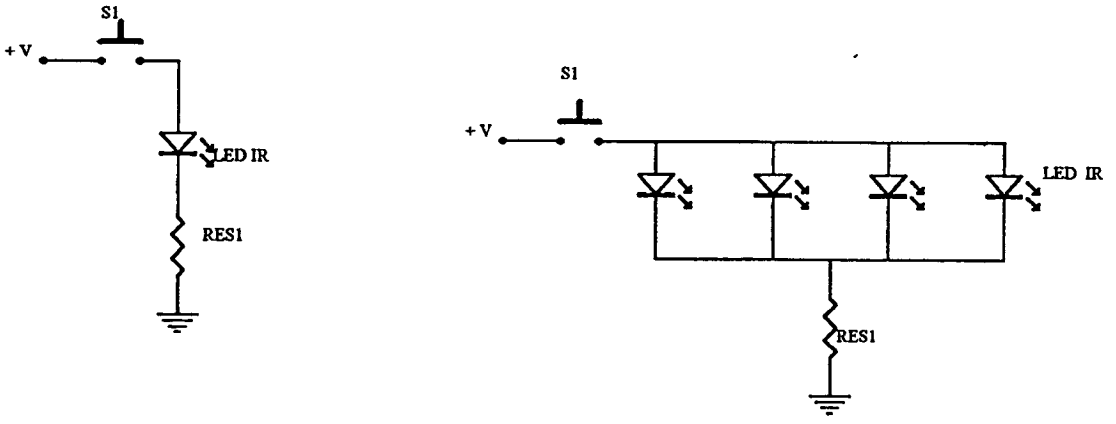
รูปที่ 5.5 แสดงบล็อกไดอะแกรมในภาคส่งอินฟาเรด



รูปที่ 5.6 แสดงการเชื่อมต่อ MC145026 เข้ากับ 89C2051

ซึ่งทำให้ได้ค่าสัญญาณความถี่ในการส่งข้อมูลออกเป็น 4 KHZ และเงื่อนไขในการส่งสัญญาณออกอีกข้อหนึ่งคือการตั้งค่าขา TE (ขา 14) ให้เป็นศูนย์ (low) ขา TE นี้จะทำตัวมีลักษณะคล้ายกับเป็นสวิทช์ในการส่งข้อมูลหรือไม่ส่งข้อมูลโดยจะมีสภาวะ ON (ส่ง) เมื่อตั้งค่าให้เป็นสูนย์และสภาวะ OFF (ไม่ส่งข้อมูล) เมื่อตั้งค่าให้เป็นหนึ่ง หลังจากนั้นตัว MC 145026 ก็จะทำการเข้ารหัสออกเป็นสัญญาณแบบอนุกรมที่ขา 15 โดยสัญญาณที่เราได้ออกมานี้สามารถที่จะไปต่อกับวงจรขับ LED อินฟราเรดได้เลย แต่อาจจะไปได้ไม่ค่อยไกลนัก ดังนั้นเราจึงต้องทำการมอดูเลทสัญญาณเอาท์พุทที่ออกมาด้วยค่าความถี่สูงก่อน ซึ่งในตอนนี้เราจะใช้การกำเนิดสัญญาณความถี่จากวงจร RC Oscillator อีกเช่นกัน และจะใช้ตัวแนนเกทในการมอดูเลท ความถี่ทั้งสองเข้าด้วยกันในวงจรการมอดูเลตนี้เราจะใช้ตัว MC 14011UB ที่เป็นแนนเกทแบบซีมอสในการมอดูเลท เมื่อเราได้ค่าสัญญาณที่ทำให้เราสามารถส่งสัญญาณออกไปได้ไกลๆ แล้ว ก็จะถึงกระบวนการแปลงข้อมูลทางไฟฟ้าให้กลายเป็นข้อมูลทางแสงต่อไป เลือกใช้การส่งแบบอินฟราเรดที่มีภาคขับสัญญาณใช้ทรานซิสเตอร์ 2N 2222 ในการทำการขยายสัญญาณและเลือกใช้ LED แบบ MLED

81 ซึ่งเป็น LED กำเนิดสัญญาณแสงในย่านอินฟราเรดและเลือกต่อสัญญาณไฟเลี้ยงให้กับ LED เป็น 12 โวลต์ โดยในตอนนี้เราสารถเลือกที่จะสร้างความเข้มแสงของอินฟราเรดให้มากขึ้นตามรูปวงจรดังต่อไปนี้



รูปที่ 5.7 แสดงการต่อวงจร LED ให้มีความเข้มเพิ่มขึ้น

จากวงจรในข้างต้นเราก็จะได้ค่าความเข้มแสงที่ออกมาจาก LED มีค่าความเข้มเพิ่มขึ้น และค่าในการกำเนิดสัญญาณความถี่คลื่นพาห์นั้นกำหนดได้ดังนี้คือ

R_1 (ต่อกับขา 2) = 220 KΩ

R_2 (ต่อกับขา 3) = 730 KΩ

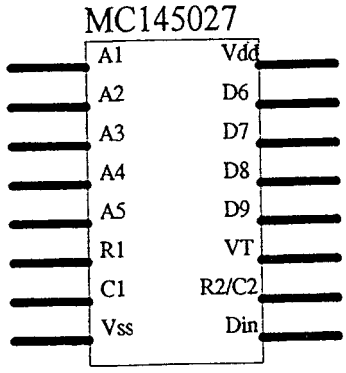
C_1 (ต่อกับขา 4) = 10 PF

จะได้ความถี่ประมาณ 50KΩ

5.3 วงภาครับอินฟราเรด

5.3.1 หลักการทำงานของ MC 145027

ตัวถอดรหัส MC 145027 จะได้รับสัญญาณอนุกรมซึ่งจะมีสัญญาณรหัสของ address 5 บิต ทำให้สามารถได้การเข้ารหัสทั้งหมดเป็น 243 address ถ้าใช้เป็นแบบ 2 สถานะก็จะทำการสร้าง address ออกมาได้ 32 address สัญญาณอนุกรมที่อ้างถึงนี้จะหมายถึง 4 บิตของข้อมูลแบบ 2 สถานะ สำหรับขา Valid transmission จะเป็นสถานะ High เมื่อ พบว่า 2 เงื่อนไขคือ 1) 2 address ที่ได้รับถูกต้องและตรงกับ address ที่ส่งออกมา 2) คือบิตของข้อมูลอีก 4 บิต ที่ส่งออกมาเหมือนกับการรับข้อมูลครั้งแรกเพื่อเป็นการป้องกันการคลาดเคลื่อนของข้อมูล ซึ่งจะมีรายละเอียดดังต่อไปนี้



รูปที่ 5.8 แสดงตัวถังของ MC145027

ตัวถอดรหัสจะได้รับสัญญาณข้อมูลอนุกรมจากตัวเข้ารหัส ถ้าเป็นข้อมูลที่ถูกต้อง การส่งข้อมูลจะประกอบไปด้วย 2 word โดยเราจะสมมุติให้ 5 บิตแรกเป็น สัญญาณ address ถ้าได้รับสัญญาณ address ตรงกับข้อมูลที่ส่งออกมา ก็จะทำการรับค่าอีก 4 บิต ต่อไปซึ่งเป็นบิตของข้อมูล แต่จะไม่มีการส่งผ่านไปยัง data latch output การเข้ารหัสครั้งที่ 2 จะได้รับและบิต address ก็ยังซ้ำกันเหมือนเดิม บิตข้อมูลก็จะทำการตรวจลอบอีกครั้งหนึ่ง ถ้า 2 word ของข้อมูลยังเหมือนกัน ข้อมูลก็จะถูกส่งออกไปยัง data latch output และไปทำให้ขา VT เป็น high จนกว่าจะมีข้อมูลใหม่เข้ามาแทนที่ จนกว่า จะมีการผิดพลาดของข้อมูลเกิดขึ้นก็จะทำให้ขา VT ตกลงเป็น low อีกครั้งหนึ่ง

รายละเอียดขาของ MC 145027

A1 - A5 , A1 - A9

Address input (ขา1 - 5)

จะเป็นขา Address input สถานะของขาเหล่านี้ จะต้องเข้ากันกับสถานะขาเข้าของตัวเข้ารหัส จึงจะทำให้ขา VT เกิดสภาวะ high ขึ้น

D6 - D9

Data Output (ขา 15 , 14 , 13 , 12)

ขาเหล่านี้จะแสดงถึงรายละเอียดแบบ 2 สถานะ ที่ส่งมาเข้ารหัสในขา A6 / D6 ถึง A9 / D9

Din

Data in (ขา 9)

ขานี้จะเป็นสัญญาณข้อมูลแบบอนุกรมที่เข้าไปยังตัวถอดรหัส ค่า input voltage ที่ใช้จะเป็นค่าที่ลอจิกของ CMOS ที่สภาวะ high

R1 , C1

ตัวต้านทาน 1 , ตัวเก็บประจุ 1 (ขา 6,7)

ทั้งสองตัวนี้จะใช้ในการสร้างสัญญาณลูกคลื่นกว้างหรือแคบแล้วแต่การใช้ค่า R1 และ C1 ซึ่งจะได้ค่าคงที่ของเวลาเป็น R1× C1 และคาบนาฬิกา การเข้ารหัสเป็น

$$R_1C_1 = 3.95R_{TC}C_{TC}$$

R2 /C2

ตัวต้านทาน 2 / ตัวเก็บประจุ 2 (ขา 10)

ตัวต้านทานและตัวเก็บประจุทั้งสองนี้จะใช้ในการตรวจจับทั้งการสิ้นสุดของ word ที่ได้รับและการสิ้นสุดของการส่ง ค่าคงที่ของเวลาเป็น R2 ×C2

$$R_2C_2 = 77\ R_{TC}C_{TC}$$

VT

Valid transmission Output (ขา 11)

ขานี้จะเป็นสภาวะ High ก็ต่อเมื่อ word ข้อมูลทั้งสองที่ต่อกันมาตรงตามเงื่อนไขต่อไปนี้

1. ได้รับสัญญาณ address ที่ 2 word ติดต่อกันและคล้องจองกันกับสัญญาณ address ที่ส่งออกมา
2. ได้รับสัญญาณข้อมูลทั้งสองติดต่อกันและตรงกัน 2 word

ขา VT จะเป็นสภาวะ High จนกระทั่งเกิดการไม่คล้องจองกัน หรือไม่มีสัญญาณข้อมูลขาเข้า 4 บิต

Vss

Negative Power Supply (ขา 8)

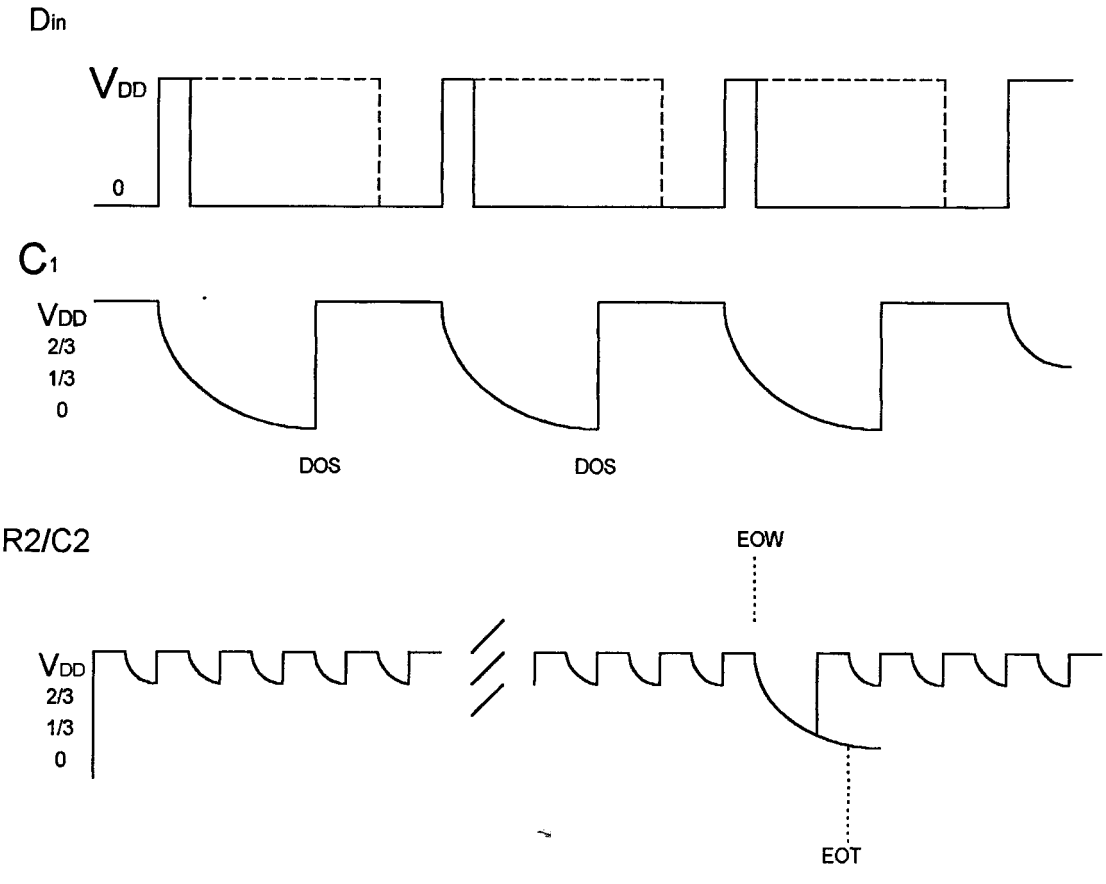
เป็นขาที่ต่อลง GND

V_{DD}

Positive Power Supply (ขา 16)

เป็นขาบวกเครื่องจ่ายไฟ

สำหรับคาบเวลาของ MC 145027 สามารถตรวจสัญญาณคลื่นได้จากขา C1 และ R2 / C2 ที่จะไปเปรียบเทียบกับสัญญาณข้อมูลขาเข้าของ Din การคายประจุของ C1 จะลดลง 1/3 ของ V_{DD} ก่อนที่จะ Reset V_{DD} อีกครั้งหนึ่งซึ่งขอบของข้อมู,ที่ได้จากจะต้องเป็น " 1 " หรือ " 0 " ตรงการลดลงของจุดที่จะ reset นี้



รูปที่ 5.9 แสดงช่วงเวลาของ MC145027

ส่วนสัญญาณนาฬิกาอื่นๆ ทีทำการตรวจบน MC145027 บน R_2/C_2 การลดลงของ R-C คือ หมายถึงว่าเกิดการส่งของข้อมูลจะมีเพียงการส่งระหว่าง WORD เท่านั้นที่ R_2/C_2 จะลดลง เรียกว่าเป็น end-of-word (EOW) ขอบของการลดลงนี้จะต้องอยู่ในช่วง 2/3 ของ V_{DD} และในการส่งผ่านข้อมูล (EOT) end-of-transmission ของของการส่งที่ปรากฏอยู่ในช่วง 1/3 ของ V_{DD} เมื่อเราสังเกตุสัญญาณลูกคลื่นก็จะเห็นว่าตัว R_2/C_2 จะต้องลดลงอยู่ในช่วง 2/3 และ 1/3 ของระดับ V_{DD} ดังรูปที่ 5.9

5.3.2 หลักการทำงานของวงจร

รูปที่ 5.10 เป็น บล็อคไดอะแกรมและรูปวงจรที่ 5.11 ของภาครับสัญญาณเริ่มจากโฟโตไดโอดที่ทำหน้าที่เป็นตัวรับแสง แล้วทำการแปลงสัญญาณแสงออกมาเป็นสัญญาณทางไฟฟ้า ซึ่งในตัวรับสัญญาณแสงนี้เองจะเป็นตัวที่ทำการขยายสัญญาณที่รับเข้ามาได้ให้มีค่ามากขึ้น พร้อม ๆ กับทำการตัดสัญญาณความถี่สูงออกไป นั่นก็หมายถึงการทำงานในการส่วนของการดีมอดูเลทเองดังนั้นหมายความว่าในการใช้ตัวรับอินฟราเรดของเรา 1 ตัว สามารถที่จะรวมวงจรในภาคขยายสัญญาณและได้ทำการรวมวงจรวงจรดีมอดูเลตเข้าไว้ในตัวเดียวกัน

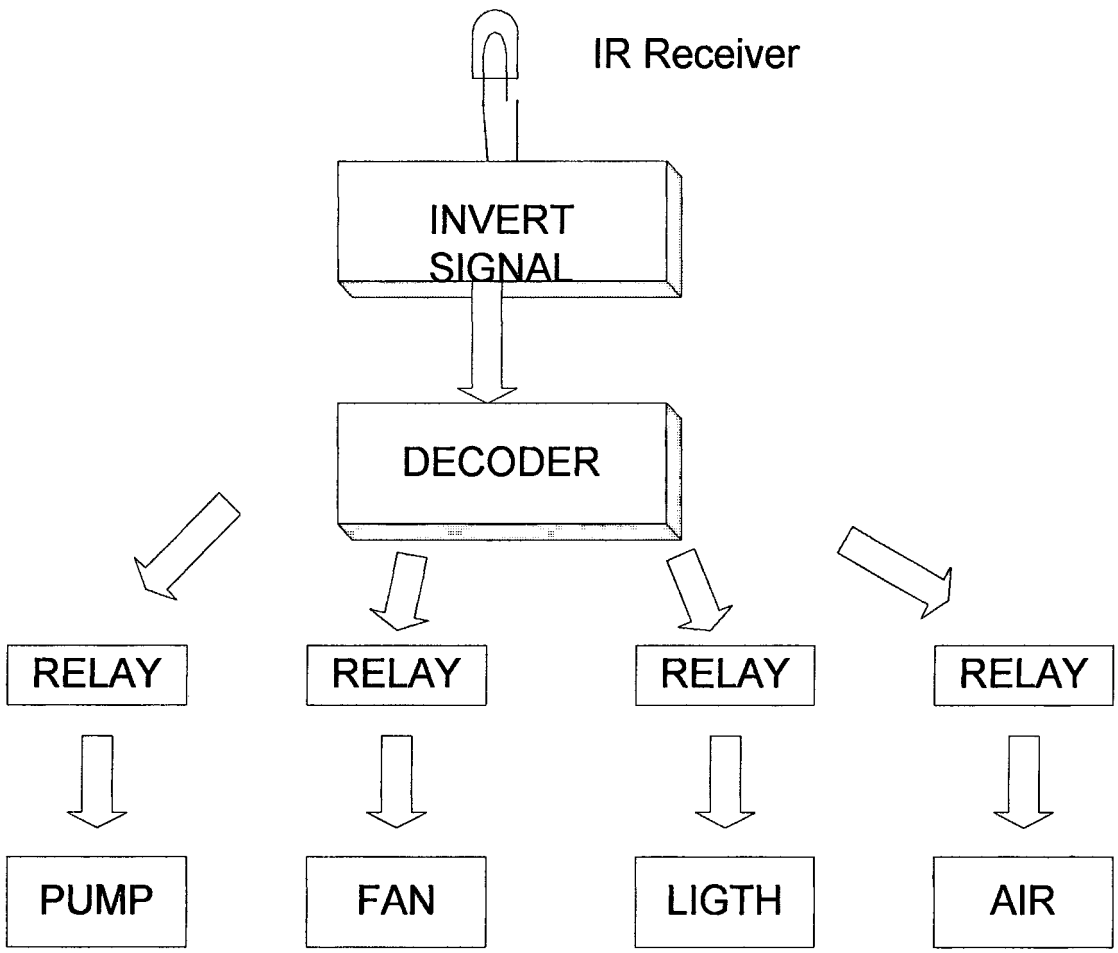
เมื่อเราได้สัญญาณทางไฟฟ้าออกมาแล้วก็มาถึงการกลับสัญญาณโดยอาศัยตัว MC 14001 UB ซึ่งเป็นตัวอินเวอร์สเตอร์โดยเราเอาสัญญาณที่ได้ออกมาจากตัวรับอินฟราเรดแล้วนำมาต่อเข้าที่ขา 1 ของัว MC 14001 UB และนำเอาเอาท์พุทที่ออกมจากสัญญาณขาสองออกมาใช้ ซึ่งสัญญาณที่ได้ออกมาจากขานี่จะมีลักษณะของเวิร์ดข้อมูลเหมือนกับที่เราทำการส่งออกมา เมื่อเรานำสัญญาณที่ผ่านการอินเวอร์สแล้วนี้ไปเข้ากับตัว MC 145027 ที่ขา Din 9 โดยในการรับสัญญาณขาเข้านี้เราจะต้องเลือกค่าความถี่ที่เราส่งออกมาคือ 4 KHZ เลือกใช้ค่าความถี่ที่ขา R1 และ C1 (ขา 6 และ 7 ตามลำดับ) และค่าความถี่ที่เกิดจาก R_2 / C_2 (ขา 10) โดยในการเลือกใช้ ให้สามารถเป็นความถี่ขนาด 4KHZ จะต้องเลือกค่าความต้านทานและตัวเก็บประจุดังนี้คือ

R1 = 390 KΩ

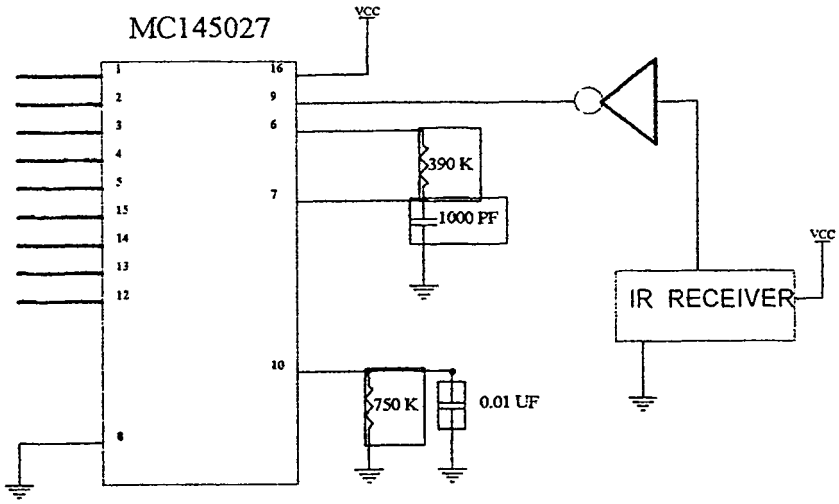
C1 = 1000 PF

R2 = 730 KΩ

C2 = 0.01 μF



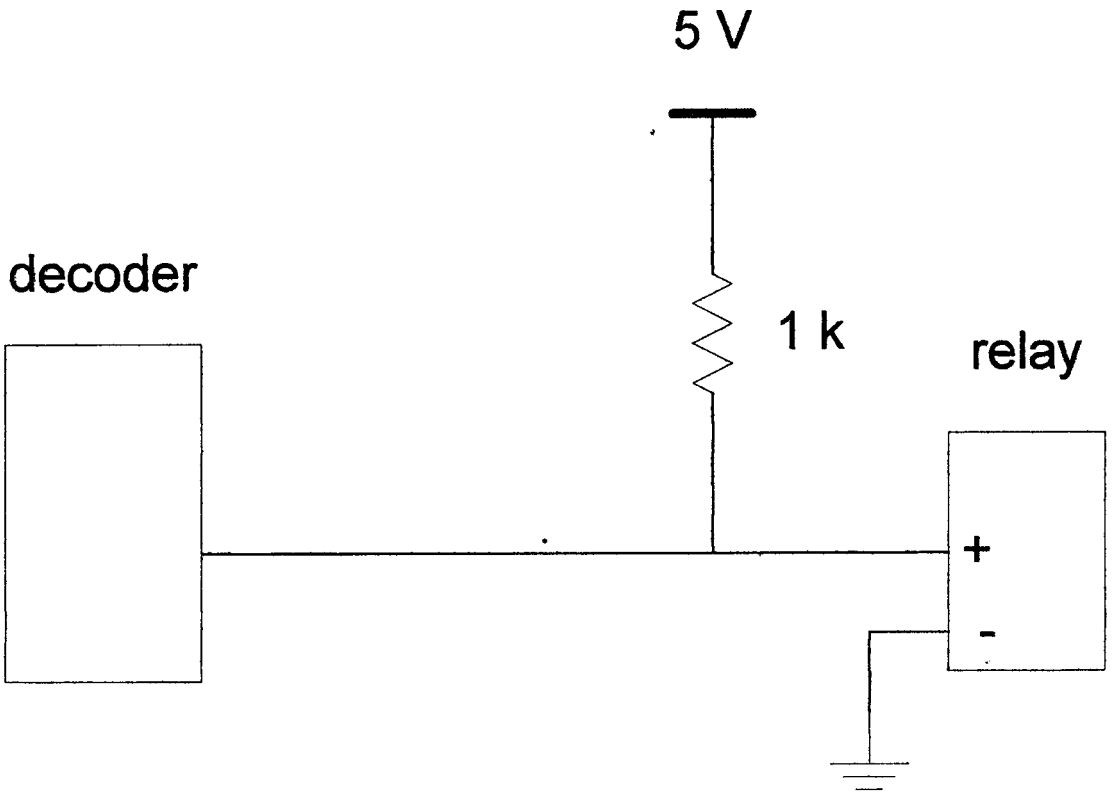
รูปที่ 5.10 แสดงบล็อคไดอะแกรมภาครับอินฟาเรด



รูปที่ 5.11 วงจรในภาครับอินฟาเรด

สำหรับในการเลือกสัญญาณข้อมูลที่มาจากภาคส่งจะมีทั้งหมด 9 บิต ทำให้สามารถเลือกข้อมูลโดยแบ่งตามจำนวนบิตที่ได้คือ A1-A5 จะเป็นบิตของ addressที่เราเลือกไว้ และบิต D6-D9 เป็นบิตที่เราใช้เป็นข้อมูลในการสั่งงาน ดังนั้นจึงเลือกใช้ประโยชน์ในการเลือกข้อมูลได้ ทั้งนี้ในตัวถอดรหัสจะทำการตรวจสอบความถูกต้องของตัวบิต address ซึ่งใน 5 บิตที่เราส่งออกมานั้นจะต้องทำการตั้งค่าให้ตรงกันทั้งสองตัวระหว่างตัวเข้ารหัสและตัวถอดรหัส เมื่อตรวจสอบถึงการส่งข้อมูลที่ถูกต้องของ address แต่ถ้าเกิดการไม่ตรงกันของข้อมูลก็จะเกิดการไม่ถอดรหัสขึ้นโดยเราสามารถที่จะดูได้จากสัญญาณในขา VT (ขา 11) จะเป็นขาที่แสดงถึงสภาวะการตรงกันของข้อมูลที่ตรวจสอบว่าเมื่อถูกต้องแล้วจะเปลี่ยนสถานะจาก low เป็น High ทันที

เมื่อตัวถอดรหัสได้ทำการตรวจสอบว่าค่า address ที่ได้ออกมาตรงกับค่าที่ถูกส่งออกมาแล้ว ก็จะทำการถอดรหัสสัญญาณข้อมูลออกมา 4 บิต คือ บิตที่ D6-D9 (ขา 12-15) โดยที่เราจะนำขาทั้ง 4 บิตข้อมูลนี้ไปใช้ในการสั่งเปิด - ปิด อุปกรณ์ไฟฟ้าได้ 4 อุปกรณ์ ผ่านการสั่งงานของรีเลย์ที่เลือกใช้เป็นแบบ โซลิดสเตทรีเลย์ที่ทำงานเมื่อมีการจ่ายไฟที่ขา อินพุท 5 V จะไปทำการเปิด-ปิด ไฟที่ 220 โวลต์ได้ ซึ่งก็หมายถึงอุปกรณ์ไฟฟ้าเอง วงจรที่ใช้ในการต่อวงจรตัวถอดรหัสเพื่อไปใช้ในการ trig สัญญาณ ให้ตัวโซลิดสเตทรีเลย์เป็นดังนี้คือ



รูปที่ 5.12 การเชื่อมต่อกับรีเลย์

บทที่ 6

ผลการทดลอง

6.1 ภาคส่งอินฟราเรด

การใช้อุปกรณ์ตัวเข้ารหัสนี้ได้กล่าวไว้ในบทก่อน ๆ ว่าเป็นการเข้ารหัสแบบข้อมูลแบบขนาน แล้วทำการแปลงข้อมูลให้เป็นแบบอนุกรม โดยสายทั้ง 9 ของสัญญาณ input จะแบ่งออกเป็น 2 ชนิดคือ

1) address bit คือ A1-A5 หมายถึงสายสัญญาณข้อมูลที่ทำใช้ในการตรวจสอบข้อมูลหรือหมายถึงการเลือกเงื่อนไขในการส่งตามที่เราต้องการ ในที่นี้เราจะใช้ address bit ในการบอกถึงรหัสของห้องต่าง ๆ ที่เราจะทำการเปิด-ปิดไฟดังตารางดังต่อไปนี้

A1	A2	A3	A4	A5	หมายเลขห้อง
0	0	0	0	0	1
0	0	0	0	1	2
0	0	0	1	0	3
0	0	0	1	1	4
0	0	1	0	0	5
0	0	1	0	1	6
0	0	1	1	0	7
0	0	1	1	1	8
1	1	0	0	0	9
1	1	0	0	1	10
1	1	0	1	0	11
1	1	0	1	1	12
1	1	1	0	0	13
1	1	1	0	1	14
1	1	1	1	0	15
1	1	1	1	1	16

ตารางที่ 6.1 การเข้ารหัสเพื่อเลือกห้อง

หมายเหตุ สำหรับในกรณีนี้เราจะเลือกใช้ในแบบ 2 สถานะ (binary code)

คือในการเลือกสัญญาณการเข้ารหัสดังตารางข้างต้นแล้วหมายความว่าเราสามารถเลือกที่จะเข้ารหัสได้ทั้งหมด 16 รหัส 16 ห้อง ที่มีรหัสต่าง ๆ กันไป แต่ในการเชื่อมต่อเข้ากับตัวไมโครคอนโทรลเลอร์ข้อมูลที่ออกมาจากตัว 89C2051 จะมีสายสัญญาณออกมาทั้งหมด 8 เส้น ดังนั้นเราจึงต่อสัญญาณจาก P1.0 ไปเข้ากับขาA1 และ A2 ของ mc 145026 จากข้อดีของการเลือกใช้ mc 145026 นี้เอง เมื่อเราส่งสัญญาณทั้ง 9 บิต ออกไป ตัว address bit ทั้ง 5 นี้จะเป็นตัวเลือกว่ารหัสที่เราส่งออกไปนั้นหมายถึงการใช้อุปกรณ์ไฟฟ้าในห้องใด ภายในห้องแต่ละห้องนั้นเราก็จะทำการติดตั้งภาครับอินฟราเรดไว้และกำหนด address code ให้ตรงกับที่ส่งออกมาแตกต่างกันไปตามแต่จำนวนห้องคือเป็นการเลือกจำนวนห้องเอง

2) Data bit หรือ บิตข้อมูล จะมีทั้งหมด 4 เส้นคือ A6/D6 - A9/D9 เป็นบิตที่ใช้ในการเปิด-ปิด อุปกรณ์ไฟฟ้าคือเป็นบิตที่เราจะใช้เป็นสายสัญญาณในการไปเปิด-ปิดอุปกรณ์ไฟฟ้าเหมือนสวิตช์ โดยผ่านทางหน้าจอคอมพิวเตอร์สายทั้ง 4 เส้นเราจะนำไปต่อเข้ากับอุปกรณ์ไฟฟ้า 4 ชิ้นตามสายสัญญาณที่ออกมา แสดงได้ดังตารางการส่งอุปกรณ์ข้างล่างนี้

ตารางที่ 6.2 ความหมายในการควบคุมอุปกรณ์

A6/D6	A7/D7	A8/D8	A9/D9
0	0	0	0
0	0	0	1
0	0	1	0
0	0	1	1
0	1	0	0
0	1	1	0
0	1	1	1
1	0	0	0
1	0	0	1
1	0	1	0
1	0	1	1

1	1	0	0
1	1	0	1
1	1	1	0
.1	1	1	1

"1" หมายถึง อุปกรณ์ปิด

"0" หมายถึง อุปกรณ์เปิด

สายสัญญาณเส้นที่ D6 จะควบคุมอุปกรณ์ไฟฟ้าชนิดที่ 1
สายสัญญาณเส้นที่ D7 จะควบคุมอุปกรณ์ไฟฟ้าชนิดที่ 2
สายสัญญาณเส้นที่ D8 จะควบคุมอุปกรณ์ไฟฟ้าชนิดที่ 3
สายสัญญาณเส้นที่ D9 จะควบคุมอุปกรณ์ไฟฟ้าชนิดที่ 4

ดังนั้นถ้าเรานำข้อมูลทั้งหมดที่ส่งออกมาให้ครบ 9 บิตจากตัวเข้ารหัสจะหมายถึงการทำงานต่าง ๆ เช่น ตัวอย่างต่อไปนี้

กรณีที่ 1

ต้องการให้อุปกรณ์ไฟฟ้าตัวที่ 1 และตัวที่ 2 ในห้องที่ 1 ติดเราจะส่งสัญญาณข้อมูลออกไปได้

A1	A2	A3	A4	A5	D6	D7	D8	D9
0	0	0	0	0	1	1	0	0

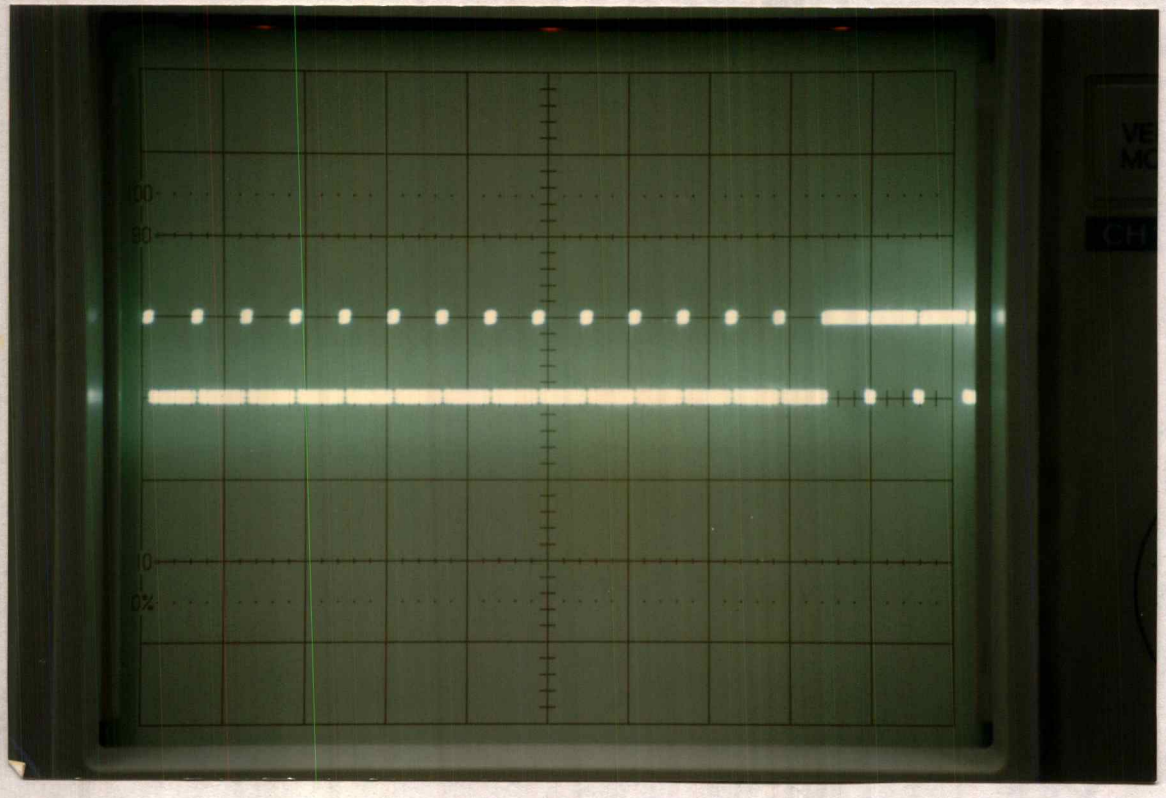
กรณีที่ 2

ต้องการให้อุปกรณ์ไฟฟ้าตัวที่ 1 ติดอย่างเดียวตัวอื่นดับหมด ภายในห้องที่ 2

A1	A2	A3	A4	A5	D6	D7	D8	D9
0	0	0	0	1	1	0	0	0

สำหรับในโหมดการตั้งเวลานั้นต้องทำก่อนที่เราจะเลือกห้องและเลือกเปิด-ปิดอุปกรณ์ไฟฟ้า โดยจะต้องทำการตั้งค่า 2 ครั้ง ครั้งแรกคือ เราจะต้องตั้งวันและเวลาในการเดินของไมโครคอนโทรลเลอร์ก่อนซึ่งจะเหมือนกับการตั้งเวลาแบบทั่ว ๆ ไป และสามารถที่ตั้งเวลาเปิด-ปิดล่วงหน้าได้ด้วย เมื่อเราทำการตั้งค่าวันและเวลาที่เป็นอยู่ในปัจจุบันให้กับไมโครคอนโทรลเลอร์ได้แล้วต่อไปก็จะเป็นการเลือกวันและเวลาที่เราต้องการที่จะเปิด-ปิด อุปกรณ์ไฟฟ้าตัวใด วันที่เท่าไร และเวลาอะไร ดังที่กล่าวมาแล้วในข้างต้น

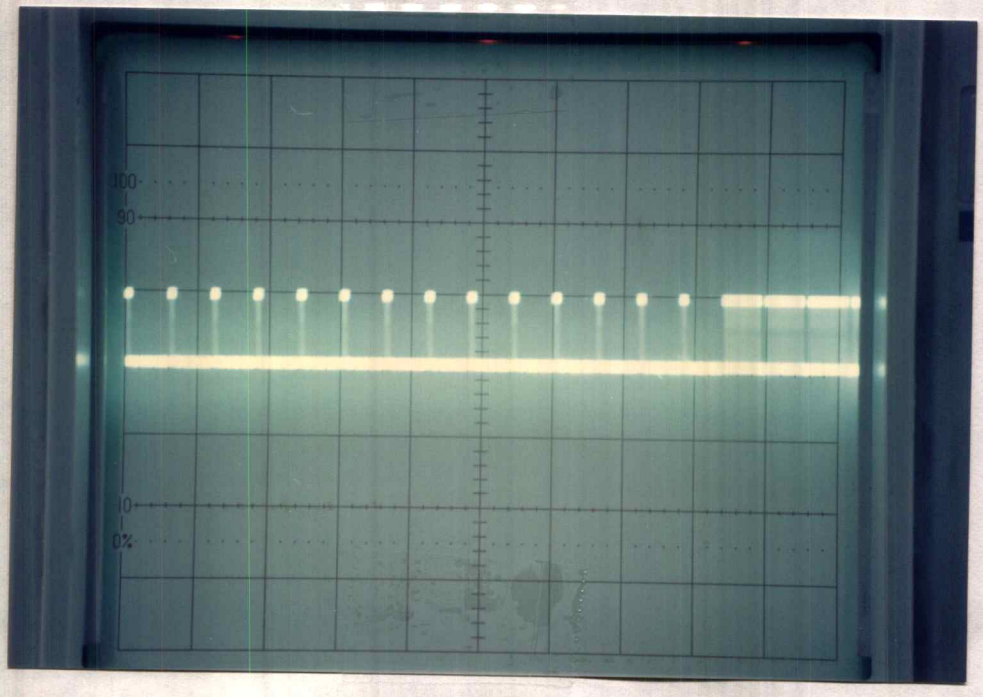
สัญญาณที่เราทำการส่งออกไปจากตัวเข้ารหัสส่งออกไปจากตัวเข้ารหัสที่เป็นสัญญาณข้อมูลอนุกรมนั้นจะทำงานสัมพันธ์กับฐานวันและเวลาที่ตั้งไว้คือเมื่อถึงวันและเวลาที่เรากำหนดไว้แล้วตัวไมโครคอนโทรลเลอร์ก็จะส่งชุดข้อมูลออกมา 8 บิตผ่านตัวเข้ารหัสส่งออกมาโดยมีสัญญาณดังรูปต่อไปนี้



รูปที่ 6.1 สัญญาณที่ส่งออกมาจากตัวไมโครคอนโทรลเลอร์แล้วทำการเข้ารหัสที่ตัว MC145026

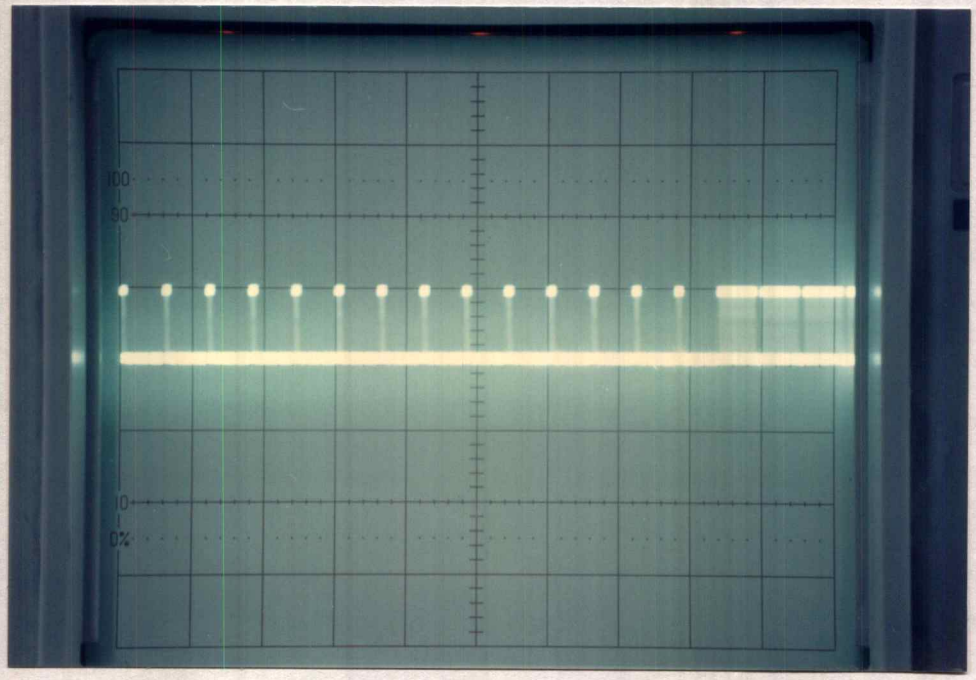
ในรูปข้างต้นจะหมายถึงสัญญาณที่ส่งออกมาจากตัวไมโครคอนโทรลเลอร์แล้วทำการเข้ารหัสออกมาเป็นสัญญาณแบบอนุกรม

สัญญาณที่เราทำการเข้ารหัสแล้วจะต้องมาทำการมอดดูเลตกับสัญญาณความถี่สูงดังรูป



รูปที่ 6.2 เกิดจากวงจรกำเนิดสัญญาณความถี่ RC จะให้ค่าความถี่ออกมาประมาณ 50 kHz

เมื่อเราเอาสัญญาณทั้งสองมามอดดูเลตกันแล้วจะได้สัญญาณข้อมูลซ้อนทับอยู่บนสัญญาณคลื่นพาห์เพื่ออาศัยคุณสมบัติการมอดดูเลตทำให้สามารถส่งผ่านคลื่นออกไปไกล ๆ ได้ สัญญาณที่ได้ออกมาจะแสดงดังรูป

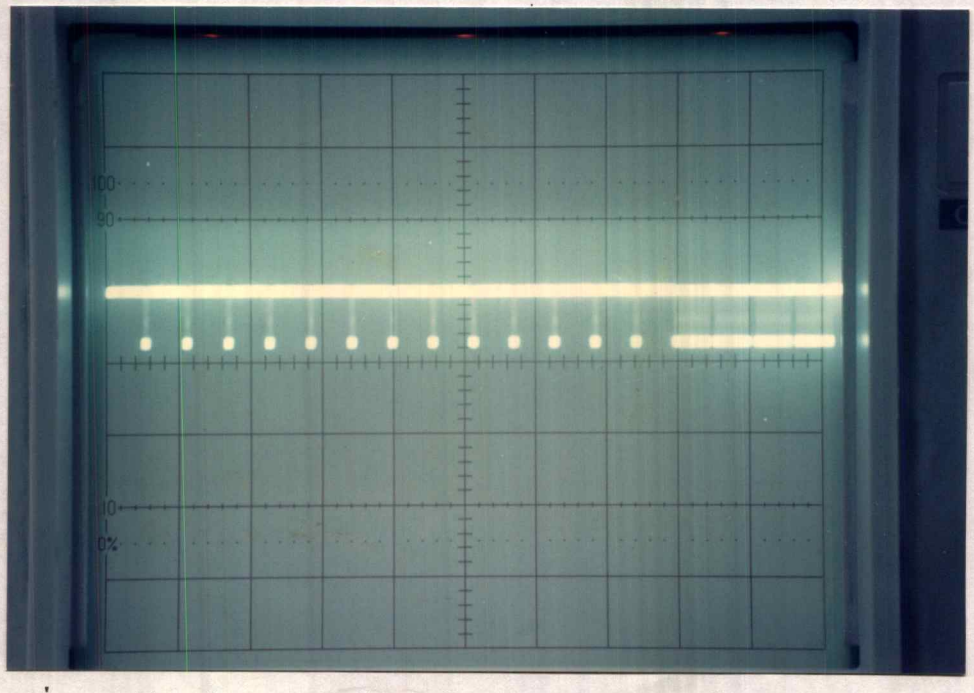


รูปที่ 6.3 แสดงถึงการสัญญาณที่มอดดูเลตแล้ว

สัญญาณที่ได้จะไปต่อเข้ากับภาคขยายสัญญาณเพื่อนำไปต่อให้กับ LED อินฟราเรด กราฟที่ได้ออกมากจะเหมือนการขยายโวลเตจให้เพิ่มขึ้นนั่นเอง

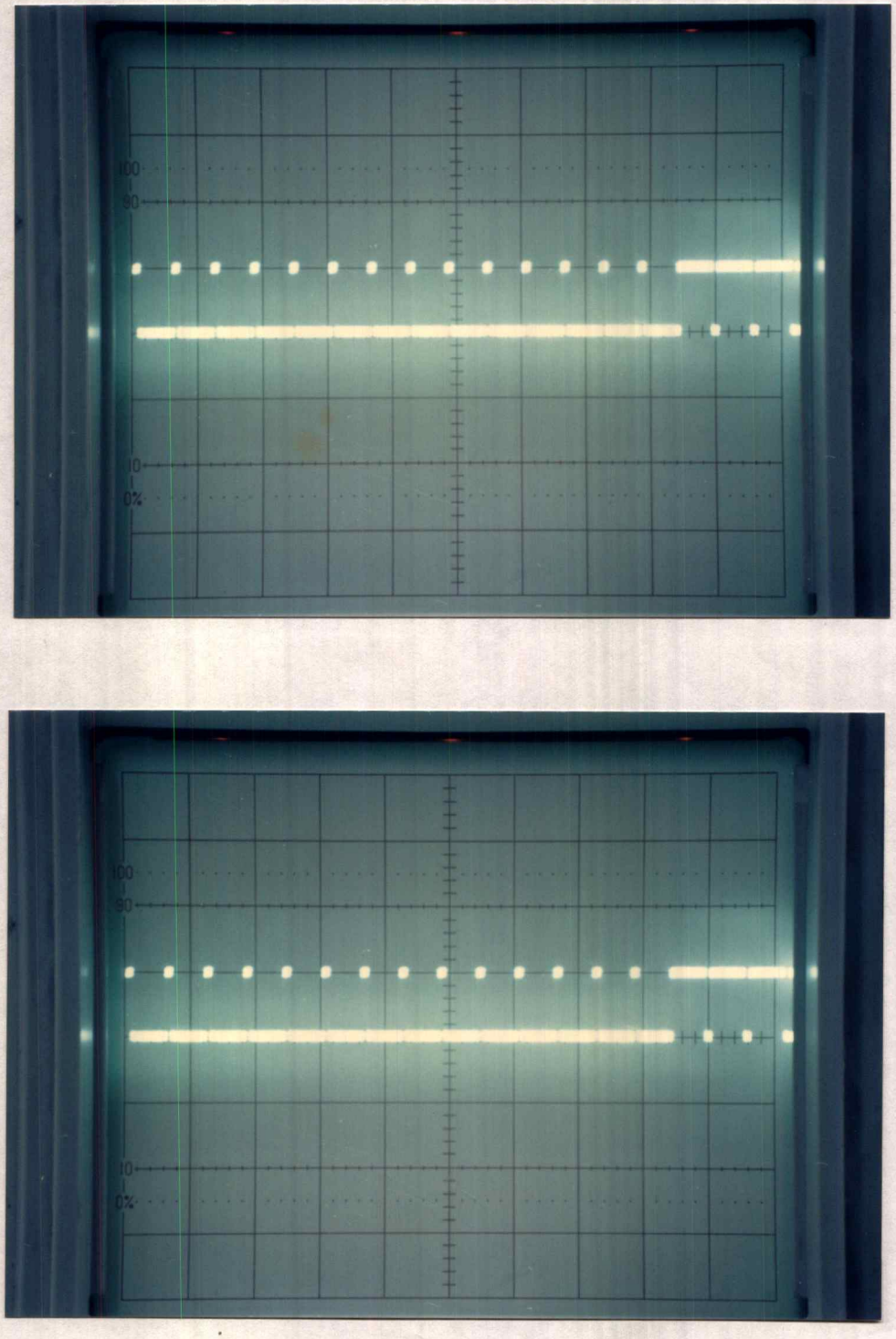
6.2 ภาคการรับอินฟราเรด

ภาคการส่งอินฟราเรดนี้สัญญาณที่รับได้โดยตัวรับอินฟราเรดจะตัดสัญญาณความถี่สูงออกทั้งหมดทำให้ได้สัญญาณเฉพาะในย่านความถี่ต่ำเท่านั้น (ความถี่ย่านอินฟราเรด) แต่สัญญาณที่ได้ออกมานี้จะกลับกันกับสัญญาณที่ส่งออกมาดังรูป



รูปที่ 6.4 เปรียบเทียบการกลับกันของสัญญาณจากขาออกของตัว MC145026 กับสัญญาณที่ออกมาจากตัวตรวจจับอินฟราเรด

ดังนั้นเราจึงต้องทำการกลับสัญญาณด้วย invertor เสียก่อนที่จะไปเข้าตัวถอดรหัสซึ่งจะได้สัญญาณเหมือนกับตอนแรกที่ออกมาจากตัวเข้ารหัส



รูปที่ 6.5 แสดงสัญญาณที่ออกมาจากตัวเข้ารหัสและสัญญาณที่ผ่านไปในตัวถอดรหัส

เมื่อตัวถอดรหัสได้ตรวจสอบสัญญาณข้อมูลทุกอย่างแล้วว่าเป็นไปตามค่าบิต address ที่ถูกต้อง ชุดข้อมูลอีก 4 บิตที่ตามมาก็จะถูก latch ส่งออกไปเป็นแบบขนาน โดยในแต่ละบิตนั้นเราจะนำไปต่อเข้ากับโซลิดสเตจรีเลย์เพื่อไปทำการเปิด-ปิดสัญญาณไฟ 220 โวลต์ หรือเป็นการเปิด-ปิด อุปกรณ์ไฟฟ้าตามคำสั่งนั่นเอง

โซลิดสเตจรีเลย์จะมีขาในการ trigg สัญญาณ 5 โวลต์ที่จะทำให้เกิดสภาวะเปิดที่ด้านไฟสูง (220 โวลต์) สัญญาณไฟ 5 โวลต์นี้เราจะได้มาจากการต่อวงจรให้มีความหมายว่าเมื่อคำสั่งที่ออกมาแล้วให้อุปกรณ์ไฟฟ้าติดนั้นจะเป็น สภาวะลอจิก LOW เมื่อเกิดสภาวะนี้ขึ้นแล้วก็จะทำให้ไฟ 5 โวลต์เกิดการไหลในวงจรทำให้เกิดการไปเปิดตัวอุปกรณ์ไฟฟ้า 220 โวลต์

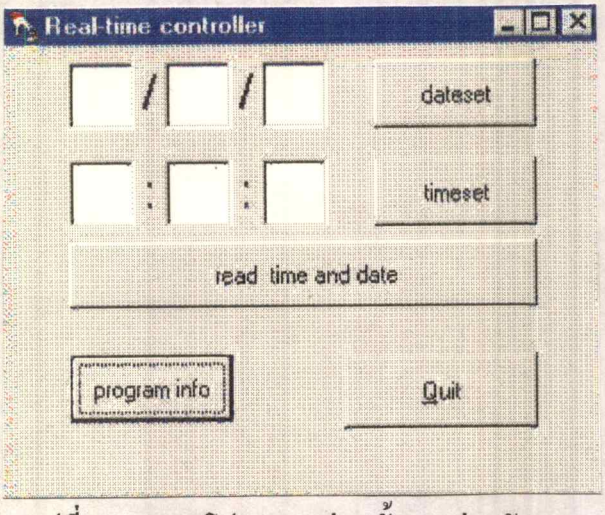
6.3 สรุปผลการทดลอง

ในการทำโครงงานพิเศษชุดนี้ได้ทำอุปกรณ์ Real-Time controller with IR Link. ที่สามารถไปควบคุมอุปกรณ์ไฟฟ้าตามบ้านเรือนหรือตัวอาคารโดยผ่านการควบคุมแบบเวลาจริง และมีการเชื่อมต่อแบบอินฟราเรด ทำให้อุปกรณ์ไฟฟ้าต่าง ๆ สามารถเปิด-ปิด อุปกรณ์ไฟฟ้าตามวันและเวลาที่เรากำหนดได้ ซึ่งในการทำโครงงานพิเศษชุดนี้ได้ใช้ตัวไมโครคอนโทรลเลอร์แบบชิพเดี่ยว 89C2051 ในการประมวณผลและในการให้กำเนิดสัญญาณนาฬิกา ต่อจากนั้นก็นำสัญญาณที่ได้ออกแบบขนานไปต่อเข้ากับคู่เข้ารหัสและถอดรหัสคือ MC145026 และ MC145027 ตามลำดับคู่ตัวเข้าและถอดรหัสนี้เหมาะสำหรับการให้ในการเชื่อมต่อแบบอินฟราเรด ดังนั้นตัวกลางที่เราเลือกใช้จึงเป็นการเชื่อมต่อแบบไร้สาย แทนการใช้สายไฟแบบทั่ว ๆ ไปอันเป็นการสิ้นเปลืองทรัพยากรธรรมชาติ สำหรับการเลือกใช้ภาษาในการควบคุมไมโครคอนโทรลเลอร์แบบชิพเดี่ยวนนี้ใช้ภาษาแอสแซมบลีแต่จะมีข้อยุ่งยากบางประการคือทำให้ผู้ใช้ไม่สามารถทำความเข้าใจได้ง่ายนักดังนั้นเราจึงทำการประยุกต์ด้วยภาษา VISUAL BASIC โดยแสดงออกมาในรูปแบบของ GUI คืออธิบายด้วยภาพที่สวยงามทำให้ผู้ใช้สามารถทำความเข้าใจได้ทันที

6.4 ผลการแสดงของโปรแกรม

โปรแกรมที่ได้จากการใช้ GUI ประกอบด้วยส่วนต่าง ๆ ดังนี้

1) ส่วนของการตั้งวันและเวลาให้กับบอร์ด Real-time ที่สร้างขึ้นในโครงงานพิเศษแสดงได้ดังนี้



รูปที่ 6.6 แสดงโปรแกรมส่วนตั้งและอ่านวันเวลา

ผู้ใช้สามารถกำหนดค่า วัน เดือน ปี ลงในช่อง □/□/□ และค่าวัน เดือน ปี ที่กำหนดขึ้นนั้นจะถูกส่งไปยังบอร์ด Real-time เมื่อมีการกดปุ่ม dateset และกำหนดค่าเวลา (ชั่วโมง นาที วินาที) ในช่อง □:□:□ และเมื่อมีการกดปุ่ม timeset ค่าเวลาจะถูกส่งไปยังบอร์ด Real-time

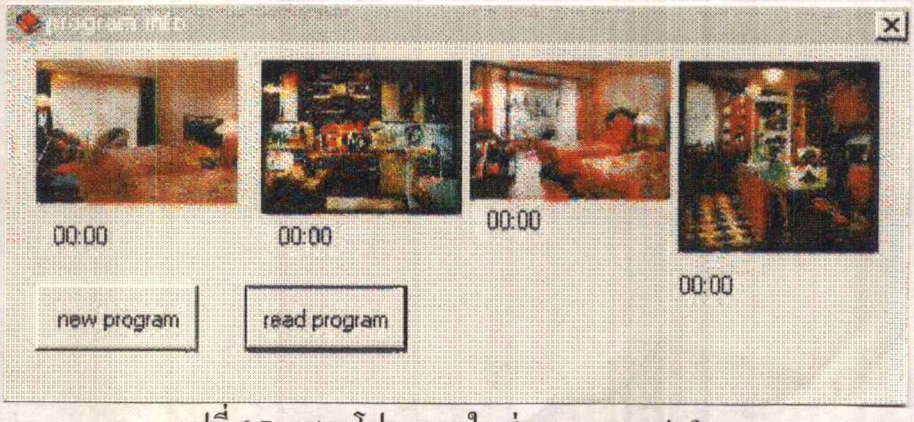
2) ส่วนการอ่านวันและเวลาจากบอร์ด Real-time ที่สร้างขึ้นในโครงงานพิเศษ

เมื่อกดปุ่ม read timeand date ค่าวันเดือนปีและค่าเวลา (ชั่วโมง นาที วินาที) จากบอร์ด Real-time จะถูกอ่านพร้อมกับแสดงค่าออกมาปรากฏที่ช่องของการแสดงวันและเวลา

เมื่อต้องการยกเลิกการใช้โปรแกรมให้กดปุ่ม Quit

3) ส่วนของการอ่านและการส่งโปรแกรมควบคุมเครื่องใช้ไฟฟ้า

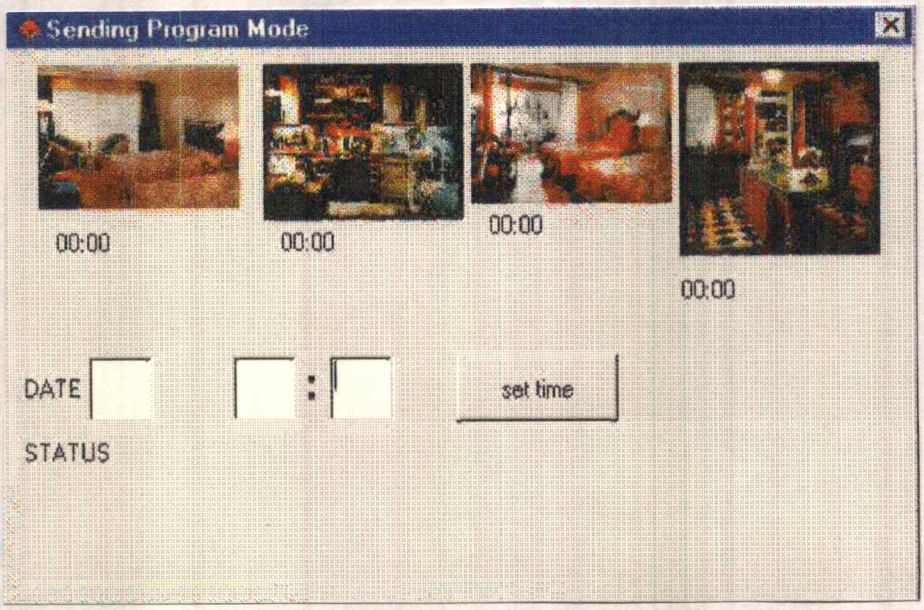
ส่วนนี้จะถูกแสดงเมื่อมีการกดปุ่ม program info ดังรูป



รูปที่ 6.7 แสดงโปรแกรมในส่วน program info

โปรแกรมจะแสดงภาพห้องต่าง ๆ ให้ผู้ใช้เลือกและให้ผู้ใช้เลือกที่จะทำการอ่านหรือส่งโปรแกรม โดยเลือกกดปุ่ม new program ถ้าต้องการส่งค่าการสั่งควบคุมเครื่องใช้ไฟฟ้าในวันและเวลาต่าง ๆ และกดปุ่ม read program ถ้าต้องการอ่านโปรแกรมคำสั่งครั้งล่าสุดที่บันทึกอยู่ในบอร์ด real-time

<u>กรณีกดปุ่ม new program</u>

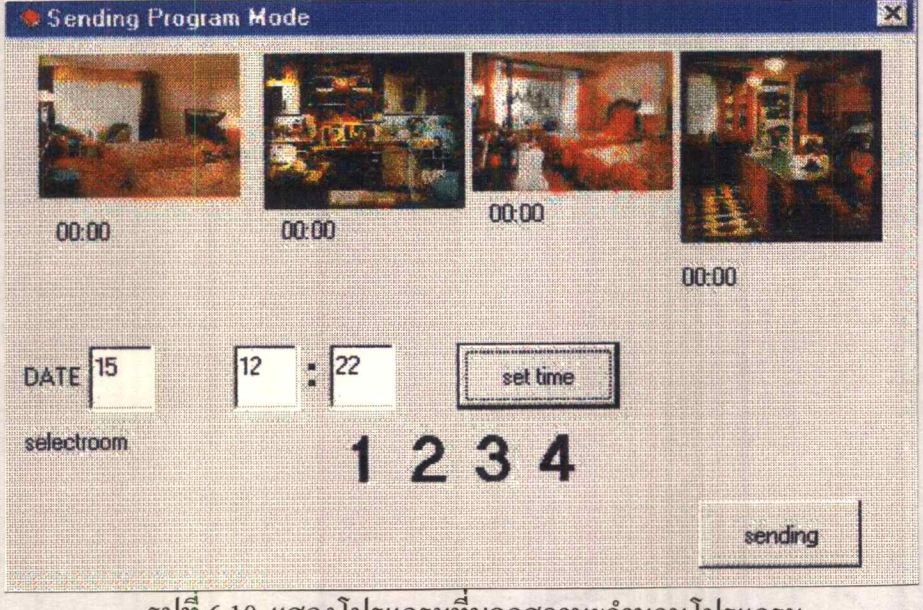


รูปที่ 6.8 แสดงโปรแกรมเมื่อกดปุ่ม new program

โปรแกรมจะแสดงส่วนที่ให้ผู้ใช้ทำการตั้งค่าวันและเวลาในการสั่งงานควบคุมเครื่องใช้ไฟฟ้าต่าง ๆ เมื่อตั้งค่าเหล่านี้แล้วให้กดปุ่ม set time and date โปรแกรมจะแสดงสถานะ (STATUS) ให้ผู้ใช้ทราบว่าขณะนี้โปรแกรมกำลังทำอะไรอยู่และต้องทำอย่างไรต่อไป ขั้นต่อไปจะแสงการเลือกที่จะสั่งการควบคุมเครื่องใช้ไฟฟ้าในห้องต่างๆ เช่น แสดงได้ดังรูป

รูปที่6.9 แสดงโปรแกรมรูปห้อง

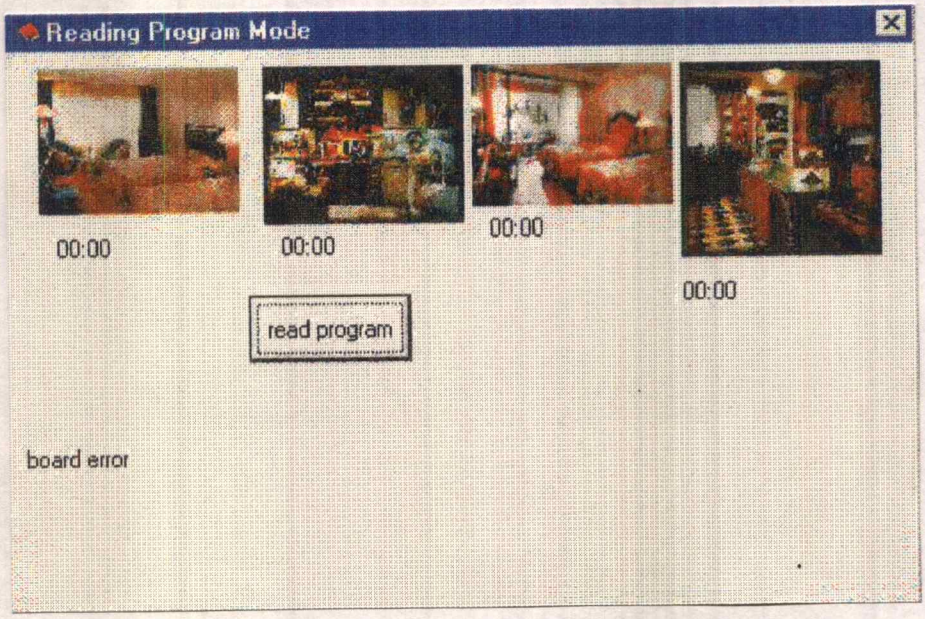
โปรแกรมแสดงรูปห้องซึ่งแสดงค่าวันและเวลาที่ผู้ใช้กำหนดขึ้น ผู้ใช้สามารถเลือกที่จะเปิด-ปิดไฟได้โดยการคลิกเมาส์ไปที่ดวงไฟนั้น ๆ เมื่อกำหนดค่าเรียบร้อยแล้วให้กดปุ่ม set program จำนวนโปรแกรมที่ถูกส่งออกไปโดยผู้ใช้หรือโปรแกรมที่ถูกบันทึกไว้ภายในบอร์ด Real-time จะแสดงจำนวนโปรแกรมต่าง ๆ นั้นบริเวณที่บอกสถานะของโปรแกรมดังรูป



รูปที่ 6.10 แสดงโปรแกรมที่บอกสถานะจำนวนโปรแกรม

<u>กรณีกดปุ่ม read program</u>

ผลของโปรแกรมแสดงดังรูป



รูปที่ 6.11 แสดงโปรแกรมเมื่อกดปุ่ม read program

แสดงจำนวนของโปรแกรมที่ถูกอ่านขึ้นมาจากบอร์ด Real-time ซึ่งผู้ใช้สามารถเลือกที่จะดูโปรแกรมต่าง ๆ เหล่านี้ได้โดยการคลิกเมาส์ไปที่เลขต่าง ๆ ของโปรแกรม

เอกสารอ้างอิง

1. สุทธิศักดิ์ พงศ์ธนาพานิช , Visual Basic 4.0 Professional , ซีเอ็ดยูเคชั่น 2539
2. รีโมต เครื่องควบคุมไร้สาย , ซีเอ็ดยูเคชั่น 2538
3. สุเจตน์ จันทรังษ์ , ไมโครคอนโทรลเลอร์ชิพเดี่ยว , วิทยาลัยมหานคร 2535
4. Thomas L. Floyd , Digital Fundamentals , Prentice Hall 1977

เอกสารอ้างอิง

1. สุทธิศักดิ์ พงศ์ธนาพานิช , Visual Basic 4.0 Professional , ซีเอ็ดยูเคชั่น 2539
2. รีโมต เครื่องควบคุมไร้สาย , ซีเอ็ดยูเคชั่น 2538
3. สุเจตน์ จันทรังษ์ , ไมโครคอนโทรลเลอร์ชิพเดี่ยว , วิทยาลัยมหานคร 2535
4. Thomas L. Floyd , Digital Fundamentals , Prentice Hall 1977

ภาคผนวก

ภาคผนวก

```
;*****************************************************************
;
;               Miniature Realtime Controller V2.0
;                      ws  25 July 1994
;                  1st modified 3 December 1995
;                  for a miniature 89C2051 SCM
;                        2KB code memory
; specifications:
; 89c2051 @3.579545MHz + MAX232C + 74LS07
; programmed via serial port using on-chip data memory for savin
; control code.
;*****************************************************************

;
;

                CPU       "8051.TBL"
                HOF       "INT8"

;
;
;MCS-51 INTERNAL REGISTERS
;
   B:       EQU       0F0H       ;B REGISTER
 ACC:       EQU       0E0H       ;ACCUMULATOR
 PSW:       EQU       0D0H       ;PROGRAM STATUS WORD
 IPC:       EQU       0B8H       ;INTERRUPT PRIORITY
  P3:       EQU       0B0H       ;PORT 3
 IEC:       EQU       0A8H       ;INTERRUPT ENABLE
  P2:       EQU       0A0H       ;PORT 2
SBUF:       EQU       99H        ;SEND BUFFER
SCON:       EQU       98H        ;SERIAL CONTROL
  P1:       EQU       90H        ;PORT 1
 TH1:       EQU       8DH        ;TIMER 1 HIGH
 TH0:       EQU       8CH        ;TIMER 0 HIGH
 TL1:       EQU       8BH        ;TIMER 1 LOW
 TL0:       EQU       8AH        ;TIMER 0 LOW
TMOD:       EQU       89H        ;TIMER MODE
TCON:       EQU       88H        ;TIMER CONTROL
PCON:       EQU       87H        ;POWER CONTROL REGISTER
 DPH:       EQU       83H        ;DATA POINTER HIGH
 DPL:       EQU       82H        ;DATA POINTER LOW
  SP:       EQU       81H        ;STACK POINTER
  P0:       EQU       80H        ;PORT 0
   ;
   ;MCS-51 INTERNAL BIT ADDRESSES
   ;
  CY:       EQU       0D7H       ;CARRY FLAG
  AC:       EQU       0D6H       ;AUXILIARY-CARRY FLAG
  F0:       EQU       0D5H       ;USER FLAG 0
 RS1:       EQU       0D4H       ;REGISTER SELECT MSB
 RS0:       EQU       0D3H       ;REGISTER SELECT LSB
  OV:       EQU       0D2H       ;OVERFLOW FLAG
   P:       EQU       0D0H       ;PARITY FLAG
  PS:       EQU       0BCH       ;PRIORITY SERIAL PORT
 PT1:       EQU       0BBH       ;PRIORITY TIMER 1
 PX1:       EQU       0BAH       ;PRIORITY EXTERNAL 1
 PT0:       EQU       0B9H       ;PRIORITY TIMER 0
 PX0:       EQU       0B8H       ;PRIORITY EXTERNAL 0
  EA:       EQU       0AFH       ;ENABLE ALL INTERRUPT
  ES:       EQU       0ACH       ;ENABLE SERIAL INTERRUPT
 ET1:       EQU       0ABH       ;ENABLE TIMER 1 INTERRUPT
 EX1:       EQU       0AAH       ;ENABLE EXTERNAL 1 INTERR
 ET0:       EQU       0A9H       ;ENABLE TIMER 0 INTERRUPT
 EX0:       EQU       0A8H       ;ENABLE EXTERNAL 0 INTERR
```

```
;*****************************************************************
;
;               Miniature Realtime Controller V2.0
;                      ws  25 July 1994
;                  1st modified 3 December 1995
;                  for a miniature 89C2051 SCM
;                        2KB code memory
; specifications:
; 89c2051 @3.579545MHz + MAX232C + 74LS07
; programmed via serial port using on-chip data memory for savin
; control code.
;*****************************************************************

;
;

                CPU       "8051.TBL"
                HOF       "INT8"
;
;
;MCS-51 INTERNAL REGISTERS
;
   B:       EQU       0F0H       ;B REGISTER
 ACC:       EQU       0E0H       ;ACCUMULATOR
 PSW:       EQU       0D0H       ;PROGRAM STATUS WORD
 IPC:       EQU       0B8H       ;INTERRUPT PRIORITY
  P3:       EQU       0B0H       ;PORT 3
 IEC:       EQU       0A8H       ;INTERRUPT ENABLE
  P2:       EQU       0A0H       ;PORT 2
SBUF:       EQU       99H        ;SEND BUFFER
SCON:       EQU       98H        ;SERIAL CONTROL
  P1:       EQU       90H        ;PORT 1
 TH1:       EQU       8DH        ;TIMER 1 HIGH
 TH0:       EQU       8CH        ;TIMER 0 HIGH
 TL1:       EQU       8BH        ;TIMER 1 LOW
 TL0:       EQU       8AH        ;TIMER 0 LOW
TMOD:       EQU       89H        ;TIMER MODE
TCON:       EQU       88H        ;TIMER CONTROL
PCON:       EQU       87H        ;POWER CONTROL REGISTER
 DPH:       EQU       83H        ;DATA POINTER HIGH
 DPL:       EQU       82H        ;DATA POINTER LOW
  SP:       EQU       81H        ;STACK POINTER
  P0:       EQU       80H        ;PORT 0
   ;
   ;MCS-51 INTERNAL BIT ADDRESSES
   ;
  CY:       EQU       0D7H       ;CARRY FLAG
  AC:       EQU       0D6H       ;AUXILIARY-CARRY FLAG
  F0:       EQU       0D5H       ;USER FLAG 0
 RS1:       EQU       0D4H       ;REGISTER SELECT MSB
 RS0:       EQU       0D3H       ;REGISTER SELECT LSB
  OV:       EQU       0D2H       ;OVERFLOW FLAG
   P:       EQU       0D0H       ;PARITY FLAG
  PS:       EQU       0BCH       ;PRIORITY SERIAL PORT
 PT1:       EQU       0BBH       ;PRIORITY TIMER 1
 PX1:       EQU       0BAH       ;PRIORITY EXTERNAL 1
 PT0:       EQU       0B9H       ;PRIORITY TIMER 0
 PX0:       EQU       0B8H       ;PRIORITY EXTERNAL 0
  EA:       EQU       0AFH       ;ENABLE ALL INTERRUPT
  ES:       EQU       0ACH       ;ENABLE SERIAL INTERRUPT
 ET1:       EQU       0ABH       ;ENABLE TIMER 1 INTERRUPT
 EX1:       EQU       0AAH       ;ENABLE EXTERNAL 1 INTERR
 ET0:       EQU       0A9H       ;ENABLE TIMER 0 INTERRUPT
 EX0:       EQU       0A8H       ;ENABLE EXTERNAL 0 INTERR
```

```
    SM0:      EQU       09FH      ;SERIAL MODE 0
    SM1:      EQU       09EH      ;SERIAL MODE 1
    SM2:      EQU       09DH      ;SERIAL MODE 2
    REN:      EQU       09CH      ;SERIAL RECEPTION ENABLE
    TB8:      EQU       09BH      ;TRANSMITT BIT 8
    RB8:      EQU       09AH      ;RECEIVE BIT 8
     TI:      EQU       099H      ;TRANSMIT INTERRUPT FLAG
     RI:      EQU       098H      ;RECEIVE INTERRUPT FLAG
    TF1:      EQU       08FH      ;TIMER 1 OVERFLOW FLAG
    TR1:      EQU       08EH      ;TIMER 1 RUN CONTROL BIT
    TF0:      EQU       08DH      ;TIMER 0 OVERFLOW FLAG
    TR0:      EQU       08CH      ;TIMER 0 RUN CONTROL BIT
    IE1:      EQU       08BH      ;EXT INTERR. 1 EDGE FLAG
    IT1:      EQU       08AH      ;EXT INTERR. 1 TYPE FLAG
    IE0:      EQU       089H      ;EXT INTERR. 0 EDGE FLAG
    IT0:      EQU       088H      ;EXT INTERR. 0 TYPE FLAG

  P1.0:       EQU       090H      ;  START COMMAND
  P1.1:       EQU       091H      ;  select A-B only or A-B-C-D
  P1.2:       EQU       092H      ;
  P1.3:       EQU       093H      ;
  P1.4:       EQU       094H      ; output soleniod #4  option
  P1.5:       EQU       095H      ; output solenoid #3  option
  P1.6:       EQU       096H      ; output solenoid #2 ---> A-C
  P1.7:       EQU       097H      ; output solenoid #1 ---> A-B

  p2.0:     equ         0a0h      ; time base indicator
  p2.1:     equ         0a1h      ; clock out 50.000 Hz for calibration
  p2.2:     equ         0a2h      ; tone output

  p3.0:     equ         0b0h
  p3.1:     equ         0b1h
  p3.2:     equ         0b2h
  p3.3:     equ         0b3h
  p3.4:     equ         0b4h
  p3.5:     equ         0b5h
  p3.6:     equ         0b6h
  p3.7:     equ         0b7h


 SAVE_L:    EQU         048H      ; SAVE POINTER USE IN INTERRUPT SERVI
 SAVE_H:    EQU         049H      ; ROUTINE
  STATE:    EQU         038H      ; STATE BYTE 1 = TIME


                ; RAM BIT ADDRESS

 serial:    equ          18h      ; on/off send data
    sim:    equ          19h      ; simulate mode (speed-up test)
  blink:    equ          1Ah
 output:    equ          21h      ; 16 bit output cuntrol 21h 22h

    KEY:    EQU          0BH
    LED:    EQU          08H      ; INTERRUPT BLINK
XOFF_FLAG:  EQU          0AH
ZEROING:    EQU          0DH
  VALVE:    EQU          0EH
CYCLE_COUNT:EQU       0FH


      ;current time storage

 sec100:      equ         25h
    sec:      equ         26h
```

```
      SM0:      EQU       09FH      ;SERIAL MODE 0
      SM1:      EQU       09EH      ;SERIAL MODE 1
      SM2:      EQU       09DH      ;SERIAL MODE 2
      REN:      EQU       09CH      ;SERIAL RECEPTION ENABLE
      TB8:      EQU       09BH      ;TRANSMITT BIT 8
      RB8:      EQU       09AH      ;RECEIVE BIT 8
       TI:      EQU       099H      ;TRANSMIT INTERRUPT FLAG
       RI:      EQU       098H      ;RECEIVE INTERRUPT FLAG
      TF1:      EQU       08FH      ;TIMER 1 OVERFLOW FLAG
      TR1:      EQU       08EH      ;TIMER 1 RUN CONTROL BIT
      TF0:      EQU       08DH      ;TIMER 0 OVERFLOW FLAG
      TR0:      EQU       08CH      ;TIMER 0 RUN CONTROL BIT
      IE1:      EQU       08BH      ;EXT INTERR. 1 EDGE FLAG
      IT1:      EQU       08AH      ;EXT INTERR. 1 TYPE FLAG
      IE0:      EQU       089H      ;EXT INTERR. 0 EDGE FLAG
      IT0:      EQU       088H      ;EXT INTERR. 0 TYPE FLAG

    P1.0:       EQU       090H      ;  START COMMAND
    P1.1:       EQU       091H      ;  select A-B only or A-B-C-D
    P1.2:       EQU       092H      ;
    P1.3:       EQU       093H      ;
    P1.4:       EQU       094H      ; output soleniod #4  option
    P1.5:       EQU       095H      ; output solenoid #3  option
    P1.6:       EQU       096H      ; output solenoid #2 ---> A-C
    P1.7:       EQU       097H      ; output solenoid #1 ---> A-B

    p2.0:     equ       0a0h      ; time base indicator
    p2.1:     equ       0a1h      ; clock out 50.000 Hz for calibration
    p2.2:     equ       0a2h      ; tone output

    p3.0:     equ       0b0h
    p3.1:     equ       0b1h
    p3.2:     equ       0b2h
    p3.3:     equ       0b3h
    p3.4:     equ       0b4h
    p3.5:     equ       0b5h
    p3.6:     equ       0b6h
    p3.7:     equ       0b7h


  SAVE_L:     EQU       048H      ; SAVE POINTER USE IN INTERRUPT SERVI
  SAVE_H:     EQU       049H      ; ROUTINE
   STATE:     EQU       038H      ; STATE BYTE 1 = TIME


                ; RAM BIT ADDRESS

  serial:     equ         18h     ; on/off send data
     sim:     equ         19h     ; simulate mode (speed-up test)
   blink:     equ         1Ah
  output:     equ         21h     ; 16 bit output cuntrol 21h 22h

     KEY:     EQU         0BH
     LED:     EQU         08H     ; INTERRUPT BLINK
XOFF_FLAG:    EQU         0AH
  ZEROING:    EQU         0DH
   VALVE:     EQU         0EH
CYCLE_COUNT:  EQU      0FH


       ;current time storage

  sec100:      equ         25h
     sec:      equ         26h
```

```
       min:       equ        27h
      hour:       equ        28h
       day:       equ        29h
     MONTH:       EQU        2AH
      YEAR:       EQU        2BH
 warm_code:       equ        2Eh
     light:       equ        2dh
     econo:       equ        2ch
    tl0adj:       equ        2fh
pgm_buffer:       equ        30h     ; start from 30h


CEIVE_BUFFER:     EQU        4AH

   TOP_FIFO:      EQU       1000H    ; TOP OF FIFO BUFFER

        CR:        EQU        0DH
        LF:        EQU        0AH
       EOS:       EQU        10H
      BELL:      EQU        07H


        ;*******************************************************
        ;

        ORG        000H
                   LJMP      MAIN_TIMER

        org        003h
        ljmp       service_int0  ; for airconditioner start manually

        ORG        00BH
                   LJMP      SERVICE_T0 ; TIMER COUNTER 0 INTERRUPT


                   ORG       0100H

        ;INIT_RS232C INITIALIZED 89C2051 SERIAL PORT
        ;            BAUD RATE: 9600
        ;            DATA LENGTH: 8 BIT
        ;            STOP BIT: 1 BIT
        ;            PARITY: NO
        ;            X-TAL: 3.579545 MHz

    INIT:     MOV        TMOD,#00100001B
              MOV        SCON,#01010010B
              MOV        PCON,#10000000B ; smod = 1
              MOV        TH1,#0FEH       ;TIMER1 LOAD VALUE
              SETB       TR1             ;START TIMER1
              RET


        ;SEND     SEND A TO SERIAL PORT
        ;         CHECK TRANSMITTER BUFFER BEFORE SEND
        ;
        ;         ENTRY: A
        ;         EXIT: NO
        ;
    SEND:     JNB        TI,SEND
              CLR        TI
              MOV        SBUF,A          ;OK BUFFER EMPTY SEND OUT...
              RET
```

```
        min:        equ         27h
       hour:        equ         28h
        day:        equ         29h
      MONTH:        EQU         2AH
       YEAR:        EQU         2BH
  warm_code:        equ         2Eh
      light:        equ         2dh
      econo:        equ         2ch
     tl0adj:        equ         2fh
 pgm_buffer:        equ         30h     ; start from 30h


CEIVE_BUFFER:       EQU         4AH

    TOP_FIFO:       EQU        1000H    ; TOP OF FIFO BUFFER

         CR:         EQU         0DH
         LF:         EQU         0AH
        EOS:        EQU         10H
       BELL:       EQU         07H


        ;*******************************************************
        ;

        ORG         000H
                    LJMP        MAIN_TIMER

        org         003h
        ljmp        service_int0  ; for airconditioner start manually

        ORG         00BH
                    LJMP        SERVICE_T0 ; TIMER COUNTER 0 INTERRUPT


                    ORG         0100H

        ;INIT_RS232C INITIALIZED 89C2051 SERIAL PORT
        ;            BAUD RATE: 9600
        ;            DATA LENGTH: 8 BIT
        ;            STOP BIT: 1 BIT
        ;            PARITY: NO
        ;            X-TAL: 3.579545 MHz

      INIT:     MOV         TMOD,#00100001B
                MOV         SCON,#01010010B
                MOV         PCON,#10000000B ; smod = 1
                MOV         TH1,#0FEH       ;TIMER1 LOAD VALUE
                SETB        TR1             ;START TIMER1
                RET


        ;SEND    SEND A TO SERIAL PORT
        ;        CHECK TRANSMITTER BUFFER BEFORE SEND
        ;
        ;        ENTRY: A
        ;        EXIT: NO
        ;
      SEND:     JNB         TI,SEND
                CLR         TI
                MOV         SBUF,A          ;OK BUFFER EMPTY SEND OUT...
                RET
```

```
        ;
        ;RECIVES RECIVE BYTE FROM SERIAL PORT WITH TIME-OUT FUNCTION
        ;        ENTRY: NO
        ;        EXIT: A = ASCII NUMBER OR STRING COMMAND
        ;                                        A = FF  TIME-OUT

 RECIVES:    MOV     R7,#05H
   AGAIN:    MOV     R6,#00H          ; TIME-OUT DELAY
  GET_RI:    JB      RI,READY

                    DJNZ    R6,GET_RI
                    DJNZ    R7,AGAIN
                    MOV     A,#0FFH          ; TIME-OUT NO DATA RECEPTE
                    RET
    READY:    CLR     RI
                    MOV     A,SBUF
                    RET


        ;RECIVE  RECIVE BYTE PUT TO A FROM SERIAL PORT
        ;        WAIT UNTIL RECIVER BUFFER READY
        ;        ENTRY: NO
        ;        EXIT: A

   RECIVE:  JNB        RI,RECIVE
            CLR        RI
            MOV        A,SBUF       ; GET IT
            LCALL      SEND         ; ECHO TO TERMINAL
            RET

        ;SEND_PROMPT

SEND_PROMPT: LCALL    CR_LF
             MOV      DPTR,#PROMPT
             LCALL    SEND_STRING
             RET


        ;CR_SEND SEND CR TO RS232
        ;        ENTRY: NO
        ;        EXIT: NO
        ;
     CR_LF:   MOV        A,#CR
              LCALL      SEND
              MOV        A,#LF
              LCALL      SEND
              RET

      XOFF:      EQU         13H
       XON:      EQU         11H
     BREAK:      EQU         00H
       EOF:      equ         0FH


        ;SEND_XOFF SEND XOFF TO RS232
        ;        ENTRY: NO
        ;        EXIT: NO
        ;
 SEND_XOFF: MOV      A,#XOFF    ;GET XOFF CHARACTER
            LCALL     SEND
            RET

        ;SEND_XON
        ;
```

```
        ;
        ;RECIVES RECIVE BYTE FROM SERIAL PORT WITH TIME-OUT FUNCTION
        ;        ENTRY: NO
        ;        EXIT: A = ASCII NUMBER OR STRING COMMAND
        ;                                          A = FF  TIME-OUT

RECIVES:    MOV     R7,#05H
  AGAIN:    MOV     R6,#00H          ; TIME-OUT DELAY
 GET_RI:    JB      RI,READY

                    DJNZ    R6,GET_RI
                    DJNZ    R7,AGAIN
                    MOV     A,#0FFH          ; TIME-OUT NO DATA RECEPTE
                    RET
  READY:    CLR      RI
                    MOV     A,SBUF
                    RET


        ;RECIVE  RECIVE BYTE PUT TO A FROM SERIAL PORT
        ;        WAIT UNTIL RECIVER BUFFER READY
        ;        ENTRY: NO
        ;        EXIT: A

  RECIVE:  JNB      RI,RECIVE
           CLR      RI
           MOV      A,SBUF      ; GET IT
           LCALL    SEND        ; ECHO TO TERMINAL
           RET

        ;SEND_PROMPT

SEND_PROMPT: LCALL   CR_LF
             MOV     DPTR,#PROMPT
             LCALL   SEND_STRING
             RET


        ;CR_SEND SEND CR TO RS232
        ;        ENTRY: NO
        ;        EXIT: NO
        ;
   CR_LF:   MOV      A,#CR
            LCALL    SEND
            MOV      A,#LF
            LCALL    SEND
            RET

    XOFF:     EQU      13H
     XON:     EQU      11H
   BREAK:     EQU      00H
     EOF:     equ      0FH


        ;SEND_XOFF SEND XOFF TO RS232
        ;        ENTRY: NO
        ;        EXIT: NO
        ;
SEND_XOFF: MOV     A,#XOFF    ;GET XOFF CHARACTER
           LCALL     SEND
           RET

        ;SEND_XON
        ;
```

```
        ;
        ;
SEND_XON: MOV       A,#XON
          LCALL     SEND
          RET
        ;SPACE

   SPACE:     MOV      A,#20H
              LCALL    SEND
              RET

        ;ASCII_BIN  CONVERTS SINGLE ASCII CHARACTER TO SINGLE NIBBLE
        ;           BINARY
        ;          ENTRY: A ( 30-39 for 0-9, 41-46 for A-F, 61-66 for a-f
        ;          EXIT: A
        ;          ignor lower case clr bit 5 before

ASCII_BIN:     CLR       C
               MOV       R6,A
               SUBB      A,#41H
               JNC       ASCII_AF
               MOV       A,R6
               CLR       C
               SUBB      A,#30H
               RET
 ASCII_F:      CLR       C
               MOV       A,R6
               SUBB      A,#37H
               RET

        ;COMBINE DATA 2 BYTE TO SINGLE BYTE
        ;         ENTRY: R4= LOW NIBBLE
        ;                R5= HIGH NIBBLE
        ;         EXIT:  A
        ;
 COMBINE:      MOV         A,R5
               SWAP        A
               ADD         A,R4
               RET

        ;GET_BYTE GET DATA FROM SERIAL PORT TWO BYTE SAVE TO
        ;        R5 AS A HIGH NIBBLE AND R4 AS A LOW NIBBLE
        ;        ENTRY: NO, serial port must be initialzed before callin
        ;        EXIT: A
        ;
        ;
GET_BYTE:   LCALL       RECIVE ; FIRST READING MUST GO TO R5 !!
            LCALL       ASCII_BIN
            MOV         R5,A
            LCALL       RECIVE
            LCALL       ASCII_BIN
            MOV         R4,A
            LCALL       COMBINE
            RET


        ;BIN_ASCII CONVERT BIN TO ASCII CODE
        ;         ENTRY: A
        ;         EXIT : A
BIN_ASCII:
                  ANL      A,#0FH
                  MOV      R2,A
                  CLR      C
                  SUBB     A,#0AH
```

```
        ;
        ;
SEND_XON: MOV       A,#XON
          LCALL     SEND
          RET
        ;SPACE

   SPACE:     MOV      A,#20H
              LCALL    SEND
              RET

        ;ASCII_BIN  CONVERTS SINGLE ASCII CHARACTER TO SINGLE NIBBLE
        ;           BINARY
        ;          ENTRY: A ( 30-39 for 0-9, 41-46 for A-F, 61-66 for a-f
        ;          EXIT: A
        ;          ignor lower case clr bit 5 before

ASCII_BIN:     CLR       C
               MOV       R6,A
               SUBB      A,#41H
               JNC       ASCII_AF
               MOV       A,R6
               CLR       C
               SUBB      A,#30H
               RET
 ASCII_F:      CLR       C
               MOV       A,R6
               SUBB      A,#37H
               RET

        ;COMBINE DATA 2 BYTE TO SINGLE BYTE
        ;         ENTRY: R4= LOW NIBBLE
        ;                R5= HIGH NIBBLE
        ;         EXIT:  A
        ;
 COMBINE:      MOV         A,R5
               SWAP        A
               ADD         A,R4
               RET

        ;GET_BYTE GET DATA FROM SERIAL PORT TWO BYTE SAVE TO
        ;         R5 AS A HIGH NIBBLE AND R4 AS A LOW NIBBLE
        ;         ENTRY: NO, serial port must be initialzed before callin
        ;         EXIT: A
        ;
        ;
GET_BYTE:   LCALL       RECIVE ; FIRST READING MUST GO TO R5 !!
            LCALL       ASCII_BIN
            MOV         R5,A
            LCALL       RECIVE
            LCALL       ASCII_BIN
            MOV         R4,A
            LCALL       COMBINE
            RET


        ;BIN_ASCII CONVERT BIN TO ASCII CODE
        ;          ENTRY: A
        ;          EXIT : A
BIN_ASCII:
                   ANL      A,#0FH
                   MOV      R2,A
                   CLR      C
                   SUBB     A,#0AH
```

```
                JNC     ASCII_AF2
                MOV     A,R2
                ADD     A,#30H
                RET
ASCII_AF2:      MOV     A,R2
                ADD     A,#37H
                RET

        ;BYTE_ASCII  CONVERT A TO ASCII IN R4, R5

BYTE_ASCII:     MOV     B,A
                LCALL   BIN_ASCII

                MOV     R4,A
                MOV     A,B
                SWAP    A
                LCALL   BIN_ASCII
                MOV     R5,A
                RET

        ;SEND_ASCII SEND ASCII IN R5(HI) R4(LO) BYTE TO MONITOR
        ;           INPUT : A
        ;           OUTPUT : NONE
SEND_ASCII:     LCALL   BYTE_ASCII
                MOV     A,R5
                LCALL   SEND
                MOV     A,R4
                LCALL    SEND
                RET


        ;NIBBLE_SHIFT SHIFT FOUR DIGIT BCD NUMBER LEFT A NIBBLE
        ;            ENTRY : R0 POINTED TO LSD
        ;            EXIT : NO

        NIBBLE_SHIFT:           INC   R0
                                MOV   A,@R0
                                SWAP  A
                                ANL   A,#0F0H
                                MOV   R1,A
                                DEC   R0
                                MOV   A,@R0
                                SWAP  A
                                MOV   R2,A
                                ANL   A,#0FH
                                ADD   A,R1
                                INC   R0
                                MOV   @R0,A
                                MOV   A,R2
                                ANL   A,#0F0H
                                DEC   R0
                                MOV   @R0,A
                                RET


        ;SHIFT_16BIT SHIFT LEFT 1 BIT ADDRESS $3D,$3E
        ;            $3D  LO BYTE
        ;            $3E  HI BYTE
        ;            ENTRY: NO
        ;            EXIT: Carry Flag

        SHIFT_16BIT:        CLR     C           ;     {($3E,$3D)} X 2
                            MOV     A,3DH
                            ADD     A,3DH
```

```
                JNC     ASCII_AF2
                MOV     A,R2
                ADD     A,#30H
                RET
ASCII_AF2:      MOV     A,R2
                ADD     A,#37H
                RET

        ;BYTE_ASCII  CONVERT A TO ASCII IN R4, R5

BYTE_ASCII:     MOV     B,A
                LCALL   BIN_ASCII

                MOV     R4,A
                MOV     A,B
                SWAP    A
                LCALL   BIN_ASCII
                MOV     R5,A
                RET

        ;SEND_ASCII SEND ASCII IN R5(HI) R4(LO) BYTE TO MONITOR
        ;           INPUT : A
        ;           OUTPUT : NONE
SEND_ASCII:     LCALL   BYTE_ASCII
                MOV     A,R5
                LCALL   SEND
                MOV     A,R4
                LCALL    SEND
                RET


        ;NIBBLE_SHIFT SHIFT FOUR DIGIT BCD NUMBER LEFT A NIBBLE
        ;           ENTRY : R0 POINTED TO LSD
        ;           EXIT : NO

        NIBBLE_SHIFT:           INC   R0
                                MOV   A,@R0
                                SWAP  A
                                ANL   A,#0F0H
                                MOV   R1,A
                                DEC   R0
                                MOV   A,@R0
                                SWAP  A
                                MOV   R2,A
                                ANL   A,#0FH
                                ADD   A,R1
                                INC   R0
                                MOV   @R0,A
                                MOV   A,R2
                                ANL   A,#0F0H
                                DEC   R0
                                MOV   @R0,A
                                RET


        ;SHIFT_16BIT SHIFT LEFT 1 BIT ADDRESS $3D,$3E
        ;           $3D  LO BYTE
        ;           $3E  HI BYTE
        ;           ENTRY: NO
        ;           EXIT: Carry Flag

        SHIFT_16BIT:    CLR     C           ;       {($3E,$3D)} X 2
                        MOV     A,3DH
                        ADD     A,3DH
```

```
                    MOV     3DH,A
                    MOV     A,3EH
                    ADDC    A,3EH
                    MOV     3EH,A
                    RET                     ; CARRY GONE

;BIN_BCD CONVERT 16 BIT BINARY TO BCD
;        SHIFT LEFT 1 BIT FROM MOST SIGNIFICANT BIT TO LEAST SIG
;        BIT , IF CARRY = 1 THEN BCD= (2^N + BCD)
;
;                 ENTRY: 16 BIT DATA
;               $3D = LO BYTE
;               $3E = HI BYTE
;       EXIT: 6 DIGIT BCD
;               $36L =      0
;               $36H =     10
;               $37L =    100
;               $37H =   1000
;               $38L =  10000
;               $38H = 100000

BIN_BCD:    MOV     36H,#00     ; CLEAR BCD BEFORE
            MOV     37H,#00
            MOV     38H,#00

            MOV     R1,#00H
            MOV     R7,#16D
BIN_1:     LCALL   SHIFT_16BIT
           JNC     NEXT_BIT

;      OK CARRY = 1 ADD BCD TO 2^N CONSTANT IN TABLE2
            MOV     DPTR,#TABLE2
            CLR     C
            MOV     R0,#36H
            MOV     R6,#03H
BIN_2:      MOV     A,R1
            MOVC    A,@A+DPTR
            ADDC     A,@R0
            DAA
            MOV     @R0,A
            INC     DPTR
            INC     R0
            DJNZ    R6,BIN_2

NEXT_BIT:   INC     R1
            INC     R1
            INC     R1
            DJNZ    R7,BIN_1
            RET

  TABLE2:   DFB     68H,27H,03H
            DFB     84H,63H,01H
            DFB     92H,81H,00H
            DFB     96H,40H,00H
            DFB     48H,20H,00H
            DFB     24H,10H,00H
            DFB     12H,05H,00H
            DFB     56H,02H,00H
            DFB     28H,01H,00H
            DFB     64H,00H,00H
            DFB     32H,00H,00H
            DFB     16H,00H,00H
            DFB     08H,00H,00H
            DFB     04H,00H,00H
```

```
                    MOV     3DH,A
                    MOV     A,3EH
                    ADDC    A,3EH
                    MOV     3EH,A
                    RET                     ; CARRY GONE

;BIN_BCD CONVERT 16 BIT BINARY TO BCD
;        SHIFT LEFT 1 BIT FROM MOST SIGNIFICANT BIT TO LEAST SIG
;        BIT , IF CARRY = 1 THEN BCD= (2^N + BCD)
;
;                 ENTRY: 16 BIT DATA
;               $3D = LO BYTE
;               $3E = HI BYTE
;       EXIT: 6 DIGIT BCD
;               $36L =      0
;               $36H =     10
;               $37L =    100
;               $37H =   1000
;               $38L =  10000
;               $38H = 100000

BIN_BCD:    MOV     36H,#00     ; CLEAR BCD BEFORE
            MOV     37H,#00
            MOV     38H,#00

            MOV     R1,#00H
            MOV     R7,#16D
BIN_1:    LCALL   SHIFT_16BIT
          JNC     NEXT_BIT

;     OK CARRY = 1 ADD BCD TO 2^N CONSTANT IN TABLE2
            MOV     DPTR,#TABLE2
            CLR     C
            MOV     R0,#36H
            MOV     R6,#03H
BIN_2:      MOV     A,R1
            MOVC    A,@A+DPTR
            ADDC     A,@R0
            DAA
            MOV     @R0,A
            INC     DPTR
            INC     R0
            DJNZ    R6,BIN_2

NEXT_BIT:   INC     R1
            INC     R1
            INC     R1
            DJNZ    R7,BIN_1
            RET

  TABLE2:   DFB     68H,27H,03H
            DFB     84H,63H,01H
            DFB     92H,81H,00H
            DFB     96H,40H,00H
            DFB     48H,20H,00H
            DFB     24H,10H,00H
            DFB     12H,05H,00H
            DFB     56H,02H,00H
            DFB     28H,01H,00H
            DFB     64H,00H,00H
            DFB     32H,00H,00H
            DFB     16H,00H,00H
            DFB     08H,00H,00H
            DFB     04H,00H,00H
```

```
            DFB     02H,00H,00H
            DFB     01H,00H,00H

;#################################################################
;           TIMER 0 INTERRUPT SERVICE ROUTINE
;          enter to this routine every 1/10 Hz
;            update current time and date
;            check current day and time every 1 sec
;            if current time =  set time the activates 8 output
;
;#################################################################

SERVICE_T0:                             PUSH    PSW
                                        PUSH    ACC
                                        PUSH    B
                                        PUSH    DPL
                                        PUSH    DPH
                                        MOV     PSW,#00001000B  ; SELEC

; TIMER 0 SERVICE BODY

;           cpl     p3.5                ; check time base should be

            MOV     TH0,#COUNT_H
            MOV     TL0,tl0adj          ; get low byte adjustable
           lcall    light_on
            inc     sec100              ;
            mov     a,sec100
            cjne    a,#0AH,CHECK_SEC    ; update second every 10 cou

            inc     econo
            mov     a,econo
            cjne    a,#05h,skip_econo
            mov     econo,#00h
            setb    blink
            clr     p3.7                ; turn on light every 5 seco
                                ; since inc sec100 as binary
skip_econo:

          lcall  io_out
          jnb    serial,no_send

LCALL SEND_TIME

no_send:  mov     sec100,#00h
          mov     a,sec
          add     a,#01d
          daa
          mov     sec,a


CHECK_SEC:
          mov     a,sec
          cjne    a,#60H,CHECK_MIN
          jnb     sim,min_send
          lcall   send_time

min_send:
          mov     sec,#00d
          mov     a,min
          add     a,#01d
          daa
          mov     min,a
```

```
            DFB     02H,00H,00H
            DFB     01H,00H,00H

;#############################################################
;           TIMER 0 INTERRUPT SERVICE ROUTINE
;          enter to this routine every 1/10 Hz
;            update current time and date
;            check current day and time every 1 sec
;            if current time =  set time the activates 8 output
;
;#############################################################

SERVICE_T0:                             PUSH    PSW
                                        PUSH    ACC
                                        PUSH    B
                                        PUSH    DPL
                                        PUSH    DPH
                                        MOV     PSW,#00001000B  ; SELEC

; TIMER 0 SERVICE BODY

;           cpl     p3.5                ; check time base should be

            MOV     TH0,#COUNT_H
            MOV     TL0,tl0adj          ; get low byte adjustable
           lcall    light_on
            inc     sec100              ;
            mov     a,sec100
            cjne    a,#0AH,CHECK_SEC    ; update second every 10 cou

            inc     econo
            mov     a,econo
            cjne    a,#05h,skip_econo
            mov     econo,#00h
            setb    blink
            clr     p3.7                ; turn on light every 5 seco
                                ; since inc sec100 as binary
skip_econo:

          lcall  io_out
          jnb    serial,no_send

LCALL SEND_TIME

no_send:  mov     sec100,#00h
          mov     a,sec
          add     a,#01d
          daa
          mov     sec,a


CHECK_SEC:
          mov     a,sec
          cjne    a,#60H,CHECK_MIN
          jnb     sim,min_send
          lcall   send_time

min_send:
          mov     sec,#00d
          mov     a,min
          add     a,#01d
          daa
          mov     min,a
```

```
CHECK_MIN:
            mov     a,min
            cjne    a,#60H,CHECK_HOUR
            mov     min,#00d
            mov     a,hour
            add     a,#01d
            daa
            mov     hour,a

CHECK_HOUR:
            mov   a,hour
            cjne  a,#24h,check_day
            mov   hour,#00d
            mov   a,day
            add   a,#01d
            daa
            mov   day,a

 check_day:
            lcall get_month_day
            mov   a,day
            cjne  a,b,check_month
            mov   day,#01d
            mov   a,month
            add   a,#01d
            daa
            mov   month,a

check_month:
            mov   a,month
            cjne  a,#13h,check_year
            mov   month,#01d
            mov   a,year
            add   a,#01d
            daa
            mov   year,a

 check_year:


                        POP   DPH
                        POP   DPL
                        POP   B
                        POP   ACC
                        POP   PSW
                        RETI


   ;send_output send output (display) to terminal for simulate test

   send_outputs:
                        lcall space
                        lcall space
                        mov   r6,#01d                    ; tiny timer
                        mov   dptr,#output_prompt
                        lcall send_string
                        mov   r0,#output
          outer:        mov   r7,#08d
                        mov   a,@r0
                        cpl   a                          ; sink curren
                        MOV   P1,A                       ; send curren
                        cpl   a
                        mov   b,a
          inner:        mov   a,b
```

```
CHECK_MIN:
            mov     a,min
            cjne    a,#60H,CHECK_HOUR
            mov     min,#00d
            mov     a,hour
            add     a,#01d
            daa
            mov     hour,a

CHECK_HOUR:
            mov   a,hour
            cjne  a,#24h,check_day
            mov   hour,#00d
            mov   a,day
            add   a,#01d
            daa
            mov   day,a

 check_day:
            lcall get_month_day
            mov   a,day
            cjne  a,b,check_month
            mov   day,#01d
            mov   a,month
            add   a,#01d
            daa
            mov   month,a

check_month:
            mov   a,month
            cjne  a,#13h,check_year
            mov   month,#01d
            mov   a,year
            add   a,#01d
            daa
            mov   year,a

 check_year:


                      POP   DPH
                      POP   DPL
                      POP   B
                      POP   ACC
                      POP   PSW
                      RETI


  ;send_output send output (display) to terminal for simulate test

  send_outputs:
                      lcall space
                      lcall space
                      mov   r6,#01d                    ; tiny timer
                      mov   dptr,#output_prompt
                      lcall send_string
                      mov   r0,#output
        outer:        mov   r7,#08d
                      mov   a,@r0
                      cpl   a                          ; sink curren
                      MOV   P1,A                       ; send curren
                      cpl   a
                      mov   b,a
        inner:        mov   a,b
```

```
                    rlc   a
                    mov   b,a
                    jc    on_contact
                    mov   a,#"-"
                    lcall send
                    lcall space
                    sjmp  next_bits

  on_contact:
                    mov   a,#"0"
                    lcall send
                    lcall space

   next_bits:       djnz  r7,inner
                    inc   r0
                    djnz  r6,outer
                    ret

;light_on blink led shows time base is run

    light_on:   jnb   blink,led_on ; if blink not set do not cou
                mov   a,light
                inc   a
                mov   light,a
                cjne  a,#01h,led_on
                mov   light,#00h
               setb   p3.7          ; off light
               clr    blink         ; finished blink
   led_on:     ret

;send_tone send tone like beep beep to p2.2

send_tone:  mov   r7,#00h
tone0:      clr   p2.2
            mov   r6,#50h
tone1:      djnz  r6,tone1
setb  p2.2
mov   r6,#50h

tone2:      djnz  r6,tone2
            djnz  r7,tone0
            ret


;BCD_ASCII CONVERT BCD NUMBER TO ASCII CODE
;          ENTRY: A {LOW NIBBLE ONLY}
;          EXIT: A

BCD_ASCII:  ANL       A,#0FH          ;GET ONLY LOW NIBBLE
            ADD       A,#30H          ;CONVERT TO ASCII CODE
            RET

;send current time

send_time:

LCALL CR_LF
MOV   A,DAY
LCALL SEND_ASCII
MOV   A,#"-"
LCALL SEND
MOV   A,MONTH
LCALL SEND_ASCII
MOV   A,#"-"
```

```
                    rlc   a
                    mov   b,a
                    jc    on_contact
                    mov   a,#"-"
                    lcall send
                    lcall space
                    sjmp  next_bits

  on_contact:
                    mov   a,#"O"
                    lcall send
                    lcall space

   next_bits:       djnz  r7,inner
                    inc   r0
                    djnz  r6,outer
                    ret

;light_on blink led shows time base is run

    light_on:   jnb   blink,led_on ; if blink not set do not cou
                mov   a,light
                inc   a
                mov   light,a
                cjne  a,#01h,led_on
                mov   light,#00h
               setb   p3.7          ; off light
               clr    blink         ; finished blink
   led_on:     ret

;send_tone send tone like beep beep to p2.2

send_tone:  mov   r7,#00h
tone0:      clr   p2.2
            mov   r6,#50h
tone1:      djnz  r6,tone1
setb  p2.2
mov   r6,#50h

tone2:      djnz  r6,tone2
            djnz  r7,tone0
            ret


;BCD_ASCII CONVERT BCD NUMBER TO ASCII CODE
;          ENTRY: A {LOW NIBBLE ONLY}
;          EXIT: A

BCD_ASCII:  ANL       A,#0FH          ;GET ONLY LOW NIBBLE
            ADD       A,#30H          ;CONVERT TO ASCII CODE
            RET

;send current time

send_time:

LCALL CR_LF
MOV   A,DAY
LCALL SEND_ASCII
MOV   A,#"-"
LCALL SEND
MOV   A,MONTH
LCALL SEND_ASCII
MOV   A,#"-"
```

```
   LCALL SEND
   MOV   A,#25H
   LCALL SEND_ASCII
   MOV   A,YEAR
   LCALL SEND_ASCII
   LCALL SPACE


   MOV   A,HOUR
   LCALL SEND_ASCII
   MOV   A,#":"
   LCALL SEND
   MOV   A,MIN
   LCALL SEND_ASCII
   MOV   A,#":"
   LCALL SEND
   MOV   A,SEC
   LCALL SEND_ASCII
   lcall send_outputs


   RET



   ;io_out send output o1-o8 when current time = setting time
   ;         scan setting time and compare current time
   ;               if current time = set time then activate i/o
   ;                els next set time till eof found

   io_out:         mov   r0,#pgm_buffer
      io0:         mov   a,@r0
                   cjne  a,#eof,io1
                   ret


      io1:         jb    p3.5,skip_day     ; if p3.7 = 0 then skip da
                   cjne  a,day,io4         ; check day

skip_day:          inc   r0
                   mov   a,@r0
                   cjne  a,hour,io2        ; check hour
                   inc   r0
                   mov   a,@r0
                   cjne  a,min,io3         ; check min
                   inc   r0                ; current time = set time
                   mov   a,@r0             ; get output
                   mov   output,a          ; put to output byte
                   cpl   a                 ; i.e., sink current is us
                   mov   p1,a              ; activate port 1 also
                   ret

      io4:         inc   r0
      io2:         inc   r0
      io3:         inc   r0
                   inc   r0
                   sjmp  io0


   ;GET_MONTH_DAY get day-end of each month, e.g. Jan=31, Feb=28, M
   ;              ,etc.
   ;            entry: current month 01,02,03,...,12
   ;            exit:  B = day-end
   ;
```

```
   LCALL SEND
   MOV   A,#25H
   LCALL SEND_ASCII
   MOV   A,YEAR
   LCALL SEND_ASCII
   LCALL SPACE


   MOV   A,HOUR
   LCALL SEND_ASCII
   MOV   A,#":"
   LCALL SEND
   MOV   A,MIN
   LCALL SEND_ASCII
   MOV   A,#":"
   LCALL SEND
   MOV   A,SEC
   LCALL SEND_ASCII
   lcall send_outputs


   RET



   ;io_out send output o1-o8 when current time = setting time
   ;         scan setting time and compare current time
   ;               if current time = set time then activate i/o
   ;                els next set time till eof found

   io_out:         mov   r0,#pgm_buffer
      io0:         mov   a,@r0
                   cjne  a,#eof,io1
                   ret


      io1:         jb    p3.5,skip_day     ; if p3.7 = 0 then skip da
                   cjne  a,day,io4         ; check day

skip_day:          inc   r0
                   mov   a,@r0
                   cjne  a,hour,io2        ; check hour
                   inc   r0
                   mov   a,@r0
                   cjne  a,min,io3         ; check min
                   inc   r0                ; current time = set time
                   mov   a,@r0             ; get output
                   mov   output,a          ; put to output byte
                   cpl   a                 ; i.e., sink current is us
                   mov   p1,a              ; activate port 1 also
                   ret

      io4:         inc   r0
      io2:         inc   r0
      io3:         inc   r0
                   inc   r0
                   sjmp  io0


   ;GET_MONTH_DAY get day-end of each month, e.g. Jan=31, Feb=28, M
   ;               ,etc.
   ;             entry: current month 01,02,03,...,12
   ;             exit:  B = day-end
   ;
```

```
get_month_day: mov  a,month   ; get current month
               clr  c
               subb a,#10h
               jc   bcd_is_binary
               mov  a,month
               clr  c
               subb a,#06h
               sjmp convert_to_binary

bcd_is_binary:  mov  a,month
convert_to_binary:  dec  a
               mov  dptr,#day_end_table
               movc a,@a+dptr
               mov  b,a
               inc  b                ; to compensate increment
               ret

 day_end_table: dfb  31h  ; Jan
               dfb  28h  ; Feb
               dfb  31h  ; Mar
               dfb  30h  ; Apr
               dfb  31h  ; May
               dfb  30h  ; Jun
               dfb  31h  ; Jul
               dfb  31h  ; Aug
               dfb  30h  ; Sep
               dfb  31h  ; Oct
               dfb  30h  ; Nov
               dfb  31h  ; Dec


;BUFFER_ASCII CONVERT BCD IN BUFFER START $30 - $35
;         ENTRY: BCD NUMBER IN TRANSMIT BUFFER
;         EXIT: ASCII CODE IN TRANSMIT BUFFER

BUFFER_ASCII:       MOV    R7,#06H
                    MOV    R0,#30H
  NEXT_ASCII:       MOV    A,@R0
                    LCALL  BCD_ASCII
                    MOV    @R0,A
                    INC    R0
                    DJNZ   R7,NEXT_ASCII
                    RET

;SEND_BUFFER SEND ASCII CODE IN TRANSMIT BUFFER TO SERIAL PORT
;         THEN DATA SEPERATOR /32/32
;         ENTRY: ASCII CODE IN TRANSMIT BUFFER
;         EXIT: NO

SEND_BUFFER:        MOV     R7,#06H
                    MOV     R0,#35H
  NEXT_BYTE:        MOV     A,@R0
                    LCALL   SEND        ;SEND OUT TO SERIAL PORT
                    DEC     R0
                    DJNZ    R7,NEXT_BYTE
                    lcall   cr_lf

                    RET


;service_int0 activates three output port manually

service_int0:       push acc
```

```
get_month_day: mov  a,month    ; get current month
               clr  c
               subb a,#10h
               jc   bcd_is_binary
               mov  a,month
               clr  c
               subb a,#06h
               sjmp convert_to_binary

bcd_is_binary:  mov  a,month
convert_to_binary:  dec  a
                    mov  dptr,#day_end_table
                    movc a,@a+dptr
                    mov  b,a
                    inc  b                ; to compensate increment
                    ret

day_end_table: dfb  31h  ; Jan
               dfb  28h  ; Feb
               dfb  31h  ; Mar
               dfb  30h  ; Apr
               dfb  31h  ; May
               dfb  30h  ; Jun
               dfb  31h  ; Jul
               dfb  31h  ; Aug
               dfb  30h  ; Sep
               dfb  31h  ; Oct
               dfb  30h  ; Nov
               dfb  31h  ; Dec


;BUFFER_ASCII CONVERT BCD IN BUFFER START $30 - $35
;          ENTRY: BCD NUMBER IN TRANSMIT BUFFER
;          EXIT: ASCII CODE IN TRANSMIT BUFFER

BUFFER_ASCII:       MOV    R7,#06H
                    MOV    R0,#30H
  NEXT_ASCII:       MOV    A,@R0
                    LCALL  BCD_ASCII
                    MOV    @R0,A
                    INC    R0
                    DJNZ   R7,NEXT_ASCII
                    RET

;SEND_BUFFER SEND ASCII CODE IN TRANSMIT BUFFER TO SERIAL PORT
;          THEN DATA SEPERATOR /32/32
;          ENTRY: ASCII CODE IN TRANSMIT BUFFER
;          EXIT: NO

SEND_BUFFER:        MOV     R7,#06H
                    MOV     R0,#35H
  NEXT_BYTE:        MOV     A,@R0
                    LCALL   SEND        ;SEND OUT TO SERIAL PORT
                    DEC     R0
                    DJNZ    R7,NEXT_BYTE
                    lcall   cr_lf

                    RET


;service_int0 activates three output port manually

service_int0:       push acc
```

```
                    setb 0fh
                    mov  a,output
                    cpl  a
                    mov  p1,a
                    pop  acc
                    reti



;COMPARE TWO 16 BIT BINARY NUMBER
;         ENTRY: R0 FIRST OPERAND
;                R1 SECOND OPERAND
;         EXIT: C=1 FIRST OPERAND < SECOND OPERAND
;               C=0 FIRST OPERAND >= SECOND OPERAND

COMPARE:      CLR     C              ;
              MOV     A,@R0
              SUBB    A,@R1
              INC     R0
              INC     R1
              MOV     A,@R0
              SUBB    A,@R1
              RET
;
;#####################################################################
;#                                  MAIN PROGRAM
;#####################################################################

CURRENT_TIME_L: EQU 40H
CURRENT_TIME_H: EQU 41H
CURRENT_PROG_L: EQU 42H
CURRENT_PROG_H: EQU 43H

COUNT_L:        EQU 92h     ;7AH       ; <--- find adjust in case of
                            ; frequency is not a 3.579545 MHz
COUNT_H:        EQU 8BH     ; interrupt every 3.579545 MHz/12*(65536
                            ; = 10 Hz


MAIN_TIMER:
              MOV SP,#60H                ;  stack area start $60-$
              mov   light,#00h
              mov   p1,#11111111b        ; off output
              clr   serial               ;
              clr   blink
              mov   sec100,#00h
              mov   econo,#00h
              mov   tl0adj,#92h          ; for adjustment later vi
              mov  a,warm_code
              cjne a,#"%",cold_boot
              sjmp  warm_boot

   cold_boot:
              mov  warm_code,#"%"
              mov  pgm_buffer,#eof        ; blank pgm at power up

              MOV  SEC,#00D
              MOV  MIN,#30H
              MOV  HOUR,#12H
              MOV  DAY,#29H
              MOV  MONTH,#04H
              MOV  YEAR,#37H
              mov  output,#00h
   warm_boot:
```

```
                        setb 0fh
                        mov  a,output
                        cpl  a
                        mov  p1,a
                        pop  acc
                        reti



    ;COMPARE TWO 16 BIT BINARY NUMBER
    ;         ENTRY: R0 FIRST OPERAND
    ;                R1 SECOND OPERAND
    ;         EXIT: C=1 FIRST OPERAND < SECOND OPERAND
    ;               C=0 FIRST OPERAND >= SECOND OPERAND

    COMPARE:      CLR     C              ;
                  MOV     A,@R0
                  SUBB    A,@R1
                  INC     R0
                  INC     R1
                  MOV     A,@R0
                  SUBB    A,@R1
                  RET
    ;
    ;#################################################################
    ;#                                MAIN PROGRAM
    ;#################################################################

CURRENT_TIME_L: EQU 40H
CURRENT_TIME_H: EQU 41H
CURRENT_PROG_L: EQU 42H
CURRENT_PROG_H: EQU 43H

COUNT_L:        EQU 92h     ;7AH       ; <--- find adjust in case of
                            ; frequency is not a 3.579545 MHz
COUNT_H:        EQU 8BH     ; interrupt every 3.579545 MHz/12*(65536
                            ; = 10 Hz


MAIN_TIMER:
                MOV SP,#60H                 ;  stack area start $60-$
                mov   light,#00h
                mov   p1,#11111111b         ; off output
                clr   serial                ;
                clr   blink
                mov   sec100,#00h
                mov   econo,#00h
                mov   tl0adj,#92h           ; for adjustment later vi
                mov  a,warm_code
                cjne a,#"%",cold_boot
                sjmp  warm_boot

    cold_boot:
                mov  warm_code,#"%"
                mov  pgm_buffer,#eof         ; blank pgm at power up

                MOV  SEC,#00D
                MOV  MIN,#30H
                MOV  HOUR,#12H
                MOV  DAY,#29H
                MOV  MONTH,#04H
                MOV  YEAR,#37H
                mov  output,#00h
    warm_boot:
```

```
                SETB   EA                      ; ENABLE ALL INTERRUPT
                SETB   ET0                     ; ENABLE TIMER 0 INTERRUPT
                setb   ex0                     ; enable external interrup
                MOV    TMOD,#00000001B         ; SET MODE 1 T0 (16 BIT DI
                LCALL  INIT
                SETB   PT0                     ; TIMER 0 HIGHEST PIORITY
                MOV    DPTR,#TITLE
                LCALL  SEND_STRING
                mov    th0,#count_h            ;start timer
                mov    tl0,#count_l
                setb   tr0                     ;start timer

        TEST10: jnb    ri,test10               ; polls receive buffer and
                clr    ri
                mov    a,sbuf

                cjne   a,#"A",skip_modem_init     ; SKIP MODEM INITIA
                lcall  recive                    ; SEARCH "AT" STRING
                cjne   a,#"T",skip_modem_init

    dummy_init: lcall  recive
                cjne   a,#cr,dummy_init          ; UNTIL END OF STRIN
                sjmp   test10

skip_modem_init:

                orl    a,#20h                  ; accept upper & lower
                cjne   a,#"o",com10
                sjmp   off_serial

        com10:  cjne   a,#"t",com11
                sjmp   transmit_time

        com11:  cjne   a,#"z",com12
                sjmp   adjust_clock

        com12:  cjne   a,#"e",com13
                sjmp   set_time

        com13:  cjne   a,#"d",com14
                sjmp    set_date

        com14:  cjne   a,#"?",com15
                ljmp    print_com

        com15:  cjne   a,#"s",com16
                ljmp    speed_up

        com16:  cjne   a,#"p",com17
                ljmp    program_time


        com17:  cjne    a,#"r",com18
                ljmp    read_pgm

        com18:  cjne    a,#"c",com19
                ljmp    clear_io

        com19:  cjne    a,#"/",com20
                ljmp    once

        com20:
                lcall   send_prompt
                LJMP   TEST10
```

```
                SETB   EA                      ; ENABLE ALL INTERRUPT
                SETB   ET0                     ; ENABLE TIMER 0 INTERRUPT
                setb   ex0                     ; enable external interrup
                MOV    TMOD,#00000001B         ; SET MODE 1 T0 (16 BIT DI
                LCALL  INIT
                SETB   PT0                     ; TIMER 0 HIGHEST PIORITY
                MOV    DPTR,#TITLE
                LCALL  SEND_STRING
                mov    th0,#count_h            ;start timer
                mov    tl0,#count_l
                setb   tr0                     ;start timer

        TEST10: jnb    ri,test10               ; polls receive buffer and
                clr    ri
                mov    a,sbuf

                cjne   a,#"A",skip_modem_init     ; SKIP MODEM INITIA
                lcall  recive                    ; SEARCH "AT" STRING
                cjne   a,#"T",skip_modem_init

    dummy_init: lcall  recive
                cjne   a,#cr,dummy_init          ; UNTIL END OF STRIN
                sjmp   test10

skip_modem_init:

                orl    a,#20h                  ; accept upper & lower
                cjne   a,#"o",com10
                sjmp   off_serial

        com10:  cjne   a,#"t",com11
                sjmp   transmit_time

        com11:  cjne   a,#"z",com12
                sjmp   adjust_clock

        com12:  cjne   a,#"e",com13
                sjmp   set_time

        com13:  cjne   a,#"d",com14
                sjmp    set_date

        com14:  cjne   a,#"?",com15
                ljmp    print_com

        com15:  cjne   a,#"s",com16
                ljmp    speed_up

        com16:  cjne   a,#"p",com17
                ljmp    program_time


        com17:  cjne    a,#"r",com18
                ljmp    read_pgm

        com18:  cjne    a,#"c",com19
                ljmp    clear_io

        com19:  cjne    a,#"/",com20
                ljmp    once

        com20:
                lcall   send_prompt
                LJMP   TEST10
```

```
RUN2:       MOV TL0,#COUNT_L
            MOV TH0,#COUNT_H
            SETB TR0            ; START TIMER
            RET


    ;off_serial stop transmit current date and time to terminal
off_serial:
            clr   serial
            LCALL SEND_PROMPT
            sjmp test10

    ;transmit_time start send current date and time to terminal
transmit_time: setb serial
            sjmp  test10

    ;adjust_clock change low byte of timer 0

 adjust_clock: lcall cr_lf
            mov  a,tl0adj
            lcall send_ascii
            lcall cr_lf
            mov  a,#"?"
            lcall send
            lcall get_byte
            mov  tl0adj,a
            lcall send_prompt
            ljmp  test10

    ;set_time enter current time

 set_time:      mov    dptr,#time_set
            lcall  send_string
            lcall get_byte
            mov  hour,a             ; save hour
            lcall get_byte
            mov  min,a              ; save min
            lcall get_byte
            mov  sec,a              ; save sec
            lcall  send_time
            lcall  send_prompt
            ljmp   test10

    ;set_date enter current date

 set_date:      mov    dptr,#date_set
            lcall  send_string
            lcall  get_byte
            mov    day,a
            lcall  get_byte
            mov    month,a
            lcall  get_byte
            mov    year,a
            lcall  send_time
            lcall  send_prompt
            ljmp  test10


    ;print_com shows command listing

 print_com:     mov    dptr,#command
            lcall  send_string
            lcall  send_prompt
```

```
RUN2:       MOV TL0,#COUNT_L
            MOV TH0,#COUNT_H
            SETB TR0            ; START TIMER
            RET

    ;off_serial stop transmit current date and time to terminal
off_serial:
            clr   serial
            LCALL SEND_PROMPT
            sjmp test10

    ;transmit_time start send current date and time to terminal
transmit_time: setb serial
            sjmp  test10

    ;adjust_clock change low byte of timer 0

 adjust_clock: lcall cr_lf
            mov  a,tl0adj
            lcall send_ascii
            lcall cr_lf
            mov  a,#"?"
            lcall send
            lcall get_byte
            mov  tl0adj,a
            lcall send_prompt
            ljmp  test10

    ;set_time enter current time

 set_time:      mov    dptr,#time_set
            lcall  send_string
            lcall get_byte
            mov  hour,a             ; save hour
            lcall get_byte
            mov  min,a              ; save min
            lcall get_byte
            mov  sec,a              ; save sec
            lcall  send_time
            lcall  send_prompt
            ljmp   test10

    ;set_date enter current date

 set_date:      mov    dptr,#date_set
            lcall  send_string
            lcall  get_byte
            mov    day,a
            lcall  get_byte
            mov    month,a
            lcall  get_byte
            mov    year,a
            lcall  send_time
            lcall  send_prompt
            ljmp  test10


    ;print_com shows command listing

 print_com:     mov    dptr,#command
            lcall  send_string
            lcall  send_prompt
```

```
                ljmp    test10

    ;speed_up simulate output at high speed

 speed_up:      cpl     sim
                ljmp    test10


    ;program_time wait file from terminal write to on chip ram
    ;             start $30-$63 60 bytes or 20 line accepted
    ;             program format:
    ;             :01 05 2030 1 1 0 1 0 0 1 1crlf
    ;             :02 06 2120 1 0 0 0 1 1 0 1crlf
    ;             :00crlf
    ;     where
    ;             : initiate start of line
    ;              01 record 1,2,3,...
    ;                 05 date
    ;                    2030 time
    ;                         1 1 0 1 0 0 1 1 digital output
    ;                                         crlf end of line flag
    ;              00 end of record flag
    ;
    ;     maximum program storage provides 7 days, i.e. each day req
    ;     byte, thus total byte is 28 bytes from $30 - $50

 program_time:  clr   tr0              ; stop timer before so after pr
                mov   dptr,#program    ; loading is finished actual ti
                lcall send_string      ; to be adjusted again.
                lcall  cr_lf
                mov    r0,#pgm_buffer

 pgm1:          lcall  recive
                cjne   a,#":",pgm1


                lcall  get_byte         ; skip line


                cjne   a,#00h,pgm2      ; end of file detected
                sjmp   stop_pgm

pgm2:
                lcall  recive           ; skip space
                lcall  get_byte         ; get date byte
                mov    @r0,a            ; save date

                lcall  recive           ; skip space
                inc    r0

                lcall  get_byte         ; get hour byte
                mov    @r0,a            ; save hour

                inc    r0
                lcall  get_byte
                mov    @r0,a            ; save min


                inc    r0

                mov    r7,#08d          ; get control I/O bit

 pgm3:          lcall  recive           ; skip space
                lcall  recive
```

```
                ljmp    test10

    ;speed_up simulate output at high speed

 speed_up:      cpl     sim
                ljmp    test10


    ;program_time wait file from terminal write to on chip ram
    ;               start $30-$63 60 bytes or 20 line accepted
    ;               program format:
    ;               :01 05 2030 1 1 0 1 0 0 1 1crlf
    ;               :02 06 2120 1 0 0 0 1 1 0 1crlf
    ;               :00crlf
    ;     where
    ;               : initiate start of line
    ;                01 record 1,2,3,...  .
    ;                   05 date
    ;                      2030 time
    ;                           1 1 0 1 0 0 1 1 digital output
    ;                                           crlf end of line flag
    ;                00 end of record flag
    ;
    ;     maximum program storage provides 7 days, i.e. each day req
    ;     byte, thus total byte is 28 bytes from $30 - $50

 program_time:  clr     tr0             ; stop timer before so after pr
                mov     dptr,#program   ; loading is finished actual ti
                lcall   send_string     ; to be adjusted again.
                lcall   cr_lf
                mov     r0,#pgm_buffer

 pgm1:          lcall   recive
                cjne    a,#":",pgm1


                lcall   get_byte        ; skip line


                cjne    a,#00h,pgm2     ; end of file detected
                sjmp    stop_pgm

pgm2:
                lcall   recive          ; skip space
                lcall   get_byte        ; get date byte
                mov     @r0,a           ; save date

                lcall   recive          ; skip space
                inc     r0

                lcall   get_byte        ; get hour byte
                mov     @r0,a           ; save hour

                inc     r0
                lcall   get_byte
                mov     @r0,a           ; save min


                inc     r0

                mov     r7,#08d         ; get control I/O bit

 pgm3:          lcall   recive          ; skip space
                lcall   recive
```

```
            ;   mov     b,a
            ;   lcall   SEND_XOFF          ; pause string sending from t
            ;   mov     a,b
                cjne    a,#"0",pgm4
                clr     c
                sjmp    pgm5

pgm4:           setb    c
pgm5:           mov     a,@r0
                rlc     a
                mov     @r0,a
              ; lcall   SEND_XON           ; ready to receive again
                djnz    r7,pgm3            ; loop 8 times

pgm6:           lcall   recive
                cjne    a,#lf,pgm6

                inc     r0
                sjmp    pgm1               ; read next line

stop_pgm:       mov     @r0,#eof           ; end of file
                lcall   recive
                cjne    a,#lf,stop_pgm
                mov     dptr,#finish_pgm
                lcall   send_string
                lcall   send_prompt

                setb    tr0                ; resume timer interrupt again
                ljmp    test10


    ;           read_pgm read internal program for checking
    ;           reteive internal program send to terminal for checking
    ;           or changing current settings

read_pgm:       lcall   cr_lf
                mov     r0,#pgm_buffer
read2:          mov     a,@r0
                cjne    a,#eof,read3   ; until eos (0FH) found
                mov     dptr,#end_of_file
                lcall   send_string
                sjmp    read4

read3:          lcall   send_ascii     ; send day
                lcall   space
                inc     r0
                mov     a,@r0          ; get hour
                lcall   send_ascii     ; send hour
                mov     a,#":"
                lcall   send
                inc     r0
                mov     a,@r0
                lcall   send_ascii     ; send min
                lcall   space

                mov     dptr,#output_prompt
                lcall   send_string
                inc     r0
                mov     a,@r0

                mov     r7,#08h
read_io:        rlc     a
                mov     b,a
                jc      send_on
```

```
              ;   mov    b,a
              ;   lcall  SEND_XOFF          ; pause string sending from t
              ;   mov    a,b
                  cjne   a,#"0",pgm4
                  clr    c
                  sjmp   pgm5

    pgm4:         setb   c
    pgm5:         mov    a,@r0
                  rlc    a
                  mov    @r0,a
                ; lcall  SEND_XON           ; ready to receive again
                  djnz   r7,pgm3            ; loop 8 times

    pgm6:         lcall  recive
                  cjne   a,#1f,pgm6

                  inc    r0
                  sjmp   pgm1               ; read next line

    stop_pgm:     mov    @r0,#eof           ; end of file
                  lcall  recive
                  cjne   a,#1f,stop_pgm
                  mov    dptr,#finish_pgm
                  lcall  send_string
                  lcall  send_prompt

                  setb   tr0                ; resume timer interrupt again
                  ljmp   test10


        ;         read_pgm read internal program for checking
        ;         reteive internal program send to terminal for checking
        ;         or changing current settings

    read_pgm:     lcall  cr_lf
                  mov    r0,#pgm_buffer
    read2:        mov    a,@r0
                  cjne   a,#eof,read3   ; until eos (0FH) found
                  mov    dptr,#end_of_file
                  lcall  send_string
                  sjmp   read4

    read3:        lcall  send_ascii     ; send day
                  lcall  space
                  inc    r0
                  mov    a,@r0          ; get hour
                  lcall  send_ascii     ; send hour
                  mov    a,#":"
                  lcall  send
                  inc    r0
                  mov    a,@r0
                  lcall  send_ascii     ; send min
                  lcall  space

                  mov    dptr,#output_prompt
                  lcall  send_string
                  inc    r0
                  mov    a,@r0

                  mov    r7,#08h
    read_io:      rlc    a
                  mov    b,a
                  jc     send_on
```

```
                mov     a,#"-"
                lcall   send
                sjmp    next_io_bit

   send_on:     mov     a,#"O"
                lcall   send

next_io_bit:
                lcall   space
                mov     a,b
                djnz    r7,read_io

                lcall   cr_lf
                inc     r0
                sjmp    read2

     read4:
                lcall   send_prompt
                ljmp    test10


       ;clear_io clear all 8 bit p1.0-p1.7

   clear_io:    mov     output,#00000000b
                mov     p1,#11111111b       ; clear port1 also
                lcall   send_time           ; display results
                lcall   send_prompt
                ljmp    test10

        ;once read current time once

   once:        lcall   send_time
                lcall   send_prompt
                ljmp    test10




       ;SEND_STRING SEND STRING CONSTANT TO TERMINAL
       ;            ENTRY: DPTR
       ;            EXIT: FOUND EOS
       ;
     SEND_STRING:          CLR   A
                           MOVC  A,@A+DPTR
                           CJNE  A,#EOS,SEND_STRING1
                           RET

    SEND_STRING1:          PUSH  DPL
                           PUSH  DPH
                           LCALL SEND
                           POP   DPH
                           POP   DPL
                           INC   DPTR
                           SJMP  SEND_STRING
       ;*************************************************************
       ;           STRING CONSTANT AREA
       ;*************************************************************

   TITLE:     DFB  cr,lf,"Miniature Realtime Controller RTC-3707",CR,LF
              DFB         "Designed by Wichit Sirichote"
              DFB  CR,LF
  PROMPT:     DFB  "& ",EOS
 command:   dfb cr,lf
            dfb "o   return prompt",cr,lf
```

```
                mov     a,#"-"
                lcall   send
                sjmp    next_io_bit

    send_on:    mov     a,#"O"
                lcall   send

next_io_bit:
                lcall   space
                mov     a,b
                djnz    r7,read_io

                lcall   cr_lf
                inc     r0
                sjmp    read2

      read4:
                lcall   send_prompt
                ljmp    test10


        ;clear_io clear all 8 bit p1.0-p1.7

    clear_io:   mov     output,#00000000b
                mov     p1,#11111111b       ; clear port1 also
                lcall   send_time           ; display results
                lcall   send_prompt
                ljmp    test10

         ;once read current time once

    once:       lcall   send_time
                lcall   send_prompt
                ljmp    test10



        ;SEND_STRING SEND STRING CONSTANT TO TERMINAL
        ;            ENTRY: DPTR
        ;            EXIT: FOUND EOS
        ;
      SEND_STRING:          CLR   A
                            MOVC  A,@A+DPTR
                            CJNE  A,#EOS,SEND_STRING1
                            RET

     SEND_STRING1:          PUSH  DPL
                            PUSH  DPH
                            LCALL SEND
                            POP   DPH
                            POP   DPL
                            INC   DPTR
                            SJMP  SEND_STRING
        ;*****************************************************************
        ;           STRING CONSTANT AREA
        ;*****************************************************************

   TITLE:     DFB  cr,lf,"Miniature Realtime Controller RTC-3707",CR,LF
              DFB         "Designed by Wichit Sirichote"
              DFB  CR,LF
  PROMPT:     DFB  "& ",EOS
 command:  dfb cr,lf
           dfb "o   return prompt",cr,lf
```

```
            dfb "t   send time",cr,lf
            dfb "z   adj. time base",cr,lf
            dfb "e   enter current time",cr,lf
            dfb "d   enter current date",cr,lf
            dfb "s   stroke setting",cr,lf
            dfb "p   program timer",cr,lf
            dfb "r   read program",cr,lf
            dfb "c   clear I/O",cr,lf
            dfb "/   send time once",cr,lf
            dfb "?   help command",cr,lf
            dfb eos

finish_pgm: dfb "Program stored",cr,lf,eos
  program:  dfb cr,lf,"Ready to receive program",eos
output_prompt:
            dfb "OUTPUT [1..8] ",eos
 time_set:  dfb cr,lf,"Enter current time > ",eos
 date_set:  dfb cr,lf,"Enter current date > ",eos
end_of_file: dfb "EOF",eos
    WHAT?:  DFB  "WHAT?",CR,LF,EOS

                        END
```

```
                dfb "t   send time",cr,lf
                dfb "z   adj. time base",cr,lf
                dfb "e   enter current time",cr,lf
                dfb "d   enter current date",cr,lf
                dfb "s   stroke setting",cr,lf
                dfb "p   program timer",cr,lf
                dfb "r   read program",cr,lf
                dfb "c   clear I/O",cr,lf
                dfb "/   send time once",cr,lf
                dfb "?   help command",cr,lf
                dfb eos

  finish_pgm: dfb "Program stored",cr,lf,eos
    program:  dfb cr,lf,"Ready to receive program",eos
output_prompt:
                dfb "OUTPUT [1..8] ",eos
    time_set: dfb cr,lf,"Enter current time > ",eos
    date_set: dfb cr,lf,"Enter current date > ",eos
 end_of_file: dfb "EOF",eos
       WHAT?: DFB  "WHAT?",CR,LF,EOS

                                END
```

DATA SHEET

DATA SHEET

MC145026
MC145027
MC145028
SC41343
SC41344

2

Encoder and Decoder Pairs
CMOS

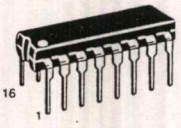
These devices are designed to be used as encoder/decoder pairs in remote control applications.

The MC145026 encodes nine lines of information and serially sends this information upon receipt of a transmit enable ($\overline{TE}$) signal. The nine lines may be encoded with trinary data (low, high, or open) or binary data (low or high). The words are transmitted twice per encoding sequence to increase security.

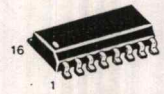
The MC145027 decoder receives the serial stream and interprets five of the trinary digits as an address code. Thus, 243 addresses are possible. If binary data is used at the encoder, 32 addresses are possible. The remaining serial information is interpreted as four bits of binary data. The valid transmission (VT) output goes high on the MC145027 when two conditions are met. First, two addresses must be consecutively received (in one encoding sequence) which both match the local address. Second, the 4 bits of data must match the last valid data received. The active VT indicates that the information at the Data output pins has been updated.

The MC145028 decoder treats all nine trinary digits as an address which allows 19,683 codes. If binary data is encoded, 512 codes are possible. The VT output goes high on the MC145028 when two addresses are consecutively received (in one encoding sequence) which both match the local address.

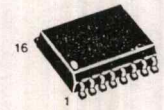
- Operating Temperature Range: – 40 to + 85°C
- Very–Low Standby Current for the Encoder: 300 nA Maximum @ 25°C
- Interfaces with RF, Ultrasonic, or Infrared Modulators and Demodulators
- RC Oscillator, No Crystal Required
- High External Component Tolerance; Can Use ± 5% Components
- Internal Power–On Reset Forces All Decoder Outputs Low
- For Infrared Applications, See Applications Notes AN1016 and AN1126
- Operating Voltage Range: MC145026 = 2.5 to 18 V*
  MC145027, MC145028 = 4.5 to 18 V
- Low–Voltage Versions Available:
  SC41343 = 2.8 to 10 V Version of the MC145027
  SC41344 = 2.8 to 10 V Version of the MC145028



P SUFFIX
PLASTIC DIP
CASE 648



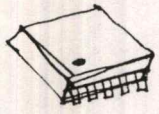
D SUFFIX
SOG PACKAGE
CASE 751B



DW SUFFIX
SOG PACKAGE
CASE 751G

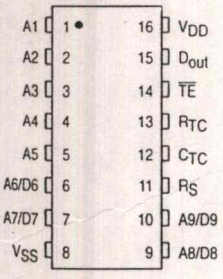
ORDERING INFORMATION

MC145026P	Plastic DIP
MC145026D	SOG Package
MC145027P, SC41343P	Plastic DIP
MC145027DW, SC41343DW	SOG Package
MC145028P, SC41344P	Plastic DIP
MC145028DW, SC41344DW	SOG Package

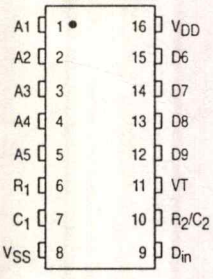


PIN ASSIGNMENTS

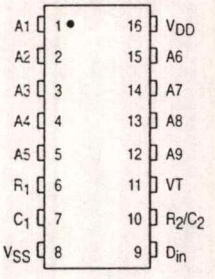
MC145026 ENCODER



MC145027/SC41343 DECODERS



MC145028/SC41344 DECODERS



* All MC145026 devices manufactured after date code 9314 or 314 are guaranteed over this wider voltage range. All previous designs using the low–voltage SC41342 should convert to the MC145026, which is a drop–in replacement. The SC41342 part number will be discontinued.

MC145026
MC145027
MC145028
SC41343
SC41344

2

Encoder and Decoder Pairs
CMOS

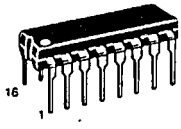
These devices are designed to be used as encoder/decoder pairs in remote control applications.

The MC145026 encodes nine lines of information and serially sends this information upon receipt of a transmit enable ($\overline{TE}$) signal. The nine lines may be encoded with trinary data (low, high, or open) or binary data (low or high). The words are transmitted twice per encoding sequence to increase security.

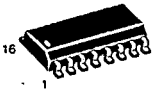
The MC145027 decoder receives the serial stream and interprets five of the trinary digits as an address code. Thus, 243 addresses are possible. If binary data is used at the encoder, 32 addresses are possible. The remaining serial information is interpreted as four bits of binary data. The valid transmission (VT) output goes high on the MC145027 when two conditions are met. First, two addresses must be consecutively received (in one encoding sequence) which both match the local address. Second, the 4 bits of data must match the last valid data received. The active VT indicates that the information at the Data output pins has been updated.

The MC145028 decoder treats all nine trinary digits as an address which allows 19,683 codes. If binary data is encoded, 512 codes are possible. The VT output goes high on the MC145028 when two addresses are consecutively received (in one encoding sequence) which both match the local address.

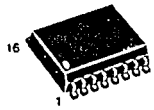
- Operating Temperature Range: – 40 to + 85°C
- Very–Low Standby Current for the Encoder: 300 nA Maximum @ 25°C
- Interfaces with RF, Ultrasonic, or Infrared Modulators and Demodulators
- RC Oscillator, No Crystal Required
- High External Component Tolerance; Can Use ± 5% Components
- Internal Power–On Reset Forces All Decoder Outputs Low
- For Infrared Applications, See Applications Notes AN1016 and AN1126
- Operating Voltage Range: MC145026 = 2.5 to 18 V*
  MC145027, MC145028 = 4.5 to 18 V
- Low–Voltage Versions Available:
  SC41343 = 2.8 to 10 V Version of the MC145027
  SC41344 = 2.8 to 10 V Version of the MC145028



P SUFFIX
PLASTIC DIP
CASE 648



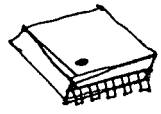
D SUFFIX
SOG PACKAGE
CASE 751B



DW SUFFIX
SOG PACKAGE
CASE 751G

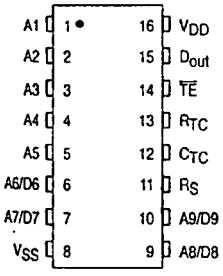
ORDERING INFORMATION

MC145026P	Plastic DIP
MC145026D	SOG Package
MC145027P, SC41343P	Plastic DIP
MC145027DW, SC41343DW	SOG Package
MC145028P, SC41344P	Plastic DIP
MC145028DW, SC41344DW	SOG Package

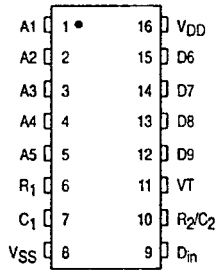


PIN ASSIGNMENTS

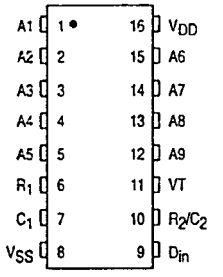
MC145026
ENCODER



MC145027/SC41343
DECODERS



MC145028/SC41344
DECODERS



*All MC145026 devices manufactured after date code 9314 or 314 are guaranteed over this wider voltage range. All previous designs using the low–voltage SC41342 should convert to the MC145026, which is a drop–in replacement. The SC41342 part number will be discontinued.

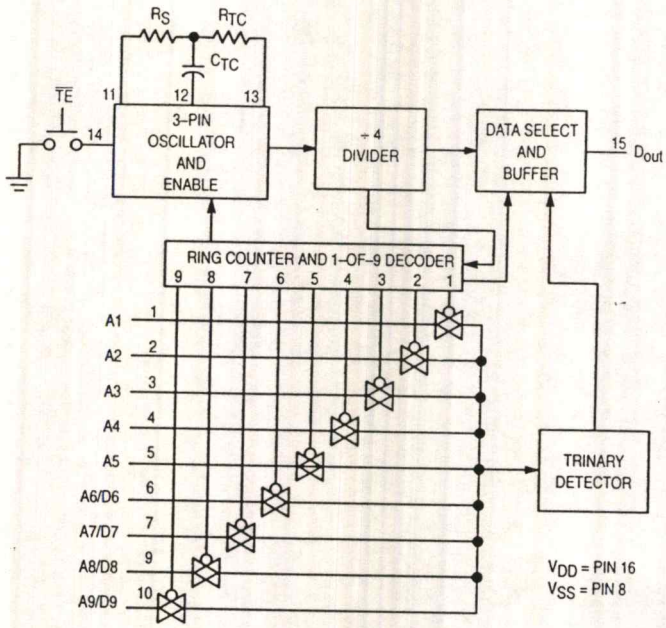


Figure 1. MC145026 Encoder Block Diagram

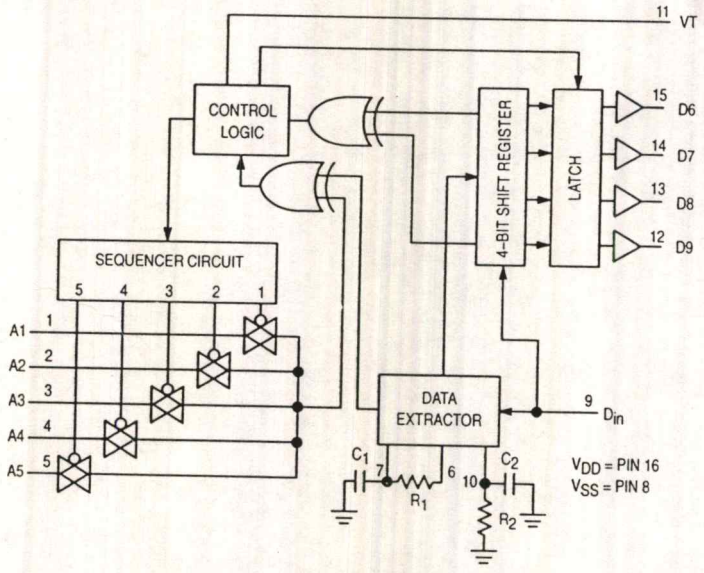


Figure 2. MC145027 Decoder Block Diagram

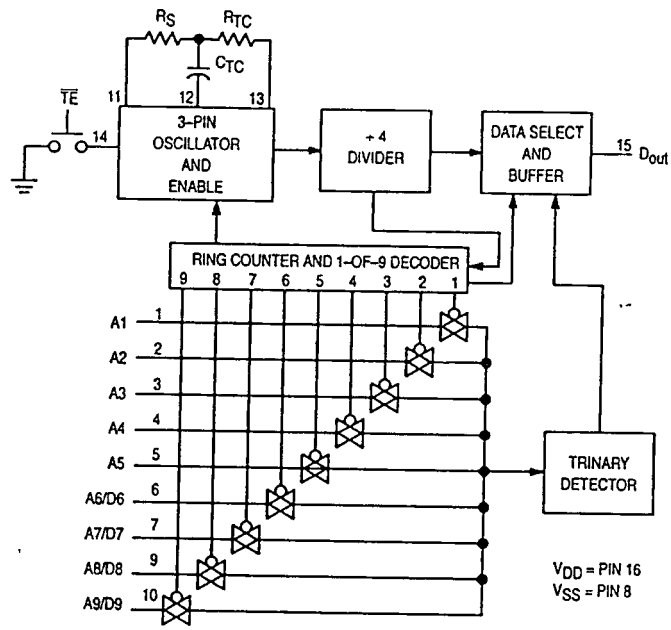


Figure 1. MC145026 Encoder Block Diagram

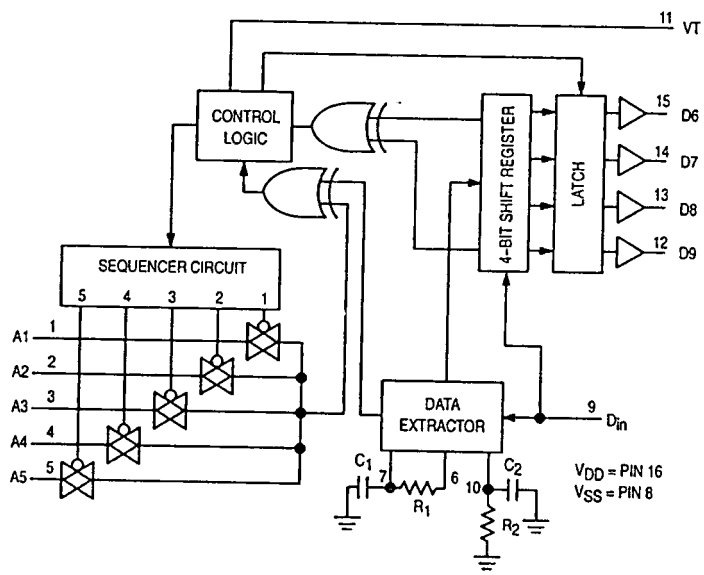


Figure 2. MC145027 Decoder Block Diagram

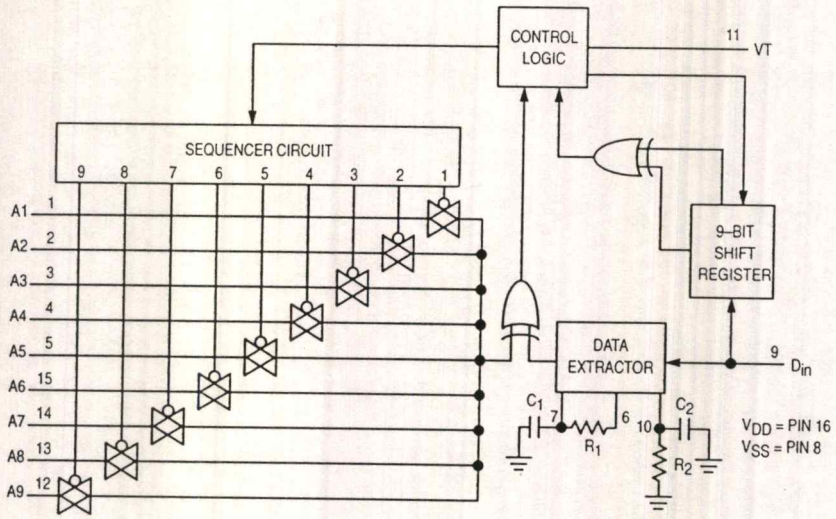


Figure 3. MC145028 Decoder Block Diagram

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

Rating	Symbol	Value	Unit
V_{DD}	DC Supply Voltage (except SC41343, SC41344)	– 0.5 to + 18	V
V_{DD}	DC Supply Voltage (SC41343, SC41344 only)	– 0.5 to + 10	V
V_{in}	DC Input Voltage	– 0.5 to V_{DD} + 0.5	V
V_{out}	DC Output Voltage	– 0.5 to V_{DD} + 0.5	V
I_{in}	DC Input Current, per Pin	± 10	mA
I_{out}	DC Output Current, per Pin	± 10	mA
P_D	Power Dissipation, per Package	500	mW
T_{stg}	Storage Temperature	– 65 to + 150	°C
T_L	Lead Temperature, 1 mm from Case for 10 Seconds	260	°C

* Maximum Ratings are those values beyond which damage to the device may occur. Functional operation should be restricted to the limits in the Electrical Characteristics tables or Pin Descriptions section.

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high–impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.

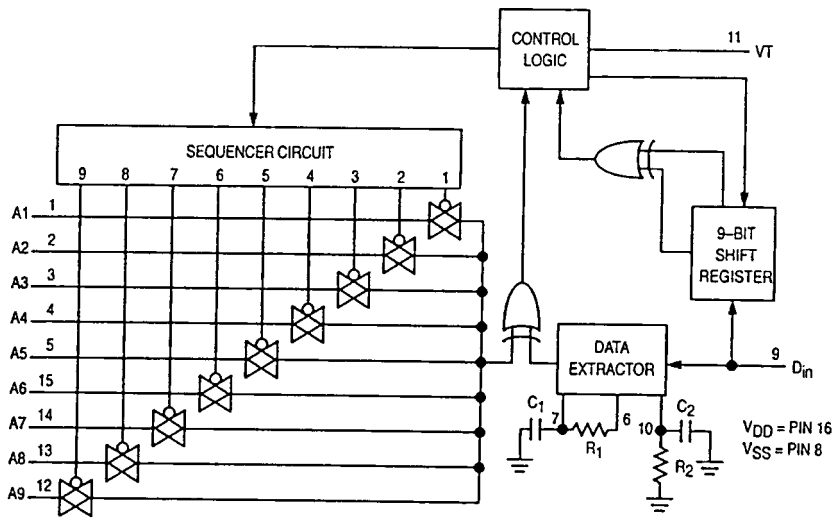


Figure 3. MC145028 Decoder Block Diagram

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

Rating	Symbol	Value	Unit
V_{DD}	DC Supply Voltage (except SC41343, SC41344)	− 0.5 to + 18	V
V_{DD}	DC Supply Voltage (SC41343, SC41344 only)	− 0.5 to + 10	V
V_{in}	DC Input Voltage	− 0.5 to V_{DD} + 0.5	V
V_{out}	DC Output Voltage	− 0.5 to V_{DD} + 0.5	V
I_{in}	DC Input Current, per Pin	± 10	mA
I_{out}	DC Output Current, per Pin	± 10	mA
P_D	Power Dissipation, per Package	500	mW
T_{stg}	Storage Temperature	− 65 to + 150	°C
T_L	Lead Temperature, 1 mm from Case for 10 Seconds	260	°C

* Maximum Ratings are those values beyond which damage to the device may occur. Functional operation should be restricted to the limits in the Electrical Characteristics tables or Pin Descriptions section.

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.

ELECTRICAL CHARACTERISTICS — MC145026*, MC145027, and MC145028 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = V_{DD} or 0)	5.0	—	0.05	—	0.05	—	0.05	V
		10	—	0.05	—	0.05	—	0.05	
		15	—	0.05	—	0.05	—	0.05	
V_{OH}	High–Level Output Voltage (V_{in} = 0 or V_{DD})	5.0	4.95	—	4.95	—	4.95	—	V
		10	9.95	—	9.95	—	9.95	—	
		15	14.95	—	14.95	—	14.95	—	
V_{IL}	Low–Level Input Voltage								V
	(V_{out} = 4.5 or 0.5 V)	5.0	—	1.5	—	1.5	—	1.5	
	(V_{out} = 9.0 or 1.0 V)	10	—	3.0	—	3.0	—	3.0	
	(V_{out} = 13.5 or 1.5 V)	15	—	4.0	—	4.0	—	4.0	
V_{IH}	High–Level Input Voltage								V
	(V_{out} = 0.5 or 4.5 V)	5.0	3.5	—	3.5	—	3.5	—	
	(V_{out} = 1.0 or 9.0 V)	10	7.0	—	7.0	—	7.0	—	
	(V_{out} = 1.5 or 13.5 V)	15	11	—	11	—	11	—	
I_{OH}	High–Level Output Current								mA
	(V_{out} = 2.5 V)	5.0	−2.5	—	−2.1	—	−1.7	—	
	(V_{out} = 4.6 V)	5.0	−0.52	—	−0.44	—	−0.36	—	
	(V_{out} = 9.5 V)	10	−1.3	—	−1.1	—	−0.9	—	
	(V_{out} = 13.5 V)	15	−3.6	—	−3.0	—	−2.4	—	
I_{OL}	Low–Level Output Current								mA
	(V_{out} = 0.4 V)	5.0	0.52	—	0.44	—	0.36	—	
	(V_{out} = 0.5 V)	10	1.3	—	1.1	—	0.9	—	
	(V_{out} = 1.5 V)	15	3.6	—	3.0	—	2.4	—	
I_{in}	Input Current — $\overline{TE}$	5.0	—	—	3.0	11	—	—	µA
	(MC145026, Pull–Up Device)	10	—	—	16	60	—	—	
		15	—	—	35	120	—	—	
I_{in}	Input Current R_S (MC145026), D_{in} (MC145027, MC145028)	15	—	±0.3	—	±0.3	—	±1.0	µA
I_{in}	Input Current								µA
	A1 – A5, A6/D6 – A9/D9 (MC145026),	5.0	—	—	—	±110	—	—	
	A1 – A5 (MC145027),	10	—	—	—	±500	—	—	
	A1 – A9 (MC145028)	15	—	—	—	±1000	—	—	
C_{in}	Input Capacitance (V_{in} = 0)	—	—	—	—	7.5	—	—	pF
I_{DD}	Quiescent Current — MC145026	5.0	—	—	—	0.1	—	—	µA
		10	—	—	—	0.2	—	—	
		15	—	—	—	0.3	—	—	
I_{DD}	Quiescent Current — MC145027, MC145028	5.0	—	—	—	50	—	—	µA
		10	—	—	—	100	—	—	
		15	—	—	—	150	—	—	
I_{dd}	Dynamic Supply Current — MC145026	5.0	—	—	—	200	—	—	µA
	(f_c = 20 kHz)	10	—	—	—	400	—	—	
		15	—	—	—	600	—	—	
I_{dd}	Dynamic Supply Current — MC145027, MC145028	5.0	—	—	—	400	—	—	µA
	(f_c = 20 kHz)	10	—	—	—	800	—	—	
		15	—	—	—	1200	—	—	

* Also see next Electrical Characteristics table for 2.5 V specifications.

ELECTRICAL CHARACTERISTICS — MC145026*, MC145027, and MC145028 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = V_{DD} or 0)	5.0	—	0.05	—	0.05	—	0.05	V
		10	—	0.05	—	0.05	—	0.05	
		15	—	0.05	—	0.05	—	0.05	
V_{OH}	High–Level Output Voltage (V_{in} = 0 or V_{DD})	5.0	4.95	—	4.95	—	4.95	—	V
		10	9.95	—	9.95	—	9.95	—	
		15	14.95	—	14.95	—	14.95	—	
V_{IL}	Low–Level Input Voltage								V
	(V_{out} = 4.5 or 0.5 V)	5.0	—	1.5	—	1.5	—	1.5	
	(V_{out} = 9.0 or 1.0 V)	10	—	3.0	—	3.0	—	3.0	
	(V_{out} = 13.5 or 1.5 V)	15	—	4.0	—	4.0	—	4.0	
V_{IH}	High–Level Input Voltage								V
	(V_{out} = 0.5 or 4.5 V)	5.0	3.5	—	3.5	—	3.5	—	
	(V_{out} = 1.0 or 9.0 V)	10	7.0	—	7.0	—	7.0	—	
	(V_{out} = 1.5 or 13.5 V)	15	11	—	11	—	11	—	
I_{OH}	High–Level Output Current								mA
	(V_{out} = 2.5 V)	5.0	−2.5	—	−2.1	—	−1.7	—	
	(V_{out} = 4.6 V)	5.0	−0.52	—	−0.44	—	−0.36	—	
	(V_{out} = 9.5 V)	10	−1.3	—	−1.1	—	−0.9	—	
	(V_{out} = 13.5 V)	15	−3.6	—	−3.0	—	−2.4	—	
I_{OL}	Low–Level Output Current								mA
	(V_{out} = 0.4 V)	5.0	0.52	—	0.44	—	0.36	—	
	(V_{out} = 0.5 V)	10	1.3	—	1.1	—	0.9	—	
	(V_{out} = 1.5 V)	15	3.6	—	3.0	—	2.4	—	
I_{in}	Input Current — $\overline{TE}$	5.0	—	—	3.0	11	—	—	µA
	(MC145026, Pull–Up Device)	10	—	—	16	60	—	—	
		15	—	—	35	120	—	—	
I_{in}	Input Current	15	—	±0.3	—	±0.3	—	±1.0	µA
	R_S (MC145026), D_{in} (MC145027, MC145028)								
I_{in}	Input Current								µA
	A1 – A5, A6/D6 – A9/D9 (MC145026),	5.0	—	—	—	±110	—	—	
	A1 – A5 (MC145027),	10	—	—	—	±500	—	—	
	A1 – A9 (MC145028)	15	—	—	—	±1000	—	—	
C_{in}	Input Capacitance (V_{in} = 0)	—	—	—	—	7.5	—	—	pF
I_{DD}	Quiescent Current — MC145026	5.0	—	—	—	0.1	—	—	µA
		10	—	—	—	0.2	—	—	
		15	—	—	—	0.3	—	—	
I_{DD}	Quiescent Current — MC145027, MC145028	5.0	—	—	—	50	—	—	µA
		10	—	—	—	100	—	—	
		15	—	—	—	150	—	—	
I_{dd}	Dynamic Supply Current — MC145026	5.0	—	—	—	200	—	—	µA
	(f_c = 20 kHz)	10	—	—	—	400	—	—	
		15	—	—	—	600	—	—	
I_{dd}	Dynamic Supply Current — MC145027, MC145028	5.0	—	—	—	400	—	—	µA
	(f_c = 20 kHz)	10	—	—	—	800	—	—	
		15	—	—	—	1200	—	—	

* Also see next Electrical Characteristics table for 2.5 V specifications.

ELECTRICAL CHARACTERISTICS — MC145026 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.5	—	0.05	—	0.05	—	0.05	V
V_{OH}	High–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.5	2.45	—	2.45	—	2.45	—	V
V_{IL}	Low–Level Input Voltage (V_{out} = 0.5 V or 2.0 V)	2.5	—	0.3	—	0.3	—	0.3	V
V_{IH}	High–Level Input Voltage (V_{out} = 0.5 V or 2.0 V)	2.5	2.2	—	2.2	—	2.2	—	V
I_{OH}	High–Level Output Current (V_{out} = 1.25 V)	2.5	0.28	—	0.25	—	0.2	—	mA
I_{OL}	Low–Level Output Current (V_{out} = 0.4 V)	2.5	0.22	—	0.2	—	0.16	—	mA
I_{in}	Input Current ($\overline{TE}$ — Pull–Up Device)	2.5	—	—	0.09	1.8	—	—	µA
I_{in}	Input Current (A1–A5, A6/D6–A9/D9)	2.5	—	—	—	± 25	—	—	µA
I_{DD}	Quiescent Current	2.5	—	—	—	0.05	—	—	µA
I_{dd}	Dynamic Supply Current (f_c = 20 kHz)	2.5	—	—	—	40	—	—	µA

ELECTRICAL CHARACTERISTICS — SC41343 and SC41344 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.8	—	0.05	—	0.05	—	0.05	V
		5.0	—	0.05	—	0.05	—	0.05	
		10	—	0.05	—	0.05	—	0.05	
V_{OH}	High–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.8	2.75	—	2.75	—	2.75	—	V
		5.0	4.95	—	4.95	—	4.95	—	
		10	9.95	—	9.95	—	9.95	—	
V_{IL}	Low–Level Input Voltage								V
	(V_{out} = 2.3 V or 0.5 V)	2.8	—	0.84	—	0.84	—	0.84	
	(V_{out} = 4.5 V or 0.5 V)	5.0	—	1.5	—	1.5	—	1.5	
	(V_{out} = 9.0 V or 1.0 V)	10	—	3.0	—	3.0	—	3.0	
V_{IH}	High–Level Input Voltage								V
	(V_{out} = 0.5 V or 2.3 V)	2.8	1.96	—	1.96	—	1.96	—	
	(V_{out} = 0.5 V or 4.5 V)	5.0	3.5	—	3.5	—	3.5	—	
	(V_{out} = 1.0 V or 9.0 V)	10	7.0	—	7.0	—	7.0	—	
I_{OH}	High–Level Output Current								mA
	(V_{out} = 1.4 V)	2.8	− 0.73	—	− 0.7	—	− 0.55	—	
	(V_{out} = 4.5 V)	5.0	− 0.59	—	− 0.5	—	− 0.41	—	
	(V_{out} = 9.0 V)	10	− 1.3	—	− 1.1	—	− 0.9	—	
I_{OL}	Low- Level Output Current								mA
	(V_{out} = 0.4 V)	2.8	0.35	—	0.3	—	0.24	—	
	(V_{out} = 0.5 V)	5.0	0.8	—	0.6	—	0.4	—	
	(V_{out} = 1.0 V)	10	3.5	—	2.9	—	2.3	—	
I_{in}	Input Current — D_{in}	10	—	± 0.3	—	± 0.3	—	± 1.0	µA
I_{in}	Input Current	2.8	—	—	—	± 30	—	—	µA
	A1 – A5 (SC41343)	5.0	—	—	—	± 140	—	—	
	A1 – A9 (SC41344)	10	—	—	—	± 600	—	—	
C_{in}	Input Capacitance (V_{in} = 0)	—	—	—	—	7.5	—	—	pF
I_{DD}	Quiescent Current	2.8	—	—	—	60	—	—	µA
		5.0	—	—	—	75	—	—	
		10	—	—	—	150	—	—	
I_{dd}	Dynamic Supply Current (f_c = 20 kHz)	2.8	—	—	—	300	—	—	µA
		5.0	—	—	—	500	—	—	
		10	—	—	—	1000	—	—	

2

ELECTRICAL CHARACTERISTICS — MC145026 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.5	—	0.05	—	0.05	—	0.05	V
V_{OH}	High–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.5	2.45	—	2.45	—	2.45	—	V
V_{IL}	Low–Level Input Voltage (V_{out} = 0.5 V or 2.0 V)	2.5	—	0.3	—	0.3	—	0.3	V
V_{IH}	High–Level Input Voltage (V_{out} = 0.5 V or 2.0 V)	2.5	2.2	—	2.2	—	2.2	—	V
I_{OH}	High–Level Output Current (V_{out} = 1.25 V)	2.5	0.28	—	0.25	—	0.2	—	mA
I_{OL}	Low–Level Output Current (V_{out} = 0.4 V)	2.5	0.22	—	0.2	—	0.16	—	mA
I_{in}	Input Current ($\overline{TE}$ — Pull–Up Device)	2.5	—	—	0.09	1.8	—	—	µA
I_{in}	Input Current (A1–A5, A6/D6–A9/D9)	2.5	—	—	—	±25	—	—	µA
I_{DD}	Quiescent Current	2.5	—	—	—	0.05	—	—	µA
I_{dd}	Dynamic Supply Current (f_c = 20 kHz)	2.5	—	—	—	40	—	—	µA

ELECTRICAL CHARACTERISTICS — SC41343 and SC41344 (Voltage Referenced to V_{SS})

Symbol	Characteristic	V_{DD} V	Guaranteed Limit						Unit
			−40°C		25°C		85°C		
			Min	Max	Min	Max	Min	Max	
V_{OL}	Low–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.8	—	0.05	—	0.05	—	0.05	V
		5.0	—	0.05	—	0.05	—	0.05	
		10	—	0.05	—	0.05	—	0.05	
V_{OH}	High–Level Output Voltage (V_{in} = 0 V or V_{DD})	2.8	2.75	—	2.75	—	2.75	—	V
		5.0	4.95	—	4.95	—	4.95	—	
		10	9.95	—	9.95	—	9.95	—	
V_{IL}	Low–Level Input Voltage								V
	(V_{out} = 2.3 V or 0.5 V)	2.8	—	0.84	—	0.84	—	0.84	
	(V_{out} = 4.5 V or 0.5 V)	5.0	—	1.5	—	1.5	—	1.5	
	(V_{out} = 9.0 V or 1.0 V)	10	—	3.0	—	3.0	—	3.0	
V_{IH}	High–Level Input Voltage								V
	(V_{out} = 0.5 V or 2.3 V)	2.8	1.96	—	1.96	—	1.96	—	
	(V_{out} = 0.5 V or 4.5 V)	5.0	3.5	—	3.5	—	3.5	—	
	(V_{out} = 1.0 V or 9.0 V)	10	7.0	—	7.0	—	7.0	—	
I_{OH}	High–Level Output Current								mA
	(V_{out} = 1.4 V)	2.8	−0.73	—	−0.7	—	−0.55	—	
	(V_{out} = 4.5 V)	5.0	−0.59	—	−0.5	—	−0.41	—	
	(V_{out} = 9.0 V)	10	−1.3	—	−1.1	—	−0.9	—	
I_{OL}	Low- Level Output Current								mA
	(V_{out} = 0.4 V)	2.8	0.35	—	0.3	—	0.24	—	
	(V_{out} = 0.5 V)	5.0	0.8	—	0.6	—	0.4	—	
	(V_{out} = 1.0 V)	10	3.5	—	2.9	—	2.3	—	
I_{in}	Input Current — D_{in}	10	—	±0.3	—	±0.3	—	±1.0	µA
I_{in}	Input Current	2.8	—	—	—	±30	—	—	µA
	A1 – A5 (SC41343)	5.0	—	—	—	±140	—	—	
	A1 – A9 (SC41344)	10	—	—	—	±600	—	—	
C_{in}	Input Capacitance (V_{in} = 0)	—	—	—	—	7.5	—	—	pF
I_{DD}	Quiescent Current	2.8	—	—	—	60	—	—	µA
		5.0	—	—	—	75	—	—	
		10	—	—	—	150	—	—	
I_{dd}	Dynamic Supply Current (f_c = 20 kHz)	2.8	—	—	—	300	—	—	µA
		5.0	—	—	—	500	—	—	
		10	—	—	—	1000	—	—	

2

SWITCHING CHARACTERISTICS — MC145026*, MC145027, and MC145028 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4,8	5.0 10 15	— — —	200 100 80	ns
t_r	D_{in} Rise Time — Decoders	5	5.0 10 15	— — —	15 15 15	µs
t_f	D_{in} Fall Time — Decoders	5	5.0 10 15	— — —	15 5.0 4.0	µs
f_{osc}	Encoder Clock Frequency	6	5.0 10 15	0.001 0.001 0.001	2.0 5.0 10	MHz
f	Decoder Frequency — Referenced to Encoder Clock	12	5.0 10 15	1.0 1.0 1.0	240 410 450	kHz
t_w	$\overline{TE}$ Pulse Width — Encoders	7	5.0 10 15	65 30 20	— — —	ns

* Also see next Switching Characteristics table for 2.5 V specifications.

SWITCHING CHARACTERISTICS — MC145026 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4, 8	2.5	—	450	ns
f_{osc}	Encoder Clock Frequency	6	2.5	1.0	250	kHz
t_w	$\overline{TE}$ Pulse Width	7	2.5	1.5	—	µs

SWITCHING CHARACTERISTICS — SC41343 and SC41344 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4, 8	2.8 5.0 10	— — —	320 200 100	ns
t_r	D_{in} Rise Time	5	2.8 5.0 10	— — —	15 15 15	µs
t_f	D_{in} Fall Time	5	2.8 5.0 10	— — —	15 15 5.0	µs
f	Decoder Frequency — Referenced to Encoder Clock	12	2.8 5.0 10	1.0 1.0 1.0	100 240 410	kHz

SWITCHING CHARACTERISTICS — MC145026*, MC145027, and MC145028 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4,8	5.0 10 15	— — —	200 100 80	ns
t_r	D_{in} Rise Time — Decoders	5	5.0 10 15	— — —	15 15 15	µs
t_f	D_{in} Fall Time — Decoders	5	5.0 10 15	— — —	15 5.0 4.0	µs
f_{osc}	Encoder Clock Frequency	6	5.0 10 15	0.001 0.001 0.001	2.0 5.0 10	MHz
f	Decoder Frequency — Referenced to Encoder Clock	12	5.0 10 15	1.0 1.0 1.0	240 410 450	kHz
t_w	$\overline{TE}$ Pulse Width — Encoders	7	5.0 10 15	65 30 20	— — —	ns

* Also see next Switching Characteristics table for 2.5 V specifications.

SWITCHING CHARACTERISTICS — MC145026 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4, 8	2.5	—	450	ns
f_{osc}	Encoder Clock Frequency	6	2.5	1.0	250	kHz
t_w	$\overline{TE}$ Pulse Width	7	2.5	1.5	—	µs

SWITCHING CHARACTERISTICS — SC41343 and SC41344 (C_L = 50 pF, T_A = 25°C)

Symbol	Characteristic	Figure No.	V_{DD}	Guaranteed Limit		Unit
				Min	Max	
t_{TLH}, t_{THL}	Output Transition Time	4, 8	2.8 5.0 10	— — —	320 200 100	ns
t_r	D_{in} Rise Time	5	2.8 5.0 10	— — —	15 15 15	µs
t_f	D_{in} Fall Time	5	2.8 5.0 10	— — —	15 15 5.0	µs
f	Decoder Frequency — Referenced to Encoder Clock	12	2.8 5.0 10	1.0 1.0 1.0	100 240 410	kHz

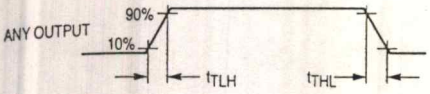


Figure 4.

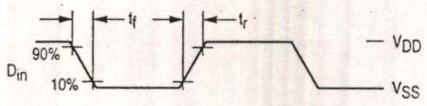


Figure 5.

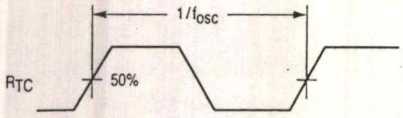


Figure 6.

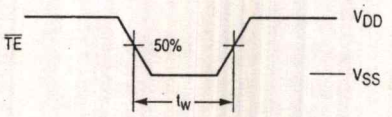
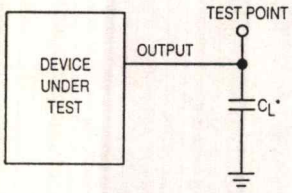


Figure 7.



* Includes all probe and fixture capacitance.

Figure 8. Test Circuit

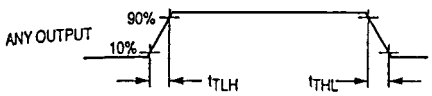


Figure 4.

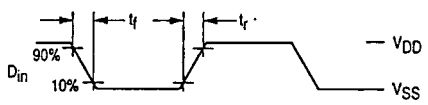


Figure 5.

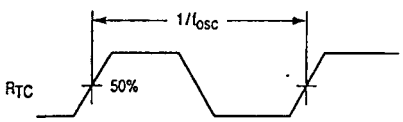


Figure 6.

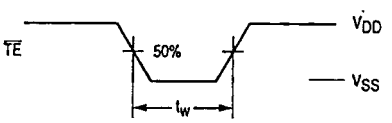
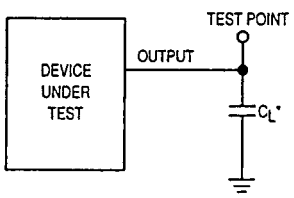


Figure 7.



* Includes all probe and fixture capacitance.

Figure 8. Test Circuit

OPERATING CHARACTERISTICS

MC145026

The encoder serially transmits trinary data as defined by the state of the A1 – A5 and A6/D6 – A9/D9 input pins. These pins may be in either of three states (low, high, or open) allowing 19,683 possible codes. The transmit sequence is initiated by a low level on the $\overline{TE}$ input pin. Upon power–up, the MC145026 can continuously transmit as long as $\overline{TE}$ remains low (also, the device can transmit two–word sequences by pulsing $\overline{TE}$ low). However, no MC145026 application should be designed to rely upon the first data word transmitted immediately after power–up because this word may be invalid. Between the two data words, no signal is sent for three data periods (see Figure 10).

Each transmitted trinary digit is encoded into pulses (see Figure 11). A logic 0 (low) is encoded as two consecutive short pulses, a logic 1 (high) as two consecutive long pulses, and an open (high impedance) as a long pulse followed by a short pulse. The input state is determined by using a weak "output" device to try to force each input high then low. If only a high state results from the two tests, the input is assumed to be hardwired to V_{DD}. If only a low state is obtained, the input is assumed to be hardwired to V_{SS}. If both a high and a low can be forced at an input, an open is assumed and is encoded as such. The "high" and "low" levels are 70% and 30% of the supply voltage as shown in the Electrical Characteristics table. The weak "output" device sinks/sources up to 110 μA at a 5 V supply level, 500 μA at 10 V, and 1 mA at 15 V.

The $\overline{TE}$ input has an internal pull–up device so that a simple switch may be used to force the input low. While $\overline{TE}$ is high, the encoder is completely disabled, the oscillator is inhibited, and the current drain is reduced to quiescent current. When $\overline{TE}$ is brought low, the oscillator is started and the transmit sequence begins. The inputs are then sequentially selected, and determinations are made as to the input logic states. This information is serially transmitted via the D_{out} pin.

MC145027

This decoder receives the serial data from the encoder and outputs the data, if it is valid. The transmitted data, consisting of two identical words, is examined bit by bit during reception. The first five trinary digits are assumed to be the address. If the received address matches the local address, the next four (data) bits are internally stored, but are not transferred to the output data latch. As the second encoded word is received, the address must again match. If a match occurs, the new data bits are checked against the previously stored data bits. If the two nibbles of data (four bits each) match, the data is transferred to the output data latch by VT and remains until new data replaces it. At the same time, the VT output pin is brought high and remains high until an error is received or until no input signal is received for four data periods (see Figure 10).

Although the address information may be encoded in trinary, the data information must be either a 1 or 0. A trinary (open) data line is decoded as a logic 1.

MC145028

This decoder operates in the same manner as the MC145027 except that nine address lines are used and no data output is available. The VT output is used to indicate that a valid address has been received. For transmission security, two identical transmitted words must be consecutively received before a VT output signal is issued.

The MC145028 allows 19,683 addresses when trinary levels are used. 512 addresses are possible when binary levels are used.

PIN DESCRIPTIONS

MC145026 ENCODER

A1 – A5, A6/D6 – A9/D9
Address, Address/Data Inputs (Pins 1 – 7, 9, and 10)

These address/data inputs are encoded and the data is sent serially from the encoder via the D_{out} pin.

R_S, C_{TC}, R_{TC}
(Pins 11, 12, and 13)

These pins are part of the oscillator section of the encoder (see Figure 9).

If an external signal source is used instead of the internal oscillator, it should be connected to the R_S input and the R_{TC} and C_{TC} pins should be left open.

$\overline{TE}$
Transmit Enable (Pin 14)

This active–low transmit enable input initiates transmission when forced low. An internal pull–up device keeps this input normally high. The pull–up current is specified in the Electrical Characteristics table.

D_{out}
Data Out (Pin 15)

This is the output of the encoder that serially presents the encoded data word.

V_{SS}
Negative Power Supply (Pin 8)

The most–negative supply potential. This pin is usually ground.

V_{DD}
Positive Power Supply (Pin 16)

The most–positive power supply pin.

MC145027 AND MC145028 DECODERS

A1 – A5, A1 – A9
Address Inputs (Pins 1 – 5) — MC145027,
Address Inputs (Pins 1 – 5, 15, 14, 13, 12) — MC145028

These are the local address inputs. The states of these pins must match the appropriate encoder inputs for the VT pin to go high. The local address may be encoded with trinary or binary data.

D6 – D9
Data Outputs (Pins 15, 14, 13, 12) — MC145027 Only

These outputs present the binary information that is on encoder inputs A6/D6 through A9/D9. Only binary data is

OPERATING CHARACTERISTICS

MC145026

The encoder serially transmits trinary data as defined by the state of the A1 – A5 and A6/D6 – A9/D9 input pins. These pins may be in either of three states (low, high, or open) allowing 19,683 possible codes. The transmit sequence is initiated by a low level on the $\overline{TE}$ input pin. Upon power–up, the MC145026 can continuously transmit as long as $\overline{TE}$ remains low (also, the device can transmit two–word sequences by pulsing $\overline{TE}$ low). However, no MC145026 application should be designed to rely upon the first data word transmitted immediately after power–up because this word may be invalid. Between the two data words, no signal is sent for three data periods (see Figure 10).

Each transmitted trinary digit is encoded into pulses (see Figure 11). A logic 0 (low) is encoded as two consecutive short pulses, a logic 1 (high) as two consecutive long pulses, and an open (high impedance) as a long pulse followed by a short pulse. The input state is determined by using a weak "output" device to try to force each input high then low. If only a high state results from the two tests, the input is assumed to be hardwired to V_{DD}. If only a low state is obtained, the input is assumed to be hardwired to V_{SS}. If both a high and a low can be forced at an input, an open is assumed and is encoded as such. The "high" and "low" levels are 70% and 30% of the supply voltage as shown in the Electrical Characteristics table. The weak "output" device sinks/sources up to 110 μA at a 5 V supply level, 500 μA at 10 V, and 1 mA at 15 V.

The $\overline{TE}$ input has an internal pull–up device so that a simple switch may be used to force the input low. While $\overline{TE}$ is high, the encoder is completely disabled, the oscillator is inhibited, and the current drain is reduced to quiescent current. When $\overline{TE}$ is brought low, the oscillator is started and the transmit sequence begins. The inputs are then sequentially selected, and determinations are made as to the input logic states. This information is serially transmitted via the D_{out} pin.

MC145027

This decoder receives the serial data from the encoder and outputs the data, if it is valid. The transmitted data, consisting of two identical words, is examined bit by bit during reception. The first five trinary digits are assumed to be the address. If the received address matches the local address, the next four (data) bits are internally stored, but are not transferred to the output data latch. As the second encoded word is received, the address must again match. If a match occurs, the new data bits are checked against the previously stored data bits. If the two nibbles of data (four bits each) match, the data is transferred to the output data latch by VT and remains until new data replaces it. At the same time, the VT output pin is brought high and remains high until an error is received or until no input signal is received for four data periods (see Figure 10).

Although the address information may be encoded in trinary, the data information must be either a 1 or 0. A trinary (open) data line is decoded as a logic 1.

MC145028

This decoder operates in the same manner as the MC145027 except that nine address lines are used and no data output is available. The VT output is used to indicate that a valid address has been received. For transmission security, two identical transmitted words must be consecutively received before a VT output signal is issued.

The MC145028 allows 19,683 addresses when trinary levels are used. 512 addresses are possible when binary levels are used.

PIN DESCRIPTIONS

MC145026 ENCODER

A1 – A5, A6/D6 – A9/D9
Address, Address/Data Inputs (Pins 1 – 7, 9, and 10)

These address/data inputs are encoded and the data is sent serially from the encoder via the D_{out} pin.

R_S, C_{TC}, R_{TC}
(Pins 11, 12, and 13)

These pins are part of the oscillator section of the encoder (see Figure 9).

If an external signal source is used instead of the internal oscillator, it should be connected to the R_S input and the R_{TC} and C_{TC} pins should be left open.

$\overline{TE}$
Transmit Enable (Pin 14)

This active–low transmit enable input initiates transmission when forced low. An internal pull–up device keeps this input normally high. The pull–up current is specified in the Electrical Characteristics table.

D_{out}
Data Out (Pin 15)

This is the output of the encoder that serially presents the encoded data word.

V_{SS}
Negative Power Supply (Pin 8)

The most–negative supply potential. This pin is usually ground.

V_{DD}
Positive Power Supply (Pin 16)

The most–positive power supply pin.

MC145027 AND MC145028 DECODERS

A1 – A5, A1 – A9
Address Inputs (Pins 1 – 5) — MC145027,
Address Inputs (Pins 1 – 5, 15, 14, 13, 12) — MC145028

These are the local address inputs. The states of these pins must match the appropriate encoder inputs for the VT pin to go high. The local address may be encoded with trinary or binary data.

D6 – D9
Data Outputs (Pins 15, 14, 13, 12) — MC145027 Only

These outputs present the binary information that is on encoder inputs A6/D6 through A9/D9. Only binary data is

acknowledged; a trinary open at the MC145026 encoder is decoded as a high level (logic 1).

D_{in}
Data In (Pin 9)

This pin is the serial data input to the decoder. The input voltage must be at CMOS logic levels. The signal source driving this pin must be dc coupled.

R_1, C_1
Resistor 1, Capacitor 1 (Pins 6, 7)

As shown in Figures 2 and 3, these pins accept a resistor and capacitor that are used to determine whether a narrow pulse or wide pulse has been received. The time constant $R_1 \times C_1$ should be set to 1.72 encoder clock periods:

$$R_1 C_1 = 3.95 R_{TC} C_{TC}$$

R_2/C_2
Resistor 2/Capacitor 2 (Pin 10)

As shown in Figures 2 and 3, this pin accepts a resistor and capacitor that are used to detect both the end of a received word and the end of a transmission. The time constant $R_2 \times C_2$ should be 33.5 encoder clock periods (four data periods per Figure 11): $R_2 C_2 = 77 R_{TC} C_{TC}$. This time constant is used to determine whether the D_{in} pin has remained low for four data periods (end of transmission). A separate on–chip comparator looks at the voltage–equivalent two data periods ($0.4 R_2 C_2$) to detect the dead time between received words within a transmission.

VT
Valid Transmission Output (Pin 11)

This valid transmission output goes high after the second word of an encoding sequence when the following conditions are satisfied:

1. the received addresses of both words match the local decoder address, and
2. the received data bits of both words match.

VT remains high until either a mismatch is received or no input signal is received for four data periods.

V_{SS}
Negative Power Supply (Pin 8)

The most–negative supply potential. This pin is usually ground.

V_{DD}
Positive Power Supply (Pin 16)

The most–positive power supply pin.

acknowledged; a trinary open at the MC145026 encoder is decoded as a high level (logic 1).

D_{in}
Data In (Pin 9)

This pin is the serial data input to the decoder. The input voltage must be at CMOS logic levels. The signal source driving this pin must be dc coupled.

R_1, C_1
Resistor 1, Capacitor 1 (Pins 6, 7)

As shown in Figures 2 and 3, these pins accept a resistor and capacitor that are used to determine whether a narrow pulse or wide pulse has been received. The time constant $R_1 \times C_1$ should be set to 1.72 encoder clock periods:

$$R_1 C_1 = 3.95 R_{TC} C_{TC}$$

R_2/C_2
Resistor 2/Capacitor 2 (Pin 10)

As shown in Figures 2 and 3, this pin accepts a resistor and capacitor that are used to detect both the end of a received word and the end of a transmission. The time constant $R_2 \times C_2$ should be 33.5 encoder clock periods (four data periods per Figure 11): $R_2 C_2 = 77 R_{TC} C_{TC}$. This time constant is used to determine whether the D_{in} pin has remained low for four data periods (end of transmission). A separate on–chip comparator looks at the voltage–equivalent two data periods ($0.4 R_2 C_2$) to detect the dead time between received words within a transmission.

VT
Valid Transmission Output (Pin 11)

This valid transmission output goes high after the second word of an encoding sequence when the following conditions are satisfied:

1. the received addresses of both words match the local decoder address, and
2. the received data bits of both words match.

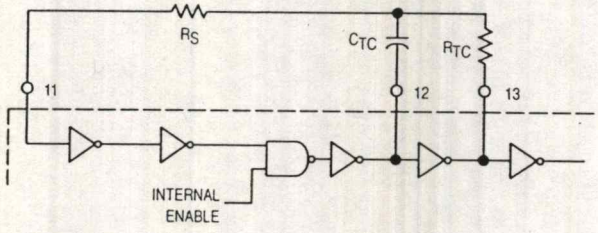
VT remains high until either a mismatch is received or no input signal is received for four data periods.

V_{SS}
Negative Power Supply (Pin 8)

The most–negative supply potential. This pin is usually ground.

V_{DD}
Positive Power Supply (Pin 16)

The most–positive power supply pin.



2

This oscillator operates at a frequency determined by the external RC network; i.e.,

$$f = \frac{1}{2.3\, R_{TC}\, C_{TC'}} \text{ (Hz)}$$

for 1 kHz ≤ f ≤ 400 kHz

where: $C_{TC'} = C_{TC} + C_{layout} + 12$ pF

$R_S = 2\, R_{TC}$

$R_S \geq 20$ k

$R_{TC} \geq 10$ k

400 pF < C_{TC} < 15 μF

The value for R_S should be chosen to be ≥ 2 times R_{TC}. This range ensures that current through R_S is insignificant compared to current through R_{TC}. The upper limit for R_S must ensure that R_S x 5 pF (input capacitance) is small compared to R_{TC} x C_{TC}.

For frequencies outside the indicated range, the formula is less accurate. The minimum recommended oscillation frequency of this circuit is 1 kHz. Susceptibility to externally induced noise signals may occur for frequencies below 1 kHz and/or when resistors utilized are greater than 1 MΩ.

Figure 9. Encoder Oscillator Information

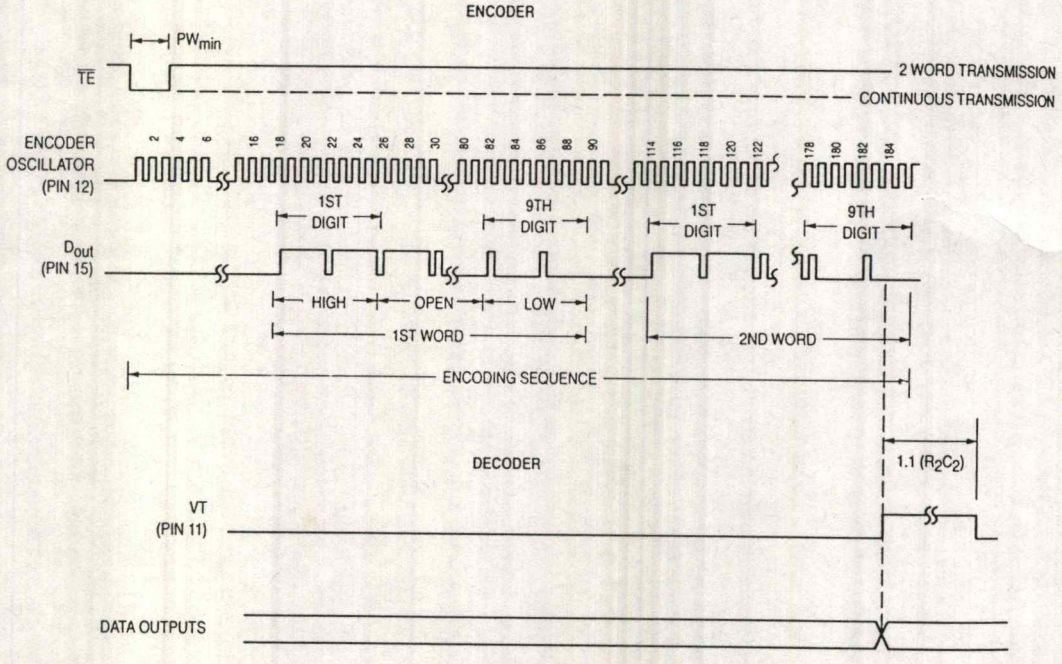
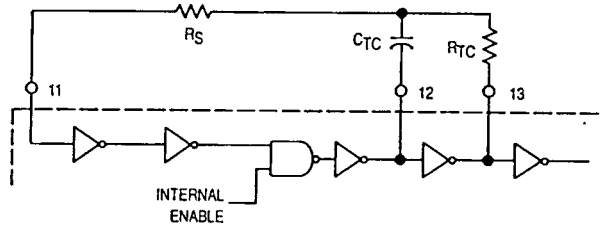


Figure 10. Timing Diagram



2

This oscillator operates at a frequency determined by the external RC network; i.e.,

$$f \approx \frac{1}{2.3\, R_{TC}\, C_{TC'}} \text{ (Hz)}$$

for 1 kHz ≤ f ≤ 400 kHz

where: $C_{TC'} = C_{TC} + C_{layout} + 12$ pF

$R_S = 2\, R_{TC}$

$R_S \geq 20$ k

$R_{TC} \geq 10$ k

400 pF < C_{TC} < 15 µF

The value for R_S should be chosen to be ≥ 2 times R_{TC}. This range ensures that current through R_S is insignificant compared to current through R_{TC}. The upper limit for R_S must ensure that R_S x 5 pF (input capacitance) is small compared to R_{TC} x C_{TC}.

For frequencies outside the indicated range, the formula is less accurate. The minimum recommended oscillation frequency of this circuit is 1 kHz. Susceptibility to externally induced noise signals may occur for frequencies below 1 kHz and/or when resistors utilized are greater than 1 MΩ.

Figure 9. Encoder Oscillator Information

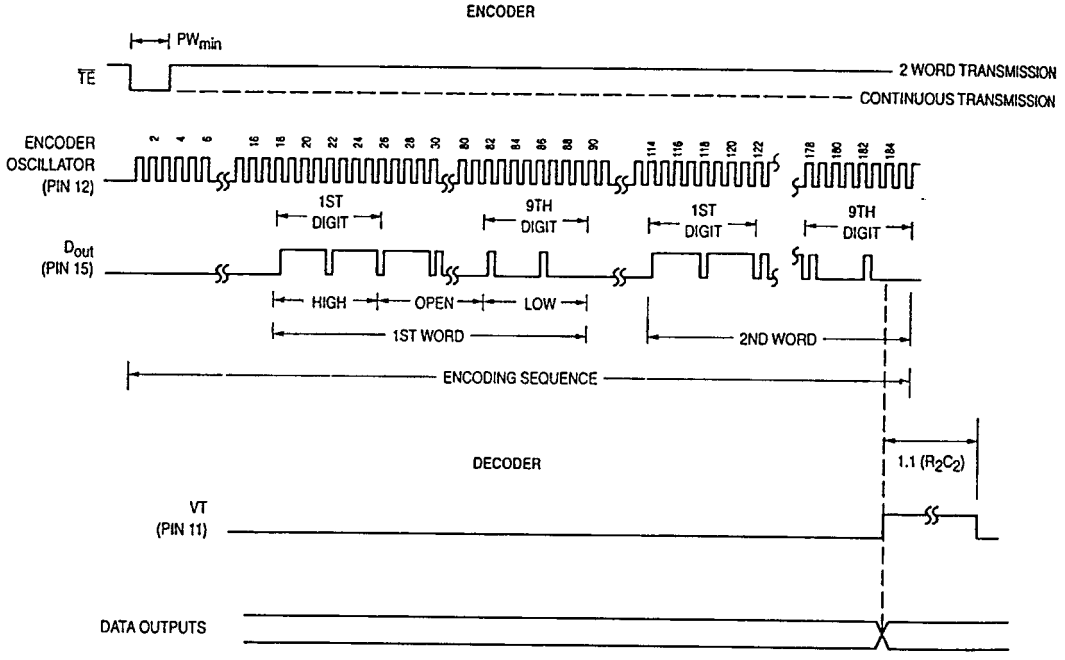


Figure 10. Timing Diagram

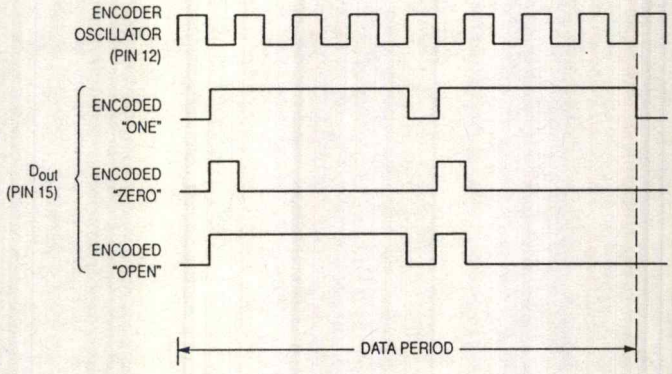


Figure 11. Encoder Data Waveforms

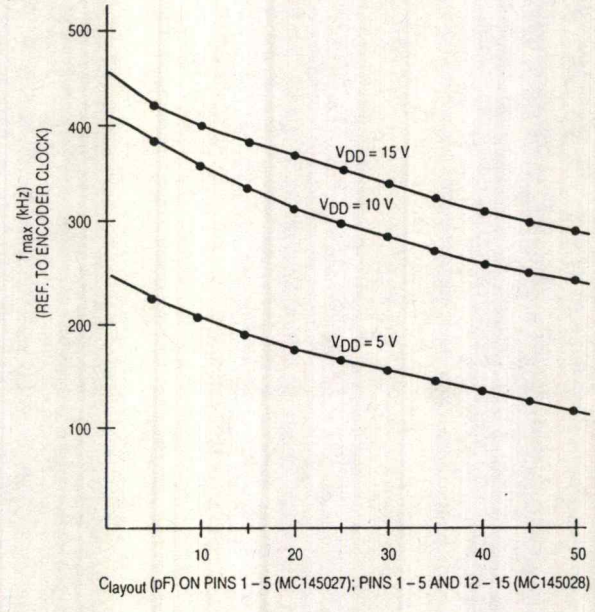


Figure 12. f_{max} vs C_{layout} — Decoders Only

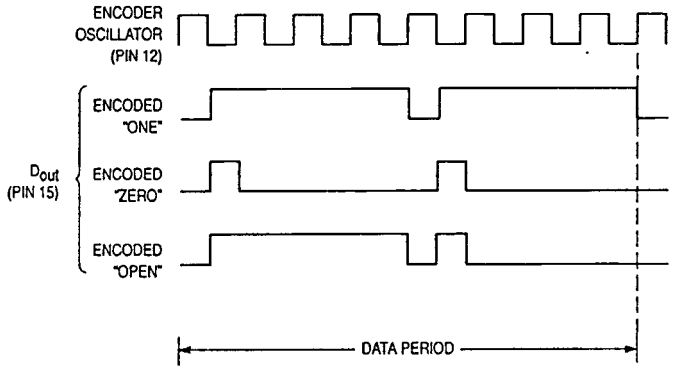


Figure 11. Encoder Data Waveforms

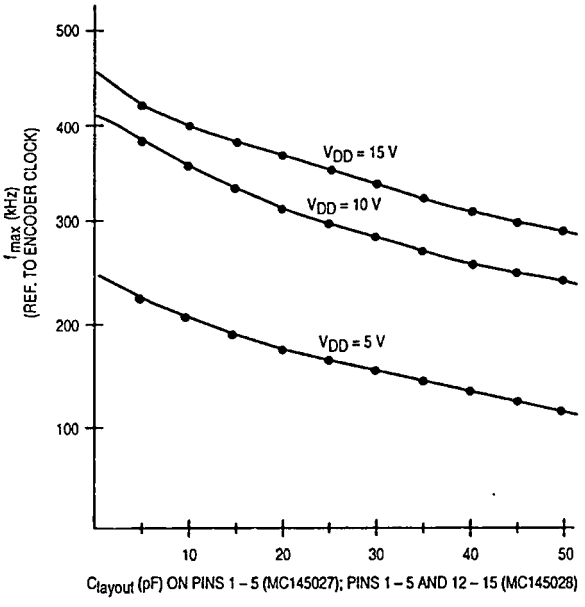


Figure 12. f_{max} vs C_{layout} — Decoders Only

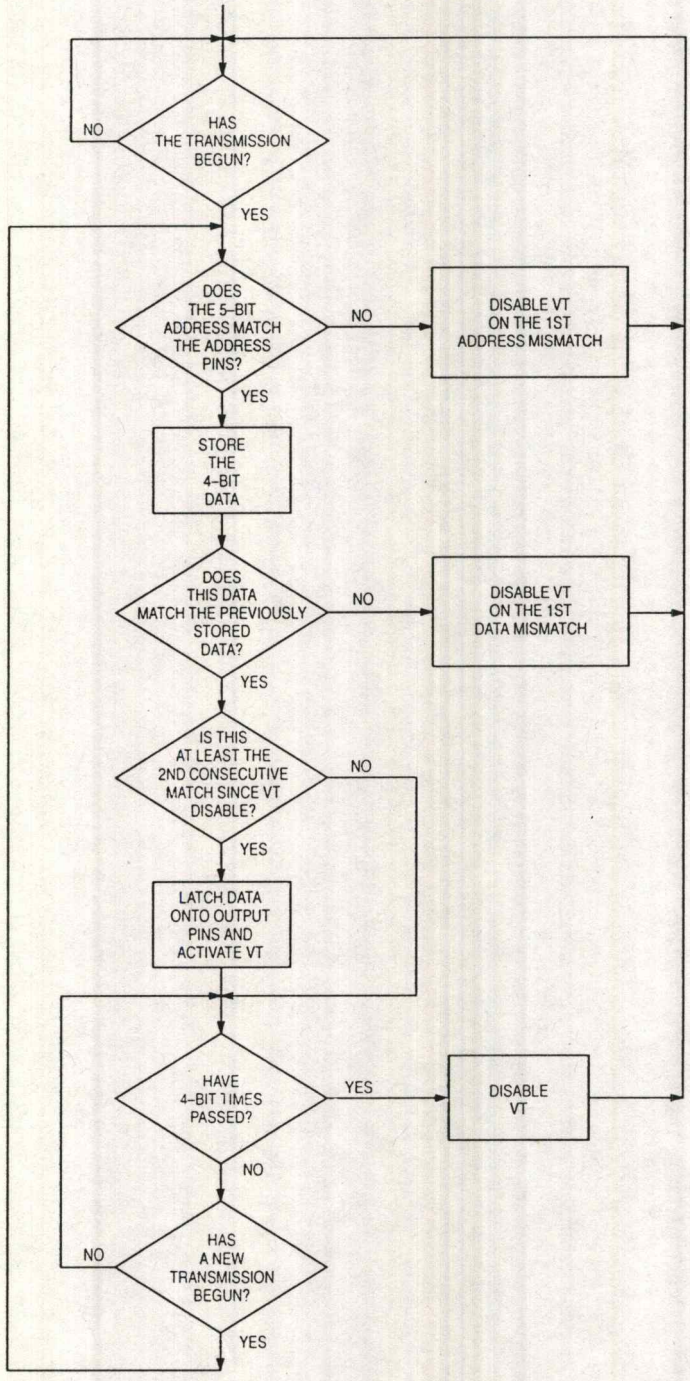


Figure 13. MC145027 Flowchart

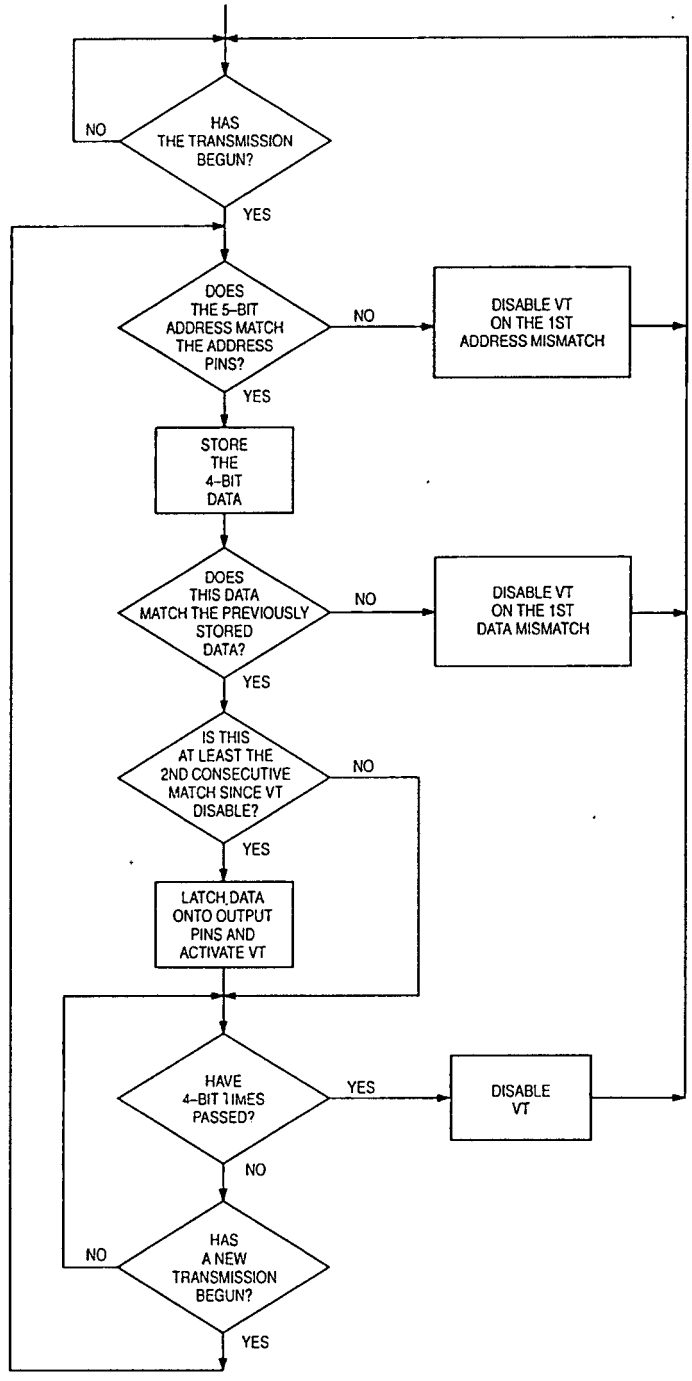


Figure 13. MC145027 Flowchart

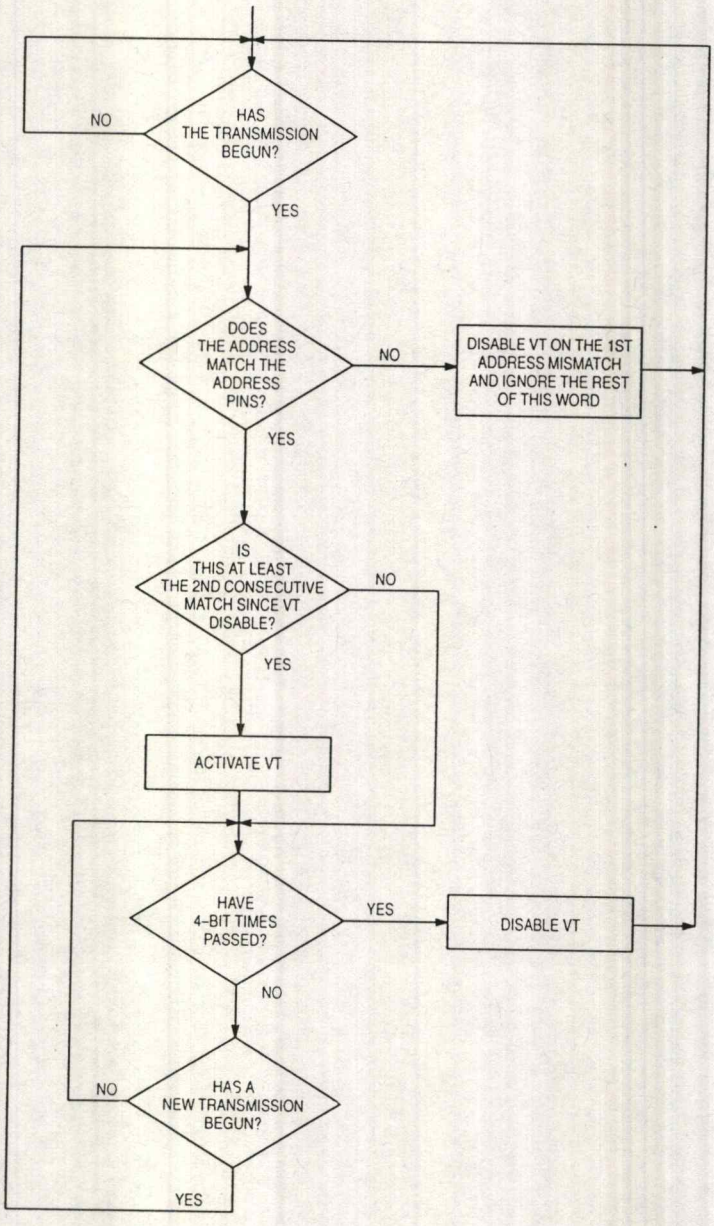


Figure 14. MC145028 Flowchart

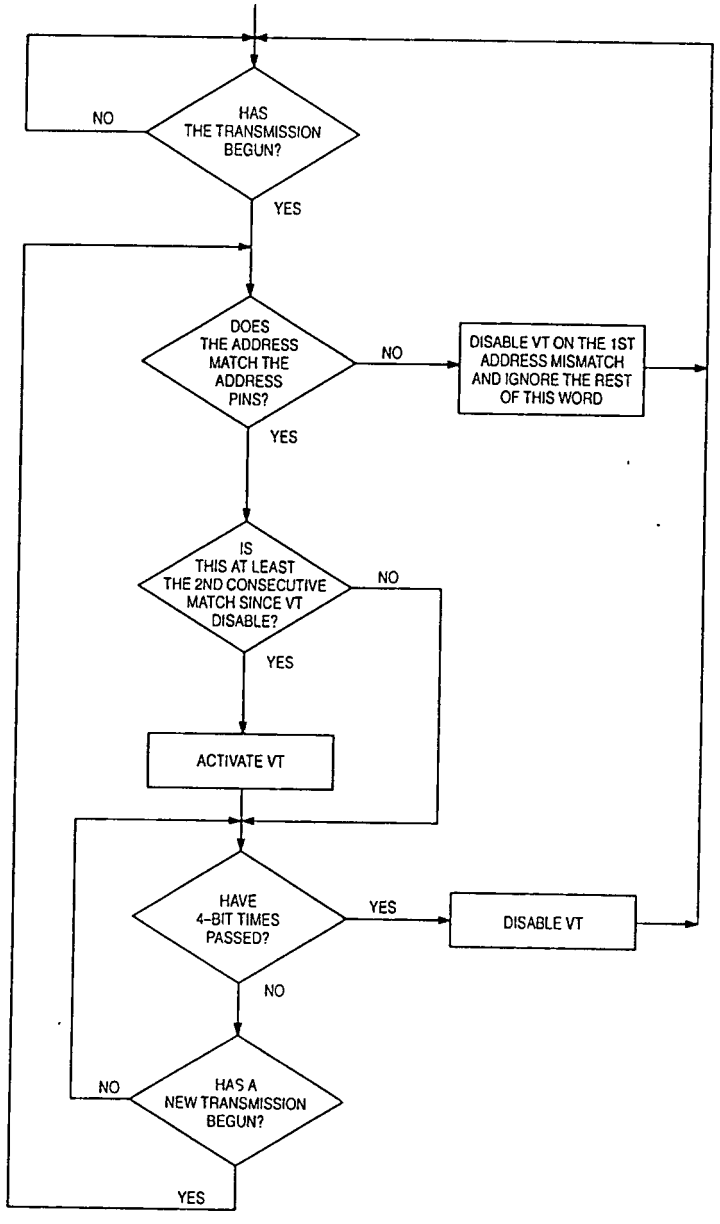


Figure 14. MC145028 Flowchart

MC145027 AND MC145028 TIMING

To verify the MC145027 or MC145028 timing, check the waveforms on C1 (Pin 7) and R2/C2 (Pin 10) as compared to the incoming data waveform on D_{in} (Pin 9).

The R–C decay seen on C1 discharges down to 1/3 V_{DD} before being reset to V_{DD}. This point of reset (labelled "DOS" in Figure 15) is the point in time where the decision is made whether the data seen on D_{in} is a 1 or 0. DOS should not be too close to the D_{in} data edges or intermittent operation may occur.

The other timing to be checked on the MC145027 and MC145028 is on R2/C2 (see Figure 16). The R–C decay is continually reset to V_{DD} as data is being transmitted. Only between words and after the end–of–transmission (EOT) does R2/C2 decay significantly from V_{DD}. R2/C2 can be used to identify the internal end–of–word (EOW) timing edge which is generated when R2/C2 decays to 2/3 V_{DD}. The internal EOT timing edge occurs when R2/C2 decays to 1/3 V_{DD}. When the waveform is being observed, the R–C decay should go down between the 2/3 and 1/3 V_{DD} levels, but not too close to either level before data transmission on D_{in} resumes.

Verification of the timing described above should ensure a good match between the MC145026 transmitter and the MC145027 and MC145028 receivers.

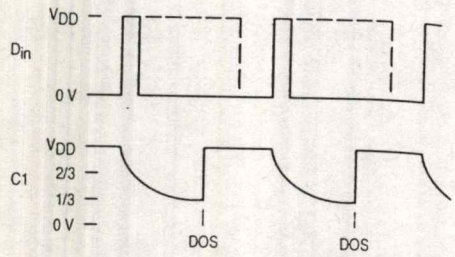


Figure 15. R–C Decay on Pin 7 (C1)

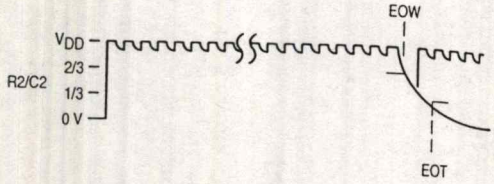


Figure 16. R–C Decay on Pin 10 (R2/C2)

MC145027 AND MC145028 TIMING

To verify the MC145027 or MC145028 timing, check the waveforms on C1 (Pin 7) and R2/C2 (Pin 10) as compared to the incoming data waveform on D_{in} (Pin 9).

The R–C decay seen on C1 discharges down to 1/3 V_{DD} before being reset to V_{DD}. This point of reset (labelled "DOS" in Figure 15) is the point in time where the decision is made whether the data seen on D_{in} is a 1 or 0. DOS should not be too close to the D_{in} data edges or intermittent operation may occur.

The other timing to be checked on the MC145027 and MC145028 is on R2/C2 (see Figure 16). The R–C decay is continually reset to V_{DD} as data is being transmitted. Only between words and after the end–of–transmission (EOT) does R2/C2 decay significantly from V_{DD}. R2/C2 can be used to identify the internal end–of–word (EOW) timing edge which is generated when R2/C2 decays to 2/3 V_{DD}. The internal EOT timing edge occurs when R2/C2 decays to 1/3 V_{DD}. When the waveform is being observed, the R–C decay should go down between the 2/3 and 1/3 V_{DD} levels, but not too close to either level before data transmission on D_{in} resumes.

Verification of the timing described above should ensure a good match between the MC145026 transmitter and the MC145027 and MC145028 receivers.

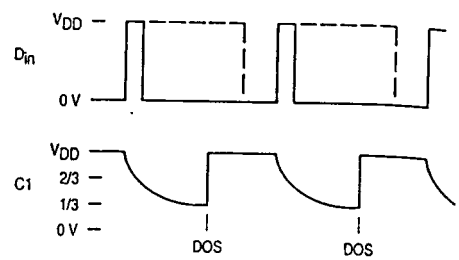


Figure 15. R–C Decay on Pin 7 (C1)

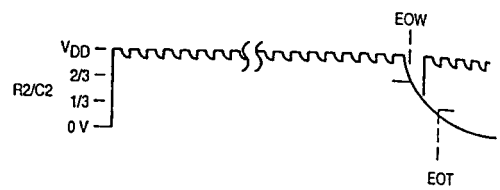
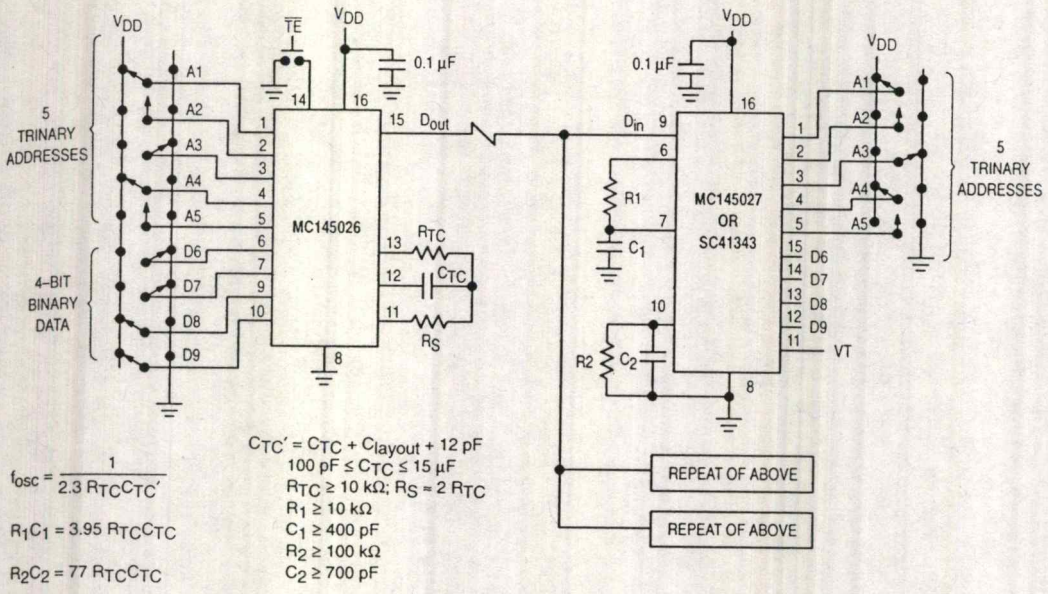


Figure 16. R–C Decay on Pin 10 (R2/C2)



$$f_{osc} = \frac{1}{2.3\ R_{TC}C_{TC'}}$$

$R_1C_1 = 3.95\ R_{TC}C_{TC}$

$R_2C_2 = 77\ R_{TC}C_{TC}$

$C_{TC'} = C_{TC} + C_{layout} + 12\ pF$

$100\ pF \le C_{TC} \le 15\ \mu F$

$R_{TC} \ge 10\ k\Omega$; $R_S \approx 2\ R_{TC}$

$R_1 \ge 10\ k\Omega$

$C_1 \ge 400\ pF$

$R_2 \ge 100\ k\Omega$

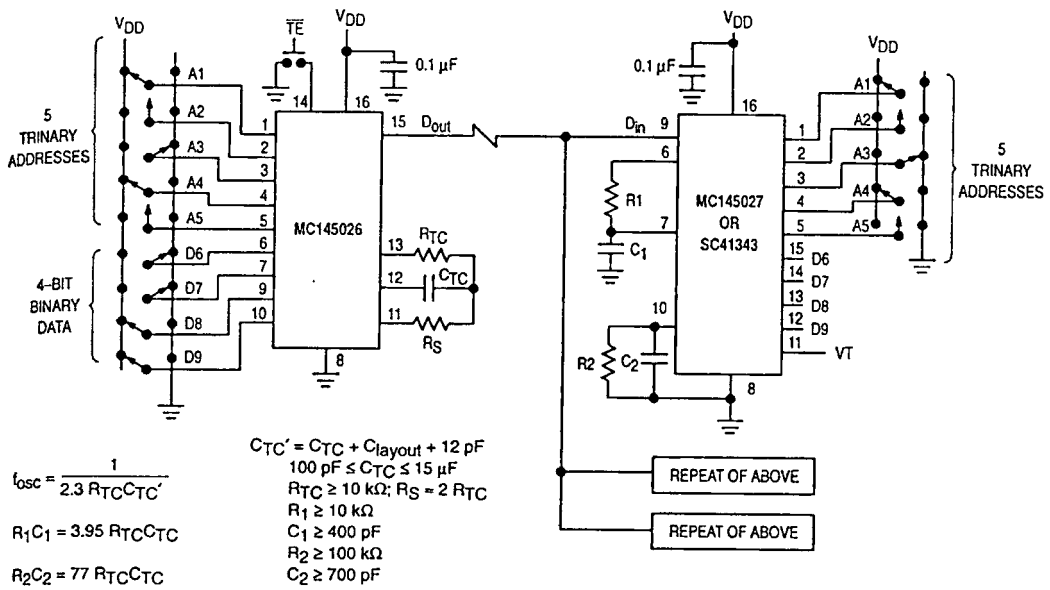
$C_2 \ge 700\ pF$

Example R/C Values (All Resistors and Capacitors are ± 5%)

($C_{TC'} = C_{TC} + 20\ pF$)

f_{osc} (kHz)	R_{TC}	$C_{TC'}$	R_S	R_1	C_1	R_2	C_2
362	10 k	120 pF	20 k	10 k	470 pF	100 k	910 pF
181	10 k	240 pF	20 k	10 k	910 pF	100 k	1800 pF
88.7	10 k	490 pF	20 k	10 k	2000 pF	100 k	3900 pF
42.6	10 k	1020 pF	20 k	10 k	3900 pF	100 k	7500 pF
21.5	10 k	2020 pF	20 k	10 k	8200 pF	100 k	0.015 µF
8.53	10 k	5100 pF	20 k	10 k	0.02 µF	200 k	0.02 µF
1.71	50 k	5100 pF	100 k	50 k	0.02 µF	200 k	0.1 µF

Figure 17. Typical Application



$$f_{osc} = \frac{1}{2.3\,R_{TC}C_{TC}'}$$

$R_1C_1 = 3.95\,R_{TC}C_{TC}$

$R_2C_2 = 77\,R_{TC}C_{TC}$

$C_{TC}' = C_{TC} + C_{layout} + 12\text{ pF}$

$100\text{ pF} \le C_{TC} \le 15\ \mu\text{F}$

$R_{TC} \ge 10\text{ k}\Omega$; $R_S \approx 2\,R_{TC}$

$R_1 \ge 10\text{ k}\Omega$

$C_1 \ge 400\text{ pF}$

$R_2 \ge 100\text{ k}\Omega$

$C_2 \ge 700\text{ pF}$

Example R/C Values (All Resistors and Capacitors are ± 5%)

($C_{TC}' = C_{TC} + 20$ pF)

f_{osc} (kHz)	R_{TC}	C_{TC}'	R_S	R_1	C_1	R_2	C_2
362	10 k	120 pF	20 k	10 k	470 pF	100 k	910 pF
181	10 k	240 pF	20 k	10 k	910 pF	100 k	1800 pF
88.7	10 k	490 pF	20 k	10 k	2000 pF	100 k	3900 pF
42.6	10 k	1020 pF	20 k	10 k	3900 pF	100 k	7500 pF
21.5	10 k	2020 pF	20 k	10 k	8200 pF	100 k	0.015 µF
8.53	10 k	5100 pF	20 k	10 k	0.02 µF	200 k	0.02 µF
1.71	50 k	5100 pF	100 k	50 k	0.02 µF	200 k	0.1 µF

Figure 17. Typical Application

APPLICATIONS INFORMATION

INFRARED TRANSMITTER

In Figure 18, the MC145026 encoder is set to run at an oscillator frequency of about 4 to 9 kHz. Thus, the time required for a complete two–word encoding sequence is about 20 to 40 ms. The data output from the encoder gates an RC oscillator running at 50 kHz; the oscillator shown starts rapidly enough to be used in this application. When the "send" button is not depressed, both the MC145026 and oscillator are in a low–power standby state. The RC oscillator has to be trimmed for 50 kHz and has some drawbacks for frequency stability. A superior system uses a ceramic resonator oscillator running at 400 kHz. This oscillator feeds a divider as shown in Figure 19. The unused inputs of the MC14011UB must be grounded.

The MLED81 IRED is driven with the 50 kHz square wave at about 200 to 300 mA to generate the carrier. If desired, two IREDs wired in series can be used (see Application Note AN1016 for more information). The bipolar IRED switch, shown in Figure 18, offers two advantages over a FET. First, a logic FET has too much gate capacitance for the MC14011UB to drive without waveform distortion. Second, the bipolar drive permits lower supply voltages, which are an advantage in portable battery–powered applications.

The configuration shown in Figure 18 operates over a supply range of 4.5 to 18 V. A low–voltage system which operates down to 2.5 V could be realized if the oscillator section of a MC74HC4060 is used in place of the MC14011UB. The data output of the MC145026 is inverted and fed to the RESET pin of the MC74HC4060. Alternately, the MC74HCU04 could be used for the oscillator.

Information on the MC14011UB is in book number DL131/D. The MC74HCU04 and MC74HC4060 are found in book number DL129/D.

INFRARED RECEIVER

The receiver in Figure 20 couples an IR–sensitive diode to input preamp A1, followed by band–pass amplifier A2 with a gain of about 10. Limiting stage A3 follows, with an output of about 800 mV p–p. The limited 50 kHz burst is detected by comparator A4 that passes only positive pulses, and peak–detected and filtered by a diode/RC network to extract the data envelope from the burst. Comparator A5 boosts the signal to logic levels compatible with the MC145027/28 data input. The D_{in} pin of these decoders is a standard CMOS high–impedance input which must ***not*** be allowed to float. Therefore, direct coupling from A5 to the decoder input is utilized.

Shielding should be used on at least A1 and A2, with good ground and high–sensitivity circuit layout techniques applied.

For operation with supplies higher than + 5 V, limiter A4's positive output swing needs to be limited to 3 to 5 V. This is accomplished via adding a zener diode in the negative feedback path, thus avoiding excessive system noise. The biasing resistor stack should be adjusted such that V3 is 1.25 to 1.5 V.

This system works up to a range of about 10 meters. The gains of the system may be adjusted to suit the individual design needs. The 100 Ω resistor in the emitter of the first 2N5088 and the 1 kΩ resistor feeding A2 may be altered if different gain is required. In general, more gain does not necessarily result in increased range. This is due to noise floor limitations. The designer should increase transmitter power and/or increase receiver aperature with Fresnal lensing to greatly improve range. See Application Note AN1016 for additional information.

Information on the MC34074 is in data book DL128/D.

TRINARY SWITCH MANUFACTURERS

Midland Ross–Electronic Connector Div.
617/491–5400
Greyhill
312/354–1040
Augat/Alcoswitch
617/685–4371
Aries Electronics
201/996–6841

The above companies may not have the switches in a DIP. For more information, call them or consult *eem Electronic Engineers Master Catalog* or the *Gold Book*. ***Ask for SPDT with center OFF.***

Alternative: An SPST can be placed in series between a SPDT and the Encoder or Decoder to achieve trinary action.

Motorola cannot recommend one supplier over another and in no way suggests that this is a complete listing of trinary switch manufacturers.

APPLICATIONS INFORMATION

INFRARED TRANSMITTER

In Figure 18, the MC145026 encoder is set to run at an oscillator frequency of about 4 to 9 kHz. Thus, the time required for a complete two–word encoding sequence is about 20 to 40 ms. The data output from the encoder gates an RC oscillator running at 50 kHz; the oscillator shown starts rapidly enough to be used in this application. When the "send" button is not depressed, both the MC145026 and oscillator are in a low–power standby state. The RC oscillator has to be trimmed for 50 kHz and has some drawbacks for frequency stability. A superior system uses a ceramic resonator oscillator running at 400 kHz. This oscillator feeds a divider as shown in Figure 19. The unused inputs of the MC14011UB must be grounded.

The MLED81 IRED is driven with the 50 kHz square wave at about 200 to 300 mA to generate the carrier. If desired, two IREDs wired in series can be used (see Application Note AN1016 for more information). The bipolar IRED switch, shown in Figure 18, offers two advantages over a FET. First, a logic FET has too much gate capacitance for the MC14011UB to drive without waveform distortion. Second, the bipolar drive permits lower supply voltages, which are an advantage in portable battery–powered applications.

The configuration shown in Figure 18 operates over a supply range of 4.5 to 18 V. A low–voltage system which operates down to 2.5 V could be realized if the oscillator section of a MC74HC4060 is used in place of the MC14011UB. The data output of the MC145026 is inverted and fed to the RESET pin of the MC74HC4060. Alternately, the MC74HCU04 could be used for the oscillator.

Information on the MC14011UB is in book number DL131/D. The MC74HCU04 and MC74HC4060 are found in book number DL129/D.

INFRARED RECEIVER

The receiver in Figure 20 couples an IR–sensitive diode to input preamp A1, followed by band–pass amplifier A2 with a gain of about 10. Limiting stage A3 follows, with an output of about 800 mV p–p. The limited 50 kHz burst is detected by comparator A4 that passes only positive pulses, and peak–detected and filtered by a diode/RC network to extract the data envelope from the burst. Comparator A5 boosts the signal to logic levels compatible with the MC145027/28 data input. The D_{in} pin of these decoders is a standard CMOS high–impedance input which must ***not*** be allowed to float. Therefore, direct coupling from A5 to the decoder input is utilized.

Shielding should be used on at least A1 and A2, with good ground and high–sensitivity circuit layout techniques applied.

For operation with supplies higher than + 5 V, limiter A4's positive output swing needs to be limited to 3 to 5 V. This is accomplished via adding a zener diode in the negative feedback path, thus avoiding excessive system noise. The biasing resistor stack should be adjusted such that V3 is 1.25 to 1.5 V.

This system works up to a range of about 10 meters. The gains of the system may be adjusted to suit the individual design needs. The 100 Ω resistor in the emitter of the first 2N5088 and the 1 kΩ resistor feeding A2 may be altered if different gain is required. In general, more gain does not necessarily result in increased range. This is due to noise floor limitations. The designer should increase transmitter power and/or increase receiver aperature with Fresnal lensing to greatly improve range. See Application Note AN1016 for additional information.

Information on the MC34074 is in data book DL128/D.

TRINARY SWITCH MANUFACTURERS

Midland Ross–Electronic Connector Div.
617/491–5400
Greyhill
312/354–1040
Augat/Alcoswitch
617/685–4371
Aries Electronics
201/996–6841

The above companies may not have the switches in a DIP. For more information, call them or consult *eem Electronic Engineers Master Catalog* or the *Gold Book.* ***Ask for SPDT with center OFF.***

Alternative: An SPST can be placed in series between a SPDT and the Encoder or Decoder to achieve trinary action.

Motorola cannot recommend one supplier over another and in no way suggests that this is a complete listing of trinary switch manufacturers.

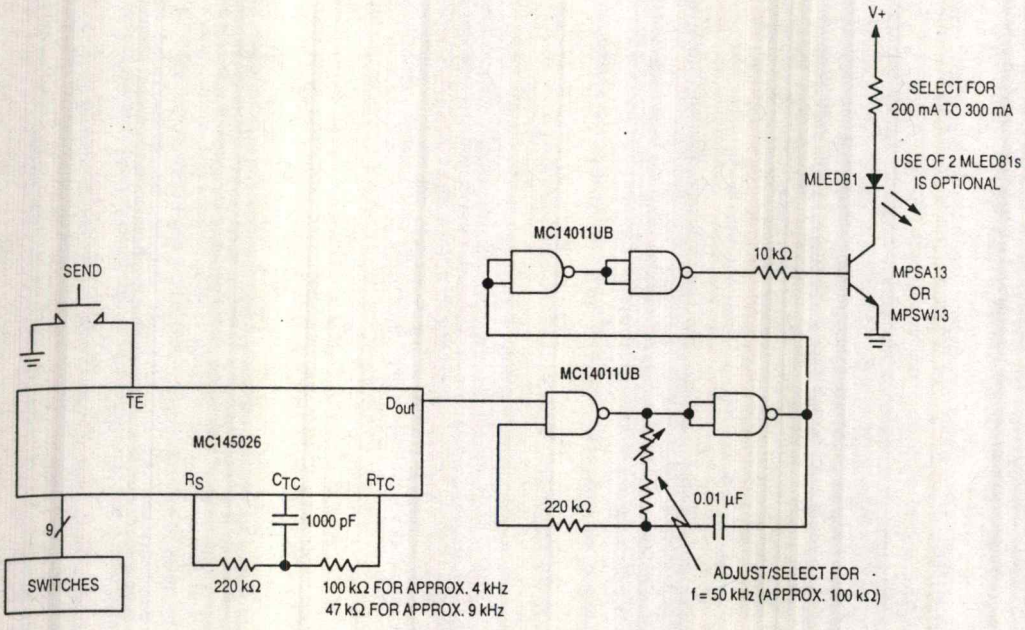


Figure 18. IRED Transmitter Using RC Oscillator to Generate Carrier Frequency

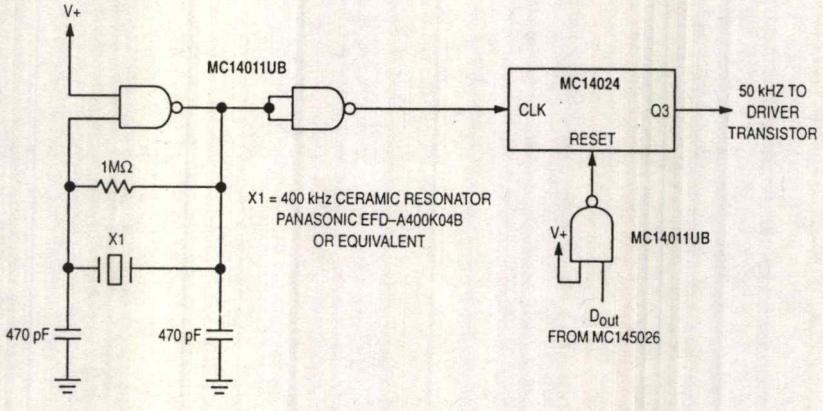


Figure 19. Using a Ceramic Resonator to Generate Carrier Frequency

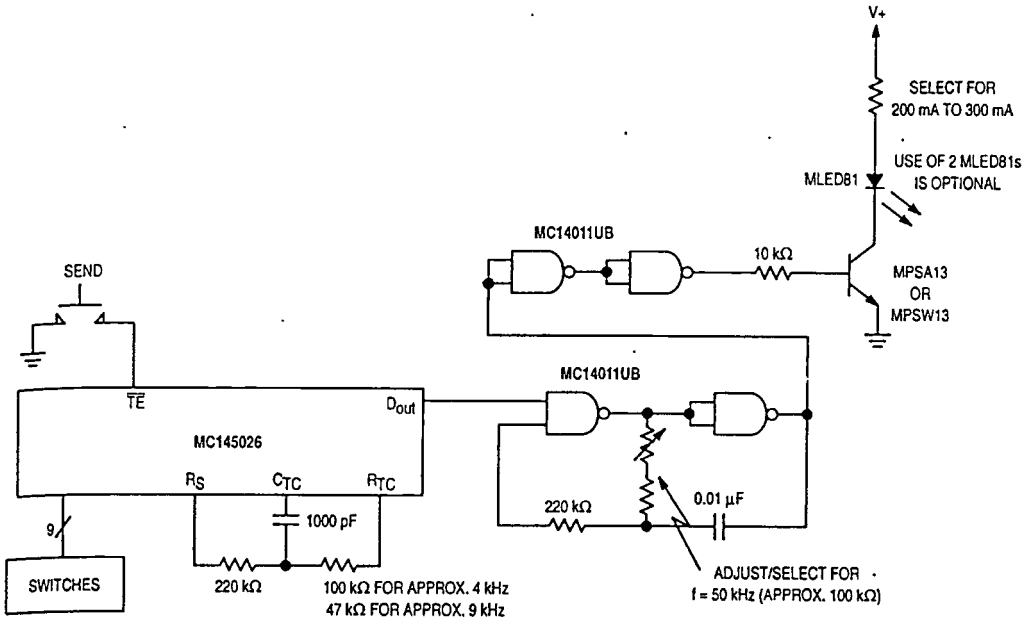


Figure 18. IRED Transmitter Using RC Oscillator to Generate Carrier Frequency

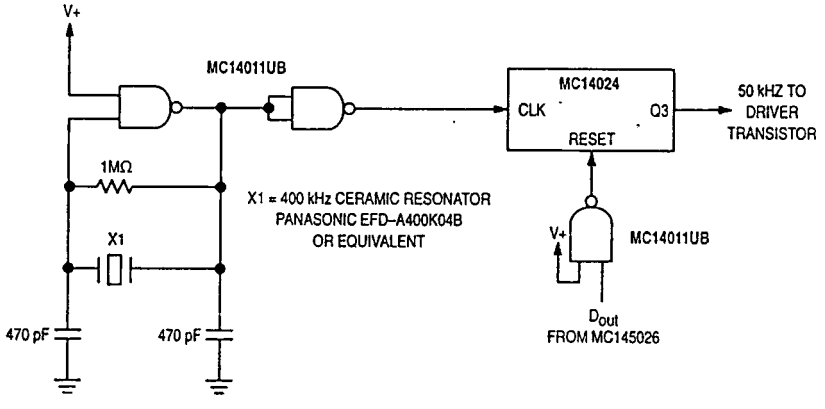


Figure 19. Using a Ceramic Resonator to Generate Carrier Frequency

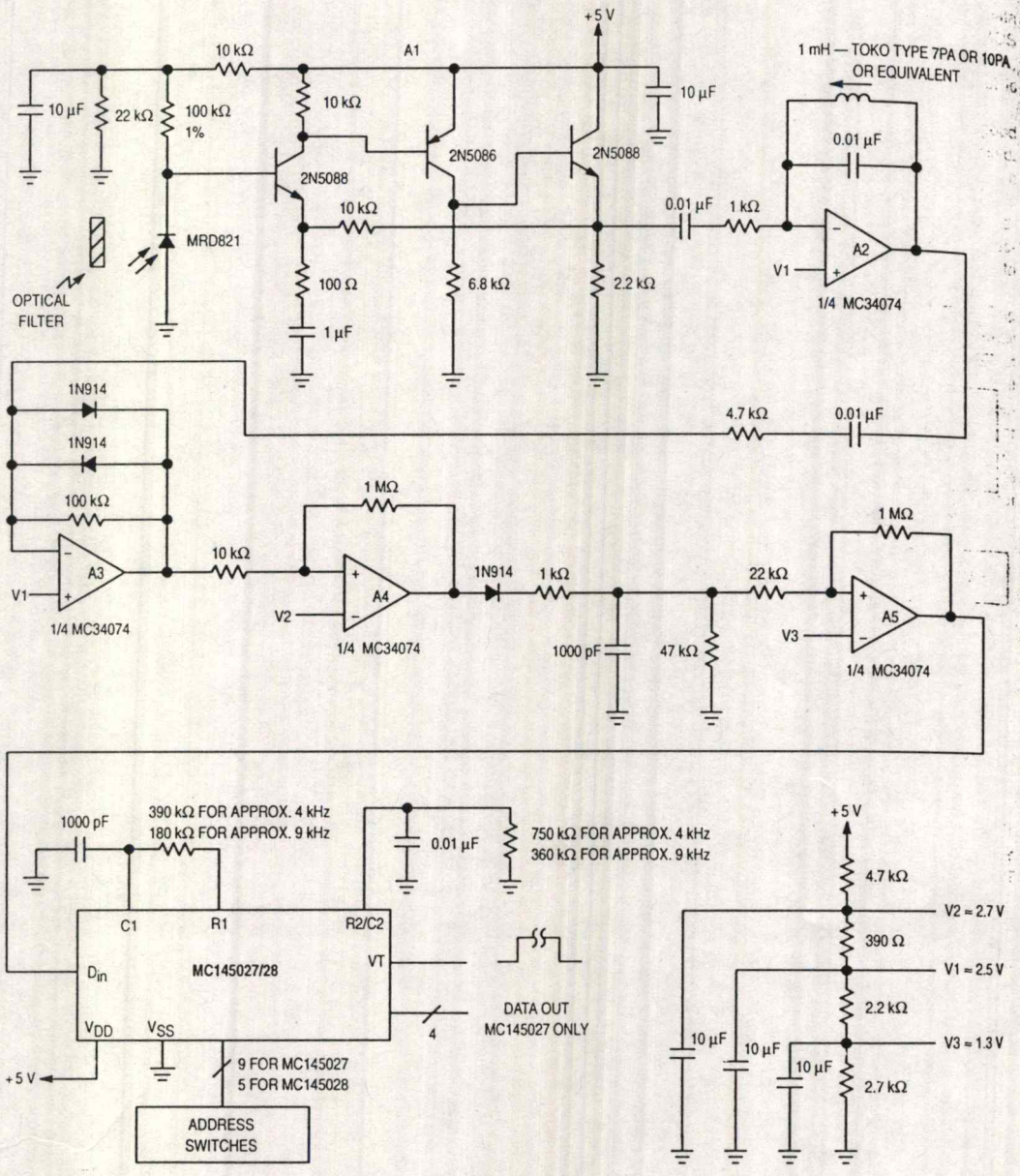


Figure 20. Infrared Receiver

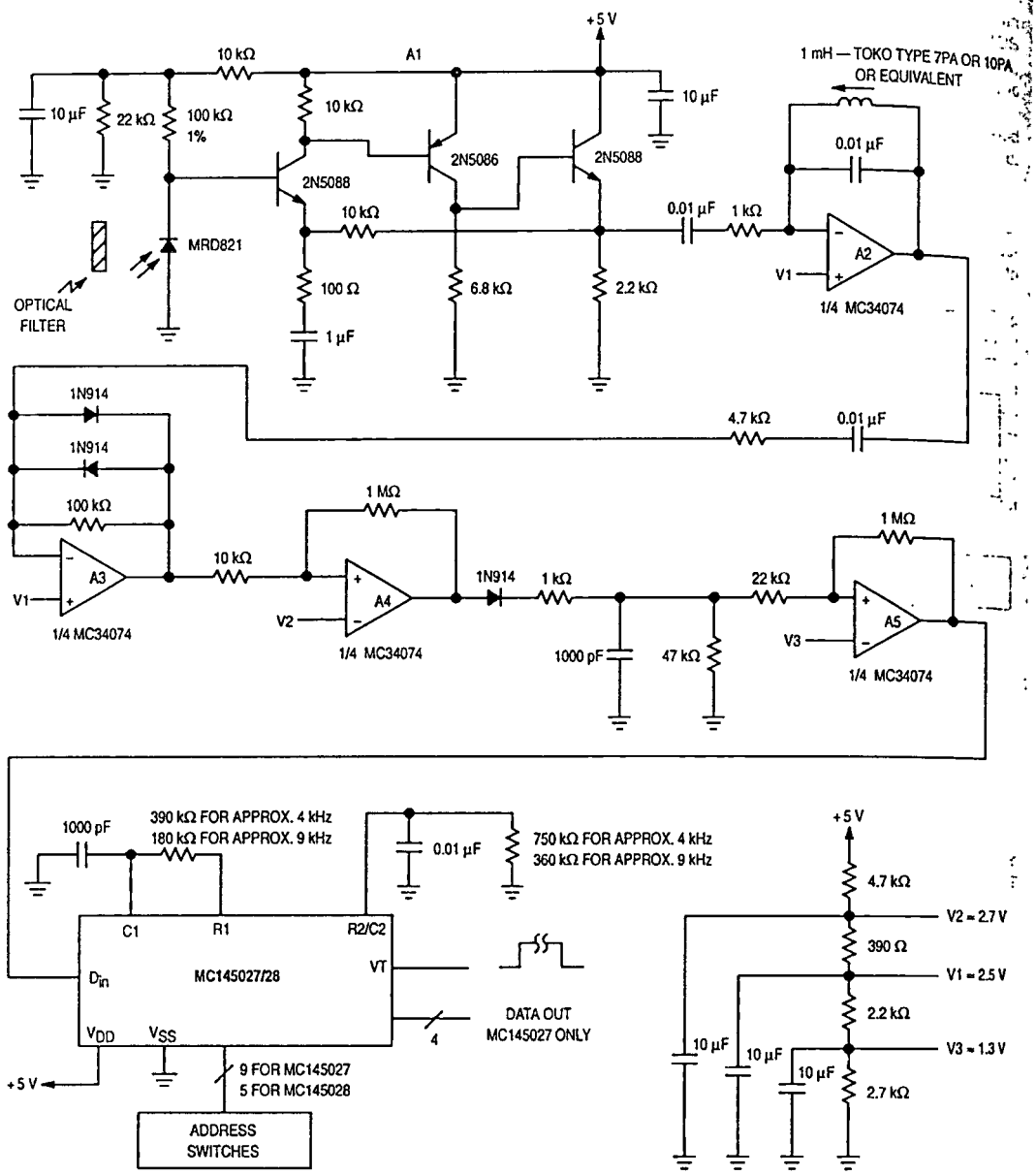


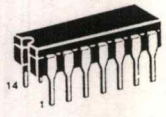
Figure 20. Infrared Receiver

MOTOROLA

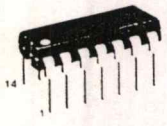
B-SUFFIX SERIES CMOS GATES

The B Series logic gates are constructed with P and N channel enhancement mode devices in a single monolithic structure (Complementary MOS). Their primary use is where low power dissipation and/or high noise immunity is desired.

- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- All Outputs Buffered
- Capable of Driving Two Low-power TTL Loads or One Low-power Schottky TTL Load Over the Rated Temperature Range.
- Double Diode Protection on All Inputs Except: Triple Diode Protection on MC14011B and MC14081B
- Pin-for-Pin Replacements for Corresponding CD4000 Series B Suffix Devices (Exceptions: MC14068B and MC14078B)



L SUFFIX
CERAMIC
CASE 632



P SUFFIX
PLASTIC
CASE 646

D SUFFIX
SOIC
CASE 751A

ORDERING INFORMATION

MC14XXXBCP	Plastic
MC14XXXBCL	Ceramic
MC14XXXBD	SOIC

T_A = −55 to 125 C for all packages.

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

Symbol	Parameter	Value	Unit
V_{DD}	DC Supply Voltage	−0.5 to +18.0	V
V_{in}, V_{out}	Input or Output Voltage (DC or Transient)	−0.5 to V_{DD} +0.5	V
I_{in}, I_{out}	Input or Output Current (DC or Transient), per Pin	±10	mA
P_D	Power Dissipation, per Package†	500	mW
T_{stg}	Storage Temperature	−65 to +150	°C
T_L	Lead Temperature (8-Second Soldering)	260	°C

*Maximum Ratings are those values beyond which damage to the device may occur.
†Temperature Derating: Plastic "P and D/DW" Packages: −7.0 mW/°C From 65°C To 125°C
Ceramic "L" Packages: −12 mW/°C From 100°C To 125°C

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.
Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}). Unused outputs must be left open.

MC14001B
Quad 2-Input NOR Gate

MC14002B
Dual 4-Input Nor Gate

MC14011B
Quad 2-Input NAND Gate

MC14012B
Dual 4-Input NAND Gate

MC14023B
Triple 3-Input NAND Gate

MC14025B
Triple 3-Input NOR Gate

MC14068B
8-Input NAND Gate

MC14071B
Quad 2-Input OR Gate

MC14072B
Dual 4-Input OR Gate

MC14073B
Triple 3-Input AND Gate

MC14075B
Triple 3-Input OR Gate

MC14078B
8-Input NOR Gate

MC14081B
Quad 2-Input AND Gate

MC14082B
Dual 4-Input AND Gate

MOTOROLA

MC14001B
Quad 2-Input NOR Gate

MC14002B
Dual 4-Input Nor Gate

MC14011B
Quad 2-Input NAND Gate

MC14012B
Dual 4-Input NAND Gate

MC14023B
Triple 3-Input NAND Gate

MC14025B
Triple 3-Input NOR Gate

MC14068B
8-Input NAND Gate

MC14071B
Quad 2-Input OR Gate

MC14072B
Dual 4-Input OR Gate

MC14073B
Triple 3-Input AND Gate

MC14075B
Triple 3-Input OR Gate

MC14078B
8-Input NOR Gate

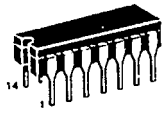
MC14081B
Quad 2-Input AND Gate

MC14082B
Dual 4-Input AND Gate

B-SUFFIX SERIES CMOS GATES

The B Series logic gates are constructed with P and N channel enhancement mode devices in a single monolithic structure (Complementary MOS). Their primary use is where low power dissipation and/or high noise immunity is desired.

- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- All Outputs Buffered
- Capable of Driving Two Low-power TTL Loads or One Low-power Schottky TTL Load Over the Rated Temperature Range.
- Double Diode Protection on All Inputs Except: Triple Diode Protection on MC14011B and MC14081B
- Pin-for-Pin Replacements for Corresponding CD4000 Series B Suffix Devices (Exceptions: MC14068B and MC14078B)



L SUFFIX
CERAMIC
CASE 632

P SUFFIX
PLASTIC
CASE 646

D SUFFIX
SOIC
CASE 751A

ORDERING INFORMATION

MC14XXXBCP Plastic
MC14XXXBCL Ceramic
MC14XXXBD SOIC

T_A 55 to 125 C for all packages

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

Symbol	Parameter	Value	Unit
V_{DD}	DC Supply Voltage	- 0.5 to + 18.0	V
V_{in}, V_{out}	Input or Output Voltage (DC or Transient)	- 0.5 to V_{DD} + 0.5	V
I_{in}, I_{out}	Input or Output Current (DC or Transient) per Pin	± 10	mA
P_D	Power Dissipation, per Package†	500	mW
T_{stg}	Storage Temperature	- 65 to + 150	°C
T_L	Lead Temperature (8-Second Soldering)	260	°C

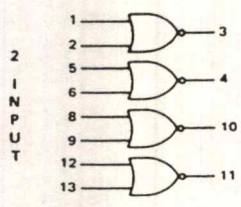
*Maximum Ratings are those values beyond which damage to the device may occur
†Temperature Derating: Plastic "P and D/DW" Packages - 7.0 mW/°C From 65°C To 125°C
Ceramic "L" Packages: - 12 mW/°C From 100°C To 125°C

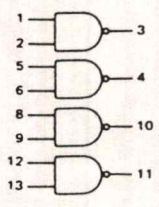
This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.
Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}). Unused outputs must be left open.

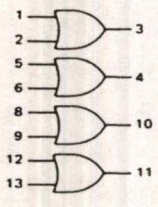
CMOS B-SERIES GATES

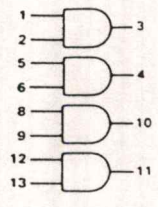
LOGIC DIAGRAMS

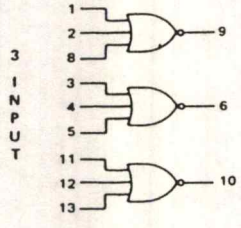
	NOR	NAND	OR	AND
2 INPUT	MC14001B Quad 2-Input NOR Gate	MC14011B Quad 2-Input NAND Gate	MC14071B Quad 2-Input OR Gate	MC14081B Quad 2-Input AND Gate
3 INPUT	MC14025B Triple 3-Input NOR Gate	MC14023B Triple 3-Input NAND Gate	MC14075B Triple 3-Input OR Gate	MC14073B Triple 3-Input AND Gate
4 INPUT	MC14002B Dual 4-Input NOR Gate	MC14012B Dual 4-Input NAND Gate	MC14072B Dual 4-Input OR Gate	MC14082B Dual 4-Input AND Gate
8 INPUT	MC14078B 8-Input NOR Gate	MC14068B 8-Input NAND Gate		

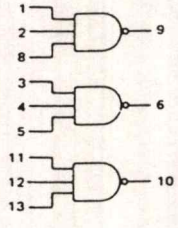


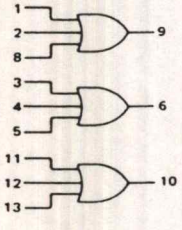


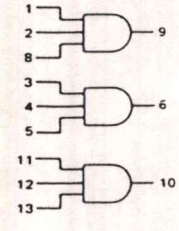


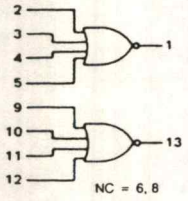


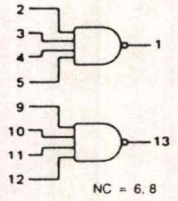


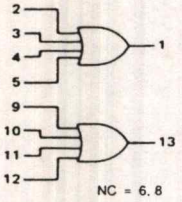


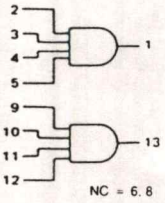


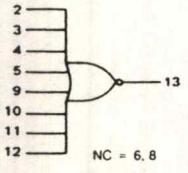


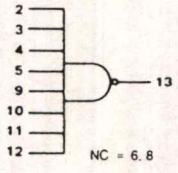








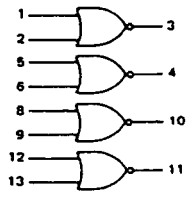
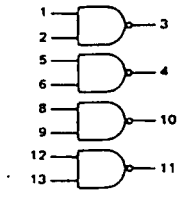
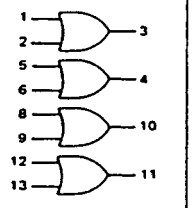
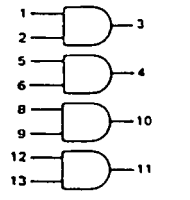
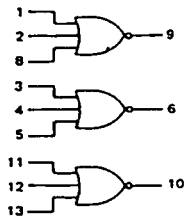
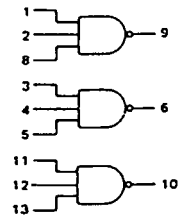
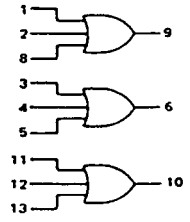
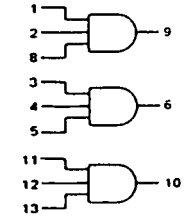
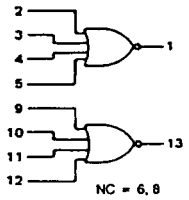
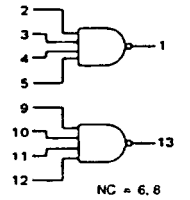
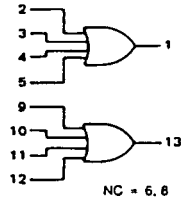
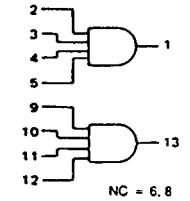
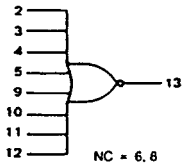
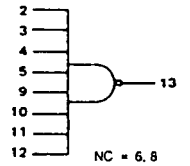




V_{DD} = Pin 14
V_{SS} = Pin 7
for All Devices

CMOS B-SERIES GATES

LOGIC DIAGRAMS

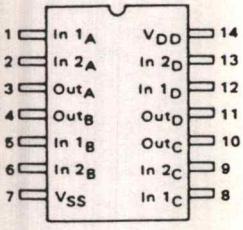
	NOR	NAND	OR	AND
2 INPUT	MC14001B Quad 2-Input NOR Gate	MC14011B Quad 2-Input NAND Gate	MC14071B Quad 2-Input OR Gate	MC14081B Quad 2-Input AND Gate
				
3 INPUT	MC14025B Triple 3-Input NOR Gate	MC14023B Triple 3-Input NAND Gate	MC14075B Triple 3-Input OR Gate	MC14073B Triple 3-Input AND Gate
				
4 INPUT	MC14002B Dual 4-Input NOR Gate	MC14012B Dual 4-Input NAND Gate	MC14072B Dual 4-Input OR Gate	MC14082B Dual 4-Input AND Gate
				
8 INPUT	MC14078B 8-Input NOR Gate	MC14068B 8-Input NAND Gate		
				

V_{DD} = Pin 14
V_{SS} = Pin 7
for All Devices

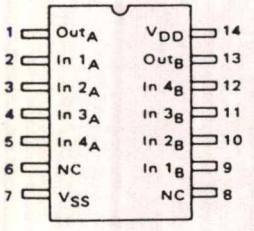
CMOS B-SERIES GATES

PIN ASSIGNMENTS

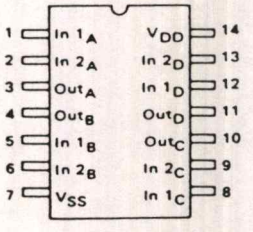
MC14001B
Quad 2-Input NOR Gate



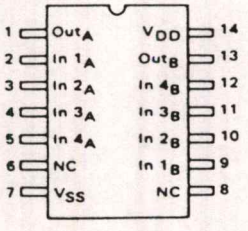
MC14002B
Dual 4-Input NOR Gate



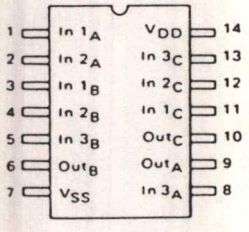
MC14011B
Quad 2-Input NAND Gate



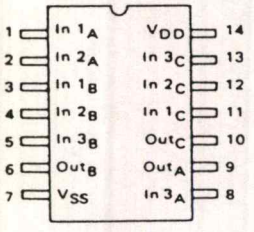
MC14012B
Dual 4-Input NAND Gate



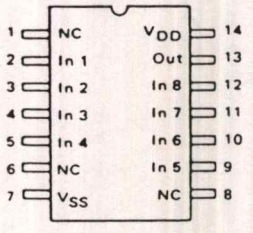
MC14023B
Triple 3-Input NAND Gate



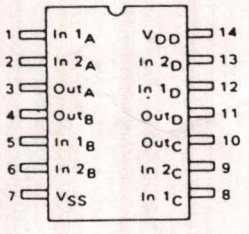
MC14025B
Triple 3-Input NOR Gate



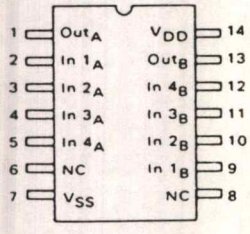
MC14068B
8-Input NAND Gate



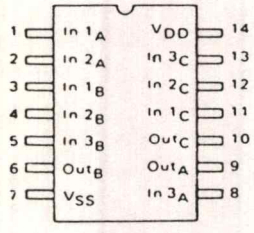
MC14071B
Quad 2-Input OR Gate



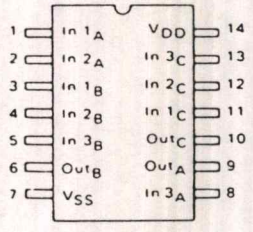
MC14072B
Dual 4-Input OR Gate



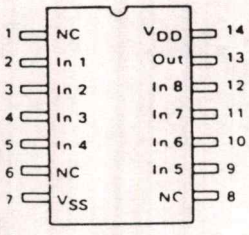
MC14073B
Triple 3-Input AND Gate



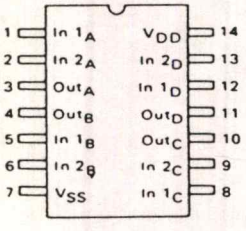
MC14075B
Triple 3-Input OR Gate



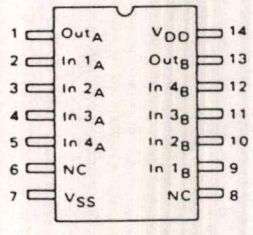
MC14078B
8-Input NOR Gate



MC14081B
Quad 2-Input AND Gate



MC14082B
Dual 4-Input AND Gate

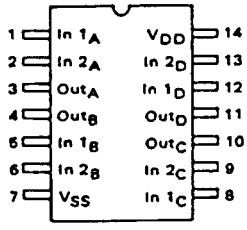


NC = No Connection

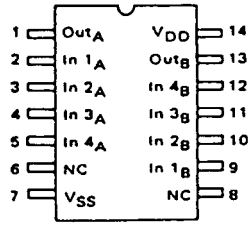
CMOS B-SERIES GATES

PIN ASSIGNMENTS

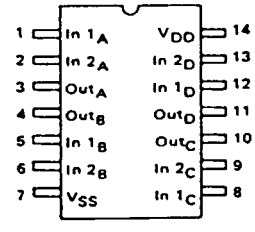
MC14001B
Quad 2-Input NOR Gate



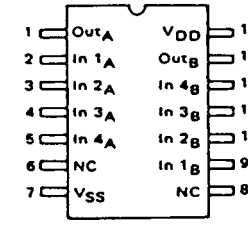
MC14002B
Dual 4-Input NOR Gate



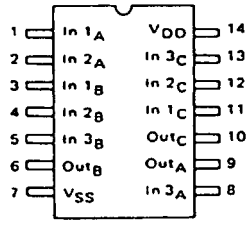
MC14011B
Quad 2-Input NAND Gate



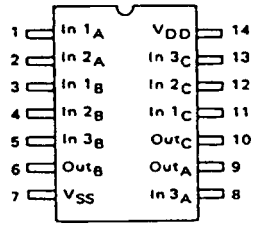
MC14012B
Dual 4-Input NAND Gate



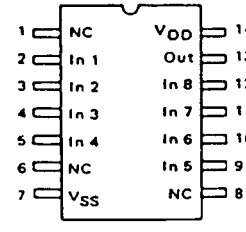
MC14023B
Triple 3-Input NAND Gate



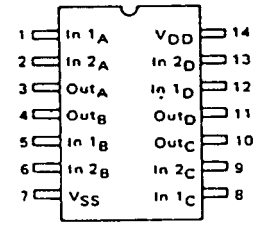
MC14025B
Triple 3-Input NOR Gate



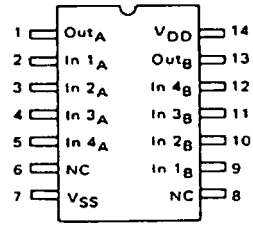
MC14068B
8-Input NAND Gate



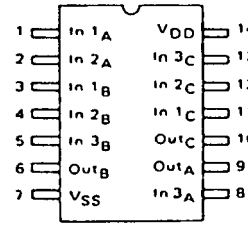
MC14071B
Quad 2-Input OR Gate



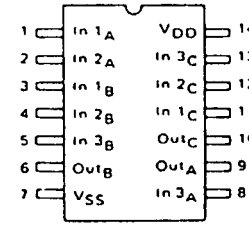
MC14072B
Dual 4-Input OR Gate



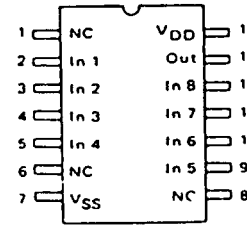
MC14073B
Triple 3-Input AND Gate



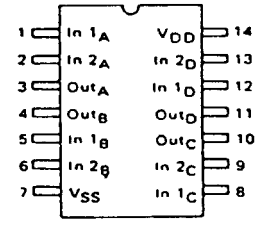
MC14075B
Triple 3-Input OR Gate



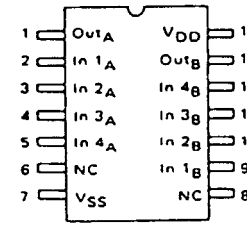
MC14078B
8-Input NOR Gate



MC14081B
Quad 2-Input AND Gate



MC14082B
Dual 4-Input AND Gate



NC = No Connection

ELECTRICAL CHARACTERISTICS (Voltages Referenced to V_{SS})

Characteristic	Symbol	V_{DD} Vdc	−55°C Min	−55°C Max	25°C Min	25°C Typ #	25°C Max	125°C Min	125°C Max	Unit
Output Voltage "0" Level V_{in} = V_{DD} or 0	V_{OL}	5.0	—	0.05	—	0	0.05	—	0.05	Vdc
		10	—	0.05	—	0	0.05	—	0.05	
		15	—	0.05	—	0	0.05	—	0.05	
"1" Level V_{in} = 0 or V_{DD}	V_{OH}	5.0	4.95	—	4.95	5.0	—	4.95	—	Vdc
		10	9.95	—	9.95	10	—	9.95	—	
		15	14.95	—	14.95	15	—	14.95	—	
Input Voltage "0" Level (V_O = 4.5 or 0.5 Vdc)	V_{IL}	5.0	—	1.5	—	2.25	1.5	—	1.5	Vdc
(V_O = 9.0 or 1.0 Vdc)		10	—	3.0	—	4.50	3.0	—	3.0	
(V_O = 13.5 or 1.5 Vdc)		15	—	4.0	—	6.75	4.0	—	4.0	
"1" Level (V_O = 0.5 or 4.5 Vdc)	V_{IH}	5.0	3.5	—	3.5	2.75	—	3.5	—	Vdc
(V_O = 1.0 or 9.0 Vdc)		10	7.0	—	7.0	5.50	—	7.0	—	
(V_O = 1.5 or 13.5 Vdc)		15	11	—	11	8.25	—	11	—	
Output Drive Current (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	−3.0	—	−2.4	−4.2	—	−1.7	—	mAdc
(V_{OH} = 4.6 Vdc)		5.0	−0.64	—	−0.51	−0.88	—	−0.36	—	
(V_{OH} = 9.5 Vdc)		10	−1.6	—	−1.3	−2.25	—	−0.9	—	
(V_{OH} = 13.5 Vdc)		15	−4.2	—	−3.4	−8.8	—	−2.4	—	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	—	0.51	0.88	—	0.36	—	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	—	1.3	2.25	—	0.9	—	
(V_{OL} = 1.5 Vdc)		15	4.2	—	3.4	8.8	—	2.4	—	
Input Current	I_{in}	15	—	±0.1	—	±0.00001	±0.1	—	±1.0	μAdc
Input Capacitance (V_{in} = 0)	C_{in}	—	—	—	—	5.0	7.5	—	—	pF
Quiescent Current (Per Package)	I_{DD}	5.0	—	0.25	—	0.0005	0.25	—	7.5	μAdc
		10	—	0.5	—	0.0010	0.5	—	15	
		15	—	1.0	—	0.0015	1.0	—	30	
Total Supply Current**† (Dynamic plus Quiescent, Per Gate, C_L = 50 pF)	I_T	5.0	I_T = (0.3 μA/kHz) f + I_{DD}/N							μAdc
		10	I_T = (0.6 μA/kHz) f + I_{DD}/N							
		15	I_T = (0.9 μA/kHz) f + I_{DD}/N							

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

**The formulas given are for the typical characteristics only at 25°C.

†To calculate total supply current at loads other than 50 pF:

$$I_T(C_L) = I_T(50\ pF) + (C_L - 50)\ Vfk$$

where: I_T is in μA (per package), C_L in pF, V = (V_{DD} − V_{SS}) in volts, f in kHz is input frequency, and k = 0.001 × the number of exercised gates per package.

ELECTRICAL CHARACTERISTICS (Voltages Referenced to V_{SS})

Characteristic	Symbol	V_{DD} Vdc	−55°C Min	−55°C Max	25°C Min	25°C Typ #	25°C Max	125°C Min	125°C Max	Unit
Output Voltage "0" Level V_{in} = V_{DD} or 0	V_{OL}	5.0	—	0.05	—	0	0.05	—	0.05	Vdc
		10	—	0.05	—	0	0.05	—	0.05	
		15	—	0.05	—	0	0.05	—	0.05	
"1" Level V_{in} = 0 or V_{DD}	V_{OH}	5.0	4.95	—	4.95	5.0	—	4.95	—	Vdc
		10	9.95	—	9.95	10	—	9.95	—	
		15	14.95	—	14.95	15	—	14.95	—	
Input Voltage "0" Level (V_O = 4.5 or 0.5 Vdc)	V_{IL}	5.0	—	1.5	—	2.25	1.5	—	1.5	Vdc
(V_O = 9.0 or 1.0 Vdc)		10	—	3.0	—	4.50	3.0	—	3.0	
(V_O = 13.5 or 1.5 Vdc)		15	—	4.0	—	6.75	4.0	—	4.0	
"1" Level (V_O = 0.5 or 4.5 Vdc)	V_{IH}	5.0	3.5	—	3.5	2.75	—	3.5	—	Vdc
(V_O = 1.0 or 9.0 Vdc)		10	7.0	—	7.0	5.50	—	7.0	—	
(V_O = 1.5 or 13.5 Vdc)		15	11	—	11	8.25	—	11	—	
Output Drive Current (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	−3.0	—	−2.4	−4.2	—	−1.7	—	mAdc
(V_{OH} = 4.6 Vdc)		5.0	−0.64	—	−0.51	−0.88	—	−0.36	—	
(V_{OH} = 9.5 Vdc)		10	−1.6	—	−1.3	−2.25	—	−0.9	—	
(V_{OH} = 13.5 Vdc)		15	−4.2	—	−3.4	−8.8	—	−2.4	—	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	—	0.51	0.88	—	0.36	—	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	—	1.3	2.25	—	0.9	—	
(V_{OL} = 1.5 Vdc)		15	4.2	—	3.4	8.8	—	2.4	—	
Input Current	I_{in}	15	—	±0.1	—	±0.00001	±0.1	—	±1.0	μAdc
Input Capacitance (V_{in} = 0)	C_{in}	—	—	—	—	5.0	7.5	—	—	pF
Quiescent Current (Per Package)	I_{DD}	5.0	—	0.25	—	0.0005	0.25	—	7.5	μAdc
		10	—	0.5	—	0.0010	0.5	—	15	
		15	—	1.0	—	0.0015	1.0	—	30	
Total Supply Current**† (Dynamic plus Quiescent, Per Gate, C_L = 50 pF)	I_T	5.0	I_T = (0.3 μA/kHz) f + I_{DD}/N							μAdc
		10	I_T = (0.6 μA/kHz) f + I_{DD}/N							
		15	I_T = (0.9 μA/kHz) f + I_{DD}/N							

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance

**The formulas given are for the typical characteristics only at 25°C.

†To calculate total supply current at loads other than 50 pF:

$$I_T(C_L) = I_T(50\ pF) + (C_L - 50)\ Vfk$$

where, I_T is in μA (per package), C_L in pF, V = (V_{DD} − V_{SS}) in volts, f in kHz is input frequency, and k = 0.001 × the number of exercised gates per package.

B-SERIES GATE SWITCHING TIMES

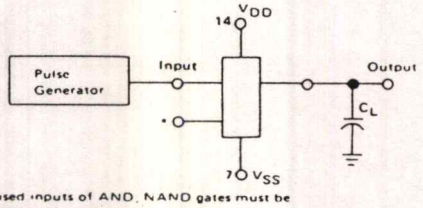
SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD} Vdc	Min	Typ #	Max	Unit
Output Rise Time, All B-Series Gates	t_{TLH}					ns
t_{TLH} = (1.35 ns/pF) C_L + 33 ns		5.0	—	100	200	
t_{TLH} = (0.60 ns/pF) C_L + 20 ns		10	—	50	100	
t_{TLH} = (0.40 ns/pF) C_L + 20 ns		15	—	40	80	
Output Fall Time, All B-Series Gates	t_{THL}					ns
t_{THL} = (1.35 ns/pF) C_L + 33 ns		5.0	—	100	200	
t_{THL} = (0.60 ns/pF) C_L + 20 ns		10	—	50	100	
t_{THL} = (0.40 ns/pF) C_L + 20 ns		15	—	40	80	
Propagation Delay Time	t_{PLH}, t_{PHL}					ns
MC14001B, MC14011B only						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 80 ns		5.0	—	125	250	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 32 ns		10	—	50	100	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 27 ns		15	—	40	80	
All Other 2, 3, and 4 Input Gates						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 115 ns		5.0	—	160	300	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 47 ns		10	—	65	130	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 37 ns		15	—	50	100	
8-Input Gates (MC14068B, MC14078B)						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 155 ns		5.0	—	200	350	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 62 ns		10	—	80	150	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 47 ns		15	—	60	110	

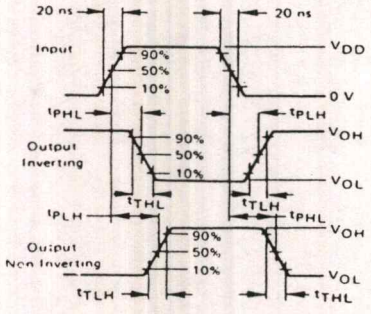
*The formulas given are for the typical characteristics only at 25°C.

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

FIGURE 1 – SWITCHING TIME TEST CIRCUIT AND WAVEFORMS



*All unused inputs of AND, NAND gates must be connected to V_{DD}.
All unused inputs of OR, NOR gates must be connected to V_{SS}.



B-SERIES GATE SWITCHING TIMES

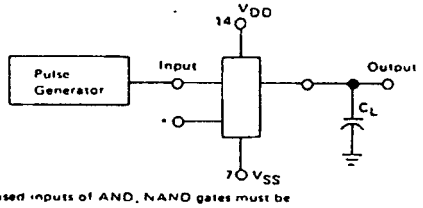
SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD} Vdc	Min	Typ #	Max	Unit
Output Rise Time, All B-Series Gates	t_{TLH}					ns
t_{TLH} = (1.35 ns/pF) C_L + 33 ns		5.0	—	100	200	
t_{TLH} = (0.60 ns/pF) C_L + 20 ns		10	—	50	100	
t_{TLH} = (0.40 ns/pF) C_L + 20 ns		15	—	40	80	
Output Fall Time, All B-Series Gates	t_{THL}					ns
t_{THL} = (1.35 ns/pF) C_L + 33 ns		5.0	—	100	200	
t_{THL} = (0.60 ns/pF) C_L + 20 ns		10	—	50	100	
t_{THL} = (0.40 ns/pF) C_L + 20 ns		15	—	40	80	
Propagation Delay Time	t_{PLH}, t_{PHL}					ns
MC140018, MC140118 only						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 80 ns		5.0	—	125	250	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 32 ns		10	—	50	100	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 27 ns		15	—	40	80	
All Other 2, 3, and 4 Input Gates						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 115 ns		5.0	—	160	300	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 47 ns		10	—	65	130	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 37 ns		15	—	50	100	
8 Input Gates (MC140688, MC140788)						
t_{PLH}, t_{PHL} = (0.90 ns/pF) C_L + 155 ns		5.0	—	200	350	
t_{PLH}, t_{PHL} = (0.36 ns/pF) C_L + 62 ns		10	—	80	150	
t_{PLH}, t_{PHL} = (0.26 ns/pF) C_L + 47 ns		15	—	60	110	

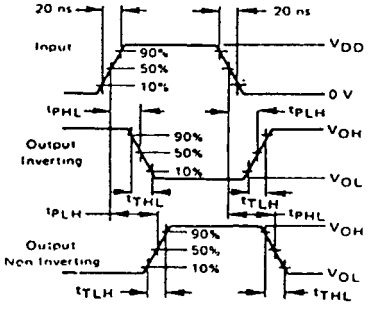
*The formulas given are for the typical characteristics only at 25°C.

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance

FIGURE 1 – SWITCHING TIME TEST CIRCUIT AND WAVEFORMS



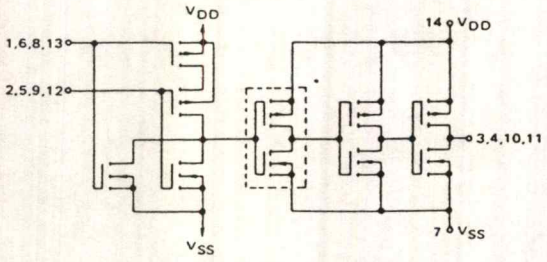
*All unused inputs of AND, NAND gates must be connected to V_{DD}
All unused inputs of OR, NOR gates must be connected to V_{SS}.



CMOS B-SERIES GATES

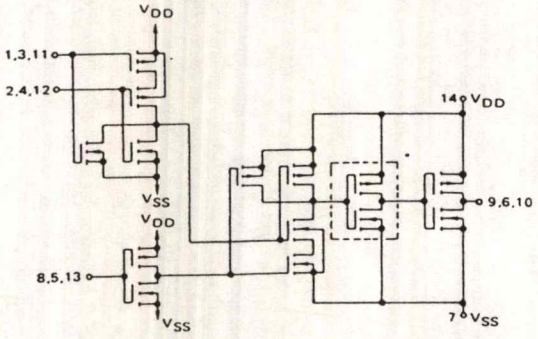
CIRCUIT SCHEMATIC
NOR, OR Gates

MC14001B
MC14071B
One of Four
Gates Shown



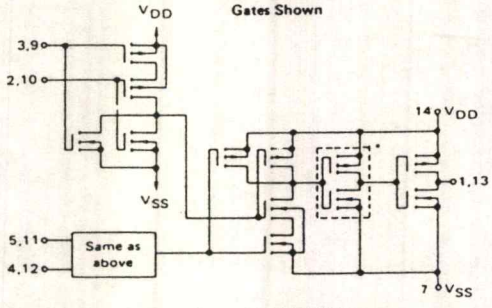
*Inverter Omitted in MC14001B

MC14025B
MC14075B
One of Three
Gates Shown

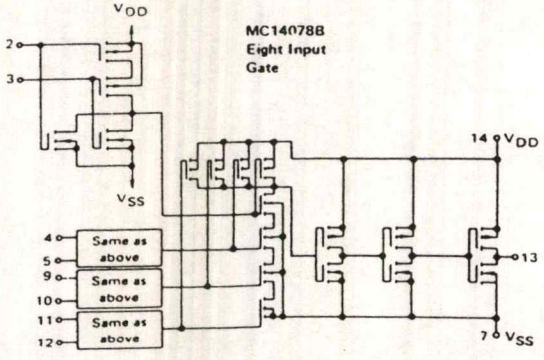


*Inverter Omitted in MC14025B

MC14002B
MC14072B
One of Two
Gates Shown



*Inverter Omitted in MC14002B



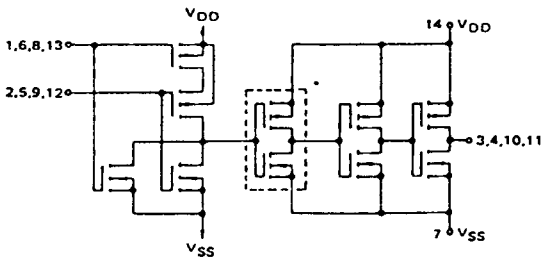
MC14078B
Eight Input
Gate

6

CMOS B-SERIES GATES

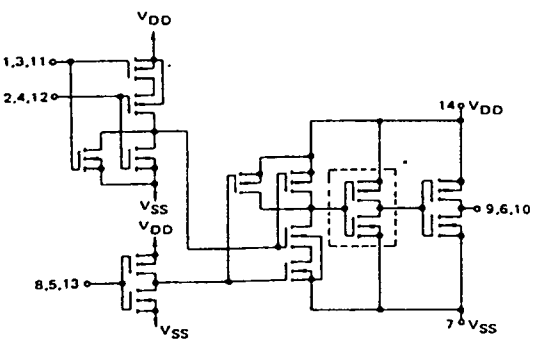
CIRCUIT SCHEMATIC
NOR, OR Gates

MC14001B
MC14071B
One of Four
Gates Shown



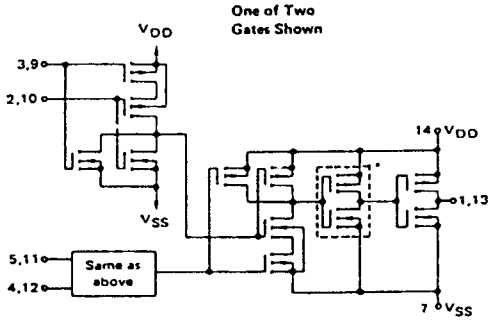
*Inverter Omitted in MC14001B

MC14025B
MC14075B
One of Three
Gates Shown

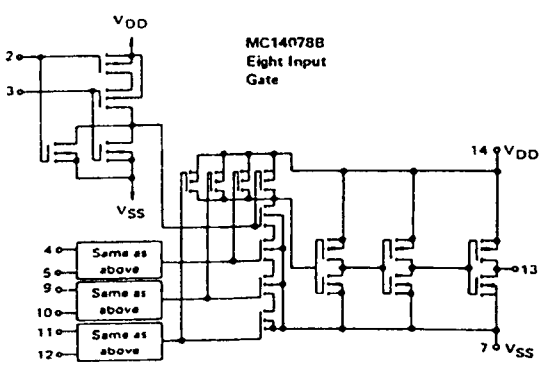


*Inverter Omitted in MC14025B

MC14002B
MC14072B
One of Two
Gates Shown



*Inverter Omitted in MC14002B

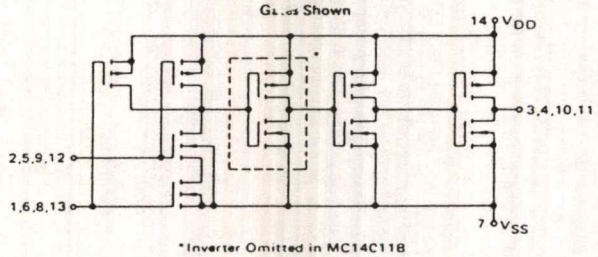


6

CMOS B-SERIES GATES

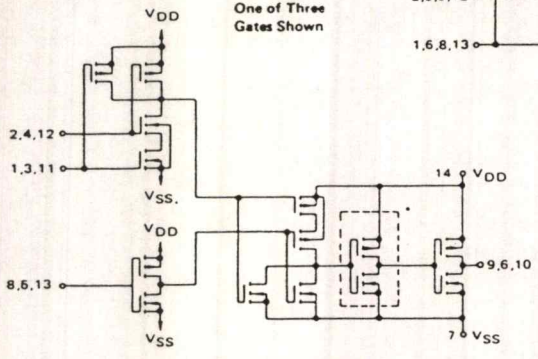
CIRCUIT SCHEMATICS
NAND, AND Gates

MC14011B
MC14081B
One of Four
Gates Shown



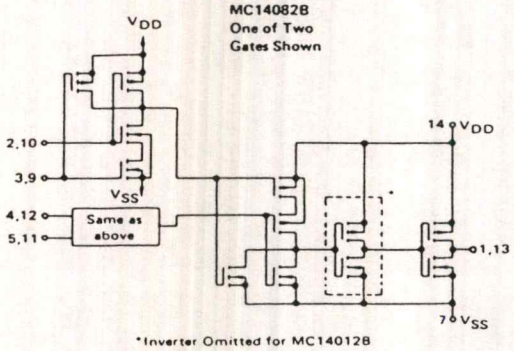
*Inverter Omitted in MC14C11B

MC14023B
MC14073B
One of Three
Gates Shown



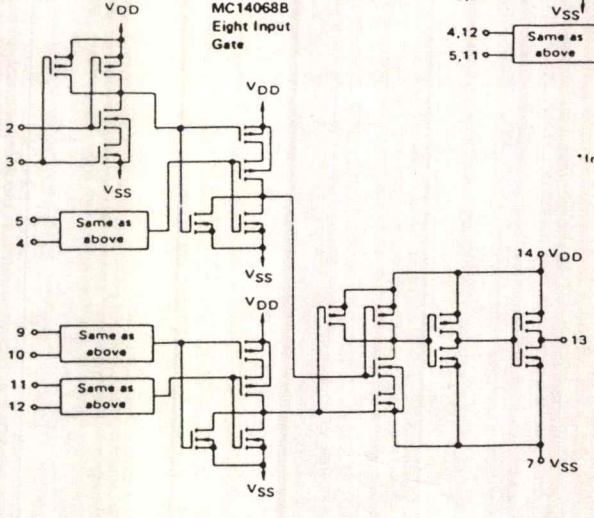
*Inverter Omitted in MC14023B

MC14012B
MC14082B
One of Two
Gates Shown



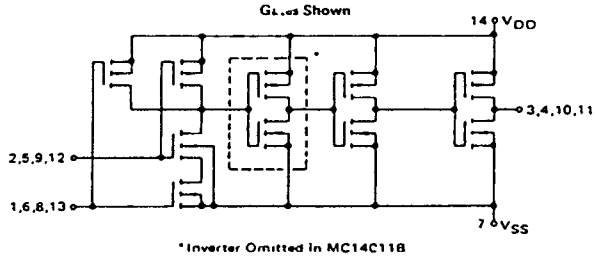
*Inverter Omitted for MC14012B

MC14068B
Eight Input
Gate



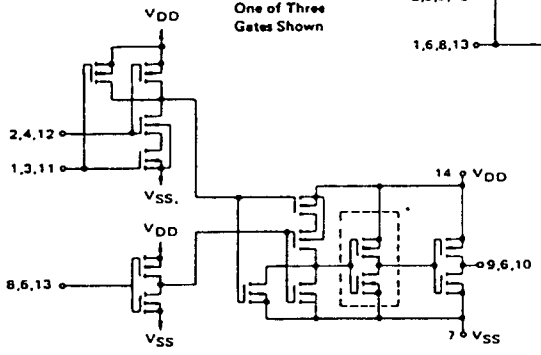
CIRCUIT SCHEMATICS
NAND, AND Gates

MC14011B
MC14081B
One of Four
Gates Shown



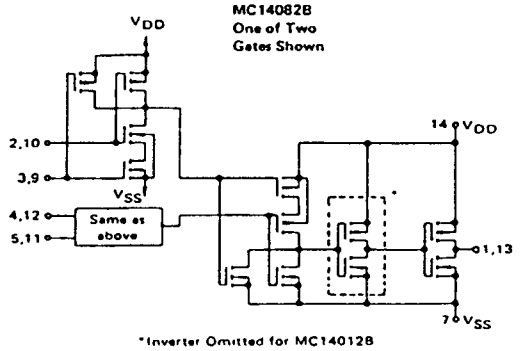
*Inverter Omitted in MC14C11B

MC14023B
MC14073B
One of Three
Gates Shown



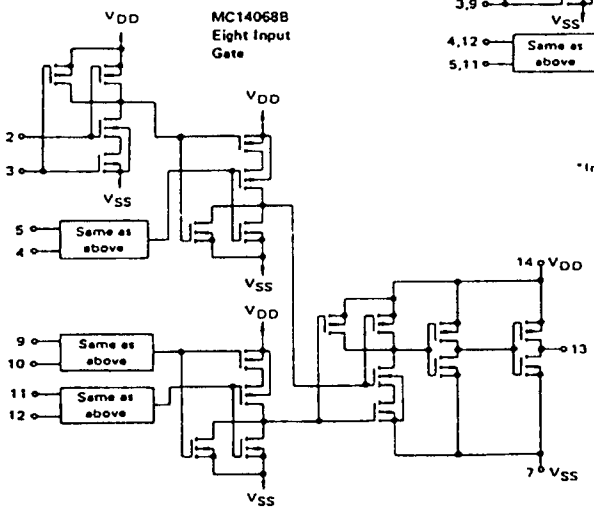
*Inverter Omitted in MC14023B

MC14012B
MC14082B
One of Two
Gates Shown



*Inverter Omitted for MC14012B

MC14068B
Eight Input
Gate



TYPICAL B-SERIES GATE CHARACTERISTICS

N-CHANNEL DRAIN CURRENT (SINK)

FIGURE 2 – V_{GS} = 5.0 Vdc

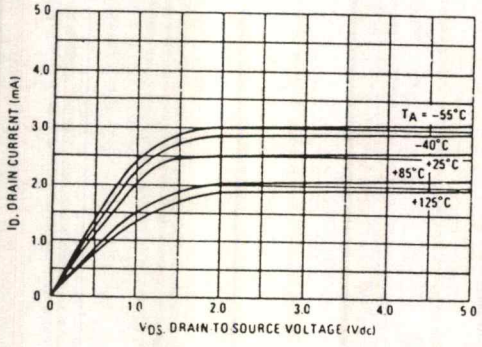


FIGURE 4 – V_{GS} = 10 Vdc

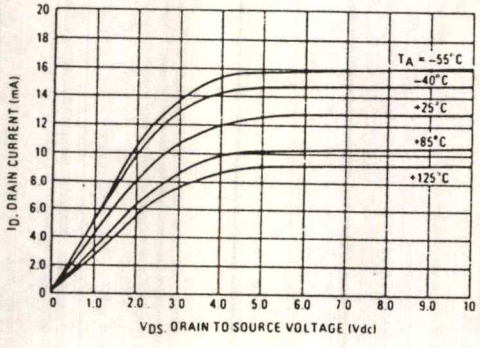
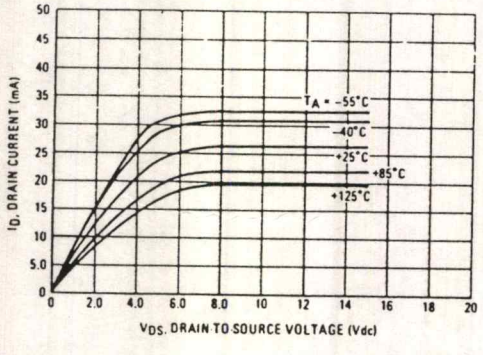


FIGURE 6 – V_{GS} = 15 Vdc



P-CHANNEL DRAIN CURRENT (SOURCE)

FIGURE 3 – V_{GS} = −5.0 Vdc

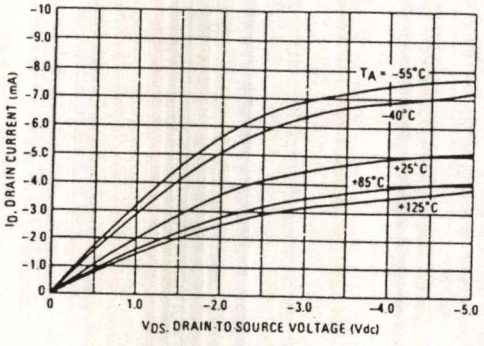


FIGURE 5 – V_{GS} = −10 Vdc

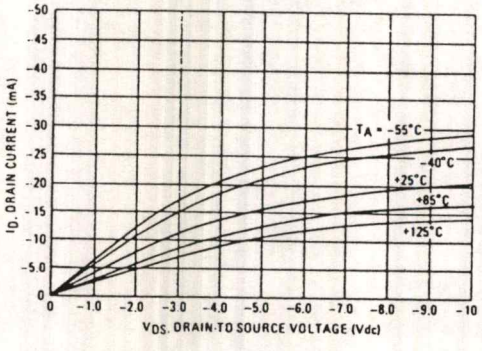
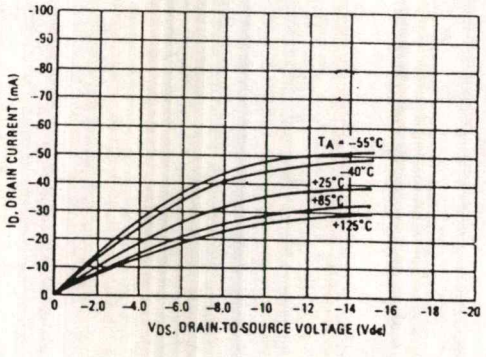


FIGURE 7 – V_{GS} = −15 Vdc

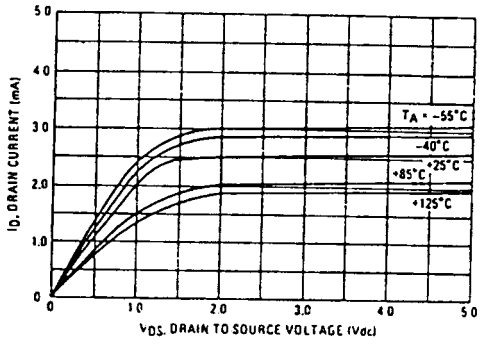


These typical curves are not guarantees, but are design aids.
Caution: The maximum rating for output current is 10 mA per pin.

TYPICAL B-SERIES GATE CHARACTERISTICS

N-CHANNEL DRAIN CURRENT
(SINK)

FIGURE 2 – V_{GS} = 5.0 Vdc



P-CHANNEL DRAIN CURRENT
(SOURCE)

FIGURE 3 – V_{GS} = −5.0 Vdc

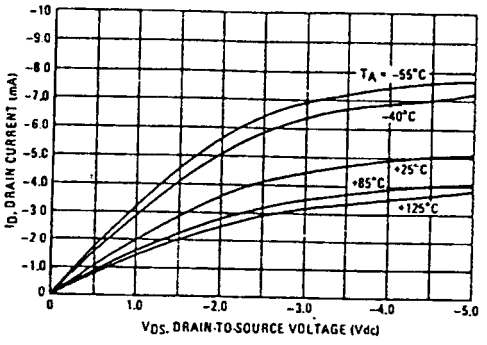


FIGURE 4 – V_{GS} = 10 Vdc

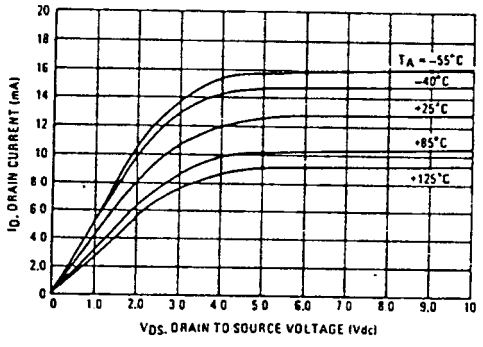


FIGURE 5 – V_{GS} = −10 Vdc

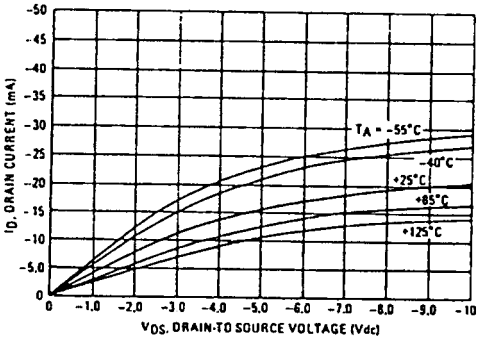


FIGURE 6 – V_{GS} = 15 Vdc

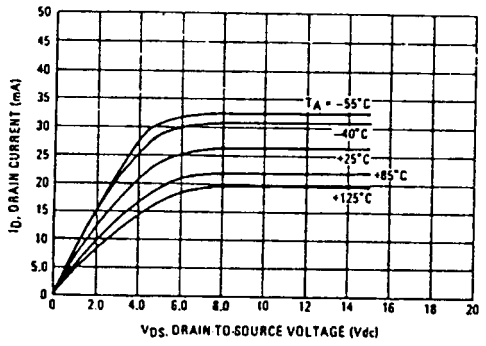
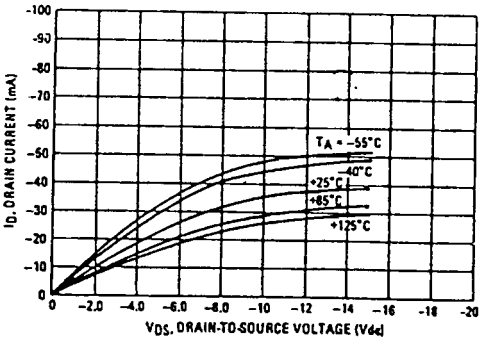


FIGURE 7 – V_{GS} = −15 Vdc



These typical curves are not guarantees, but are design aids.
Caution The maximum rating for output current is 10 mA per pin.

TYPICAL B-SERIES GATE CHARACTERISTICS (cont'd)

VOLTAGE TRANSFER CHARACTERISTICS

FIGURE 8 – V_{DD} = 5.0 Vdc

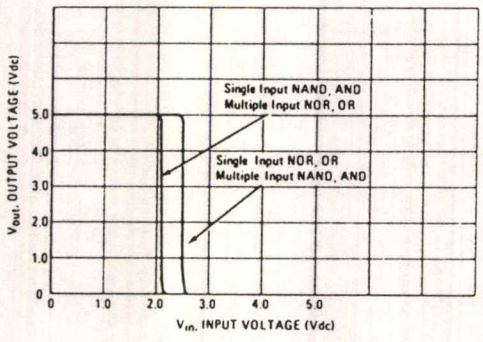


FIGURE 9 – V_{DD} = 10 Vdc

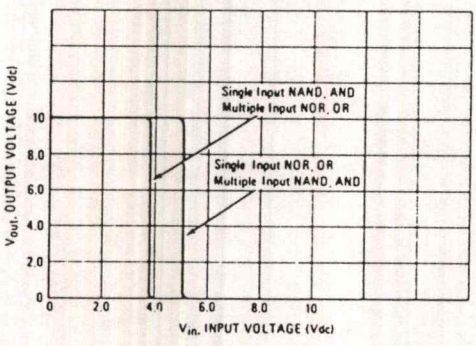
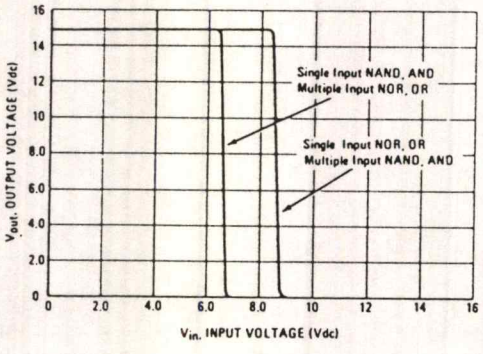


FIGURE 10 – V_{DD} = 15 Vdc



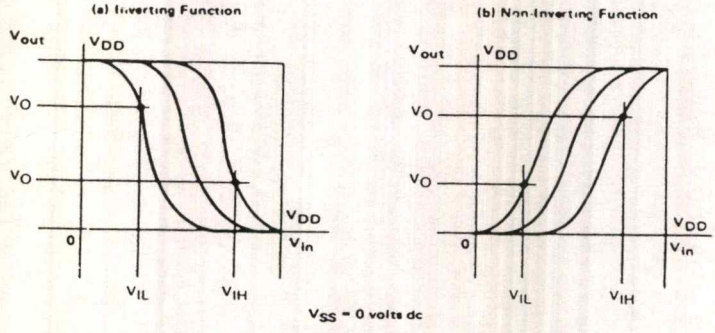
DC NOISE MARGIN

The DC noise margin is defined as the input voltage range from an ideal "1" or "0" input level which does not produce output state change(s). The typical and guaranteed limit values of the input values V_{IL} and V_{IH} for the output(s) to be at a fixed voltage V_O are given in the Electrical Characteristics table. V_{IL} and V_{IH} are presented graphically in Figure 11.

Guaranteed minimum noise margins for both the "1" and "0" levels =

1.0 V with a 5.0 V supply
2.0 V with a 10.0 V supply
2.5 V with a 15.0 V supply

FIGURE 11 – DC NOISE IMMUNITY



V_{SS} = 0 volts dc

TYPICAL B-SERIES GATE CHARACTERISTICS (cont'd)

VOLTAGE TRANSFER CHARACTERISTICS

FIGURE 8 – V_{DD} = 5.0 Vdc

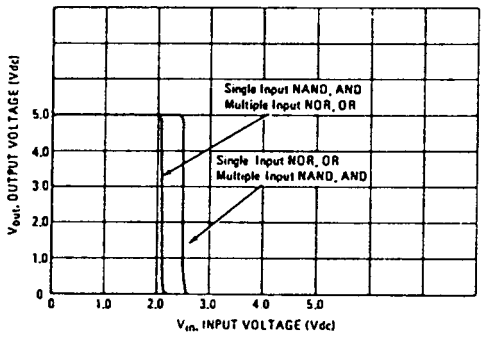


FIGURE 9 – V_{DD} = 10 Vdc

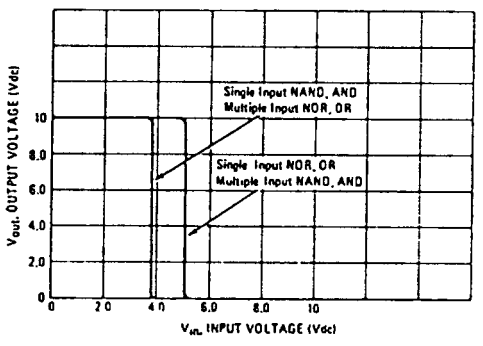
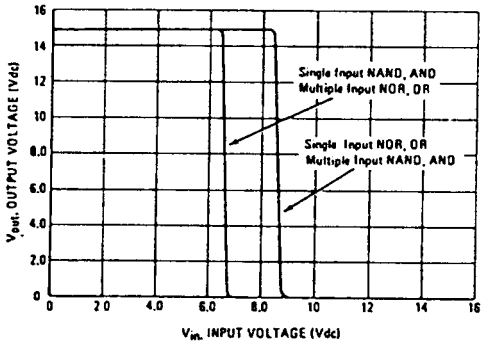


FIGURE 10 – V_{DD} = 15 Vdc



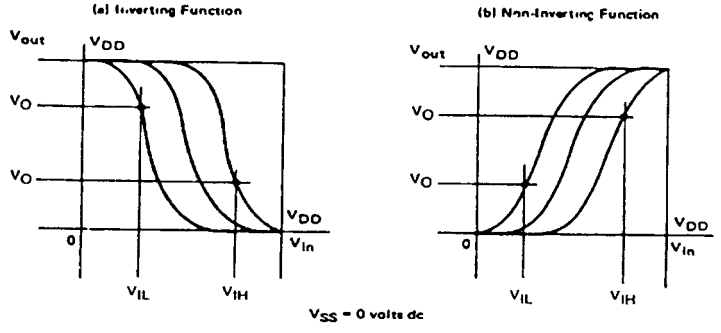
DC NOISE MARGIN

The DC noise margin is defined as the input voltage range from an ideal "1" or "0" input level which does not produce output state change(s) The typical and guaranteed limit values of the input values V_{IL} and V_{IH} for the output(s) to be at a fixed voltage V_O are given in the Electrical Characteristics table. V_{IL} and V_{IH} are presented graphically in Figure 11.

Guaranteed minimum noise margins for both the "1" and "0" levels =

1.0 V with a 5.0 V supply
2.0 V with a 10.0 V supply
2.5 V with a 15.0 V supply

FIGURE 11 – DC NOISE IMMUNITY



MOTOROLA

MC14001UB
Quad 2-Input NOR Gate

MC14002UB
Dual 4-Input NOR Gate

MC14011UB
Quad 2-Input NAND Gate

MC14012UB
Dual 4-Input NAND Gate

MC14023UB
Triple 3-Input NAND Gate

MC14025UB
Triple 3-Input NOR Gate

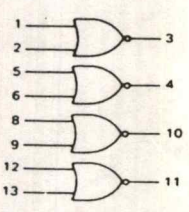
UB-SUFFIX SERIES CMOS GATES

The UB Series logic gates are constructed with P and N channel enhancement mode devices in a single monolithic structure (Complementary MOS). Their primary use is where low power dissipation and/or high noise immunity is desired. The UB set of CMOS gates are inverting non-buffered functions.

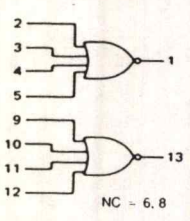
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Linear and Oscillator Applications
- Capable of Driving Two Low-power TTL Loads or One Low-power Schottky TTL Load Over the Rated Temperature Range.
- Double Diode Protection on All Inputs
- Pin-for-Pin Replacements for Corresponding CD4000 Series UB Suffix Devices

LOGIC DIAGRAMS

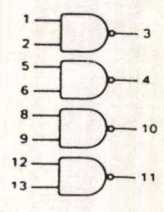
MC14001UB
Quad 2-Input NOR Gate



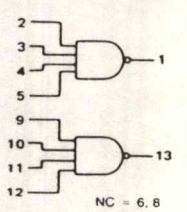
MC14002UB
Dual 4-Input NOR Gate



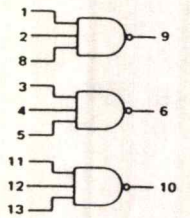
MC14011UB
Quad 2-Input NAND Gate



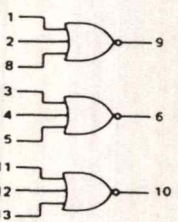
MC14012UB
Dual 4-Input NAND Gate



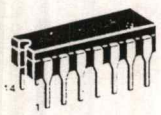
MC14023UB
Triple 3-Input NAND Gate



MC14025UB
Triple 3-Input NOR Gate



V_{DD} = Pin 14
V_{SS} = Pin 7
for All Devices



L SUFFIX
CERAMIC
CASE 632

P SUFFIX
PLASTIC
CASE 646

D SUFFIX
SOIC
CASE 751A

ORDERING INFORMATION

MC14XXXUBCP	Plastic
MC14XXXUBCL	Ceramic
MC14XXXUBD	SOIC

T_A = −55 to 125°C for all packages.

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}). Unused outputs must be left open.

6

MOTOROLA

MC14001UB
Quad 2-Input NOR Gate

MC14002UB
Dual 4-Input NOR Gate

MC14011UB
Quad 2-Input NAND Gate

MC14012UB
Dual 4-Input NAND Gate

MC14023UB
Triple 3-Input NAND Gate

MC14025UB
Triple 3-Input NOR Gate

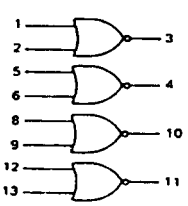
UB-SUFFIX SERIES CMOS GATES

The UB Series logic gates are constructed with P and N channel enhancement mode devices in a single monolithic structure (Complementary MOS). Their primary use is where low power dissipation and/or high noise immunity is desired. The UB set of CMOS gates are inverting non-buffered functions.

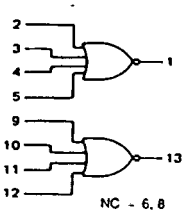
- Supply Voltage Range = 3.0 Vdc to 18 Vdc
- Linear and Oscillator Applications
- Capable of Driving Two Low-power TTL Loads or One Low-power Schottky TTL Load Over the Rated Temperature Range.
- Double Diode Protection on All Inputs
- Pin-for-Pin Replacements for Corresponding CD4000 Series UB Suffix Devices

LOGIC DIAGRAMS

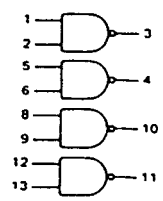
MC14001UB
Quad 2-Input NOR Gate



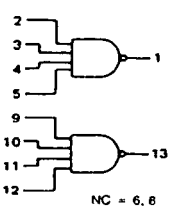
MC14002UB
Dual 4-Input NOR Gate



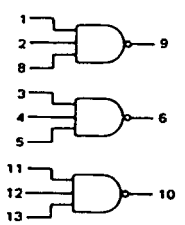
MC14011UB
Quad 2-Input NAND Gate



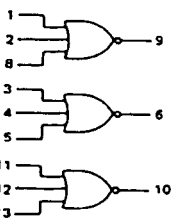
MC14012UB
Dual 4-Input NAND Gate



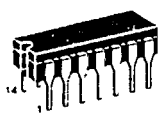
MC14023UB
Triple 3-Input NAND Gate



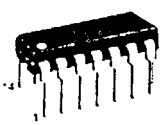
MC14025UB
Triple 3-Input NOR Gate



V_{DD} = Pin 14
V_{SS} = Pin 7
for All Devices



L SUFFIX
CERAMIC
CASE 632



P SUFFIX
PLASTIC
CASE 646

D SUFFIX
SOIC
CASE 751A

ORDERING INFORMATION

MC14XXXUBCP	Plastic
MC14XXXUBCL	Ceramic
MC14XXXUBD	SOIC

T_A = -55 to 125°C for all packages.

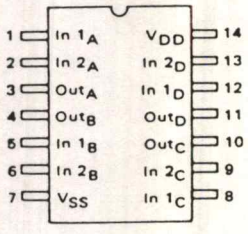
This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high-impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range $V_{SS} \leq (V_{in}$ or $V_{out}) \leq V_{DD}$.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}). Unused outputs must be left open.

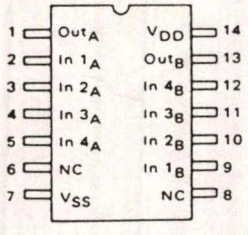
CMOS UB-SERIES GATES

PIN ASSIGNMENTS

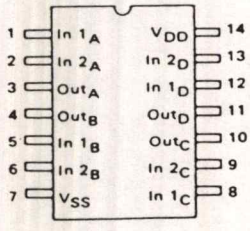
MC14001UB
Quad 2-Input NOR Gate



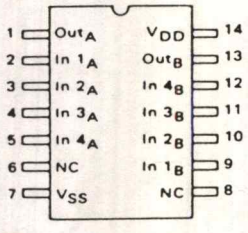
MC14002UB
Dual 4-Input NOR Gate



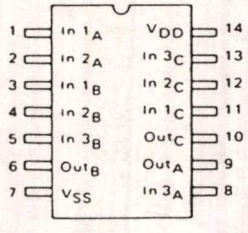
MC14011UB
Quad 2-Input NAND Gate



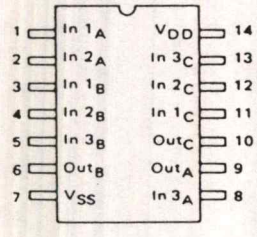
MC14012UB
Dual 4-Input NAND Gate



MC14023UB
Triple 3-Input NAND Gate



MC14025UB
Triple 3-Input NOR Gate



NC = No Connection

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

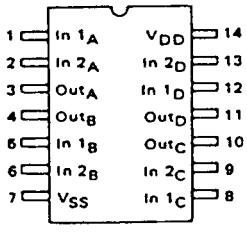
Symbol	Parameter	Value	Unit
V_{DD}	DC Supply Voltage	-0.5 to +18.0	V
V_{in}, V_{out}	Input or Output Voltage (DC or Transient)	-0.5 to V_{DD} +0.5	V
I_{in}, I_{out}	Input or Output Current (DC or Transient), per Pin	±10	mA
P_D	Power Dissipation, per Package†	500	mW
T_{stg}	Storage Temperature	-65 to +150	°C
T_L	Lead Temperature (8-Second Soldering)	260	°C

*Maximum Ratings are those values beyond which damage to the device may occur.
†Temperature Derating: Plastic "P and D/DW" Packages: 7.0 mW/°C From 65°C To 125°C
Ceramic "L" Packages: 12 mW/°C From 100°C To 125°C

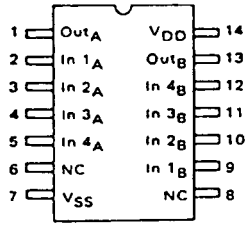
CMOS UB-SERIES GATES

PIN ASSIGNMENTS

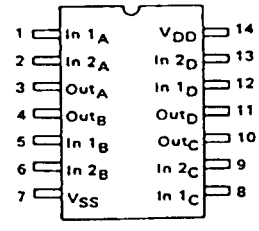
MC14001UB
Quad 2-Input NOR Gate



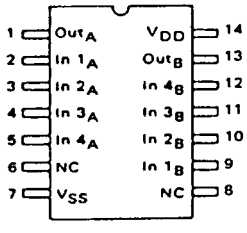
MC14002UB
Dual 4-Input NOR Gate



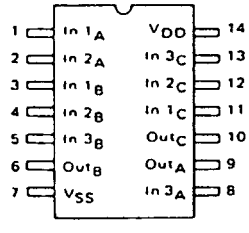
MC14011UB
Quad 2-Input NAND Gate



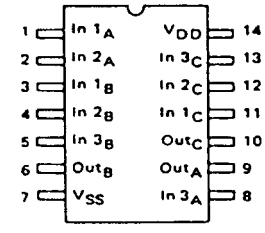
MC14012UB
Dual 4-Input NAND Gate



MC14023UB
Triple 3-Input NAND Gate



MC14025UB
Triple 3-Input NOR Gate



NC = No Connection

MAXIMUM RATINGS* (Voltages Referenced to V_{SS})

Symbol	Parameter	Value	Unit
V_{DD}	DC Supply Voltage	-0.5 to +18.0	V
V_{in}, V_{out}	Input or Output Voltage (DC or Transient)	-0.5 to V_{DD} +0.5	V
I_{in}, I_{out}	Input or Output Current (DC or Transient), per Pin	±10	mA
P_D	Power Dissipation, per Package†	500	mW
T_{stg}	Storage Temperature	-65 to +150	°C
T_L	Lead Temperature (8-Second Soldering)	260	°C

*Maximum Ratings are those values beyond which damage to the device may occur.
†Temperature Derating: Plastic "P and D/DW" Packages 7.0 mW/°C From 65°C To 125°C
Ceramic "L" Packages 12 mW/°C From 100°C To 125°C

ELECTRICAL CHARACTERISTICS (Voltages Referenced to V_{SS})

Characteristic	Symbol	V_{DD} Vdc	−55°C Min	−55°C Max	25°C Min	25°C Typ #	25°C Max	125°C Min	125°C Max	Unit
Output Voltage "0" Level V_{in} = V_{DD} or 0	V_{OL}	5.0	—	0.05	—	0	0.05	—	0.05	Vdc
		10	—	0.05	—	0	0.05	—	0.05	
		15	—	0.05	—	0	0.05	—	0.05	
"1" Level V_{in} = 0 or V_{DD}	V_{OH}	5.0	4.95	—	4.95	5.0	—	4.95	—	Vdc
		10	9.95	—	9.95	10	—	9.95	—	
		15	14.95	—	14.95	15	—	14.95	—	
Input Voltage "0" Level (V_O = 4.5 Vdc)	V_{IL}	5.0	—	1.0	—	2.25	1.0	—	1.0	Vdc
(V_O = 9.0 Vdc)		10	—	2.0	—	4.50	2.0	—	2.0	
(V_O = 13.5 Vdc)		15	—	2.5	—	6.75	2.5	—	2.5	
"1" Level (V_O = 0.5 Vdc)	V_{IH}	5.0	4.0	—	4.0	2.75	—	4.0	—	Vdc
(V_O = 1.0 Vdc)		10	8.0	—	8.0	5.50	—	8.0	—	
(V_O = 1.5 Vdc)		15	12.5	—	12.5	8.25	—	12.5	—	
Output Drive Current (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	−1.2	—	−1.0	−1.7	—	−0.7	—	mAdc
(V_{OH} = 4.6 Vdc)		5.0	−0.25	—	−0.2	−0.36	—	−0.14	—	
(V_{OH} = 9.5 Vdc)		10	−0.62	—	−0.5	−0.9	—	−0.35	—	
(V_{OH} = 13.5 Vdc)		15	−1.8	—	−1.5	−3.5	—	−1.1	—	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	—	0.51	0.88	—	0.36	—	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	—	1.3	2.25	—	0.9	—	
(V_{OL} = 1.5 Vdc)		15	4.2	—	3.4	8.8	—	2.4	—	
Input Current	I_{in}	15	—	±0.1	—	±0.00001	±0.1	—	±1.0	μAdc
Input Capacitance (V_{in} = 0)	C_{in}	—	—	—	—	5.0	7.5	—	—	pF
Quiescent Current (Per Package)	I_{DD}	5.0	—	0.25	—	0.0005	0.25	—	7.5	μAdc
		10	—	0.5	—	0.0010	0.5	—	15	
		15	—	1.0	—	0.0015	1.0	—	30	
Total Supply Current**† (Dynamic plus Quiescent, Per Gate, C_L = 50 pF)	I_T	5.0	I_T = (0.3 μA/kHz) f + I_{DD}/N							μAdc
		10	I_T = (0.6 μA/kHz) f + I_{DD}/N							
		15	I_T = (0.8 μA/kHz) f + I_{DD}/N							

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

**The formulas given are for the typical characteristics only at 25°C.

†To calculate total supply current at loads other than 50 pF:

$$I_T(C_L) = I_T(50\ pF) + (C_L - 50)\ Vfk$$

where: I_T is in μH (per package), C_L in pF, V = (V_{DD} − V_{SS}) in volts, f in kHz is input frequency, and k = 0.001 × the number of exercised gates per package.

ELECTRICAL CHARACTERISTICS (Voltages Referenced to V_{SS})

Characteristic	Symbol	V_{DD} Vdc	−55°C Min	−55°C Max	25°C Min	25°C Typ #	25°C Max	125°C Min	125°C Max	Unit
Output Voltage "0" Level V_{in} = V_{DD} or 0	V_{OL}	5.0	—	0.05	—	0	0.05	—	0.05	Vdc
		10	—	0.05	—	0	0.05	—	0.05	
		15	—	0.05	—	0	0.05	—	0.05	
"1" Level V_{in} = 0 or V_{DD}	V_{OH}	5.0	4.95	—	4.95	5.0	—	4.95	—	Vdc
		10	9.95	—	9.95	10	—	9.95	—	
		15	14.95	—	14.95	15	—	14.95	—	
Input Voltage "0" Level (V_O = 4.5 Vdc)	V_{IL}	5.0	—	1.0	—	2.25	1.0	—	1.0	Vdc
(V_O = 9.0 Vdc)		10	—	2.0	—	4.50	2.0	—	2.0	
(V_O = 13.5 Vdc)		15	—	2.5	—	6.75	2.5	—	2.5	
"1" Level (V_O = 0.5 Vdc)	V_{IH}	5.0	4.0	—	4.0	2.75	—	4.0	—	Vdc
(V_O = 1.0 Vdc)		10	8.0	—	8.0	5.50	—	8.0	—	
(V_O = 1.5 Vdc)		15	12.5	—	12.5	8.25	—	12.5	—	
Output Drive Current (V_{OH} = 2.5 Vdc) Source	I_{OH}	5.0	−1.2	—	−1.0	−1.7	—	−0.7	—	mAdc
(V_{OH} = 4.6 Vdc)		5.0	−0.25	—	−0.2	−0.36	—	−0.14	—	
(V_{OH} = 9.5 Vdc)		10	−0.62	—	−0.5	−0.9	—	−0.35	—	
(V_{OH} = 13.5 Vdc)		15	−1.8	—	−1.5	−3.5	—	−1.1	—	
(V_{OL} = 0.4 Vdc) Sink	I_{OL}	5.0	0.64	—	0.51	0.88	—	0.36	—	mAdc
(V_{OL} = 0.5 Vdc)		10	1.6	—	1.3	2.25	—	0.9	—	
(V_{OL} = 1.5 Vdc)		15	4.2	—	3.4	8.8	—	2.4	—	
Input Current	I_{in}	15	—	±0.1	—	±0.00001	±0.1	—	±1.0	μAdc
Input Capacitance (V_{in} = 0)	C_{in}	—	—	—	—	5.0	7.5	—	—	pF
Quiescent Current (Per Package)	I_{DD}	5.0	—	0.25	—	0.0005	0.25	—	7.5	μAdc
		10	—	0.5	—	0.0010	0.5	—	15	
		15	—	1.0	—	0.0015	1.0	—	30	
Total Supply Current**† (Dynamic plus Quiescent, Per Gate, C_L = 50 pF)	I_T	5.0	I_T = (0.3 μA/kHz) f + I_{DD}/N							μAdc
		10	I_T = (0.6 μA/kHz) f + I_{DD}/N							
		15	I_T = (0.8 μA/kHz) f + I_{DD}/N							

#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

**The formulas given are for the typical characteristics only at 25°C

†To calculate total supply current at loads other than 50 pF:

$$I_T(C_L) = I_T(50\ pF) - (C_L - 50)\ Vfk$$

where: I_T is in μH (per package), C_L in pF, V = (V_{DD} − V_{SS}) in volts, f in kHz is input frequency, and k = 0.001 × the number of exercised gates per package.

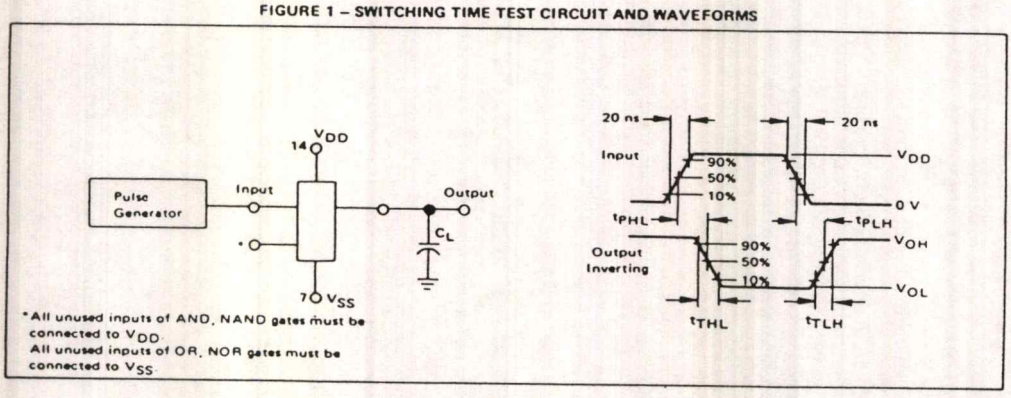
SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD} Vdc	Min	Typ #	Max	Unit
Output Rise Time	t_{TLH}					ns
t_{TLH} = (3.0 ns/pF) C_L + 30 ns		5.0	–	180	360	
t_{TLH} = (1.5 ns/pF) C_L + 15 ns		10	–	90	180	
t_{TLH} = (1.1 ns/pF) C_L + 10 ns		15	–	65	130	
Output Fall Time	t_{THL}					ns
t_{THL} = (1.5 ns/pF) C_L + 25 ns		5.0	–	100	200	
t_{THL} = (0.75 ns/pF) C_L + 12.5 ns		10	–	50	100	
t_{THL} = (0.55 ns/pF) C_L + 9.5 ns		15	–	40	80	
Propagation Delay Time	t_{PLH}, t_{PHL}					ns
t_{PLH}, t_{PHL} = (1.7 ns/pF) C_L + 30 ns		5.0	–	90	180	
t_{PLH}, t_{PHL} = (0.66 ns/pF) C_L + 22 ns		10	–	50	100	
t_{PLH}, t_{PHL} = (0.50 ns/pF) C_L + 15 ns		15	–	40	80	

*The formulas given are for the typical characteristics only at 25°C.

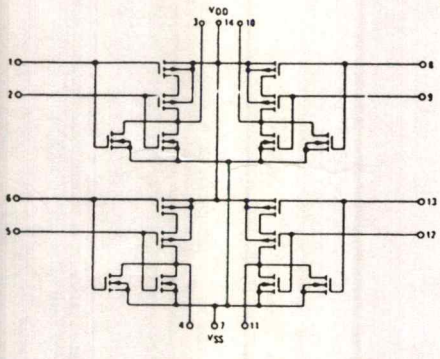
#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

FIGURE 1 – SWITCHING TIME TEST CIRCUIT AND WAVEFORMS

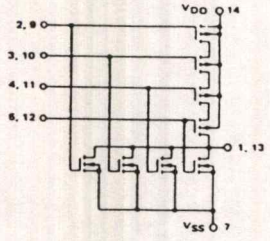


*All unused inputs of AND, NAND gates must be connected to V_{DD}.
All unused inputs of OR, NOR gates must be connected to V_{SS}.

MC14001UB CIRCUIT SCHEMATIC



MC14002UB CIRCUIT SCHEMATIC
(1/2 of Device Shown)



6

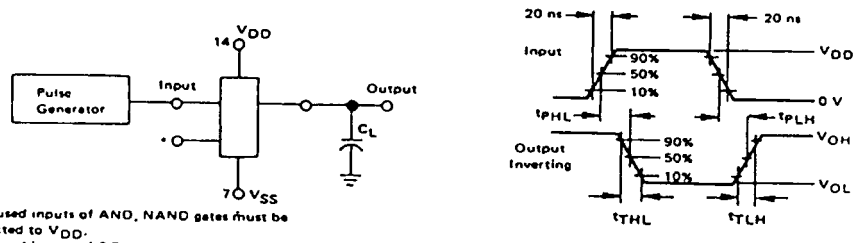
SWITCHING CHARACTERISTICS* (C_L = 50 pF, T_A = 25°C)

Characteristic	Symbol	V_{DD} Vdc	Min	Typ #	Max	Unit
Output Rise Time	t_{TLH}					ns
t_{TLH} = (3.0 ns/pF) C_L + 30 ns		5.0	–	180	360	
t_{TLH} = (1.5 ns/pF) C_L + 15 ns		10	–	90	180	
t_{TLH} = (1.1 ns/pF) C_L + 10 ns		15	–	65	130	
Output Fall Time	t_{THL}					ns
t_{THL} = (1.5 ns/pF) C_L + 25 ns		5.0	–	100	200	
t_{THL} = (0.75 ns/pF) C_L + 12.5 ns		10	–	50	100	
t_{THL} = (0.55 ns/pF) C_L + 9.5 ns		15	–	40	80	
Propagation Delay Time	t_{PLH}, t_{PHL}					ns
t_{PLH}, t_{PHL} = (1.7 ns/pF) C_L + 30 ns		5.0	–	90	180	
t_{PLH}, t_{PHL} = (0.66 ns/pF) C_L + 22 ns		10	–	50	100	
t_{PLH}, t_{PHL} = (0.50 ns/pF) C_L + 15 ns		15	–	40	80	

*The formulas given are for the typical characteristics only at 25°C.

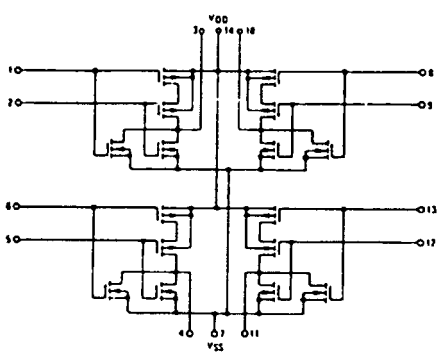
#Data labelled "Typ" is not to be used for design purposes but is intended as an indication of the IC's potential performance.

FIGURE 1 – SWITCHING TIME TEST CIRCUIT AND WAVEFORMS

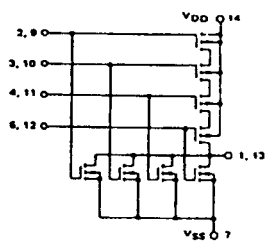


*All unused inputs of AND, NAND gates must be connected to V_{DD}.
All unused inputs of OR, NOR gates must be connected to V_{SS}

MC14001UB CIRCUIT SCHEMATIC



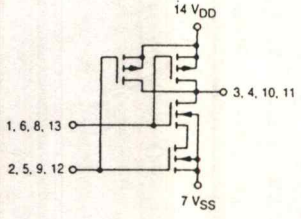
MC14002UB CIRCUIT SCHEMATIC
(1/2 of Device Shown)



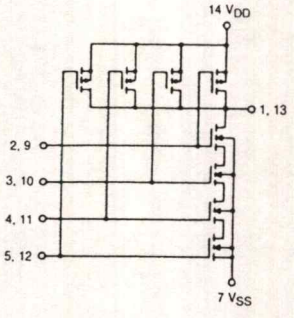
6

CMOS UB-SERIES GATES

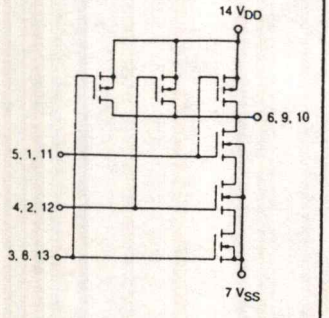
MC14011UB CIRCUIT SCHEMATIC
(1/4 of Device Shown)



MC14012UB CIRCUIT SCHEMATIC
(1/2 of Device Shown)



MC14023UB CIRCUIT SCHEMATIC
(1/3 of Device Shown)



MC14025UB CIRCUIT SCHEMATIC
(1/3 of Device Shown)

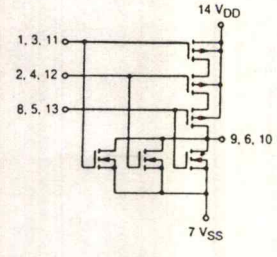


FIGURE 2 – TYPICAL VOLTAGE AND CURRENT TRANSFER CHARACTERISTICS

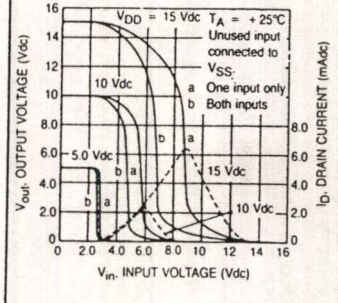


FIGURE 3 – TYPICAL VOLTAGE TRANSFER CHARACTERISTICS versus TEMPERATURE

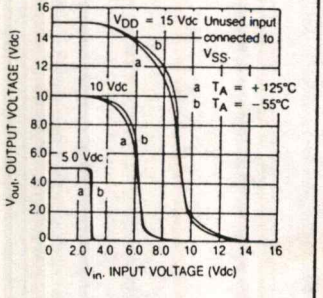


FIGURE 4 – TYPICAL OUTPUT SOURCE CHARACTERISTICS

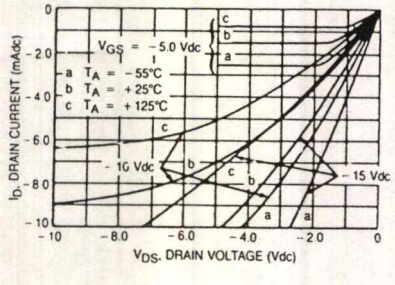
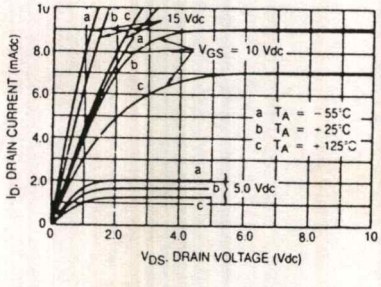
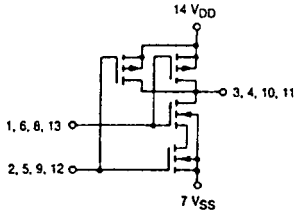


FIGURE 5 – TYPICAL OUTPUT SINK CHARACTERISTICS

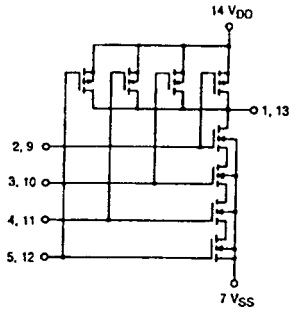


6

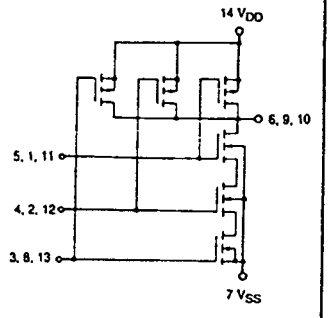
MC14011UB CIRCUIT SCHEMATIC
(1/4 of Device Shown)



MC14012UB CIRCUIT SCHEMATIC
(1/2 of Device Shown)



MC14023UB CIRCUIT SCHEMATIC
(1/3 of Device Shown)



MC14025UB CIRCUIT SCHEMATIC
(1/3 of Device Shown)

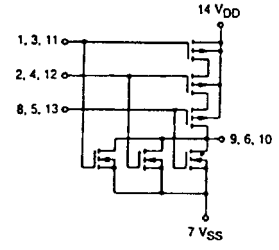


FIGURE 2 – TYPICAL VOLTAGE AND CURRENT TRANSFER CHARACTERISTICS

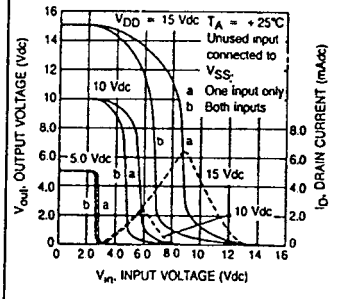


FIGURE 3 – TYPICAL VOLTAGE TRANSFER CHARACTERISTICS versus TEMPERATURE

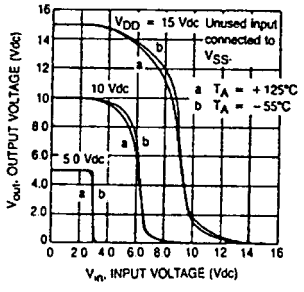


FIGURE 4 – TYPICAL OUTPUT SOURCE CHARACTERISTICS

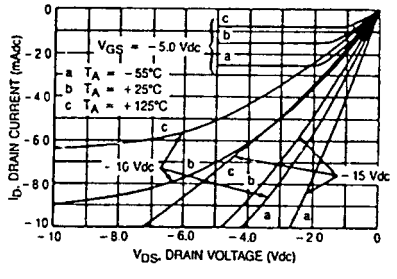
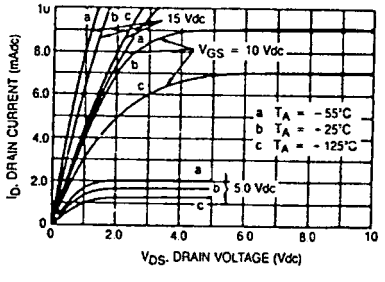


FIGURE 5 – TYPICAL OUTPUT SINK CHARACTERISTICS



6