ชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ 2 คู่สายนอก 8 คู่สายใน

PRIVATE AUTOMATIC BRANCH EXCHANGE 2 TO 8

โดย

นายเดโช อินทรรุจิกุล

นายทวี กุสุมาวลี

นายสุธน แสงปรีดีกรณ์

วัน เดือน ปี 18 ต.ค. 2541

เลขทะเบียน 039076

เลขเรียกหนังสือ โ ๓๐๙๒ ค ๘๑๗

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

039076

ชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ 2 คู่สายนอก 8 คู่สายใน

PRIVATE AUTOMATIC BRANCH EXCHANGE 2 TO 8

โดย

นายเดโช อินทรรุจิกุล 37014128

นายทวี กุสุมาวลี 37014137

นายสุธน แสงปรีดีกรณ์ 37014511

อาจารย์ที่ปรึกษา

ผศ.ดร.ไกรสิน ส่งวัฒนา

ปริญญานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมโทรคมนาคม

สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

ปีการศึกษา 2540

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ปริญญานิพนธ์ ปีการศึกษา 2541

ภาควิชาวิศวกรรมโทรคมนาคม

คณะวิศวกรรมศาสตร์ สถาบันเทคโนโลยีพระจอมเกล้าเจ้าคุณทหารลาดกระบัง

เรื่อง **ชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ 2 คู่สายนอก 8 คู่สายใน**

PRIVATE AUTOMATIC BRANCH EXCHANGE 2 TO 8

ผู้จัดทำ 1. นาย เดโช อินทรรุจิกุล 37014128

2. นาย ทวี กุสุมาวลี 37014137

3. นาย สุธน แสงปรีดีกรณ์ 37014511

..... อาจารย์ที่ปรึกษา

(ผศ. ดร. ไกรสิน ส่งวัฒนา)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ 2 คู่สายนอก 8 คู่สายใน
PRIVATE AUTOMATIC BRANCH EXCHANGE 2 TO 8

โดย นายเดโช อินทรรุจิกุล 37014128

นาย ทวี กุสุมาวลี 37014137

นายสุธน แสงปรีดีกรณ์ 37014511

อาจารย์ที่ปรึกษา ผศ. ด.ร. ไกรสิน ส่งวัฒนา

บทคัดย่อ

โครงงานนี้นำเสนอเกี่ยวกับชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ ขนาด 2 คู่สายนอก ขยายเป็น 8 คู่สายใน โดยนำไอซีตระกูล MCS - 51 เป็นส่วนควบคุมและประมวลผลหลัก โดยการทำงานของชุมสายสามารถโอนสายพักสาย และสนทนา 3 สายได้ รวมทั้งยังพัฒนาให้สามารถบันทึกข้อมูลต่าง ๆ เพื่อใช้งานสามารถฝากข้อความอัตโนมัติซึ่งเป็นหมายเลขที่ใช้ในการติดต่อกลับ และส่วนแสดงผลสามารถแสดงผลทางจอแสดงผล LCD ได้

ARSTRACT

This project presents automatic private branch exchange 2 to 8's operation, where MCS - 51 microcontroller is used as control and central processing unit. As a result, automatic PBX can be used for line fransferring, line holding, and three-people conversation. This project is also developed for automatic PBX to be able to record telephone number for call back, and the work result of display unit can be shown on LCD display.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญ

		หน้า
บทที่ 1	**บทนำ**	1
บทที่ 2	**ทฤษฎีและหลักการ**	2
	2.1 ความรู้เบื้องต้นเกี่ยวกับโทรศัพท์	2
	2.2 วงจรถอดรหัสความถี่โทรศัพท์	3
	2.3 ISD 12xx/14xx SERIES (ไอซีบันทึกเสียง)	6
	2.4 ชุมสายโทรศัพท์	10
	2.5 ส่วนแสดงผลชนิด LCD MODULE	12
	2.6 ตัวสร้างสัญญาณนาฬิกาหรือ RTC	24
	2.4 ไอซี MM74C922ตัวเข้ารหัสการกดคีย์	25
บทที่ 3	**การคำนวณและการสร้าง**	27
	3.1 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก	30
	3.2 ส่วนครอสพอยท์สวิทช์	32
	3.3 ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF	34
	3.4 ส่วนกำเนิดสัญญาณเสียง	38
	3.5 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน	40
	3.6 ส่วนตอบรับโทรศัพท์อัตโนมัติ	41
	3.7 ส่วนจอแสดงผล,ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล	42
	3.8 ส่วนอินพุท/เอาท์พุทพอร์ท	43
	3.9 ส่วนแหล่งจ่ายไฟเลี้ยง	46
	3.10 ส่วนควบคุมระบบ	49
บทที่ 4	**การทดลองและผลการทดลอง**	57
	4.1 การทดลองส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก	57
	4.2 การทดลองส่วนครอสพอยท์สวิทช์	60
	4.3 การทดลองส่วนตรวจจับสัญญาณเสียง	60
	4.4 การทดลองส่วนกำเนิดสัญญาณเสียง	65
	4.5 การทดลองส่วนถอดรหัสสัญญาณ DTMF	67
	4.6 การทดลองส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน	68
	4.7 การทดลองส่วนตอบรับอัตโนมัติ	70
	4.8 การทดลองส่วนเข้ารหัสการกดคีย์บอร์ด	71
	4.9 การทดลองส่วนแหล่งจ่ายไฟเลี้ยง	71
	4.10 ขั้นตอนการทดลองการโทรเข้า	73

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

4.11 ขั้นตอนการทดลองการโทรออก 73
4.12 ขั้นตอนการโอนสาย 73
4.13 ขั้นตอนการพักสาย 73
4.14 ขั้นตอนการประชุม 3 สาย 73
4.15 ขั้นตอนการทดลองการฝากเลขหมาย 74
4.16 ขั้นตอนการตรวจสอบหมายเลขติดต่อกลับ 74
บทที่ 5 บทสรุปและวิจารณ์ 77
5.1 คุณสมบัติของโครงงาน 77
5.2 ปัญหาและแนวทางแก้ไข 78
5.3 แนวทางการพัฒนาโครงงาน 78
ภาคผนวก 79
ผังงานแสดงขั้นตอนการทำงานของโปรแกรมหลัก
ผังงานแสดงการโทรเข้ามาของสายนอก
ผังงานแสดงขั้นตอนการโทรออกของสายใน
ผังงานแสดงขั้นตอนการใช้งานฟังก์ชั่นพิเศษของ PABX
ผังงานแสดงขั้นตอนการทำงานของโปรแกรมควบคุมส่วนแสดงผล
แผนผังแสดงขั้นตอนการสื่อสารระหว่างไมโครคอนโทรลเลอร์ตัวที่ 1 และ ตัวที่ 2
โปรแกรมควบคุมระบบหลัก
โปรแกรมควบคุมส่วนแสดงผล
โปรแกรมในการตรวจจับสัญญาณไม่ว่าง (busy tone) และสัญญาณเรียกกลับ (ring back tone)
โปรแกรมในการตรวจจับสัญญาณกระดิ่ง
ลายวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก
ลายวงจรส่วนครอสพอยท์สวิทช์
ลายวงจรส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF
ลายวงจรส่วนกำเนิดสัญญาณเสียง
ลายวงจรส่วนตอบรับโทรศัพท์อัตโนมัติ
ลายวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน
ลายวงจรส่วนแหล่งจ่ายไฟเลี้ยง
กิตติกรรมประกาศ
บรรณานุกรม

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญรูปภาพ

	หน้า
รูปที่ 2.1 แสดงความถี่ของโทรศัพท์ชนิดกดปุ่ม	3
รูปที่ 2.2 แสดงรายละเอียดของ MT8870	4
รูปที่ 2.3 Block diagram ของ ISD 1200/1400 series	8
รูปที่ 2.4 แสดงรายละเอียดขาของ ISD 1200/1400 series	9
รูปที่ 2.5 โครงสร้างทั่วไปของ LCD MODULE	12
รูปที่ 2.6 แสดงการอินเทอร์เฟสของ MCS-51 กับ LCM	13
รูปที่ 2.7 แสดงแผนผังเวลาในการติดต่อกับ LCM	17
รูปที่ 2.8 แสดง 16 KEY ENCODERS	26
รูปที่ 3.1 บล็อคไดอะแกรมของเครื่องชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ	29
รูปที่ 3.2 แสดงวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก	31
รูปที่ 3.3 แสดงการต่อกันของไอซี MC142100	32
รูปที่ 3.4 แสดงวงจรส่วนครอสพอยท์สวิทช์	33
รูปที่ 3.5 แสดงวงจรถอดรหัสสัญญาณ DTMF	36
รูปที่ 3.6 แสดงวงจรตรวจจับสัญญาณเสียง	37
รูปที่ 3.7 แสดงวงจรกำเนิดสัญญาณเสียง	39
รูปที่ 3.8 แสดงวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน	40
รูปที่ 3.9 แสดงวงจรตอบรับโทรศัพท์อัตโนมัติ	41
รูปที่ 3.10 วงจรเข้ารหัสการกดคีย์	42
รูปที่ 3.11 วงจรส่วนอินพุท/เอาท์พุท พอร์ท	47
รูปที่ 3.12 แสดงวงจรแหล่งจ่ายไฟเลี้ยง	48
รูปที่ 3.13 วงจรส่วนควบคุมระบบ	50
รูปที่ 3.14 แสดงส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก	51
รูปที่ 3.15 แสดงส่วนครอสพอยท์สวิทช์	51
รูปที่ 3.16 ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF	52
รูปที่ 3.17 ส่วนกำเนิดสัญญาณเสียง	52
รูปที่ 3.18 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน	53
รูปที่ 3.19 ส่วนตอบรับโทรศัพท์อัตโนมัติ	53

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 3.20 ส่วนจอแสดงผล , ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล 54

รูปที่ 3.21 ส่วนอินพุท / เอาท์พุทพอร์ท 54

รูปที่ 3.22 ส่วนแหล่งจ่ายไฟเลี้ยง 55

รูปที่ 3.23 ส่วนควบคุมระบบ 55

รูปที่ 3.24 แสดงโครงงานประกอบสำเร็จภายใน 56

รูปที่ 3.25 แสดงโครงงานประกอบสำเร็จภายนอก 56

รูปที่ 4.1 สัญญาณไฟตรงคร่อมคู่สายภายนอกขณะวางหู 57

รูปที่ 4.2 สัญญาณไฟตรงคร่อมคู่สายภายนอกขณะยกหู 58

รูปที่ 4.3 สัญญาณแสดงสถานะการยกหู - วางหู ของคู่สายภายนอก 58

รูปที่ 4.4 สัญญาณคลื่นสี่เหลี่ยมที่ส่งเข้าส่วนอินเตอร์เฟสคู่สายภายนอกกับสัญญาณเอาท์พุทที่ออกจากส่วนอินเตอร์เฟสคู่สายภายนอก 59

รูปที่ 4.5 สัญญาณ DTMF ที่มาจากคู่สายภายนอกกับสัญญาณ DTMF ที่ออกจากส่วนอินเตอร์เฟสคู่สายภายนอกไปส่วนถอดรหัสสัญญาณ DTMF 59

รูปที่ 4.6 สัญญาณคลื่นสี่เหลี่ยมที่ส่งเข้าและออกจากส่วนครอสพอยท์สวิทช์เมื่อมีการกดสวิทช์เกิดขึ้น 60

รูปที่ 4.7 สัญญาณโทนหรือสัญญาณเรียกกลับและสัญญาณไม่ว่าง 61

รูปที่ 4.8 สัญญาณเรียกกลับและสัญญาณแสดงการตรวจพบสัญญาณโทนความถี่ 400 Hz 62

รูปที่ 4.9 สัญญาณไม่ว่างและสัญญาณแสดงการตรวจพบสัญญาณโทนความถี่ 400 Hz 62

รูปที่ 4.10 สัญญาณอินเตอร์รัพท์ และสัญญาณตรวจพบสัญญาณกระดิ่งของสายนอกที่ 1 63

รูปที่ 4.11 สัญญาณอินเตอร์รัพท์และสัญญาณสั่งให้ยกหูของสายนอกที่ 1 63

รูปที่ 4.12 สัญญาณอินเตอร์รัพท์และสัญญาณตรวจพบสัญญาณกระดิ่งของสายนอกสายที่ 2 64

รูปที่ 4.13 สัญญาณอินเตอร์รัพท์และสัญญาณสั่งให้ยกหูของสายนอกสายที่ 2 64

รูปที่ 4.14 สัญญาณพร้อมให้กดเลขหมาย 65

รูปที่ 4.15 สัญญาณเรียกกลับ (Ring back tone) 66

รูปที่ 4.16 สัญญาณไม่ว่าง (Busy tone) 66

รูปที่ 4.17 สัญญาณ DTMF หมายเลข "1" และสัญญาณแสดงการตรวจพบ DTMF ที่ขา STD 67

รูปที่ 4.18 สัญญาณแสดงการตรวจพบ DTMF และสัญญาณอินเตอร์รัพท์ 68

รูปที่ 4.19 สัญญาณไฟตรงคร่อมคู่สายภายในขณะวางหู 69

รูปที่ 4.20 สัญญาณไฟตรงคร่อมคู่สายภายในขณะยกหูและสัญญาณแสดงสถานะการยกหู 69

รูปที่ 4.21 สัญญาณควบคุมการส่งสัญญาณกระดิ่งและสัญญาณกระดิ่ง 70

รูปที่ 4.22 สัญญาณคลื่นสี่เหลี่ยมที่บันทึกและเล่นกลับออกมาจากไอซี ISD 1420 70

รูปที่ 4.23 สัญญาณจากขา DA ของไอซี 74C922 และสัญญาณเสียงของการกดคีย์บอร์ด 71

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน้า

รูปที่ 4.24 สัญญาณไฟตรง 5 V และ 24 V จากแหล่งจ่ายไฟเลี้ยง 72
รูปที่ 4.25 สัญญาณคลื่นสี่เหลี่ยมที่สร้างจากไอซี 555 และสัญญาณกระดิ่ง 72
รูปที่ 4.26 หน้าจอ LCD ก่อนเข้าสู่เมนูหลัก 74
รูปที่ 4.27 หน้าจอ LCD แสดงเมนูหลัก 75
รูปที่ 4.28 หน้าจอ LCD ขณะที่มีการฝากเลขหมายติดต่อกลับ 75
รูปที่ 4.29 หน้าจอ LCD แสดงเลขหมายติดต่อกลับ และเวลาที่ฝากเลขหมาย 76

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สารบัญตาราง

	หน้า
ตารางที่ 2.1 แสดงค่าที่ถอดรหัสได้จากความถี่ต่าง ๆ	4
ตารางที่ 2.2 แสดงคุณสมบัติทางไฟฟ้าของตระกูล ISD 12xx/14xx SERIES	7
ตารางที่ 2.3 แสดงขาสัญญาณของ จอแสดงผล LCD	15
ตารางที่ 2.4 แสดงการทำงานของสัญญาณ Enable	16
ตารางที่ 2.5 แสดงรายละเอียดชุดคำสั่ง HD44780	18
ตารางที่ 3.1 แสดงค่า BCD ที่ได้จากการกดหมายเลขโทรศัพท์	35

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 1
บทนำ

การติดต่อสื่อสารในปัจจุบันนี้ ได้เข้ามามีบทบาทสำคัญต่อชีวิตประจำวันมากขึ้นมีความต้องการในการใช้บริการทางการติดต่อสื่อสารเพิ่มมากยิ่งขึ้นในอนาคต เพราะฉะนั้นการพัฒนาทางด้านเทคโนโลยีทางด้านการติดต่อสื่อสารจึงเป็นสิ่งสำคัญอย่างยิ่งในการที่จะรองรับการเพิ่มปริมาณการสื่อสารที่จะเกิดขึ้น

แต่เดิมในการติดต่อสื่อสารในระบบโทรศัพท์ระหว่างภายในกับภายนอกนั้น จะแยกระบบอุปกรณ์การติดต่อระหว่างภายในกับภายนอกออกจากกัน เช่น มีโทรศัพท์ติดต่อกับชุมสายภายนอกได้ 2 -4 คู่สาย และมีการใช้อินเตอร์คอม (INTERCOM) สำหรับการติดต่อกับภายใน ซึ่งจะเห็นว่าเป็นการสิ้นเปลืองค่าใช้จ่ายมาก จึงได้มีการพัฒนาโดยจัดทำเป็นระบบชุมสายโทรศัพท์สาขาอัตโนมัติ (PABX) ขึ้น ซึ่งเป็นชุมสายขนาดเล็กสามารถควบคุมได้โดยใช้พนักงานควบคุมเพียงคนเดียว (OPERATOR) แต่ในปัจจุบันได้มีการพัฒนาชุมสายขึ้นโดยไม่ต้องมีพนักงานตัดต่อให้ ทำให้สะดวกรวดเร็วในการติดต่อ

ในการพัฒนาทางด้านอุปกรณ์การสื่อสารอีกอย่างหนึ่งก็คือ การพัฒนาทางด้านระบบชุมสายโทรศัพท์ ซึ่งจะเห็นได้ว่า มีการพัฒนาจากยุคของชุมสายโทรศัพท์ระบบครอสบาร์ (CROSS BAR) ซึ่งเป็นยุคเริ่มต้นของระบบชุมสายโทรศัพท์ จนถึงปัจจุบันนี้ได้มีการพัฒนาจนเป็นระบบชุมสายแบบ ISDN (INTERGRATED SERVICE DIGITAL NETWORK) ซึ่งสามารถรวมการให้บริการต่าง ๆ แบบ ดิจิตอลไว้ในโครงข่ายเดียวกันได้ ทำให้ความสามารถของระบบเพิ่มขึ้นอย่างมาก

ชุมสายโทรศัพท์สาขาปลายทางแบบอัตโนมัติ หรือ PABX (PRIVATE AUTOMATIC BRANCH EXCHANGE) ก็เป็นอุปกรณ์ภายในสำนักงานชนิดหนึ่งที่นิยมใช้กันมากในปัจจุบัน ซึ่งสามารถอำนวยความสะดวกให้แก่ผู้ใช้ภายในชุมสายได้เป็นอย่างมาก สามารถให้บริการผู้ที่ต้องการใช้โทรศัพท์ภายในได้เป็นจำนวนมาก

สำหรับโครงงานนี้ ได้เสนอระบบชุมสายโทรศัพท์สาขาปลายทางแบบอัตโนมัติ ที่มีการควบคุมการทำงานของระบบด้วยไมโครคอนโทรลเลอร์ตระกูล MCS-51 โดยได้แบ่งเป็นส่วนฮาร์ดแวร์ซึ่งขยายจาก 2 คู่สายนอกเป็น 8 คู่สายใน และส่วนซอฟแวร์ที่ควบคุมการทำงานส่วนต่าง ๆ ของระบบ และโครงงานนี้พัฒนาให้สามารถรับฝากเลขหมายผ่านทางโทรศัพท์และแสดงผลการใช้งานต่าง ๆ เพื่ออำนวยความสะดวกในการติดต่อสื่อสาร ทำให้การติดต่อสื่อสารมีประสิทธิภาพมากยิ่งขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 2
ทฤษฎีและหลักการ

2.1 ความรู้เบื้องต้นเกี่ยวกับโทรศัพท์

2.1.1 หน้าที่ของเครื่องโทรศัพท์

โทรศัพท์ที่ใช้กันอยู่ในปัจจุบัน มีอยู่ 2 แบบคือ แบบกดปุ่มและแบบหมุน แต่หน้าที่ของทั้ง 2 ระบบเหมือน ๆ กัน แต่จะต่างกันที่แบบกดปุ่มจะส่งสัญญาณออกไปเป็นความถี่ที่แตกต่างกัน 2 ความถี่ ส่วนแบบหมุนจะส่งสัญญาณเป็นจำนวนพัลส์ หน้าที่หลักๆ ของทั้ง 2 แบบ สามารถสรุปได้ดังนี้

2.1.1.1 เครื่องโทรศัพท์จะรับรู้ว่ามีผู้ต้องการใช้โทรศัพท์ เมื่อมีการยกหูโทรศัพท์ขึ้น

2.1.1.2 เครื่องโทรศัพท์จะส่งสัญญาณที่เรียกว่า สัญญาณหมุน (dial tone) บอกว่าพร้อมที่จะทำการกดปุ่ม หรือหมุนหมายเลขที่จะติดต่อได้ซึ่งก็คือ เสียงที่ได้ยินเมื่อเวลายกหู เป็นสัญญาณเสียงที่มีความถี่ 425 Hz มอดูเลตด้วย 50 Hz

2.1.1.3 เครื่องโทรศัพท์ทำหน้าที่ส่งรหัสหมายเลขที่ผู้เรียกต้องการจะติดต่อด้วยไปยังชุมสายที่ควบคุม

2.1.1.4 เครื่องโทรศัพท์จะส่งสัญญาณบอกผู้เรียกว่า หมายเลขที่ต้องการติดต่อด้วยไม่ว่างก็จะส่งสัญญาณกลับ (ring back) ซึ่งมีความถี่ 425 Hz โดยจะส่ง 1 วินาที แล้วหยุด 4 วินาที สลับกันไป แต่ถ้าหมายเลขที่ต้องการจะเรียกไม่ว่าง ก็จะส่งสัญญาณไม่ว่าง (Busy tone) ไปหาผู้เรียก ซึ่งมีความถี่425 Hz ที่ขาดตอนเป็นช่วง ๆ โดยส่ง 0.3 วินาที หยุด 0.5 วินาที

2.1.1.5 สามารถเปลี่ยนรูปพลังงานเสียงเป็นสัญญาณไฟฟ้า และสัญญาณไฟฟ้ากลับมาเป็นพลังงานเสียง

2.1.1.6 เครื่องโทรศัพท์จะส่งสัญญาณไปชุมสายเพื่อแจ้งให้ทราบว่าสิ้นสุดการใช้งานแล้ว และ ให้ชุมสายเลิกทำการติดต่อกับอีกฝ่ายหนึ่งได้

2.1.2 ระบบโทรศัพท์แบบส่งสัญญาณความถี่คู่ (Dual tone multifrequency type)

เป็นระบบการส่งสัญญาณ ซึ่งจะพบได้มากกว่าในระบบการส่งเป็นสัญญาณพัลส์ ระบบนี้หรือเรียกชื่อย่อว่า DTMF มีวิธีการส่งหมายเลขของผู้ที่ต้องการจะติดต่อด้วย โดยการส่งสัญญาณความถี่ 2 ความถี่พร้อมกันออกไป ซึ่งจะเป็นตัวแทนของหมายเลขที่กด ซึ่งความถี่ที่ถูกส่งออกไปจะอยู่ในย่านความถี่ของเสียงพูด (0-4 kHz) ซึ่งค่าความถี่ที่ต่ำกว่าจะเป็นความถี่ที่แสดงในแนวนอนและอีกค่าหนึ่งก็จะเป็นความถี่ในแนวตั้งซึ่งค่าต่าง ๆ จะแสดงไว้ในรูปที่ 2.1 ตัวอย่างเช่น เมื่อมีการกดหมายเลข 5 ก็จะมีความถี่ 770 Hz และ 1336 Hz มอดูเลตกันออกมา

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ความถี่ต่ำ (Hz)	รหัสหรือหมายเลข			
697	1	2	3	A
770	4	5	6	B
852	7	8	9	C
941	*	0	#	D
	1209	1336	1477	1633

ปุ่มที่ได้รับการเผื่อเอาไว้

ความถี่สูง (Hz)

รูปที่ 2.1 แสดงความถี่ของโทรศัพท์ชนิดกดปุ่ม

2.2 วงจรถอดรหัสความถี่โทรศัพท์

จะใช้ ไอซี MT8870 ถอดรหัสสัญญาณโทรศัพท์ชนิดกดปุ่ม (DTMF) ให้เป็นตัวเลข BCD ขนาด 4 บิทโดยเพียงใช้งานร่วมกับคริสตอล 3.579 MHz เท่านั้น

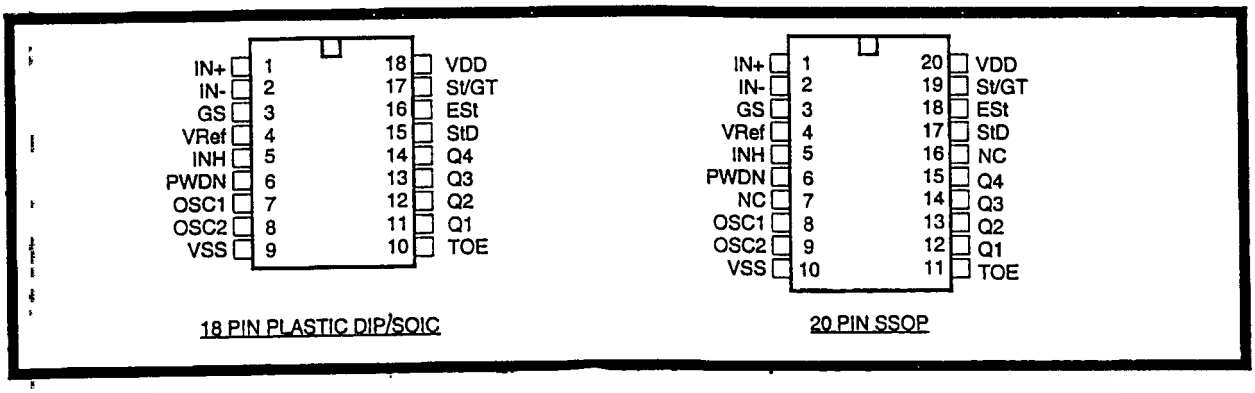
การถอดรหัสความถี่โทรศัพท์ หมายถึง การแปลงสัญญาณความถี่ซึ่งเกิดจากการกดปุ่มตัวเลขของโทรศัพท์ชนิดกดปุ่ม (ชนิด Tone หรือ DTMF) ให้เป็นระบบตัวเลขทางดิจิตอล ซึ่งไอซี MT8870 ใช้แปลงความถี่โทรศัพท์ให้เป็นเลขฐานสองขนาด 4 บิต

ในยุคก่อน การออกแบบวงจรถอดรหัสความถี่ของโทรศัพท์ มักใช้ไอซีจำพวกเฟสล็อกลูปซึ่งสร้างปัญหาสารพัด ไม่ว่าเรื่องของความถี่ที่เปลี่ยนแปลงไป การปรับแต่งวงจร ขนาดของวงจรที่ใหญ่ เพราะต้องใช้ไอซีจำนวนมาก

2.2.1 คุณสมบัติของ MT8870

- เป็นตัวรับและถอดรหัสความถี่ (DTMF receiver)
- กินไฟน้อย ใช้ไฟเลี้ยงระดับเดียวกับ TTL
- สามารถตั้งอัตราขยายภายในตัวไอซีได้
- สามารถปรับการ์ดไทม์ (Guard time) ได้
- เป็นไอซีคุณภาพสูง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.2 แสดงรายละเอียดขาของ MT8870

2.2.2 การนำ MT8870 ไปใช้งาน

- นำไปใช้งานด้านรีโมตคอนโทรล
- เครื่องป้องกันโทรศัพท์ทางไกล
- ใช้ในงานเกี่ยวกับเครดิตการ์ด
- ใช้งานร่วมกับคอมพิวเตอร์
- ใช้ในเครื่องชุมสายขนาดย่อย หรือ PABX
- ใช้กับงานทางด้านโทรศัพท์ทั่วไป
- เครื่องกันขโมย
- การควบคุมอุปกรณ์ทางโทรศัพท์
- ใช้ทำเครื่องสอบถามทางโทรศัพท์

f_{LOW}	f_{HIGH}	KEY	TOE	Q_4	Q_3	Q_2	Q_1
697	1209	1	1	0	0	0	1
697	1336	2	1	0	0	1	0
697	1477	3	1	0	0	1	1
770	1209	4	1	0	1	0	0
770	1336	5	1	0	1	0	1
770	1477	6	1	0	1	1	0
852	1209	7	1	0	1	1	1
852	1336	8	1	1	0	0	0
852	1477	9	1	1	0	0	1
941	1209	0	1	1	0	1	0
941	1336	*	1	1	0	1	1
941	1477	#	1	1	1	0	0
697	1633	A	1	1	1	0	1
770	1633	B	1	1	1	1	0
852	1633	C	1	1	1	1	1
941	1633	D	1	0	0	0	0
-	-	ANY	0	Z	Z	Z	Z

Table 1. MT8870 Output Truth Table

ตารางที่ 2.1 แสดงค่าที่ถอดรหัสได้จากความถี่ต่าง ๆ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.2.3 โครงสร้างของ MT8870

โครงสร้างภายในของ MT8870 ประกอบไปด้วยวงจรกรองความถี่และวงจรถอดรหัสฟังก์ชันทางดิจิตอล เป็นไอซีที่สร้างโดยใช้เทคโนโลยี ISO^2--CMOS ในส่วนของวงจรกรองความถี่ใช้เทคนิคของสวิตช์คาปาซิเตอร์ฟิลเตอร์ สำหรับกรองความถี่สูงและต่ำ ส่วนวงจรถอดรหัสใช้เทคนิคการนับทางดิจิตอลเพื่อตรวจจับและถอดรหัสทั้ง 16 ความถี่ออกเป็นเลขฐานสองขนาด 4 บิต และเช็คช่วงเวลาที่สัญญาณเข้ามา ส่วนภาคอินพุทเป็นออปแอมป์ ซึ่งสามารถปรับอัตราขยายได้โดยต่ออุปกรณ์ภายนอกเอาท์พุทเป็นวงจรแลตช์ 3 สถานะ

2.2.4 ฟังก์ชันการทำงานภายใน MT8870

ภายใน MT8870 ประกอบด้วยส่วนสำคัญ 5 ส่วน คือ

2.2.4.1 ภาคกรองสัญญาณความถี่

ในส่วนนี้จะแยกสัญญาณ DTMF ที่เข้ามาออกเป็น 2 กลุ่มความถี่ คือ ช่วงความถี่สูงและช่วงความถี่ต่ำ โดยใช้วงจรกรองแถบความถี่อันดับ 6 ชนิดสวิตช์คาปาซิเตอร์ (six-order switched band pass filter) ซึ่งความถี่ที่แยกได้มี 2 ช่วง คือช่วงความถี่สูงและช่วงความถี่ต่ำ

2.2.4.2 ภาคถอดรหัส

ความถี่ DTMF ที่ถูกกรองเรียบร้อยแล้วจะผ่านเข้าวงจรถอดรหัสความถี่ออกเป็นตัวเลข โดยใช้เทคนิคการนับแบบดิจิตอล และมีการตรวจสอบความถี่ที่เข้ามาว่าเป็นความถี่มาตรฐาน DTMF หรือไม่ เพื่อป้องกันความถี่อื่นเข้ามาผสมเมื่อตรวจสอบว่าความถี่นั้นถูกต้อง สัญญาณที่ขา Est (early steering) ก็จะเอกทีฟสำหรับค่าที่ถอดรหัสได้จากความถี่ต่าง ๆ นั้น

2.2.4.3 ภาคตรวจสอบสัญญาณ

ก่อนที่จะมีการถอดรหัสความถี่ออกไปที่เอาท์พุท จะมีการตรวจสอบช่วงความถีที่เข้ามาว่ามีระยะเวลาตามที่กำหนดหรือไม่ โดยสังเกตจากระยะเวลาการกดปุ่มโทรศัพท์ ซึ่งต้องกดปุ่มให้มีความถี่ออกมาเป็นระยะเวลาพอสมควรมิฉะนั้นวงจรส่วนนี้จะไม่รับ โดยถือว่าสัญญาณนั้นไม่ถูกต้องส่วนช่วงเวลายาวเท่าใดสามารถตั้งได้โดยใช้ RC ต่อภายนอก สัญญาณที่ขา Est จะเป็น "High"นานใกล้เคียงกับระยะเวลาที่มีความถี่ DTMF เข้ามา เมื่อขา Est เป็น "High" ทำให้ V_c สูงขึ้นตัวเก็บประจุ C จะคายประจุทำให้แรงดัน V_c สูงขึ้นจนถึงค่าเทรชโฮลด์ วงจรถอดรหัส จึงจะถอดรหัสออกเป็นตัวเลขขนาด 4 บิต

2.2.4.4 ภาคขยายสัญญาณความแตกต่าง

วงจรส่วนอินพุทของ MT8870 เป็นภาคขยายออปแอมป์ที่สามารถปรับอัตราขยายโดยต่อวงจรภายนอกเพิ่มเข้าไป

2.2.4.5 ภาคกำเนิดความถี่

ในภาคนี้ภายในไอซีจะมีวงจรเวลาอยู่ภายใน เพียงแต่ต่อแร่คริสตอลขนาด 3.58 MHz ก็สามารถใช้งานได้ทันที

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.3 ISD 12xx /14xx SERIES (ไอซีบันทึกเสียง)

ไอซีในตระกูล ISD นี้อาศัยเทคโนโลยีการบันทึกเสียงทางอนาลอกโดยตรงและภายในประกอบไปด้วยส่วนต่าง ๆ ที่ทำหน้าที่เก็บข้อมูลไว้ภายใน หรือเรียกว่า NV RAM (Nonvolatile RAM) ทำให้สามารถบันทึกเสียงได้นานตั้งแต่ 10-20 วินาที ตามเบอร์ไอซีในตระกูล ISD 12xx/14xx ซึ่งการใช้งานและการจัดขาของไอซีตระกูลนี้เหมือนกันหมดต่างกันตรงความสามารถในการบันทึกเสียงเท่านั้นเองคุณลักษณะที่แตกต่างไปจากไอซีเสียงทั่วไปก็คือระยะเวลาบันทึกเสียงด้วยตัวของไอซีเอง(ไม่มีหน่วยความจำมาต่อเพิ่มภายนอก) สามารถบันทึกเสียงได้นาน โดยจะกำหนดมาเป็นเบอร์ซึ่งแต่ละเบอร์ไอซีจะมีระยะเวลาต่างกันในการบันทึกเสียงและก็อยู่ในตระกูล ISD เช่นกัน

2.3.1 คุณสมบัติของ ISD 12xx/14xx

1. เมื่อใช้ในฟังก์ชั่นการบันทึกและเล่นกลับด้วยตัวไอซีเองง่ายมาก
   - ไม่มีไอซีเบอร์อื่นๆ ประกอบเพิ่มเติมภายนอก
   - ต่ออุปกรณ์พาสซีฟภายนอกน้อยมาก
2. ให้ระดับสัญญาณในการบันทึกที่มีประสิทธิภาพสูง
3. สามารถต่อกับสวิตช์ควบคุมการบันทึกเล่นกลับ, หยุดชั่วขณะและปรับระดับสัญญาณต่างๆ ได้
4. ข้อมูลที่ถูกบันทึกไว้ไม่สูญหายถึงแม้ว่าจะไม่มีแรงดันจ่ายให้กับไอซี และไม่ต้องการแบตเตอรี่สำรอง
5. เก็บข้อมูลได้นานถึง 100 ปี แม้ไม่มีแรงดันไฟเลี้ยง
6. สามารถบันทึกใหม่ได้ถึง 100,000 ครั้ง
7. มีวงจรฐานเวลาภายใน
8. ไม่มีการโปรแกรมในตัวไอซีและไม่ต้องพัฒนาระบบเพิ่มเติมเพื่อให้ทำงานได้
9. มีระบบสแตนด์บายเพื่อประหยัดพลังงานจากแหล่งจ่ายเมื่อไม่มีการบันทึกหรือเล่นกลับ
10. ใช้แรงดันไฟเลี้ยงเดี่ยว กินกระแสขณะสแตนด์บาย 0.5 ไมโครแอมป์

2.3.2 คุณสมบัติทางไฟฟ้าของ ISD 12xx /14xx

จากตารางนี้จะทำให้ทราบว่าค่าพารามิเตอร์บางอย่างที่แตกต่างกันของไอซีแต่ละเบอร์ในตระกูล ISD 12xx/14xx นี้ ซึ่งบอกถึงความสามารถของระยะเวลาในการบันทึกที่แตกต่างกัน รวมทั้งความถี่ของการสุ่มสัญญาณก็ต่างกัน

โดยทั่วไปของ ISD 12xx/14xx แล้ว ก็มีรูปร่างหน้าตาเหมือนกับไอซีโดยทั่วไป แต่ว่าขาใช้งานและบล็อกไดอะแกรมการทำงานภายในทำงานไม่เหมือนกัน โดยเฉพาะไอซีในตระกูล ISD 12xx/14xx นี้จะถูกผลิตขึ้นมาเป็นไอซีที่ทำหน้าที่บันทึกและเล่นกลับ จนจบกระบวนการภายในไอซีเพียงตัวเดียว เพื่อประโยชน์และต้องการให้เป็นอุปกรณ์ ที่ทำหน้าที่บันทึกและเล่นกลับ ก็ถือได้ว่าดีมากไม่แตกต่างจากการบันทึกลงตลับเทปคลาสเซต

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

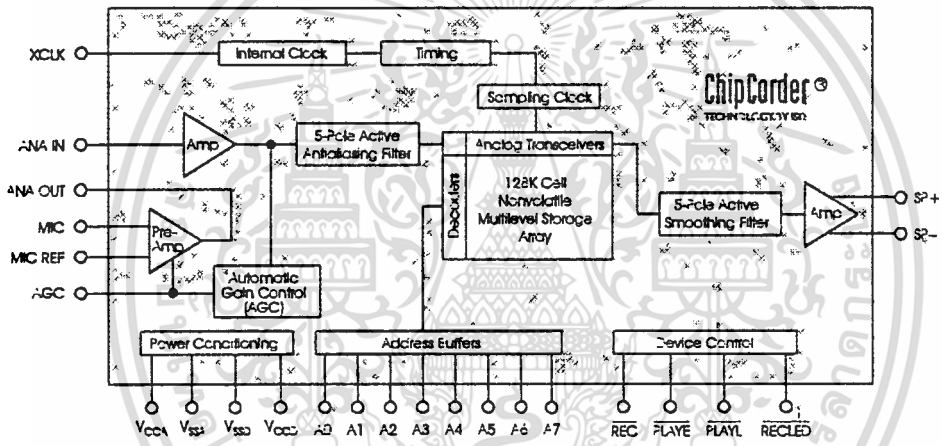
สัญลักษณ์	คุณสมบัติ	ISD1210	ISD1212	ISD1416	ISD1420	หน่วย
F_S	อัตราความถี่ในการแซมปลิ้งสัญญาณ	6.4	5.3	8	6.4	kHz
BW	ความกว้างของแบนด์	2.7	2.3	3.4	2.7	kHz
T_{RPW}	ความกว้างของพัลส์ขณะบันทึก	10	12	16	20	sec
T_{PLAY}	ระยะเวลาในการบันทึก(สูงสุด)	10	12	16	20	sec
T_{LED2}	การหน่วงเวลาที่ขา $\overline{\text{RECLED}}$ ขณะปิด	48.6	58.3	38.9	48.6	msec
T_{RPUD}	การหน่วงเวลาขณะเริ่มบันทึก	32	39	26	32	msec
T_{RPDD}	การหน่วงเวลาขณะหยุดบันทึก	32	39	26	32	msec
T_{PPUD}	การหน่วงเวลาขณะเล่นกลับ	32	39	26	32	msec
T_{PPDD}	การหน่วงเวลาขณะเล่นกลับสิ้นสุด	8.1	9.7	6.5	8.1	msec
THD	ความเพี้ยนทางฮาร์มอนิกรวม	1	1	1	1	%
T_{LED1}	หน่วงเวลาขณะเริ่มบันทึกของ LED	5	5	5	5	µsec
P_{OUT}	กำลังขับลำโพงทางเอาท์พุต	12.2	12.2	12.2	12.2	mW
V_{OUT}	แรงดันตกคร่อมขาต่อลำโพง	2.5	2.5	2.5	2.5	V_{P-P}
V_{IN1}	แรงดันอินพุตที่ไมโครโฟน	20	20	20	20	mV
V_{IN2}	แรงดันอินพุตอนาลอก	50	50	50	50	mV

ตารางที่ 2.2 แสดงคุณสมบัติทางไฟฟ้าของตระกูล ISD12xx/14xx SERIES

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ภายในตัวไอซีจะประกอบด้วยส่วนการทำงานที่สำคัญทุกส่วน โดยมีอุปกรณ์พาสซีฟต่อภายในก็มี วงจรกำเนิดความถี่ฐานเวลาชนิดซีมอส, วงจรขยายสัญญาณไมโครโฟน, วงจรควบคุมอัตราการขยายอัตโนมัติ, วงจรกรองความถี่และวงจรขยายสัญญาณออกสู่ลำโพงโดยตรงจากไอซีตัวนี้แต่ไม่ดังมากพอ สามารถเพิ่มวงจรขยายเสียงภายนอกเพื่อขับลำโพงให้ได้ยินเสียงดังมากขึ้น

สัญญาณที่จะทำการบันทึกจะถูกเก็บหรือบันทึกลงบนหน่วยความจำภายในไอซี ซึ่งเป็นหน่วยความจำที่ไม่ต้องการแรงดันไฟฟ้าสำรองขณะเก็บข้อมูลไว้ได้นานเป็น 100 ปี ลักษณะการบันทึกบนหน่วยความจำภายในไอซีตัวนี้จะทำการบันทึกสัญญาณอนาลอค โดยตรง (Direct Analog Storage Technology : DAST) ซึ่งสัญญาณอนาลอค นี้อาจจะเป็นสัญญาณย่านความถี่ 20 Hz. ถึง 22 kHz. การบันทึกนี้สัญญาณจะถูกเก็บไว้ในหน่วยความจำภายในชนิด EEPROM ซึ่งเป็นอีพรอมที่สามารถบันทึก 100,000 ครั้งตลอดอายุการใช้งาน



รูปที่ 2.3 Block diagram ของ ISD1200/1400 series

2.3.3 การทำงานเบื้องต้น

การทำงานเบื้องต้นของ ISD 12xx/14xx จะเป็นชิพไอซีเพียงตัวเดียวและมีสัญญาณแบบเดี่ยว (single signal) เพื่อควบคุมที่ขา REC และสัญญาณควบคุมการเล่นกลับจะถูกควบคุมด้วยสวิตช์ควบคุมสองสวิตช์ คือ ควบคุมที่ขา PLAYE และ PLAYL นอกจากนั้นหากต้องการให้สามารถควบคุมการบันทึกได้หลายๆ ลักษณะก็สามารถใช้ขาแอดเดรสไลน์ มาทำการประยุกต์ใช้งานควบคุมได้เช่นกัน

- ประสิทธิภาพของเสียงที่บันทึก (Speech Quality)

ประสิทธิภาพของสัญญาณที่ทำการบันทึกและเล่นกลับจะมีคุณภาพของสัญญาณที่ทำการบันทึกและเล่นกลับดีมากเนื่องจากการบันทึกและเล่นกลับจะใช้เทคโนโลยี DAST ดังได้กล่าวมาแล้ว โดยที่สัญญาณที่ทำการบันทึกหรือสัญญาณเสียงพูดที่เข้ามาทางอินพุตจะถูกบันทึกหรือเก็บเข้าไปไว้ใน EEPROM ซึ่งเป็น

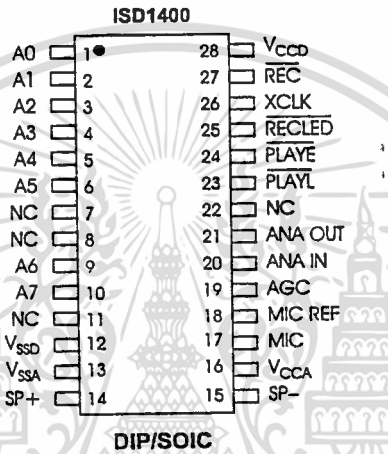
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หน่วยความจำภายในโดยตรงด้วยสัญญาณแบบอนาลอกและการเล่นกลับออกมาก็จะเหมือนกับสัญญาณที่ก่อนทำการบันทึกแน่นอน

-ปิดตัวเองเมื่อไม่มีการบันทึกหรือเล่นกลับ (Automatic Power-Down Mode)

ในขณะที่วงรอบหรือขั้นตอนการทำงานเล่นกลับหรือบันทึกสิ้นสุดลงไอซีนี้ก็จะมีฟังก์ชันการทำงานให้ตัดเข้ามาสู่โหมดของการสแตนด์บายเพื่อให้ปริมาณการใช้กำลังต่ำ ซึ่งกินกระแสเพียง 0.5 ไมโครแอมป์เท่านั้น ในช่วงที่ทำการเล่นกลับจบลงวงจรภายในก็จะตัดกลับมาสู่สภาวะสแตนด์บาย ในโหมดของการบันทึกหลังจากที่ทำการบันทึกเสร็จสิ้นลง ก็จะกลับมาสู่โหมดสแตนด์บาย เมื่อขาควบคุม REC มีระดับลอจิกเป็น "1"

2.3.4 การทำงานของขาใช้งานแต่ละขา สามารถอธิบายถึงหน้าที่และการทำงานของขาที่สำคัญได้ดังนี้



รูปที่ 2.4 แสดงรายละเอียดขาของ ISD1200/1400 series

- ขาควบคุมการบันทึก ($\overline{\text{REC}}$)

ที่ขาควบคุมการบันทึกทางอินพุตนี้จะต้องการระดับลอจิก "0" เพื่อทำการบันทึกสัญญาณและจะเริ่มทำการบันทึกเมื่อระดับลอจิกที่ขา REC นี้เป็นลอจิก "0" และสภาวะลอจิกที่ขานี้จะต้องคงสถานะอยู่ที่ "0" ตลอดขณะทำการบันทึก และการบันทึกที่ขา REC จะต้องได้รับสัญญาณให้ทำการบันทึกก่อนที่จะทำการเล่นกลับหรือก่อนที่จะมีสัญญาณมาควบคุมที่ขา PLAYE หรือ ขา PLAYL ถ้าที่ขา REC มีระดับลอจิก "0" ขณะที่มีการเล่นกลับ การเล่นกลับจะหยุดและจะเริ่มทำการบันทึก เมื่อการบันทึกจบลง ขา REC จะเพิ่มขึ้นไปเป็นไปเป็น "1" แล้วจะมีมาร์คเกอร์ (marker) ของจุดสิ้นสุดของข้อมูล จากนั้นจะเข้าสู่โหมดสแตนด์บาย

- ขาควบคุมการเล่นกลับอย่างสมบูรณ์ ($\overline{\text{PLAYE}}$)

เมื่อขาควบคุมการเล่นกลับนี้ได้รับระดับลอจิกเป็น "0" หรือได้รับการกระตุ้นด้วยลอจิก "0" ที่อินพุตนี้วงจรก็จะเริ่มทำการเล่นกลับเพื่อนำข้อมูลที่ถูกบันทึกอยู่แสดงออกมาทางลำโพง ฟังก์ชั่นการเล่นกลับนี้ จะเป็นการเล่นกลับอย่างต่อเนื่อง จนกว่าจะถึงข้อมูลสุดท้ายที่ทำการบันทึกตามเวลาที่กำหนดไว้ (10-20 วินาที) หรือการเล่นกลับจนกว่าข้อมูลที่บันทึกไว้ใน EEPROM ทุกข้อมูลจะถูกเล่นกลับออกมาทั้งหมด ซึ่งเป็นการเล่นกลับอย่างสมบูรณ์ หลังจากนั้นก็จะตัดเข้าสู่โหมดสแตนด์บาย ในระหว่างที่กำลังอยู่ในสภาวะเล่นกลับนั้น ถ้าที่ขา PLAYE มีสถานะเป็น "1" การเล่นกลับก็จะยังไม่หยุด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- ขาควบคุมการเล่นกลับ ($\overline{\text{PLAYL}}$)

เมื่อขาอินพุตนี้มีการเปลี่ยนแปลงระดับลอจิกเป็น "1" ไปถึง "0" จะเป็นการเล่นกลับแบบต่อเนื่องจนกระทั่งที่ขา PLAYL เพิ่มขึ้นเป็น "1" หมายถึง เกิดการตรวจจับการเล่นสิ้นสุดลงแล้ว หรือตรวจจับ พบมาร์คเกอร์ ซึ่งเป็นจุดสิ้นสุดข้อมูล หรือถึงจุดสิ้นสุดของข้อมูลในหน่วยความจำ และก็จะกลับมาสู่สภาวะสแตนด์บาย

- ขาอินพุทของสัญญาณอนาลอก (ANA IN)

ค่า C ภายนอกที่ต่อระหว่างขา ANA IN และขา ANA OUT สามารถกำหนดความถี่ cut off ที่ low - frequency ของความถี่เสียงได้ โดยสามารถที่จะนำสัญญาณอนาลอกมาใส่เข้าที่ขานี้ได้เลย (แทนสัญญาณจากไมโครโฟน) โดยต่อผ่าน C คัปปลิ้ง เข้ามา

- ขา Automatic Gain Control (AGC)

จุดประสงค์ของสัญญาณ AGC คือควบคุมค่า gain ของวงจร preamplifier และช่วยขยายช่วงความดังของ input signal จากเสียงกระซิบจนถึงเสียงตะโกน ซึ่งจะควบคุมให้เอาท์พุทไม่ดังเกินไป

2.3.5 การกำหนดแอดเดรส

การกำหนดแอดเดรสของการบันทึกและการเล่นกลับนั้น สามารถกำหนดได้โดยขา A0 - A5 เมื่อขา A7 ต้องเป็นลอจิก " 0 " และจะใช้ขา A6 เป็นตัวกำหนดว่าจะเป็น 10 วินาทีแรกหรือ 10 วินาทีหลัง

- เมื่อขา A6 เป็นลอจิก " 0 " จะกำหนดเป็น 10 วินาทีแรก
- เมื่อขา A6 เป็นลอจิก " 1 " จะกำหนดเป็น 10 วินาทีหลัง

ในการบันทึกหรือการเล่นกลับนั้น จะต้องกำหนดตำแหน่งแอดเดรสก่อน แล้วจึงใช้จะสามารถควบคุมให้บันทึกหรือเล่นกลับได้ โดยการควบคุมที่ขา PLAYL และ REC ตามลำดับ

2. 4 ชุมสายโทรศัพท์

เพื่อให้การใช้งานของคู่สายโทรศัพท์เกิดประโยชน์เต็มที่มากที่สุด คือ สามารถติดต่อกันได้ทุกคู่สายไม่ว่าจะภายในประเทศ หรือต่างประเทศเนื่องจากจำนวนคู่สายของ โทรศัพท์มีมากจึงไม่สามารถนำโทรศัพท์ทุกคู่สายต่อถึงกันโดยตรงได้ จึงได้มีระบบชุมสายโทรศัพท์ขึ้นมาและจากเหตุผลอีกประการหนึ่ง คือ คู่สายแต่ละคู่สายไม่ได้ใช้งานตลอดเวลาเพราะฉะนั้นเพื่อให้เกิดประโยชน์กันสายส่งเต็มที่ก็จะแบ่งกันใช้เป็นเวลา เมื่อคู่สายคู่หนึ่งไม่มีการใช้ก็ยังสามารถแบ่งไปให้คู่สายอีกคู่หนึ่งใช้งานได้ ซึ่งชุมสายโทรศัพท์ที่มีใช้อยู่มากมายสามารถแบ่งได้ดังนี้

ชุมสายโทรศัพท์สามารถแบ่งได้เป็น 2 ประเภทใหญ่ ตามการใช้งาน คือ

2.4.1 ชุมสายที่มีผู้เช่าต่อเข้าโดยตรง ได้แก่

ชุมสายโทรศัพท์ท้องถิ่น (LOCAL EXCHANGE) หมายถึง ชุมสายโทรศัพท์ที่มีเครื่องของผู้เช่าต่อเข้าโดยตรง ชุมสายโทรศัพท์แบบนี้มีขนาดตั้งแต่เป็นร้อย ๆ เลขหมายจนถึงเป็นหมื่นเลขหมาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ตู้สาขา (PRIVATE AUTOMATIC BRANCH EXCHANGE) หรือที่เรียกกันว่า PABX เป็นชุมสายโทรศัพท์ ที่มีลักษณะคล้ายกับชุมสายโทรศัพท์ท้องถิ่น แต่จะใช้ติดต่อกันภายในสำนักงานโดยไม่ต้องผ่านชุมสายโทรศัพท์ท้องถิ่น หรือถ้ามีการติดต่อไปยังภายนอกก็สามารถเรียกผ่านชุมสายท้องถิ่นได้

2.4.2 ชุมสายโทรศัพท์ต่อผ่าน หมายถึง ชุมสายโทรศัพท์ที่ไม่มีเลขหมายโทรศัพท์ของผู้เช่าต่อเข้ามาโดยตรง แต่จะบริการเรียกระหว่างชุมสายท้องถิ่นกับชุมสายท้องถิ่น ซึ่งการเรียกกันระหว่างเครื่องโทรศัพท์ 2 เลขหมายอาจจะผ่านชุมสายต่อผ่านได้หลาย ๆ ชุมสาย ชุมสายโทรศัพท์ต่อผ่านยังแบ่งได้เป็น 2 ชนิด ได้แก่

ชุมสายต่อผ่านท้องถิ่น (TANDEM EXCHANGE) ซึ่งเป็นชุมสายโทรศัพท์ ที่ใช้ต่อผ่านภายในโครงข่ายท้องถิ่นนั้น เช่น ภายในกรุงเทพ ฯ

ชุมสายต่อผ่านทางไกล (TRANSIT EXCHANGE) ซึ่งเป็นชุมสายโทรศัพท์ต่อผ่านไปยังโครงข่ายท้องถิ่นอื่น ๆ เช่น จากกรุงเทพ ฯ ไปยังหาดใหญ่

จากชุมสายโทรศัพท์ที่กล่าวมาทั้งหมดอุปกรณ์ที่เกี่ยวข้องและสำคัญ คือ อุปกรณ์สวิตชิ่ง (SWITCHING DEVICE) จะเป็นอุปกรณ์ที่ทำหน้าที่ต่อคู่สายผู้เรียกไปยังผู้ถูกเรียก สามารถแบ่งได้เป็น 2 แบบ คือ

1.แบบอนาล็อก (ANALOG)

การทำงานของอุปกรณ์สวิทชิ่งโทรศัพท์ คือการรับสัญญาณหมายเลขจากสมาชิกต้นทางคือ ผู้เรียก อาจจะเป็นแบบพัลซ์ (แบบแป้นหมุน) หรือแบบความถี่ผสม (แบบกดปุ่ม) เพื่อที่จะถอดรหัสหมายเลขปลายทางและทำการต่อวงจรให้ติดต่อกันได้ โดยที่รูปแบบของสัญญาณต้นทางและปลายทางยังคงเหมือนเดิมทุกประการ เราเรียกสวิทชิ่งแบบนี้ว่า แบบอนาล็อก

2.แบบดิจิตอล (DIGITAL)

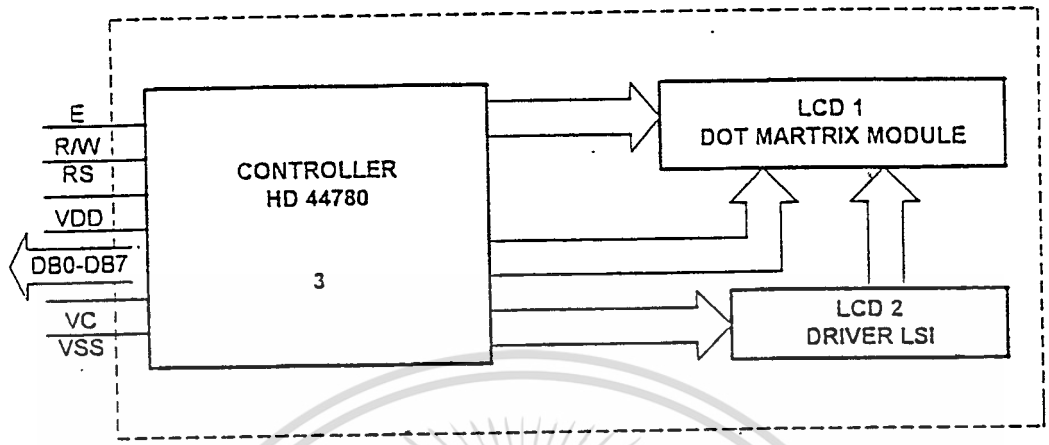
อุปกรณ์ที่ถูกนำมาใช้ในชุมสายโทรศัพท์แบบดิจิตอล จะมีหลักการที่สำคัญ คือการเปลี่ยนสัญญาณเสียงให้เป็นสัญญาณดิจิตอล (รหัส “0” และ “1”) ซึ่งต่างจากชุมสายโทรศัพท์แบบอนาล็อกซึ่งสัญญาณเสียงที่ได้จากเครื่องโทรศัพท์จะถูกเปลี่ยนให้อยู่ในรูปของรหัส “0” และ “1” การทำงานอย่างนี้เราเรียกว่า การเข้ารหัส (ENCODER) ซึ่งสัญญาณรหัส “0” และ “1” นี้เราเรียกว่า สัญญาณดิจิตอล

ปัจจุบันอุปกรณ์แสดงผลเพื่อติดต่อกับผู้ใช้ให้สามารถควบคุมและใช้งานได้สะดวกขึ้น เช่น บอกสถานะการทำงาน บอกข้อผิดพลาดที่เกิดขึ้นระหว่างทำงาน ส่วนแสดงผลอาจเป็นไฟบอกสถานะอย่างง่ายๆหรืออาจเป็นจอแสดงผลซึ่งเป็นที่นิยมกันมากแสดงข้อความเป็นตัวอักษรได้ จอแสดงผลมีข้อดีหลายประการ เช่น ใช้LED(Light Emitting Diode) หรือ LCD (Liquid Crystal Display) ซึ่งจอแสดงผลประเภทที่กำลังเป็นที่นิยมในปัจจุบันได้แก่แบบที่ใช้ LCD เนื่องจากใช้พลังงานน้อย(กินกระแสไฟต่ำ)และมีความละเอียดสูงมากสามารถแสดงตัวอักษรและรูปภาพได้หลายแบบซึ่งผู้ใช้สามารถต่อเข้ากับระบบไมโครคอนโทรลเลอร์ได้ง่ายและสะดวก สำหรับการเขียนโปรแกรมทำให้ผู้ใช้งานด้านนี้นิยมใช้จอแสดงผลเป็นส่วนหนึ่งของผลิตภัณฑ์ เช่น เครื่องเล่นวีดีโอ เครื่องเล่นคอมแพคดิสก์ วิทยุซาวด์เบาต์ เป็นต้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5 ส่วนแสดงผลชนิด LCD MODULE

ปัจจุบัน LCD ที่ขายในท้องตลาดส่วนใหญ่จะประกอบเป็นโมดูล เพื่อให้สะดวกในการใช้งาน โดยจะมีส่วนประกอบทั่วไปดังในรูปที่ 2.5



รูปที่ 2.5 โครงสร้างทั่วไปของ LCD MODULE

2.5.1 DOT MATRIX LCD MODULE แบ่งออกเป็นประเภทได้ดังนี้

- CHARACTER LCD MODULE
- GRAPHIC LCD MODULE
- SEGMENT DISPLAY TYPE LCD MODULE

ซึ่งจอแสดงผล LCD ที่นำมาใช้เป็นประเภท CHARACTER LCD MODULE ที่เรียกย่อๆว่า LCM มีส่วนประกอบใหญ่ๆแบ่งเป็น

1.DOT MATRIX LCD เป็นตัวแสดงผลให้สามารถมองเห็นในลักษณะการปิดและเปิดตัวเองกับแสงคือส่วนที่เป็นตัวกระจกบรรจุผลึกอยู่ซึ่งใช้หลักการหักเหของแสงผ่านผลึกโดยจะประกอบไปด้วยจุด(pixel) จำนวนมากที่สามารถบังคับให้ติดหรือดับได้ทุกจุด การแสดงผลจะเป็นตัวอักษรขนาด 5*8 จุดและมีตัวอักษรและบรรทัดแตกต่างกันไปในแต่ละรุ่น

2.DRIVER เป็นวงจรที่ขับ LCD โดยรับสัญญาณจากตัวควบคุมมาขับผลึกLCDอีกทีหนึ่งโดยมีเบอร์ที่นิยมใช้เช่น HD44100H MSM5295

3.CONTROLLER เป็นตัวรับข้อมูลจากอุปกรณ์ภายนอกมาจัดการควบคุมการทำงานทั้งหมดของLCD MODULE ให้ทำงานแสดงผลต่างๆ เช่น การแสดงตัวอักษร การลบจอแสดงผล เป็นต้น โดยมีไอซีเบอร์ที่นิยมใช้เช่น HD44780 ซึ่งจะใช้ในแบบ CHARACTER LCD MODULE และเบอร์ HD61830 ใช้ในแบบ GRAPHIC LCD MODULE

คุณสมบัติทั่วไปของจอแสดงผลแบบ DOT MATRIX สามารถสรุปเป็นข้อๆได้ดังนี้

1. มีให้เลือกหลายรุ่นตามการใช้งาน โดยมีจำนวนตัวอักษรและบรรทัดแตกต่างกันไป
2. ตัวอักษรแสดงด้วย จุดขนาด 5*8 จุด
3. สามารถต่อเข้ากับระบบไมโครคอนโทรลเลอร์ได้ 2 ลักษณะคือ แบบแผนภูมิหน่วยความจำ (20 -PIN LCD BUS) และแบบผ่านพอร์ท 8255 (26 - PIN 8255 BUS) โดยกรณีนี้จะใช้แผ่นแผงวงจร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(DMCAD) เป็นตัวแปลงสัญญาณ (Adapter) ให้กับ 8255 อีกที

4. การใช้งานง่ายและสะดวก ระบบไมโครคอนโทรัลเลอร์เพียงแค่ส่งข้อมูลให้กับแผงแสดงผลเท่านั้น ข้อความก็จะปรากฏบนแผงแสดง และจะคงค้างไว้ตลอด ทำให้ไม่ต้องเสียเวลาหลักของระบบ

5. มีคำสั่งพิเศษสำหรับอำนวยความสะดวกมากมาย เช่น ล้างจอแสดงผล (Clear Display), แสดงหรือไม่แสดงเคอร์เซอร์ (On Off Curser) และอื่น ๆ อีก

6. สามารถแสดงผลเป็นตัวอักษรภาษาอังกฤษและตัวเลขได้ 160 ตัว และสัญญลักษณ์พิเศษ

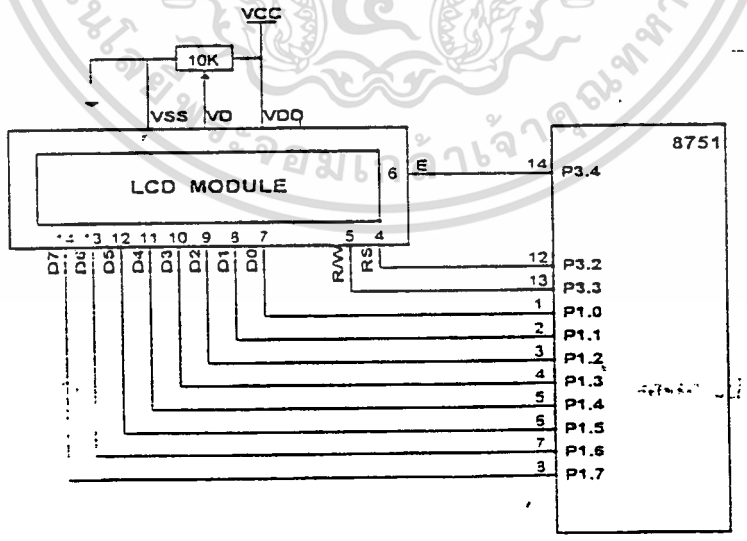
7. กินกระแสน้อย และมีน้ำหนักเบา รวมทั้งทำงานได้ด้วยไฟเลี้ยงระดับ 5 โวลท์เท่านั้น

การใช้งานLCDผู้ใช้เพียงแต่ศึกษาและทำความเข้าใจในส่วนคอลโทรลเลอร์ของLCMเท่านั้นเพราะส่วนนี้เป็นส่วนที่รับข้อมูลที่ต้องการแสดงผลจากวงจรภายนอกและควบคุมการทำงานทั้งหมดของLCMโดยจะกล่าวถึงเฉพาะที่เป็นคอลโทรลเลอร์เบอร์ HD44870 ส่วนชิปคอลโทรลเลอร์เบอร์อื่นส่วนใหญ่จะมีการใช้งานที่คล้ายกับเบอร์นี้มาก ชิปเบอร์HD44870 เป็นของบริษัท HITACHI สามารถใช้งานเพื่อควบคุม LCM กับไมโครคอลโทรลเลอร์หรือไมโครโปรเซสเซอร์ทั้งแบบ 4บิต หรือ 8บิต ได้

เนื่องจากไมโครคอนโทรลเลอร์ตระกูล MCS - 51 มีดาต้าบัสขนาด 8 บิต ดังนั้นเราจะกล่าวถึงเฉพาะการติดต่อในแบบ 8 bit I operation เท่านั้น ตัวอย่างของวงจรในการอินเตอร์เฟส MCS - 51 กับ LCM มีดังแสดงในรูปที่ 2.6

จากในรูปจะเห็นว่า LCM ติดต่อกับ MCS -51 โดย

- ใช้ขา P1.0 - P1.7 เป็นดาต้าบัส (DB0 - DB7) ในการติดต่อ
- ใช้ขา P.3.2 เป็นขาสัญญาณ RS
- ใช้ขา P.3.3 เป็นขาสัญญาณ R/W
- ใช้ขา P.3.4 เป็นขาสัญญาณ EN(E)



รูปที่ 2.6 แสดง การอินเทอร์เฟส MCS-51 กับ LCM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.2 ความเข้าใจพื้นฐาน

- การเขียนข้อมูลให้กับแผงแสดงผล จะแบ่งเป็น 2 ลักษณะคือ สัญญาณควบคุม (Instruction) และ ข้อมูล (Data) โดยจะกำหนดด้วยขาสัญญาณ RS = 0 จะหมายถึงส่งสัญญาณควบคุม หรืออ่านค่าแฟลก (Flag) สภาพการทำงานของแผงแสดงผล และถ้า RS = 1 จะหมายถึง การเขียนหรืออ่านข้อมูล กับจอแสดงผล

- หลักการในการเขียนข้อมูลให้แผงแสดงผลนี้ คือเมื่อมีการเขียนข้อมูลไปและตัวจอแสดงผล จะต้องใช้เวลาในการทำงานชั่วขณะหนึ่ง (ตามค่าเวลาการเอ็กซ์ซิคิวท์ ในตารางที่ 2.3) ซึ่งระบบไมโครคอนโทรลเลอร์สามารถตรวจสอบได้จาก บีซี่แฟลก (Busy Flag,BF) และถ้าเรียบร้อยแล้วจึงสามารถเขียนข้อมูลอันต่อไปได้ ในกรณีที่การต่อวงจรเป็นแบบอินพุทเอาท์พุทพอร์ทคือไม่สามารถอ่านข้อมูลย้อนกลับได้ ระบบไมโครคอนโทรลเลอร์ก็จะต้องใช้วิธีการหน่วงเวลาแทน

- การเขียนข้อมูลให้กับแผงแสดงผล นี้ สามารถทำได้ทั้งแบบ 8 บิท และ 4 บิทโดยกรณี 4 บิท จะใช้สายสัญญาณข้อมูล เพียง 4 เส้น คือ DB4 -DB7 (ใช้สำหรับระบบไมโครคอนโทรลเลอร์ แบบ 4 หรือเพื่อการประหยัดสาย) การเขียนข้อมูลจะกระทำเหมือนกับ 8 บิท เพียงแต่ให้เขียน 2 ครั้ง คือ DB4 -DB7 ก่อน แล้วตามด้วย DB0 -DB3 และจะต้องกำหนดคุณสมบัติตามค่า DL ในคำสั่งกำหนดหน้าที่การทำงาน (Function Set) ด้วย

- หน่วยความจำแสดงผลข้อมูล (Display Data Ram , DDRAM) คือหน่วยความจำภายในตัวแผงแสดงผลที่เป็นที่พัก (BUFFER) ของข้อมูล โดยถ้าเขียนรหัสแอสกีใด ๆ ลงไปในหน่วยความจำนี้ ก็จะปรากฏเป็นตัวอักษรที่แผงแสดงทันที

- หน่วยความจำอ้างอิงตัวอักษร (CHARACTER GENERATER RAM = CGRAM) คือหน่วยความจำภายในตัวแผงแสดงผล สำหรับเก็บภาพตัวอักษรที่ผู้ใช้สามารถสร้างได้เอง (8 ตัว) โดยจะอ้างแอสเดรส ได้ทั้งหมด 64 ไบท์ คือ 8 หลักตัวอักษร คูณกับ 8 แถว

Signal	I/O	Device Interface with	Function
RS	I	MPU	Selects registers. 0:Instruction register(for write) Busy flag:address counter(for read) 1:Data register(for write and read)
R/W	I	MPU	Selects read or write. 0:Write 1:Read
E	I	MPU	Starts data read/write.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Signal	I/O	Device Interface with	Function
DB4 to DB7	I/O	MPU	Four high order bidirectional tristate data bus pins. Used for data transfer and receive between the MPUand the HD44780 .DB7 can be used as a busy flag.
DB0 to DB3	I/O	MPU	Four low order bidirectional tristate data bus pins. Used for data transfer and receive between the MPU and the HD44780 .These pins are not used during 4-bits operation.
COM1 to COM16	O	LCD	Common signals that are not used are changed to non-selection waveforms.COM9 to COM16 are non-selection waveforms at 1/8 duty factor and COM12 to COM16 are non-selection waveform at 1/11 duty factor.
SEG1 to SEG40	O	LCD	Segment signals.
CL1	O	Extention driver	Clock to latch serial data D sent to the extention driver.
CL2	O	Extention driver	Clock to Shift serial data D
L	O	Extention driver	Character pattern data corresponding to each segment signal
V1 to V5	—	Power supply	Power supply for LCD drive V_{CC}-V5 = 11V(max)
V_{CC} , GND	—	Power supply	V_{CC}:2.7V to 5.5V , GND:0V
OSC1 ,OSC2	—	Oscillation resister clock	When crystal oscillation is performed,a resister must be connected externally.When the pin input is an external clock,it musy be input to OSC1

ตารางที่ 2.3 แสดงขาสัญญาณ ของ จอแสดงผล LCD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.5.3 รายละเอียดของแต่ละขาสัญญาณ มีดังนี้

1.RS (Register Selection) จะเป็นขาเลือกรีจิสเตอร์ ภายในซึ่งในชิปคอลโทรลเลอร์มีรีจิสเตอร์อยู่สองประเภทคือ รีจิสเตอร์คำสั่ง (Command Register หรือ Instruction Register :IR) และรีจิสเตอร์ข้อมูล (Data Register :DR) โดย

สัญญาณ RS = 0 หมายถึงเลือกใช้ ชุดคำสั่ง

สัญญาณ RS =1หมายถึงเลือกใช้ข้อมูล

2. R/W(Read/Write) เป็นสัญญาณที่เลือกการเขียนหรืออ่านข้อมูลจาก LCMโดย

สัญญาณ R/W =0 หมายถึงต้องการเขียนข้อมูลจาก LCM

สัญญาณ R/W =1 หมายถึงต้องการอ่านข้อมูลไปยัง LCM

3. E (Enable Signal) เป็นขากำหนดสภาพการรับเขียนอ่านข้อมูล มีรายละเอียดดังแสดงใน ตารางที่ 2.4

RS	R/W	E	OPERATION
0	0		Write instruction code
0	1		Read busy flag and address counter
1	0		Write data
1	1		Read data

ตารางที่ 2.4 แสดงการทำงานของสัญญาณ Enable

4. DB0-DB7 เป็นขารับส่งข้อมูลจากตัวไอซี

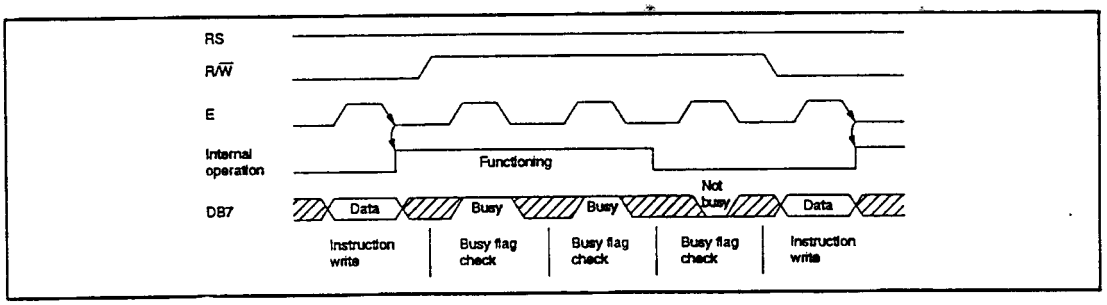
5. VDD เป็นขาจ่ายไฟเลี้ยงวงจรขนาด 5 โวลต์

6. VSS เป็นขาGND

จากรูปที่ 2.7 ซึ่งเป็นแผนผังเวลาในการตรวจสอบ Busy Flag และจากตารางจะเห็นได้ว่าในการเขียนรหัสคำสั่งทุกครั้ง

- RS และR/W ต้องมีค่าเป็น ศูนย์และส่งข้อมูลไปในขณะที่สัญญาณEnable เปลี่ยนจาก 1เป็น 0ในการเขียนข้อมูลทุกครั้ง
- RS =1และ R/W=0 และส่งข้อมูลไปในขณะที่สัญญาณEnable เปลี่ยนจาก 1เป็น 0ในการอ่านBusy Flag และ Adress Counter ทุกครั้ง
- RS =0 และ R/W=1 และรับข้อมูลเข้ามาในขณะที่สัญญาณ Enable เป็น 1ในการอ่านข้อมูลทุกครั้ง
- RS=1 และ R/W=1 และรับข้อมูลเข้ามาในขณะที่สัญญาณ Enable เป็น1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.7 แสดงแผนผังเวลาในการติดต่อกับLCM

2.5.4 ชุดคำสั่งควบคุมและการแสดงข้อความ

การเขียนหรืออ่านข้อมูลกับแผงแสดงผล ก็คือการกำหนดคุณสมบัติต่าง ๆ ในการใช้งานของจอแสดงผล ตามชุดคำสั่งควบคุมและรวมถึงการเขียนข้อมูลที่เป็นข้อความ เพื่อให้ปรากฏบนจอแสดงผล ชิปคอลโทรลเลอร์ HD44780 มีการทำงานได้หลายอย่างขึ้นอยู่กับคำสั่งที่รับเข้ามา การเริ่มต้นใช้งาน LCM เพื่อให้ทำงานได้ถูก ต้องจำเป็นต้องมีการรีเซตเช่นเดียวกับชิปไมโครโปรเซสเซอร์การรีเซต HD44780 มีวิธีดังนี้

1. เริ่มต้นใช้งานโดยวงจรที่ทำหน้าที่รีเซตภายในชิปคอลโทรลเลอร์ HD44780 สามารถรีเซตวงจรภายในได้เองในขณะเริ่มต้นใช้งานโดยมันจะเริ่มทำงานด้วยคำสั่ง เมื่อมีการรีเซต สัญญาณบีซีแฟลก (Busy Flag) จะอยู่ในสถานะไม่ว่าง (Busy) จนกระบวนการรีเซตภายในเสร็จสัญญาณบีซีแฟลก จะมีค่าเป็น 1 เป็นเวลา 10 วินาที ภายหลังที่ไฟเลี้ยงชิปมีค่าเป็น 4.5 โวลต์ คำสั่งที่ทำงานในช่วงไม่ว่าง หรือในช่วงรีเซตมีดังต่อไปนี้

-Display Clear ทำการลบข้อมูลบนจอภาพLCD

-Function Set โดยเซตค่าภายใน

DL =1 :เป็นการเซตให้มีการติดต่อแบบ 8 บิต

N=0 : เซตเป็น 1 บรรทัดการแสดงผล

F=0 :5*8 Dot ต่อหนึ่งตัวอักษร

-Display On /Off

D=0 : Display Off

C=0 :Cursor Off

B=0 :Blink Off

-Entry Mode Set I/D=1 :+1 (เพิ่มค่าCounterขึ้น 1)

S=0 :No Shift

หลังจากช่วงนี้ผ่านไปเริ่มต้นใช้งานได้ เมื่อเราเริ่มเปิดเครื่องทำงานแล้วก็จะต้องส่งคำสั่งควบคุมให้เริ่มทำงานดังตารางที่ 2.5 หาก Rise Time ของแหล่งจ่ายไฟ(ช่วงเวลาที่แรงดันเปลี่ยนจาก 0.2โวลต์ เป็น 4.5โวลต์) มีค่าอยู่นอกช่วง 0.1-10 มิลลิวินาทีหรือเมื่อไฟเลี้ยงที่จ่ายให้ LCM มีค่าตกลงมาต่ำกว่า0.2โวลต์น้อยกว่า 1 มิลลิวินาทีวงจรที่ทำหน้าที่รีเซตโดยอัตโนมัติภายในชิปจะไม่สามารถทำงานได้ ในกรณีที่การรีเซตของวงจรภายในทำงานผิดพลาดเนื่องมาจากสาเหตุใดก็ตามจำเป็นที่วงจรภายนอกจะต้องทำการรีเซต LCM เพื่อให้การใช้งานไม่เกิดข้อผิดพลาดทั้งนี้โดยการส่งคำสั่งไปควบคุมLCM

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2. ค่าเริ่มต้นใช้งานโดยส่งคำสั่งไปควบคุมLCMเองเพื่อป้องกันไม่ให้เกิดการรีเซตผิดพลาดหรือเพื่อให้แน่ใจว่า LCM ทำงานได้ถูกต้องแน่นอน วงจรภายนอกควรทำการรีเซต LCM เอง

Instruction	Code										Execution time (f_{osc} is 270kHz)
	RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0	
Clear display	0	0	0	0	0	0	0	0	0	1	-
Cursor home	0	0	0	0	0	0	0	0	1	*	1.52ms
Entry mode set	0	0	0	0	0	0	0	0	I/D	S	37μs
Display on/off control	0	0	0	0	0	0	1	D	C	B	37μs
Cursor/display shift	0	0	0	0	0	1	S/C	R/L	*	*	37μs
Function set	0	0	0	0	1	DL	N	F	*	*	37μs
Set CGRAM adress	0	0	0	1	CGRAM adress						37μs
Set DDRAM address	0	0	1	DDRAM address							37μs
Read busy flag and address counter	0	1	BF	DDRAM address							0μs
write CGRAM or DDRAM	1	0	write data								37μs
read from CGRAM or DDRAM	1	1	read data								37μs

* = Don't care

Execution time changes when frequency changes

Example: When f_{osc} is 250 kHz , 37μs x 270 / 250 = 40 uS

ตารางที่ 2.5 แสดงรายละเอียดชุดคำสั่ง HD44780

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

รายละเอียดของชุดคำสั่ง HD44780

1. คำสั่งลบจอแสดงผล (CLEAR DISPLAY)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	0	1

คำสั่งลบจอแสดงผล (CLEAR DISPLAY) ทำโดยการเขียนตัวอักษรว่าง (Space : ASCII 20H) ลงในหน่วยความจำแสดงผลข้อมูลทั้งหมดและกำหนดค่าตำแหน่งหน่วยความจำแสดงผลข้อมูล (DDRAM Address) ให้เป็น 0 พร้อมทั้งเคอร์เซอร์จะกลับไปที่ตำแหน่งซ้ายบนสุดของจอภาพ เซ็ท I/D = 1 ,S ไม่มีการเปลี่ยน

2. คำสั่งให้เคอร์เซอร์ไปอยู่ที่จุดเริ่มต้น (RETURN HOME)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	1	*

* หมายถึง ไม่มีผล (NO EFFECT)

คำสั่งให้เคอร์เซอร์ไปอยู่ที่จุดเริ่มต้น (Cursor At Home) สำหรับกำหนดค่าตำแหน่งของหน่วยความจำแสดงผลให้เป็น 0 พร้อมทั้งเคอร์เซอร์จะไปอยู่ที่ตำแหน่งซ้ายบนสุดของจอภาพโดยที่ข้อมูลในหน่วยความจำแสดงผล ไม่มีการเปลี่ยนแปลง

3. คำสั่งกำหนดรูปแบบการใส่ข้อมูล (ENTRY MODE SET)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	0	0	I/D	S

- BIT I/D : โดยจะเป็นตัวกำหนดให้ว่าเมื่อเขียนหรืออ่านข้อมูลแล้วจะทำให้ DD RAM ADDRESS เพิ่มขึ้นหนึ่งหรือลดลงหนึ่งโดย 1 = เพิ่ม , 0 = ลด

- BIT S : เป็นตัวกำหนดการแสดงผลโดยถ้า S =1 เป็นการใส่ข้อมูลแล้วตัว CURSOR อยู่กับที่ข้อมูลจะถูกดันไปทางซ้าย (จะถูกเลื่อนไปในทิศทางตามค่า I/D) ถ้า S =0 ข้อมูลจะอยู่กับที่ ตัว CURSOR จะไปทางขวามือ (จะถูกเลื่อนไปในทิศทางตามค่า I/D)

การกำหนด I/D และ S นี้ ให้กำหนดก่อนการเขียนข้อมูลในหน่วยความจำแสดงผล และเมื่อกำหนดแล้ว จะต้องไม่ใช้คำสั่งแสดงเคอร์เซอร์ อีก

4. คำสั่งแสดง หรือไม่แสดงหน้าจอ (DISPLAY ON/OFF)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	0	1	D	C	B

BIT D : เป็น BIT ให้เปิดปิดหน้าจอภาพโดยถ้า

D = 1 จะให้แสดงผลหน้าจอ (On Display)

D = 0 กำหนดให้ไม่แสดงผลหน้าจอ (Off Display)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้ง

039076

BIT C : ทำหน้าที่ควบคุมเคอร์เซอร์

BIT C = 1 จะให้แสดงเคอร์เซอร์ โดยเคอร์เซอร์ จะเป็นเส้นขีดใต้ตัวอักษร

BIT C = 0 หากไม่ต้องการแสดงเคอร์เซอร์ โดยตัว CURSOR จะอยู่ที่ ไลน์ที่ 8 ในแบบ 5*7 DOT และจะอยู่ ไลน์ที่ 11 ในแบบ 5*10 DOT

BIT B : เป็น บิตเซท การกระพริบของเคอร์เซอร์ โดย

B = 1 กำหนดให้มีการกระพริบที่ตำแหน่งเคอร์เซอร์ (กระพริบเป็นรูปสี่เหลี่ยม)

B = 0 กำหนดให้ไม่มีการกระพริบที่ตำแหน่งเคอร์เซอร์ โดยมีระยะเวลาการกระพริบประมาณ 379.2 ms

5. คำสั่งการเลื่อนตัวอักษร (Display shift)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	0	1	S/C	R/L	*	*

เป็นคำสั่งกำหนดให้ตำแหน่ง เคอร์เซอร์ หรือข้อมูลไปเกิดทางซ้ายหรือขวาโดยไม่ต้องใช้คำสั่งเขียนหรืออ่าน โดย

S/C	R/L	Description
0	0	ทำการย้าย เคอร์เซอร์ ไปจากตำแหน่งเดิมไปซ้ายมือ 1 ตำแหน่ง
0	1	ทำการย้าย เคอร์เซอร์ ไปจากตำแหน่งเดิมไปขวามือ 1 ตำแหน่ง
1	0	เป็นการดันตัวอักษรที่เกิดไปทางซ้าย
1	1	เป็นการดันตัวอักษรที่เกิดไปทางขวา

6. คำสั่งกำหนดหน้าที่การทำงาน (FUNCTION SET)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	0	1	DL	N	F	*	*

* หมายถึง ไม่มีผลกระทบ (No Effect)

- BIT DL : เป็นการเซ็ท การติดต่อว่าจะให้เป็นแบบ 8 BIT หรือ 4 BIT โดย

ถ้าต้องการติดต่อ 4 BIT DL = 0

ถ้าต้องการติดต่อ 8 BIT DL = 1

- N : เป็นการเซ็ทบรรทัดการแสดงผล

N = 0 แสดง 1 บรรทัด

N = 1 แสดง 2 บรรทัด

ในกรณีมากกว่า 2 บรรทัดก็ให้ SET N = 1.

- F : เป็นการเซ็ทขนาด DOT การแสดงผล โดย

F = 0 เป็นแบบ 5*7

F = 1 เป็นแบบ 5*10

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

N F	NO. of Display line	Character Font	Duty Facter	Remarks
0 0	1	5*7 Dots	1/8	
0 1	1	5*10 Dots	1/11	
1 *	2	5*7 Dots	1/16	ไม่สามารถแสดง 2 ไลน์ ด้วยอักษรขนาด 5*10 ดอท

* หมายถึง ไม่มีผลกระทบ

7. คำสั่งกำหนดตำแหน่งของหน่วยความจำอ้างอิงตัวอักษร (Set CGRAM Address)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	0	1	A	A	A	A	A	A

← บิทลำดับสูง บิทลำดับต่ำ →

ใน HD44780 จะมีหน่วยความจำอยู่ 2 ชุด คือ DISPLAY DATA RAM (DD RAM) จำนวน 80*8 บิท และ CHARACTER GENERATOR ROM CG RAM จำนวน 512 บิท และ 7200 บิท คำสั่งนี้จะเป็นการเซ็ทแอดเดรส ใน CG RAM โดยต้องทำการเซ็ทแอดเดรส ก่อนเขียนหรืออ่านข้อมูลจาก CG RAM ด้วย

8. คำสั่งกำหนดตำแหน่งของหน่วยความจำแสดงผลข้อมูล (Set DDRAM Address)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	1	A	A	A	A	A	A	A

←บิทลำดับสูง บิทลำดับต่ำ→

เป็นคำสั่ง SET ค่าแอดเดรสใน DD RAM ในการเขียนหรืออ่านค่าจาก DD RAM (DD RAM คือ ส่วนที่จะแสดงผลหน้าจอ LCD) โดยจำนวนแอดเดรสที่จะเกิดขึ้นบนจอ LCD จะอยู่กับการเซ็ท ค่า N ด้วย

N = 0 (1 บรรทัด) ADDRESS จะอยู่ที่ 00H - 4FH

N = 1 (2 บรรทัด) ADDRESS จะอยู่ที่ 00H - 27H สำหรับบรรทัดที่ 1 และ 40H - 67H สำหรับบรรทัดที่ 2

แบบการจัด ADDRESS ของ DD RAM หน้าจอ LCD

- แบบ 16 ตัวอักษร 1 บรรทัด
- แบบ 16 ตัวอักษร 2 บรรทัด
- แบบ 16 ตัวอักษร 4 บรรทัด
- แบบ 20 ตัวอักษร 1 บรรทัด
- แบบ 20 ตัวอักษร 2 บรรทัด
- แบบ 40 ตัวอักษร 2 บรรทัด

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

16 ตัวอักษร 1 บรรทัด

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F

16 ตัวอักษร 2 บรรทัด

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F

16 ตัวอักษร 4 บรรทัด

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
50	51	52	53	54	55	56	57	58	59	5A	5B	5C	5D	5E	5F

20 ตัวอักษร 1 บรรทัด

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13

20 ตัวอักษร 2 บรรทัด

00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11	12	13
40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	51	52	53

9. บิทแสดงสถานะพร้อม/ไม่พร้อม และอ่านค่าตำแหน่ง (Busy Flag and Address)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
0	0	BF	A	A	A	A	A	A	A

←บิทลำดับสูง บิทลำดับต่ำ→

เป็นคำสั่งอ่านค่า BUSY FLAG ซึ่งจะเป็นตัวบอกว่าตัว HD44780 นี้อยู่ในขบวนการทำงานภายในอยู่หรืออยู่ในสภาพพร้อมจะรับข้อมูล โดย

BF = 1 อยู่ในขบวนการทำงานภายในไม่พร้อมจะรับข้อมูลหรือคำสั่ง

BF = 0 พร้อมจะรับข้อมูลหรือคำสั่งได้

และนอกจากนี้ยังเป็นคำสั่งอ่านค่าข้อมูลแอดเดรสของ CG RAM หรือ DD RAM ด้วย

10. การเขียนข้อมูลลงในหน่วยความจำแสดงผลข้อมูล หรือหน่วยความจำอ้างอิงตัวอักษร (WRITE DATA TO DD RAM OR CG RAM)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D	D	D	D	D	D	D	D

←บิทลำดับสูง บิทลำดับต่ำ→

สำหรับการเขียนข้อมูลลงหน่วยความจำแสดงผลข้อมูล (DD RAM) หรือหน่วยความจำอ้างอิงตัวอักษร (CG RAM) โดยเมื่อทำการเขียนแล้ว แอดเดรสจะถูกเพิ่มหรือลดลงโดยอัตโนมัติ ตามที่กำหนดจากค่า I/D ในคำสั่งกำหนดรูปแบบการใส่ข้อมูล (ENTRY MODE SET) และการเขียนจะเป็นหน่วยความจำแสดงผลข้อมูล หรือหน่วยความจำอ้างอิงข้อมูลก็ขึ้นกับว่า ก่อนหน้าคำสั่งนี้ มีการกำหนด แอสเดรสที่ใดทำได้โดยการเซ็ทแอดเดรส ของ CG RAM หรือ DD RAM ขึ้นมาก่อนจะเขียนข้อมูล

11. การอ่านข้อมูลจากหน่วยความจำแสดงผลข้อมูล หรือหน่วยความจำอ้างอิงตัวอักษร(READ DATA FROM DD RAM OR CG RAM)

RS	R/W	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
1	0	D	D	D	D	D	D	D	D

←บิทลำดับสูง บิทลำดับต่ำ→

เป็นคำสั่งอ่านข้อมูลจาก CG RAM หรือ DD RAM โดยก่อนอ่านค่า CG RAM หรือ DD RAM นี้ควรใช้คำสั่งเซ็ทแอดเดรส ก่อนเพื่อให้รู้ว่าข้อมูลที่อ่านได้นั้นเป็น CG หรือ DD RAM จากตารางการทำงานจะเห็นว่าการใช้งาน LCD MODULE ง่าย เพียงแต่ส่งคำสั่งเริ่มแรกและเซตความต้องการขนาดตัวอักษร,CURSOR หลังจากนั้นก็สามารถเขียนตัวอักษรเข้าไปใน DD RAM ตามตารางตัวอักษรที่ให้มาก็จะเกิดตัวอักษรในจอภาพ LCD และยังสามารถกำหนดตำแหน่งตัวอักษรที่จะเกิดบนจอภาพได้โดยการเซ็ท DD RAM ADDRESS ตามตารางที่ให้มา

สามารถเขียนข้อมูลโดยกำหนดแอดเดรสของ DD RAM ก่อนโดยเขียนได้ 64 ตำแหน่ง บิต0-บิต5 และเมื่อกำหนดแอดเดรส แล้วก็ทำการเขียนข้อมูลลงใน CG RAM โดยเป็นลักษณะบิตต่อบิต บนจอ1ตัวอักษรคือ 5x7 DOTนั้น จะใช้ข้อมูลบิต 0 ถึง บิต 4 ต่อ 1ไบต์เท่านั้น 1ตัวอักษรใช้ข้อมูล 8ไบต์ และเมื่อเขียนข้อมูลลงใน CG RAM แล้วต้องการใช้งานแสดงผลก็ให้เขียนข้อมูลลงใน DD RAM ที่ตำแหน่ง 00H-07H

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.6 ตัวสร้างสัญญาณนาฬิกาหรือ RTC (Real Time Clock)

โครงงานนี้จะใช้ไอซีเบอร์ M48T08 ซึ่งเป็นไอซีประเภทนอนโวลาไทล์แรม หรือ NV RAM (Nonvolatile RAM) ซึ่งเป็นแรมขนาด 8 kbyte และมีรีจิสเตอร์ที่ทำหน้าที่ในการเก็บค่าเวลาต่าง ๆ เช่น วัน เดือน ปี ชั่วโมง นาที และวินาที เราสามารถทำการตั้งเวลาใหม่ได้ และโหลดค่ามาทำการคำนวณได้

2.6.1 การอ่านสัญญาณนาฬิกา

ควรทำการหยุดการอัพเดท (Update) ค่าในรีจิสเตอร์เก็บค่าเวลา (Timekeeper) ก่อนที่จะเกิดการอ่านค่าของข้อมูลเพื่อป้องกันการอ่านค่าขณะที่เกิดการเปลี่ยนสถานะของเวลาเพราะว่าพอร์ตคู่ของรีจิสเตอร์เก็บค่าเวลา (Timekeeper) ในหน่วยความจำแบบอาร์เรย์ (Array) เป็นรีจิสเตอร์ข้อมูลไม่ได้เป็นตัวนับเวลาอย่างแท้จริง การทำให้รีจิสเตอร์เก็บค่าปัจจุบันสามารถทำให้หยุดได้โดยปราศจากการรบกวนสัญญาณนาฬิกา (สัญญาณนาฬิกาควบคุมจังหวะการทำงาน)

การอัพเดท (Update) ค่าเวลาจะหยุดการทำงานเมื่อเซตบิตที่7 (D6) ในรีจิสเตอร์ควบคุมให้มีค่าเป็น 1 เมื่อใดที่ค่าของบิตนี้ยังคงเป็น 1 การอัพเดท (Update) ค่าเวลาจะหยุดหลังจากที่ถูกสั่งให้หยุดแล้ว รีจิสเตอร์ก็จะคืนค่าของเวลา วันที่ วัน ปัจจุบัน รีจิสเตอร์เก็บค่าเวลาทุกตัวจะเปลี่ยนค่าพร้อมกันการหยุดเพื่ออ่านค่าของเวลาในขณะนั้นไม่ได้เป็นการรบกวนกระบวนการของการอัพเดทค่าเวลาการคืนค่าเวลาปัจจุบันจะเกิดขึ้นในวินาทีนั้นหลังจากที่บิต D6 มีค่าเป็น 0

2.6.2 การเซตสัญญาณนาฬิกา

บิตที่ 8 ในรีจิสเตอร์ควบคุม (D7) คือบิตที่ใช้เพื่อการเขียนค่าข้อมูลเมื่อบิตนี้มีค่าเป็น1 การอัพเดทค่าเวลาจะหยุดการทำงานเหมือนการอ่านค่านาฬิกาในรีจิสเตอร์เก็บค่าเวลาจากนั้นผู้ใช้สามารถเขียนค่าวัน วันที่ เวลาภายใน 24 ชั่วโมงที่ถูกต้องแบบตัวเลขฐานสิบหลังจากนั้นเมื่อค่าของบิตนี้ถูกเซตให้เป็น 0 ค่าที่เขียนลงไปจะถูกส่งไปที่รีจิสเตอร์เก็บค่าเวลาและทำการแสดงค่าเวลาดังเดิม

2.6.3 การเริ่มและหยุดการทำงานของออสซิลเลเตอร์

ออสซิลเลเตอร์จะหยุดการทำงานทันทีที่เวลาใดๆเมื่อกระแสไฟฟ้าจากแบตเตอรี่มีค่าน้อยกว่าค่าต่ำสุดที่ออสซิลเลเตอร์สามารถทำงานได้หรือทำการเซตค่าของบิตที่แปดหรือที่เรียกว่าสต้อปบิต(stop bit) ในรีจิสเตอร์เก็บค่าวินาทีให้มีค่าของบิตเป็น 1

2.6.4 การปรับค่า (Calibrating) ของนาฬิกา

ไอซี M48T08/18 ถูกควบคุมการออสซิลเลเตอร์ด้วยผลึก ควอร์ซ (Quartz) ผลิตความถี่ 32,768 Hz ตัวผลึกถูกติดตั้งไว้ที่ส่วนบนของแบตเตอรี่ให้ค่าความแน่นอน เท่ากับ ±1นาที ต่อเวลา1เดือน ภายใต้อุณหภูมิ 25 องศาเซลเซียสโดยไม่ต้องทำการปรับค่าเวลาอีก จากการทดสอบพบว่าจะเกิดการผิดพลาดของค่าเวลาไม่เกิน 35 ส่วนต่อ 1 ล้านส่วน หรือคิดเป็น ค่าความแน่นอนเท่ากับ ± 1.53นาที ต่อเวลา 1 เดือน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.7 ไอซีMM74C922 ตัวเข้ารหัสการกดคีย์ (16-Key Encoder)

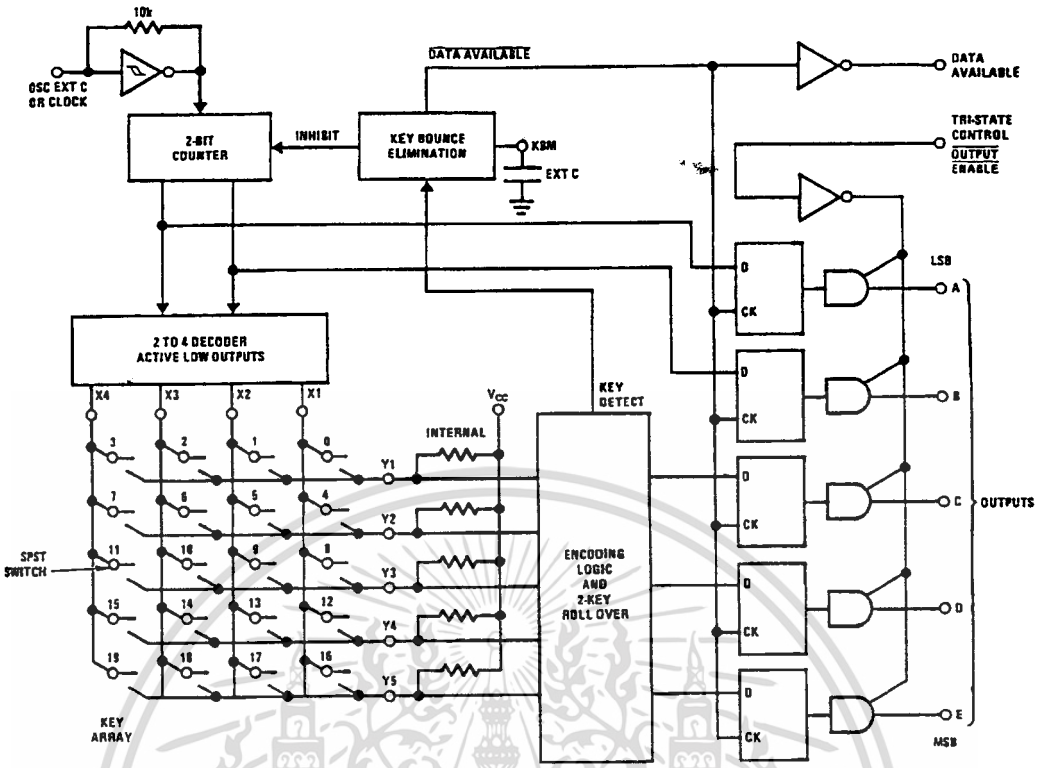
MM74C922 เป็นไอซีที่ใช้สำหรับการเข้ารหัสของคีย์บอร์ดซึ่งจะทำการอินเตอร์เฟสปุ่มกดแบบ 16 SPST (กดติดปล่อยดับ)กับสัญญาณดิจิตอลโดยที่เมื่อมีการเข้ารหัสสัญญาณจากปุ่มขณะสวิทช์ปิดจะถูกแปลงเป็นรหัส 4 บิตหรือ5 บิตผู้ออกแบบวงจรสามารถควบคุมได้ทั้งอัตราการสแกนคีย์บอร์ด และ ช่วงเวลาของการกดติดปล่อยดับของปุ่มบนคีย์บอร์ดโดยการเปลี่ยนค่าของออสซิลเลเตอร์คาปาซิเตอร์(OSCILLATOR CAPACITOR,C_{OSE}) และ เคบีเอ็ม คาปาซิเตอร์ (KEY BOUNCE CAPACITOR,C_{KBM})

ตัวเข้ารหัสของคีย์บอร์ดจะเชื่อมต่อกับปุ่มของคีย์บอร์ดแบบเมตริกซ์ขนาด 4แถวนอน4แถวตั้ง (สำหรับไอซีMM74C922) หรือขนาด 5แถวนอน4แถวตั้ง (สำหรับ ไอซี MM74C923)เมื่อไม่มีปุ่มใดของคีย์บอร์ดถูกกด สัญญาณอินพุทในแถวนอนจะถูกทำให้มีค่าสูงโดยวงจรพูลอัพ(PULL UP)ภายในและเอาท์พุทของแถวหลักจะมีค่าลอจิกเป็นศูนย์อัตราการสแกนในแถวตั้งถูกควบคุมโดยออสซิลเลเตอร์ภายในซึ่งประกอบไปด้วยวงจรของชมิตทริกเกอร์ออสซิลเลเตอร์ (Schmitt trigger Oscillator) วงจรนับสองบิต วงจรถอดรหัส 2-4 บิต

เมื่อคีย์บอร์ดถูกกดยกตัวอย่างเช่นการกด ปุ่ม 0 ยังไม่มีสิ่งใดเกิดขึ้นขณะที่ยังไม่มีอินพุทที่ขาสัญญาณแถวหลัก X1 เพราะว่าขาสัญญาณแถวนอน Y1 ยังคงมีค่าลอจิกสูงเมื่อแถวหลักของ X1 ถูกสแกนขาสัญญาณแถวหลัก X1จะเปลี่ยนเป็นลอจิกต่ำและขาสัญญาณแถวนอน Y1 จะเปลี่ยนค่าลอจิกเป็นค่าต่ำด้วยเหตุการณ์นี้ทำให้วงจรนับไม่สามารถทำงานและทำการเก็บค่าลอจิกต่ำของขาสัญญาณแถวนอนX1การที่ขาสัญญาณแถวหลัก Y1จะเปลี่ยนค่าลอจิกเป็นค่าต่ำทำให้เป็นการเริ่มต้นการทำงานของวงจรเวลา (Key bounce timing) และกันสัญญาณอินพุทของขาสัญญาณ แถวนอนYในแถวนอนอื่น ดังนั้นรหัสเอาท์พุทของการกดปุ่มคีย์บอร์ดประกอบไปด้วยค่าจากวงจรนับและค่าจากการถอดรหัสจากขาสัญญาณอินพุทแถวนอน เมื่อหมดช่วงเวลาการทำงานของวงจร Key Bounce ค่าของข้อมูลจะถูกคงค่าไว้(Latched) และค่าเอาท์พุทของสัญญาณ Data Available (DAV) มีสถานะสูง

หลังจากช่วงเวลาการกดปุ่ม อินพุทของขาสัญญาณแถวนอน Y1 จะมีค่าลอจิกสูงอีกครั้งวงจรภายในไอซีจะเริ่มทำการสแกนใหม่และรีเซตวงจร Key Bounce แม้จะมีการกดปุ่มหลายครั้งแต่ในทันทีที่กดปุ่มการสวิทช์ยังคงรักษาสถานะต่ำสำหรับช่วงเวลาของการกดนั้นอยู่และข้อมูลยังคงค่าไว้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 2.8 แสดง 16 KEY ENCODERS

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 3

การคำนวณและการสร้าง

เครื่องชุมสายโทรศัพท์ สาขาปลายทางอัตโนมัติที่นำเสนอในภาคการศึกษานี้ประกอบด้วย ส่วนสำคัญดังต่อไปนี้

1. โครงสร้างทางด้านฮาร์ดแวร์ (HARDWARE) ของตัวเครื่องชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ สามารถแบ่งได้เป็นส่วนต่าง ๆ ดังต่อไปนี้

- ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก (TRUNK -LINE INTERFACE CIRCUIT SECTION)
- ส่วนครอสพอยท์สวิทช์ (CROSS POINT SWITCH CIRCUIT SECTION)
- ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF (TONE DETECTOR AND DTMF DECODER CIRCUIT SECTION)
- ส่วนกำเนิดสัญญาณเสียง (TONE GENERATOR CIRCUIT SECTION)
- ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน (SUBSCRIBER LINE INTERFACE CIRCUIT SECTION : SLIC)
- ส่วนตอบรับโทรศัพท์อัตโนมัติ (AUTOMATIC ANSWER CIRCUIT SECTION)
- ส่วนจอแสดงผล, ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล (DOT MATRIX LCD MODULE , KEYBOARD ENCODER AND CONTROL CIRCUIT SECTION #2)
- ส่วนอินพุท/เอาท์พุทพอร์ท (I/O PORT SECTION)
- ส่วนแหล่งจ่ายไฟเลี้ยง (POWER SUPPLY CIRCUIT SECTION)
- ส่วนควบคุมระบบ (CONTROL CIRCUIT SECTION #1)

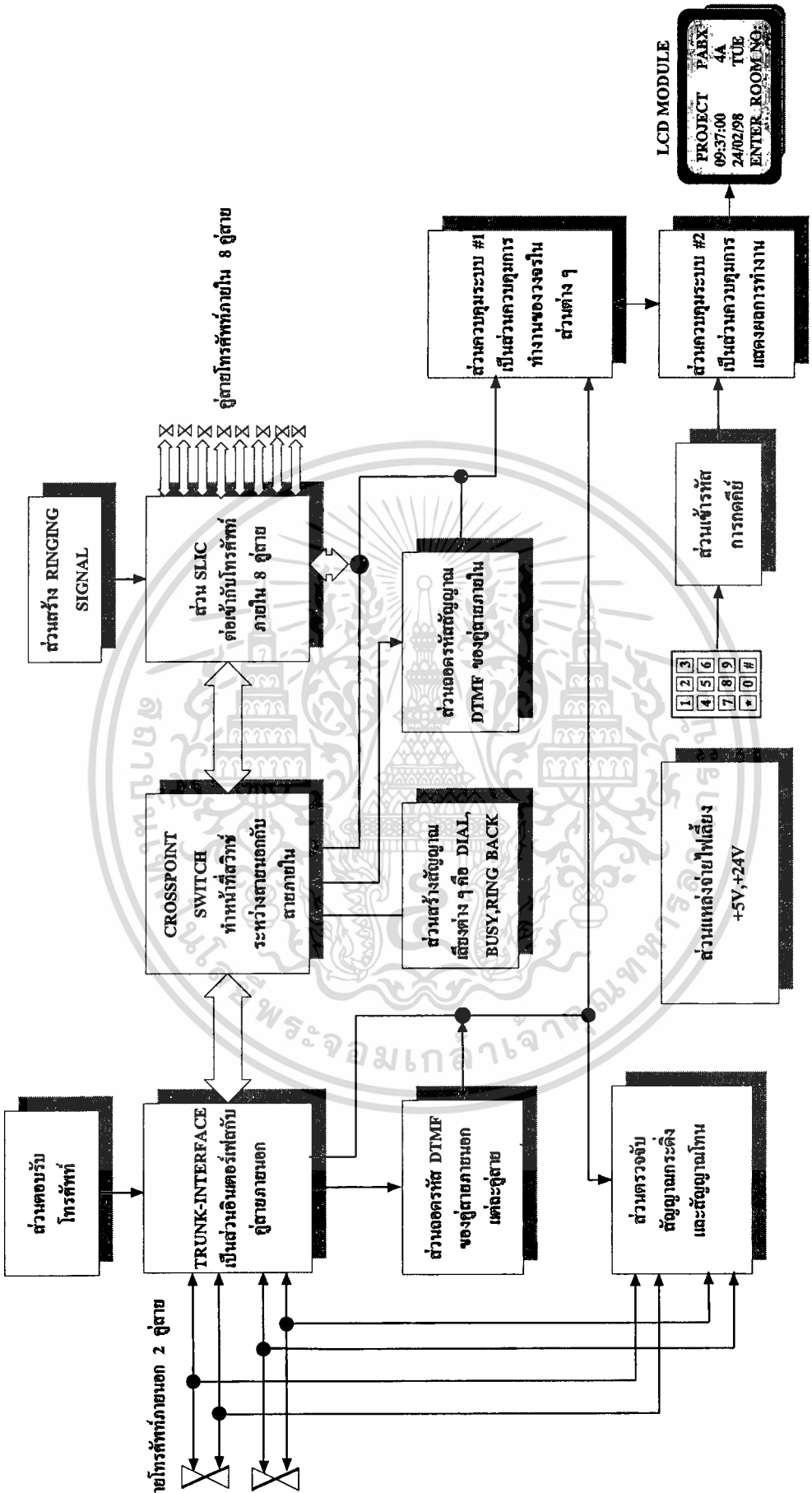
2. โครงสร้างทางซอฟแวร์ (SOFTWARE) เป็นส่วนที่เกี่ยวกับโปรแกรมคำสั่งที่ใช้ควบคุมการทำงานของ เครื่องชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ ให้ทำงานตามคุณสมบัติที่กำหนดไว้

จากบล็อคไดอะแกรมรูปที่ 3.1 แสดงภาคต่างๆ ซึ่งจะทำงานที่สัมพันธ์กัน สามารถอธิบายการทำงานรวมกันได้ดังนี้ การทำงานของวงจรทั้งหมดจะแบ่งออกเป็น 2 โหมด คือโหมดการติดต่อภายใน กับโหมดการติดต่อภายนอก เมื่อมีผู้เรียกมาจากภายนอกส่วนอินเตอร์เฟสคับคู่สายภายนอกซึ่งประกอบไปด้วยวงจรตรวจจับเสียงกระดิ่ง เมื่อจับได้แล้วก็จะส่งสัญญาณไปยังส่วนควบคุมระบบ เพื่อให้ส่วนควบคุมระบบส่งสัญญาณมาควบคุมส่วนอินเตอร์เฟสกับคู่สายภายนอก ทำการปรับอิมพีแดนซ์ ของตัวเอง ซึ่งในตอนแรกมีค่าความต้านทานมากให้เหลือค่าความต้านทาน 600 โอห์ม คือเป็นการรับสายนั่นเอง แล้วก็จะส่งสัญญาณเสียงจากส่วนตอบรับอัตโนมัติ ออกไปเพื่อให้ผู้ที่เรียกเข้ามาทำการเลือกหมายเลขของคู่สายภายในได้ จากนั้นเมื่อมีผู้เรียกกดหมายเลขของคู่สายแล้วสัญญาณ DTMF ที่จะถูกส่งมายังส่วนถอดรหัสสัญญาณ DTMF และถอดรหัสไปเป็นสัญญาณดิจิตอล เพื่อส่งไปยังส่วนควบคุม เมื่อส่วนควบคุมรับรหัสที่ได้มาแล้วก็จะตรวจดูว่าคู่สายที่ถูกเรียกเข้ามาว่างหรือไม่ ถ้าว่างก็จะไปส่งให้ส่วนครอสพอยท์ทำการตัดต่อสัญญาณเรียกกลับ (RING BACK TONE) ไปยังผู้ที่

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

เรียกเข้ามา พร้อมกับส่งสัญญาณกระดิ่ง (RINGING SIGNAL)ไปยังคู่สายภายในที่ถูกเรียก แต่ถ้าคู่สายภายในที่ถูกเรียกไม่ว่าง ส่วนควบคุม ก็จะส่งให้ส่วนครอสพอยท์ ต่อสัญญาณไม่ว่าง (BUSY TONE)ไปยังผู้เรียกเข้ามา ถ้าหากคู่สายที่ถูกเรียกว่างพร้อมที่จะใช้งานได้เมื่อได้ยินสัญญาณกระดิ่ง ถ้ามีการยกหูส่วนตรวจจับการยกหูซึ่งเป็นภาคย่อยของส่วนอินเตอร์เฟสกลับคู่สายภายใน ก็จะสามารถรับรู้ว่ามีการยกหูและส่งสัญญาณมายังส่วนควบคุม และส่วนควบคุม ก็จะสั่งให้ส่วนครอสพอยท์ทำการเชื่อมต่อคู่สายที่เรียกเข้ามากับคู่สายที่ถูกเรียก และเมื่อฝ่ายหนึ่งฝ่ายใดวางหูก็จะหมดช่วงของการสนทนา ส่วนโหมดการติดต่อภายในนั้นเมื่อมีคู่สายใดคู่สายหนึ่งของคู่สายภายในมีการยกหูขึ้นมาทางภาคการตรวจจับการยกหู ซึ่งเป็นภาคย่อยของส่วนอินเตอร์เฟสกับคู่สายภายในจะตรวจจับได้แล้วจะส่งสัญญาณมายังส่วนควบคุม หลังจากนั้นส่วนควบคุม ก็พร้อมที่รับสัญญาณที่เข้ามาอีกจากการกดหมายเลขของคู่สายภายในที่ทำการยกหู ซึ่งเมื่อผู้ยกหูกดหมายเลขสัญญาณหมายเลขที่ถูกกดก็มาผ่านส่วนถอดรหัสสัญญาณ DTMF เป็นสัญญาณดิจิตอลและส่งมายังส่วนควบคุมต่อไป เมื่อส่วนควบคุมได้รับสัญญาณจากหมายเลขที่กดได้แล้วก็จะทำการวิเคราะห์คู่สายภายในที่ถูกเรียก ถ้าหากไม่ว่าง ก็จะสั่งให้ส่วนครอสพอยท์ต่อสัญญาณไม่ว่าง ไปยังผู้เรียกเข้า ถ้าหากว่า ผู้ถูกเรียกว่างทางผู้ถูกเรียกจะได้ยินสัญญาณกระดิ่ง และต่อสัญญาณเรียกกลับ ไปยังผู้เรียก เมื่อผู้ถูกเรียกยกหูขึ้น

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.1 บล็อคไดอะแกรมของเครื่องชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก (TRUNK - LINE INTERFACE CIRCUIT SECTION)

หลักการทำงานของวงจร

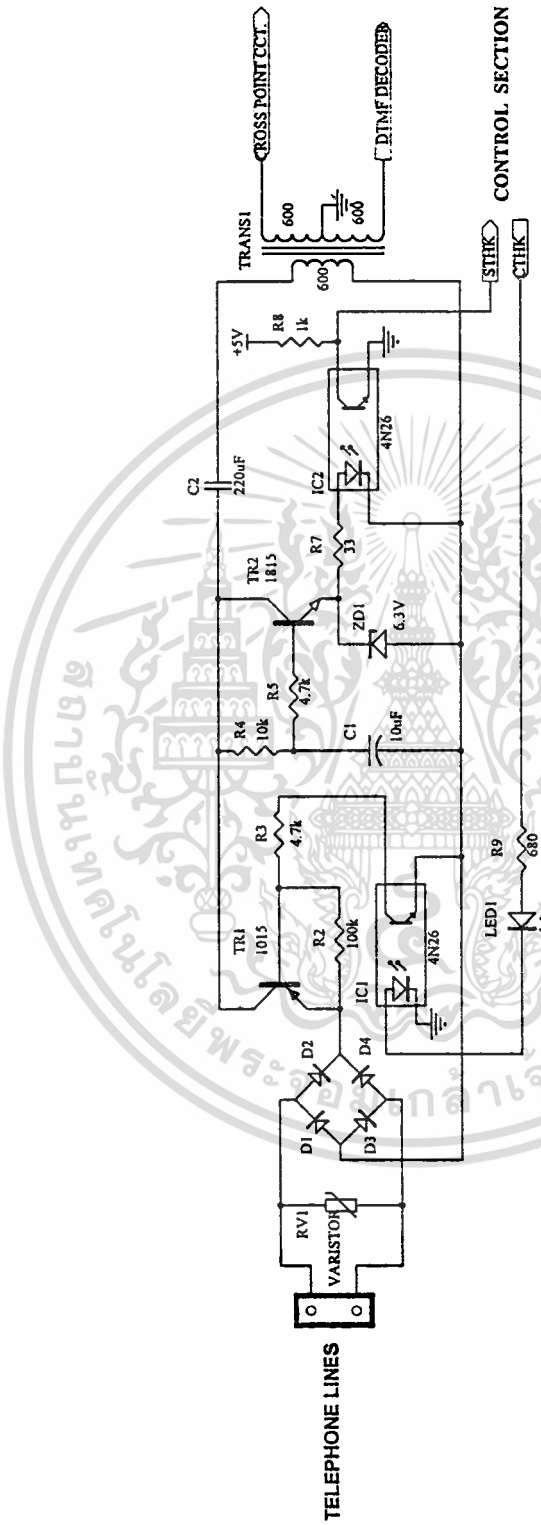
วงจรนี้จะต่อโดยตรงกับคู่สายโทรศัพท์ภายนอก 2 คู่สาย ส่วนนี้ทำหน้าที่เป็นการเชื่อมต่อระบบการติดต่อกับคู่สายขององค์การโทรศัพท์ คือการปรับสภาพของวงจรให้มีสภาพเหมือนกับการยกหูและการวางหูนั่นเอง แต่จะเป็นการยกหูที่ไม่ใช้คนยกหูแต่จะใช้หลักการทางโทรศัพท์ คือในสภาวะสายว่างโทรศัพท์โทรศัพท์จะมีแรงดัน 48 โวลต์ ซึ่งจ่ายมาจากชุมสายโทรศัพท์ และเมื่อมีผู้เรียกเข้ามา ทางชุมสายจะจ่ายสัญญาณกระดิ่ง มาเป็นไฟฟ้ากระแสสลับที่มีแรงดันประมาณ 100 Vp-p เป็นเวลา 1 วินาที และหยุด 4 วินาที เป็นจังหวะอย่างนี้ ซึ่งแรงดันจะทำให้วงจรภายในโทรศัพท์ทำงานและชุมสายจะรับทราบการยกหู และจะทำการต่อคู่สายเข้ากับวงจรภายในที่มีความต้านทานกระแสตรงต่ำ จะเกิดการครบวงจรขึ้นทำให้แรงดัน 48 โวลต์ ลดลงเหลือ 6 - 10 โวลต์

จากรูปวงจร สัญญาณกระดิ่ง จากชุมสายจะส่งมาตามคู่สายผ่านวาริสเตอร์ ซึ่งทำหน้าที่ป้องกันแรงดันไฟเกิน (OVERLOAD PROTECTION) ผ่านบริดจ์ไดโอด และเข้าขา อิมิทเตอร์(EMITTER) ของทรานซิสเตอร์ 1015 ถ้ามองเข้าไปข้างในวงจรจะมีค่าอิมพีแดนซ์มากเปรียบได้กับการวางหู ขณะนี้หากส่วนควบคุมระบบ นับจำนวนสัญญาณกระดิ่งครบแล้วจะสั่งให้มีการยกหู โดยส่งลอจิก "1" ผ่านตัวต้านทาน 680 Ω และ LED ทำให้ LED สว่าง แล้วสัญญาณจะเข้าไปยัง ขา 1 ของ 4N26 ซึ่งเป็น ออปโตทรานซิสเตอร์ (OPTO-TRANSISTER) ทำให้ออปโตทรานซิสเตอร์ ทำงาน ทำให้ ขา 5 และ ขา 4 นำกระแสซึ่งเป็นการไบแอสให้กับ 1015 ทำให้ 1015 นำกระแสด้วย ก็เลยทำให้ค่าอิมพีแดนซ์ ที่เคยสูงอยู่ก่อนจะลดลงมา เนื่องจากขา อิมิทเตอร์ และ คอลเลคเตอร์ของ 1015 นำกระแสทำให้อิมพีแดนซ์ของวงจรขณะนั้นเป็นอิมพีแดนซ์ของหม้อแปลง (TRANFORMER) คือ 600 Ω จึงเปรียบเสมือนการยกหูโทรศัพท์เกิดขึ้น จากนั้นสัญญาณจากชุมสายก็จะผ่านตัวต้านทาน และไปไบแอสทรานซิสเตอร์1815 ให้สวิทช์ออน ให้สัญญาณทางขาเบส (base) ออกมาทางอิมิทเตอร์เข้าไปกระตุ้นให้ 4N26 ทำงานที่ขา 1 แล้วขา 5 ของ 4N26 จะต่อกับไฟ +5V และตัวต้านทาน 1 k จะนำกระแสทำให้ระดับสัญญาณที่ขา 5 เป็นโลจิก "0" สัญญาณนี้ถูกส่งไปยัง ส่วนควบคุมระบบเพื่อแจ้งให้ทราบว่ามีการยกหูโทรศัพท์เกิดขึ้น

ปกติหาก 4N26 ถ้าไม่มีการยกหูโทรศัพท์หรืออีกนัยหนึ่งไม่มีสัญญาณเข้ามากระตุ้นการทำงานของ 4N26 ขา 5 จะมีระดับสัญญาณเป็นโลจิก "1"

สัญญาณจะผ่านไปยังหม้อแปลงอิมพีแดนซ์ หม้อแปลงจะเหนี่ยวนำให้สัญญาณเกิดยังด้านทุติยภูมิ (SECONDARY) แล้วส่งไปส่วน ครอสพอยท์สวิทช์ ส่วนสัญญาณอีกชุด จะส่งไปยังส่วนถอดรหัสสัญญาณ DTMF และส่วนตรวจจับสัญญาณเสียง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



TRUNK LINE INTERFACE CIRCUIT

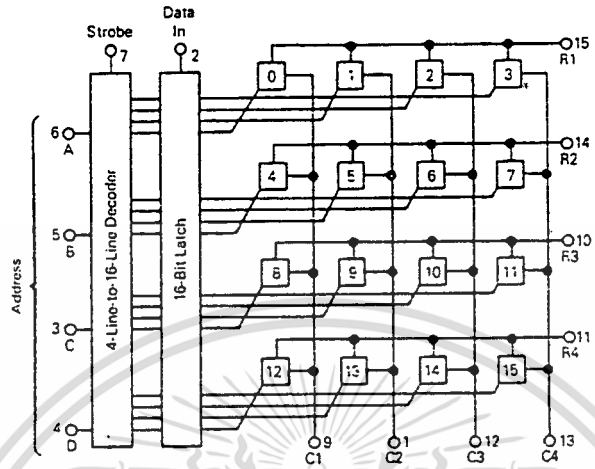
รูปที่ 3.2 แสดงวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.2 ส่วนครอสพอยท์สวิทช์ (CROSSPOINT SWITCH SECTION)

หลักการทำงาน

ภาคครอสพอยท์สวิตช์เป็นภาคที่ทำการเชื่อมต่อสัญญาณจากภาคต่าง ๆ เข้าไปยังคู่สายที่ต้องการจะประกอบด้วย ไอซี MC142100 ซึ่งทำหน้าที่เป็นอนาล็อกสวิตช์จะต่อกันเป็นลักษณะของเมตริกซ์ ดังรูป



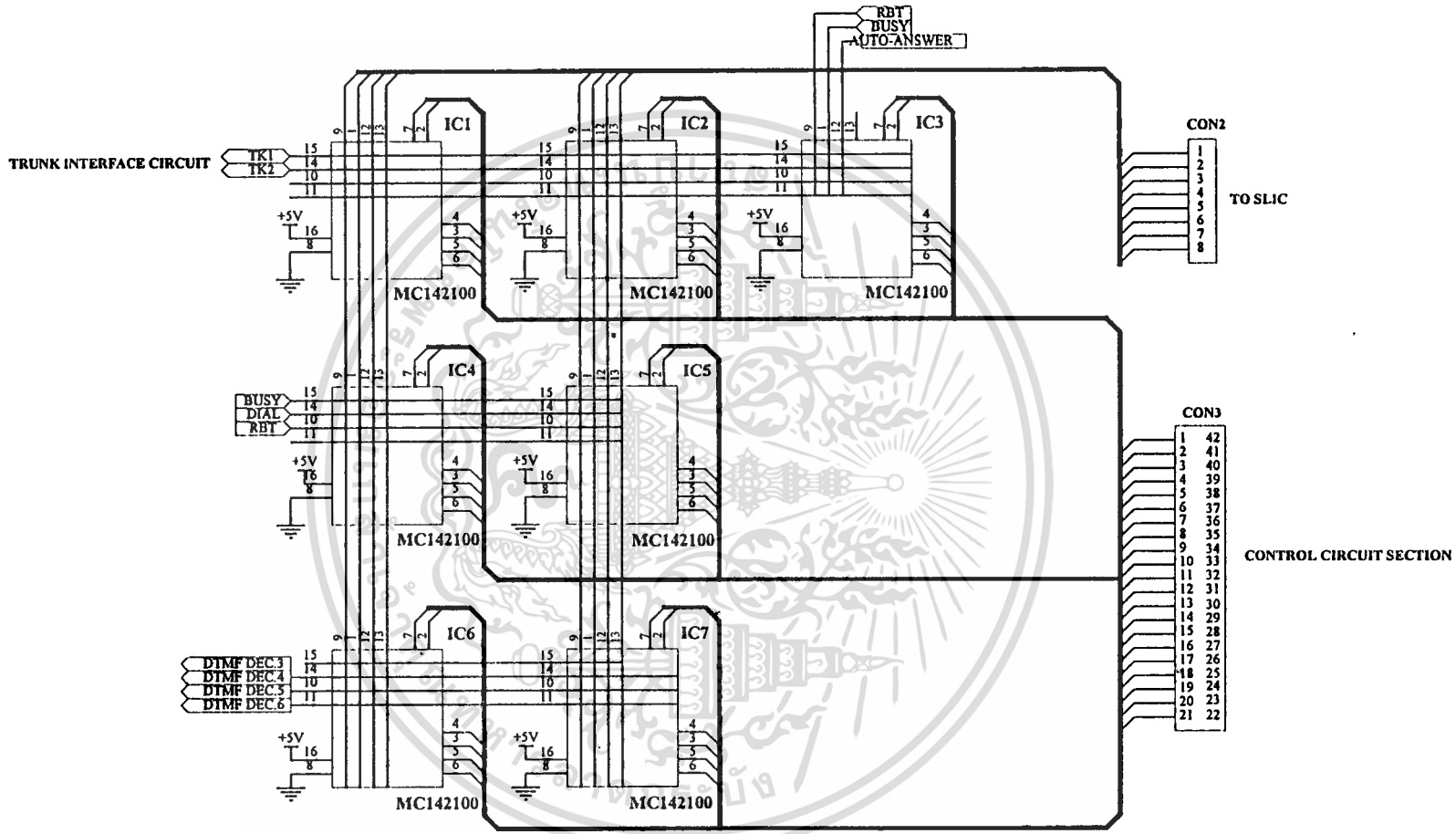
รูปที่ 3.3 แสดงการต่อกันของไอซี MC142100

จากรูปจะมีทางด้านอินพุทและเอาท์พุทของสวิทช์ ซึ่งสามารถที่จะสลับที่กันได้จะให้ C1-C4 เป็นอินพุทและ R1 - R4 เป็นเอาท์พุทก็ได้ ซึ่งทุก ๆ จุดจะสามารถต่อถึงกันได้โดยอาศัยการควบคุมจากขาแอดเดรส ของ IC ว่าจะให้จุดไหนต่อถึงกัน

เมื่อทำการเลือกแอดเดรส ได้แล้วก็ให้เปลี่ยนสภาวะของสโตรบ(STOBE) จาก "0" เป็น "1" และตามด้วยขาคาต้าอิน (DATA IN)จาก "0" เป็น "1" ก็จะทำให้แอดเดรส ที่เลือกไว้ก็จะต่อกัน เมื่อต้องการจะเลิกทำงานให้เลือกแอดเดรส แล้วเปลี่ยนขาสโตรบจาก "0" เป็น "1" โดยที่ขาดาต้าอินเป็นสอจิก "0" การเชื่อมต่อก็จะสิ้นสุดลง

ในการใช้งานจริง เนื่องจากว่า IC MC 142100 ตัวเดียวมีขนาดเล็กเกินไปจึงได้นำมาต่อรวมกัน 7 ตัว คือ ดังรูปที่ 3.4 ซึ่งจะทำให้สามารถเพิ่มจำนวนเอาท์พุทที่ต่อกับเครื่องโทรศัพท์ของคู่สายได้ครบ 8 เครื่องพอดี ส่วนทางด้านอินพุทที่เข้ามาที่คู่สายคือจะต่อให้เข้ากับขา 14 และ 15 ส่วนเอาท์พุทขา 9, 1, 12 และ 13 จะต่อกับคู่สายภายใน ส่วนการควบคุมการตัดต่อ จะให้คู่สายภายนอกต่อกับคู่สายภายใน 1-8 นั้นก็ขึ้นอยู่กับการเลือกแอดเดรส ซึ่งจะทำการเลือกโดยส่วนควบคุม ซึ่งจะสัมพันธ์กับการกดเลขหมาย เพื่อต่อกับคู่สายภายในของผู้เรียก เนื่องจากส่วนควบคุม จะทำการวิเคราะห์หมายเลขที่ได้จากถอดรหัสสัญญาณ DTMF หลังจากนั้นก็ทำการส่งสัญญาณเพื่อเลือกแอดเดรสในการต่อคู่สายเข้าด้วยกัน นอกจากการต่อในลักษณะนี้ ยังประกอบด้วยการต่อสัญญาณเสียงต่างๆ ผ่านส่วนนี้ ด้วยซึ่งจะมีวงจรใช้งานจริงทั้งหมดดังรูปที่ซึ่งจะใช้ IC MC 142100 ทั้งหมด 7 ตัว.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



CROSS POINT SWITCH CIRCUIT

รูปที่ 3.4 แสดงวงจรส่วนครอสพอยท์สวิทช์

3.3 ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF (TONE DETECTOR AND DTMF DECODER CIRCUIT SECTION)

3.3.1 วงจรตรวจจับสัญญาณเสียง

จากรูปที่ 3.6 เป็นวงจรที่ใช้สำหรับการตรวจจับสัญญาณเรียกกลับ และสัญญาณไม่ว่างที่มาจากคู่สายภายนอก ซึ่งจะใช้ LM 567 เป็นตรวจจับสัญญาณโทรศัพท์ ทั้งสัญญาณเรียกกลับ และสัญญาณไม่ว่าง ซึ่งมีความถี่ประมาณ 400 Hz ซึ่งหากสามารถตรวจสอบสัญญาณทั้งสองได้ LED จะสว่าง สัญญาณที่ตรวจจับได้จะถูกส่งไปยัง PIC 12C509 เพื่อทำหน้าที่ประมวลผลหรือตรวจสอบคาบเวลาของแต่ละสัญญาณที่ส่งเข้ามา ถ้าสัญญาณที่ได้รับตรงกับสัญญาณเรียกกลับหรือสัญญาณไม่ว่างก็จะส่งสัญญาณบอก CPU ว่าสัญญาณอินพุทขณะนี้เป็นสัญญาณใด โดยสัญญาณเรียกกลับจะดัง 1 วินาทีหยุด 4 วินาที ส่วนสัญญาณไม่ว่างจะดัง 0.5 วินาทีหยุด 0.5 วินาที

3.3.2 วงจรถอดรหัสสัญญาณ DTMF

วงจรถอดรหัสสัญญาณ DTMF จะประกอบด้วย 2 ส่วน

1. ส่วนที่หนึ่งทำหน้าที่ถอดรหัส DTMF จากการเรียกที่มาจากคู่สายภายนอก
2. ส่วนที่สองทำหน้าที่ถอดรหัส DTMF จากการติดต่อภายในด้วยกัน หรือทำหน้าที่ถอดรหัสสัญญาณ DTMF ขณะที่จะมีการติดต่อไปยังภายนอก

โดยวงจรทั้งสองทำหน้าที่เหมือนกัน แต่อยู่คนละตำแหน่งกัน

วงจรถอดรหัสสัญญาณ DTMF ส่วนที่ 1

เนื่องจากว่าสัญญาณที่ได้จากการกดหมายเลขแต่ละตัวบนหน้าปัทม์โทรศัพท์จะออกมาเป็นความถี่คลื่นรูปไซน์ 2 ความถี่ตามลักษณะของโทรศัพท์แบบกดปุ่มในการนำไปใช้งานเราจึงจำเป็นต้องเปลี่ยนสัญญาณความถี่คลื่นรูปไซน์ ให้เป็นสัญญาณดิจิตอลเสียก่อน ก็คือต้องผ่านสัญญาณความถี่คลื่นรูปไซน์เข้าไปยังวงจรถอดรหัสสัญญาณ DTMF ซึ่งเมื่อมีผู้เรียกเข้ามา ทางภาครับทำการรับคู่สายแล้วส่วนตอบรับอัตโนมัติจะส่งสัญญาณเสียงตอบรับผู้เรียกเข้ามาเพื่อให้ทำการกดหมายเลขของคู่สายภายใน เมื่อมีการกดหมายเลขของคู่สายภายในที่ได้ออกมาเป็นสัญญาณ DTMF ก็จะมาผ่านภาควงจรถอดรหัสสัญญาณ DTMF เพื่อที่แปลงสัญญาณคลื่นรูปไซด์ไปเป็นสัญญาณดิจิตอล 4 บิทเพื่อที่จะส่งให้ภาคควบคุมระบบ ทำการวิเคราะห์ต่อไป ในวงจรใช้งานจริงจะใช้ ไอซีเบอร์ MT8870 ทำหน้าที่ถอดรหัสความถี่ของโทรศัพท์แบบกดปุ่มให้เป็นตัวเลข BCD ขนาด 4 บิท โดยใช้งานร่วมกับคริสตอล 3.579 MHz ซึ่งสามารถเขียนตารางแสดงค่า BCD ที่ได้จากการกดหมายเลขแต่ละตัวดังนี้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

หมายเลข	BCD CODE	หมายเลข	BCD CODE
1	0001	7	0111
2	0010	8	1000
3	0011	9	1001
4	0100	0	1010
5	0101	*	1011
6	0110	#	1100

ตารางที่ 3.1 แสดงค่า BCD ที่ได้จากการกดหมายเลขโทรศัพท์

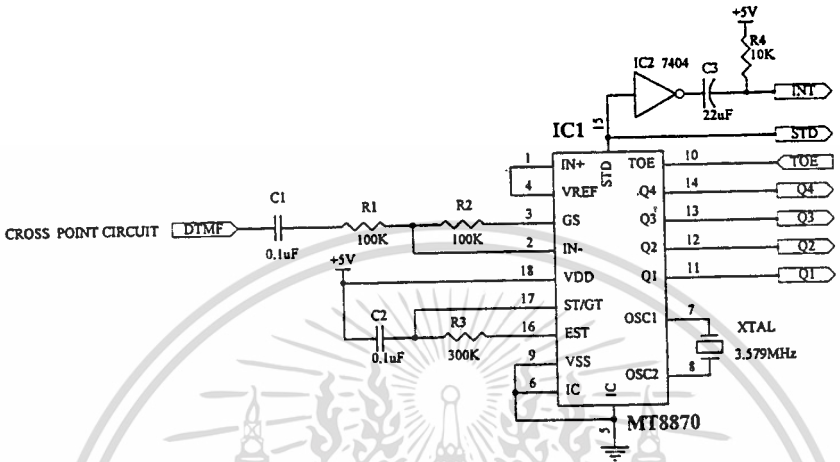
จึงมีการต่อไอซีเบอร์ MT8870 ใช้งานจริงดังวงจรรูปที่ 3.5 เมื่อส่วนอินเตอร์เฟสคู่สายภายนอกทำการปรับอิมพีแดนซ์ให้เหลือ 600 โอห์ม หรือยกหู สัญญาณ DTMF ก็สามารถผ่านภาคนี้มาได้คือ จากขั้วทุติยภูมิของหม้อแปลง ผ่านเข้ามายังขาอินพุทของไอซี MT8870 ส่วนที่ขาเอาท์พุทของไอซี MT8870 จะส่งต่อไปยังส่วนควบคุมระบบ คือ ขา Q1-Q4 เพื่อให้สามารถวิเคราะห์สัญญาณ BCD ที่ถอดรหัสมาได้ แต่การทำงานของไอซี MT8870 นั้น ก็จะถูกควบคุมด้วยขา 10 (TOE) จะเป็นระดับแรงดันโลจิก " 0 " ทำให้เอาท์พุทของไอซี MT8870 เป็นไฮซ์อิมพีแดนซ์ (High impedance) ถ้ามีอินพุทเข้ามา เมื่อไอซี MT8870 ถอดรหัสได้แล้วก็จะเป็นการแปลงสถานะทางโลจิกของขา 15 (STO) จากโลจิก " 0 " ไปเป็น " 1 " เพื่อบอกให้ส่วนควบคุมระบบทราบแล้วจะได้ส่งค่าโลจิก " 1" ออกมาที่ขา 10 (TOE) เพื่อจะได้ปรับสภาพค่าอิมพีแดนซ์ของขาเอาท์พุท (Q1 - Q4) ให้ต่ำลงและส่งข้อมูลไปยังส่วนควบคุมระบบต่อไปซึ่งในขณะปกติเอาท์พุทเป็นไฮซ์อิมพีแดนซ์อยู่นั้นจะไม่มีผลต่อการกดคีย์บอร์ดของโทรศัพท์เพราะข้อมูลส่งออกมาไม่ได้

วงจรถอดรหัสสัญญาณ DTMF ส่วนที่ 2

ในส่วนที่ 2 การทำงานของวงจรจริงนั้นก็จะทำงานเหมือนกับภาคถอดรหัสในส่วนที่ 1 ที่ได้เสนอไปแล้วแต่ที่เสนอไปแล้วนั้นจ ถอดรหัส เฉพาะในกรณีของสัญญาณ DTMF ที่เรียกเข้ามาจากสายนอกเท่านั้น ซึ่ง วงจรถอดรหัส DTMF ในชุดนี้จะเป็นการถอดรหัสสัญญาณ DTMF ของคู่สายภายใน เช่น ในกรณีที่คู่สายภายในยกหูขึ้นมา ภาคตรวจจับการยกหูตรวจสอบได้ว่ามีการยกคู่สาย ส่วนควบคุมระบบ ก็จะสั่งให้ภาคครอสพอยท์ ต่อสัญญาณให้กดหมายเลข (DIAL TONE) ไปยังคู่สายที่มีการยกหูเพื่อให้ผู้เรียกที่ทำการยกหูกดหมายเลขเพื่อที่จะให้ส่วนถอดรหัสสัญญาณ DTMF ถอดรหัสออกมาว่าต้องการติดต่อภายในหรือภายนอก เช่น ถ้ากดเลข "9" เป็นการติดต่อภายนอก ภาคควบคุมระบบ ก็จะได้ไปส่งให้ส่วนอินเตอร์เฟสกับคู่สายภายนอก ทำการปรับอิมพีแดนซ์ให้ต่ำลงเหลือ 600 โอห์ม เหมือนกับการยกหูโทรศัพท์ทั่วไป แล้วก็ให้ผู้เรียกกดหมายเลขที่ต้องการติดต่อต่อไป ส่วนในกรณีที่ผู้ยกหูโทรศัพท์กดหมายเลข "0" คือเป็นการติดต่อภายในภาควงจรถอดรหัส DTMF ก็จะถอดรหัสแล้วส่งให้ภาคควบคุมระบบ ภาคควบคุมระบบ ก็จะส่งสัญญาณให้กดหมายเลขไปยังคู่สายที่ยกหูเมื่อผู้ยกหูโทรศัพท์กดหมายเลขของคู่สายภายในที่ต้องการ ส่วนถอดรหัสสัญญาณ DTMF ก็จะถอดรหัสเพื่อที่จะทำการส่งให้ส่วนควบคุมระบบวิเคราะห์ต่อไป ว่าถ้าคู่สายที่เรียกไปว่างก็จะส่งสัญญาณกระดิ่งไปยังผู้ถูกเรียก และจะส่งสัญญาณเรียกกลับไปยังผู้เรียกส่วนในกรณีที่ส่วนควบคุมระบบ ตรวจ

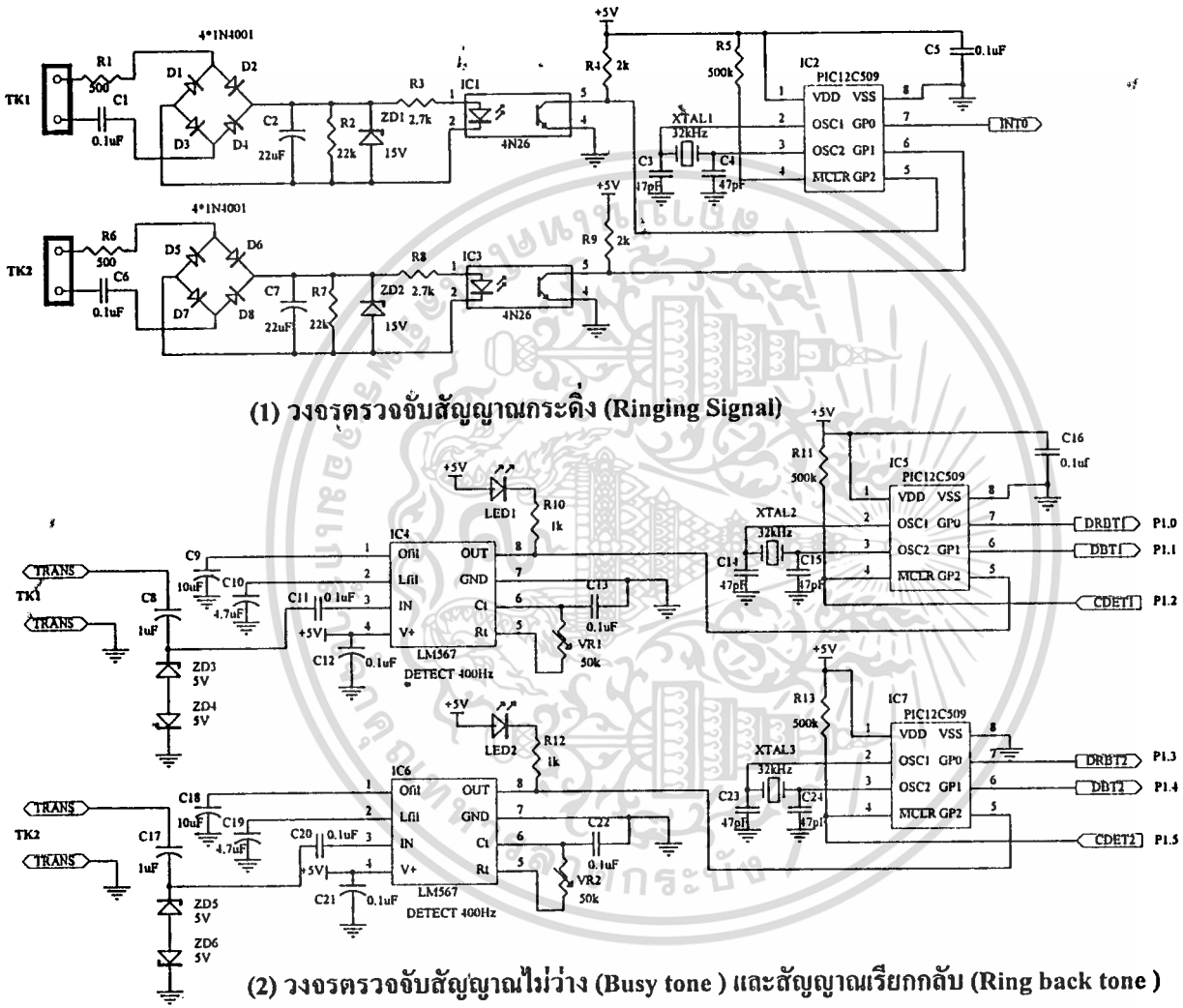
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

สอบแล้วพบว่าคู่สายย่อยที่เรียกไปไม่ว่างก็จะทำการสั่งให้ส่วนครอสพอยท์ ต่อสัญญาณไม่ว่างไปยังผู้ที่ทำการเรียกทันที ในวงจรใช้งานจริงจะประกอบด้วย ไอซี MT8870 ซึ่งด้านอินพุทของวงจรจะต่ออยู่กับ ครอสพอยท์ เพื่อที่จะควบคุมให้คู่สายภายในที่ทำการยกหูมาใช้บริการของภาควงจร ถอดรหัส DTMF นี้แบบแบ่งเวลากัน ส่วนทางด้านเอาท์พุทของวงจรต่ออยู่กับส่วนควบคุมระบบ เพื่อจะได้ทำการวิเคราะห์ข้อมูลตามที่ได้ถอดรหัสออกมาได้



รูปที่ 3.5 แสดงวงจรถอดรหัสสัญญาณ DTMF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.6 แสดงวงจรตรวจจับสัญญาณเสียง

3.4 ส่วนกำเนิดสัญญาณเสียง (TONE GENERATOR CIRCUIT SECTION)

หลักการทำงาน

วงจรในส่วนนี้ทำหน้าที่สร้างสัญญาณเสียงต่าง ๆ ดังนี้

- สัญญาณให้กดเลขหมาย (DIAL TONE)
- สัญญาณไม่ว่าง (BUSY TONE)
- สัญญาณเรียกกลับ (RING BACK TONE)
- สัญญาณเรียก (RINGING SIGNAL)

จากรูปที่ 3.7 วงจรจะประกอบด้วย IC 555 ทำงานในโหมดอะสเตเบิ้ล (ASTABLE MODE) ทั้ง 3 ตัว ซึ่งจะทำหน้าที่กำเนิดความถี่และกำหนดคาบเวลาของแต่ละสัญญาณ ในโหมดอะสเตเบิ้ล ขาเอาท์พุท (ขา 3) จะมีระดับสัญญาณเป็น " high " ในช่วง $T_{on} = 0.693*(R_A + R_B)*C$ และจะมีระดับสัญญาณเป็น " low " ในช่วง $T_{off} = 0.693 *R_B*C$

การสร้างสัญญาณให้หมุน

ที่ IC 555 ตัวที่ 2 มี $R_A = 820$, $R_B = 180$ k , $C = 0.01$ μF

ช่วงที่เอาท์พุทเป็น " high " $T_{on} = 0.693*(R_A +R_B)*C$

$= 0.693*(820 + 180*10^3)* 0.01*10^{-6}$

$= 1.2531$ ms

ช่วงที่เอาท์พุทเป็น " low " $T_{OFF} = 0.693*R_B*C$

$=0.693* 180*10^3*0.01*10^{-6}$

$= 1.2474$ ms

เพราะฉะนั้นความถี่ของ $= 1/(T_{ON} + T_{OFF}) = 400$ Hz

การสร้างสัญญาณไม่ว่าง

พิจารณา IC 555 ตัวที่ 3

ช่วงเวลาที่เอาท์พุทเป็น " high " $T_{ON} = 0.693*(R_A +R_B)*C$

$= 0.693*(200 + 24*10^3)*33*10^{-6}$

$= 0.5534$ s

ช่วงเวลาที่เอาท์พุทเป็น " low " $T_{OFF} = 0.693*R_B*C$

$= 0.693*24*10^3*33*10^{-6}$

$= 0.5488$ s

เอาท์พุทของขา 3 ของ IC 555 ตัวที่ 3 ป้อนเข้าเป็นอินพุทของ AND GATE 7408 และอินพุทอีกสัญญาณหนึ่ง มาจากเอาท์พุท ของขา 3 ของ IC 555 ตัวที่ 2 และผ่าน อินเวอร์เตอร์ 7404 ออกมาได้สัญญาณไม่ว่าง ที่ดัง 0.55 s หยุด 0.55 s และมีความถี่ประมาณ 400 Hz

การสร้างสัญญาณเรียกกลับ

พิจารณา IC 555 ตัวที่หนึ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ช่วงเวลาที่เอาท์พุทเป็น " high " = $0.693*(R_A + R_B)*C$

$= 0.693*(40*10^3 + 42*10^3)*22*10^{-6}$

$= 1.25$ s

ช่วงเวลาที่เอาท์พุทเป็น " low " = $0.693*R_B*C$

$= 0.693*42*10^3*22*10^{-6}$

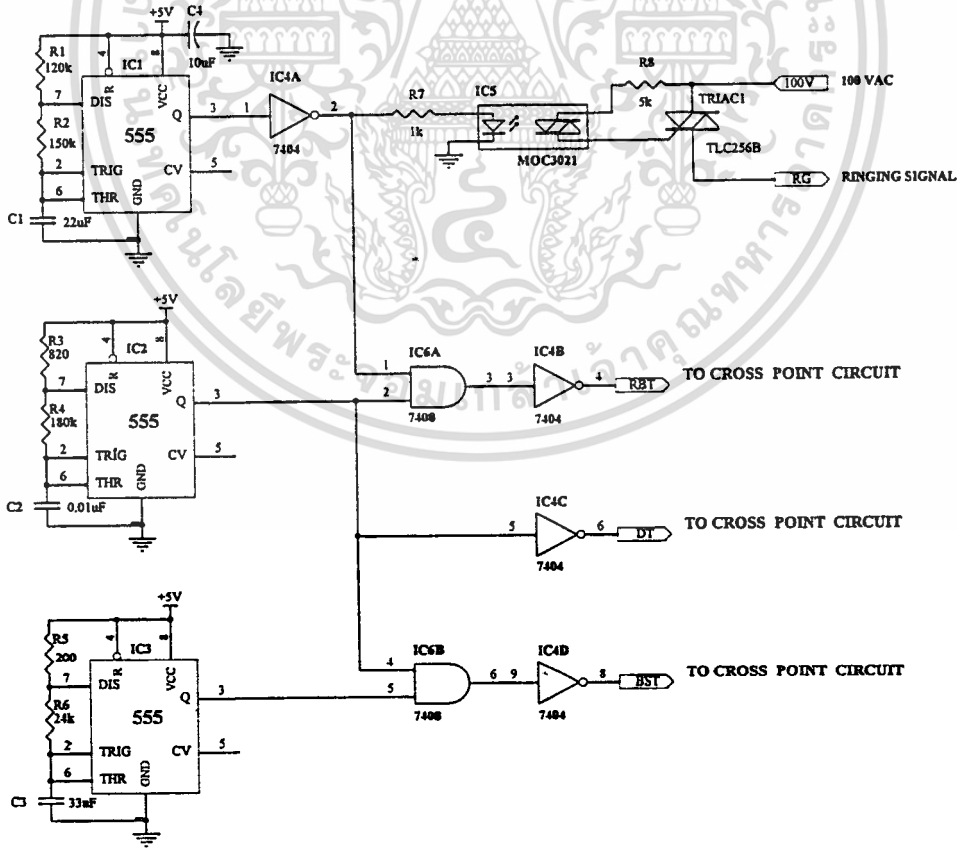
$= 0.64$ s

เอาท์พุทของ 555 ตัวที่ 1 และ ตัวที่ 2 จะป้อนเป็นอินพุทของแอนด์เกท และผ่านอินเวอร์เตอร์ จะได้สัญญาณเรียกกลับออกมา โดย 0.64 s หยุด 1.25 s ความถี่ประมาณ 400 Hz

การสร้างสัญญาณกระดิ่ง

พิจารณา IC 555 ตัวที่หนึ่ง ช่วงเวลาที่เอาท์พุทขา 3 จะมีระดับสัญญาณเป็น " low " เท่ากับ 0.64 s ผ่านอินเวอร์เตอร์ 7404 จะได้ระดับสัญญาณเป็น " high " และจะทำให้ออปโตไตรแอค (OPTOTRIAC) MOC 3021 ทำงานทำให้กระแสไฟ 100 V มาทริกไตรแอคทำให้สัญญาณ 100 V ผ่านไปยังพอร์ทที่ใช้เป็นสัญญาณกระดิ่งเพื่อจ่ายให้วงจรส่วนอินเตอร์เฟสกับคู่สายภายใน

ในทางตรงกันข้าม หากเอาท์พุท ขา 3 เป็น " high " จะทำให้ออปโตไตรแอคไม่ทำงานและทำให้ไตรแอคไม่เกิดการทริก สัญญาณไฟ 100 V จะไม่ผ่านไปยังพอร์ทสัญญาณกระดิ่งได้ เพราะฉะนั้นจะไม่มีสัญญาณกระดิ่งออกมาจึงทำให้ได้ยินสัญญาณกระดิ่งดังเป็นจังหวะเหมือนสัญญาณเรียกกลับ



TONE GENERATOR CIRCUIT

รูปที่ 3.7 แสดงวงจรกำเนิดสัญญาณเสียง

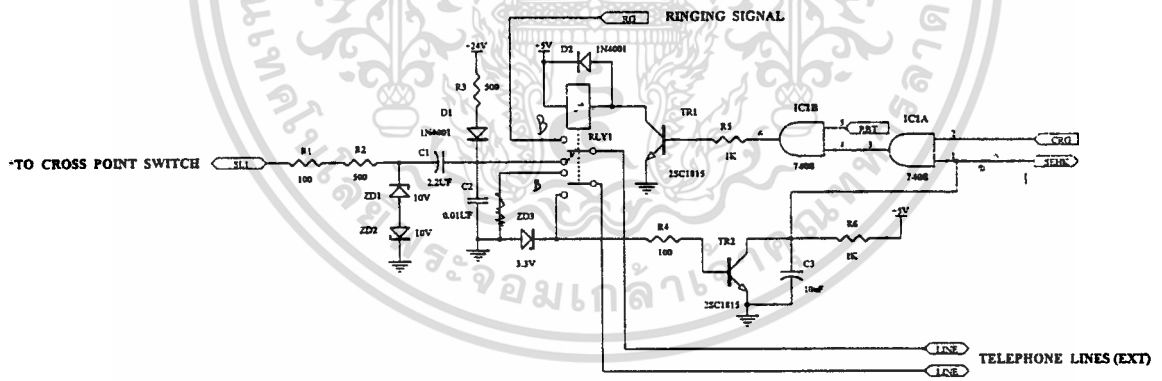
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.5 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน (SUBSCRIBER LINE INTERFACE CIRCUIT SECTION)

หลักการทำงาน

ส่วนนี้จะเป็นส่วนที่ต่ออยู่กับเครื่องโทรศัพท์คู่สายภายในและจะมีการตรวจเช็คการยกหูโทรศัพท์โดยอัตโนมัติด้วย นอกจากนี้ สัญญาณต่าง ๆ ที่จะผ่านเข้ามายังเครื่องโทรศัพท์ ก็จะมาผ่านภาคนี้ก่อน ซึ่งประกอบด้วยอุปกรณ์ต่าง ๆ ดังรูปที่ 3.8

ในกรณีของการโทรเข้ามาจากคู่สายภายนอก จะเห็นได้ว่า สัญญาณจากส่วนควบคุมที่ผ่าน ขาที่ 1 ของ แอนด์เกต 7408 ทำหน้าที่ตรวจสอบด้วยการยกหูของคู่สายภายใน ถ้าคู่สายภายในยกหูโทรศัพท์ สัญญาณที่เข้าที่ขา 1 ของ แอนด์เกต 7408 จะเป็นค่าโลจิก "0" ทำให้ไม่ได้รับสัญญาณกระดิ่งจากคู่สายภายนอก แต่ถ้าคู่สายภายในวางหู จะทำให้ได้รับสัญญาณกระดิ่งเข้ามาได้โดยการควบคุมจากส่วนควบคุมระบบ สัญญาณเอาท์พุทของแอนด์เกต จะไปขับขาเบสของ ทรานซิสเตอร์ 2SC1815 เพื่อที่จะทำหน้าที่ควบคุมรีเลย์ ที่ ทำหน้าที่เชื่อมต่อสัญญาณที่ผ่านเข้ามาจากภาคครอสพอยท์สวิตซ์ กล่าวคือขณะที่ สายในวางสายโทรศัพท์ และสามารถรับสัญญาณกระดิ่งได้ เมื่อคู่สายภายในยกหู รีเลย์ที่ต่อสัญญาณกระดิ่งอยู่ จะสวิตช์ไปให้สัญญาณเสียงพูดจากภาคครอสพอยท์ผ่านเข้ามาทำให้ ผู้สนทนาด้านคู่สายภายในสามารถได้ยินเสียงพูดจากคู่สายภายนอกได้



SUBSCRIBER LINE INTERFACE CIRCUIT

รูปที่ 3.8 แสดงวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

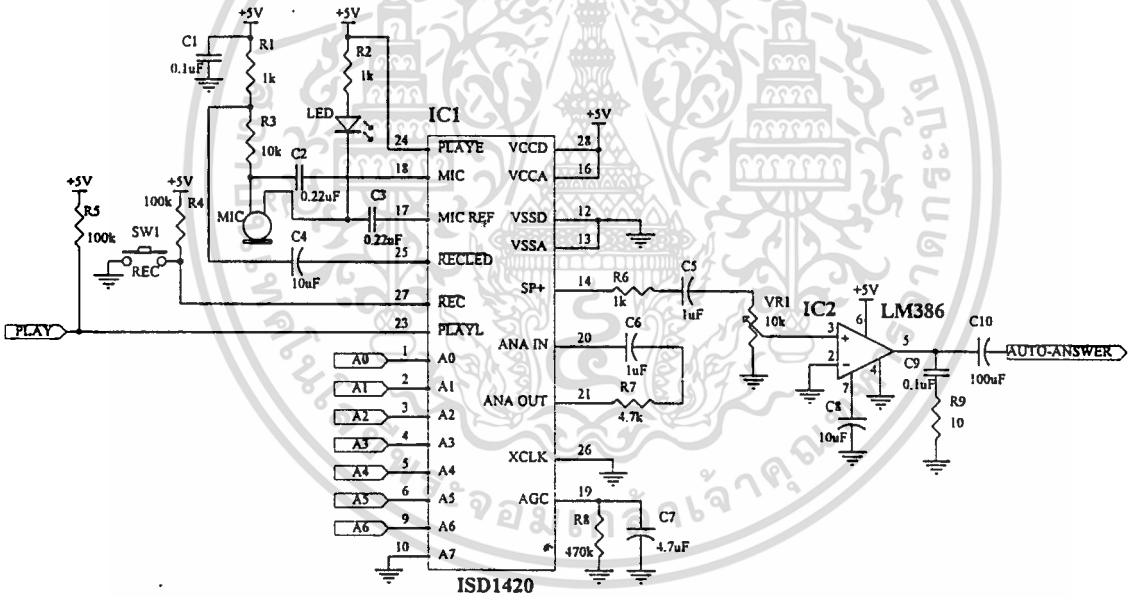
3.6 ส่วนตอบรับโทรศัพท์อัตโนมัติ (AUTOMATIC ANSWER CIRCUIT SECTION)

หลักการทำงาน

ส่วนนี้จะใช้ไอซีเบอร์ ISD1420 ซึ่งจะมีความจุ 128,000 เซล การกำหนดแอดเดรสของการบันทึกและการเล่นกลับนั้น สามารถกำหนดได้โดยขา A0 - A5 เมื่อขา A7 ต้องเป็นลอจิก " 0 " และจะใช้ขา A6 เป็นตัวกำหนดว่าจะเป็น 10 วินาทีแรกหรือ 10 วินาทีหลัง

- เมื่อขา A6 เป็นลอจิก " 0 " จะกำหนดเป็น 10 วินาทีแรก
- เมื่อขา A6 เป็นลอจิก " 1 " จะกำหนดเป็น 10 วินาทีหลัง

จากรูปที่ 3.8 เมื่อขา PLAYL ได้รับลอจิก "0" จะทำให้สัญญาณเสียงที่ถูกกำหนดตำแหน่งแอดเดรสโดยขา A0-A5 ออกจากขา 14 ของไอซี ISD1420 แล้วเข้าวงจรขยายเสียงที่ใช้ไอซี LM386 จากนั้นจึงออกลำโพง และสัญญาณเสียงนั้นจะเข้าไปที่ขาเบสของทรานซิสเตอร์ ซึ่งทำหน้าที่เป็นตัวขับ (Driver) ทำให้รีเลย์ทำงาน เสียงจึงออกไปยังคู่สายโทรศัพท์ได้โดยผ่านหม้อแปลงคับปลิ้ง ขากำหนดแอดเดรส (A0-A5), ขาA6และขา PLAYL จะต่อเข้ากับพอร์ทของ 8255 เพื่อใช้ในการควบคุมการเล่นกลับ ดังรูปที่ 3.8



รูปที่ 3 .9 แสดงวงจรตอบรับโทรศัพท์อัตโนมัติ

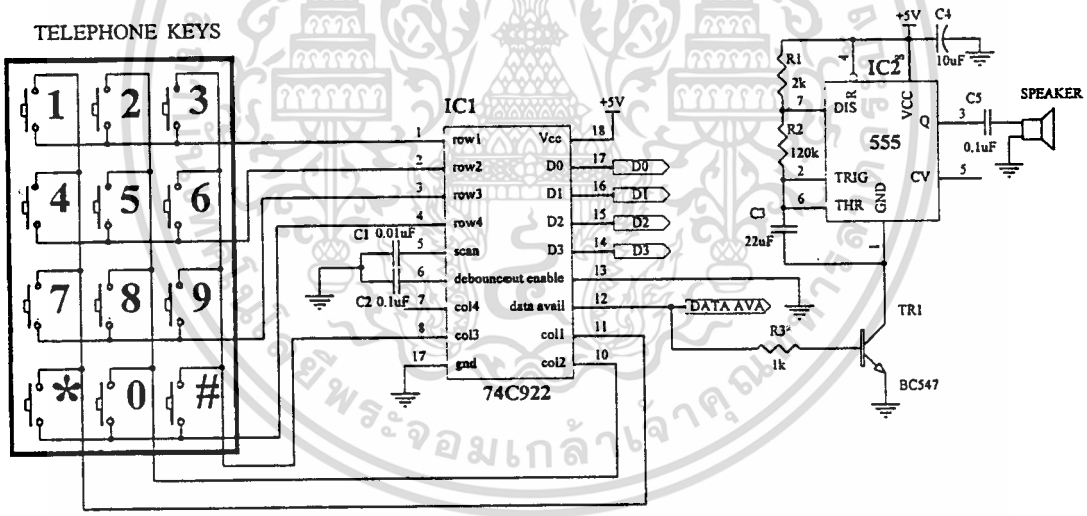
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.7 ส่วนจอแสดงผล ,ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล (DOT MATRIX LCD MODULE , KEYBOARD ENCODER AND CONTROL CIRCUIT SECTION #2)

หลักการทำงาน

ส่วนนี้จะใช้เป็นส่วนแสดงผลการใช้งานของเครื่องโดยใช้จอแสดง LCD โดยสามารถแสดงผลได้ 4 บรรทัด และจะมีคีย์บอร์ดขนาด 3 หลัก 4 แถว ในการสั่งงาน กับ ส่วนเข้ารหัสการกดคีย์โดยจะใช้ไอซีเบอร์ 74C922 เมื่อมีการกดคีย์เกิดขึ้นส่วนนี้จะทำการเข้ารหัสเป็นสัญญาณไบนารีขนาด 4 บิต ตามหมายเลขที่กด สัญญาณนี้จะถูกส่งเข้าส่วนควบคุมเพื่อทำการประมวลผลและแสดงผลออกมาตามฟังก์ชั่นที่กดทางจอ LCD ต่อไป

ในส่วนนี้จะใช้ 8255 หนึ่งตัว เป็นตัวรับสัญญาณ 4 บิตที่ได้จากตัวเข้ารหัสการกดคีย์ เพื่อส่งค่าไปส่วนควบคุมการแสดงผล และเมื่อมีการกดปุ่มแต่ละครั้งจะเกิดพัลส์สูง (high) ขึ้นที่ขา data available ซึ่งจะไปขับทรานซิสเตอร์ BC547 ให้อยู่ในสถานะออน ทำให้วงจรอะสเตเบิ้ลทำงานเกิดเสียงความถี่ประมาณ 1 khz ส่งออกไปที่ลำโพง ซึ่งเป็นการตอบสนองให้ผู้ใช้รู้ว่ามีการกดเกิดขึ้นแล้ว



รูปที่ 3.10 วงจรเข้ารหัสการกดคีย์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.8 ส่วนอินพุท/เอาท์พุท พอร์ท (I/O PORT SECTION)

หลักการทำงาน

ในส่วนนี้จะเป็นส่วนที่เป็นอินพุทและเอาท์พุทพอร์ทที่ส่วนควบคุมระบบจะใช้ในการสั่งงานและรับค่าจากวงจรต่าง ๆ เพื่อนำไปประมวลผลการทำงาน จะใช้ไอซี 8255 จำนวน 7 ตัว โดยแต่ละตัวทำหน้าที่ดังต่อไปนี้

- 8255 ตัวที่ 1 (PORTA = OUT, PORTB = IN, PORTC = IN)

PORTA : บิต A.0 - A.5 ควบคุมการส่งค่าเอาท์พุทจากตัวตรวจจับ DTMF ทั้ง 6 ตัว
บิต A.6 - A.7 ควบคุมการยกหูของสายนอกสายที่ 1 และ 2 ตามลำดับ

PORTB : บิต B.0 - B.3 รับรหัสไบนารี 4 บิตจากตัวตรวจจับ DTMF ของสายนอก
บิต B.4 - B.7 รับรหัสไบนารี 4 บิตจากตัวตรวจจับ DTMF ของสายใน

PORTC : บิต C.0 - C.5 รับสัญญาณแสดงการตรวจพบ (STD) DTMF ทั้ง 6 ตัว

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	HK2	HK1	TOE6	TOE5	TOE4	TOE3	TOE2	TOE1

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	Q4	Q3	Q2	Q1	Q4	Q3	Q2	Q1

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	-	-	STD6	STD5	STD4	STD3	STD2	STD1

- 8255 ตัวที่ 2 (PORTA = OUT, PORTB = OUT, PORTC = OUT)

PORTA : บิต A.0 - A.6 ส่งสัญญาณ STROBE ของ CROSS POINT SW

PORTB : บิต B.0 - B.6 ส่งสัญญาณ DATA IN ของ CROSS POINT SW

PORTC : บิต C.0 - C.3 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 1
บิต C.4 - C.7 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 2

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	-	SB7	SB6	SB5	SB4	SB3	SB2	SB1

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	-	DI7	DI6	DI5	DI4	DI3	DI2	DI1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	A2	B2	C2	D2	A1	B1	C1	D1

- 8255 ตัวที่ 3 (PORTA = OUT, PORTB = OUT, PORTC = OUT)

PORTA : บิต A.0 - A.3 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 3

บิต A.4 - A.7 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 4

PORTB : บิต B.0 - B.3 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 5

บิต B.4 - B.7 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 6

PORTA : บิต C.0 - C.3 ส่งค่าแอดเดรสสวิทช์ 4 บิต ของ CROSS POINT SW ตัวที่ 7

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	A4	B4	C4	D4	A3	B3	C3	D3

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	A6	B6	C6	D6	A5	B5	C5	D5

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	-	-	-	-	A7	B7	C7	D7

- 8255 ตัวที่ 4 (PORTA = IN, PORTB = OUT, PORTC = OUT)

PORTA : บิต A.0 - A.7 รับสถานะการยกหู-วางหูของสายในสายที่ 1-8

PORTB : บิต B.0 - B.7 ควบคุมการส่งสัญญาณกระดิ่งไปสายในสายที่ 1-8

PORTC : บิต C.0 - C.6 ส่งค่าแอดเดรสควบคุมการเล่นเสียงตอบรับอัตโนมัติ

บิต C.7 บิตควบคุมการเล่นเสียงตอบรับอัตโนมัติ

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	SEHK8	SEHK7	SEHK6	SEHK5	SEHK4	SEHK3	SEHK2	SEHK1

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	CRG8	CRG7	CRG6	CRG5	CRG4	CRG3	CRG2	CRG1

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	PLAY	A6	A5	A4	A3	A2	A1	A0

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- 8255 ตัวที่ 5 (PORTA = OUT, PORTB = OUT, PORTC = IN)

PORTA : บิต A.0 - A.7 ส่งสัญญาณควบคุมการส่งผ่านข้อมูล ไปส่วนควบคุม#2

PORTB : บิต B.0 - B.7 ส่งข้อมูลการใช้งานต่างๆ ไปส่วนควบคุม#2

PORTC : บิต C.0 - C.7 รับสัญญาณ ACK การรับข้อมูลจากส่วนควบคุม#2

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK

- 8255 ตัวที่ 6 (PORTA = IN, PORTB = IN, PORTC = OUT)

PORTA : บิต A.0 - A.7 รับสัญญาณควบคุมการส่งผ่านข้อมูลจากส่วนควบคุมระบบ#1

PORTB : บิต B.0 - B.7 รับข้อมูลการใช้งานต่างๆ จากส่วนควบคุมระบบ#1

PORTC : บิต C.0 - C.7 ส่งสัญญาณ ACK การรับข้อมูลไปส่วนควบคุมระบบ#1

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK	S1-LK

PORT	B.7	B.6	B.5	B.4	B.3	B.2	B.1	B.0
NAME	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK	D-LK

PORT	C.7	C.6	C.5	C.4	C.3	C.2	C.1	C.0
NAME	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK	S2-AK

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- 8255 ตัวที่ 7 (PORTA = IN)

PORTA : บิต A.0 - A.3 รับเอาท์พุทรหัสไบนารี 4 บิต จากตัวเข้ารหัสคีย์

บิต A.4 รับสัญญาณแสดงการตรวจพบการกดคีย์สวิทช์ (Data Available)

PORT	A.7	A.6	A.5	A.4	A.3	A.2	A.1	A.0
NAME	-	-	-	D-AVA	D3	D2	D1	D0

3.9 ส่วนแหล่งจ่ายไฟเลี้ยง (POWER SUPPLY CIRCUIT SECTION)

วงจรส่วนจ่ายไฟเลี้ยง แบ่งได้เป็น 4 ส่วน

1. ส่วนสร้างกระแสไฟตรง 24 โวลท์ และ 5 โวลท์ ซึ่งจะใช้ เรกูเลเตอร์ในการสร้างไฟตรง 24 V ให้แก่คู่สายภายใน และไฟตรง 5 V ใช้สำหรับไอซีในวงจรต่าง ๆ
2. ส่วนสวิตช์ชิ่ง ดีซี ทู ดีซี คอนเวอร์เตอร์ (Switching DC to DC convertor) เป็นส่วนที่จะแปลงแรงดัน 24 V ให้เป็น 5 V โดยใช้ไอซีเบอร์ MC34063 จะมีการคำนวณดังนี้

แรงดันเอาท์พุท $|Vout| = 1.25*(1+R10/R11)$

$= 1.25*(1+15k/5k)$

$= 1.25*(1+3)$

$= 5\ V$

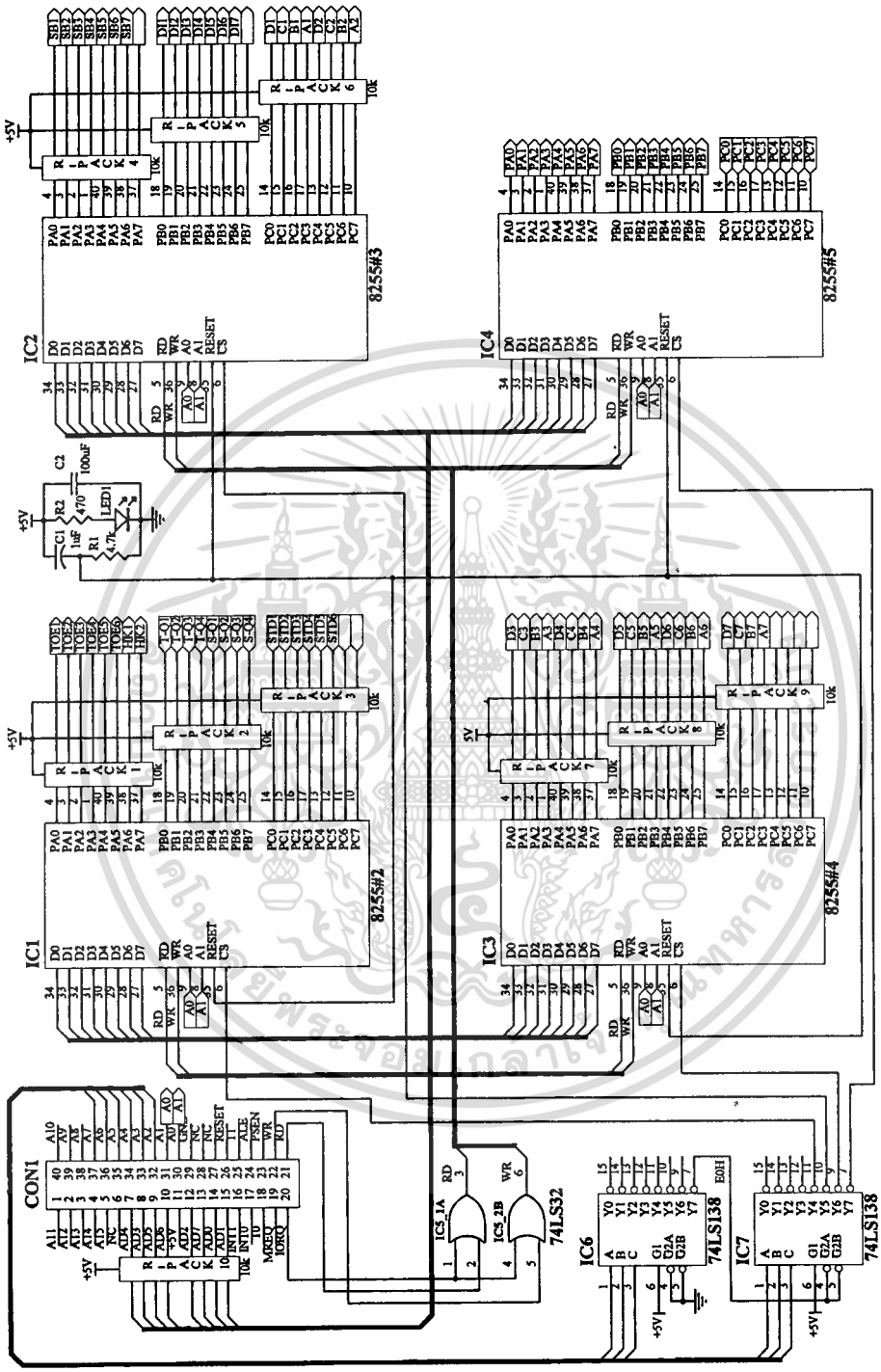
3. ส่วนชาร์จแบตเตอรี่อัตโนมัติโดยจะใช้วงจรเปรียบเทียบแรงดัน ในสภาวะปกติคือมีไฟฟ้าเข้ามา ถ้าแบตเตอรี่มีแรงดันต่ำกว่า 24 V จะทำให้รีเลย์สวิตช์ทำให้เกิดการชาร์จแบตเตอรี่ขึ้นจนกระทั่ง เมื่อแบตเตอรี่ถูกชาร์จจนเต็ม จะทำให้ที่ ขาลบของออปแอมป์ มีแรงดันมากกว่า 6 V ทำให้รีเลย์ไม่มีการสวิทช์ หรือเลิกการชาร์จ ซึ่งในช่วงระหว่างการชาร์จวงจรส่วนต่าง ๆ ของโครงงานก็ยังคงใช้ไฟจากภายนอกอยู่ จนกระทั่งเมื่อเกิดไฟฟ้าดับ กระแสจากแบตเตอรี่จึงจะสามารถไปเลี้ยงวงจรส่วนต่าง ๆ ได้
4. ส่วนสร้างสัญญาณไฟสลับ 90 โวลต์พีค (90 Vp) ความถี่ 25 Hz เพื่อนำไปใช้เป็นสัญญาณกระดิ่ง จากรูป ไอซี 555 จะสร้างสัญญาณที่มีความถี่

$f = 1.44/(1\ k + 2*27\ k)*1*10\ E-6$

$= 26.18\ Hz$

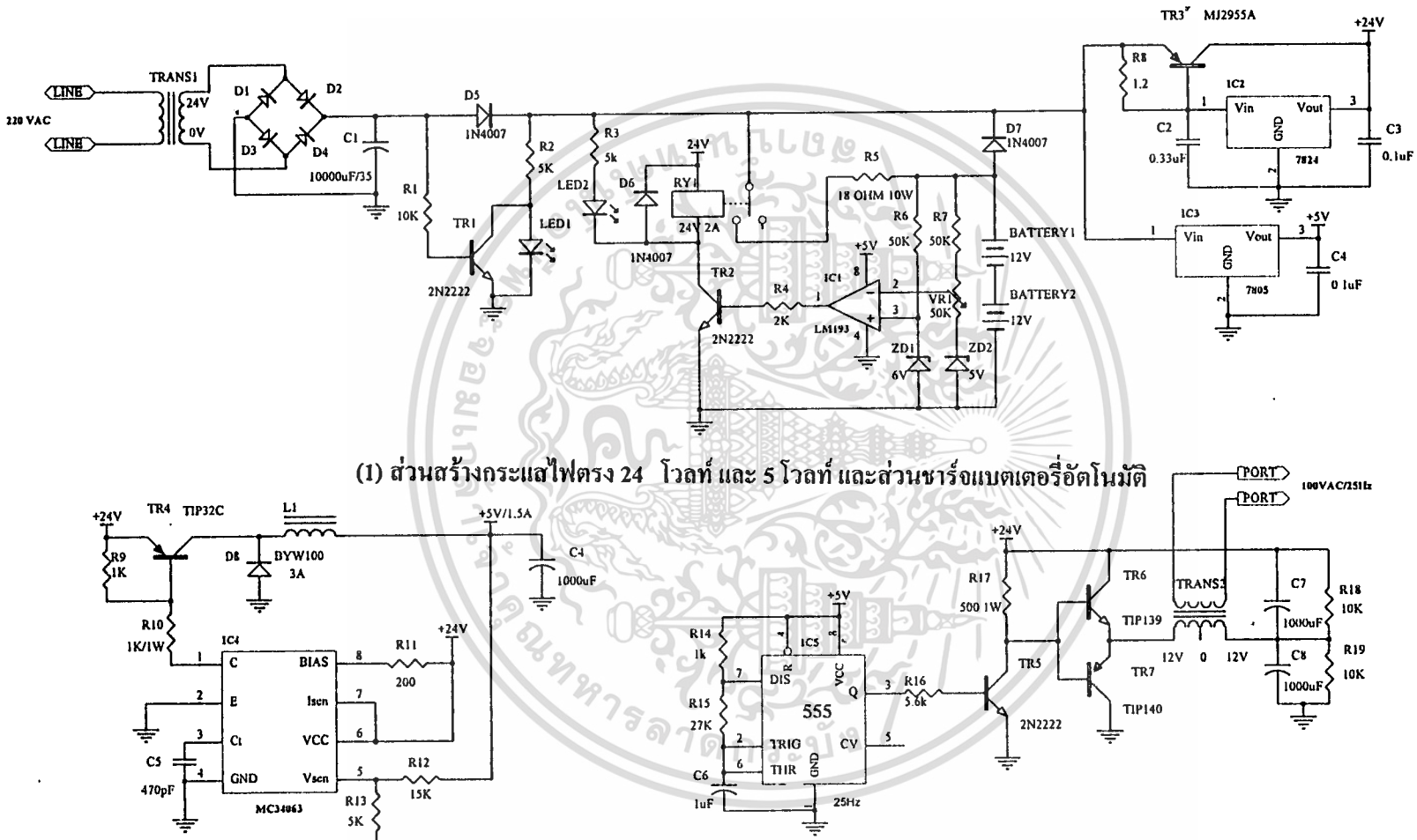
ซึ่งสัญญาณไฟฟ้ากระแสสลับนี้ จะไปขับ (DRIVE) ทรานซิสเตอร์ 2 ตัว คือ TIP31C และ TIP32C ให้เกิดสัญญาณเอซี 12 V ความถี่ประมาณ 25 Hz คร่อมหม้อแปลงด้านอินพุท (primary) ซึ่งจะเหนี่ยวนำให้เกิดสัญญาณเอซี ด้านเอาท์พุท (secondary)ของหม้อแปลงมีค่าประมาณ 100 V ความถี่ประมาณ 25 Hz ซึ่งจะนำไปใช้ในการสร้างสัญญาณกระดิ่ง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.11 วงจรส่วนอินพุท/เอาท์พุท พอร์ท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



(1) ส่วนสร้างกระแสไฟตรง 24 โวลท์ และ 5 โวลท์ และส่วนชาร์จแบตเตอรี่อัตโนมัติ

(2) ส่วนสวิตช์ชิ่ง ดีซี ทู ดีซี คอนเวอร์เตอร์ (24V to 5V)

(3) ส่วนสร้างสัญญาณไฟสลับ 100 โวลต์ 25 Hz

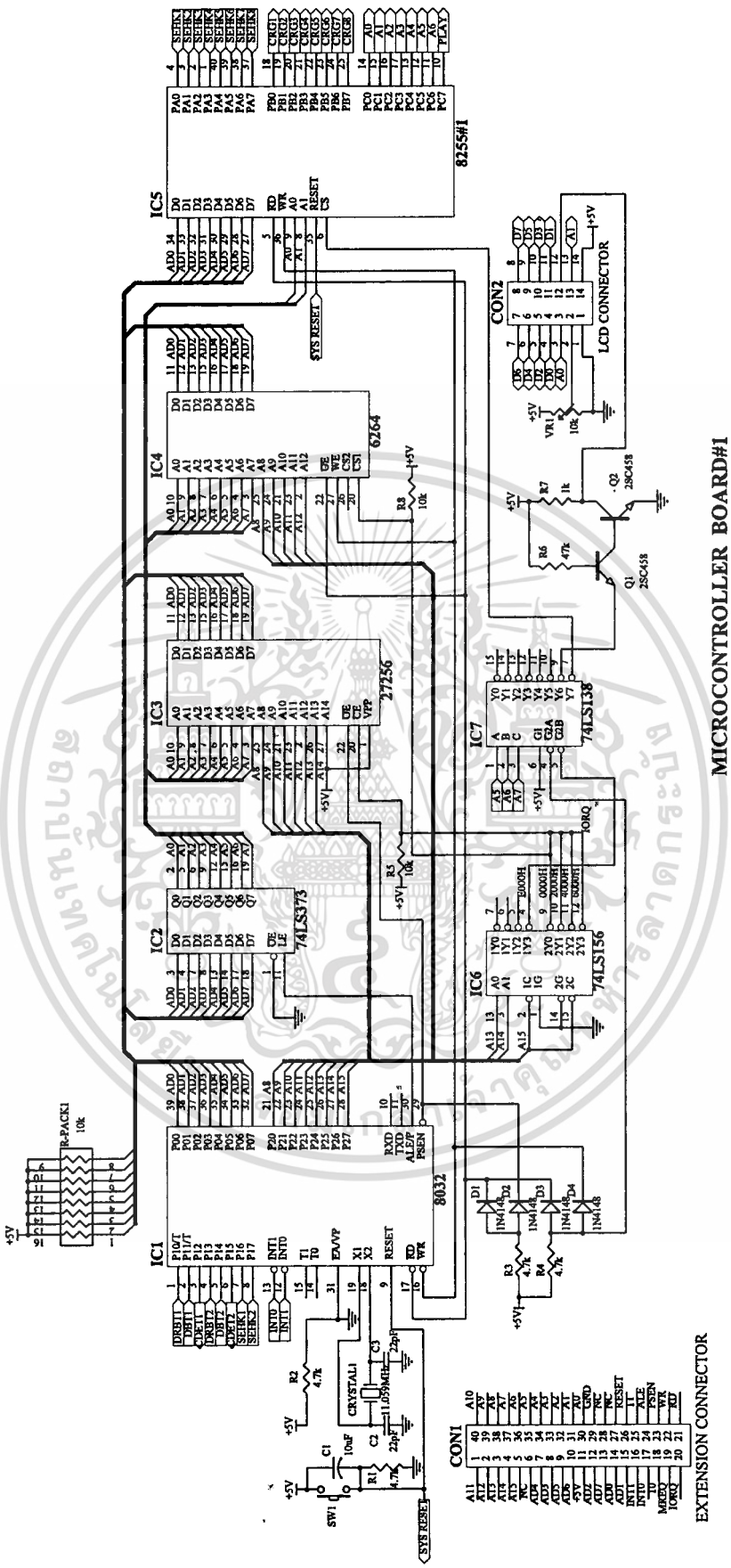
รูปที่ 3.12 แสดงวงจรแหล่งจ่ายไฟเลี้ยง

3.10 ส่วนควบคุมระบบ (CONTROL CIRCUIT SECTION #1)

ส่วนนี้เป็นส่วนที่ทำหน้าที่ในการควบคุมและประมวลผลการทำงานของชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติทั้งหมด ในส่วนนี้จะประกอบด้วย 2 ส่วนสำคัญ คือ ไมโครคอนโทรลเลอร์เบอร์ 80C31 และหน่วยความจำ

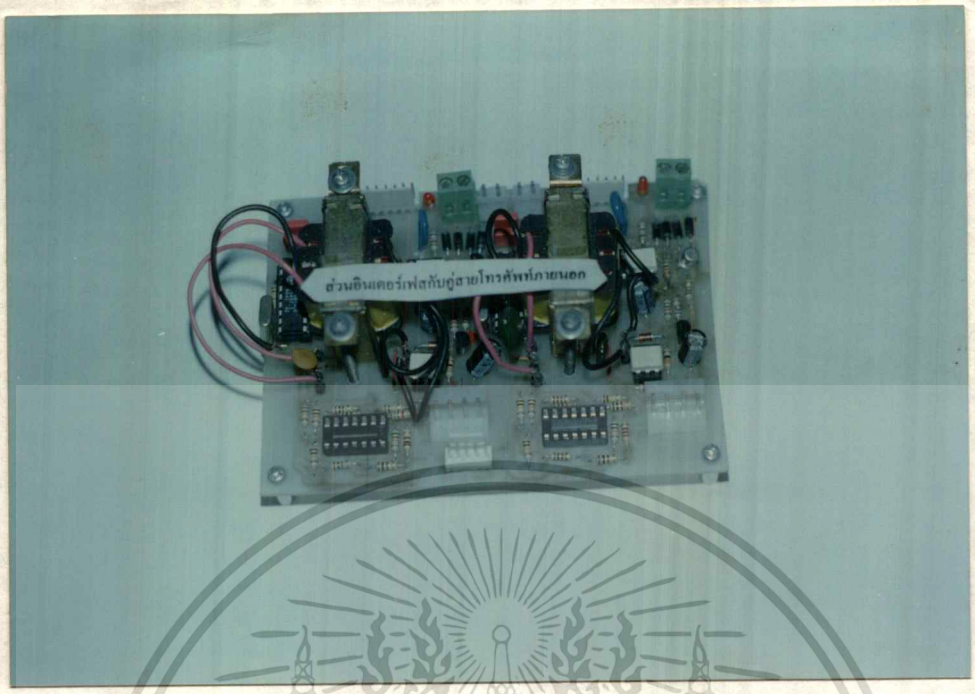
- ไมโครคอนโทรลเลอร์เบอร์ 80C31 ทำหน้าที่ควบคุมลำดับการทำงานของระบบ เป็นตระกูลเดียวกับ MCS-51 ประมวลผลกลางขนาด 8 บิต สามารถอ้างตำแหน่งของหน่วยความจำโปรแกรมได้ 64 กิโลไบต์ และสามารถอ้างตำแหน่งของหน่วยความจำข้อมูลได้ 64 กิโลไบต์ 80C31 จะมีหน่วยความจำแบบ RAM ภายในจำนวน 128 ไบต์ คือตำแหน่ง 00H - 7FH
- หน่วยความจำจะใช้ 2 ประเภท คือ หน่วยความจำโปรแกรม (ROM) และหน่วยความจำข้อมูล (RAM)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

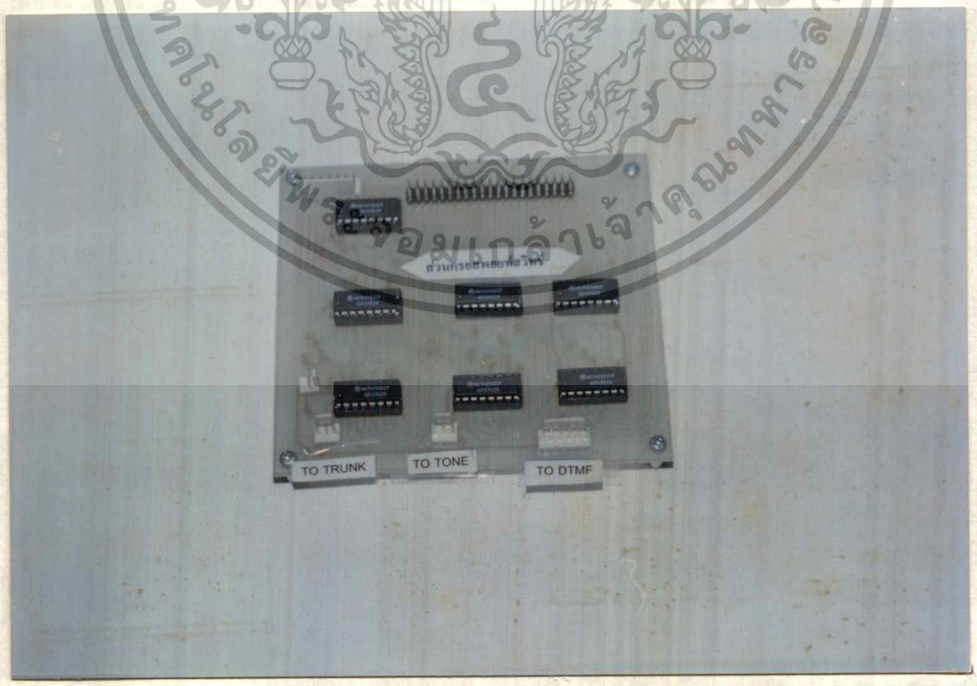


รูปที่ 3.13 วงจรส่วนควบคุมระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

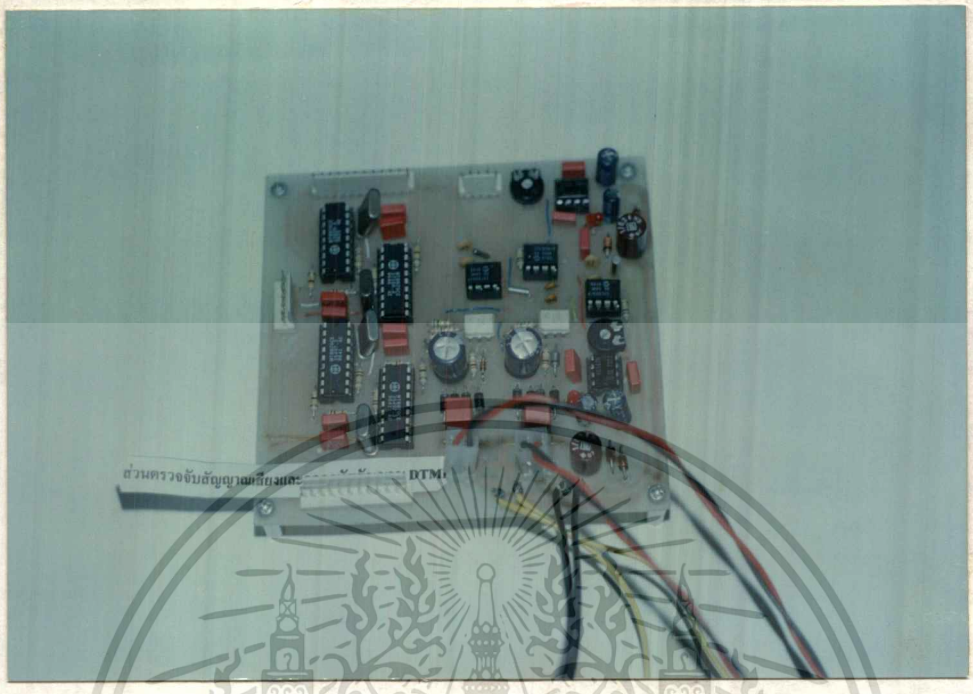


รูปที่ 3.14 แสดงส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก

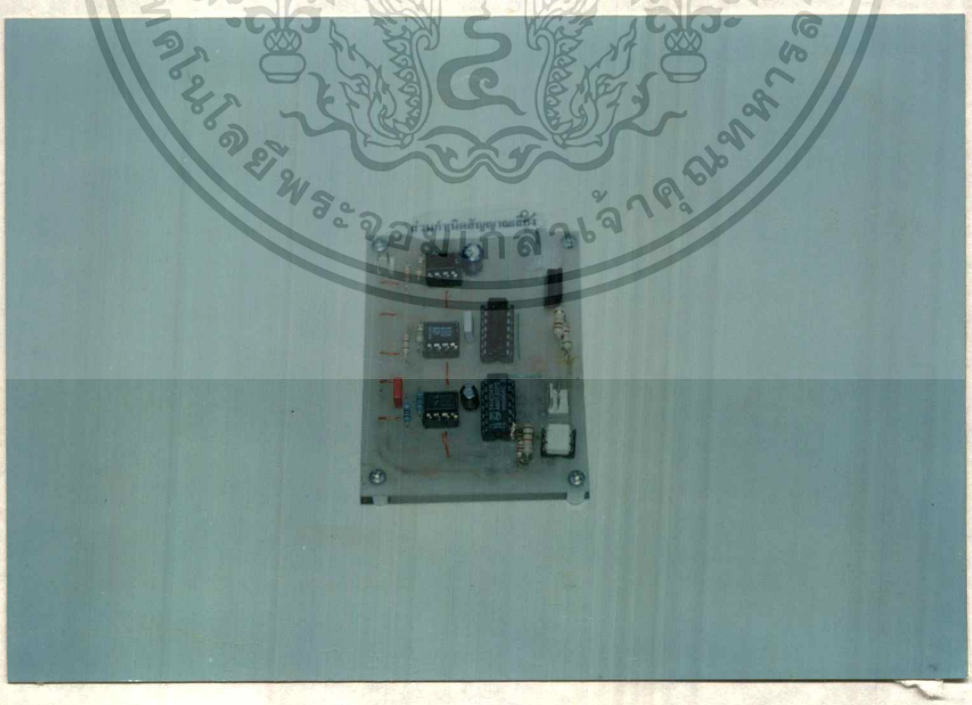


รูปที่ 3.15 แสดงส่วนครอสพอยท์สวิทช์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

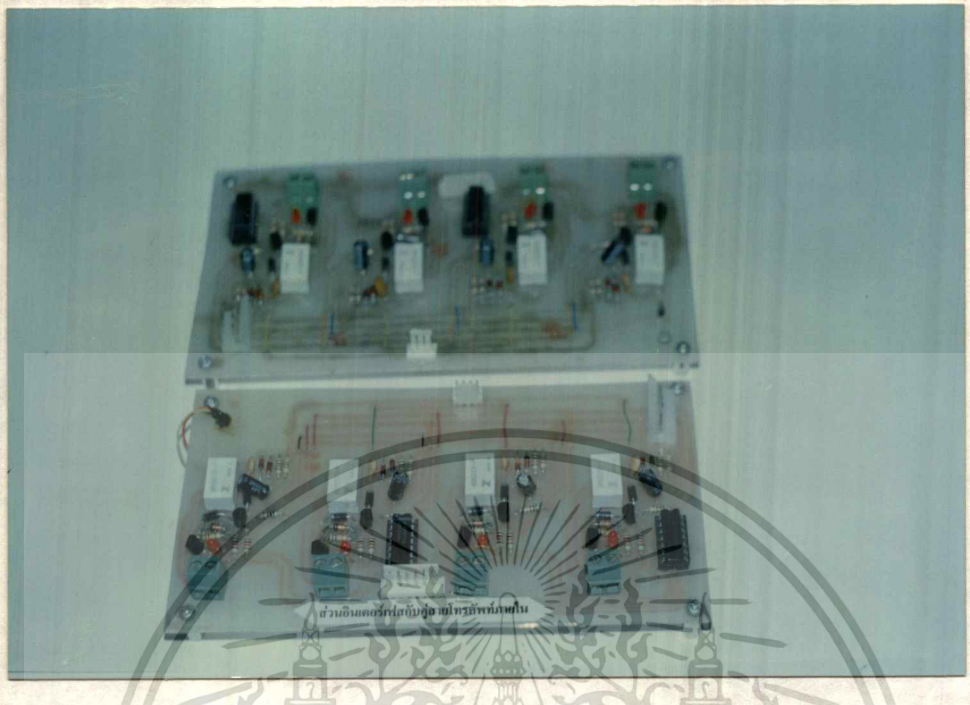


รูปที่ 3.16 ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF

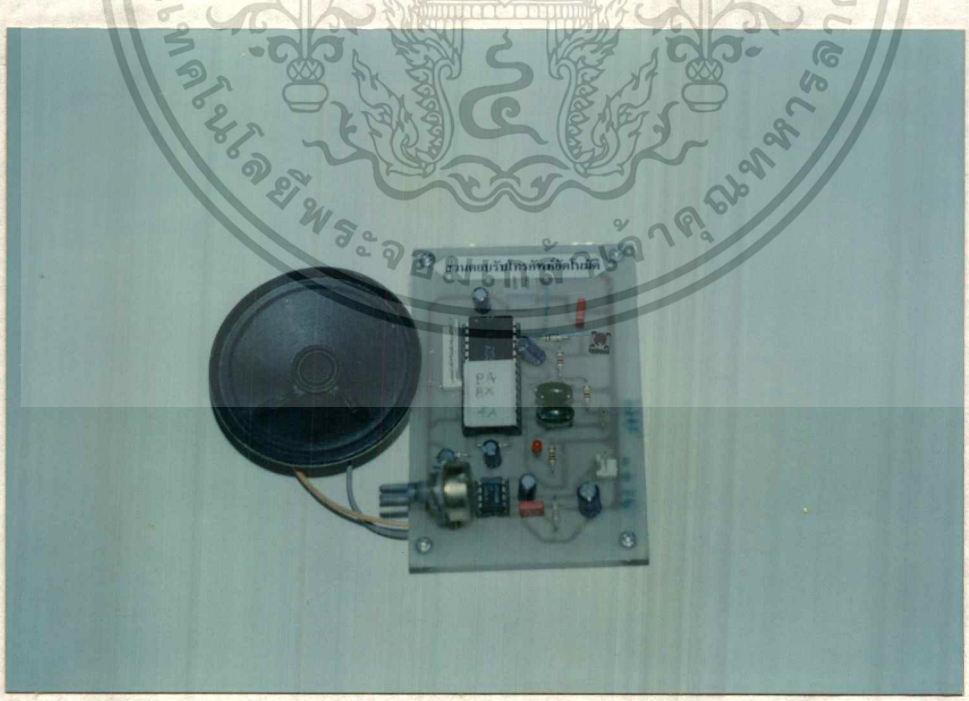


รูปที่ 3.17 ส่วนกำเนิดสัญญาณเสียง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

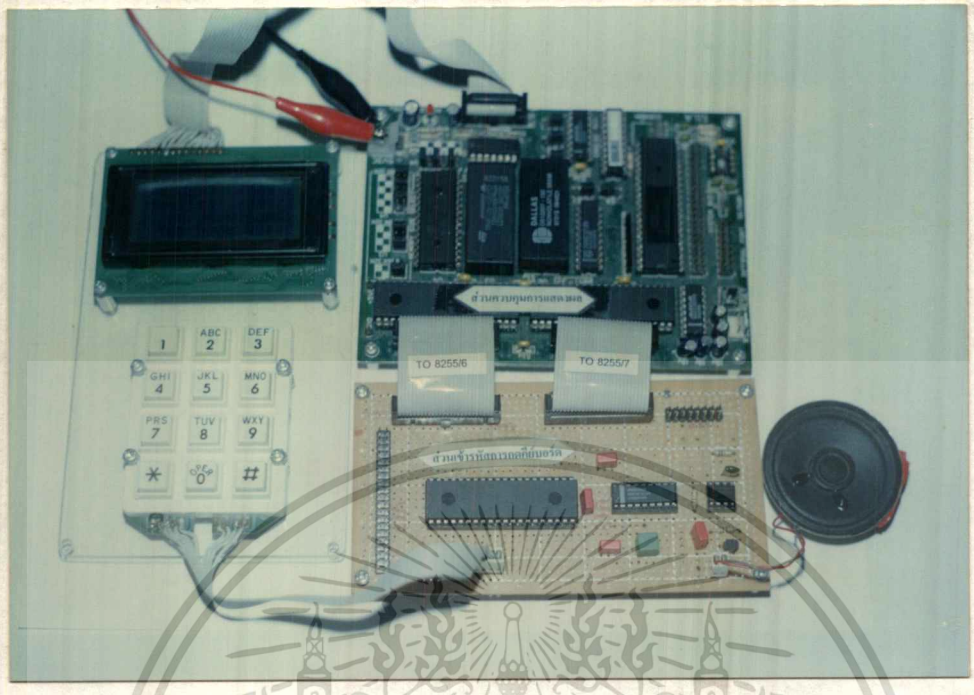


รูปที่ 3.18 ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน

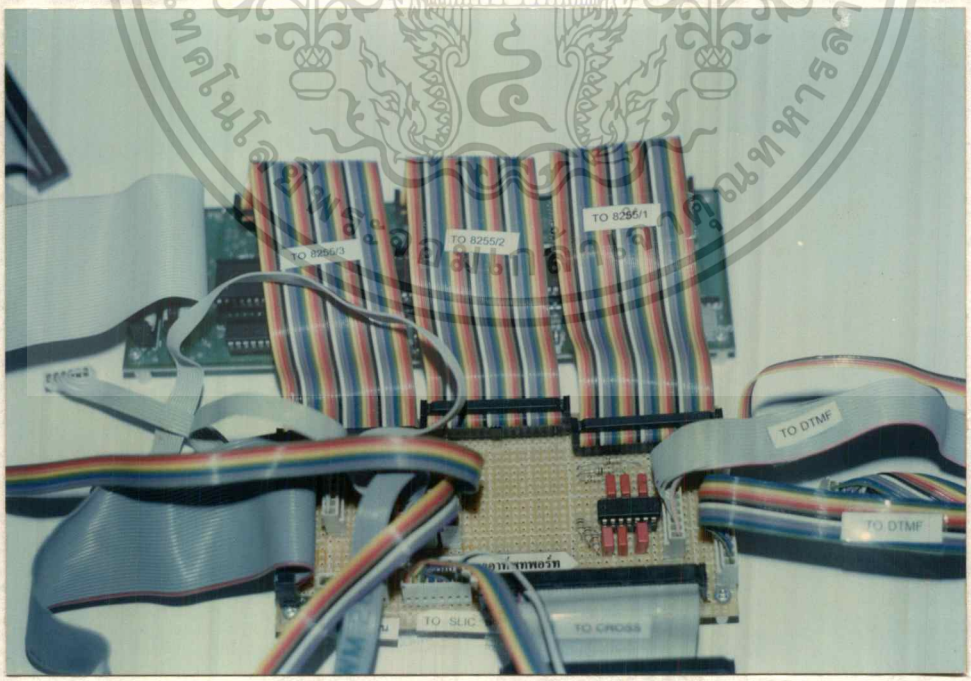


รูปที่ 3.19 ส่วนตอบรับโทรศัพท์อัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

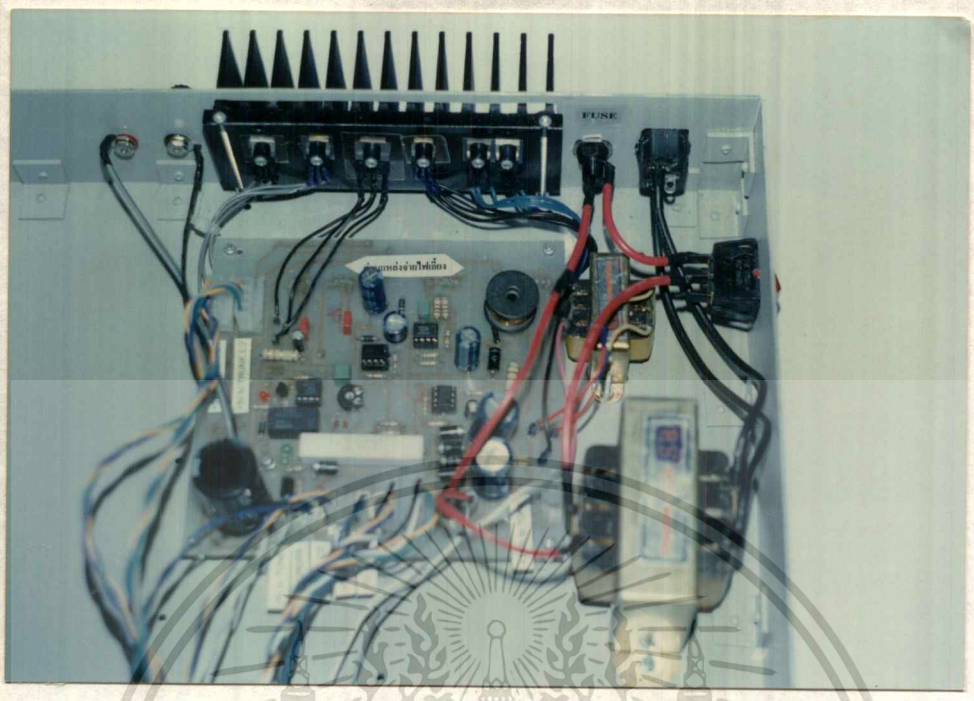


รูปที่ 3.20 ส่วนจอแสดงผล, ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล

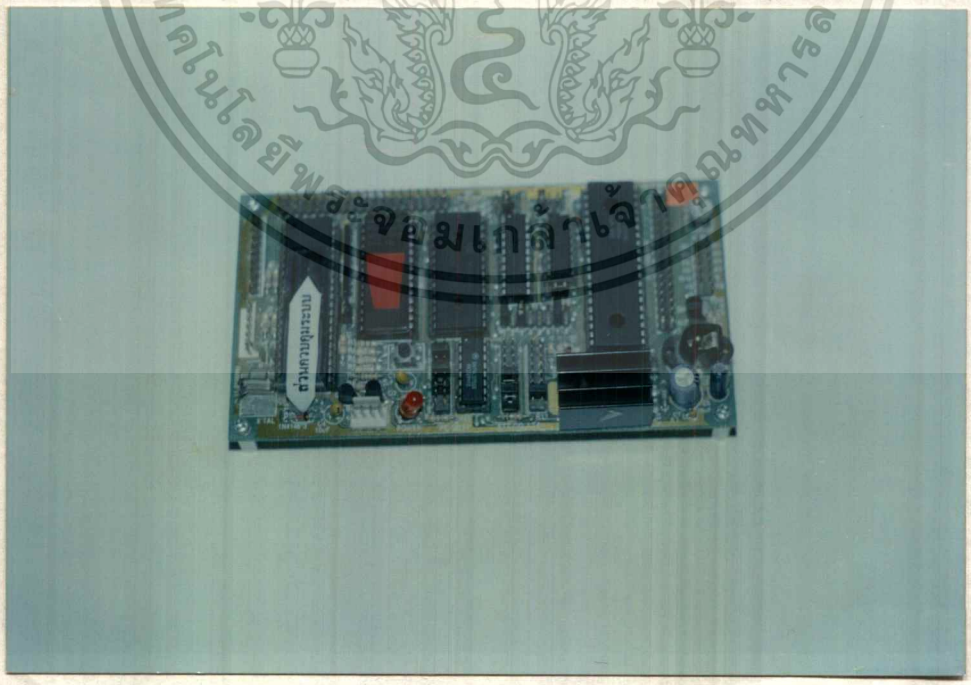


รูปที่ 3.21 ส่วนอินพุท/เอาท์พุทพอร์ท

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

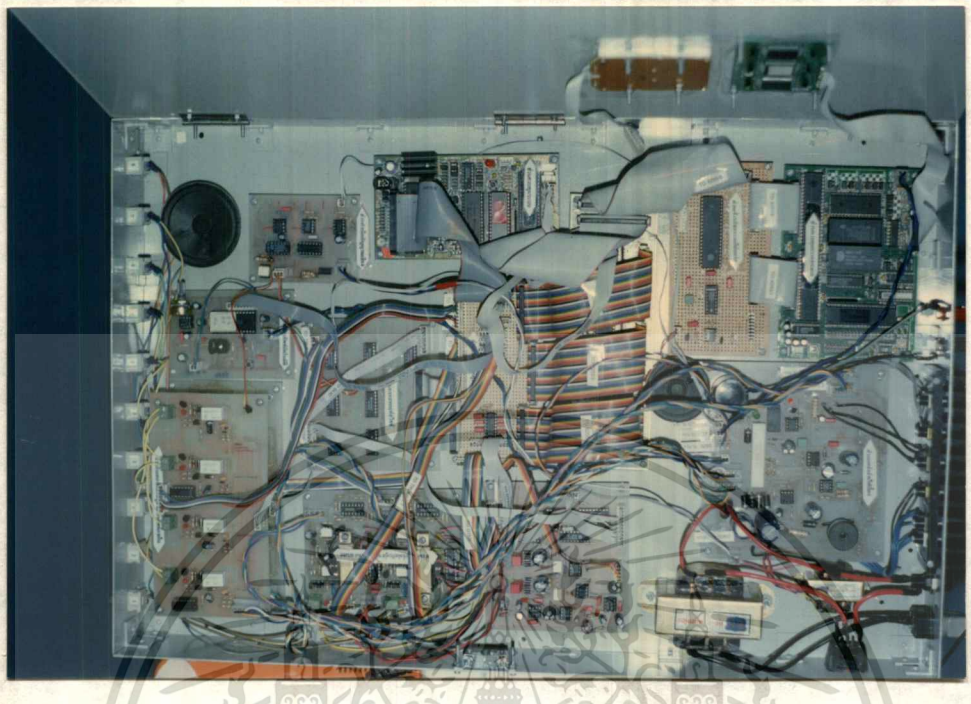


รูปที่ 3.22 ส่วนแหล่งจ่ายไฟเลี้ยง



รูปที่ 3.23 ส่วนควบคุมระบบ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 3.24 แสดงโครงงานประกอบสำเร็จภายใน

รูปที่ 3.25 แสดงโครงงานประกอบสำเร็จภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 4

การทดลองและผลการทดลอง

การทดลองของโครงงาน แบ่งเป็น 2 ขั้นตอน คือ

1. การทดลองแยกแต่ละวงจร คือจะเป็นการทดสอบแต่ละวงจรว่าจะสามารถทำงานได้ถูกต้องหรือไม่
2. การทดลองโดยรวมวงจรทุกวงจรเข้าด้วยกัน แล้วทำการควบคุมการทำงานโดยโปรแกรมภาษา Asembly ของ ไมโครคอนโทรลเลอร์ MCS-51

- การทดลองแยกแต่ละวงจร

ทำการทดลองกับวงจรในส่วนต่าง ๆ ดังที่ได้กล่าวมาในบทที่ 3 ของแต่ละวงจร ดังนี้

4.1 การทดลองส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก มีดังนี้

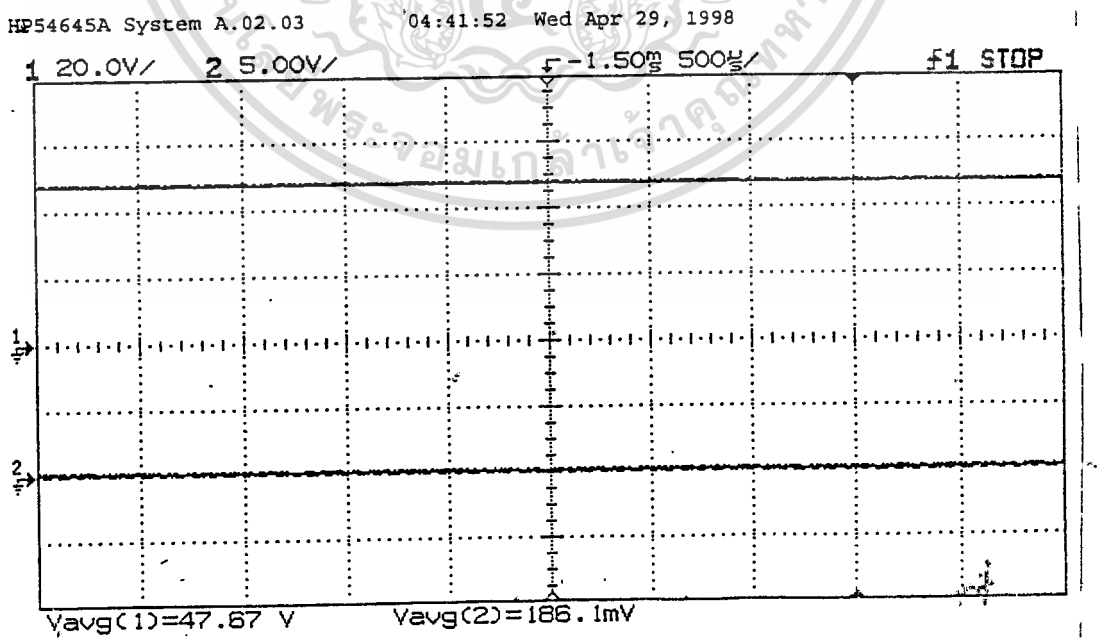
4.1.1 วัดสัญญาณไฟ DC คร่อมคู่สายภายนอกนอกขณะที่วางหู (ch1) หรือไม่มีสัญญาณสั่งให้ยกหู (ch2) ดังรูปที่ 4.1

4.1.2 วัดสัญญาณไฟ DC คร่อมคู่สายภายนอก (ch1) ขณะที่ส่วนควบคุมสั่งให้มีการยกหู โดยส่งลอจิก "1"ไปที่ ขา CTHK (ch2) ดังรูปที่ 4.2

4.1.3 วัดสัญญาณแสดงสถานะการยกหูของสายนอกสายที่ 1 ได้ลอจิก "0" (ch1) และสายนอกสายที่ 2 ซึ่งวางหู ได้ลอจิก "1" (ch2) โดยวัดที่ขา STHK ดังรูปที่ 4.3

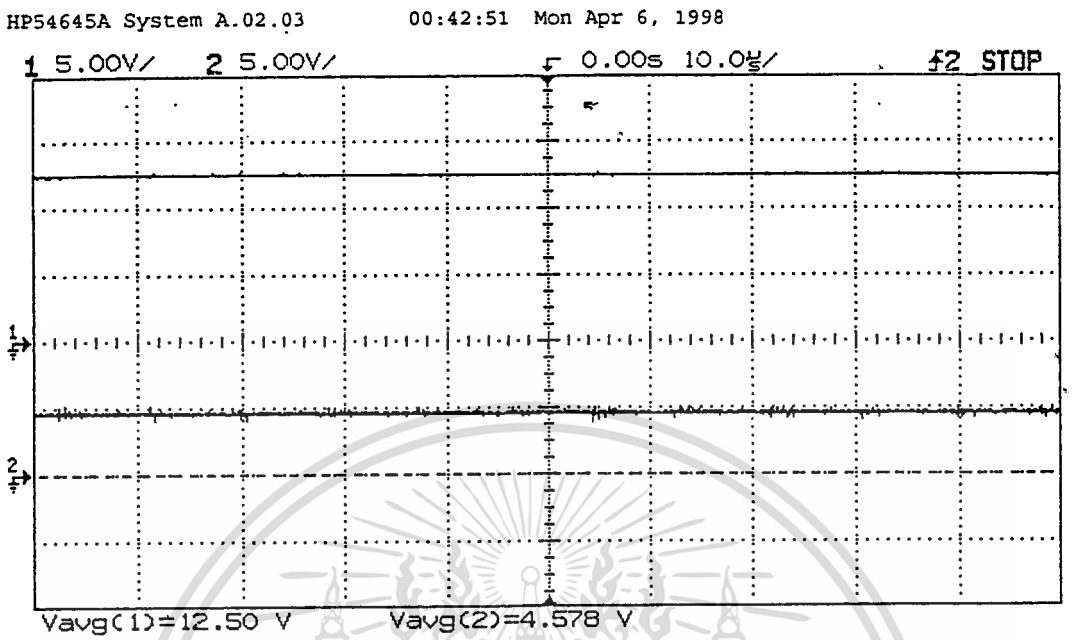
4.1.4 ส่งสัญญาณคลื่นสี่เหลี่ยมความถี่ 1.2 kHz เข้ามาจากคู่สายภายนอกแล้ววัดสัญญาณเอาท์พุทที่ได้เมื่อผ่านส่วนนี้ (ด้านเซคันดารีของหม้อแปลงอิมพิแดนท์ 600:600) ดังรูปที่ 4.4

4.1.5 ส่งสัญญาณ DTMF หมายเลข 1 เข้ามาจากคู่สายภายนอกแล้ววัดเอาท์พุทที่จะไปเข้าส่วนตรวจจับสัญญาณ DTMF ดังรูปที่ 4.5

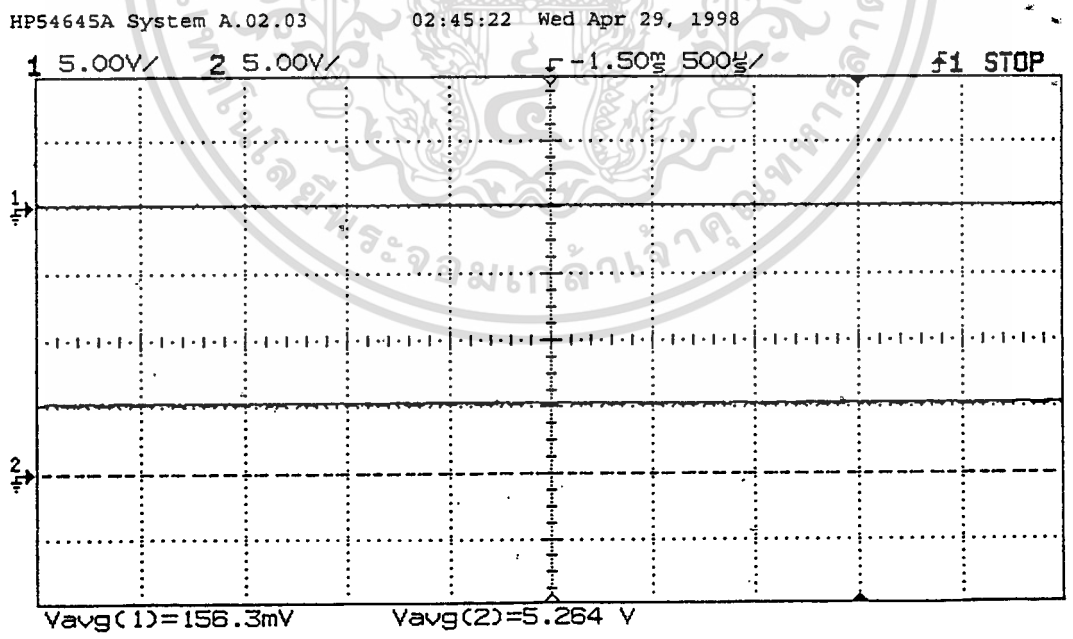


รูปที่ 4.1 สัญญาณไฟตรงคร่อมคู่สายภายนอกขณะวางหู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

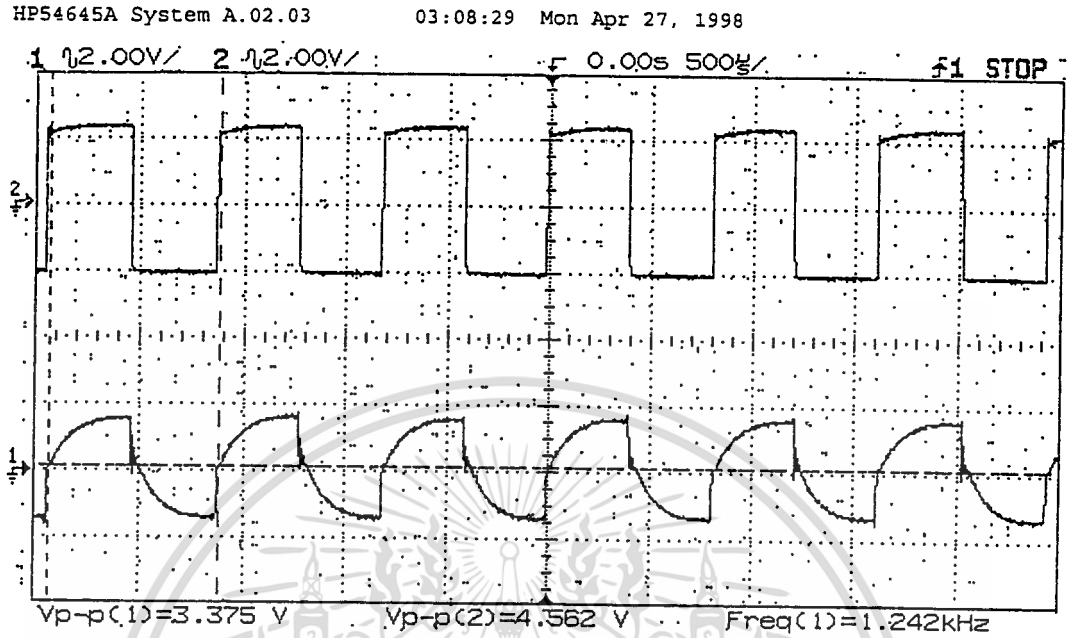


รูปที่ 4.2 สัญญาณไฟตรงคร่อมคู่สายภายนอกขณะยกหู

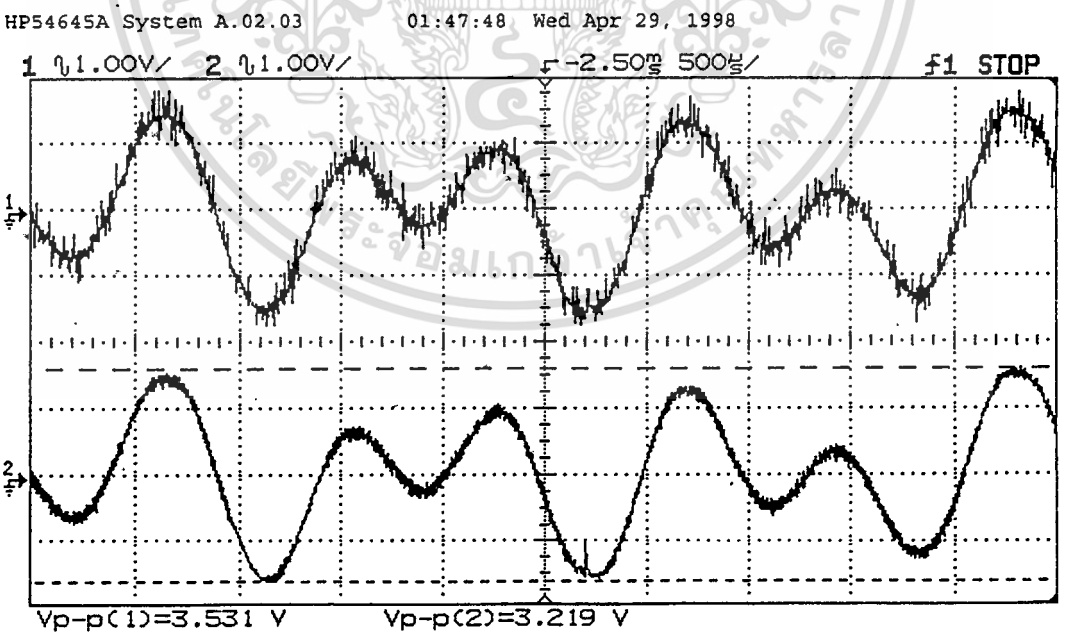


รูปที่ 4.3 สัญญาณแสดงสถานะการยกหู-วางหูของคู่สายภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



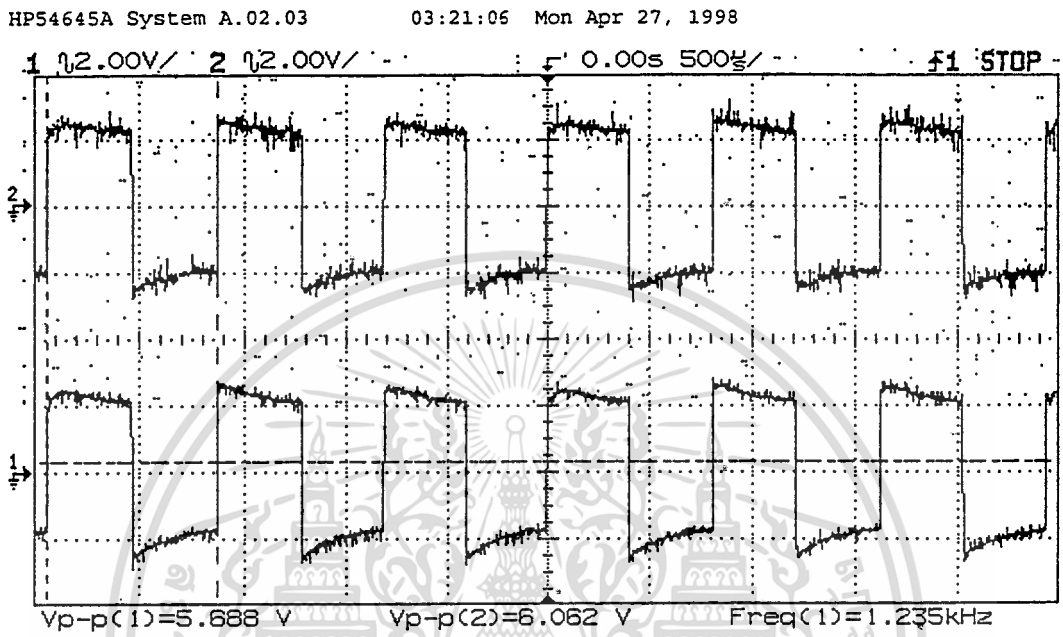
รูปที่ 4.4 สัญญาณคลื่นสี่เหลี่ยมที่ส่งเข้าส่วนอินเตอร์เฟสคู่สายภายนอกกับสัญญาณเอาท์พุทที่ออกจากส่วนอินเตอร์เฟสคู่สายภายนอก



รูปที่ 4.5 สัญญาณ DTMF ที่มาจากคู่สายภายนอกกับสัญญาณ DTMF ที่ออกจากส่วนอินเตอร์เฟสคู่สายภายนอกไปส่วนถอดรหัสสัญญาณ DTMF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 การทดลองส่วนครอสพอยท์สวิทช์ จะทำการส่งสัญญาณคลื่นสี่เหลี่ยมความถี่ 1.2 kHz เข้าไปทางขาอินพุทขาใดขาหนึ่งของไอซี 142100 (ch1) แล้ววัดสัญญาณเอาท์พุทอีกด้านหนึ่ง (ch2) เมื่อมีการสวิทช์เกิดขึ้น ดังรูปที่ 4.6



รูปที่ 4.6 สัญญาณคลื่นสี่เหลี่ยมที่ส่งเข้าและออกจากส่วนครอสพอยท์สวิทช์ เมื่อมีการสวิทช์เกิดขึ้น

4.3 การทดลองส่วนตรวจจับสัญญาณเสียง มีดังนี้

4.3.1 วัดสัญญาณเรียกกลับ (Ring back tone) และสัญญาณไม่ว่าง (Busy tone) จะเป็นสัญญาณที่มอดูเลชั่นกันมาระหว่างสัญญาณคลื่นไซน์ความถี่ 400 Hz และความถี่ 50 Hz ดังรูปที่ 4.7

4.3.2 วัดสัญญาณเรียกกลับ (ch1) เทียบกับเอาท์พุทของไอซี NE567 ที่แสดงการตรวจพบสัญญาณที่มีความถี่ 400 Hz โดยวัดที่ขา 8 (ch2) โดยเมื่อพบสัญญาณจะเปลี่ยนจากสภาวะปกติคือลอจิก "1"เป็นลอจิก "0" ดังรูปที่ 4.8

4.3.3 วัดสัญญาณไม่ว่าง (ch1) เทียบกับเอาท์พุทของไอซี NE567 ที่แสดงการตรวจพบสัญญาณที่มีความถี่ 400 Hz โดยวัดที่ขา 8 (ch2) โดยเมื่อพบสัญญาณจะเปลี่ยนจากสภาวะปกติคือลอจิก "1"เป็นลอจิก "0" ดังรูปที่ 4.9

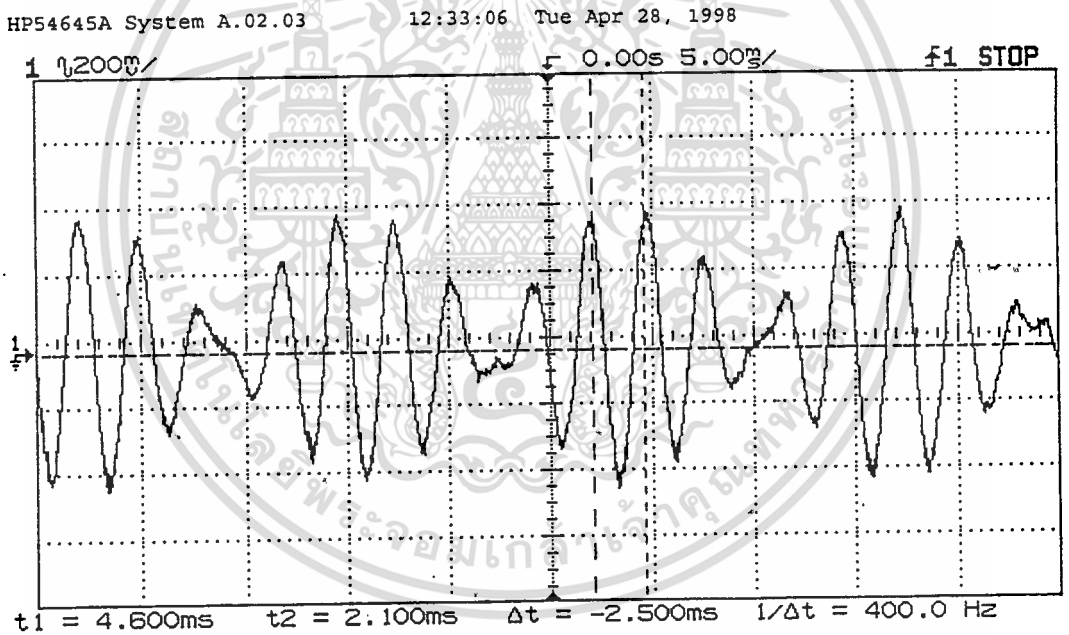
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.3.4 วัดสัญญาณที่แสดงการตรวจพบสัญญาณกระดิ่ง (Ringing signal) (ch2) โดยจะเป็นลอจิก "0" เทียบกับสัญญาณพัลส์ที่ส่งไปอินเตอร์รัพท์ส่วนควบคุมเพื่อสั่งให้ยกหูสายนอกสายที่ 1 จะเป็นพัลส์ที่มีความกว้างประมาณ 1 msec (ch1) ดังรูปที่ 4.10

4.3.5 วัดสัญญาณที่ส่งไปอินเตอร์รัพท์เมื่อพบสัญญาณกระดิ่งจากสายนอกสายที่ 1 (ch1) กับสัญญาณสั่งให้มีการยกหู (ch2) ดังรูปที่ 4.11

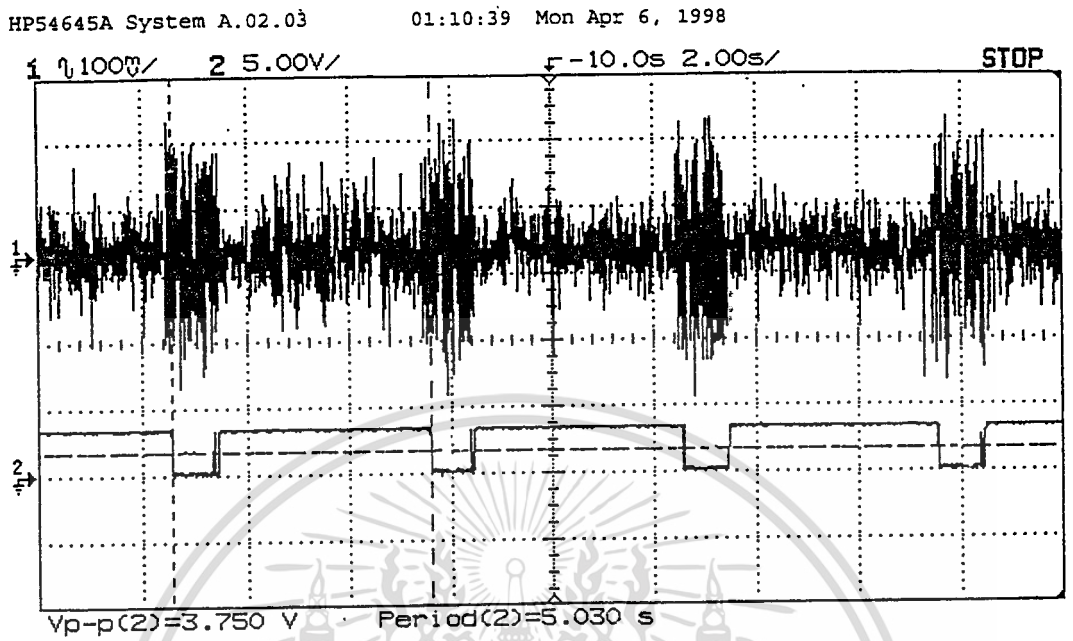
4.3.6 วัดสัญญาณที่แสดงการตรวจพบสัญญาณกระดิ่ง (Ringing signal) (ch2) โดยจะเป็นลอจิก "0" เทียบกับสัญญาณพัลส์ที่ส่งไปอินเตอร์รัพท์ส่วนควบคุมเพื่อสั่งให้ยกหูสายนอกสายที่ 2 จะเป็นพัลส์ที่มีความกว้างประมาณ 0.2 msec (ch1) ดังรูปที่ 4.12

4.3.7 วัดสัญญาณที่ส่งไปอินเตอร์รัพท์เมื่อพบสัญญาณกระดิ่งจากสายนอกสายที่ 2 (ch1) กับสัญญาณสั่งให้มีการยกหู (ch2) ดังรูปที่ 4.13

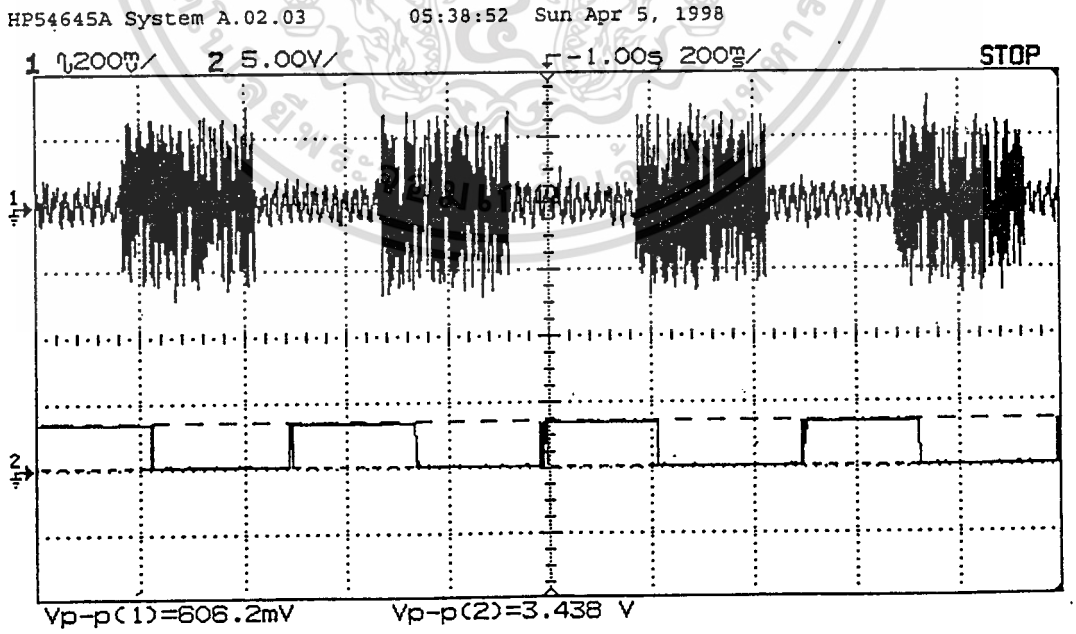


รูปที่ 4.7 สัญญาณโทนหรือสัญญาณเรียกกลับและสัญญาณไม่ว่าง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

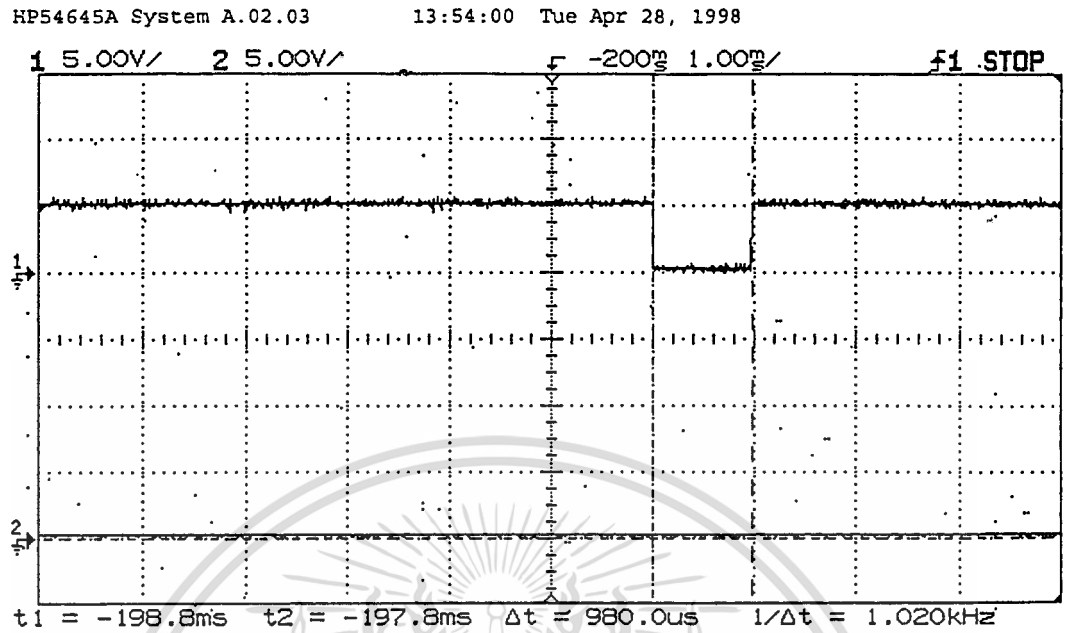


รูปที่ 4.8 สัญญาณเรียกกลับและสัญญาณแสดงการตรวจพบสัญญาณโทนความถี่ 400 Hz

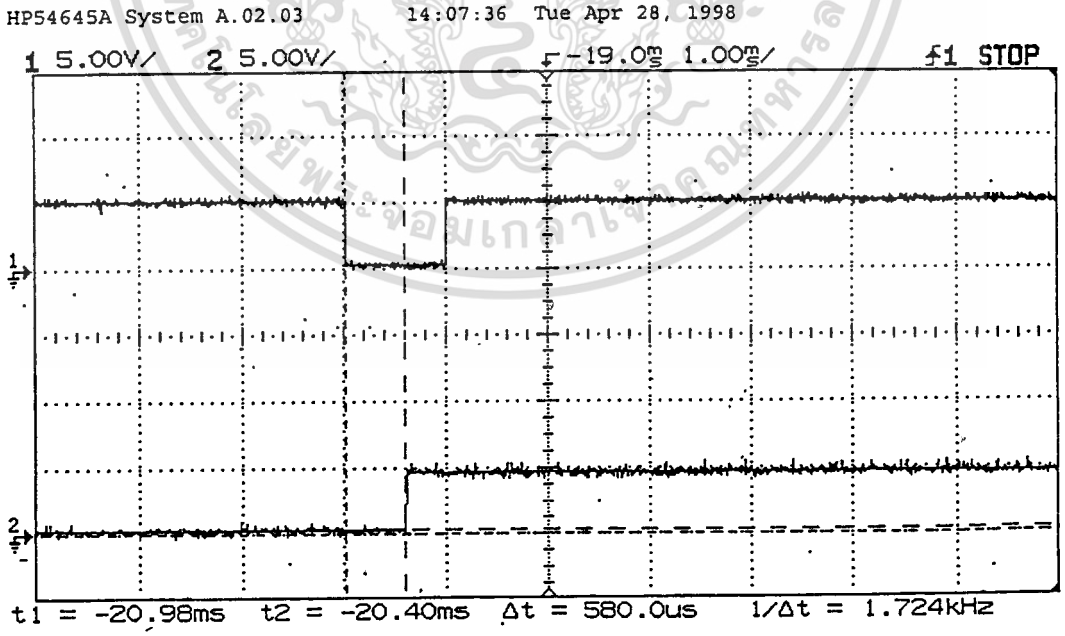


รูปที่ 4.9 สัญญาณไม่ว่างและสัญญาณแสดงการตรวจพบสัญญาณโทนความถี่ 400 Hz

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

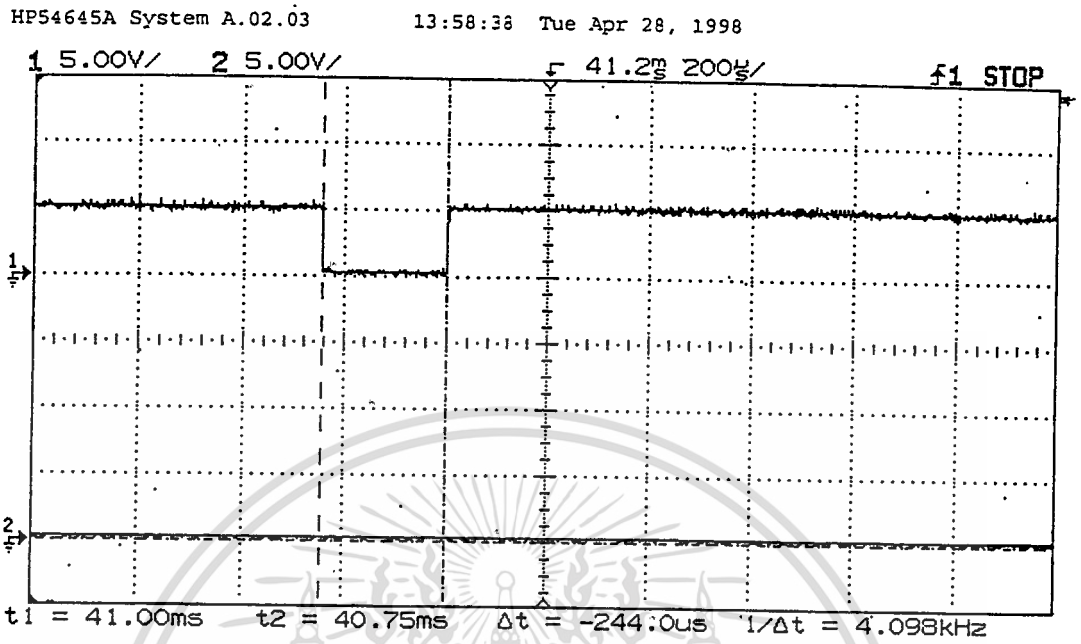


รูปที่ 4.10 สัญญาณอินเตอร์รัพท์และสัญญาณตรวจพบสัญญาณกระดิ่งของสายนอกสายที่ 1

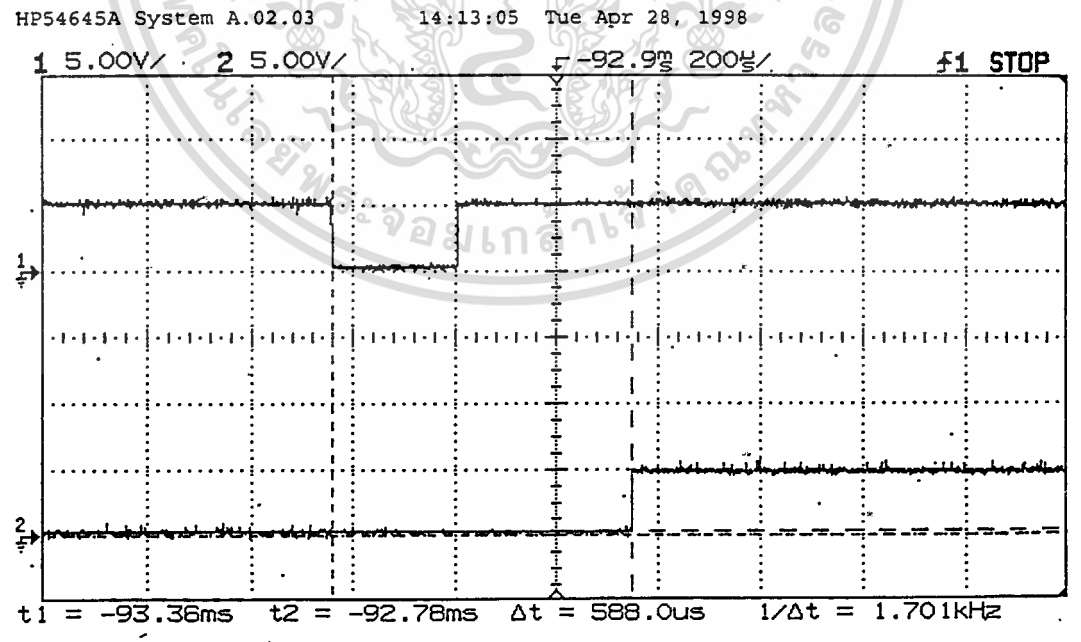


รูปที่ 4.11 สัญญาณอินเตอร์รัพท์และสัญญาณสั่งให้ยกหูของสายนอกสายที่ 1

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.12 สัญญาณอินเตอร์รัพท์และสัญญาณตรวจพบสัญญาณกระดิ่งของสายนอกสายที่ 2



รูปที่ 4.13 สัญญาณอินเตอร์รัพท์และสัญญาณสั่งให้ยกหูของสายนอกสายที่ 2

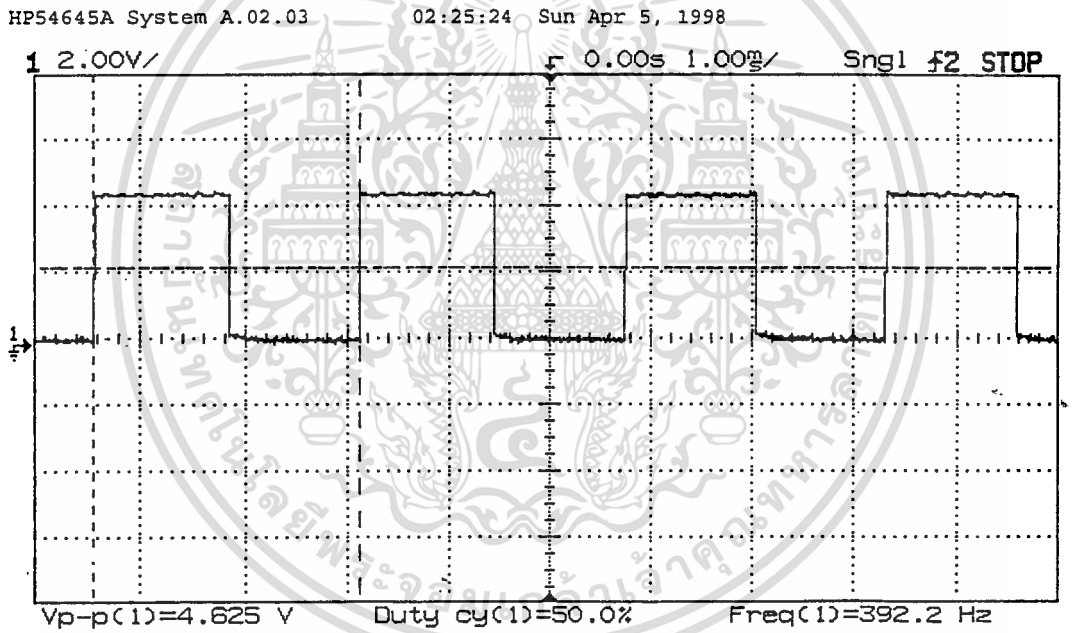
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.4 การทดลองส่วนกำเนิดสัญญาณเสียง มีดังนี้

4.4.1 วัดสัญญาณพร้อมกดเลขหมาย (Dial tone) ที่สร้างมาจากไอซี 555 ตัวที่ 2 ซึ่งเป็นสัญญาณคลื่นสี่เหลี่ยมมีความถี่ประมาณ 400 Hz ดังรูปที่ 4.14

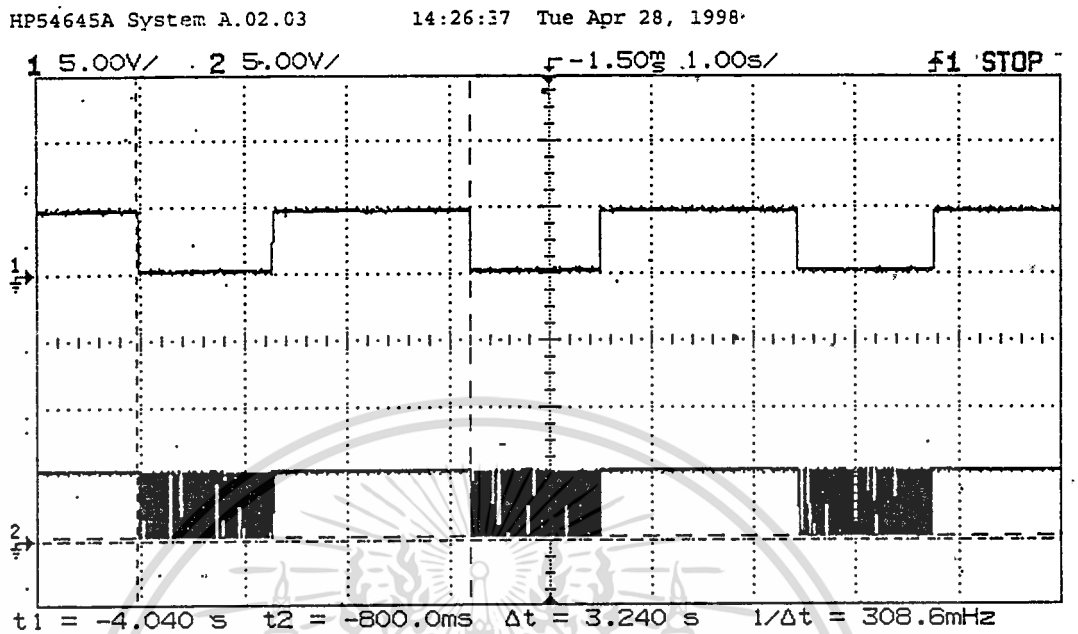
4.4.2 วัดสัญญาณควบคุมช่วงเวลาดัง-หยุดของสัญญาณเรียกกลับ หรือขาเอาท์พุทของไอซี 555 ตัวที่ 1 (ch1) ซึ่งจะควบคุมให้สัญญาณคลื่นสี่เหลี่ยมความถี่ 400 Hz ดังหยุดตามช่วงเวลาที่ตั้งไว้ และนำไปใช้เป็นสัญญาณเรียกกลับ (ch2) ดังรูปที่ 4.15

4.4.3 วัดสัญญาณควบคุมช่วงเวลาดัง-หยุดของสัญญาณไม่ว่าง หรือขาเอาท์พุทของไอซี 555 ตัวที่ 3 (ch1) ซึ่งจะควบคุมให้สัญญาณคลื่นสี่เหลี่ยมความถี่ 400 Hz ดังหยุดตามช่วงเวลาที่ตั้งไว้ และนำไปใช้เป็นสัญญาณไม่ว่าง (ch2) ดังรูปที่ 4.16

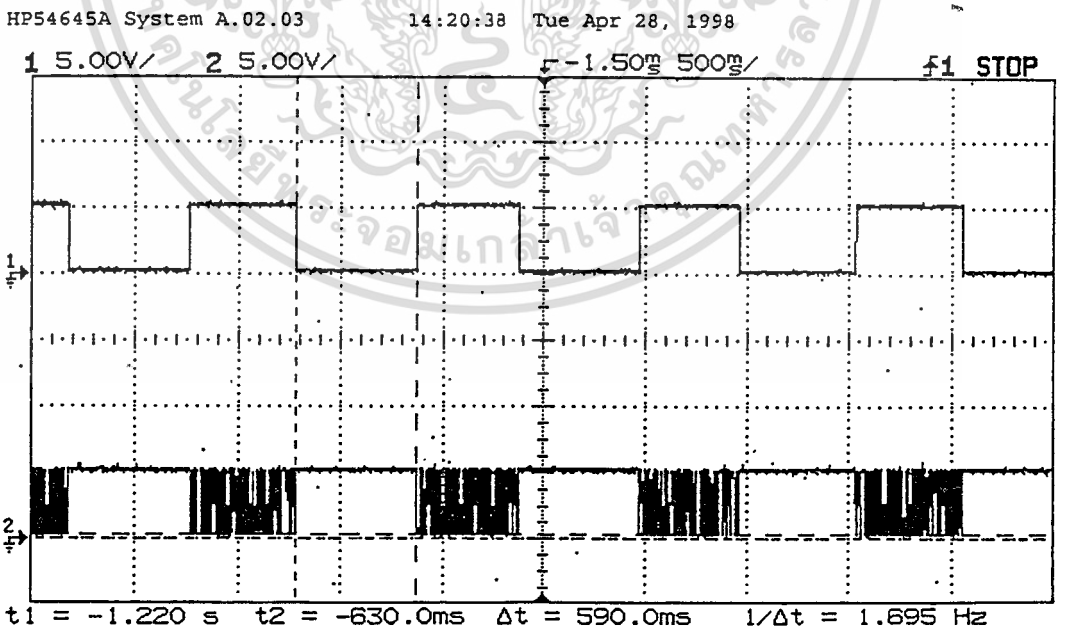


รูปที่ 4.14 สัญญาณพร้อมกดเลขหมาย (Dial tone)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.15 สัญญาณเรียกกลับ (Ring back tone)



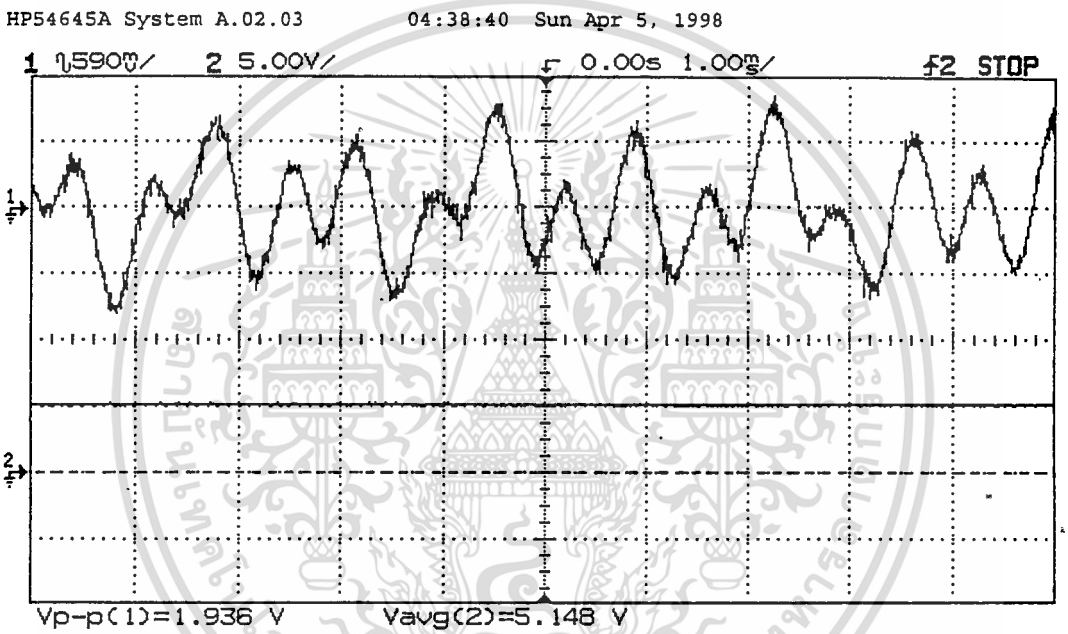
รูปที่ 4.16 สัญญาณไม่ว่าง (Busy tone)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.5 การทดลองส่วนถอดรหัสสัญญาณ DTMF มีดังนี้

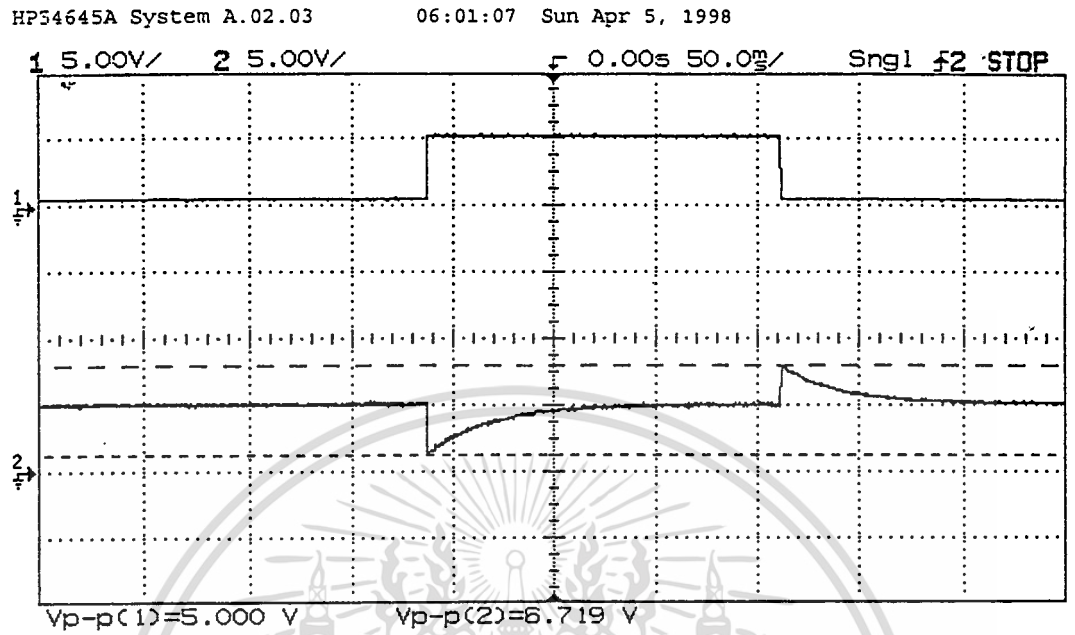
4.5.1 วัดสัญญาณ DTMF หมายเลข "1" (ch1) เทียบกับสัญญาณที่ขา STD ของไอซี MT8870 ซึ่งเป็นสัญญาณแสดงการตรวจพบสัญญาณ DTMF (ch2) จะเห็นว่าเมื่อมีการกดเลขหมายสัญญาณที่ขา STD จะเป็นลอจิก "1" ดังรูปที่ 4.17

4.5.2 วัดสัญญาณที่ขา STD (ch1) เทียบกับสัญญาณที่นำไปอินเตอร์รัพท์ส่วนควบคุมระบบเพื่อบอกว่ามีการกดเลขหมายแล้ว (ch2) ซึ่งสัญญาณอินเตอร์รัพท์นี้จะทำงานที่พัลส์โลว์ ดังรูปที่ 4.18



รูปที่ 4.17 สัญญาณ DTMF หมายเลข "1" และสัญญาณแสดงการตรวจพบ DTMF ที่ขา STD

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.18 สัญญาณแสดงการตรวจพบ DTMF และสัญญาณอินเตอร์รัพท์

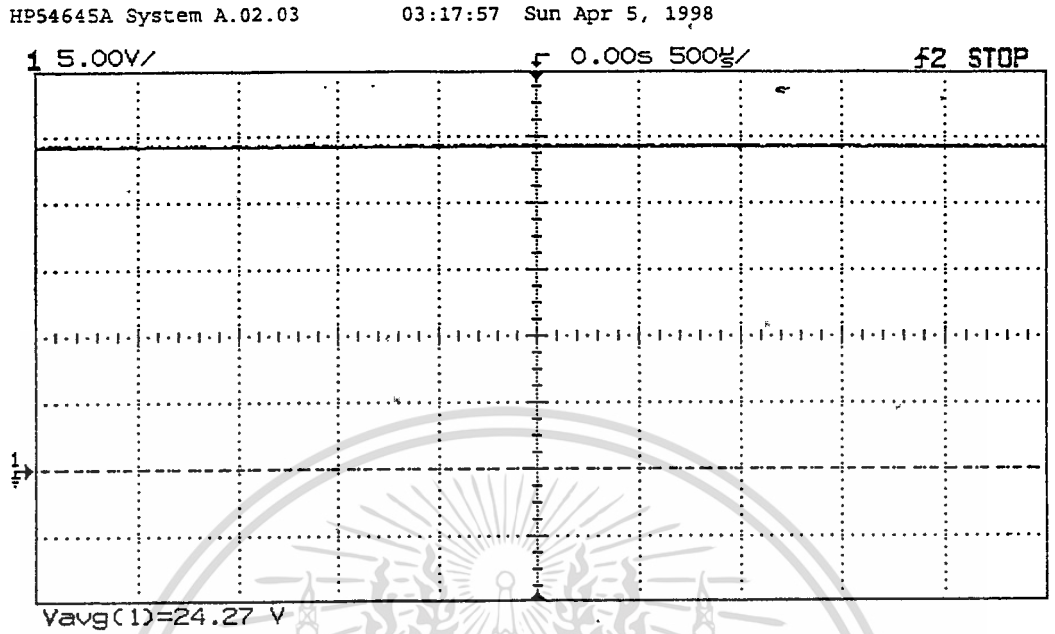
4.6 การทดลองส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน

4.6.1 วัดสัญญาณไฟ DC ที่คร่อมคู่สายภายในขณะที่วางหูอยู่ ซึ่งจะมีค่า 24 V ดังรูปที่ 4.19

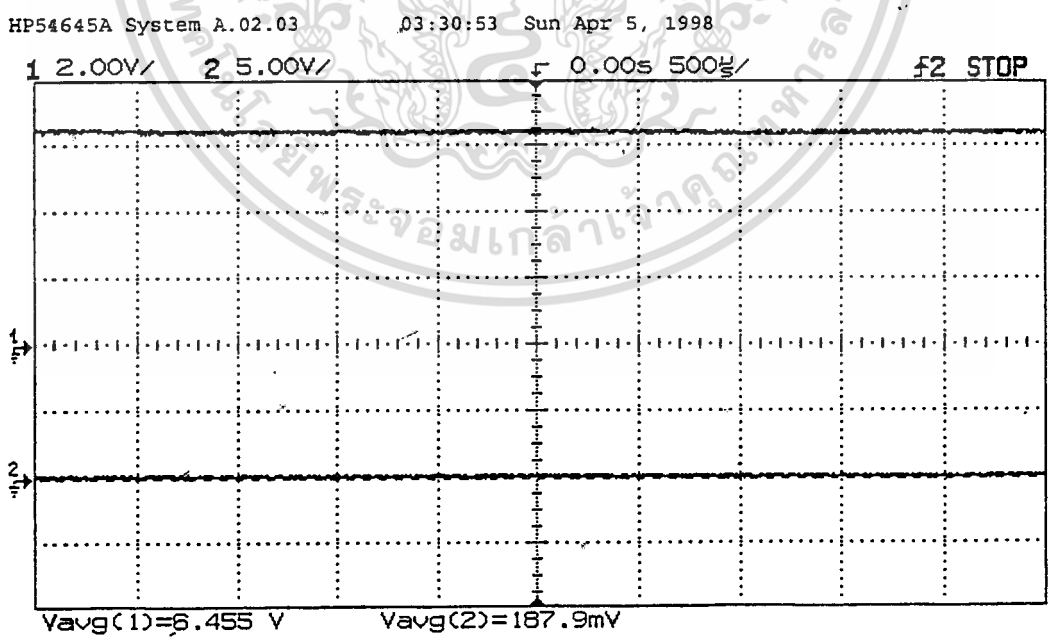
4.6.2 วัดสัญญาณไฟ DC ที่คร่อมคู่สายภายในขณะที่ยกหู (ch1) ซึ่งจะมีค่าประมาณ 6 V และวัดสัญญาณแสดงสถานะการยกหู-วางหูของคู่สายใน (SEHK:ch2) ซึ่งจะมีค่าเป็นลอจิก "0" เมื่อมีการยกหูขึ้น ดังรูปที่ 4.20

4.6.3 วัดสัญญาณ CRG หรือสัญญาณควบคุมการส่งสัญญาณกระดิ่งไปที่คู่สายภายใน (ch1) เทียบกับสัญญาณกระดิ่งที่วัดที่คู่สายภายใน (ch2) จะเห็นว่าเมื่อสัญญาณควบคุมเป็นลอจิก "1" สัญญาณกระดิ่งจะถูกส่งไปที่คู่สายภายในในช่วงเวลาเดียวกับสัญญาณเรียกกลับ ดังรูปที่ 4.21

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

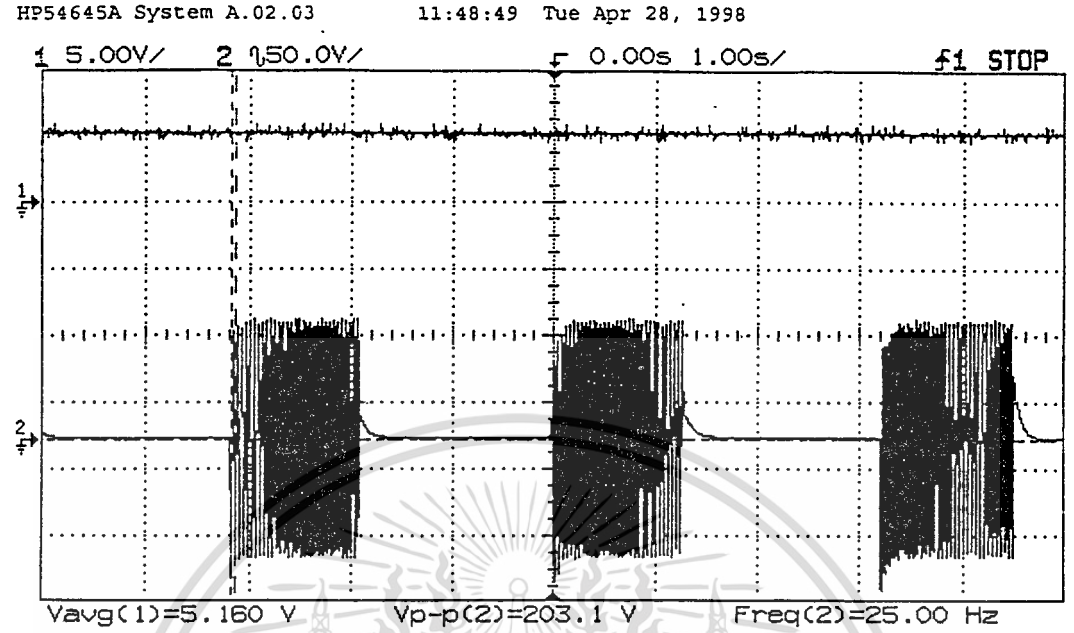


รูปที่ 4.19 สัญญาณไฟตรงคร่อมคู่สายภายในขณะวางหู



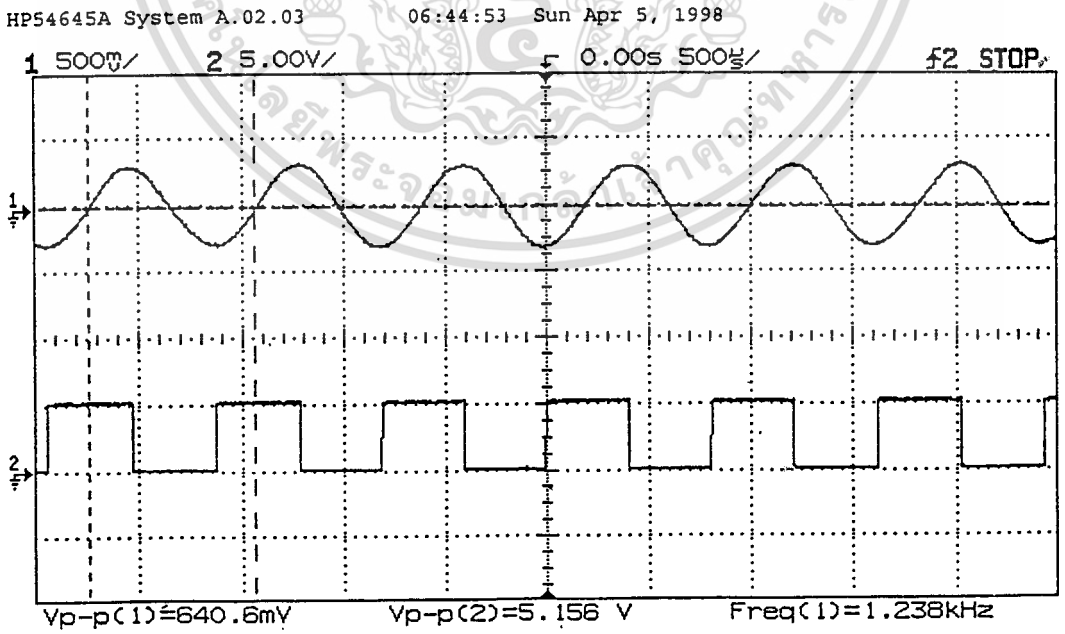
รูปที่ 4.20 สัญญาณไฟตรงคร่อมคู่สายภายในขณะยกหูและสัญญาณแสดงสถานะการยกหู

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.21 สัญญาณควบคุมการส่งสัญญาณกระดิ่งและสัญญาณกระดิ่ง

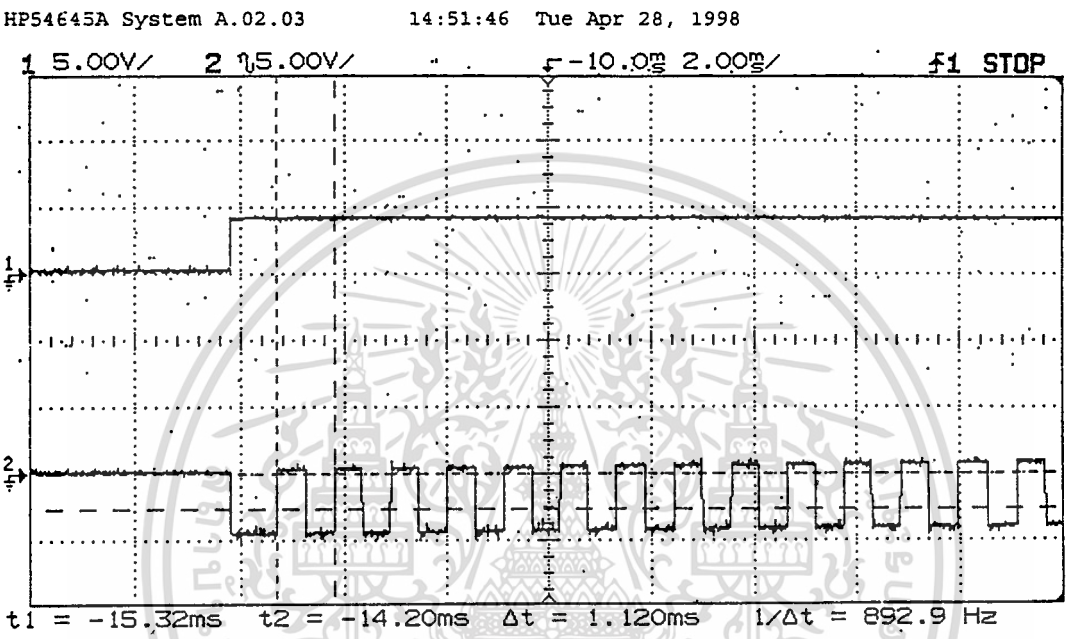
4.7 การทดลองส่วนตอบรับอัตโนมัติ จะทำการบันทึกสัญญาณคลื่นสี่เหลี่ยมความถี่ 1.2 kHz (ch1) เข้าไปในไอซีบันทึกเสียง ISD1420 แล้วทำการเพลย์สัญญาณที่บันทึกออกมา (ch2) ดังรูปที่ 4.22



รูปที่ 4.22 สัญญาณคลื่นสี่เหลี่ยมที่บันทึกและเล่นกลับออกมาจากไอซี ISD1420

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.8 การทดลองส่วนเข้ารหัสการกดคีย์บอร์ด จะทำการวัดสัญญาณ DA (Data available) ของไอซี 74C922 ซึ่งเป็นขาที่แสดงว่ามีการกดคีย์บอร์ดเกิดขึ้น (ch1) และวัดสัญญาณเสียงที่สร้างจากไอซี555 เพื่อแสดงว่ามีการกดคีย์บอร์ด (ch2) จะเห็นว่าเมื่อขา DA เป็นลอจิก "1" หรือมีการกดคีย์ จะไปทำให้ไอซี 55 ทำงาน เกิดเสียงขึ้นความถี่ประมาณ 900 Hz ดังรูปที่ 4.23



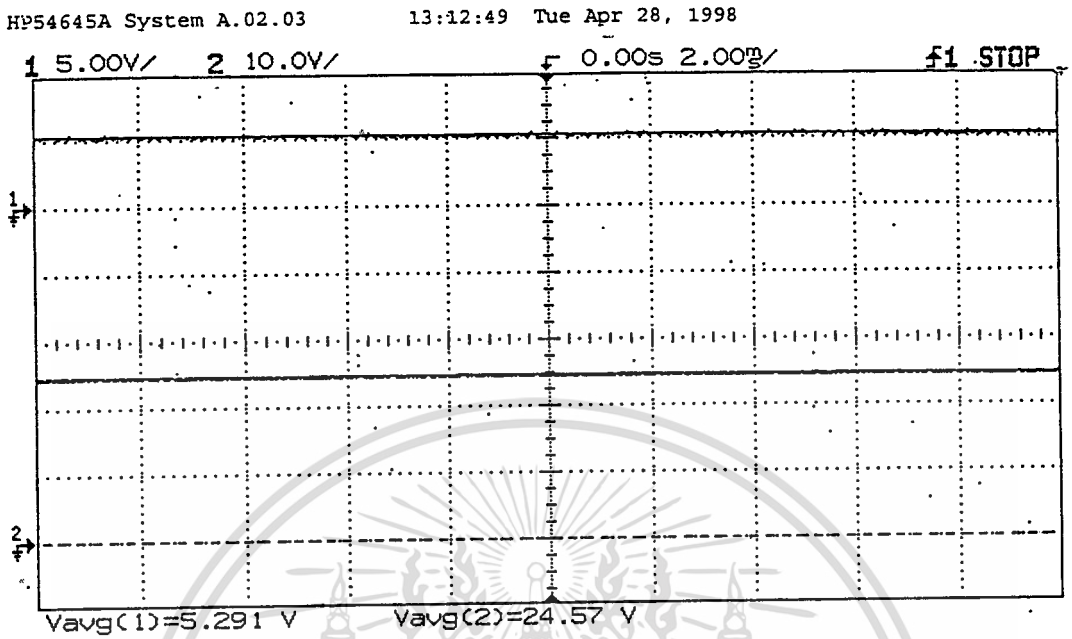
รูปที่ 4.23 สัญญาณจากขา DA ของไอซี 74C922 และสัญญาณเสียงของการกดคีย์บอร์ด

4.9 การทดลองส่วนแหล่งจ่ายไฟเลี้ยง มีดังนี้

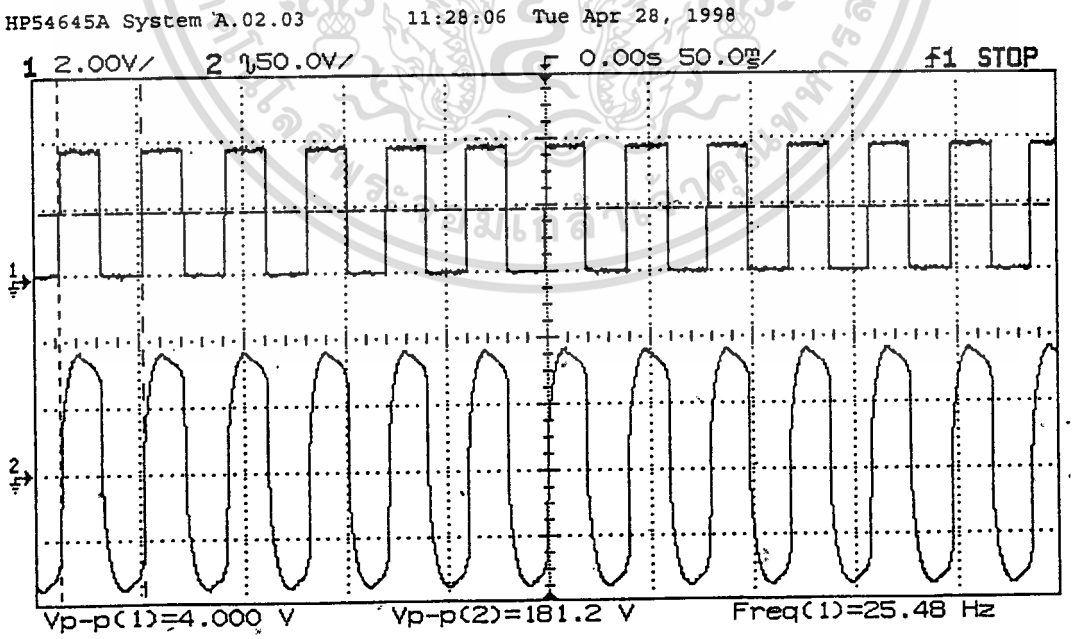
4.9.1 วัดสัญญาณไฟ DC ที่สร้างขึ้น ได้แก่ 5V และ 24V ดังรูปที่ 4.24

4.9.2 วัดสัญญาณเอาท์พุทจากไอซี 555 (ch1) ซึ่งนำไปใช้ในการสร้างสัญญาณกระดิ่ง ซึ่งเป็นคลื่นสี่เหลี่ยมที่มีความถี่ 25 Hz เทียบกับสัญญาณกระดิ่งซึ่งเป็นสัญญาณกระแสสลับความถี่ 25 Hz เช่นกัน และมีแอมปลิจูด ประมาณ 90 โวลท์ พีค ทู พีค (100 Vp-p) ดังรูปที่ 4.25

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.24 สัญญาณไฟตรง 5V และ 24V จากแหล่งจ่ายไฟเลี้ยง



รูปที่ 4.25 สัญญาณคลื่นสี่เหลี่ยมที่สร้างจากไอซี 555 และสัญญาณกระดิ่ง

เอกสารนี้เป็นเอกสาร ... ใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

- การทดลองโดยรวมวงจรทุกวงจรเข้าด้วยกัน แล้วทำการควบคุมการทำงานโดยโปรแกรม

ในส่วนจะเป็นการทดลองโทรเข้า -โทรออก และการใช้ฟังก์ชั่นพิเศษต่าง ๆ เช่นการโอนสาย พักสาย ประชุม 3 สาย และการฝากเลขหมายกรณีที่คู่สายภายในไม่มีคนรับสาย

4.10 ขั้นตอนการทดลองการโทรเข้า

4.10.1 สายนอกส่งสัญญาณกระดิ่งเข้ามาที่เครื่องชุมสาย

4.10.2 เครื่องชุมสายจะทำการยกหูอัตโนมัติ และส่งข้อความ " ที่นี่คณะวิศวกรรมศาสตร์ กรุณากดหมายเลขภายใน "

4.10.3 ผู้ที่โทรเข้ามาจะทำการกดเลขหมายของคู่สายภายในได้แก่ หมายเลข 1-8

4.10.4 เครื่องชุมสายจะส่งสัญญาณกระดิ่งไปคู่สายภายในที่ถูกเรียก และส่งสัญญาณเรียกกลับไปคู่สายภายนอก

4.10.5 เมื่อคู่สายภายในมีผู้รับสาย เครื่องจะตัดต่อครอสพอยท์สวิทช์ให้สามารถสนทนากันได้

4.11 ขั้นตอนการทดลองการโทรออก

4.11.1 คู่สายภายในยกหูโทรศัพท์ เครื่องส่งสัญญาณพร้อมให้กดเลขหมาย

4.11.2 ผู้ใช้กด "9 " เพื่อตัดออกสายนอก เครื่องสั่งให้คู่สายภายในตัดออกสายนอก

4.11.3 ผู้ใช้กดเลขหมายปลายทาง และสนทนากันต่อไป

4.12 ขั้นตอนการโอนสาย

4.12.1 เมื่อมีการโทรเข้ามาของคู่สายภายนอก และสนทนาสายในสายใดสายหนึ่งแล้ว เมื่อสายในต้องการโอนสายก็จะวางหู

4.12.2 ผู้ใช้คู่สายภายในกด "*" ตามด้วยหมายเลขห้องที่ต้องการจะโอนสายนอกไปให้

4.12.3 เครื่องจะส่งสัญญาณกระดิ่งไปคู่สายภายในที่จะโอนสายไป

4.12.4 เมื่อคู่สายในยกหู เครื่องจะโอนสายนอกไปให้คู่สายภายในคู่นั้น

4.13 ขั้นตอนการพักสาย

4.13.1 เมื่อมีการโทรเข้ามาของคู่สายภายนอก และสนทนาสายในสายใดสายหนึ่งแล้ว เมื่อสายในต้องการพักสายก็จะวางหู

4.13.2 เมื่อผู้ใช้คู่สายภายในต้องการจะสนทนากับสายนอกอีกครั้งก็จะกด "# " เครื่องก็จะทำการตัดต่อสายนอกให้อีกครั้ง

4.14 ขั้นตอนการประชุม 3 สาย

4.14.1 เมื่อมีการโทรเข้ามาของคู่สายภายนอก และสนทนาสายในสายใดสายหนึ่งแล้ว เมื่อสายในต้องการประชุม 3 สายก็จะวางหู

4.14.2 ผู้ใช้คู่สายภายในกด "* * " ตามด้วยหมายเลขห้องที่ต้องการจะประชุม 3 สายด้วยอีกสายหนึ่ง

4.14.3 เครื่องจะส่งสัญญาณกระดิ่งไปคู่สายภายในที่จะประชุม 3 สายด้วย เมื่อสายในยกหู เครื่องจะทำการประชุม 3 สายกับคู่สายภายนอกนั้นให้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.15 ขั้นตอนการทดลองการฝากเลขหมาย

4.15.1 เมื่อมีการโทรเข้ามาของคู่สายภายนอก แล้วคู่สายภายในไม่มีคนรับสายภายในเวลา 10 วินาที เครื่องส่งข้อความ " กดหมายเลขภายในเพื่อเรียกอีกครั้ง หรือฝากหมายเลขโทรกลับโดยกด * ตามด้วยหมายเลขและ # "

4.15.2 ผู้ที่โทรเข้ามาสามารถจะเลือกว่าจะเรียกสายในต่อไปโดยกดหมายเลขภายในอีกครั้ง

4.15.3 หากผู้โทรเข้ามาไม่เรียกสายในอีกครั้ง สามารถฝากเลขหมายให้ติดต่อกลับ โดยกด "*" ตามด้วยหมายเลข แล้วจบด้วย "# " เช่น " * 152670766 # "

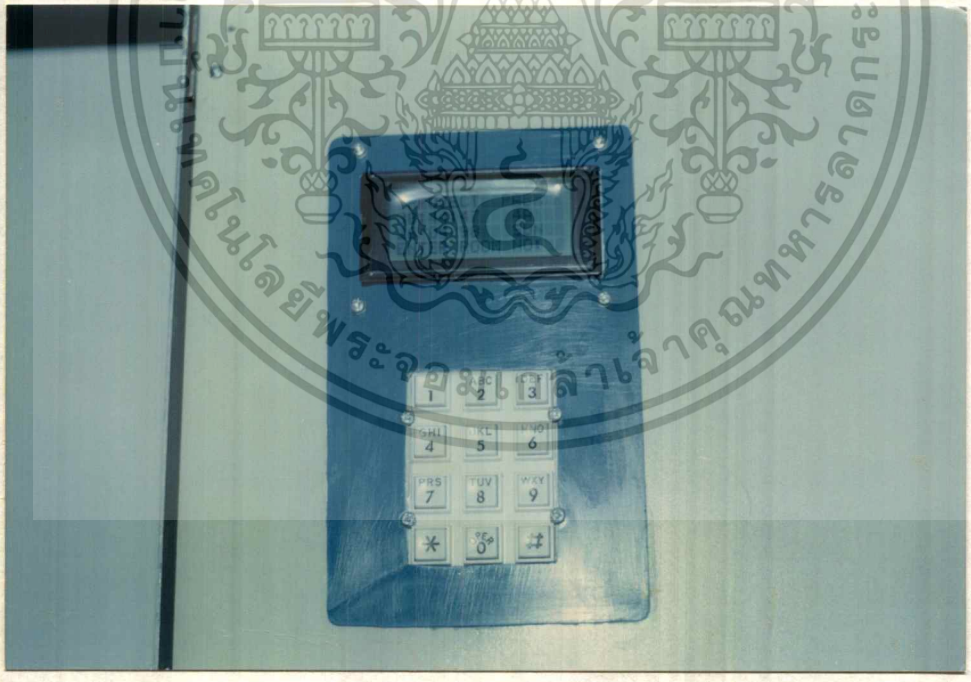
4.16 ขั้นตอนการตรวจสอบเลขหมายติดต่อกลับ

4.16.1 ผู้ใช้คู่สายภายในสามารถตรวจสอบหมายเลขติดต่อกลับจากคู่สายนอกของตน ได้โดยการกดแป้นคีย์ที่ตัวเครื่องชุมสาย

4.16.2 เริ่มต้นด้วยการกดเลขห้องของตัวเอง จะปรากฏหน้าจอเมนูหลัก

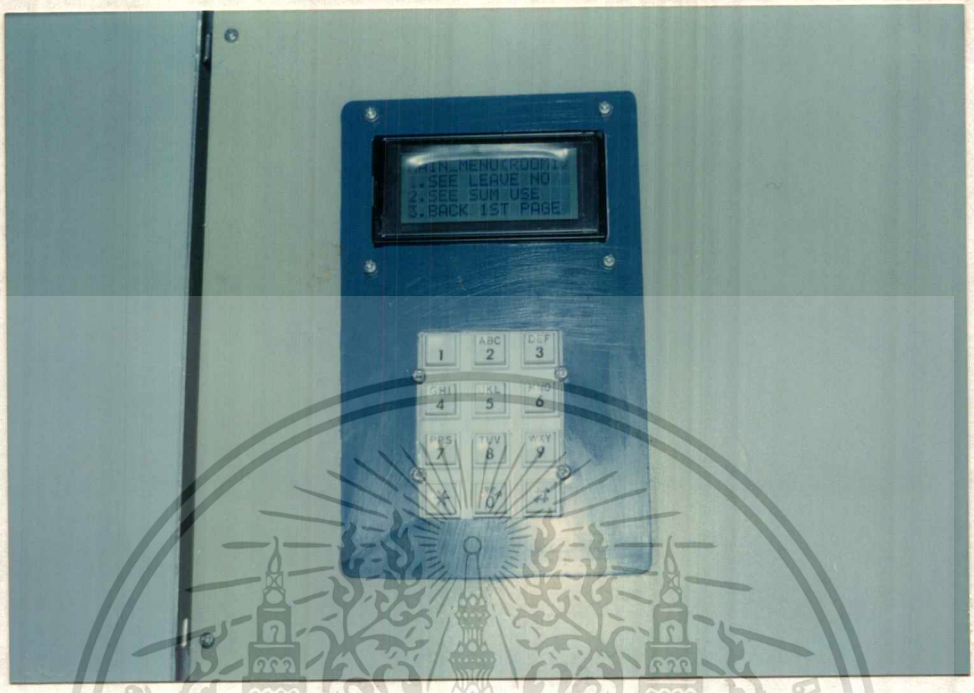
4.16.3 เลือกเมนู "SEE LEAVING NO." จะปรากฏเลขหมายที่ติดต่อกลับ พร้อมทั้งเวลาการฝากเลขหมาย

4.16.4 กดคีย์ "*" หรือ "# " เพื่อดูเลขหมายถัดไปหรือ เลขหมายก่อนหน้า ตามลำดับ

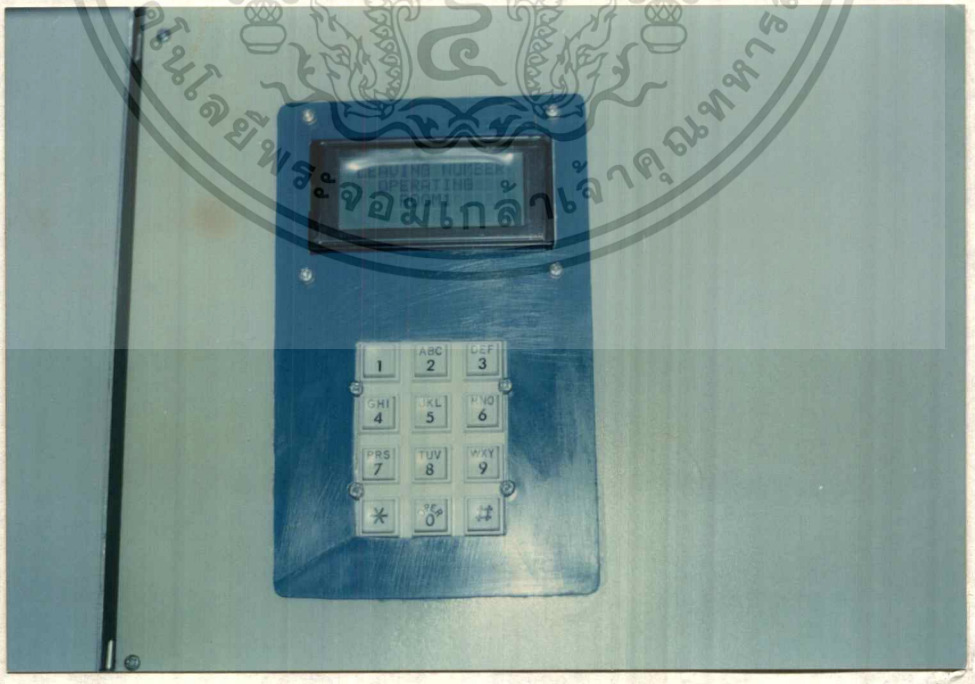


รูปที่ 4.26 หน้าจอ LCD ก่อนเข้าสู่เมนูหลัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

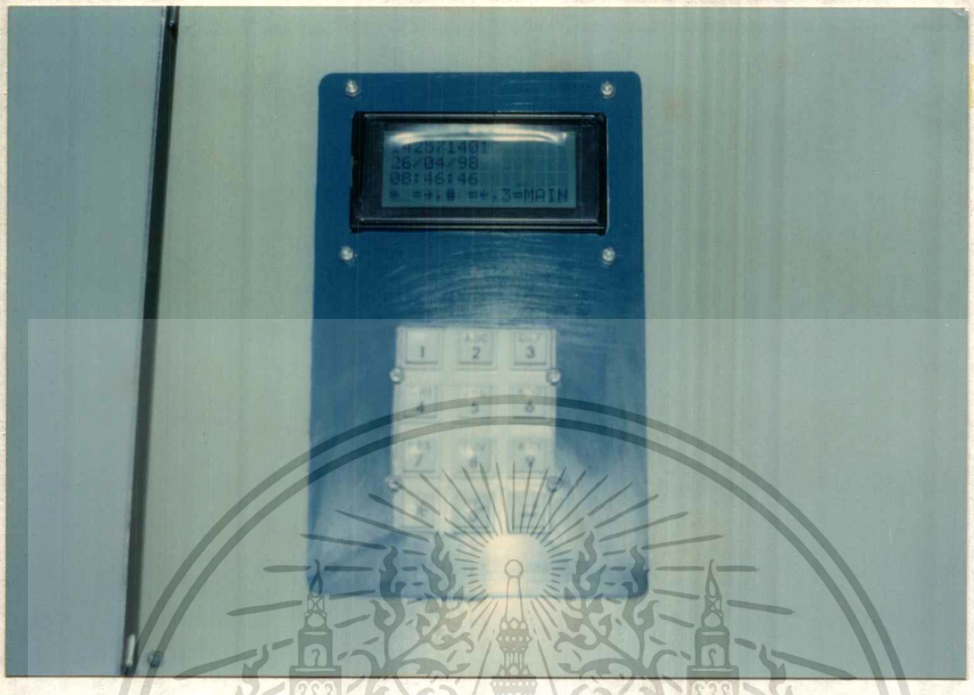


รูปที่ 4.27 หน้าจอ LCD แสดงเมนูหลัก



รูปที่ 4.28 หน้าจอ LCD ขณะที่มีการฝากเลขหมายติดต่อกลับ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



รูปที่ 4.29 หน้าจอ LCD แสดงเลขหมายติดต่อกลับ และเวลาที่ฝากเลขหมาย

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บทที่ 5
บทสรุปและวิจารณ์

5.1 คุณสมบัติของโครงงาน

โครงงานนี้ประกอบไปด้วยส่วนสำคัญ 2 ส่วน คือ ฮาร์ดแวร์ (Hardware) เป็นบอร์ดวงจรส่วนต่าง ๆ ทำหน้าที่แตกต่างกัน ซอฟแวร์ (Software) เป็นโปรแกรมภาษาแอสแซมบลี (Assembly) เขียนขึ้นเพื่อควบคุมการทำงานของโครงงานนี้

5.1.1 ฮาร์ดแวร์ประกอบด้วย

- ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก (TRUNK -LINE INTERFACE CIRCUIT SECTION)
- ส่วนครอสพอยท์สวิทช์ (CROSS POINT SWITCH CIRCUIT SECTION)
- ส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF (TONE DETECTOR AND DTMF DECODER CIRCUIT SECTION)
- ส่วนกำเนิดสัญญาณเสียง(TONE GENERATOR CIRCUIT SECTION)
- ส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน(SUBSCRIBER LINE INTERFACE CIRCUIT SECTION: SLIC)
- ส่วนตอบรับโทรศัพท์อัตโนมัติ (AUTOMATIC ANSWER CIRCUIT SECTION)
- ส่วนจอแสดงผล , ส่วนเข้ารหัสการกดคีย์บอร์ดและส่วนควบคุมการแสดงผล(DOT MATRIX LCD MODULE , KEYBOARD ENCODER AND CONTROL CIRCUIT SECTION#2)
- ส่วนอินพุท/เอาท์พุทพอร์ท (I/O PORT SECTION)
- ส่วนแหล่งจ่ายไฟเลี้ยง (POWER SUPPLY CIRCUIT SECTION)
- ส่วนควบคุมระบบ (CONTROL CIRCUIT SECTION #1)

5.1.2 ซอฟท์แวร์

- โปรแกรมหลักควบคุมการทำงานของวงจรส่วนต่างๆรับที่สัญญาณเข้ามาทำการประมวลผลและส่งสัญญาณไปยังวงจรที่เกี่ยวข้องให้ทำหน้าที่ตามที่ต้องการ
- โปรแกรมควบคุมการทำงานในส่วนจอแสดงผล LCD
- โปรแกรมเชื่อมโยงระหว่างโปรแกรมทั้งสองดังกล่าวข้างต้น

จากการสร้างและการทดลองวงจรส่วนต่างๆของโครงงานนี้ สามารถสรุปคุณสมบัติในการทำงานของเครื่องชุมสายโทรศัพท์สาขาปลายทางอัตโนมัติ ได้ดังนี้

5.1.3 สามารถทำการเชื่อมต่อการสื่อสารระหว่างคู่สายโทรศัพท์ภายนอกจากชุมสายโทรศัพท์ท้องถิ่น 2 คู่สายเข้ากับคู่สายโทรศัพท์ภายในของโครงงานจำนวน 8 คู่สาย

5.1.4 ใช้สัญญาณเสียงอัตโนมัติเพื่อตอบรับการโทรเข้าจากคู่สายภายนอกเพื่อแจ้งสถานที่และ แนะนำให้กดเลือกหมายเลขคู่สายภายใน 8 สาย ที่ต้องการติดต่อ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

5.1.5 ใช้วงจรสร้างสัญญาณเสียงโทนต่างๆเพื่อใช้ในงานชุมสายโทรศัพท์ที่สามารถกำหนดช่วงเวลาของสัญญาณให้เหมาะสมได้ เช่น สัญญาณไม่ว่าง สัญญาณเรียกกลับ และสัญญาณอนุญาตให้กดเลขหมาย

5.1.6 สามารถรับฝากหมายเลขโทรกลับของคู่สายนอกกรณีที่คู่สายในไม่มีผู้รับได้โดยผู้ใช้สามารถกลับมาดูหมายเลขที่ฝากไว้ได้จากการกดคีย์ในส่วนของจอแสดงผลและหมายเลขที่ฝากไว้จะแสดงผลทางจอแสดงผล LCD

5.1.7 สามารถทำงานได้ขณะที่เกิดเหตุขัดข้อง เช่น กระแสไฟฟ้าดับ เนื่องจากในโครงงานได้เพิ่มส่วนวงจรชาร์ทแบตเตอรี่อัตโนมัติ

5.2 ปัญหาและแนวทางแก้ไข

ในการทดสอบการทำงานของวงจรมีปัญหาที่เกิดขึ้นดังนี้

5.2.1 ฮาร์ดแวร์ ส่วน TONE DETECTOR ไม่สามารถตรวจสอบสัญญาณโทนต่างๆได้ เนื่องจากในครั้งแรกที่ทำการทดลองคู่สายนอกต่อกับ กราวด์ 1 เส้น ทำให้เสมือนเกิดการยกหูของคู่สายภายนอก จึงแก้ไขโดยให้รับสัญญาณโทนต่างๆ หลังจากสัญญาณผ่านหม้อแปลงอิมพิแดนซ์แมทชิ่งแล้ว

5.2.2 ในวงจรสร้างสัญญาณกระแสสลับ 100 โวลท์เพื่อใช้เป็นสัญญาณกระดิ่ง ขณะทดสอบเกิดความ เสียหายกับทรานซิสเตอร์ อาจเกิดจากกระแสเกินค่าที่ทรานซิสเตอร์จะทนได้

5.2.3 สัญญาณกระดิ่งที่เข้ามาที่คู่สายภายในในขณะที่มีการเรียกจากคู่สายนอกทำให้เกิดความผิดพลาดของสถานะการยกหูและวางหูของคู่สายภายใน เป็นผลให้ส่วนควบคุมระบบทำงานผิดพลาด แก้ไขโดยการใช้ รีเลย์ตัดสัญญาณกระดิ่งออกจากวงจรส่วนตรวจสอบสถานะการยกหูวางหูของคู่สายภายใน ในขณะที่มีสัญญาณกระดิ่งเข้ามา

5.3 แนวทางการพัฒนาโครงงาน

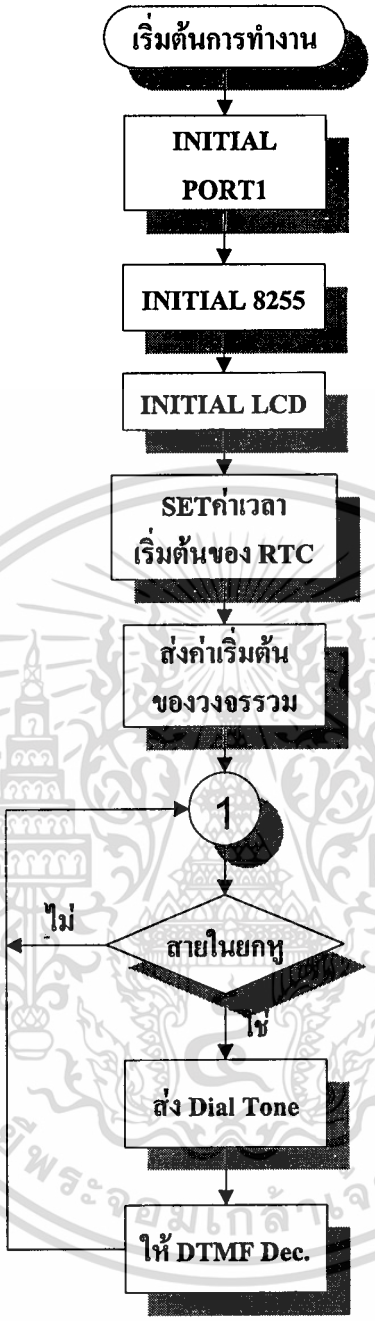
5.3.1 สามารถพัฒนาให้มีการเชื่อมต่อกับไมโครคอมพิวเตอร์เพื่อแสดงผลและควบคุมการทำงานของระบบที่ใหญ่ขึ้น เช่น เพิ่มจำนวนคู่สายภายนอกหรือสายภายในการสั่งพิมพ์สถานะการทำงานต่างๆ ออกทางเครื่องพิมพ์

5.3.2 สามารถพัฒนาให้โปรแกรมมีความยืดหยุ่นมากกว่านี้ ซึ่งอาจจะออกแบบให้มีความสะดวกในการใช้งานมากขึ้น เช่น การส่งสัญญาณเตือนเมื่อมีการฝากข้อความใหม่มาแล้ว หรือ เตือนเมื่อมีการยกหูค้างไว้นานเกินไป และสามารถพัฒนาให้คิดเงินค่าโทรศัพท์ได้

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

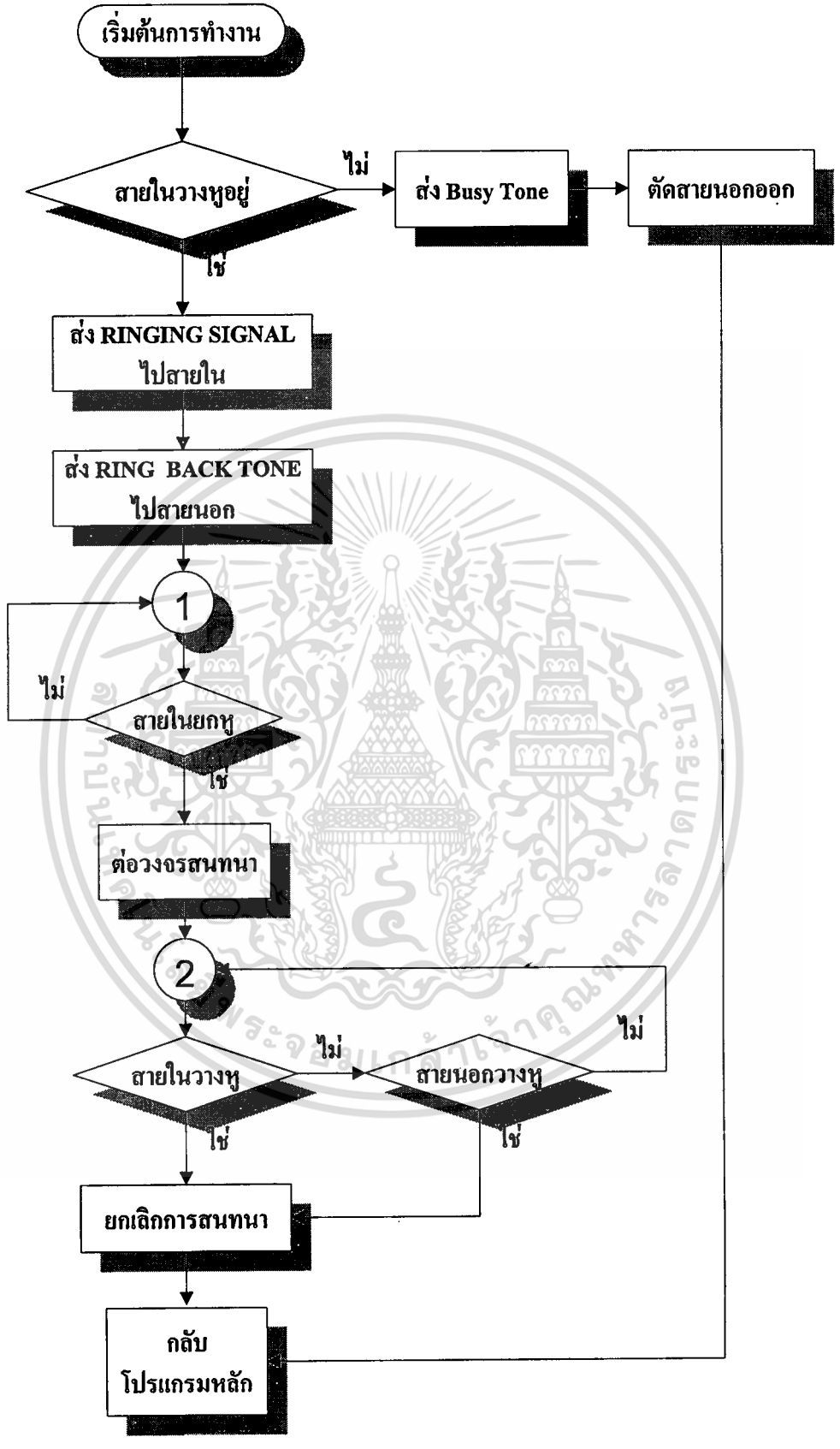
ภาคผนวก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



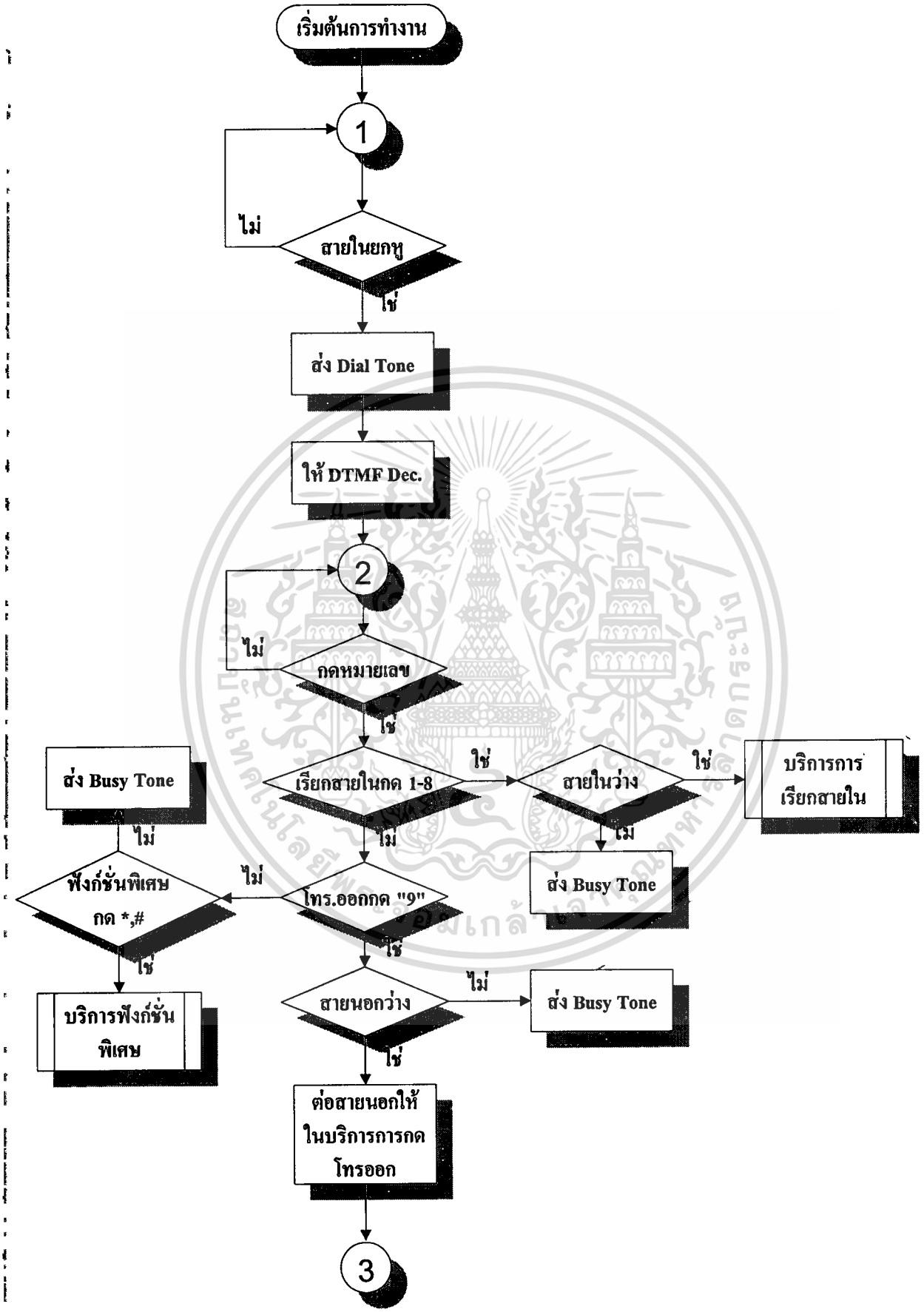
ผังงานแสดงขั้นตอนการทำงานของโปรแกรมหลัก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



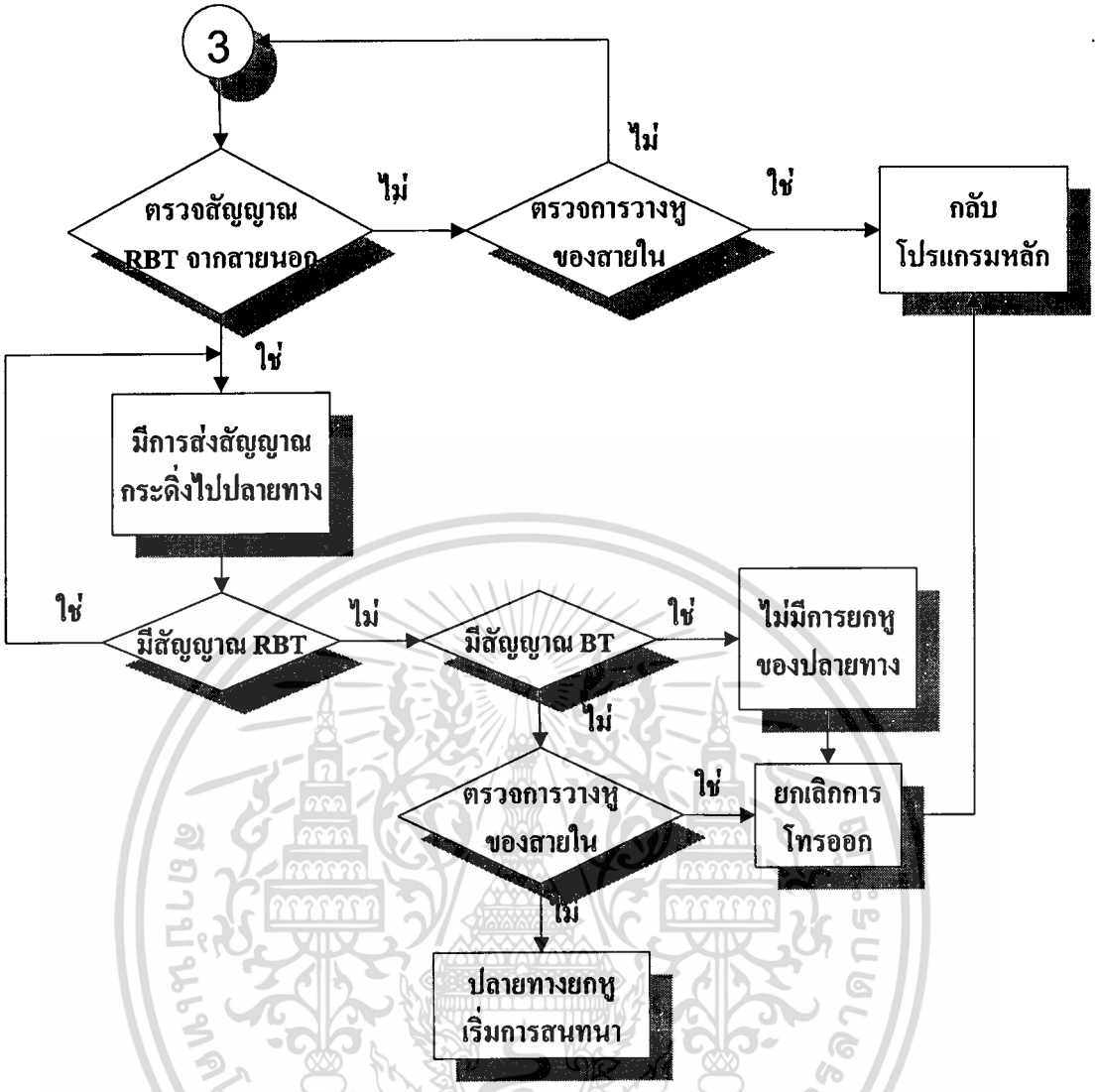
ผังงานแสดงขั้นตอนการโทรเข้ามาของสายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



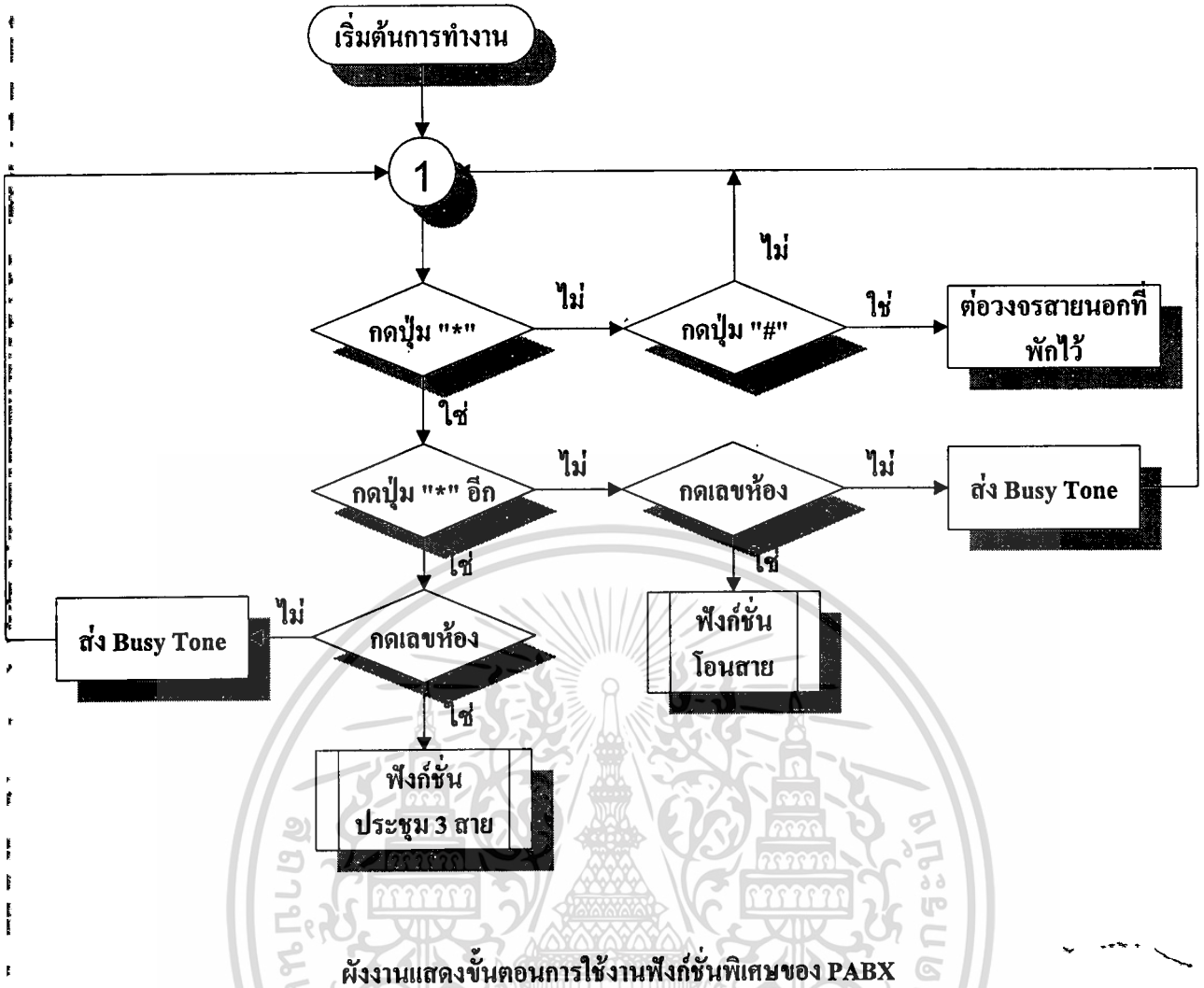
ผังงานแสดงขั้นตอนการโทรออกของสายใน (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



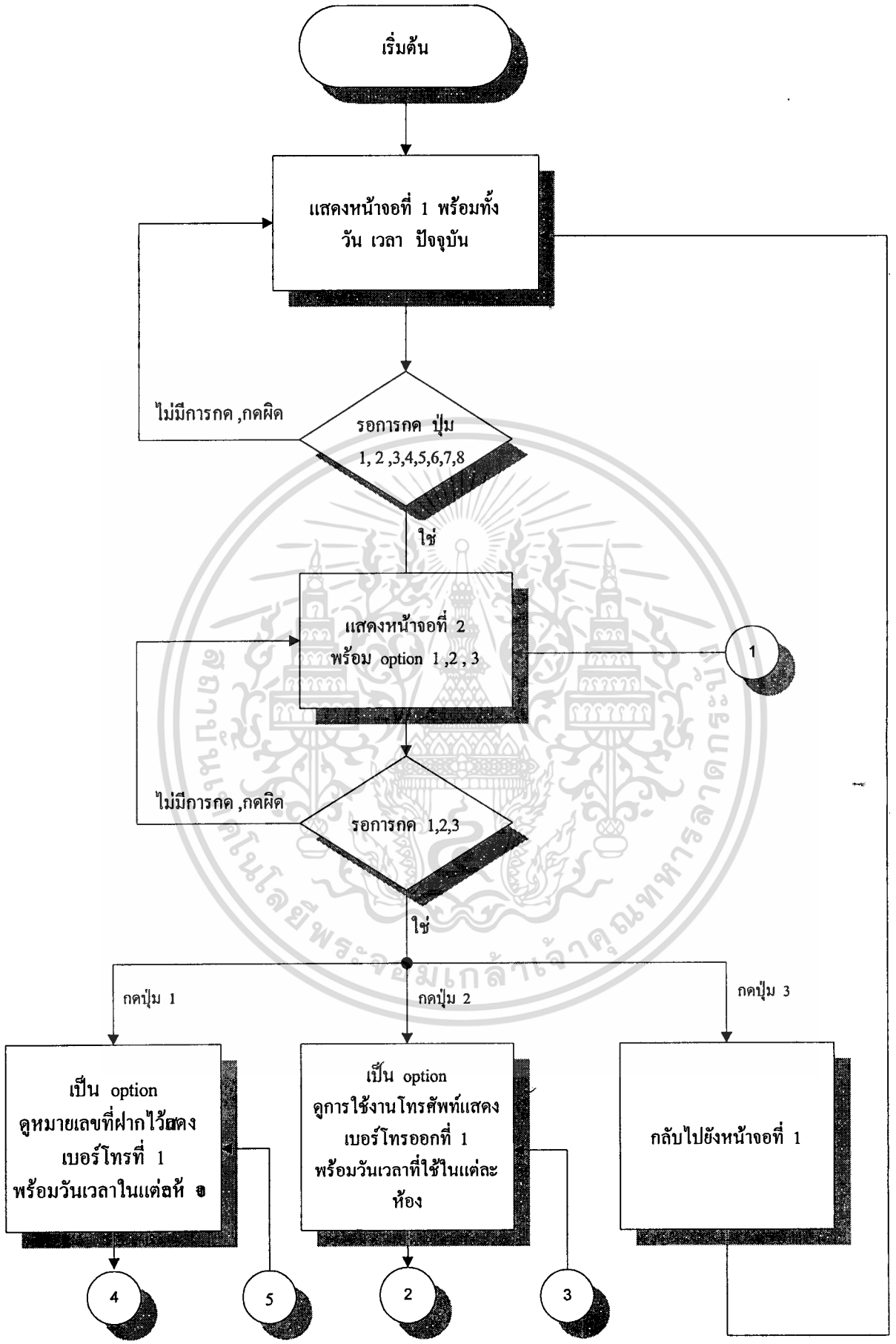
ผังงานแสดงขั้นตอนการโทรออกของสายใน (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



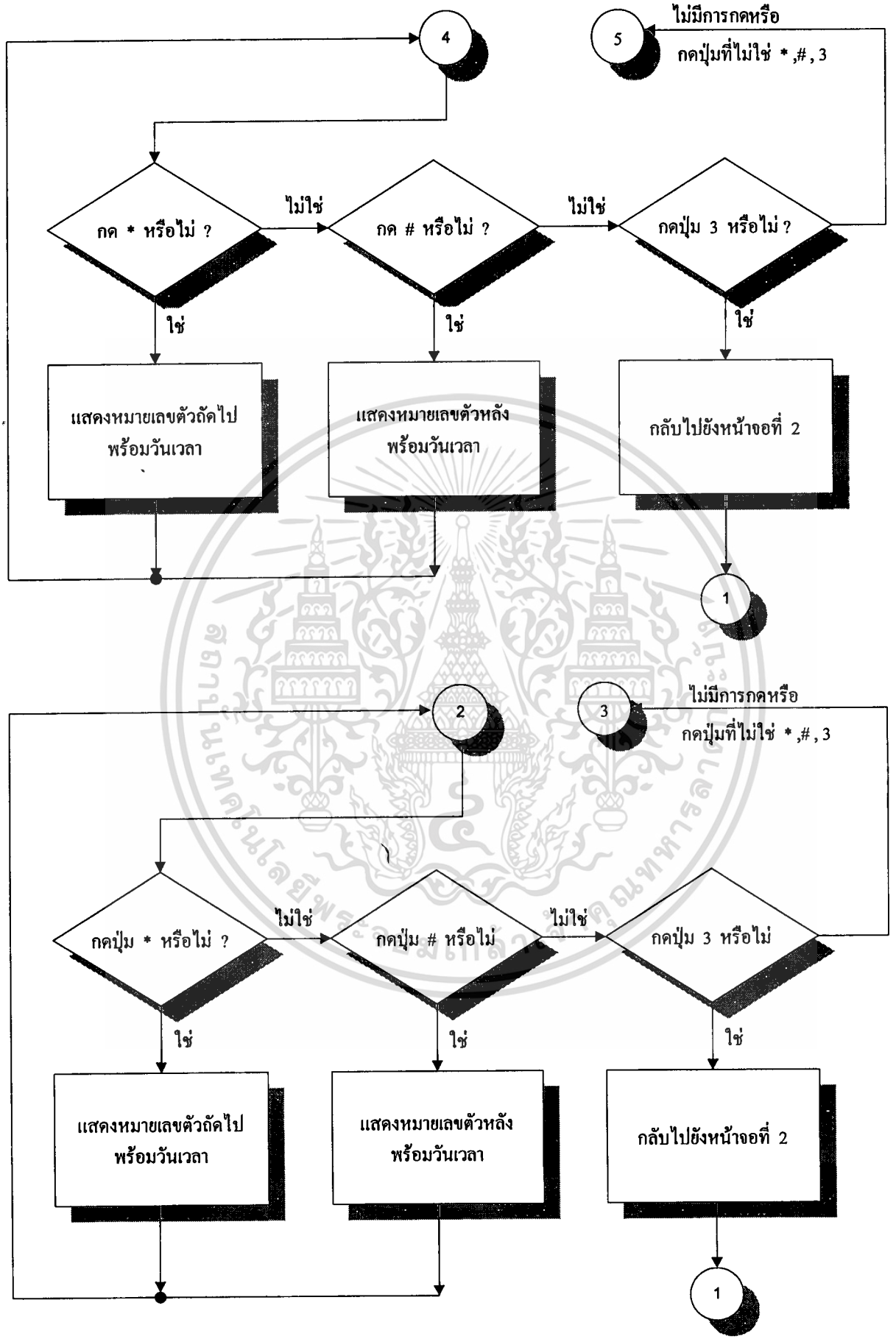
ผังงานแสดงขั้นตอนการใช้งานฟังก์ชั่นพิเศษของ PABX

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ผังงานแสดงขั้นตอนการทำงานของโปรแกรมควบคุมส่วนแสดงผล

เอกสารนี้เป็นเอกสาร[illegible]ให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ผังงานแสดงขั้นตอนการทำงานของโปรแกรมควบคุมส่วนแสดงผล (2)

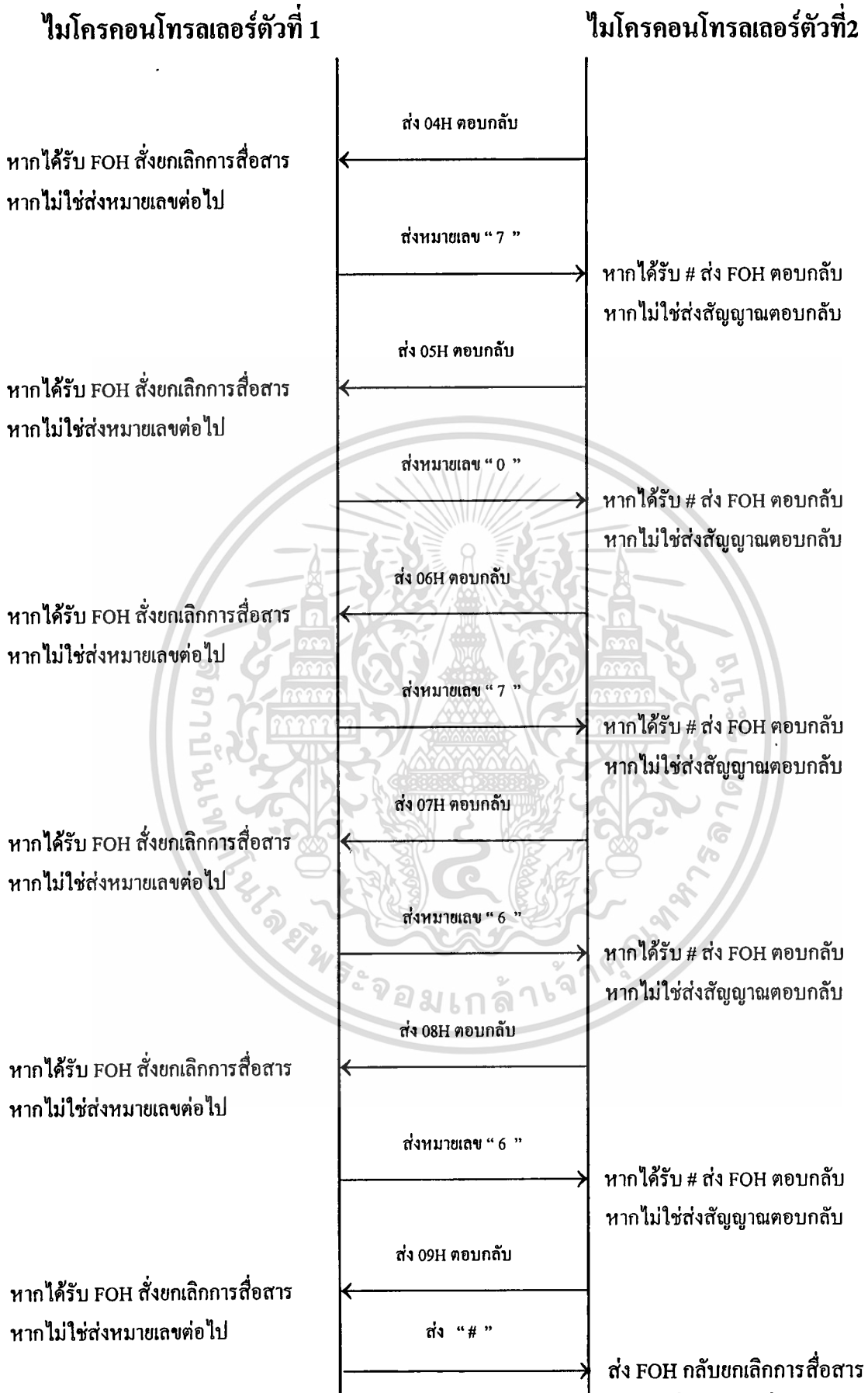
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนผังแสดงขั้นตอนการสื่อสารระหว่างไมโครคอนโทรลเลอร์ตัวที่ 1และตัวที่ 2 (1)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

แผนผังแสดงขั้นตอนการสื่อสารระหว่างไมโครคอนโทรลเลอร์ตัวที่ 1และตัวที่ 2 (2)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



แผนผังแสดงขั้นตอนการสื่อสารระหว่างไมโครคอนโทรลเลอร์ตัวที่ 1และตัวที่ 2 (3)

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
;FILENAME       PABX.ASM
;DESCRIPTION    SYSTEM
;HARDWARE       PABX
;ASSEMBLER      SXA51
;START-DATE     19/10/40
;******************VARIABLE************************
PORTA_1         EQU     0E014H  ;PORTA 8255_1
PORTB_1         EQU     0E015H  ;PORTB 8255_1
PORTC_1         EQU     0E016H  ;PORTC 8255_1
CONPORT_1       EQU     0E017H  ;CONTROL PORT 8255_1
;**************************************************
PORTA_2         EQU     0E018H  ;PORTA 8255_2
PORTB_2         EQU     0E019H  ;PORTB 8255_2
PORTC_2         EQU     0E01AH  ;PORTC 8255_2
CONPORT_2       EQU     0E01BH  ;CONTROL PORT 8255_2
;**************************************************
PORTA_3         EQU     0E01CH  ;PORTA 8255_3
PORTB_3         EQU     0E01DH  ;PORTB 8255_3
PORTC_3         EQU     0E01EH  ;PORTC 8255_3
CONPORT_3       EQU     0E01FH  ;CONTROL PORT 8255_3
;**************************************************
PORTA_4         EQU     0E0E0H  ;PORTA 8255_4
PORTB_4         EQU     0E0E1H  ;PORTB 8255_4
PORTC_4         EQU     0E0E2H  ;PORTC 8255_4
CONPORT_4       EQU     0E0E3H  ;CONTROL PORT 8255_4
;**************************************************
PORTA_5         EQU     0E010H  ;PORTA 8255_5
PORTB_5         EQU     0E011H  ;PORTB 8255_5
PORTC_5         EQU     0E012H  ;PORTC 8255_5
CONPORT_5       EQU     0E013H  ;CONTROL PORT 8255_5
;**************************************************
DRBT2           EQU     P1.0
DBT2            EQU     P1.1
DRBT1           EQU     P1.3
DBT1            EQU     P1.4
;(BIT USING)
S_EXT           EQU     20H     ; STATUS OF EXT. LINES
EXT_RG          EQU     21H     ; STATUS OF RING TO EXT.LINES
S_DTMF          EQU     22H     ; STATUS OF 4 DTMF DECODERS
EXT_DT          EQU     23H     ; DIAL TO EXT. LINES
EXT_BT          EQU     24H     ; BUSY TO EXT. LINES
EXT_RBT         EQU     25H     ; RING BACK TO EXT.LINES
CROSS_EXT       EQU     26H     ; STATUS OF 3-SLIC-SLIC CROSS POINTS
DTMF3           EQU     27H     ; USING DTMF DEC.NO.3 OF EXT.
DTMF4           EQU     28H     ; USING DTMF DEC.NO.4 OF EXT.
DTMF5           EQU     29H     ; USING DTMF DEC.NO.5 OF EXT.
DTMF6           EQU     2AH     ; USING DTMF DEC.NO.6 OF EXT.
CROSSE1         EQU     2BH     ; CONNECTING ON CROSS_E1 FOR EXT.1-8
CROSSE2         EQU     2CH     ; CONNECTING ON CROSS_E1 FOR EXT.1-8
CROSSE3         EQU     2DH     ; CONNECTING ON CROSS_E1 FOR EXT.1-8
WAIT_DF         EQU     2EH     ; EXT. LINE THAT WAIT FOR DTMF DEC.
TK_T            EQU     2FH     ; TONE TO TRUNK LINES
TK_TOE          EQU     30H     ; VALUE THAT CONTROL TK HOOK AND TOE
SOUND           EQU     31H     ; STATUS OF SOUND AUTO ANSWER
TK_DF           EQU     32H     ; STATUS OF RECEIVING
FNC1            EQU     33H     ; INCLINE LINE FUNCTION (PUSH "*")
FNC2            EQU     34H     ; CONFERENCE FUNCTION (PUSH "**")
FNC3            EQU     35H     ; RETURN TO REST LINE (PUSH "#")
FNC4            EQU     36H     ; LEAVING NUMBERS
TKE1            EQU     37H     ; EXT THAT CONNECTED TRUNK1
TKE2            EQU     38H     ; EXT THAT CONNECTED TRUNK2
;(SEQUENCE USING :SLIC NO.1-8 AND TK1=09H , TK2=0AH )
EXT1            EQU     39H     ; TRUNK THAT CONNECTED EXT1 LATELY
EXT2            EQU     3AH     ; TRUNK THAT CONNECTED EXT2 LATELY
EXT3            EQU     3BH     ; TRUNK THAT CONNECTED EXT3 LATELY
EXT4            EQU     3CH     ; TRUNK THAT CONNECTED EXT4 LATELY
EXT5            EQU     3DH     ; TRUNK THAT CONNECTED EXT5 LATELY
EXT6            EQU     3EH     ; TRUNK THAT CONNECTED EXT6 LATELY
EXT7            EQU     3FH     ; TRUNK THAT CONNECTED EXT7 LATELY
EXT8            EQU     40H     ; TRUNK THAT CONNECTED EXT8 LATELY
R_EXT1          EQU     41H     ; LINE THAT IS RINGING TO EXT.1
R_EXT2          EQU     42H     ; LINE THAT IS RINGING TO EXT.2
R_EXT3          EQU     43H     ; LINE THAT IS RINGING TO EXT.3
R_EXT4          EQU     44H     ; LINE THAT IS RINGING TO EXT.4
R_EXT5          EQU     45H     ; LINE THAT IS RINGING TO EXT.5
R_EXT6          EQU     46H     ; LINE THAT IS RINGING TO EXT.6
R_EXT7          EQU     47H     ; LINE THAT IS RINGING TO EXT.7
R_EXT8          EQU     48H     ; LINE THAT IS RINGING TO EXT.8
LROOM1          EQU     49H     ; EXT.ROOM IS LEAVED NUMBERS
LROOM2          EQU     4AH     ; EXT.ROOM IS LEAVED NUMBERS
;(SEQUENCE USING OF NUMBER 0-9,*,# )
P_NOSL1         EQU     4BH     ; 2 PREVIOUS NUMBERS OF EXT.1
P_NOSL2         EQU     4CH     ; 2 PREVIOUS NUMBERS OF EXT.2
P_NOSL3         EQU     4DH     ; 2 PREVIOUS NUMBERS OF EXT.3
P_NOSL4         EQU     4EH     ; 2 PREVIOUS NUMBERS OF EXT.4
P_NOSL5         EQU     4FH     ; 2 PREVIOUS NUMBERS OF EXT.5
P_NOSL6         EQU     50H     ; 2 PREVIOUS NUMBERS OF EXT.6
P_NOSL7         EQU     51H     ; 2 PREVIOUS NUMBERS OF EXT.7
P_NOSL8         EQU     52H     ; 2 PREVIOUS NUMBERS OF EXT.8
KP_DA           EQU     53H     ; KEEP VALUE FOR HOOK OF EXT. LINES
DF_BUF          EQU     54H
;(BIT USING)
OUT_T1          EQU     55H     ; EXT.CALL FOR USE TRUNK LINE.1
OUT_T2          EQU     56H     ; EXT.CALL FOR USE TRUNK LINE.2
SEQ_N1          EQU     57H     ; SEQUENCE OF LEAVING NUMBERS OF TK1
SEQ_N2          EQU     58H     ; SEQUENCE OF LEAVING NUMBERS OF TK2
PP1             EQU     59H     ; POINTER1 OF LEAVING NUMBERS
PP2             EQU     5AH     ; POINTER2 OF LEAVING NUMBERS
C_TK            EQU     5BH     ;
RA1             EQU     5CH     ;
RA2             EQU     5DH     ;
R_RBT1          EQU     5EH     ;
R_RBT2          EQU     5FH     ;
AD_LN1          EQU     60H     ; FIRST ADDRESS OF L_NUM.1
AD_LN2          EQU     70H     ; FIRST ADDRESS OF L_NUM.2
```

```
;****************PROGRAM******************
          ORG     0000H
          LJMP    SETSYS

          ORG     0003H
          MOV     IE,#00H
          LJMP    T_HK
INT_0:    MOV     IE,#85H
          RETI

          ORG     0013H
          MOV     IE,#00H
          LJMP    PUSH_N
INT_1:    LCALL   DELAY90
          MOV     IE,#85H
          RETI

          ORG     0100H
SETSYS:   LCALL   DELAY1
          MOV     IE,#85H          ;SET INTERRUPT ALLOWANCE
          MOV     SP,#80H          ;STACK AT ADDRESS 80H
;*************** INITIAL PORT ******************
          MOV     P1,#0DBH         ;PORTA='11011011'
          LCALL   DELAY90
          MOV     DPTR,#CONPORT_1
          MOV     A,#8BH
          MOVX    @DPTR,A          ;***PA=OUT,PB=IN,PC=IN***
          MOV     DPTR,#CONPORT_2
          MOV     A,#80H
          MOVX    @DPTR,A          ;***PA=OUT,PB=OUT,PC=OUT***
          MOV     DPTR,#CONPORT_3
          MOV     A,#80H
          MOVX    @DPTR,A          ;***PA=OUT,PB=OUT,PC=OUT***
          MOV     DPTR,#CONPORT_4
          MOV     A,#90H
          MOVX    @DPTR,A          ;***PA=IN,PB=OUT,PC=OUT***
          MOV     DPTR,#CONPORT_5
          MOV     A,#89H
          MOVX    @DPTR,A          ;***PA=OUT,PB=OUT,PC=IN***
          MOV     DPTR,#PORTA_1
          MOV     A,#00H
          MOVX    @DPTR,A
          MOV     DPTR,#PORTB_4
          MOV     A,#00H
          MOVX    @DPTR,A
          MOV     DPTR,#PORTC_4
          MOV     A,#80H
          MOVX    @DPTR,A
          MOV     DPTR,#PORTA_5
          MOV     A,#00H
          MOVX    @DPTR,A
          MOV     DPTR,#PORTB_5
          MOV     A,#00H
          MOVX    @DPTR,A
          MOV     A,#00H
          MOV     EXT_RG,A
          MOV     S_DTMF,A
          MOV     EXT_DT,A
          MOV     EXT_BT,A
          MOV     EXT_RBT,A
          MOV     CROSS_EXT,A
          MOV     DTMF3,A
          MOV     DTMF4,A
          MOV     DTMF5,A
          MOV     DTMF6,A
          MOV     CROSSE1,A
          MOV     CROSSE2,A
          MOV     CROSSE3,A
          MOV     WAIT_DF,A
          MOV     TKE1,A
          MOV     TKE2,A
          MOV     EXT1,A
          MOV     EXT2,A
          MOV     EXT3,A
          MOV     EXT4,A
          MOV     EXT5,A
          MOV     EXT6,A
          MOV     EXT7,A
          MOV     EXT8,A
          MOV     SOUND,A
          MOV     KP_DA,A
          MOV     TK_TOE,A
          MOV     R_EXT1,A
          MOV     R_EXT2,A
          MOV     R_EXT3,A
          MOV     R_EXT4,A
          MOV     R_EXT5,A
          MOV     R_EXT6,A
          MOV     R_EXT7,A
          MOV     R_EXT8,A
          MOV     LROOM1,A
          MOV     LROOM2,A
          MOV     FNC1,A
          MOV     FNC2,A
          MOV     FNC3,A
          MOV     FNC4,A
          MOV     DF_BUF,A
          MOV     TK_T,A
          MOV     OUT_T1,A
          MOV     OUT_T2,A
          MOV     RA1,A
          MOV     RA2,A
          MOV     R_RBT1,A
          MOV     R_RBT2,A
          MOV     A,#0FFH
          MOV     S_EXT,A
```

```
            MOV     TK_DF,A
            MOV     P_NOSL1,A
            MOV     P_NOSL2,A
            MOV     P_NOSL3,A
            MOV     P_NOSL4,A
            MOV     P_NOSL5,A
            MOV     P_NOSL6,A
            MOV     P_NOSL7,A
            MOV     P_NOSL8,A
            MOV     SEQ_N1,#0BH
            MOV     SEQ_N2,#0BH
            MOV     PP1,#AD_LN1
            MOV     PP2,#AD_LN2

;*****************************************************
;*****************************************************
;                    MAIN PROGRAM
;*****************************************************
;*****************************************************
MAIN_SYS:
CHK_SL:     MOV     DPTR,#PORTA_4    ;LOOP CHECK STATUS OF EXT HK
            MOVX    A,@DPTR
            CJNE    A,S_EXT,SV_SL
            JB      DBT1,JSV_BT1
            JB      DBT2,JSV_BT2
            MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,#00H,J_RLINK
            LJMP    CHK_SL
JSV_BT1:    LJMP    SV_BT1
JSV_BT2:    LJMP    SV_BT2
J_RLINK:    LCALL   DELAY90
            MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,#00H,$+6
            LJMP    CHK_SL
            LJMP    R_LINK
;******* SERVICE WHILE EXT. LINE CHANGE HOOK ********
SV_SL:      XRL     A,S_EXT          ; HON  = HOLD ON
CHK_SL1:    CJNE    A,#01H,JS2       ; HOFF = HOLD OFF
            SJMP    SL1
JS2:        LJMP    CHK_SL2
SL1:        JNB     S_EXT.0,OFF_HK1
            LJMP    ON_HK1
;***************  EXT1 HOLDING OFF ******************
OFF_HK1:    LCALL   DELAY90
            SETB    S_EXT.0
            MOV     P_NOSL1,#0FFH        ;CLEAR BUFFER OF NUMBERS
            CLR     WAIT_DF.0            ;CLEAR WAITING DTMF DEC.
            ANL     FNC1,#0FEH           ;CLEAR FUNCTION
            ANL     FNC2,#0FEH
            ANL     FNC3,#0FEH
            ANL     OUT_T1,#0FEH
            ANL     OUT_T2,#0FEH
            JB      EXT_DT.0,CLR_DT1
            JB      EXT_BT.0,CLR_BT1
            JB      EXT_RBT.0,CLR_RBT1
            LCALL   OFF_DFS1
            LJMP    OFF_LS1
CLR_DT1:    MOV     R7,#00H
            LCALL   DT_SL1
            LJMP    OFF_LS1
CLR_BT1:    MOV     R7,#00H
            LCALL   BT_SL1
            LJMP    OFF_LS1
CLR_RBT1:   MOV     R7,#00H
            LCALL   RBT_SL1
            MOV     A,R_EXT2
            CJNE    A,#01H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#01H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#01H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#01H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#01H,$+9
            LCALL   OFF_R6
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#01H,$+9
            LCALL   OFF_R7
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#01H,$+6
            LCALL   OFF_R8

;********* OFF LINE THAT CONNECT WITH EXT.1 ***********
OFF_LS1:    MOV     A,#09H
            CJNE    A,EXT1,CHKT2E1
            MOV     R7,#00H
            LCALL   TK1_SL1
            MOV     A,TKE1
            ANL     A,#01H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E1:    MOV     A,#0AH
```

```
            CJNE    A,EXT1,CHKC1E1
            MOV     R7,#00H
            LCALL   TK2_SL1
            MOV     A,TKE2
            ANL     A,#01H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E1:    JNB     CROSSE1.0,CHKC2E1
            MOV     R7,#00H
            LCALL   CROSS1_1
            LCALL   CHK_CR1
CHKC2E1:    JNB     CROSSE2.0,CHKC3E1
            MOV     R7,#00H
            LCALL   CROSS2_1
            LCALL   CHK_CR2
CHKC3E1:    JNB     CROSSE3.0,OFF_E1END
            MOV     R7,#00H
            LCALL   CROSS3_1
            LCALL   CHK_CR3
OFF_E1END:  LJMP    MAIN_SYS

;************** EXT1.HOLD ON ******************
ON_HK1:     LCALL   DELAY90
            CLR     S_EXT.0
            JB      EXT_RG.0,CRSL1
            LJMP    GDT_SL1
CRSL1:      LCALL   OFF_R1
            MOV     A,R_EXT1
            CJNE    A,#09H,CCS1T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
CCS1T2:     CJNE    A,#0AH,CCS1S2
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL1
            LJMP    MAIN_SYS
            LJMP    CCS1S3
CCS1S2:     CJNE    A,#02H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL2
            LCALL   OFF_DFS2
            MOV     A,FNC1
            ANL     A,#02H
            CJNE    A,#02H,CF2S1S2
            MOV     A,EXT2
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL1
            LJMP    NMS1S2
CF2S1S2:    MOV     A,FNC2
            ANL     A,#02H
            CJNE    A,#02H,$-7
            MOV     A,EXT2
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL1
            LCALL   CTK1_SL2
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S1S2S3
            LCALL   CTK2_SL1
            LCALL   CTK2_SL2
            LJMP    MAIN_SYS
S1S2S3:     CJNE    A,#03H,S1S2S4
            JNB     CROSSE1.2,$+9   ; CHECK CROSSP.OF EXT3
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.2,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS1S2
            JNB     CROSSE3.2,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S1S2S4:     CJNE    A,#04H,S1S2S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS1S2
            JNB     CROSSE3.3,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S1S2S5:     CJNE    A,#05H,S1S2S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS1S2
            JNB     CROSSE3.4,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S1S2S6:     CJNE    A,#06H,S1S2S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
```

```
            LJMP    NMS1S2
            JNB     CROSSE3.5,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S1S2S7:     CJNE    A,#07H,S1S2S8
            JNB     CROSSE1.6,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS1S2
            JNB     CROSSE3.6,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S1S2S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS1S2
            JNB     CROSSE3.7,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
NMS1S2:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL2           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL1
            LCALL   ONBT_SL2
            LJMP    MAIN_SYS
            LJMP    CCS1S4
CCS1S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
            MOV     A,FNC1
            ANL     A,#04H
            CJNE    A,#04H,CF2S1S3
            MOV     A,EXT3
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL1
            LJMP    NMS1S3

CF2S1S3:    MOV     A,FNC2
            ANL     A,#04H
            CJNE    A,#04H,$-7
            MOV     A,EXT3
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL1
            LCALL   CTK1_SL3
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S1S3S2
            LCALL   CTK2_SL1
            LCALL   CTK2_SL3
            LJMP    MAIN_SYS
S1S3S2:     CJNE    A,#02H,S1S3S4
            JNB     CROSSE1.1,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.1,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
            JNB     CROSSE3.1,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
S1S3S4:     CJNE    A,#04H,S1S3S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
            JNB     CROSSE3.3,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
S1S3S5:     CJNE    A,#05H,S1S3S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
            JNB     CROSSE3.4,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
S1S3S6:     CJNE    A,#06H,S1S3S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
```

```
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
S1S5S7:         CJNE    A,#07H,S1S5S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.6,$-3
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
S1S5S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.7,$-3
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
NMS1S5:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL1_SL5           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL1
                LCALL   ONBT_SL5
                LJMP    MAIN_SYS

                LJMP    CCS1S7
CCS1S6:         CJNE    A,#06H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL6
                LCALL   OFF_DFS6
                MOV     A,FNC1
                ANL     A,#20H
                CJNE    A,#20H,CF2S1S6
                MOV     A,EXT6
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL1
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL1
                LJMP    NMS1S6
CF2S1S6:        MOV     A,FNC2
                ANL     A,#20H
                CJNE    A,#20H,$-7
                MOV     A,EXT6
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL1
                LCALL   CTK1_SL6
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S1S6S2
                LCALL   CTK2_SL1
                LCALL   CTK2_SL6
                LJMP    MAIN_SYS
S1S6S2:         CJNE    A,#02H,S1S6S3
                JNB     CROSSE1.1,$+9
                LCALL   C1SL1_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL1_SL6
                LJMP    MAIN_SYS
                LJMP    NMS1S6
                JNB     CROSSE3.1,$-3
                LCALL   C3SL1_SL6
                LJMP    MAIN_SYS
S1S6S3:         CJNE    A,#03H,S1S6S4
                JNB     CROSSE1.2,$+9
                LCALL   C1SL1_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.2,$+12
                LCALL   C2SL1_SL6
                LJMP    MAIN_SYS
                LJMP    NMS1S6
                JNB     CROSSE3.2,$-3
                LCALL   C3SL1_SL6
                LJMP    MAIN_SYS
S1S6S4:         CJNE    A,#04H,S1S6S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL1_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL1_SL6
                LJMP    MAIN_SYS
                LJMP    NMS1S6
                JNB     CROSSE3.3,$-3
                LCALL   C3SL1_SL6
                LJMP    MAIN_SYS
S1S6S5:         CJNE    A,#05H,S1S6S7
                JNB     CROSSE1.4,$+9
                LCALL   C1SL1_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL1_SL6
                LJMP    MAIN_SYS
                LJMP    NMS1S6
                JNB     CROSSE3.4,$-3
```

```
                JNB     CROSSE3.5,$-3
                LCALL   C3SL1_SL4
                LJMP    MAIN_SYS
S1S4S7:         CJNE    A,#07H,S1S4S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL1_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL1_SL4
                LJMP    MAIN_SYS
                LJMP    NMS1S4
                JNB     CROSSE3.6,$-3
                LCALL   C3SL1_SL4
                LJMP    MAIN_SYS
S1S4S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL1_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL1_SL4
                LJMP    MAIN_SYS
                LJMP    NMS1S4
                JNB     CROSSE3.7,$-3
                LCALL   C3SL1_SL4
                LJMP    MAIN_SYS
NMS1S4:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL1_SL4
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL1_SL4
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL1_SL4           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL1
                LCALL   ONBT_SL4
                LJMP    MAIN_SYS
                LJMP    CCS1S6
CCS1S5:         CJNE    A,#05H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL5
                LCALL   OFF_DFS5
                MOV     A,FNC1
                ANL     A,#10H
                CJNE    A,#10H,CF2S1S5
                MOV     A,EXT5
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL1
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL1
                LJMP    NMS1S5
CF2S1S5:        MOV     A,FNC2
                ANL     A,#10H
                CJNE    A,#10H,$-7
                MOV     A,EXT5
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL1
                LCALL   CTK1_SL5
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S1S5S2
                LCALL   CTK2_SL1
                LCALL   CTK2_SL5
                LJMP    MAIN_SYS
S1S5S2:         CJNE    A,#02H,S1S5S3
                JNB     CROSSE1.1,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.1,$-3
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
S1S5S3:         CJNE    A,#03H,S1S5S4
                JNB     CROSSE1.2,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.2,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.2,$-3
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
S1S5S4:         CJNE    A,#04H,S1S5S6
                JNB     CROSSE1.3,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.3,$-3
                LCALL   C3SL1_SL5
                LJMP    MAIN_SYS
S1S5S6:         CJNE    A,#06H,S1S5S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL1_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL1_SL5
                LJMP    MAIN_SYS
                LJMP    NMS1S5
                JNB     CROSSE3.5,$-3
```

```
            JNB     CROSSE3.5,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
S1S3S7:     CJNE    A,#07H,S1S3S8
            JNB     CROSSE1.6,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
            JNB     CROSSE3.6,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
S1S3S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            LJMP    NMS1S3
            JNB     CROSSE3.7,$-3
            LCALL   C3SL1_SL3
            LJMP    MAIN_SYS
NMS1S3:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL3
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL3           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL1
            LCALL   ONBT_SL3
            LJMP    MAIN_SYS

            LJMP    CCS1S5
CCS1S4:     CJNE    A,#04H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL4
            LCALL   OFF_DFS4
            MOV     A,FNC1
            ANL     A,#08H
            CJNE    A,#08H,CF2S1S4
            MOV     A,EXT4
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL1
            LJMP    NMS1S4

CF2S1S4:    MOV     A,FNC2
            ANL     A,#08H
            CJNE    A,#08H,$-7
            MOV     A,EXT4
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL1
            LCALL   CTK1_SL4
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S1S4S2
            LCALL   CTK2_SL1
            LCALL   CTK2_SL4
            LJMP    MAIN_SYS
S1S4S2:     CJNE    A,#02H,S1S4S3
            JNB     CROSSE1.1,$+9
            LCALL   C1SL1_SL4
            LJMP    MAIN_SYS
            JNB     CROSSE2.1,$+12
            LCALL   C2SL1_SL4
            LJMP    MAIN_SYS
            LJMP    NMS1S4
            JNB     CROSSE3.1,$-3
            LCALL   C3SL1_SL4
            LJMP    MAIN_SYS
S1S4S3:     CJNE    A,#03H,S1S4S5
            JNB     CROSSE1.2,$+9
            LCALL   C1SL1_SL4
            LJMP    MAIN_SYS
            JNB     CROSSE2.2,$+12
            LCALL   C2SL1_SL4
            LJMP    MAIN_SYS
            LJMP    NMS1S4
            JNB     CROSSE3.2,$-3
            LCALL   C3SL1_SL4
            LJMP    MAIN_SYS
S1S4S5:     CJNE    A,#05H,S1S4S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL1_SL4
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL1_SL4
            LJMP    MAIN_SYS
            LJMP    NMS1S4
            JNB     CROSSE3.4,$-3
            LCALL   C3SL1_SL4
            LJMP    MAIN_SYS
S1S4S6:     CJNE    A,#06H,S1S4S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL4
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL4
            LJMP    MAIN_SYS
            LJMP    NMS1S4
```

```
                LJMP    NMS1S2
                JNB     CROSSE3.5,$-3
                LCALL   C3SL1_SL2
                LJMP    MAIN_SYS
S1S2S7:         CJNE    A,#07H,S1S2S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL1_SL2
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL1_SL2
                LJMP    MAIN_SYS
                LJMP    NMS1S2
                JNB     CROSSE3.6,$-3
                LCALL   C3SL1_SL2
                LJMP    MAIN_SYS
S1S2S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL1_SL2
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL1_SL2
                LJMP    MAIN_SYS
                LJMP    NMS1S2
                JNB     CROSSE3.7,$-3
                LCALL   C3SL1_SL2
                LJMP    MAIN_SYS
NMS1S2:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL1_SL2
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL1_SL2
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL1_SL2           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL1
                LCALL   ONBT_SL2
                LJMP    MAIN_SYS
                LJMP    CCS1S4
CCS1S3:         CJNE    A,#03H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL3
                LCALL   OFF_DFS3
                MOV     A,FNC1
                ANL     A,#04H
                CJNE    A,#04H,CF2S1S3
                MOV     A,EXT3
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL1
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL1
                LJMP    NMS1S3
CF2S1S3:        MOV     A,FNC2
                ANL     A,#04H
                CJNE    A,#04H,$-7
                MOV     A,EXT3
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL1
                LCALL   CTK1_SL3
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S1S3S2
                LCALL   CTK2_SL1
                LCALL   CTK2_SL3
                LJMP    MAIN_SYS
S1S3S2:         CJNE    A,#02H,S1S3S4
                JNB     CROSSE1.1,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS1S3
                JNB     CROSSE3.1,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S1S3S4:         CJNE    A,#04H,S1S3S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS1S3
                JNB     CROSSE3.3,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S1S3S5:         CJNE    A,#05H,S1S3S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS1S3
                JNB     CROSSE3.4,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S1S3S6:         CJNE    A,#06H,S1S3S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS1S3
```

```
            LCALL   C3SL1_SL7
            LJMP    MAIN_SYS
S1S7S6:     CJNE    A,#06H,S1S7S8
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL7
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL7
            LJMP    MAIN_SYS
            LJMP    NMS1S7
            JNB     CROSSE3.5,$-3
            LCALL   C3SL1_SL7
            LJMP    MAIN_SYS
S1S7S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL1_SL7
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL1_SL7
            LJMP    MAIN_SYS
            LJMP    NMS1S7
            JNB     CROSSE3.7,$-3
            LCALL   C3SL1_SL7
            LJMP    MAIN_SYS
NMS1S7:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL1
            LCALL   ONBT_SL7
            LJMP    MAIN_SYS
            LJMP    MAIN_SYS
CCS1S8:     CJNE    A,#08H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL8
            LCALL   OFF_DFS8
            MOV     A,FNC1
            ANL     A,#80H
            CJNE    A,#80H,CF2S1S8
            MOV     A,EXT8
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL1
            LJMP    NMS1S8
CF2S1S8:    MOV     A,FNC2
            ANL     A,#80H
            CJNE    A,#80H,$-7
            MOV     A,EXT8
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL1
            LCALL   CTK1_SL8
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S1S8S2
            LCALL   CTK2_SL1
            LCALL   CTK2_SL8
            LJMP    MAIN_SYS
S1S8S2:     CJNE    A,#02H,S1S8S3
            JNB     CROSSE1.1,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.1,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.1,$-3
            LCALL   C3SL1_SL8
            LJMP    MAIN_SYS
S1S8S3:     CJNE    A,#03H,S1S8S4
            JNB     CROSSE1.2,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.2,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.2,$-3
            LCALL   C3SL1_SL8
            LJMP    MAIN_SYS
S1S8S4:     CJNE    A,#04H,S1S8S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.3,$-3
            LCALL   C3SL1_SL8
            LJMP    MAIN_SYS
S1S8S5:     CJNE    A,#05H,S1S8S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.4,$-3
            LCALL   C3SL1_SL8
```

```
            LJMP    MAIN_SYS
S1S8S6:     CJNE    A,#06H,S1S8S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.5,$-3
            LCALL   C3SL1_SL8
            LJMP    MAIN_SYS
S1S8S7:     CJNE    A,#07H,$-3
            JNB     CROSSE1.6,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            LJMP    NMS1S8
            JNB     CROSSE3.6,$-3
            LCALL   C3SL1_SL8
            LJMP    MAIN_SYS
NMS1S8:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL1
            LCALL   ONBT_SL8
            LJMP    MAIN_SYS

GDT_SL1:    MOV     R7,#0FFH
            LCALL   DT_SL1              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E1:    CJNE    A,DTMF3,C_DF4E1
            MOV     R7,#0FFH
            LCALL   DF3_SL1
            LJMP    MAIN_SYS
C_DF4E1:    CJNE    A,DTMF4,C_DF5E1
            MOV     R7,#0FFH
            LCALL   DF4_SL1
            LJMP    MAIN_SYS
C_DF5E1:    CJNE    A,DTMF5,C_DF6E1
            MOV     R7,#0FFH
            LCALL   DF5_SL1
            LJMP    MAIN_SYS
C_DF6E1:    CJNE    A,DTMF6,NO_DF_E1
            MOV     R7,#0FFH
            LCALL   DF6_SL1
            LJMP    MAIN_SYS

NO_DF_E1:   SETB    WAIT_DF.0
            LJMP    MAIN_SYS
;*******************************************************
CHK_SL2:    CJNE    A,#02H,JS3          ; HOFF = HOLD OFF
            SJMP    SL2
JS3:        LJMP    CHK_SL3
SL2:        JNB     S_EXT.1,OFF_HK2
            LJMP    ON_HK2
;***************  EXT2 HOLDING OFF ******************
OFF_HK2:    LCALL   DELAY90
            SETB    S_EXT.1
            MOV     P_NOSL2,#0FFH       ;CLEAR BUFFER OF NUMBERS
            CLR     WAIT_DF.1           ;CLEAR WAITING DTMF DEC'.
            ANL     FNC1,#0FDH          ;CLEAR FUNCTION
            ANL     FNC2,#0FDH
            ANL     FNC3,#0FDH
            ANL     OUT_T1,#0FDH
            ANL     OUT_T2,#0FDH
            JB      EXT_DT.1,CLR_DT2
            JB      EXT_BT.1,CLR_BT2
            JB      EXT_RBT.1,CLR_RBT2
            LCALL   OFF_DFS2
            LJMP    OFF_LS2
CLR_DT2:    MOV     R7,#00H
            LCALL   DT_SL2
            LJMP    OFF_LS2
CLR_BT2:    MOV     R7,#00H
            LCALL   BT_SL2
            LJMP    OFF_LS2
CLR_RBT2:   MOV     R7,#00H
            LCALL   RBT_SL2
            MOV     A,R_EXT1
            CJNE    A,#02H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#02H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#02H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#02H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#02H,$+9
            LCALL   OFF_R6
```

```
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#02H,$+9
            LCALL   OFF_R7
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#02H,$+6
            LCALL   OFF_R8
;********* OFF LINE THAT CONNECT WITH EXT.2 ***********
OFF_LS2:    MOV     A,#09H
            CJNE    A,EXT2,CHKT2E2
            MOV     R7,#00H
            LCALL   TK1_SL2
            MOV     A,TKE1
            ANL     A,#02H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E2:    MOV     A,#0AH
            CJNE    A,EXT2,CHKC1E2
            MOV     R7,#00H
            LCALL   TK2_SL2
            MOV     A,TKE2
            ANL     A,#02H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E2:    JNB     CROSSE1.1,CHKC2E2
            MOV     R7,#00H
            LCALL   CROSS1_2
            LCALL   CHK_CR1
CHKC2E2:    JNB     CROSSE2.1,CHKC3E2
            MOV     R7,#00H
            LCALL   CROSS2_2
            LCALL   CHK_CR2
CHKC3E2:    JNB     CROSSE3.1,OFF_E2END
            MOV     R7,#00H
            LCALL   CROSS3_2
            LCALL   CHK_CR3
OFF_E2END:  LJMP    MAIN_SYS
;************** EXT2.HOLD ON ******************
ON_HK2:     LCALL   DELAY90
            CLR     S_EXT.1
            JB      EXT_RG.1,CRSL2
            LJMP    GDT_SL2
CRSL2:      MOV     R7,#00H
            LCALL   OFF_R2
            MOV     A,R_EXT2
            CJNE    A,#09H,CCS2T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL2
            LJMP    MAIN_SYS
CCS2T2:     CJNE    A,#0AH,CCS2S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL2
            LJMP    MAIN_SYS
            LJMP    CCS2S3
CCS2S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
            MOV     A,FNC1
            ANL     A,#01H
            CJNE    A,#01H,CF2S2S1
            MOV     A,EXT1
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL2
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL2
            LJMP    NMS2S1
CF2S2S1:    MOV     A,FNC2
            ANL     A,#01H
            CJNE    A,#01H,$-7
            MOV     A,EXT1
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL2
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S2S1S3
            LCALL   CTK2_SL2
            LCALL   CTK2_SL1
            LJMP    MAIN_SYS
S2S1S3:     CJNE    A,#03H,S2S1S4
            JNB     CROSSE1.2,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.2,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.2,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S2S1S4:     CJNE    A,#04H,S2S1S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.3,$-3
            LCALL   C3SL1_SL2
```

```
            LJMP    MAIN_SYS
S2J1S5:     CJNE    A,#05H,S2S1S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.4,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S2S1S6:     CJNE    A,#06H,S2S1S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.5,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S2S1S7:     CJNE    A,#07H,S2S1S8
            JNB     CROSSE1.6,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.6,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
S2S1S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            LJMP    NMS2S1
            JNB     CROSSE3.7,$-3
            LCALL   C3SL1_SL2
            LJMP    MAIN_SYS
NMS2S1:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL2
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL2
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL2           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL2
            LCALL   ONBT_SL1
            LJMP    MAIN_SYS

            LJMP    CCS2S4
CCS2S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
            MOV     A,FNC1
            ANL     A,#04H
            CJNE    A,#04H,CF2S2S3
            MOV     A,EXT3
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL2
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL2
            LJMP    NMS2S3
CF2S2S3:    MOV     A,FNC2
            ANL     A,#04H
            CJNE    A,#04H,$-7
            MOV     A,EXT3
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL2
            LCALL   CTK1_SL3
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S2S3S1
            LCALL   CTK2_SL2
            LCALL   CTK2_SL3
            LJMP    MAIN_SYS
S2S3S1:     CJNE    A,#01H,S2S3S4
            JNB     CROSSE1.0,$+9
            LCALL   C1SL2_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.0,$+12
            LCALL   C2SL2_SL3
            LJMP    MAIN_SYS
            LJMP    NMS2S3
            JNB     CROSSE3.0,$-3
            LCALL   C3SL2_SL3
            LJMP    MAIN_SYS
S2S3S4:     CJNE    A,#04H,S2S3S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL2_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL2_SL3
            LJMP    MAIN_SYS
            LJMP    NMS2S3
            JNB     CROSSE3.3,$-3
            LCALL   C3SL1_SL3
```

```
                LJMP    MAIN_SYS
S2S3S5:         CJNE    A,#05H,S2S3S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL2_SL3
                LJMP    MAIN_SYS
                LJMP    NMS2S3
                JNB     CROSSE3.4,$-3
                LCALL   C3SL2_SL3
                LJMP    MAIN_SYS
S2S3S6:         CJNE    A,#06H,S2S3S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL2_SL3
                LJMP    MAIN_SYS
                LJMP    NMS2S3
                JNB     CROSSE3.5,$-3
                LCALL   C3SL2_SL3
                LJMP    MAIN_SYS
S2S3S7:         CJNE    A,#07H,S2S3S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL2_SL3
                LJMP    MAIN_SYS
                LJMP    NMS2S3
                JNB     CROSSE3.6,$-3
                LCALL   C3SL2_SL3
                LJMP    MAIN_SYS
S2S3S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL2_SL3
                LJMP    MAIN_SYS
                LJMP    NMS2S3
                JNB     CROSSE3.7,$-3
                LCALL   C3SL2_SL3
                LJMP    MAIN_SYS
NMS2S3:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL2_SL3
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL2_SL3           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL2
                LCALL   ONBT_SL3
                LJMP    MAIN_SYS
                LJMP    CCS2S5
CCS2S4:         CJNE    A,#04H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL4
                LCALL   OFF_DFS4
                MOV     A,FNC1
                ANL     A,#08H
                CJNE    A,#08H,CF2S2S4
                MOV     A,EXT4
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL2
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL2
                LJMP    NMS2S4
CF2S2S4:        MOV     A,FNC2
                ANL     A,#08H
                CJNE    A,#08H,$-7
                MOV     A,EXT4
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL2
                LCALL   CTK1_SL4
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S2S4S1
                LCALL   CTK2_SL2
                LCALL   CTK2_SL4
                LJMP    MAIN_SYS
S2S4S1:         CJNE    A,#01H,S2S4S3
                JNB     CROSSE1.0,$+9
                LCALL   C1SL2_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL2_SL4
                LJMP    MAIN_SYS
                LJMP    NMS2S4
                JNB     CROSSE3.0,$-3
                LCALL   C3SL2_SL4
                LJMP    MAIN_SYS
S2S4S3:         CJNE    A,#03H,S2S4S5
                JNB     CROSSE1.2,$+9
                LCALL   C1SL2_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.2,$+12
                LCALL   C2SL2_SL4
                LJMP    MAIN_SYS
                LJMP    NMS2S4
                JNB     CROSSE3.2,$-3
                LCALL   C3SL2_SL4
                LJMP    MAIN_SYS
```

```
S2S4S5:  CJNE   A,#05H,S2S4S6
         JNB    CROSSE1.4,$+9
         LCALL  C1SL2_SL4
         LJMP   MAIN_SYS
         JNB    CROSSE2.4,$+12
         LCALL  C2SL2_SL4
         LJMP   MAIN_SYS
         LJMP   NMS2S4
         JNB    CROSSE3.4,$-3
         LCALL  C3SL2_SL4
         LJMP   MAIN_SYS
S2S4S6:  CJNE   A,#06H,S2S4S7
         JNB    CROSSE1.5,$+9
         LCALL  C1SL2_SL4
         LJMP   MAIN_SYS
         JNB    CROSSE2.5,$+12
         LCALL  C2SL2_SL4
         LJMP   MAIN_SYS
         LJMP   NMS2S4
         JNB    CROSSE3.5,$-3
         LCALL  C3SL2_SL4
         LJMP   MAIN_SYS
S2S4S7:  CJNE   A,#07H,S2S4S8
         JNB    CROSSE1.6,$+9
         LCALL  C1SL2_SL4
         LJMP   MAIN_SYS
         JNB    CROSSE2.6,$+12
         LCALL  C2SL2_SL4
         LJMP   MAIN_SYS
         LJMP   NMS2S4
         JNB    CROSSE3.6,$-3
         LCALL  C3SL2_SL4
         LJMP   MAIN_SYS
S2S4S8:  CJNE   A,#08H,$-3
         JNB    CROSSE1.7,$+9
         LCALL  C1SL2_SL4
         LJMP   MAIN_SYS
         JNB    CROSSE2.7,$+12
         LCALL  C2SL2_SL4
         LJMP   MAIN_SYS
         LJMP   NMS2S4
         JNB    CROSSE3.7,$-3
         LCALL  C3SL2_SL4
         LJMP   MAIN_SYS
NMS2S4:  JB     CROSS_EXT.0,$+9
         LCALL  C1SL2_SL4
         LJMP   MAIN_SYS
         JB     CROSS_EXT.1,$+9
         LCALL  C2SL2_SL4
         LJMP   MAIN_SYS
         JB     CROSS_EXT.2,$+9    ;IF NOT ENOUGH CROSSP.
         LCALL  C3SL2_SL4          ;SEND BUSY TONE
         LJMP   MAIN_SYS
         LCALL  ONBT_SL2
         LCALL  ONBT_SL4
         LJMP   MAIN_SYS
         LJMP   CCS2S6
CCS2S5:  CJNE   A,#05H,$-3
         MOV    R7,#00H
         LCALL  RBT_SL5
         LCALL  OFF_DFS5
         MOV    A,FNC1
         ANL    A,#10H
         CJNE   A,#10H,CF2S2S5
         MOV    A,EXT5
         CJNE   A,#09H,$+9
         LCALL  CTK1_SL2
         LJMP   MAIN_SYS
         CJNE   A,#0AH,$+6         ;IF NOT TK1 OR TK2,NORMAL CONNECT
         LCALL  CTK2_SL2
         LJMP   NMS2S5
CF2S2S5: MOV    A,FNC2
         ANL    A,#10H
         CJNE   A,#10H,$-7
         MOV    A,EXT5
         CJNE   A,#09H,$+12
         LCALL  CTK1_SL2
         LCALL  CTK1_SL5
         LJMP   MAIN_SYS
         CJNE   A,#0AH,S2S5S1
         LCALL  CTK2_SL2
         LCALL  CTK2_SL5
         LJMP   MAIN_SYS
S2S5S1:  CJNE   A,#01H,S2S5S3
         JNB    CROSSE1.0,$+9
         LCALL  C1SL2_SL5
         LJMP   MAIN_SYS
         JNB    CROSSE2.0,$+12
         LCALL  C2SL2_SL5
         LJMP   MAIN_SYS
         LJMP   NMS2S5
         JNB    CROSSE3.0,$-3
         LCALL  C3SL2_SL5
         LJMP   MAIN_SYS
S2S5S3:  CJNE   A,#03H,S2S5S4
         JNB    CROSSE1.2,$+9
         LCALL  C1SL2_SL5
         LJMP   MAIN_SYS
         JNB    CROSSE2.2,$+12
         LCALL  C2SL2_SL5
         LJMP   MAIN_SYS
         LJMP   NMS2S5
         JNB    CROSSE3.2,$-3
         LCALL  C3SL2_SL5
         LJMP   MAIN_SYS
S2S5S4:  CJNE   A,#04H,S2S5S6
```

```
                JNB     CROSSE1.3,$+9
                LCALL   C1SL2_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL2_SL5
                LJMP    MAIN_SYS
                LJMP    NMS2S5
                JNB     CROSSE3.3,$-3
                LCALL   C3SL2_SL5
                LJMP    MAIN_SYS
S2S5S6:         CJNE    A,#06H,S2S5S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL2_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL2_SL5
                LJMP    MAIN_SYS
                LJMP    NMS2S5
                JNB     CROSSE3.5,$-3
                LCALL   C3SL2_SL5
                LJMP    MAIN_SYS
S2S5S7:         CJNE    A,#07H,S2S5S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL2_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL2_SL5
                LJMP    MAIN_SYS
                LJMP    NMS2S5
                JNB     CROSSE3.6,$-3
                LCALL   C3SL2_SL5
                LJMP    MAIN_SYS
S2S5S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL2_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL2_SL5
                LJMP    MAIN_SYS
                LJMP    NMS2S5
                JNB     CROSSE3.7,$-3
                LCALL   C3SL2_SL5
                LJMP    MAIN_SYS
NMS2S5:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL2_SL5
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL2_SL5
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL2_SL5           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL2
                LCALL   ONBT_SL5
                LJMP    MAIN_SYS
                LJMP    CCS2S7
CCS2S6:         CJNE    A,#06H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL6
                LCALL   OFF_DFS6
                MOV     A,FNC1
                ANL     A,#20H
                CJNE    A,#20H,CF2S2S6
                MOV     A,EXT6
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL2
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL2
                LJMP    NMS2S6
CF2S2S6:        MOV     A,FNC1
                ANL     A,#20H
                CJNE    A,#20H,$-7
                MOV     A,EXT6
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL2
                LCALL   CTK1_SL6
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S2S6S1
                LCALL   CTK2_SL2
                LCALL   CTK2_SL6
                LJMP    MAIN_SYS
S2S6S1:         CJNE    A,#01H,S2S6S3
                JNB     CROSSE1.0,$+9
                LCALL   C1SL2_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL2_SL6
                LJMP    MAIN_SYS
                LJMP    NMS2S6
                JNB     CROSSE3.0,$-3
                LCALL   C3SL2_SL6
                LJMP    MAIN_SYS
S2S6S3:         CJNE    A,#03H,S2S6S4
                JNB     CROSSE1.2,$+9
                LCALL   C1SL2_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.2,$+12
                LCALL   C2SL2_SL6
                LJMP    MAIN_SYS
                LJMP    NMS2S6
                JNB     CROSSE3.2,$-3
                LCALL   C3SL2_SL6
                LJMP    MAIN_SYS
S2S6S4:         CJNE    A,#04H,S2S6S5
                JNB     CROSSE1.3,$+9
```

```
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            LJMP    NMS2S6
            JNB     CROSSE3.3,$-3
            LCALL   C3SL2_SL6
            LJMP    MAIN_SYS
S2S6S5:     CJNE    A,#05H,S2S6S7
            JNB     CROSSE1.4,$+9
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            LJMP    NMS2S6
            JNB     CROSSE3.4,$-3
            LCALL   C3SL2_SL6
            LJMP    MAIN_SYS
S2S6S7:     CJNE    A,#07H,S2S6S8
            JNB     CROSSE1.6,$+9
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            LJMP    NMS2S6
            JNB     CROSSE3.6,$-3
            LCALL   C3SL2_SL6
            LJMP    MAIN_SYS
S2S6S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            LJMP    NMS2S6
            JNB     CROSSE3.7,$-3
            LCALL   C3SL2_SL6
            LJMP    MAIN_SYS
NMS2S6:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL2
            LCALL   ONBT_SL6
            LJMP    MAIN_SYS
            LJMP    CCS2S8
CCS2S7:     CJNE    A,#07H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL7
            LCALL   OFF_DFS7
            MOV     A,FNC1
            ANL     A,#40H
            CJNE    A,#40H,CF2S2S7
            MOV     A,EXT7
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL2
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL2
            LJMP    NMS2S7
CF2S2S7:    MOV     A,FNC2
            ANL     A,#40H
            CJNE    A,#40H,$-7
            MOV     A,EXT7
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL2
            LCALL   CTK1_SL7
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S2S7S1
            LCALL   CTK2_SL2
            LCALL   CTK2_SL7
            LJMP    MAIN_SYS
S2S7S1:     CJNE    A,#01H,S2S7S3
            JNB     CROSSE1.0,$+9
            LCALL   C1SL2_SL7
            LJMP    MAIN_SYS
            JNB     CROSSE2.0,$+12
            LCALL   C2SL2_SL7
            LJMP    MAIN_SYS
            LJMP    NMS2S7
            JNB     CROSSE3.0,$-3
            LCALL   C3SL2_SL7
            LJMP    MAIN_SYS
S2S7S3:     CJNE    A,#03H,S2S7S4
            JNB     CROSSE1.2,$+9
            LCALL   C1SL2_SL7
            LJMP    MAIN_SYS
            JNB     CROSSE2.2,$+12
            LCALL   C2SL2_SL7
            LJMP    MAIN_SYS
            LJMP    NMS2S7
            JNB     CROSSE3.2,$-3
            LCALL   C3SL2_SL7
            LJMP    MAIN_SYS
S2S7S4:     CJNE    A,#04H,S2S7S5
            JNB     CROSSE1.3,$+9
            LCALL   C1SL2_SL7
```

```
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL2_SL7
                LJMP    MAIN_SYS
                LJMP    NMS2S7
                JNB     CROSSE3.3,$-3
                LCALL   C3SL2_SL7
                LJMP    MAIN_SYS
S2S7S5:         CJNE    A,#05H,S2S7S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL2_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL2_SL7
                LJMP    MAIN_SYS
                LJMP    NMS2S7
                JNB     CROSSE3.4,$-3
                LCALL   C3SL2_SL7
                LJMP    MAIN_SYS
S2S7S6:         CJNE    A,#06H,S2S7S8
                JNB     CROSSE1.5,$+9
                LCALL   C1SL2_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL2_SL7
                LJMP    MAIN_SYS
                LJMP    NMS2S7
                JNB     CROSSE3.5,$-3
                LCALL   C3SL2_SL7
                LJMP    MAIN_SYS
S2S7S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL2_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL2_SL7
                LJMP    MAIN_SYS
                LJMP    NMS2S7
                JNB     CROSSE3.7,$-3
                LCALL   C3SL2_SL7
                LJMP    MAIN_SYS
NMS2S7:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL2_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL2_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL2_SL7           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL2
                LCALL   ONBT_SL7
                LJMP    MAIN_SYS
```

```
                LJMP    MAIN_SYS
CCS2S8:         CJNE    A,#08H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL8
                LCALL   OFF_DFS8
                MOV     A,FNC1
                ANL     A,#80H
                CJNE    A,#80H,CF2S2S8
                MOV     A,EXT8
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL2
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL2
                LJMP    NMS2S8
CF2S2S8:        MOV     A,FNC2
                ANL     A,#80H
                CJNE    A,#80H,$-7
                MOV     A,EXT8
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL2
                LCALL   CTK1_SL8
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S2S8S1
                LCALL   CTK2_SL2
                LCALL   CTK2_SL8
                LJMP    MAIN_SYS
S2S8S1:         CJNE    A,#01H,S2S8S3
                JNB     CROSSE1.0,$+9
                LCALL   C1SL2_SL8
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL2_SL8
                LJMP    MAIN_SYS
                LJMP    NMS2S8
                JNB     CROSSE3.0,$-3
                LCALL   C3SL2_SL8
                LJMP    MAIN_SYS
S2S8S3:         CJNE    A,#03H,S2S8S4
                JNB     CROSSE1.2,$+9
                LCALL   C1SL2_SL8
                LJMP    MAIN_SYS
                JNB     CROSSE2.2,$+12
                LCALL   C2SL2_SL8
                LJMP    MAIN_SYS
                LJMP    NMS2S8
                JNB     CROSSE3.2,$-3
                LCALL   C3SL2_SL8
                LJMP    MAIN_SYS
S2S8S4:         CJNE    A,#04H,S2S8S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL2_SL8
                LJMP    MAIN_SYS
```

```
            JNB     CROSSE2.3,$+12
            LCALL   C2SL2_SL8
            LJMP    MAIN_SYS
            LJMP    NMS2S8
            JNB     CROSSE3.3,$-3
            LCALL   C3SL2_SL8
            LJMP    MAIN_SYS
S2S8S5:     CJNE    A,#05H,S2S8S6
            JNB     CROSSE1.4,$+9
            LCALL   C1SL2_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.4,$+12
            LCALL   C2SL2_SL8
            LJMP    MAIN_SYS
            LJMP    NMS2S8
            JNB     CROSSE3.4,$-3
            LCALL   C3SL2_SL8
            LJMP    MAIN_SYS
S2S8S6:     CJNE    A,#06H,S2S8S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL2_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL2_SL8
            LJMP    MAIN_SYS
            LJMP    NMS2S8
            JNB     CROSSE3.5,$-3
            LCALL   C3SL2_SL8
            LJMP    MAIN_SYS
S2S8S7:     CJNE    A,#07H,$-3
            JNB     CROSSE1.6,$+9
            LCALL   C1SL2_SL8
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL2_SL8
            LJMP    MAIN_SYS
            LJMP    NMS2S8
            JNB     CROSSE3.6,$-3
            LCALL   C3SL2_SL8
            LJMP    MAIN_SYS
NMS2S8:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL2
            LCALL   ONBT_SL8
            LJMP    MAIN_SYS
GDT_SL2:    MOV     R7,#0FFH
            LCALL   DT_SL2              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E2:    CJNE    A,DTMF3,C_DF4E2
            MOV     R7,#0FFH
            LCALL   DF3_SL2
            LJMP    MAIN_SYS
C_DF4E2:    CJNE    A,DTMF4,C_DF5E2
            MOV     R7,#0FFH
            LCALL   DF4_SL2
            LJMP    MAIN_SYS
C_DF5E2:    CJNE    A,DTMF5,C_DF6E2
            MOV     R7,#0FFH
            LCALL   DF5_SL2
            LJMP    MAIN_SYS
C_DF6E2:    CJNE    A,DTMF6,NO_DF_E2
            MOV     R7,#0FFH
            LCALL   DF6_SL2
            LJMP    MAIN_SYS
NO_DF_E2:   SETB    WAIT_DF.1
            LJMP    MAIN_SYS
;*****************************************************
CHK_SL3:    CJNE    A,#04H,JS4          ; HOFF = HOLD OFF
            SJMP    SL3
JS4:        LJMP    CHK_SL4
SL3:        JNB     S_EXT.2,OFF_HK3
            LJMP    ON_HK3
;***************  EXT3 HOLDING OFF ******************
OFF_HK3:    LCALL   DELAY90
            SETB    S_EXT.2
            MOV     P_NOSL3,#0FFH       ;CLEAR BUFFER OF NUMBERS
            CLR     WAIT_DF.2           ;CLEAR WAITING DTMF DEC.
            ANL     FNC1,#0FBH          ;CLEAR FUNCTION
            ANL     FNC2,#0FBH
            ANL     FNC3,#0FBH
            ANL     OUT_T1,#0FBH
            ANL     OUT_T2,#0FBH
            JB      EXT_DT.2,CLR_DT3
            JB      EXT_BT.2,CLR_BT3
            JB      EXT_RBT.2,CLR_RBT3
            LCALL   OFF_DFS3
            LJMP    OFF_LS3
CLR_DT3:    MOV     R7,#00H
            LCALL   DT_SL3
            LJMP    OFF_LS3
CLR_BT3:    MOV     R7,#00H
            LCALL   BT_SL3
            LJMP    OFF_LS3
CLR_RBT3:   MOV     R7,#00H
            LCALL   RBT_SL3
            MOV     A,R_EXT1
            CJNE    A,#03H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
```

```
            MOV     A,R_EXT2
            CJNE    A,#03H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#03H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#03H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#03H,$+9
            LCALL   OFF_R6
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#03H,$+9
            LCALL   OFF_R7
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#03H,$+6
            LCALL   OFF_R8

;********* OFF LINE THAT CONNECT WITH EXT.3 ***********
OFF_LS3:    MOV     A,#09H
            CJNE    A,EXT3,CHKT2E3
            MOV     R7,#00H
            LCALL   TK1_SL3
            MOV     A,TKE1
            ANL     A,#04H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E3:    MOV     A,#0AH
            CJNE    A,EXT3,CHKC1E3
            MOV     R7,#00H
            LCALL   TK2_SL3
            MOV     A,TKE2
            ANL     A,#04H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E3:    JNB     CROSSE1.2,CHKC2E3
            MOV     R7,#00H
            LCALL   CROSS1_3
            LCALL   CHK_CR1
CHKC2E3:    JNB     CROSSE2.2,CHKC3E3
            MOV     R7,#00H
            LCALL   CROSS2_3
            LCALL   CHK_CR2
CHKC3E3:    JNB     CROSSE3.2,OFF_E3END
            MOV     R7,#00H
            LCALL   CROSS3_3
            LCALL   CHK_CR3
OFF_E3END:  LJMP    MAIN_SYS
;************** EXT3 HOLD ON ******************
ON_HK3:     LCALL   DELAY90
            CLR     S_EXT.2
            JB      EXT_RG.2,CRSL3
            LJMP    GDT_SL3
CRSL3:      MOV     R7,#00H
            LCALL   OFF_R3
            MOV     A,R_EXT3
            CJNE    A,#09H,CCS3T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL3
            LJMP    MAIN_SYS
CCS3T2:     CJNE    A,#0AH,CCS3S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL3
            LJMP    MAIN_SYS
            LJMP    CCS3S2
CCS3S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
            MOV     A,FNC1
            ANL     A,#01H
            CJNE    A,#01H,CF2S3S1
            MOV     A,EXT1
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL3
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL3
            LJMP    NMS3S1
CF2S3S1:    MOV     A,FNC2
            ANL     A,#01H
            CJNE    A,#01H,$-7
            MOV     A,EXT1
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL3
            LCALL   CTK1_SL1
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S3S1S2
            LCALL   CTK2_SL3
            LCALL   CTK2_SL1
            LJMP    MAIN_SYS
S3S1S2:     CJNE    A,#02H,S3S1S4
            JNB     CROSSE1.1,$+9
            LCALL   C1SL1_SL3
            LJMP    MAIN_SYS
            JNB     CROSSE2.1,$+12
```

```
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.1,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S3S1S4:         CJNE    A,#04H,S3S1S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.3,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S3S1S5:         CJNE    A,#05H,S3S1S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.4,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S3S1S6:         CJNE    A,#06H,S3S1S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.5,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S3S1S7:         CJNE    A,#07H,S3S1S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.6,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
S3S1S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                LJMP    NMS3S1
                JNB     CROSSE3.7,$-3
                LCALL   C3SL1_SL3
                LJMP    MAIN_SYS
NMS3S1:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL1_SL3
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL1_SL3
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9 ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL1_SL3       ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL3
                LCALL   ONBT_SL1
                LJMP    MAIN_SYS
                LJMP    CCS3S4
CCS3S2:         CJNE    A,#02H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL2
                LCALL   OFF_DFS2
                MOV     A,FNC1
                ANL     A,#02H
                CJNE    A,#02H,CF2S3S2
                MOV     A,EXT2
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL3
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL3
                LJMP    NMS3S2
CF2S3S2:        MOV     A,FNC2
                ANL     A,#02H
                CJNE    A,#02H,$-7
                MOV     A,EXT2
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL3
                LCALL   CTK1_SL2
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S3S2S1
                LCALL   CTK2_SL3
                LCALL   CTK2_SL2
                LJMP    MAIN_SYS
S3S2S1:         CJNE    A,#01H,S3S2S4
                JNB     CROSSE1.0,$+9
                LCALL   C1SL2_SL3
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL2_SL3
```

```
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.0,$-3
        LCALL   C3SL2_SL3
        LJMP    MAIN_SYS
S3S2S4: CJNE    A,#04H,S3S2S5
        JNB     CROSSE1.3,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JNB     CROSSE2.3,$+12
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.3,$-3
        LCALL   C3SL1_SL3
        LJMP    MAIN_SYS
S3S2S5: CJNE    A,#05H,S3S2S6
        JNB     CROSSE1.4,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JNB     CROSSE2.4,$+12
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.4,$-3
        LCALL   C3SL2_SL3
        LJMP    MAIN_SYS
S3S2S6: CJNE    A,#06H,S3S2S7
        JNB     CROSSE1.5,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JNB     CROSSE2.5,$+12
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.5,$-3
        LCALL   C3SL2_SL3
        LJMP    MAIN_SYS
S3S2S7: CJNE    A,#07H,S3S2S8
        JNB     CROSSE1.6,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JNB     CROSSE2.6,$+12
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.6,$-3
        LCALL   C3SL2_SL3
        LJMP    MAIN_SYS
S3S2S8: CJNE    A,#08H,$-3
        JNB     CROSSE1.7,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JNB     CROSSE2.7,$+12
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        LJMP    NMS3S2
        JNB     CROSSE3.7,$-3
        LCALL   C3SL2_SL3
        LJMP    MAIN_SYS
NMS3S2: JB      CROSS_EXT.0,$+9
        LCALL   C1SL2_SL3
        LJMP    MAIN_SYS
        JB      CROSS_EXT.1,$+9
        LCALL   C2SL2_SL3
        LJMP    MAIN_SYS
        JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
        LCALL   C3SL2_SL3           ;SEND BUSY TONE
        LJMP    MAIN_SYS
        LCALL   ONBT_SL3
        LCALL   ONBT_SL2
        LJMP    MAIN_SYS
        LJMP    CCS3S5
CCS3S4: CJNE    A,#04H,$-3
        MOV     R7,#00H
        LCALL   RBT_SL4
        LCALL   OFF_DFS4
        MOV     A,FNC1
        ANL     A,#08H
        CJNE    A,#08H,CF2S3S4
        MOV     A,EXT4
        CJNE    A,#09H,$+9
        LCALL   CTK1_SL3
        LJMP    MAIN_SYS
        CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
        LCALL   CTK2_SL3
        LJMP    NMS3S4
CF2S3S4: MOV    A,FNC2
        ANL     A,#08H
        CJNE    A,#08H,$-7
        MOV     A,EXT4
        CJNE    A,#09H,$+12
        LCALL   CTK1_SL3
        LCALL   CTK1_SL4
        LJMP    MAIN_SYS
        CJNE    A,#0AH,S3S4S1
        LCALL   CTK2_SL3
        LCALL   CTK2_SL4
        LJMP    MAIN_SYS
S3S4S1: CJNE    A,#01H,S3S4S2
        JNB     CROSSE1.0,$+9
        LCALL   C1SL3_SL4
        LJMP    MAIN_SYS
        JNB     CROSSE2.0,$+12
        LCALL   C2SL3_SL4
        LJMP    MAIN_SYS
```

```
                LJMP    NMS3S4
                JNB     CROSSE3.0,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
S3S4S2:         CJNE    A,#02H,S3S4S5
                JNB     CROSSE1.1,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                LJMP    NMS3S4
                JNB     CROSSE3.1,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
S3S4S5:         CJNE    A,#05H,S3S4S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                LJMP    NMS3S4
                JNB     CROSSE3.4,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
S3S4S6:         CJNE    A,#06H,S3S4S7
                JNB     CROSSE1.5,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                LJMP    NMS3S4
                JNB     CROSSE3.5,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
S3S4S7:         CJNE    A,#07H,S3S4S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                LJMP    NMS3S4
                JNB     CROSSE3.6,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
S3S4S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                LJMP    NMS3S4
                JNB     CROSSE3.7,$-3
                LCALL   C3SL3_SL4
                LJMP    MAIN_SYS
NMS3S4:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL3_SL4
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL3_SL4
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL3_SL4           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL3
                LCALL   ONBT_SL4
                LJMP    MAIN_SYS
                LJMP    CCS3S6
CCS3S5:         CJNE    A,#05H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL5
                LCALL   OFF_DFS5
                MOV     A,FNC1
                ANL     A,#10H
                CJNE    A,#10H,CF2S3S5
                MOV     A,EXT5
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL3
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL3
                LJMP    NMS3S5
CF2S3S5:        MOV     A,FNC2
                ANL     A,#10H
                CJNE    A,#10H,$-7
                MOV     A,EXT5
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL3
                LCALL   CTK1_SL5
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S3S5S1
                LCALL   CTK2_SL3
                LCALL   CTK2_SL5
                LJMP    MAIN_SYS
S3S5S1:         CJNE    A,#01H,S3S5S2
                JNB     CROSSE1.0,$+9
                LCALL   C1SL3_SL5
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL3_SL5
                LJMP    MAIN_SYS
                LJMP    NMS3S5
```

```
            JNB     CROSSE3.0,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
S3S5S2:     CJNE    A,#02H,S3S5S4
            JNB     CROSSE1.1,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JNB     CROSSE2.1,$+12
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            LJMP    NMS3S5
            JNB     CROSSE3.1,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
S3S5S4:     CJNE    A,#04H,S3S5S6
            JNB     CROSSE1.3,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JNB     CROSSE2.3,$+12
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            LJMP    NMS3S5
            JNB     CROSSE3.3,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
S3S5S6:     CJNE    A,#06H,S3S5S7
            JNB     CROSSE1.5,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JNB     CROSSE2.5,$+12
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            LJMP    NMS3S5
            JNB     CROSSE3.5,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
S3S5S7:     CJNE    A,#07H,S3S5S8
            JNB     CROSSE1.6,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JNB     CROSSE2.6,$+12
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            LJMP    NMS3S5
            JNB     CROSSE3.6,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
S3S5S8:     CJNE    A,#08H,$-3
            JNB     CROSSE1.7,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JNB     CROSSE2.7,$+12
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            LJMP    NMS3S5
            JNB     CROSSE3.7,$-3
            LCALL   C3SL3_SL5
            LJMP    MAIN_SYS
NMS3S5:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL3_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL3
            LCALL   ONBT_SL5
            LJMP    MAIN_SYS
            LJMP    CCS3S7
CCS3S6:     CJNE    A,#06H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL6
            LCALL   OFF_DFS6
            MOV     A,FNC1
            ANL     A,#20H
            CJNE    A,#20H,CF2S3S6
            MOV     A,EXT6
            CJNE    A,#09H,$+9
            LCALL   CTK1_SL3
            LJMP    MAIN_SYS
            CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
            LCALL   CTK2_SL3
            LJMP    NMS3S6
CF2S3S6:    MOV     A,FNC2
            ANL     A,#20H
            CJNE    A,#20H,$-7
            MOV     A,EXT6
            CJNE    A,#09H,$+12
            LCALL   CTK1_SL3
            LCALL   CTK1_SL6
            LJMP    MAIN_SYS
            CJNE    A,#0AH,S3S6S1
            LCALL   CTK2_SL3
            LCALL   CTK2_SL6
            LJMP    MAIN_SYS
S3S6S1:     CJNE    A,#01H,S3S6S2
            JNB     CROSSE1.0,$+9
            LCALL   C1SL3_SL6
            LJMP    MAIN_SYS
            JNB     CROSSE2.0,$+12
            LCALL   C2SL3_SL6
            LJMP    MAIN_SYS
            LJMP    NMS3S6
            JNB     CROSSE3.0,$-3
```

```
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
S3S6S2:         CJNE    A,#02H,S3S6S4
                JNB     CROSSE1.1,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                LJMP    NMS3S6
                JNB     CROSSE3.1,$-3
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
S3S6S4:         CJNE    A,#04H,S3S6S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                LJMP    NMS3S6
                JNB     CROSSE3.3,$-3
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
S3S6S5:         CJNE    A,#05H,S3S6S7
                JNB     CROSSE1.4,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                LJMP    NMS3S6
                JNB     CROSSE3.4,$-3
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
S3S6S7:         CJNE    A,#07H,S3S6S8
                JNB     CROSSE1.6,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.6,$+12
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                LJMP    NMS3S6
                JNB     CROSSE3.6,$-3
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
S3S6S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                LJMP    NMS3S6
                JNB     CROSSE3.7,$-3
                LCALL   C3SL3_SL6
                LJMP    MAIN_SYS
NMS3S6:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL3_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL3_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL3_SL6           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL3
                LCALL   ONBT_SL6
                LJMP    MAIN_SYS
                LJMP    CCS3S8
CCS3S7:         CJNE    A,#07H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL7
                LCALL   OFF_DFS7
                MOV     A,FNC1
                ANL     A,#40H
                CJNE    A,#40H,CF2S3S7
                MOV     A,EXT7
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL3
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL3
                LJMP    NMS3S7
CF2S3S7:        MOV     A,FNC2
                ANL     A,#40H
                CJNE    A,#40H,$-7
                MOV     A,EXT7
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL3
                LCALL   CTK1_SL7
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S3S7S1
                LCALL   CTK2_SL3
                LCALL   CTK2_SL7
                LJMP    MAIN_SYS
S3S7S1:         CJNE    A,#01H,S3S7S2
                JNB     CROSSE1.0,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.0,$-3
                LCALL   C3SL3_SL7
```

```
                LJMP    MAIN_SYS
S3S7S2:         CJNE    A,#02H,S3S7S4
                JNB     CROSSE1.1,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.1,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.1,$-3
                LCALL   C3SL3_SL7
                LJMP    MAIN_SYS
S3S7S4:         CJNE    A,#04H,S3S7S5
                JNB     CROSSE1.3,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.3,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.3,$-3
                LCALL   C3SL3_SL7
                LJMP    MAIN_SYS
S3S7S5:         CJNE    A,#05H,S3S7S6
                JNB     CROSSE1.4,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.4,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.4,$-3
                LCALL   C3SL3_SL7
                LJMP    MAIN_SYS
S3S7S6:         CJNE    A,#06H,S3S7S8
                JNB     CROSSE1.5,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.5,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.5,$-3
                LCALL   C3SL3_SL7
                LJMP    MAIN_SYS
S3S7S8:         CJNE    A,#08H,$-3
                JNB     CROSSE1.7,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JNB     CROSSE2.7,$+12
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                LJMP    NMS3S7
                JNB     CROSSE3.7,$-3
                LCALL   C3SL3_SL7
                LJMP    MAIN_SYS
NMS3S7:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL3_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL3_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL3_SL7           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL3
                LCALL   ONBT_SL7
                LJMP    MAIN_SYS
                LJMP    MAIN_SYS
CCS3S8:         CJNE    A,#08H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL8
                LCALL   OFF_DFS8
                MOV     A,FNC1
                ANL     A,#80H
                CJNE    A,#80H,CF2S3S8
                MOV     A,EXT8
                CJNE    A,#09H,$+9
                LCALL   CTK1_SL3
                LJMP    MAIN_SYS
                CJNE    A,#0AH,$+6      ;IF NOT TK1 OR TK2,NORMAL CONNECT
                LCALL   CTK2_SL3
                LJMP    NMS2S8
CF2S3S8:        MOV     A,FNC2
                ANL     A,#80H
                CJNE    A,#80H,$-7
                MOV     A,EXT8
                CJNE    A,#09H,$+12
                LCALL   CTK1_SL3
                LCALL   CTK1_SL8
                LJMP    MAIN_SYS
                CJNE    A,#0AH,S3S8S1
                LCALL   CTK2_SL3
                LCALL   CTK2_SL8
                LJMP    MAIN_SYS
S3S8S1:         CJNE    A,#01H,S3S8S2
                JNB     CROSSE1.0,$+9
                LCALL   C1SL3_SL8
                LJMP    MAIN_SYS
                JNB     CROSSE2.0,$+12
                LCALL   C2SL3_SL8
                LJMP    MAIN_SYS
                LJMP    NMS2S8
                JNB     CROSSE3.0,$-3
                LCALL   C3SL3_SL8
                LJMP    MAIN_SYS
```

```
S3S8S2:  CJNE   A,#02H,S3S8S4
         JNB    CROSSE1.1,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JNB    CROSSE2.1,$+12
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         LJMP   NMS2S8
         JNB    CROSSE3.1,$-3
         LCALL  C3SL3_SL8
         LJMP   MAIN_SYS
S3S8S4:  CJNE   A,#04H,S3S8S5
         JNB    CROSSE1.3,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JNB    CROSSE2.3,$+12
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         LJMP   NMS2S8
         JNB    CROSSE3.3,$-3
         LCALL  C3SL3_SL8
         LJMP   MAIN_SYS
S3S8S5:  CJNE   A,#05H,S3S8S6
         JNB    CROSSE1.4,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JNB    CROSSE2.4,$+12
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         LJMP   NMS2S8
         JNB    CROSSE3.4,$-3
         LCALL  C3SL3_SL8
         LJMP   MAIN_SYS
S3S8S6:  CJNE   A,#06H,S3S8S7
         JNB    CROSSE1.5,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JNB    CROSSE2.5,$+12
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         LJMP   NMS2S8
         JNB    CROSSE3.5,$-3
         LCALL  C3SL3_SL8
         LJMP   MAIN_SYS
S3S8S7:  CJNE   A,#07H,$-3
         JNB    CROSSE1.6,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JNB    CROSSE2.6,$+12
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         LJMP   NMS2S8
         JNB    CROSSE3.6,$-3
         LCALL  C3SL3_SL8
         LJMP   MAIN_SYS
NMS3S8:  JB     CROSS_EXT.0,$+9
         LCALL  C1SL3_SL8
         LJMP   MAIN_SYS
         JB     CROSS_EXT.1,$+9
         LCALL  C2SL3_SL8
         LJMP   MAIN_SYS
         JB     CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
         LCALL  C3SL3_SL8           ;SEND BUSY TONE
         LJMP   MAIN_SYS
         LCALL  ONBT_SL3
         LCALL  ONBT_SL8
         LJMP   MAIN_SYS
GDT_SL3: MOV    R7,#0FFH
         LCALL  DT_SL3              ;GIVE DIAL TONE
         MOV    A,#00H
C_DF3E3: CJNE   A,DTMF3,C_DF4E3
         MOV    R7,#0FFH
         LCALL  DF3_SL3
         LJMP   MAIN_SYS
C_DF4E3: CJNE   A,DTMF4,C_DF5E3
         MOV    R7,#0FFH
         LCALL  DF4_SL3
         LJMP   MAIN_SYS
C_DF5E3: CJNE   A,DTMF5,C_DF6E3
         MOV    R7,#0FFH
         LCALL  DF5_SL3
         LJMP   MAIN_SYS
C_DF6E3: CJNE   A,DTMF6,NO_DF_E3
         MOV    R7,#0FFH
         LCALL  DF6_SL3
         LJMP   MAIN_SYS

NO_DF_E3: SETB  WAIT_DF.2
         LJMP   MAIN_SYS
;*******************************************************
CHK_SL4: CJNE   A,#08H,JS5          ; HOFF = HOLD OFF
         SJMP   SL4
JS5:     LJMP   CHK_SL5
SL4:     JNB    S_EXT.3,OFF_HK4
         LJMP   ON_HK4
;***************  EXT4 HOLDING OFF ******************
OFF_HK4: LCALL  DELAY90
         SETB   S_EXT.3
         MOV    P_NOSL4,#0FFH       ;CLEAR BUFFER OF NUMBERS
         CLR    WAIT_DF.3           ;CLEAR WAITING DTMF DEC.
         ANL    FNC1,#0F7H          ;CLEAR FUNCTION
         ANL    FNC2,#0F7H
         ANL    FNC3,#0F7H
         ANL    OUT_T1,#0F7H
         ANL    OUT_T2,#0F7H
         JB     EXT_DT.3,CLR_DT4
```

```
            JB      EXT_BT.3,CLR_BT4
            JB      EXT_RBT.3,CLR_RBT4
            LCALL   OFF_DFS4
            LJMP    OFF_LS4
CLR_DT4:    MOV     R7,#00H
            LCALL   DT_SL4
            LJMP    OFF_LS4
CLR_BT4:    MOV     R7,#00H
            LCALL   BT_SL4
            LJMP    OFF_LS4
CLR_RBT4:   MOV     R7,#00H
            LCALL   RBT_SL4
            MOV     A,R_EXT1
            CJNE    A,#04H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
            MOV     A,R_EXT2
            CJNE    A,#04H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#04H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#04H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#04H,$+9
            LCALL   OFF_R6
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#04H,$+9
            LCALL   OFF_R7
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#04H,$+6
            LCALL   OFF_R8
;********* OFF LINE THAT CONNECT WITH EXT.4 **********
OFF_LS4:    MOV     A,#09H
            CJNE    A,EXT4,CHKT2E4
            MOV     R7,#00H
            LCALL   TK1_SL4
            MOV     A,TKE1
            ANL     A,#08H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E4:    MOV     A,#0AH
            CJNE    A,EXT4,CHKC1E4
            MOV     R7,#00H
            LCALL   TK2_SL4
            MOV     A,TKE2
            ANL     A,#08H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E4:    JNB     CROSSE1.3,CHKC2E4
            MOV     R7,#00H
            LCALL   CROSS1_4
            LCALL   CHK_CR1
CHKC2E4:    JNB     CROSSE2.3,CHKC3E4
            MOV     R7,#00H
            LCALL   CROSS2_4
            LCALL   CHK_CR2
CHKC3E4:    JNB     CROSSE3.3,OFF_E4END
            MOV     R7,#00H
            LCALL   CROSS3_4
            LCALL   CHK_CR3
OFF_E4END:  LJMP    MAIN_SYS
;************* EXT4.HOLD ON *****************
ON_HK4:     LCALL   DELAY90
            CLR     S_EXT.3
            JB      EXT_RG.3,H_RG4
            LJMP    GDT_SL4
H_RG4:      MOV     R7,#00H
            LCALL   OFF_R4
            MOV     A,R_EXT4
            CJNE    A,#09H,CCS4T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL4
            LJMP    MAIN_SYS
CCS4T2:     CJNE    A,#0AH,CCS4S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL4
            LJMP    MAIN_SYS
            LJMP    CCS4S2
CCS4S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
NMS4S1:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL4           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL1
            LJMP    MAIN_SYS
```

```
            LJMP    CCS4S3
CCS4S2:     CJNE    A,#02H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL2
            LCALL   OFF_DFS2
NMS4S2:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL4           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL2
            LJMP    MAIN_SYS
            LJMP    CCS4S5
CCS4S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
NMS4S3:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL3_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL3_SL4
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL3_SL4           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL3
            LJMP    MAIN_SYS
            LJMP    CCS4S6
CCS4S5:     CJNE    A,#05H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL5
            LCALL   OFF_DFS5
NMS4S5:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL5
            LJMP    MAIN_SYS
            LJMP    CCS4S7
CCS4S6:     CJNE    A,#06H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL6
            LCALL   OFF_DFS6
NMS4S6:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL6
            LJMP    MAIN_SYS
            LJMP    CCS4S8
CCS4S7:     CJNE    A,#07H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL7
            LCALL   OFF_DFS7
NMS4S7:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL7
            LJMP    MAIN_SYS
CCS4S8:     CJNE    A,#08H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL8
            LCALL   OFF_DFS8
NMS4S8:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL4
            LCALL   ONBT_SL8
            LJMP    MAIN_SYS
GDT_SL4:    MOV     R7,#0FFH
            LCALL   DT_SL4              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E4:    CJNE    A,DTMF3,C_DF4E4
            MOV     R7,#0FFH
```

```
          LCALL   DF3_SL4
          LJMP    MAIN_SYS
C_DF4E4:  CJNE    A,DTMF4,C_DF5E4
          MOV     R7,#0FFH
          LCALL   DF4_SL4
          LJMP    MAIN_SYS
C_DF5E4:  CJNE    A,DTMF5,C_DF6E4
          MOV     R7,#0FFH
          LCALL   DF5_SL4
          LJMP    MAIN_SYS
C_DF6E4:  CJNE    A,DTMF6,NO_DF_E4
          MOV     R7,#0FFH
          LCALL   DF6_SL4
          LJMP    MAIN_SYS

NO_DF_E4: SETB    WAIT_DF.3
          LJMP    MAIN_SYS
;*********************************************************
CHK_SL5:  CJNE    A,#10H,JS6          ; HOFF = HOLD OFF
          SJMP    SL5
JS6:      LJMP    CHK_SL6
SL5:      JNB     S_EXT.4,OFF_HK5
          LJMP    ON_HK5
;***************  EXT5 HOLDING OFF ******************
OFF_HK5:  LCALL   DELAY90
          SETB    S_EXT.4
          MOV     P_NOSL5,#0FFH          ;CLEAR BUFFER OF NUMBERS
          CLR     WAIT_DF.4              ;CLEAR WAITING DTMF DEC.
          ANL     FNC1,#0EFH             ;CLEAR FUNCTION
          ANL     FNC2,#0EFH
          ANL     FNC3,#0EFH
          ANL     OUT_T1,#0EFH
          ANL     OUT_T2,#0EFH
          JB      EXT_DT.4,CLR_DT5
          JB      EXT_BT.4,CLR_BT5
          JB      EXT_RBT.4,CLR_RBT5
          LCALL   OFF_DFS5
          LJMP    OFF_LS5
CLR_DT5:  MOV     R7,#00H
          LCALL   DT_SL5
          LJMP    OFF_LS5
CLR_BT5:  MOV     R7,#00H
          LCALL   BT_SL5
          LJMP    OFF_LS5
CLR_RBT5: MOV     R7,#00H
          LCALL   RBT_SL5
          MOV     A,R_EXT1
          CJNE    A,#05H,$+9
          LCALL   OFF_R1
          LJMP    OFF_LS1
          MOV     A,R_EXT2
          CJNE    A,#05H,$+9
          LCALL   OFF_R2
          LJMP    OFF_LS1
          MOV     A,R_EXT3
          CJNE    A,#05H,$+9
          LCALL   OFF_R3
          LJMP    OFF_LS1
          MOV     A,R_EXT4
          CJNE    A,#05H,$+9
          LCALL   OFF_R4
          LJMP    OFF_LS1
          MOV     A,R_EXT6
          CJNE    A,#05H,$+9
          LCALL   OFF_R6
          LJMP    OFF_LS1
          MOV     A,R_EXT7
          CJNE    A,#05H,$+9
          LCALL   OFF_R7
          LJMP    OFF_LS1
          MOV     A,R_EXT8
          CJNE    A,#05H,$+6
          LCALL   OFF_R8
;********* OFF LINE THAT CONNECT WITH EXT.5 ***********
OFF_LS5:  MOV     A,#09H
          CJNE    A,EXT5,CHKT2E5
          MOV     R7,#00H
          LCALL   TK1_SL5
          MOV     A,TKE1
          ANL     A,#10H
          CJNE    A,#00H,$+8
          MOV     R7,#00H
          LCALL   TK1_HK
CHKT2E5:  MOV     A,#0AH
          CJNE    A,EXT5,CHKC1E5
          MOV     R7,#00H
          LCALL   TK2_SL5
          MOV     A,TKE2
          ANL     A,#10H
          CJNE    A,#00H,$+8
          MOV     R7,#00H
          LCALL   TK2_HK
CHKC1E5:  JNB     CROSSE1.4,CHKC2E5
          MOV     R7,#00H
          LCALL   CROSS1_5
          LCALL   CHK_CR1
CHKC2E5:  JNB     CROSSE2.4,CHKC3E5
          MOV     R7,#00H
          LCALL   CROSS2_5
          LCALL   CHK_CR2
CHKC3E5:  JNB     CROSSE3.4,OFF_E5END
          MOV     R7,#00H
          LCALL   CROSS3_5
          LCALL   CHK_CR3
OFF_E5END: LJMP   MAIN_SYS
;************* EXT5.HOLD.ON *****************
```

```
ON_HK5:     LCALL   DELAY90
            CLR     S_EXT.4
            JB      EXT_RG.4,H_RG5
            LJMP    GDT_SL5
H_RG5:      MOV     R7,#00H
            LCALL   OFF_R5
            MOV     A,R_EXT5
            CJNE    A,#09H,CCS5T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL5
            LJMP    MAIN_SYS
CCS5T2:     CJNE    A,#0AH,CCS5S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL5
            LJMP    MAIN_SYS
            LJMP    CCS5S2
CCS5S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
NMS5S1:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL1
            LJMP    MAIN_SYS
            LJMP    CCS5S3
CCS5S2:     CJNE    A,#02H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL2
            LCALL   OFF_DFS2
NMS5S2:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL2
            LJMP    MAIN_SYS
            LJMP    CCS5S4
CCS5S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
NMS5S3:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL3_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL3_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL3_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL3
            LJMP    MAIN_SYS
            LJMP    CCS5S6
CCS5S4:     CJNE    A,#04H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL4
            LCALL   OFF_DFS4
NMS5S4:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL5
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL5           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL4
            LJMP    MAIN_SYS
            LJMP    CCS5S7
CCS5S6:     CJNE    A,#06H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL6
            LCALL   OFF_DFS6
NMS5S6:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL5_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL5_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL5_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL6
            LJMP    MAIN_SYS
            LJMP    CCS5S8
CCS5S7:     CJNE    A,#07H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL7
```

```
            LCALL   OFF_DFS7
NMS5S7:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL5_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL5_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL5_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL7
            LJMP    MAIN_SYS
CCS5S8:     CJNE    A,#08H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL8
            LCALL   OFF_DFS8
NMS5S8:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL5_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL5_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL5_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL5
            LCALL   ONBT_SL8
            LJMP    MAIN_SYS
GDT_SL5:    MOV     R7,#0FFH
            LCALL   DT_SL5              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E5:    CJNE    A,DTMF3,C_DF4E5
            MOV     R7,#0FFH
            LCALL   DF3_SL5
            LJMP    MAIN_SYS
C_DF4E5:    CJNE    A,DTMF4,C_DF5E5
            MOV     R7,#0FFH
            LCALL   DF4_SL5
            LJMP    MAIN_SYS
C_DF5E5:    CJNE    A,DTMF5,C_DF6E5
            MOV     R7,#0FFH
            LCALL   DF5_SL5
            LJMP    MAIN_SYS
C_DF6E5:    CJNE    A,DTMF6,NO_DF_E5
            MOV     R7,#0FFH
            LCALL   DF6_SL5
            LJMP    MAIN_SYS

NO_DF_E5:   SETB    WAIT_DF.4
            LJMP    MAIN_SYS
;*****************************************************
CHK_SL6:    CJNE    A,#20H,JS7          ; HOFF = HOLD OFF
            SJMP    SL6
JS7:        LJMP    ChK_SL7
SL6:        JNB     S_EXT.5,OFF_HK6
            LJMP    ON_HK6
;****************  EXT6 HOLDING OFF ******************
OFF_HK6:    LCALL   DELAY90
            SETB    S_EXT.5
            MOV     P_NOSL6,#0FFH       ;CLEAR BUFFER OF NUMBERS
            CLR     WAIT_DF.5           ;CLEAR WAITING DTMF DEC.
            ANL     FNC1,#0DFH          ;CLEAR FUNCTION
            ANL     FNC2,#0DFH
            ANL     FNC3,#0DFH
            ANL     OUT_T1,#0DFH
            ANL     OUT_T2,#0DFH
            JB      EXT_DT.5,CLR_DT6
            JB      EXT_BT.5,CLR_BT6
            JB      EXT_RBT.5,CLR_RBT6
            LCALL   OFF_DFS6
            LJMP    OFF_LS6
CLR_DT6:    MOV     R7,#00H
            LCALL   DT_SL6
            LJMP    OFF_LS6
CLR_BT6:    MOV     R7,#00H
            LCALL   BT_SL6
            LJMP    OFF_LS6
CLR_RBT6:   MOV     R7,#00H
            LCALL   RBT_SL6
            MOV     A,R_EXT1
            CJNE    A,#06H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
            MOV     A,R_EXT2
            CJNE    A,#06H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#06H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#06H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#06H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#06H,$+9
            LCALL   OFF_R7
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#06H,$+6
```

```
            LCALL   OFF_R8
;********** OFF LINE THAT CONNECT WITH EXT.6 ***********
OFF_LS6:    MOV     A,#09H
            CJNE    A,EXT6,CHKT2E6
            MOV     R7,#00H
            LCALL   TK1_SL6
            MOV     A,TKE1
            ANL     A,#20H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E6:    MOV     A,#0AH
            CJNE    A,EXT6,CHKC1E6
            MOV     R7,#00H
            LCALL   TK2_SL6
            MOV     A,TKE2
            ANL     A,#20H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E6:    JNB     CROSSE1.5,CHKC2E6
            MOV     R7,#00H
            LCALL   CROSS1_6
            LCALL   CHK_CR1
CHKC2E6:    JNB     CROSSE2.5,CHKC3E6
            MOV     R7,#00H
            LCALL   CROSS2_6
            LCALL   CHK_CR2
CHKC3E6:    JNB     CROSSE3.5,OFF_E6END
            MOV     R7,#00H
            LCALL   CROSS3_6
            LCALL   CHK_CR3
OFF_E6END:  LJMP    MAIN_SYS
;************** EXT6.HOLD ON ******************
ON_HK6:     LCALL   DELAY90
            CLR     S_EXT.5
            JB      EXT_RG.5,H_RG6
            LJMP    GDT_SL6
H_RG6:      MOV     R7,#00H
            LCALL   OFF_R6
            MOV     A,R_EXT6
            CJNE    A,#09H,CCS6T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL6
            LJMP    MAIN_SYS
CCS6T2:     CJNE    A,#0AH,CCS6S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL6
            LJMP    MAIN_SYS
            LJMP    CCS6S2
CCS6S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
NMS6S1:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL6
            LCALL   ONBT_SL1
            LJMP    MAIN_SYS
            LJMP    CCS6S3
CCS6S2:     CJNE    A,#02H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL2
            LCALL   OFF_DFS2
NMS6S2:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL6
            LCALL   ONBT_SL2
            LJMP    MAIN_SYS
            LJMP    CCS6S4
CCS6S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
NMS6S3:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL3_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL3_SL6
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL3_SL6           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL6
            LCALL   ONBT_SL3
            LJMP    MAIN_SYS
            LJMP    CCS6S5
CCS6S4:     CJNE    A,#04H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL4
```

```
                LCALL   OFF_DFS4
NMS6S4:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL4_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL4_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL4_SL6           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL6
                LCALL   ONBT_SL4
                LJMP    MAIN_SYS
                LJMP    CCS6S7
CCS6S5:         CJNE    A,#05H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL5
                LCALL   OFF_DFS5
NMS6S5:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL5_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL5_SL6
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL5_SL6           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL6
                LCALL   ONBT_SL5
                LJMP    MAIN_SYS
                LJMP    CCS6S8
CCS6S7:         CJNE    A,#07H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL7
                LCALL   OFF_DFS7
NMS6S7:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL6_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.1,$+9
                LCALL   C2SL6_SL7
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL6_SL7           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL6
                LCALL   ONBT_SL7
                LJMP    MAIN_SYS
CCS6S8:         CJNE    A,#08H,$-3
                MOV     R7,#00H
                LCALL   RBT_SL8
                LCALL   OFF_DFS8
NMS6S8:         JB      CROSS_EXT.0,$+9
                LCALL   C1SL6_SL8
                LJMP    MAIN_SYS
                JB      CROSC_EXT.1,$+9
                LCALL   C2SL6_SL8
                LJMP    MAIN_SYS
                JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
                LCALL   C3SL6_SL8           ;SEND BUSY TONE
                LJMP    MAIN_SYS
                LCALL   ONBT_SL6
                LCALL   ONBT_SL8
                LJMP    MAIN_SYS
GDT_SL6:        MOV     R7,#0FFH
                LCALL   DT_SL6              ;GIVE DIAL TONE
                MOV     A,#00H
C_DF3E6:        CJNE    A,DTMF3,C_DF4E6
                MOV     R7,#0FFH
                LCALL   DF3_SL6
                LJMP    MAIN_SYS
C_DF4E6:        CJNE    A,DTMF4,C_DF5E6
                MOV     R7,#0FFH
                LCALL   DF4_SL6
                LJMP    MAIN_SYS
C_DF5E6:        CJNE    A,DTMF5,C_DF6E6
                MOV     R7,#0FFH
                LCALL   DF5_SL6
                LJMP    MAIN_SYS
C_DF6E6:        CJNE    A,DTMF6,NO_DF_E6
                MOV     R7,#0FFH
                LCALL   DF6_SL6
                LJMP    MAIN_SYS

NO_DF_E6:       SETB    WAIT_DF.5
                LJMP    MAIN_SYS
;*********************************************************
CHK_SL7:        CJNE    A,#40H,JS8          ; HOFF = HOLD OFF
                SJMP    SL7
JS8:            LJMP    CHK_SL8
SL7:            JNB     S_EXT.6,OFF_HK7
                LJMP    ON_HK7

;***************  EXT7 HOLDING OFF ******************
OFF_HK7:        LCALL   DELAY90
                SETB    S_EXT.6
                MOV     P_NOSL7,#0FFH       ;CLEAR BUFFER OF NUMBERS
                CLR     WAIT_DF.6           ;CLEAR WAITING DTMF DEC.
                ANL     FNC1,#0BFH          ;CLEAR FUNCTION
                ANL     FNC2,#0BFH
                ANL     FNC3,#0BFH
                ANL     OUT_T1,#0BFH
                ANL     OUT_T2,#0BFH
                JB      EXT_DT.6,CLR_DT7
                JB      EXT_BT.6,CLR_BT7
                JB      EXT_RBT.6,CLR_RBT7
                LCALL   OFF_DFS7
```

```
            LJMP    OFF_LS7
CLR_DT7:    MOV     R7,#00H
            LCALL   DT_SL7
            LJMP    OFF_LS7
CLR_BT7:    MOV     R7,#00H
            LCALL   BT_SL7
            LJMP    OFF_LS7
CLR_RBT7:   MOV     R7,#00H
            LCALL   RBT_SL7
            MOV     A,R_EXT1
            CJNE    A,#07H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
            MOV     A,R_EXT2
            CJNE    A,#07H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#07H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#07H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#07H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#07H,$+9
            LCALL   OFF_R6
            LJMP    OFF_LS1
            MOV     A,R_EXT8
            CJNE    A,#07H,$+6
            LCALL   OFF_R8
;********* OFF LINE THAT CONNECT WITH EXT.7 ***********
OFF_LS7:    MOV     A,#09H
            CJNE    A,EXT7,CHKT2E7
            MOV     R7,#00H
            LCALL   TK1_SL7
            MOV     A,TKE1
            ANL     A,#40H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E7:    MOV     A,#0AH
            CJNE    A,EXT7,CHKC1E7
            MOV     R7,#00H
            LCALL   TK2_SL7
            MOV     A,TKE2
            ANL     A,#40H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E7:    JNB     CROSSE1.6,CHKC2E7
            MOV     R7,#00H
            LCALL   CROSS1_7
            LCALL   CHK_CR1
CHKC2E7:    JNB     CROSSE2.6,CHKC3E7
            MOV     R7,#00H
            LCALL   CROSS2_7
            LCALL   CHK_CR2
CHKC3E7:    JNB     CROSSE3.6,OFF_E7END
            MOV     R7,#00H
            LCALL   CROSS3_7
            LCALL   CHK_CR3
OFF_E7END:  LJMP    MAIN_SYS
;************* EXT7.HOLD ON *****************
ON_HK7:     LCALL   DELAY90
            CLR     S_EXT.6
            JB      EXT_RG.6,H_RG7
            LJMP    GDT_SL7
H_RG7:      MOV     R7,#00H
            LCALL   OFF_R7
            MOV     A,R_EXT7
            CJNE    A,#09H,CCS7T2
            MOV     R7,#00H
            LCALL   TK1_RBT
            LCALL   CTK1_SL7
            LJMP    MAIN_SYS
CCS7T2:     CJNE    A,#0AH,CCS7S1
            MOV     R7,#00H
            LCALL   TK2_RBT
            LCALL   CTK2_SL7
            LJMP    MAIN_SYS
            LJMP    CCS7S2
CCS7S1:     CJNE    A,#01H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL1
            LCALL   OFF_DFS1
NMS7S1:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL1_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL1_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL1_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL1
            LJMP    MAIN_SYS
            LJMP    CCS7S3
CCS7S2:     CJNE    A,#02H,$-3
            MOV     R7,#00H
```

```
            LCALL   RBT_SL2
            LCALL   OFF_DFS2
NMS7S2:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL2_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL2_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL2_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL2
            LJMP    MAIN_SYS
            LJMP    CCS7S4
CCS7S3:     CJNE    A,#03H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL3
            LCALL   OFF_DFS3
NMS7S3:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL3_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL3_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL3_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL3
            LJMP    MAIN_SYS
            LJMP    CCS7S5
CCS7S4:     CJNE    A,#04H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL4
            LCALL   OFF_DFS4
NMS7S4:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL4_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL4_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL4_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL4
            LJMP    MAIN_SYS
            LJMP    CCS7S6
CCS7S5:     CJNE    A,#05H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL5
            LCALL   OFF_DFS5
NMS7S5:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL5_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL5_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL5_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL5
            LJMP    MAIN_SYS
            LJMP    CCS7S8
CCS7S6:     CJNE    A,#06H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL6
            LCALL   OFF_DFS6
NMS7S6:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL6_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL6_SL7
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL6_SL7           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL6
            LJMP    MAIN_SYS
CCS7S8:     CJNE    A,#08H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL8
            LCALL   OFF_DFS8
NMS7S8:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL7_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL7_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL7_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL7
            LCALL   ONBT_SL8
            LJMP    MAIN_SYS
GDT_SL7:    MOV     R7,#0FFH
            LCALL   DT_SL7              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E7:    CJNE    A,DTMF3,C_DF4E7
            MOV     R7,#0FFH
            LCALL   DF3_SL7
            LJMP    MAIN_SYS
C_DF4E7:    CJNE    A,DTMF4,C_DF5E7
```

```
            MOV     R7,#0FFH
            LCALL   DF4_SL7
            LJMP    MAIN_SYS
C_DF5E7:    CJNE    A,DTMF5,C_DF6E7
            MOV     R7,#0FFH
            LCALL   DF5_SL7
            LJMP    MAIN_SYS
C_DF6E7:    CJNE    A,DTMF6,NO_DF_E7
            MOV     R7,#0FFH
            LCALL   DF6_SL7
            LJMP    MAIN_SYS

NO_DF_E7:   SETB    WAIT_DF.6
            LJMP    MAIN_SYS
;*************************************************
CHK_SL8:    CJNE    A,#80H,J_MAIN       ; HOFF = HOLD OFF
            SJMP    SL8
J_MAIN:     LJMP    MAIN_SYS
SL8:        JNB     S_EXT.7,OFF_HK8
            LJMP    ON_HK8
;***************  EXT8 HOLDING OFF ******************
OFF_HK8:    LCALL   DELAY90
            SETB    S_EXT.7
            MOV     P_NOSL8,#0FFH          ;CLEAR BUFFER OF NUMBERS
            CLR     WAIT_DF.7              ;CLEAR WAITING DTMF DEC.
            ANL     FNC1,#7FH              ;CLEAR FUNCTION
            ANL     FNC2,#7FH
            ANL     FNC3,#7FH
            ANL     OUT_T1,#07FH
            ANL     OUT_T2,#07FH
            JB      EXT_DT.7,CLR_DT8       ;
            JB      EXT_BT.7,CLR_BT8
            JB      EXT_RBT.7,CLR_RBT8
            LCALL   OFF_DFS8
            LJMP    OFF_LS8
CLR_DT8:    MOV     R7,#00H
            LCALL   DT_SL8
            LJMP    OFF_LS8
CLR_BT8:    MOV     R7,#00H
            LCALL   BT_SL8
            LJMP    OFF_LS8
CLR_RBT8:   MOV     R7,#00H
            LCALL   RBT_SL8
            MOV     A,R_EXT1
            CJNE    A,#08H,$+9
            LCALL   OFF_R1
            LJMP    OFF_LS1
            MOV     A,R_EXT2
            CJNE    A,#08H,$+9
            LCALL   OFF_R2
            LJMP    OFF_LS1
            MOV     A,R_EXT3
            CJNE    A,#08H,$+9
            LCALL   OFF_R3
            LJMP    OFF_LS1
            MOV     A,R_EXT4
            CJNE    A,#08H,$+9
            LCALL   OFF_R4
            LJMP    OFF_LS1
            MOV     A,R_EXT5
            CJNE    A,#08H,$+9
            LCALL   OFF_R5
            LJMP    OFF_LS1
            MOV     A,R_EXT6
            CJNE    A,#08H,$+9
            LCALL   OFF_R6
            LJMP    OFF_LS1
            MOV     A,R_EXT7
            CJNE    A,#08H,$+6
            LCALL   OFF_R7
;********* OFF LINE THAT CONNECT WITH EXT.8 ***********
OFF_LS8:    MOV     A,#09H
            CJNE    A,EXT8,CHKT2E8
            MOV     R7,#00H
            LCALL   TK1_SL8
            MOV     A,TKE1
            ANL     A,#80H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK1_HK
CHKT2E8:    MOV     A,#0AH
            CJNE    A,EXT8,CHKC1E8
            MOV     R7,#00H
            LCALL   TK2_SL8
            MOV     A,TKE2
            ANL     A,#80H
            CJNE    A,#00H,$+8
            MOV     R7,#00H
            LCALL   TK2_HK
CHKC1E8:    JNB     CROSSE1.7,CHKC2E8
            MOV     R7,#00H
            LCALL   CROSS1_8
            LCALL   CHK_CR1
CHKC2E8:    JNB     CROSSE2.7,CHKC3E8
            MOV     R7,#00H
            LCALL   CROSS2_8
            LCALL   CHK_CR2
CHKC3E8:    JNB     CROSSE3.7,OFF_E8END
            MOV     R7,#00H
            LCALL   CROSS3_8
            LCALL   CHK_CR3
OFF_E8END:  LJMP    MAIN_SYS
;************** EXT8.HOLD ON *****************
ON_HK8:     LCALL   DELAY90
            CLR     S_EXT.7
            JB      EXT_RG.7,H_RG8
```

```
          LJMP    GDT_SL8
I_RG8:    MOV     R7,#00H
          LCALL   OFF_R8
          MOV     A,R_EXT8
          CJNE    A,#09H,CCS8T2
          MOV     R7,#00H
          LCALL   TK1_RBT
          LCALL   CTK1_SL8
          LJMP    MAIN_SYS
CCS8T2:   CJNE    A,#0AH,CCS8S1
          MOV     R7,#00H
          LCALL   TK2_RBT
          LCALL   CTK2_SL8
          LJMP    MAIN_SYS
          LJMP    CCS8S2
CCS8S1:   CJNE    A,#01H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL1
          LCALL   OFF_DFS1
NMS8S1:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL1_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.1,$+9
          LCALL   C2SL1_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
          LCALL   C3SL1_SL8           ;SEND BUSY TONE
          LJMP    MAIN_SYS
          LCALL   ONBT_SL8
          LCALL   ONBT_SL1
          LJMP    MAIN_SYS
          LJMP    CCS8S3
CCS8S2:   CJNE    A,#02H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL2
          LCALL   OFF_DFS2
NMS8S2:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL2_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.1,$+9
          LCALL   C2SL2_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
          LCALL   C3SL2_SL8           ;SEND BUSY TONE
          LJMP    MAIN_SYS
          LCALL   ONBT_SL8
          LCALL   ONBT_SL2
          LJMP    MAIN_SYS
          LJMP    CCS8S4
CCS8S3:   CJNE    A,#03H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL3
          LCALL   OFF_DFS3
NMS8S3:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL3_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.1,$+9
          LCALL   C2SL3_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
          LCALL   C3SL3_SL8           ;SEND BUSY TONE
          LJMP    MAIN_SYS
          LCALL   ONBT_SL8
          LCALL   ONBT_SL3
          LJMP    MAIN_SYS
          LJMP    CCS8S5
CCS8S4:   CJNE    A,#04H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL4
          LCALL   OFF_DFS4
NMS8S4:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL4_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.1,$+9
          LCALL   C2SL4_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
          LCALL   C3SL4_SL8           ;SEND BUSY TONE
          LJMP    MAIN_SYS
          LCALL   ONBT_SL8
          LCALL   ONBT_SL4
          LJMP    MAIN_SYS
          LJMP    CCS8S6
CCS8S5:   CJNE    A,#05H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL5
          LCALL   OFF_DFS5
NMS8S5:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL5_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.1,$+9
          LCALL   C2SL5_SL8
          LJMP    MAIN_SYS
          JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
          LCALL   C3SL5_SL8           ;SEND BUSY TONE
          LJMP    MAIN_SYS
          LCALL   ONBT_SL8
          LCALL   ONBT_SL5
          LJMP    MAIN_SYS
          LJMP    CCS8S7
CCS8S6:   CJNE    A,#06H,$-3
          MOV     R7,#00H
          LCALL   RBT_SL6
          LCALL   OFF_DFS6
NMS8S6:   JB      CROSS_EXT.0,$+9
          LCALL   C1SL6_SL8
```

```
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL6_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL6_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL8
            LCALL   ONBT_SL6
            LJMP    MAIN_SYS
CCS8S7:     CJNE    A,#07H,$-3
            MOV     R7,#00H
            LCALL   RBT_SL7
            LCALL   OFF_DFS7
NMS8S7:     JB      CROSS_EXT.0,$+9
            LCALL   C1SL7_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.1,$+9
            LCALL   C2SL7_SL8
            LJMP    MAIN_SYS
            JB      CROSS_EXT.2,$+9     ;IF NOT ENOUGH CROSSP.
            LCALL   C3SL7_SL8           ;SEND BUSY TONE
            LJMP    MAIN_SYS
            LCALL   ONBT_SL8
            LCALL   ONBT_SL7
            LJMP    MAIN_SYS
GDT_SL8:    MOV     R7,#0FFH
            LCALL   DT_SL8              ;GIVE DIAL TONE
            MOV     A,#00H
C_DF3E8:    CJNE    A,DTMF3,C_DF4E8
            MOV     R7,#0FFH
            LCALL   DF3_SL8
            LJMP    MAIN_SYS
C_DF4E8:    CJNE    A,DTMF4,C_DF5E8
            MOV     R7,#0FFH
            LCALL   DF4_SL8
            LJMP    MAIN_SYS
C_DF5E8:    CJNE    A,DTMF5,C_DF6E8
            MOV     R7,#0FFH
            LCALL   DF5_SL8
            LJMP    MAIN_SYS
C_DF6E8:    CJNE    A,DTMF6,NO_DF_E8
            MOV     R7,#0FFH
            LCALL   DF6_SL8
            LJMP    MAIN_SYS
NO_DF_E8:   SETB    WAIT_DF.7
            LJMP    MAIN_SYS
;*************************************************
CHK_CR1:    MOV     A,CROSSE1
            CJNE    A,#00H,$+5
            CLR     CROSS_EXT.0
            RET
CHK_CR2:    MOV     A,CROSSE2
            CJNE    A,#00H,$+5
            CLR     CROSS_EXT.1
            RET
CHK_CR3:    MOV     A,CROSSE3
            CJNE    A,#00H,$+5
            CLR     CROSS_EXT.2
            RET
;*********** SUB PROGRAM DETECT TONES *************
SV_BT1:     MOV     R7,#00H
            LCALL   TK1_HK
            MOV     TKE1,#00H
            MOV     OUT_T1,#00H
            LJMP    MAIN_SYS
SV_BT2:     MOV     R7,#00H
            LCALL   TK2_HK
            MOV     TKE2,#00H
            MOV     OUT_T2,#00H
            LJMP    MAIN_SYS
;************* SUB ISR_0 FOR TRUNK RINGING ************
T_HK:       LCALL   DELAY500            ;** TRUNK1 INT0 = 1 msec **
            JB      T0,TK2_SER          ;** TRUNK2 INT0 = 0.125 msec **
TK1_SER:    ANL     TK_DF,#0FEH         ; GIVE DTMF DEC. TO TK1
            MOV     R7,#0FFH
            LCALL   TK1_HK
            LCALL   DELAY1
            MOV     R6,#80H
            LCALL   AUTO_ANS1
            LCALL   DELAY4
            LCALL   AUT1
            LJMP    INT_0

TK2_SER:    ANL     TK_DF,#0FDH         ; GIVE DTMF DEC. TO TK2
            MOV     R7,#0FFH
            LCALL   TK2_HK
            LCALL   DELAY1
            MOV     R6,#80H
            LCALL   AUTO_ANS2
            LCALL   DELAY4
            LCALL   AUT2
            LJMP    INT_0
;*************** SUB HOOK OF TRUNK1  *********************
TK1_HK:     MOV     A,TK_TOE
            CJNE    R7,#00H,TK1_ON
            ANL     A,#10111111B
            SJMP    LD_TK1
TK1_ON:     ORL     A,#01000000B
LD_TK1:     MOV     DPTR,#PORTA_1
            MOVX    @DPTR,A
            MOV     TK_TOE,A
            RET
;*************** SUB HOOK OF TRUNK2  *********************
TK2_HK:     MOV     A,TK_TOE
            CJNE    R7,#00H,TK2_ON
```

```
            ANL     A,#01111111B
            SJMP    LD_TK2
TK2_ON:     ORL     A,#10000000B
LD_TK2:     MOV     DPTR,#PORTA_1
            MOVX    @DPTR,A
            MOV     TK_TOE,A
            RET
;************ SUB ISR_1 FOR PUSH NUMBER ************
PUSH_N:     LCALL   LD_STD
CHK_PT1:    CJNE    A,#01H,J_PT2
            MOV     A,OUT_T1
            CJNE    A,#00H,$+8
            MOV     A,TK_DF
            JNB     ACC.0,CON_PT1
            LCALL   DELAY05
            LJMP    INT_1               ;NOT ALLOW DTMF OF T1
J_PT2:      LJMP    CHK_PT2
CON_PT1:    LCALL   ON_TOE1
            LCALL   TK_BCD
TK1_NO1:    CJNE    A,#01H,TK1_NO2
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#01H,$+6
            LJMP    STR_NUM1
            ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
            LCALL   LD_SEXT
            JNB     ACC.0,T1E1_BUSY
            MOV     R_EXT1,#09H         ;TK1 RINGING TO SL1
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   TK1_RBT
            LCALL   C_LINK
            LJMP    INT_1
T1E1_BUSY:  LJMP    TK1_FHK

TK1_NO2:    CJNE    A,#02H,TK1_NO3
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#01H,$+6
            LJMP    STR_NUM1
            ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
            LCALL   LD_SEXT
            JNB     ACC.1,T1E2_BUSY
            MOV     R_EXT2,#09H         ;TK1 RINGING TO SL2
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   TK1_RBT
            LJMP    INT_1
T1E2_BUSY:  LJMP    TK1_FHK

TK1_NO3:    CJNE    A,#03H,TK1_NO4
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#01H,$+6
            LJMP    STR_NUM1
            ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
            LCALL   LD_SEXT
            JNB     ACC.2,T1E3_BUSY
            MOV     R_EXT3,#09H         ;TK1 RINGING TO SL3
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   TK1_RBT
            LJMP    INT_1
T1E3_BUSY:  LJMP    TK1_FHK
TK1_NO4:    CJNE    A,#04H,TK1_NO5
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#01H,$+6
            LJMP    STR_NUM1
            ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
            LCALL   LD_SEXT
            JNB     ACC.3,T1E4_BUSY
            MOV     R_EXT4,#09H         ;TK1 RINGING TO SL4
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   TK1_RBT
            LJMP    INT_1
T1E4_BUSY:  LJMP    TK1_FHK
TK1_NO5:    CJNE    A,#05H,TK1_NO6
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#01H,$+6
            LJMP    STR_NUM1
            ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
            LCALL   LD_SEXT
            JNB     ACC.4,T1E5_BUSY
            MOV     R_EXT5,#09H         ;TK1 RINGING TO SL5
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   TK1_RBT
            LJMP    INT_1
T1E5_BUSY:  LJMP    TK1_FHK
TK1_NO6:    CJNE    A,#06H,TK1_NO7
            MOV     R5,A
            LCALL   OFF_TOE1
            MOV     A,FNC4
            ANL     A,#01H
```

```
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
                LCALL   LD_SEXT
                JNB     ACC.5,T1E6_BUSY
                MOV     R_EXT6,#09H         ;TK1 RINGING TO SL6
                LCALL   ON_R6
                MOV     R7,#0FFH
                LCALL   TK1_RBT
                LJMP    INT_1
T1E6_BUSY:      LJMP    TK1_FHK
TK1_NO7:        CJNE    A,#07H,TK1_NO8
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
                LCALL   LD_SEXT
                JNB     ACC.6,T1E7_BUSY
                MOV     R_EXT7,#09H         ;TK1 RINGING TO SL7
                LCALL   ON_R7
                MOV     R7,#0FFH
                LCALL   TK1_RBT
                LJMP    INT_1
T1E7_BUSY:      LJMP    TK1_FHK
TK1_NO8:        CJNE    A,#08H,TK1_NO9
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                ORL     TK_DF,#01H          ;NOT GIVE DTMF FOR TK1
                LCALL   LD_SEXT
                JNB     ACC.7,T1E8_BUSY
                MOV     R_EXT8,#09H         ;TK1 RINGING TO SL8
                LCALL   ON_R8
                MOV     R7,#0FFH
                LCALL   TK1_RBT
                LJMP    INT_1
T1E8_BUSY:      LJMP    TK1_FHK
TK1_NO9:        CJNE    A,#09H,TK1_NO0
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                SJMP    TK1_ERR
TK1_NO0:        CJNE    A,#0AH,TK1_NOB
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                SJMP    TK1_ERR
TK1_NOB:        CJNE    A,#0BH,TK1_NOC      ;PUSH '*'
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                MOV     A,RA1
                CJNE    A,#0FFH,$+6
                LJMP    STR_NUM1
                SJMP    TK1_ERR
TK1_NOC:        CJNE    A,#0CH,TK1_ERR      ;PUSH '#'
                MOV     R5,A
                LCALL   OFF_TOE1
                MOV     A,FNC4
                ANL     A,#01H
                CJNE    A,#01H,$+6
                LJMP    STR_NUM1
                MOV     A,RA1
                CJNE    A,#0FFH,$+6
                LJMP    STR_NUM1
                SJMP    TK1_ERR
TK1_FHK:        MOV     R7,#0FFH
                LCALL   TK1_BT
                LCALL   DELAY2
                MOV     R7,#00H
                LCALL   TK1_HK
                LJMP    INT_1
TK1_ERR:        MOV     R7,#0FFH
                LCALL   TK1_BT
                LCALL   DELAY2
                MOV     R7,#00H
                LCALL   TK1_BT
                LJMP    INT_1
;*****************************************************************
CHK_PT2:        CJNE    A,#02H,J_PDF3
                MOV     A,OUT_T2
                CJNE    A,#00H,$+8
                MOV     A,TK_DF
                JNB     ACC.1,CON_PT2
                LCALL   DELAY05
                LJMP    INT_1
J_PDF3:         LJMP    CHK_PDF3
CON_PT2:        LCALL   ON_TOE2
                LCALL   TK_BCD
TK2_NO1:        CJNE    A,#01H,TK2_NO2
                MOV     R5,A
```

```
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.0,T2E1_BUSY
                MOV     R_EXT1,#0AH         ;TK2 RINGING TO SL1
                LCALL   ON_R1
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LCALL   C_LINK
                LJMP    INT_1
T2E1_BUSY:      LJMP    TK2_FHK
TK2_NO2:        CJNE    A,#02H,TK2_NO3
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.1,T2E2_BUSY
                MOV     R_EXT2,#0AH         ;TK2 RINGING TO SL2
                LCALL   ON_R2
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E2_BUSY:      LJMP    TK2_FHK
TK2_NO3:        CJNE    A,#03H,TK2_NO4
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.2,T2E3_BUSY
                MOV     R_EXT3,#0AH         ;TK2 RINGING TO SL3
                LCALL   ON_R3
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E3_BUSY:      LJMP    TK2_FHK
TK2_NO4:        CJNE    A,#04H,TK2_NO5
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.3,T2E4_BUSY
                MOV     R_EXT4,#0AH         ;TK2 RINGING TO SL4
                LCALL   ON_R4
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E4_BUSY:      LJMP    TK2_FHK
TK2_NO5:        CJNE    A,#05H,TK2_NO6
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.4,T2E5_BUSY
                MOV     R_EXT5,#0AH         ;TK2 RINGING TO SL5
                LCALL   ON_R5
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E5_BUSY:      LJMP    TK2_FHK
TK2_NO6:        CJNE    A,#06H,TK2_NO7
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.5,T2E6_BUSY
                MOV     R_EXT6,#0AH         ;TK2 RINGING TO SL6
                LCALL   ON_R6
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E6_BUSY:      LJMP    TK2_FHK
TK2_NO7:        CJNE    A,#07H,TK2_NO8
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.6,T2E7_BUSY
                MOV     R_EXT7,#0AH         ;TK2 RINGING TO SL7
                LCALL   ON_R7
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E7_BUSY:      LJMP    TK2_FHK
TK2_NO8:        CJNE    A,#08H,TK2_NO9
                MOV     R5,A
                LCALL   OFF_TOE2
                ORL     TK_DF,#02H          ;NOT GIVE DTMF FOR TK2
                LCALL   LD_SEXT
                JNB     ACC.7;T2E8_BUSY
                MOV     R_EXT8,#0AH         ;TK2 RINGING TO SL8
                LCALL   ON_R8
                MOV     R7,#0FFH
                LCALL   TK2_RBT
                LJMP    INT_1
T2E8_BUSY:      LJMP    TK2_FHK
TK2_NO9:        CJNE    A,#09H,TK2_NO0
                MOV     R5,A
                LCALL   OFF_TOE2
                SJMP    TK2_ERR
TK2_NO0:        CJNE    A,#0AH,TK2_NOB
                MOV     R5,A
                LCALL   OFF_TOE2
                SJMP    TK2_ERR
TK2_NOB:        CJNE    A,#0BH,TK2_NOC      ;PUSH '*'
                MOV     R5,A
                LCALL   OFF_TOE2
```

```
            SJMP    TK2_ERR
TK2_NOC:    CJNE    A,#0CH,TK2_ERR    ;PUSH '#'
            MOV     R5,A
            LCALL   OFF_TOE2
            SJMP    TK2_ERR
TK2_FHK:    MOV     R7,#0FFH
            LCALL   TK2_BT
            LCALL   DELAY2
            MOV     R7,#00H
            LCALL   TK2_HK
            LJMP    INT_1
TK2_ERR:    MOV     R7,#0FFH
            LCALL   TK2_BT
            LCALL   DELAY2
            MOV     R7,#00H
            LCALL   TK2_BT
            LJMP    INT_1
;***************** SUB CHECK DTMF NO.3 ************************
CHK_PDF3:   CJNE    A,#04H,J_PDF4
CSL1_DF3:   JB      DTMF3.0,SL1_DF3
            LJMP    CSL2_DF3
J_PDF4:     LJMP    CHK_PDF4
S13_RMA:    LJMP    INT_1
SL1_DF3:    JB      EXT_BT.0,S13_RMA
            JB      EXT_RBT.0,S13_RMA
            MOV     A,P_NOSL1
            CJNE    A,#0BBH,D3SL1_F1
            ORL     FNC2,#01H
            ANL     FNC1,#0FEH
            AJMP    LDS1_DF3
D3SL1_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS1_DF3
            ORL     FNC1,#01H
LDS1_DF3:   MOV     R7,#00H
            LCALL   DT_SL1
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL1_NO1
CSL2_DF3:   JB      DTMF3.1,SL2_DF3
            LJMP    CSL3_DF3
S23_RMA:    LJMP    INT_1
SL2_DF3:    JB      EXT_BT.1,S23_RMA
            JB      EXT_RBT.1,S23_RMA
            MOV     A,P_NOSL2
            CJNE    A,#0BBH,D3SL2_F1
            ORL     FNC2,#02H
            ANL     FNC1,#0FDH
            AJMP    LDS2_DF3
D3SL2_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS2_DF3
            ORL     FNC1,#02H
LDS2_DF3:   MOV     R7,#00H
            LCALL   DT_SL2
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL2_NO1
CSL3_DF3:   JB      DTMF3.2,SL3_DF3
            LJMP    CSL4_DF3
S33_RMA:    LJMP    INT_1
SL3_DF3:    JB      EXT_BT.2,S33_RMA
            JB      EXT_RBT.2,S33_RMA
            MOV     A,P_NOSL3
            CJNE    A,#0BBH,D3SL3_F1
            ORL     FNC2,#04H
            ANL     FNC1,#0FBH
            AJMP    LDS3_DF3
D3SL3_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS3_DF3
            ORL     FNC1,#04H
LDS3_DF3:   MOV     R7,#00H
            LCALL   DT_SL3
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL3_NO1
CSL4_DF3:   JB      DTMF3.3,SL4_DF3
            LJMP    CSL5_DF3
S43_RMA:    LJMP    INT_1
SL4_DF3:    JB      EXT_BT.3,S43_RMA
            JB      EXT_RBT.3,S43_RMA
            MOV     A,P_NOSL4
            CJNE    A,#0BBH,D3SL4_F1
            ORL     FNC2,#08H
            ANL     FNC1,#0F7H
            AJMP    LDS4_DF3
D3SL4_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS4_DF3
            ORL     FNC1,#08H
LDS4_DF3:   MOV     R7,#00H
            LCALL   DT_SL4
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL4_NO1
CSL5_DF3:   JB      DTMF3.4,SL5_DF3
            LJMP    CSL6_DF3
S53_RMA:    LJMP    INT_1
SL5_DF3:    JB      EXT_BT.4,S53_RMA
            JB      EXT_RBT.4,S53_RMA
            MOV     A,P_NOSL5
```

```
            CJNE    A,#0BBH,D3SL5_F1
            ORL     FNC2,#10H
            ANL     FNC1,#0EFH
            LJMP    LDS5_DF3
D3SL5_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS5_DF3
            ORL     FNC1,#10H
LDS5_DF3:   MOV     R7,#00H
            LCALL   DT_SL5
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL5_NO1
CSL6_DF3:   JB      DTMF3.5,SL6_DF3
            LJMP    CSL7_DF3
S63_RMA:    LJMP    INT_1
SL6_DF3:    JB      EXT_BT.5,S63_RMA
            JB      EXT_RBT.5,S63_RMA
            MOV     A,P_NOSL6
            CJNE    A,#0BBH,D3SL6_F1
            ORL     FNC2,#20H
            ANL     FNC1,#0DFH
            AJMP    LDS6_DF3
D3SL6_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS6_DF3
            ORL     FNC1,#20H
LDS6_DF3:   MOV     R7,#00H
            LCALL   DT_SL6
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL6_NO1
CSL7_DF3:   JB      DTMF3.6,SL7_DF3
            LJMP    CSL8_DF3
S73_RMA:    LJMP    INT_1
SL7_DF3:    JB      EXT_BT.6,S73_RMA
            JB      EXT_RBT.6,S73_RMA
            MOV     A,P_NOSL7
            CJNE    A,#0BBH,D3SL7_F1
            ORL     FNC2,#40H
            ANL     FNC1,#0BFH
            LJMP    LDS7_DF3
D3SL7_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS7_DF3
            ORL     FNC1,#40H
LDS7_DF3:   MOV     R7,#00H
            LCALL   DT_SL7
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL7_NO1
CSL8_DF3:   JB      DTMF3.7,SL8_DF3
S83_RMA:    LJMP    INT_1
SL8_DF3:    JB      EXT_BT.7,S83_RMA
            JB      EXT_RBT.7,S83_RMA
            MOV     A,P_NOSL8
            CJNE    A,#0BBH,D3SL8_F1
            ORL     FNC2,#80H
            ANL     FNC1,#7FH
            AJMP    LDS8_DF3
D3SL8_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS8_DF3
            ORL     FNC1,#80H
LDS8_DF3:   MOV     R7,#00H
            LCALL   DT_SL8
            LCALL   ON_TOE3
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE3
            LJMP    SL8_NO1
;***************** SUB CHECK DTMF NO.4 *************************
CHK_PDF4:   CJNE    A,#08H,J_PDF5
CSL1_DF4:   JB      DTMF4.0,SL1_DF4
            LJMP    CSL2_DF4
J_PDF5:     LJMP    CHK_PDF5
S14_RMA:    LJMP    INT_1
SL1_DF4:    JB      EXT_BT.0,S14_RMA
            JB      EXT_RBT.0,S14_RMA
            MOV     A,P_NOSL1
            CJNE    A,#0BBH,D4SL1_F1
            ORL     FNC2,#01H
            ANL     FNC1,#0FEH
            AJMP    LDS1_DF4
D4SL1_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS1_DF4
            ORL     FNC1,#01H
LDS1_DF4:   MOV     R7,#00H
            LCALL   DT_SL1
            LCALL   ON_TOE4
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE4
            LJMP    SL1_NO1
CSL2_DF4:   JB      DTMF4.1,SL2_DF4
            LJMP    CSL3_DF4
S24_RMA:    LJMP    INT_1
SL2_DF4:    JB      EXT_BT.1,S24_RMA
            JB      EXT_RBT.1,S24_RMA
            MOV     A,P_NOSL2
            CJNE    A,#0BBH,D4SL2_F1
            ORL     FNC2,#02H
            ANL     FNC1,#0FDH
            AJMP    LDS2_DF4
```

```
D4SL2_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS2_DF4
           ORL     FNC1,#02H
LDS2_DF4:  MOV     R7,#00H
           LCALL   DT_SL2
           LCALL   ON_TOE4
           LCALL   SL_BCD
           MOV     DF_BUF,A
           LCALL   OFF_TOE4
           LJMP    SL2_NO1
CSL3_DF4:  JB      DTMF4.2,SL3_DF4
           LJMP    CSL4_DF4
S34_RMA:   LJMP    INT_1
SL3_DF4:   JB      EXT_BT.2,S34_RMA
           JB      EXT_RBT.2,S34_RMA
           MOV     A,P_NOSL3
           CJNE    A,#0BBH,D4SL3_F1
           ORL     FNC2,#04H
           ANL     FNC1,#0FBH
           AJMP    LDS3_DF4
D4SL3_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS3_DF4
           ORL     FNC1,#04H
LDS3_DF4:  MOV     R7,#00H
           LCALL   DT_SL3
           LCALL   ON_TOE4
           LCALL   SL_BCD
           MOV     DF_BUF,A
           LCALL   OFF_TOE4
           LJMP    SL3_NO1
CSL4_DF4:  JB      DTMF4.3,SL4_DF4
           LJMP    CSL5_DF4
S44_RMA:   LJMP    INT_1
SL4_DF4:   JB      EXT_BT.3,S44_RMA
           JB      EXT_RBT.3,S44_RMA
           MOV     A,P_NOSL4
           CJNE    A,#0BBH,D4SL4_F1
           ORL     FNC2,#08H
           ANL     FNC1,#0F7H
           AJMP    LDS4_DF4
D4SL4_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS4_DF4
           ORL     FNC1,#08H
LDS4_DF4:  MOV     R7,#00H
           LCALL   DT_SL4
           LCALL   ON_TOE4
           LCALL   SL_BCD
           MOV     DF_BUF,A
           LCALL   OFF_TOE4
           LJMP    SL4_NO1
CSL5_DF4:  JB      DTMF4.4,SL5_DF4
           LJMP    CSL6_DF4
S54_RMA:   LJMP    INT_1
SL5_DF4:   JB      EXT_BT.4,S54_RMA
           JB      EXT_RBT.4,S54_RMA
           MOV     A,P_NOSL5
           CJNE    A,#0BBH,D4SL5_F1
           ORL     FNC2,#10H
           ANL     FNC1,#0EFH
           AJMP    LDS5_DF4
D4SL5_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS5_DF4
           ORL     FNC1,#10H
LDS5_DF4:  MOV     R7,#00H
           LCALL   DT_SL5
           LCALL   ON_TOE4
           LCALL   SL_BCD
           MOV     DF_BUF,A
           LCALL   OFF_TOE4
           LJMP    SL5_NO1
CSL6_DF4:  JB      DTMF4.5,SL6_DF4
           LJMP    CSL7_DF4
S64_RMA:   LJMP    INT_1
SL6_DF4:   JB      EXT_BT.5,S64_RMA
           JB      EXT_RBT.5,S64_RMA
           MOV     A,P_NOSL6
           CJNE    A,#0BBH,D4SL6_F1
           ORL     FNC2,#20H
           ANL     FNC1,#0DFH
           AJMP    LDS6_DF4
D4SL6_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS6_DF4
           ORL     FNC1,#20H
LDS6_DF4:  MOV     R7,#00H
           LCALL   DT_SL6
           LCALL   ON_TOE4
           LCALL   SL_BCD
           MOV     DF_BUF,A
           LCALL   OFF_TOE4
           LJMP    SL6_NO1
CSL7_DF4:  JB      DTMF4.6,SL7_DF4
           LJMP    CSL8_DF4
S74_RMA:   LJMP    INT_1
SL7_DF4:   JB      EXT_BT.6,S74_RMA
           JB      EXT_RBT.6,S74_RMA
           MOV     A,P_NOSL7
           CJNE    A,#0BBH,D4SL7_F1
           ORL     FNC2,#40H
           ANL     FNC1,#0BFH
           AJMP    LDS7_DF4
D4SL7_F1:  ANL     A,#0F0H
           CJNE    A,#0B0H,LDS7_DF4
           ORL     FNC1,#40H
LDS7_DF4:  MOV     R7,#00H
           LCALL   DT_SL7
           LCALL   ON_TOE4
```

```
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE4
            LJMP    SL7_NO1
CSL8_DF4:   JB      DTMF4.7,SL8_DF4
S84_RMA:    LJMP    INT_1
SL8_DF4:    JB      EXT_BT.7,S84_RMA
            JB      EXT_RBT.7,S84_RMA
            MOV     A,P_NOSL8
            CJNE    A,#0BBH,D4SL8_F1
            ORL     FNC2,#80H
            ANL     FNC1,#7FH
            AJMP    LDS8_DF4
D4SL8_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS8_DF4
            ORL     FNC1,#80H
LDS8_DF4:   MOV     R7,#00H
            LCALL   DT_SL8
            LCALL   ON_TOE4
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE4
            LJMP    SL8_NO1
;***************** SUB CHECK DTMF NO.5 ************************
CHK_PDF5:   CJNE    A,#10H,J_PDF6
CSL1_DF5:   JB      DTMF5.0,SL1_DF5
            LJMP    CSL2_DF5
J_PDF6:     LJMP    CHK_PDF6
S15_RMA:    LJMP    INT_1
SL1_DF5:    JB      EXT_BT.0,S15_RMA
            JB      EXT_RBT.0,S15_RMA
            MOV     A,P_NOSL1
            CJNE    A,#0BBH,D5SL1_F1
            ORL     FNC2,#01H
            ANL     FNC1,#0FEH
            AJMP    LDS1_DF5
D5SL1_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS1_DF5
            ORL     FNC1,#01H
LDS1_DF5:   MOV     R7,#00H
            LCALL   DT_SL1
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL1_NO1
CSL2_DF5:   JB      DTMF5.1,SL2_DF5
            LJMP    CSL3_DF5
S25_RMA:    LJMP    INT_1
SL2_DF5:    JB      EXT_BT.1,S25_RMA
            JB      EXT_RBT.1,S25_RMA
            MOV     A,P_NOSL2
            CJNE    A,#0BBH,D5SL2_F1
            ORL     FNC2,#02H
            ANL     FNC1,#0FDH
            AJMP    LDS2_DF5
D5SL2_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS2_DF5
            ORL     FNC1,#02H
LDS2_DF5:   MOV     R7,#00H
            LCALL   DT_SL2
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL2_NO1
CSL3_DF5:   JB      DTMF5.2,SL3_DF5
            LJMP    CSL4_DF5
S35_RMA:    LJMP    INT_1
SL3_DF5:    JB      EXT_BT.2,S35_RMA
            JB      EXT_RBT.2,S35_RMA
            MOV     A,P_NOSL3
            CJNE    A,#0BBH,D5SL3_F1
            ORL     FNC2,#04H
            ANL     FNC1,#0FBH
            AJMP    LDS3_DF5
D5SL3_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS3_DF5
            ORL     FNC1,#04H
LDS3_DF5:   MOV     R7,#00H
            LCALL   DT_SL3
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL3_NO1
CSL4_DF5:   JB      DTMF5.3,SL4_DF5
            LJMP    CSL5_DF5
S45_RMA:    LJMP    INT_1
SL4_DF5:    JB      EXT_BT.3,S45_RMA
            JB      EXT_RBT.3,S45_RMA
            MOV     A,P_NOSL4
            CJNE    A,#0BBH,D5SL4_F1
            ORL     FNC2,#08H
            ANL     FNC1,#0F7H
            AJMP    LDS4_DF5
D5SL4_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS4_DF5
            ORL     FNC1,#08H
LDS4_DF5:   MOV     R7,#00H
            LCALL   DT_SL4
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL4_NO1
```

```
CSL5_DF5:   JB      DTMF5.4,SL5_DF5
            LJMP    CSL6_DF5
S55_RMA:    LJMP    INT_1
SL5_DF5:    JB      EXT_BT.4,S55_RMA
            JB      EXT_RBT.4,S55_RMA
            MOV     A,P_NOSL5
            CJNE    A,#0BBH,D5SL5_F1
            ORL     FNC2,#10H
            ANL     FNC1,#0EFH
            AJMP    LDS5_DF5
D5SL5_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS5_DF5
            ORL     FNC1,#10H
LDS5_DF5:   MOV     R7,#00H
            LCALL   DT_SL5
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL5_NO1
CSL6_DF5:   JB      DTMF5.5,SL6_DF5
            LJMP    CSL7_DF5
S65_RMA:    LJMP    INT_1
SL6_DF5:    JB      EXT_BT.5,S65_RMA
            JB      EXT_RBT.5,S65_RMA
            MOV     A,P_NOSL6
            CJNE    A,#0BBH,D5SL6_F1
            ORL     FNC2,#20H
            ANL     FNC1,#0DFH
            AJMP    LDS6_DF5
D5SL6_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS6_DF5
            ORL     FNC1,#20H
LDS6_DF5:   MOV     R7,#00H
            LCALL   DT_SL6
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL6_NO1
CSL7_DF5:   JB      DTMF5.6,SL7_DF5
            LJMP    CSL8_DF5
S75_RMA:    LJMP    INT_1
SL7_DF5:    JB      EXT_BT.6,S75_RMA
            JB      EXT_RBT.6,S75_RMA
            MOV     A,P_NOSL7
            CJNE    A,#0BBH,D5SL7_F1
            ORL     FNC2,#40H
            ANL     FNC1,#0BFH
            AJMP    LDS7_DF5
D5SL7_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS7_DF5
            ORL     FNC1,#40H

LDS7_DF5:   MOV     R7,#00H
            LCALL   DT_SL7
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL7_NO1
CSL8_DF5:   JB      DTMF5.7,SL8_DF5
S85_RMA:    LJMP    INT_1
SL8_DF5:    JB      EXT_BT.7,S85_RMA
            JB      EXT_RBT.7,S85_RMA
            MOV     A,P_NOSL8
            CJNE    A,#0BBH,D5SL8_F1
            ORL     FNC2,#80H
            ANL     FNC1,#7FH
            AJMP    LDS8_DF5
D5SL8_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS8_DF5
            ORL     FNC1,#80H
LDS8_DF5:   MOV     R7,#00H
            LCALL   DT_SL8
            LCALL   ON_TOE5
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE5
            LJMP    SL8_NO1
;***************** SUB CHECK DTMF NO.6 *************************
CHK_PDF6:   CJNE    A,#20H,STD_ERR
CSL1_DF6:   JB      DTMF6.0,SL1_DF6
            LJMP    CSL2_DF6
STD_ERR:    LCALL   DELAY05
            LJMP    INT_1
S16_RMA:    LJMP    INT_1
SL1_DF6:    JB      EXT_BT.0,S16_RMA
            JB      EXT_RBT.0,S16_RMA
            MOV     A,P_NOSL1
            CJNE    A,#0BBH,D6SL1_F1
            ORL     FNC2,#01H
            ANL     FNC1,#0FEH
            SJMP    LDS1_DF6
D6SL1_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS1_DF6
            ORL     FNC1,#01H
LDS1_DF6:   MOV     R7,#00H
            LCALL   DT_SL1
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL1_NO1
CSL2_DF6:   JB      DTMF6.1,SL2_DF6
            LJMP    CSL3_DF6
S26_RMA;    LJMP    INT_1
```

```
SL2_DF6:    JB      EXT_BT.1,S26_RMA
            JB      EXT_RBT.1,S26_RMA
            MOV     A,P_NOSL2
            CJNE    A,#0BBH,D6SL2_F1
            ORL     FNC2,#02H
            ANL     FNC1,#0FDH
            AJMP    LDS2_DF6
D6SL2_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS2_DF6
            ORL     FNC1,#02H
LDS2_DF6:   MOV     R7,#00H
            LCALL   DT_SL2
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL2_NO1
CSL3_DF6:   JB      DTMF6.2,SL3_DF6
            LJMP    CSL4_DF6
S36_RMA:    LJMP    INT_1
SL3_DF6:    JB      EXT_BT.2,S36_RMA
            JB      EXT_RBT.2,S36_RMA
            MOV     A,P_NOSL3
            CJNE    A,#0BBH,D6SL3_F1
            ORL     FNC2,#04H
            ANL     FNC1,#0FBH
            AJMP    LDS3_DF6
D6SL3_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS3_DF6
            ORL     FNC1,#04H
LDS3_DF6:   MOV     R7,#00H
            LCALL   DT_SL3
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL3_NO1
CSL4_DF6:   JB      DTMF6.3,SL4_DF6
            LJMP    CSL5_DF6
S46_RMA:    LJMP    INT_1
SL4_DF6:    JB      EXT_BT.3,S46_RMA
            JB      EXT_RBT.3,S46_RMA
            MOV     A,P_NOSL4
            CJNE    A,#0BBH,D6SL4_F1
            ORL     FNC2,#08H
            ANL     FNC1,#0F7H
            AJMP    LDS4_DF6
D6SL4_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS4_DF6
            ORL     FNC1,#08H
LDS4_DF6:   MOV     R7,#00H
            LCALL   DT_SL4
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL4_NO1

CSL5_DF6:   JB      DTMF6.4,SL5_DF6
            LJMP    CSL6_DF6
S56_RMA:    LJMP    INT_1
SL5_DF6:    JB      EXT_BT.4,S56_RMA
            JB      EXT_RBT.4,S56_RMA
            MOV     A,P_NOSL5
            CJNE    A,#0BBH,D6SL5_F1
            ORL     FNC2,#10H
            ANL     FNC1,#0EFH
            AJMP    LDS5_DF6
D6SL5_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS5_DF6
            ORL     FNC1,#10H
LDS5_DF6:   MOV     R7,#00H
            LCALL   DT_SL5
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL5_NO1
CSL6_DF6:   JB      DTMF6.5,SL6_DF6
            LJMP    CSL7_DF6
S66_RMA:    LJMP    INT_1
SL6_DF6:    JB      EXT_BT.5,S66_RMA
            JB      EXT_RBT.5,S66_RMA
            MOV     A,P_NOSL6
            CJNE    A,#0BBH,D6SL6_F1
            ORL     FNC2,#20H
            ANL     FNC1,#0DFH
            SJMP    LDS6_DF6
D6SL6_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS6_DF6
            ORL     FNC1,#20H
LDS6_DF6:   MOV     R7,#00H
            LCALL   DT_SL6
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL6_NO1
CSL7_DF6:   JB      DTMF6.6,SL7_DF6
            LJMP    CSL8_DF6
S76_RMA:    LJMP    INT_1
SL7_DF6:    JB      EXT_BT.6,S76_RMA
            JB      EXT_RBT.6,S76_RMA
            MOV     A,P_NOSL7
            CJNE    A,#0BBH,D6SL7_F1
            ORL     FNC2,#40H
```

```
            ANL     FNC1,#0BFH
            AJMP    LDS7_DF6
D6SL7_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS7_DF6
            ORL     FNC1,#40H
LDS7_DF6:   MOV     R7,#00H
            LCALL   DT_SL7
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL7_NO1
CSL8_DF6:   JB      DTMF6.7,SL8_DF6
S86_RMA:    LJMP    INT_1
SL8_DF6:    JB      EXT_BT.7,S86_RMA
            JB      EXT_RBT.7,S86_RMA
            MOV     A,P_NOSL8
            CJNE    A,#0BBH,D6SL8_F1
            ORL     FNC2,#80H
            ANL     FNC1,#7FH
            AJMP    LDS8_DF6
D6SL8_F1:   ANL     A,#0F0H
            CJNE    A,#0B0H,LDS8_DF6
            ORL     FNC1,#80H
LDS8_DF6:   MOV     R7,#00H
            LCALL   DT_SL8
            LCALL   ON_TOE6
            LCALL   SL_BCD
            MOV     DF_BUF,A
            LCALL   OFF_TOE6
            LJMP    SL8_NO1
;************* SUB SERVICE PUSH BUTTON OF SLICS ************
SL1_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL1_NO2
            LJMP    SL1_BRET
SL1_NO2:    CJNE    A,#02H,SL1_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E1E2_BUSY
            MOV     R_EXT2,#01H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E2_BUSY:  LJMP    SL1_BRET
SL1_NO3:    CJNE    A,#03H,SL1_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E1E3_BUSY
            MOV     R_EXT3,#01H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E3_BUSY:  LJMP    SL1_BRET
SL1_NO4:    CJNE    A,#04H,SL1_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E1E4_BUSY
            MOV     R_EXT4,#01H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E4_BUSY:  LJMP    SL1_BRET
SL1_NO5:    CJNE    A,#05H,SL1_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E1E5_BUSY
            MOV     R_EXT5,#01H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E5_BUSY:  LJMP    SL1_BRET
SL1_NO6:    CJNE    A,#06H,SL1_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E1E6_BUSY
            MOV     R_EXT6,#01H
            LCALL   ON_R6
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E6_BUSY:  LJMP    SL1_BRET
SL1_NO7:    CJNE    A,#07H,SL1_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E1E7_BUSY
            MOV     R_EXT7,#01H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E7_BUSY:  LJMP    SL1_BRET
SL1_NO8:    CJNE    A,#08H,SL1_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E1E8_BUSY
            MOV     R_EXT8,#01H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL1
            LJMP    INT_1
E1E8_BUSY:  LJMP    SL1_BRET
SL1_NO9:    CJNE    A,#09H,SL1_NOB
            MOV     A,P_NOSL1
            SJMP    $+5
            LJMP    SL1_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E1TK2
```

```
            MOV     EXT1,#09H
            MOV     R7,#07FH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL1
            LCALL   OFF_DFS1
            ORL     OUT_T1,#01H
            LCALL   DELAY05
            LJMP    INT_1
C_E1TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E1T_BUSY
            MOV     EXT1,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL1
            LCALL   OFF_DFS1
            ORL     OUT_T2,#01H
            LCALL   DELAY05
            LJMP    INT_1
E1T_BUSY:   LJMP    SL1_BRET
SL1_NO0:    CJNE    A,#0AH,SL1_NOB
            LCALL   DELAY90
            LJMP    SL1_BRET
SL1_NOB:    CJNE    A,#0BH,SL1_NOC
            MOV     A,P_NOSL1
            CJNE    A,#0BBH,SL1_KF
            LJMP    SL1_BRET
SL1_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL1,A
            LJMP    INT_1
SL1_NOC:    CJNE    A,#0CH,JSL1_BRET
            MOV     A,P_NOSL1
            CJNE    A,#0FFH,SL1_BRET
            ORL     FNC3,#01H
            MOV     A,TKE1
            ANL     A,#01H
            CJNE    A,#01H,SL1_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL1_BRET
            MOV     R7,#0FFH          ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL1
            LCALL   OFF_DFS1
            LJMP    INT_1
SL1_F3T2:   MOV     A,TKE2
            ANL     A,#01H
            CJNE    A,#01H,SL1_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL1_BRET
            MOV     R7,#0FFH          ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL1
            LCALL   OFF_DFS1
            LJMP    INT_1
SL1_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL1
            ANL     FNC1,#0FEH
            ANL     FNC2,#0FEH
            ANL     FNC3,#0FEH
            MOV     P_NOSL1,#0FFH
            LJMP    INT_1
;*********************************************
SL2_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL2_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E2E1_BUSY
            MOV     R_EXT1,#02H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E1_BUSY:  LJMP    SL2_BRET
SL2_NO2:    CJNE    A,#02H,SL2_NO3
            LJMP    SL2_BRET
SL2_NO3:    CJNE    A,#03H,SL2_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E2E3_BUSY
            MOV     R_EXT3,#02H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E3_BUSY:  LJMP    SL2_BRET
SL2_NO4:    CJNE    A,#04H,SL2_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E2E4_BUSY
            MOV     R_EXT4,#02H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E4_BUSY:  LJMP    SL2_BRET
SL2_NO5:    CJNE    A,#05H,SL2_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E2E5_BUSY
            MOV     R_EXT5,#02H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E5_BUSY:  LJMP    SL2_BRET
SL2_NO6:    CJNE    A,#06H,SL2_NO7
            LCALL   LD_SEXT
```

```
            JNB     ACC.5,E2E6_BUSY
            MOV     R_EXT6,#02H
            LCALL   ON_R6
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E6_BUSY:  LJMP    SL2_BRET
SL2_NO7:    CJNE    A,#07H,SL2_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E2E7_BUSY
            MOV     R_EXT7,#02H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E7_BUSY:  LJMP    SL2_BRET

SL2_NO8:    CJNE    A,#08H,SL2_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E2E8_BUSY
            MOV     R_EXT8,#02H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL2
            LJMP    INT_1
E2E8_BUSY:  LJMP    SL2_BRET

SL2_NO9:    CJNE    A,#09H,SL2_NOB
            MOV     A,P_NOSL2
            SJMP    $+5
            LJMP    SL2_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E2TK2
            MOV     EXT2,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL2
            LCALL   OFF_DFS2
            ORL     OUT_T1,#02H
            LCALL   DELAY05
            LJMP    INT_1
C_E2TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E2T_BUSY
            MOV     EXT2,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL2
            LCALL   OFF_DFS2
            ORL     OUT_T2,#02H
            LCALL   DELAY05
            LJMP    INT_1
E2T_BUSY:   LJMP    SL2_BRET
SL2_NO0:    CJNE    A,#0AH,SL2_NOB
            LJMP    SL2_BRET
SL2_NOB:    CJNE    A,#0BH,SL2_NOC
            MOV     A,P_NOSL2
            CJNE    A,#0BBH,SL2_KF
            LJMP    SL2_BRET
SL2_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL2,A
            LJMP    INT_1
JSL2_BR:    LJMP    SL2_BRET
SL2_NOC:    CJNE    A,#0CH,JSL2_BR
            MOV     A,P_NOSL2
            CJNE    A,#0FFH,SL2_BRET
            ORL     FNC3,#02H
            MOV     A,TKE1
            ANL     A,#02H
            CJNE    A,#02H,SL2_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL2_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL2
            LCALL   OFF_DFS2
            LJMP    INT_1
SL2_F3T2:   MOV     A,TKE2
            ANL     A,#02H
            CJNE    A,#02H,SL2_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL2_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL2
            LCALL   OFF_DFS2
            LJMP    INT_1
SL2_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL2
            ANL     FNC1,#0FDH
            ANL     FNC2,#0FDH
            ANL     FNC3,#0FDH
            MOV     P_NOSL2,#0FFH
            LJMP    INT_1
;*********************************************************************
SL3_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL3_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E3E1_BUSY
            MOV     R_EXT1,#03H
            LCALL   ON_R1
```

```
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E1_BUSY:  LJMP    SL3_BRET
SL3_NO2:    CJNE    A,#02H,SL3_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E3E2_BUSY
            MOV     R_EXT2,#03H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E2_BUSY:  LJMP    SL3_BRET
SL3_NO3:    CJNE    A,#03H,SL3_NO4
            LJMP    SL3_BRET
SL3_NO4:    CJNE    A,#04H,SL3_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E3E4_BUSY
            MOV     R_EXT4,#03H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E4_BUSY:  LJMP    SL3_BRET
SL3_NO5:    CJNE    A,#05H,SL3_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E3E5_BUSY
            MOV     R_EXT5,#03H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E5_BUSY:  LJMP    SL3_BRET
SL3_NO6:    CJNE    A,#06H,SL3_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E3E6_BUSY
            MOV     R_EXT6,#03H
            LCALL   ON_R6
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E6_BUSY:  LJMP    SL3_BRET
SL3_NO7:    CJNE    A,#07H,SL3_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E3E7_BUSY
            MOV     R_EXT7,#03H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E7_BUSY:  LJMP    SL3_BRET
SL3_NO8:    CJNE    A,#08H,SL3_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E3E8_BUSY
            MOV     R_EXT8,#03H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL3
            LJMP    INT_1
E3E8_BUSY:  LJMP    SL3_BRET
SL3_NO9:    CJNE    A,#09H,SL3_NOB
            MOV     A,P_NOSL3
            SJMP    $+5
            LJMP    SL3_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E3TK2
            MOV     EXT3,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL3
            LCALL   OFF_DFS3
            ORL     OUT_T1,#04H
            LCALL   DELAY05
            LJMP    INT_1
C_E3TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E3T_BUSY
            MOV     EXT3,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL3
            LCALL   OFF_DFS3
            ORL     OUT_T2,#04H
            LCALL   DELAY05
            LJMP    INT_1
E3T_BUSY:   LJMP    SL3_BRET
SL3_NO0:    CJNE    A,#0AH,SL3_NOB
            LJMP    SL3_BRET
SL3_NOB:    CJNE    A,#0BH,SL3_NOC
            MOV     A,P_NOSL3
            CJNE    A,#0BBH,SL3_KF
            LJMP    SL3_BRET
SL3_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL3,A
            LJMP    INT_1
JSL3_BR:    LJMP    SL3_BRET
SL3_NOC:    CJNE    A,#0CH,JSL3_BR
            MOV     A,P_NOSL3
            CJNE    A,#0FFH,SL3_BRET
            ORL     FNC3,#04H
            MOV     A,TKE1
```

```
            ANL     A,#04H
            CJNE    A,#04H,SL3_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL3_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL3
            LCALL   OFF_DFS3
            LJMP    INT_1
SL3_F3T2:   MOV     A,TKE2
            ANL     A,#04H
            CJNE    A,#04H,SL3_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL3_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL3
            LCALL   OFF_DFS3
            LJMP    INT_1
SL3_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL3
            ANL     FNC1,#0FBH
            ANL     FNC2,#0FBH
            ANL     FNC3,#0FBH
            MOV     P_NOSL3,#0FFH
            LJMP    INT_1
;****************************************
SL4_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL4_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E4E1_BUSY
            MOV     R_EXT1,#04H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E1_BUSY:  LJMP    SL4_BRET
SL4_NO2:    CJNE    A,#02H,SL4_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E4E2_BUSY
            MOV     R_EXT2,#04H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E2_BUSY:  LJMP    SL4_BRET
SL4_NO3:    CJNE    A,#03H,SL4_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E4E3_BUSY
            MOV     R_EXT3,#04H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E3_BUSY:  LJMP    SL4_BRET
SL4_NO4:    CJNE    A,#04H,SL4_NO5
            LJMP    SL4_BRET
SL4_NO5:    CJNE    A,#05H,SL4_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E4E5_BUSY
            MOV     R_EXT5,#04H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E5_BUSY:  LJMP    SL4_BRET
SL4_NO6:    CJNE    A,#06H,SL4_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E4E6_BUSY
            MOV     R_EXT6,#04H
            LCALL   ON_R6
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E6_BUSY:  LJMP    SL4_BRET
SL4_NO7:    CJNE    A,#07H,SL4_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E4E7_BUSY
            MOV     R_EXT7,#04H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E7_BUSY:  LJMP    SL4_BRET
SL4_NO8:    CJNE    A,#08H,SL4_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E4E8_BUSY
            MOV     R_EXT8,#04H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL4
            LJMP    INT_1
E4E8_BUSY:  LJMP    SL4_BRET
SL4_NO9:    CJNE    A,#09H,SL4_NOB
            MOV     A,P_NOSL4
            SJMP    $+5
            LJMP    SL4_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E4TK2
            MOV     EXT4,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL4
```

```
            LCALL   OFF_DFS4
            ORL     OUT_T1,#08H
            LCALL   DELAY05
            LJMP    INT_1
C_E4TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E4T_BUSY
            MOV     EXT4,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL4
            LCALL   OFF_DFS4
            ORL     OUT_T2,#08H
            LCALL   DELAY05
            LJMP    INT_1
E4T_BUSY:   LJMP    SL4_BRET
SL4_NO0:    CJNE    A,#0AH,SL4_NOB
            LJMP    SL4_BRET
SL4_NOB:    CJNE    A,#0BH,SL4_NOC
            MOV     A,P_NOSL4
            CJNE    A,#0BBH,SL4_KF
            LJMP    SL4_BRET
SL4_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL4,A
            LJMP    INT_1
JSL4_BR:    LJMP    SL4_BRET
SL4_NOC:    CJNE    A,#0CH,JSL4_BR
            MOV     A,P_NOSL4
            CJNE    A,#0FFH,SL4_BRET
            ORL     FNC3,#08H
            MOV     A,TKE1
            ANL     A,#08H
            CJNE    A,#08H,SL4_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL4_BRET
            MOV     R7,#0FFH        ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL4
            LCALL   OFF_DFS4
            LJMP    INT_1
SL4_F3T2:   MOV     A,TKE2
            ANL     A,#08H
            CJNE    A,#08H,SL4_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL4_BRET
            MOV     R7,#0FFH        ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL4
            LCALL   OFF_DFS4
            LJMP    INT_1
SL4_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL4
            ANL     FNC1,#0F7H
            ANL     FNC2,#0F7H
            ANL     FNC3,#0F7H
            MOV     P_NOSL4,#0FFH
            LJMP    INT_1

;*********************************************************************
SL5_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL5_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E5E1_BUSY
            MOV     R_EXT1,#05H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E1_BUSY:  LJMP    SL5_BRET
SL5_NO2:    CJNE    A,#02H,SL5_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E5E2_BUSY
            MOV     R_EXT2,#05H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E2_BUSY:  LJMP    SL5_BRET
SL5_NO3:    CJNE    A,#03H,SL5_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E5E3_BUSY
            MOV     R_EXT3,#05H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E3_BUSY:  LJMP    SL5_BRET
SL5_NO4:    CJNE    A,#04H,SL5_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E5E4_BUSY
            MOV     R_EXT4,#05H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E4_BUSY:  LJMP    SL5_BRET
SL5_NO5:    CJNE    A,#05H,SL5_NO6
            LJMP    SL5_BRET
SL5_NO6:    CJNE    A,#06H,SL5_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E5E6_BUSY
            MOV     R_EXT6,#05H
            LCALL   ON_R6
            MOV     R7,#0FFH
```

```
            LCALL   RBT_SL5
            LJMP    INT_1
E5E6_BUSY:  LJMP    SL5_BRET
SL5_NO7:    CJNE    A,#07H,SL5_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E5E7_BUSY
            MOV     R_EXT7,#05H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E7_BUSY:  LJMP    SL5_BRET
SL5_NO8:    CJNE    A,#08H,SL5_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E5E8_BUSY
            MOV     R_EXT8,#05H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL5
            LJMP    INT_1
E5E8_BUSY:  LJMP    SL5_BRET
SL5_NO9:    CJNE    A,#09H,SL5_NOB
            MOV     A,P_NOSL5
            SJMP    $+5
            LJMP    SL5_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E5TK2
            MOV     EXT5,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL5
            LCALL   OFF_DFS5
            ORL     OUT_T1,#10H
            LCALL   DELAY05
            LJMP    INT_1
C_E5TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E5T_BUSY
            MOV     EXT5,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL5
            LCALL   OFF_DFS5
            ORL     OUT_T2,#10H
            LCALL   DELAY05
            LJMP    INT_1
E5T_BUSY:   LJMP    SL5_BRET
SL5_NO0:    CJNE    A,#0AH,SL5_NOB
            LJMP    SL5_BRET
SL5_NOB:    CJNE    A,#0BH,SL5_NOC
            MOV     A,P_NOSL5
            CJNE    A,#0BBH,SL5_KF
            LJMP    SL5_BRET
SL5_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL5,A
            LJMP    INT_1
JSL5_BR:    LJMP    SL5_BRET
SL5_NOC:    CJNE    A,#0CH,JSL5_BR
            MOV     A,P_NOSL5
            CJNE    A,#0FFH,SL5_BRET
            ORL     FNC3,#10H
            MOV     A,TKE1
            ANL     A,#10H
            CJNE    A,#10H,SL5_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL5_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL5
            LCALL   OFF_DFS5
            LJMP    INT_1
SL5_F3T2:   MOV     A,TKE2
            ANL     A,#10H
            CJNE    A,#10H,SL5_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL5_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL5
            LCALL   OFF_DFS5
            LJMP    INT_1
SL5_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL5
            ANL     FNC1,#0EFH
            ANL     FNC2,#0EFH
            ANL     FNC3,#0EFH
            MOV     P_NOSL5,#0FFH
            LJMP    INT_1
;*****************************************************************
SL6_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL6_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E6E1_BUSY
            MOV     R_EXT1,#06H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E1_BUSY:  LJMP    SL6_BRET
SL6_NO2:    CJNE    A,#02H,SL6_NO3
            LCALL   LD_SEXT
```

```
            JNB     ACC.1,E6E2_BUSY
            MOV     R_EXT2,#06H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E2_BUSY:  LJMP    SL6_BRET
SL6_NO3:    CJNE    A,#03H,SL6_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E6E3_BUSY
            MOV     R_EXT3,#06H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E3_BUSY:  LJMP    SL6_BRET
SL6_NO4:    CJNE    A,#04H,SL6_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E6E4_BUSY
            MOV     R_EXT4,#06H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E4_BUSY:  LJMP    SL6_BRET
SL6_NO5:    CJNE    A,#05H,SL6_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E6E5_BUSY
            MOV     R_EXT5,#06H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E5_BUSY:  LJMP    SL6_BRET
SL6_NO6:    CJNE    A,#06H,SL6_NO7
            LJMP    SL6_BRET
SL6_NO7:    CJNE    A,#07H,SL6_NO8
            LCALL   LD_SEXT
            JNB     ACC.6,E6E7_BUSY
            MOV     R_EXT7,#06H
            LCALL   ON_R7
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
E6E7_BUSY:  LJMP    SL6_BRET
SL6_NO8:    CJNE    A,#08H,SL6_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E6E8_BUSY
            MOV     R_EXT8,#06H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL6
            LJMP    INT_1
```

```
E6E8_BUSY:  LJMP    SL6_BRET
SL6_NO9:    CJNE    A,#09H,SL6_NOB
            MOV     A,P_NOSL6
            SJMP    $+5
            LJMP    SL6_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E6TK2
            MOV     EXT6,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL6
            LCALL   OFF_DFS6
            ORL     OUT_T1,#20H
            LCALL   DELAY05
            LJMP    INT_1
C_E6TK2:    MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#00H,E6T_BUSY
            MOV     EXT6,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL6
            LCALL   OFF_DFS6
            ORL     OUT_T2,#20H
            LCALL   DELAY05
            LJMP    INT_1
E6T_BUSY:   LJMP    SL6_BRET
SL6_NO0:    CJNE    A,#0AH,SL6_NOB
            LJMP    SL6_BRET
SL6_NOB:    CJNE    A,#0BH,SL6_NOC
            MOV     A,P_NOSL6
            CJNE    A,#0BBH,SL6_KF
            LJMP    SL6_BRET
SL6_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL6,A
            LJMP    INT_1
JSL6_BR:    LJMP    SL6_BRET
SL6_NOC:    CJNE    A,#0CH,JSL6_BR
            MOV     A,P_NOSL6
            CJNE    A,#0FFH,SL6_BRET
            ORL     FNC3,#20H
            MOV     A,TKE1
            ANL     A,#20H
            CJNE    A,#20H,SL6_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL6_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK1 BY "#"
```

```
            LCALL   TK1_SL6
            LCALL   OFF_DFS6
            LJMP    INT_1
SL6_F3T2:   MOV     A,TKE2
            ANL     A,#20H
            CJNE    A,#20H,SL6_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL6_BRET
            MOV     R7,#0FFH            ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL6
            LCALL   OFF_DFS6
            LJMP    INT_1
SL6_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL6
            ANL     FNC1,#0DFH
            ANL     FNC2,#0DFH
            ANL     FNC3,#0DFH
            MOV     P_NOSL6,#0FFH
            LJMP    INT_1
;*************************************************************
SL7_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL7_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E7E1_BUSY
            MOV     R_EXT1,#07H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E1_BUSY:  LJMP    SL7_BRET

SL7_NO2:    CJNE    A,#02H,SL7_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E7E2_BUSY
            MOV     R_EXT2,#07H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E2_BUSY:  LJMP    SL7_BRET
SL7_NO3:    CJNE    A,#03H,SL7_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E7E3_BUSY
            MOV     R_EXT3,#07H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E3_BUSY:  LJMP    SL7_BRET
SL7_NO4:    CJNE    A,#04H,SL7_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E7E4_BUSY
            MOV     R_EXT4,#07H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E4_BUSY:  LJMP    SL7_BRET
SL7_NO5:    CJNE    A,#05H,SL7_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E7E5_BUSY
            MOV     R_EXT5,#07H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E5_BUSY:  LJMP    SL7_BRET
SL7_NO6:    CJNE    A,#06H,SL7_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E7E6_BUSY
            MOV     R_EXT6,#07H
            LCALL   ON_R6
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E6_BUSY:  LJMP    SL7_BRET
SL7_NO7:    CJNE    A,#07H,SL7_NO8
            LJMP    SL7_BRET
SL7_NO8:    CJNE    A,#08H,SL7_NO9
            LCALL   LD_SEXT
            JNB     ACC.7,E7E8_BUSY
            MOV     R_EXT8,#07H
            LCALL   ON_R8
            MOV     R7,#0FFH
            LCALL   RBT_SL7
            LJMP    INT_1
E7E8_BUSY:  LJMP    SL7_BRET
SL7_NO9:    CJNE    A,#09H,SL7_NOB
            MOV     A,P_NOSL7
            SJMP    $+5
            LJMP    SL7_BRET
            CJNE    A,#0FFH,$-3
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#00H,C_E7TK2
            MOV     EXT7,#09H
            MOV     R7,#0FFH
            LCALL   TK1_HK
            MOV     R7,#0FFH
            LCALL   TK1_SL7
            LCALL   OFF_DFS7
            ORL     OUT_T1,#40H
            LCALL   DELAY05
            LJMP    INT_1
C_E7TK2:    MOV     A,TK_TOE
```

```
            ANL     A,#80H
            CJNE    A,#00H,E7T_BUSY
            MOV     EXT7,#0AH
            MOV     R7,#0FFH
            LCALL   TK2_HK
            MOV     R7,#0FFH
            LCALL   TK2_SL7
            LCALL   OFF_DFS7
            ORL     OUT_T2,#40H
            LCALL   DELAY05
            LJMP    INT_1
E7T_BUSY:   LJMP    SL7_BRET
SL7_NO0:    CJNE    A,#0AH,SL7_NOB
            LJMP    SL7_BRET
SL7_NOB:    CJNE    A,#0BH,SL7_NOC
            MOV     A,P_NOSL7
            CJNE    A,#0BBH,SL7_KF
            LJMP    SL7_BRET
SL7_KF:     SWAP    A
            ANL     A,#0BFH
            MOV     P_NOSL7,A
            LJMP    INT_1
JSL7_BR:    LJMP    SL7_BRET
SL7_NOC:    CJNE    A,#0CH,JSL7_BR
            MOV     A,P_NOSL7
            CJNE    A,#0FFH,SL7_BRET
            ORL     FNC3,#40H
            MOV     A,TKE1
            ANL     A,#40H
            CJNE    A,#40H,SL7_F3T2
            MOV     A,TK_TOE
            ANL     A,#40H
            CJNE    A,#40H,SL7_BRET
            MOV     R7,#0FFH        ;RETURN TO CONNECT TK1 BY "#"
            LCALL   TK1_SL7
            LCALL   OFF_DFS7
            LJMP    INT_1
SL7_F3T2:   MOV     A,TKE2
            ANL     A,#40H
            CJNE    A,#40H,SL7_BRET
            MOV     A,TK_TOE
            ANL     A,#80H
            CJNE    A,#80H,SL7_BRET
            MOV     R7,#0FFH        ;RETURN TO CONNECT TK2 BY "#"
            LCALL   TK2_SL7
            LCALL   OFF_DFS7
            LJMP    INT_1
SL7_BRET:   MOV     R7,#0FFH
            LCALL   BT_SL7
            ANL     FNC1,#0BFH
            ANL     FNC2,#0BFH
            ANL     FNC3,#0BFH
            MOV     P_NOSL7,#0FFH
            LJMP    INT_1
;*************************************************************************
SL8_NO1:    MOV     A,DF_BUF
            CJNE    A,#01H,SL8_NO2
            LCALL   LD_SEXT
            JNB     ACC.0,E8E1_BUSY
            MOV     R_EXT1,#08H
            LCALL   ON_R1
            MOV     R7,#0FFH
            LCALL   RBT_SL8
            LJMP    INT_1
E8E1_BUSY:  LJMP    SL8_BRET
SL8_NO2:    CJNE    A,#02H,SL8_NO3
            LCALL   LD_SEXT
            JNB     ACC.1,E8E2_BUSY
            MOV     R_EXT2,#08H
            LCALL   ON_R2
            MOV     R7,#0FFH
            LCALL   RBT_SL8
            LJMP    INT_1
E8E2_BUSY:  LJMP    SL8_BRET
SL8_NO3:    CJNE    A,#03H,SL8_NO4
            LCALL   LD_SEXT
            JNB     ACC.2,E8E3_BUSY
            MOV     R_EXT3,#08H
            LCALL   ON_R3
            MOV     R7,#0FFH
            LCALL   RBT_SL8
            LJMP    INT_1
E8E3_BUSY:  LJMP    SL8_BRET
SL8_NO4:    CJNE    A,#04H,SL8_NO5
            LCALL   LD_SEXT
            JNB     ACC.3,E8E4_BUSY
            MOV     R_EXT4,#08H
            LCALL   ON_R4
            MOV     R7,#0FFH
            LCALL   RBT_SL8
            LJMP    INT_1
E8E4_BUSY:  LJMP    SL8_BRET
SL8_NO5:    CJNE    A,#05H,SL8_NO6
            LCALL   LD_SEXT
            JNB     ACC.4,E8E5_BUSY
            MOV     R_EXT5,#08H
            LCALL   ON_R5
            MOV     R7,#0FFH
            LCALL   RBT_SL8
            LJMP    INT_1
E8E5_BUSY:  LJMP    SL8_BRET
SL8_NO6:    CJNE    A,#06H,SL8_NO7
            LCALL   LD_SEXT
            JNB     ACC.5,E8E6_BUSY
            MOV     R_EXT6,#07H
            LCALL   ON_R6
```

```
                MOV     R7,#0FFH
                LCALL   RBT_SL8
                LJMP    INT_1
E8E6_BUSY:      LJMP    SL8_BRET
SL8_NO7:        CJNE    A,#07H,SL8_NO8
                LCALL   LD_SEXT
                JNB     ACC.6,E8E7_BUSY
                MOV     R_EXT7,#08H
                LCALL   ON_R7
                MOV     R7,#0FFH
                LCALL   RBT_SL8
                LJMP    INT_1
E8E7_BUSY:      LJMP    SL8_BRET
SL8_NO8:        CJNE    A,#08H,SL8_NO9
                LJMP    SL8_BRET
SL8_NO9:        CJNE    A,#09H,SL8_NOB
                MOV     A,P_NOSL8
                SJMP    $+5
                LJMP    SL8_BRET
                CJNE    A,#0FFH,$-3
                MOV     A,TK_TOE
                ANL     A,#40H
                CJNE    A,#00H,C_E8TK2
                MOV     EXT8,#09H
                MOV     R7,#0FFH
                LCALL   TK1_HK
                MOV     R7,#0FFH
                LCALL   TK1_SL8
                LCALL   OFF_DFS8
                ORL     OUT_T1,#80H
                LCALL   DELAY05
                LJMP    INT_1
C_E8TK2:        MOV     A,TK_TOE
                ANL     A,#80H
                CJNE    A,#00H,E8T_BUSY
                MOV     EXT8,#0AH
                MOV     R7,#0FFH
                LCALL   TK2_HK
                MOV     R7,#0FFH
                LCALL   TK2_SL8
                LCALL   OFF_DFS8
                ORL     OUT_T2,#80H
                LCALL   DELAY05
                LJMP    INT_1
E8T_BUSY:       LJMP    SL8_BRET
SL8_NO0:        CJNE    A,#0AH,SL8_NOB
                LJMP    SL8_BRET
SL8_NOB:        CJNE    A,#0BH,SL8_NOC
                MOV     A,P_NOSL8
                CJNE    A,#0BBH,SL8_KF
                LJMP    SL8_BRET
SL8_KF:         SWAP    A
                ANL     A,#0BFH
                MOV     P_NOSL8,A
                LJMP    INT_1
JSL8_BR:        LJMP    SL8_BRET
SL8_NOC:        CJNE    A,#0CH,JSL8_BR
                MOV     A,P_NOSL8
                CJNE    A,#0FFH,SL8_BRET
                ORL     FNC3,#80H
                MOV     A,TKE1
                ANL     A,#80H
                CJNE    A,#80H,SL8_F3T2
                MOV     A,TK_TOE
                ANL     A,#40H
                CJNE    A,#40H,SL8_BRET
                MOV     R7,#0FFH            ;RETURN TO CONNECT TK1 BY "#"
                LCALL   TK1_SL8
                LCALL   OFF_DFS8
                LJMP    INT_1
SL8_F3T2:       MOV     A,TKE2
                ANL     A,#80H
                CJNE    A,#80H,SL8_BRET
                MOV     A,TK_TOE
                ANL     A,#80H
                CJNE    A,#80H,SL8_BRET
                MOV     R7,#0FFH            ;RETURN TO CONNECT TK2 BY "#"
                LCALL   TK2_SL8
                LCALL   OFF_DFS8
                LJMP    INT_1
SL8_BRET:       MOV     R7,#0FFH
                LCALL   BT_SL8
                ANL     FNC1,#7FH
                ANL     FNC2,#7FH
                ANL     FNC3,#7FH
                MOV     P_NOSL8,#0FFH
                LJMP    INT_1
;************ LOAD STATUS OF EXT. LINES ************
LD_SEXT:        MOV     DPTR,#PORTA_4
                MOVX    A,@DPTR
                RET
;************** LOAD BCD FROM PORT *****************
TK_BCD:         MOV     DPTR,#PORTB_1
                MOVX    A,@DPTR
                ANL     A,#0FH
                RET
SL_BCD:         MOV     DPTR,#PORTB_1
                MOVX    A,@DPTR
                ANL     A,#0F0H
                SWAP    A
                RET
;************** LOAD STD FROM PORT *****************
LD_STD:         MOV     DPTR,#PORTC_1
                MOVX    A,@DPTR
                ANL     A,#3FH          ; '00111111'
                RET
```

```
;************** ON TOE FOR RECEIVE BCD ************
ON_TOE1:    ORL     TK_TOE,#01H
            MOV     A,TK_TOE
            SJMP    LD_TOE
ON_TOE2:    ORL     TK_TOE,#02H
            MOV     A,TK_TOE
            SJMP    LD_TOE
ON_TOE3:    ORL     TK_TOE,#04H
            MOV     A,TK_TOE
            SJMP    LD_TOE
ON_TOE4:    ORL     TK_TOE,#08H
            MOV     A,TK_TOE
            SJMP    LD_TOE
ON_TOE5:    ORL     TK_TOE,#10H
            MOV     A,TK_TOE
            SJMP    LD_TOE
ON_TOE6:    ORL     TK_TOE,#20H
            MOV     A,TK_TOE
            SJMP    LD_TOE
;************** OFF TOE FOR RECEIVE BCD ************
OFF_TOE1:   ANL     TK_TOE,#0FEH
            MOV     A,TK_TOE
            SJMP    LD_TOE
OFF_TOE2:   ANL     TK_TOE,#0FDH
            MOV     A,TK_TOE
            SJMP    LD_TOE
OFF_TOE3:   ANL     TK_TOE,#0FBH
            MOV     A,TK_TOE
            SJMP    LD_TOE
OFF_TOE4:   ANL     TK_TOE,#0F7H
            MOV     A,TK_TOE
            SJMP    LD_TOE
OFF_TOE5:   ANL     TK_TOE,#0EFH
            MOV     A,TK_TOE
            SJMP    LD_TOE
OFF_TOE6:   ANL     TK_TOE,#0DFH
            MOV     A,TK_TOE
            SJMP    LD_TOE
;************** LOAD TOE TO PORT ******************
LD_TOE:     MOV     DPTR,#PORTA_1
            MOVX    @DPTR,A
            RET
;********** ON RINGING FOR EXT. LINES ************
ON_R1:      SETB    EXT_RG.0
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R2:      SETB    EXT_RG.1
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R3:      SETB    EXT_RG.2
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R4:      SETB    EXT_RG.3
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R5:      SETB    EXT_RG.4
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R6:      SETB    EXT_RG.5
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R7:      SETB    EXT_RG.6
            MOV     A,EXT_RG
            SJMP    LD_RG
ON_R8:      SETB    EXT_RG.7
            MOV     A,EXT_RG
            SJMP    LD_RG
;********** OFF RINGING FOR EXT. LINES ***********
OFF_R1:     CLR     EXT_RG.0
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R2:     CLR     EXT_RG.1
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R3:     CLR     EXT_RG.2
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R4:     CLR     EXT_RG.3
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R5:     CLR     EXT_RG.4
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R6:     CLR     EXT_RG.5
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R7:     CLR     EXT_RG.6
            MOV     A,EXT_RG
            SJMP    LD_RG
OFF_R8:     CLR     EXT_RG.7
            MOV     A,EXT_RG
            SJMP    LD_RG
;************** LOAD RINGING TO EXT. LINES ******************
LD_RG:      MOV     DPTR,#PORTB_4
            MOVX    @DPTR,A
            RET
;****************** CROSS POINT ***********************
CROSS1_1:   MOV     DPTR,#PORTC_2
            MOV     A,#01H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_1ON
            CLR     CROSSE1.0
            MOV     A,#00H
            SJMP    DI1_1
C1_1ON:     SETB    CROSSE1.0
            MOV     A,#01H
```

```
DI1_1:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS1_2:   MOV     DPTR,#PORTC_2
            MOV     A,#09H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_2ON
            CLR     CROSSE1.1
            MOV     A,#00H
            SJMP    DI1_2
C1_2ON:     SETB    CROSSE1.1
            MOV     A,#01H
DI1_2:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS1_3:   MOV     DPTR,#PORTC_2
            MOV     A,#05H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_3ON
            CLR     CROSSE1.2
            MOV     A,#00H
            SJMP    DI1_3
C1_3ON:     SETB    CROSSE1.2
            MOV     A,#01H
DI1_3:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS1_4:   MOV     DPTR,#PORTC_2
            MOV     A,#0DH
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_4ON
            CLR     CROSSE1.3
            MOV     A,#00H
            SJMP    DI1_4
C1_4ON:     SETB    CROSSE1.3
            MOV     A,#01H
DI1_4:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS1_5:   MOV     DPTR,#PORTC_2
            MOV     A,#10H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_5ON
            CLR     CROSSE1.4
            MOV     A,#00H
            SJMP    DI1_5
C1_5ON:     SETB    CROSSE1.4
            MOV     A,#02H
DI1_5:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS1_6:   MOV     DPTR,#PORTC_2
            MOV     A,#90H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_6ON
            CLR     CROSSE1.5
            MOV     A,#00H
            SJMP    DI1_6
C1_6ON:     SETB    CROSSE1.5
            MOV     A,#02H
DI1_6:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS1_7:   MOV     DPTR,#PORTC_2
            MOV     A,#50H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_7ON
            CLR     CROSSE1.6
            MOV     A,#00H
            SJMP    DI1_7
C1_7ON:     SETB    CROSSE1.6
            MOV     A,#02H
DI1_7:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS1_8:   MOV     DPTR,#PORTC_2
            MOV     A,#0D0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C1_8ON
            CLR     CROSSE1.7
            MOV     A,#00H
            SJMP    DI1_8
C1_8ON:     SETB    CROSSE1.7
            MOV     A,#02H
DI1_8:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS2_1:   MOV     DPTR,#PORTC_2
            MOV     A,#03H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_1ON
            CLR     CROSSE2.0
            MOV     A,#00H
            SJMP    DI2_1
C2_1ON:     SETB    CROSSE2.0
            MOV     A,#01H
DI2_1:      LCALL   DI
            MOV     A,#01H
```

```
            LCALL   STB
            RET
CROSS2_2:   MOV     DPTR,#PORTC_2
            MOV     A,#0BH
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_2ON
            CLR     CROSSE2.1
            MOV     A,#00H
            SJMP    DI2_2
C2_2ON:     SETB    CROSSE2.1
            MOV     A,#01H
DI2_2:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS2_3:   MOV     DPTR,#PORTC_2
            MOV     A,#07H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_3ON
            CLR     CROSSE2.2
            MOV     A,#00H
            SJMP    DI2_3
C2_3ON:     SETB    CROSSE2.2
            MOV     A,#01H
DI2_3:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS2_4:   MOV     DPTR,#PORTC_2
            MOV     A,#0FH
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_4ON
            CLR     CROSSE2.3
            MOV     A,#00H
            SJMP    DI2_4
C2_4ON:     SETB    CROSSE2.3
            MOV     A,#01H
DI2_4:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
CROSS2_5:   MOV     DPTR,#PORTC_2
            MOV     A,#30H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_5ON
            CLR     CROSSE2.4
            MOV     A,#00H
            SJMP    DI2_5
C2_5ON:     SETB    CROSSE2.4
            MOV     A,#02H
DI2_5:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS2_6:   MOV     DPTR,#PORTC_2
            MOV     A,#0B0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_6ON
            CLR     CROSSE2.5
            MOV     A,#00H
            SJMP    DI2_6
C2_6ON:     SETB    CROSSE2.5
            MOV     A,#02H
DI2_6:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS2_7:   MOV     DPTR,#PORTC_2
            MOV     A,#70H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_7ON
            CLR     CROSSE2.6
            MOV     A,#00H
            SJMP    DI2_7
C2_7ON:     SETB    CROSSE2.6
            MOV     A,#02H
DI2_7:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS2_8:   MOV     DPTR,#PORTC_2
            MOV     A,#0F0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C2_8ON
            CLR     CROSSE2.7
            MOV     A,#00H
            SJMP    DI2_8
C2_8ON:     SETB    CROSSE2.7
            MOV     A,#02H
DI2_8:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
CROSS3_1:   MOV     DPTR,#PORTA_3
            MOV     A,#30H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_1ON
            CLR     CROSSE3.0
            MOV     A,#00H
            SJMP    DI3_1
C3_1ON:     SETB    CROSSE3.0
            MOV     A,#08H
DI3_1:      LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
```

```
CROSS3_2:   MOV     DPTR,#PORTA_3
            MOV     A,#0B0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_2ON
            CLR     CROSSE3.1
            MOV     A,#00H
            SJMP    DI3_2
C3_2ON:     SETB    CROSSE3.1
            MOV     A,#08H
DI3_2:      LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
CROSS3_3:   MOV     DPTR,#PORTA_3
            MOV     A,#70H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_3ON
            CLR     CROSSE3.2
            MOV     A,#00H
            SJMP    DI3_3
C3_3ON:     SETB    CROSSE3.2
            MOV     A,#08H
DI3_3:      LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
CROSS3_4:   MOV     DPTR,#PORTA_3
            MOV     A,#0F0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_4ON
            CLR     CROSSE3.3
            MOV     A,#00H
            SJMP    DI3_4
C3_4ON:     SETB    CROSSE3.3
            MOV     A,#08H
DI3_4:      LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
CROSS3_5:   MOV     DPTR,#PORTB_3
            MOV     A,#03H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_5ON
            CLR     CROSSE3.4
            MOV     A,#00H
            SJMP    DI3_5
C3_5ON:     SETB    CROSSE3.4
            MOV     A,#10H
DI3_5:      LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
CROSS3_6:   MOV     DPTR,#PORTB_3
            MOV     A,#0B0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_6ON
            CLR     CROSSE3.5
            MOV     A,#00H
            SJMP    DI3_6
C3_6ON:     SETB    CROSSE3.5
            MOV     A,#10H
DI3_6:      LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
CROSS3_7:   MOV     DPTR,#PORTB_3
            MOV     A,#07H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_7ON
            CLR     CROSSE3.6
            MOV     A,#00H
            SJMP    DI3_7
C3_7ON:     SETB    CROSSE3.6
            MOV     A,#10H
DI3_7:      LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
CROSS3_8:   MOV     DPTR,#PORTB_3
            MOV     A,#01H
            MOVX    @DPTR,A
            CJNE    R7,#00H,C3_8ON
            CLR     CROSSE3.7
            MOV     A,#00H
            SJMP    DI3_8
C3_8ON:     SETB    CROSSE3.7
            MOV     A,#10H
DI3_8:      LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET

;************* TRUNK1 CONNECT TO SLICS ********************
TK1_SL1:    MOV     DPTR,#PORTC_2
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS11
            MOV     A,#00H
            SJMP    DTS11
TS11:       MOV     A,#01H
DTS11:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK1_SL2:    MOV     DPTR,#PORTC_2
            MOV     A,#08H
```

```
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS12
            MOV     A,#00H
            SJMP    DTS12
TS12:       MOV     A,#01H
DTS12:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK1_SL3:    MOV     DPTR,#PORTC_2
            MOV     A,#04H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS13
            MOV     A,#00H
            SJMP    DTS13
TS13:       MOV     A,#01H
DTS13:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK1_SL4:    MOV     DPTR,#PORTC_2
            MOV     A,#0CH
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS14
            MOV     A,#00H
            SJMP    DTS14
TS14:       MOV     A,#01H
DTS14:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK1_SL5:    MOV     DPTR,#PORTC_2
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS15
            MOV     A,#00H
            SJMP    DTS15
TS15:       MOV     A,#02H
DTS15:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
TK1_SL6:    MOV     DPTR,#PORTC_2
            MOV     A,#80H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS16
            MOV     A,#00H
            SJMP    DTS16
TS16:       MOV     A,#02H
DTS16:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
TK1_SL7:    MOV     DPTR,#PORTC_2
            MOV     A,#40H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS17
            MOV     A,#00H
            SJMP    DTS17
TS17:       MOV     A,#02H
DTS17:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
TK1_SL8:    MOV     DPTR,#PORTC_2
            MOV     A,#0C0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS18
            MOV     A,#00H
            SJMP    DTS18
TS18:       MOV     A,#02H
DTS18:      LCALL   DI
            MOV     A,#02H
            LCALL   STB
            RET
;***************** TRUNK2 CONNECT TO SLICS ******************
TK2_SL1:    MOV     DPTR,#PORTC_2
            MOV     A,#02H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS21
            MOV     A,#00H
            SJMP    DTS21
TS21:       MOV     A,#01H
DTS21:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK2_SL2:    MOV     DPTR,#PORTC_2
            MOV     A,#0AH
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS22
            MOV     A,#00H
            SJMP    DTS22
TS22:       MOV     A,#01H
DTS22:      LCALL   DI
            MOV     A,#01H
            LCALL   STB
            RET
TK2_SL3:    MOV     DPTR,#PORTC_2
            MOV     A,#06H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TS23
            MOV     A,#00H
            SJMP    DTS23
TS23:       MOV     A,#01H
DTS23:      LCALL   DI
```

```
                MOV     A,#01H
                LCALL   STB
                RET
TK2_SL4:        MOV     DPTR,#PORTC_2
                MOV     A,#0EH
                MOVX    @DPTR,A
                CJNE    R7,#00H,TS24
                MOV     A,#00H
                SJMP    DTS24
TS24:           MOV     A,#01H
DTS24:          LCALL   DI
                MOV     A,#01H
                LCALL   STB
                RET
TK2_SL5:        MOV     DPTR,#PORTC_2
                MOV     A,#20H
                MOVX    @DPTR,A
                CJNE    R7,#00H,TS25
                MOV     A,#00H
                SJMP    DTS25
TS25:           MOV     A,#02H
DTS25:          LCALL   DI
                MOV     A,#02H
                LCALL   STB
                RET
TK2_SL6:        MOV     DPTR,#PORTC_2
                MOV     A,#0A0H
                MOVX    @DPTR,A
                CJNE    R7,#00H,TS26
                MOV     A,#00H
                SJMP    DTS26
TS26:           MOV     A,#02H
DTS26:          LCALL   DI
                MOV     A,#02H
                LCALL   STB
                RET
TK2_SL7:        MOV     DPTR,#PORTC_2
                MOV     A,#60H
                MOVX    @DPTR,A
                CJNE    R7,#00H,TS27
                MOV     A,#00H
                SJMP    DTS27
TS27:           MOV     A,#02H
DTS27:          LCALL   DI
                MOV     A,#02H
                LCALL   STB
                RET
TK2_SL8:        MOV     DPTR,#PORTC_2
                MOV     A,#0E0H
                MOVX    @DPTR,A
                CJNE    R7,#00H,TS28
                MOV     A,#00H
                SJMP    DTS28
TS28:           MOV     A,#02H
DTS28:          LCALL   DI
                MOV     A,#02H
                LCALL   STB
                RET
;************** SUB DIAL TONE FOR EXTS **************
DT_SL1:         MOV     DPTR,#PORTA_3
                MOV     A,#20H
                MOVX    @DPTR,A
                CJNE    R7,#00H,DS1
                CLR     EXT_DT.0
                MOV     A,#00H
                SJMP    DDS1
DS1:            SETB    EXT_DT.0
                MOV     A,#08H
DDS1:           LCALL   DI
                MOV     A,#08H
                LCALL   STB
                RET
DT_SL2:         MOV     DPTR,#PORTA_3
                MOV     A,#0A0H
                MOVX    @DPTR,A
                CJNE    R7,#00H,DS2
                CLR     EXT_DT.1
                MOV     A,#00H
                SJMP    DDS2
DS2:            SETB    EXT_DT.1
                MOV     A,#08H
DDS2:           LCALL   DI
                MOV     A,#08H
                LCALL   STB
                RET
DT_SL3:         MOV     DPTR,#PORTA_3
                MOV     A,#60H
                MOVX    @DPTR,A
                CJNE    R7,#00H,DS3
                CLR     EXT_DT.2
                MOV     A,#00H
                SJMP    DDS3
DS3:            SETB    EXT_DT.2
                MOV     A,#08H
DDS3:           LCALL   DI
                MOV     A,#08H
                LCALL   STB
                RET
DT_SL4:         MOV     DPTR,#PORTA_3
                MOV     A,#0E0H
                MOVX    @DPTR,A
                CJNE    R7,#00H,DS4
                CLR     EXT_DT.3
                MOV     A,#00H
                SJMP    DDS4
DS4:            SETB    EXT_DT.3
```

```
            MOV     A,#08H
DDS4:       LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
DT_SL5:     MOV     DPTR,#PORTB_3
            MOV     A,#02H
            MOVX    @DPTR,A
            CJNE    R7,#00H,DS5
            CLR     EXT_DT.4
            MOV     A,#00H
            SJMP    DDS5
DS5:        SETB    EXT_DT.4
            MOV     A,#10H
DDS5:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
DT_SL6:     MOV     DPTR,#PORTB_3
            MOV     A,#0AH
            MOVX    @DPTR,A
            CJNE    R7,#00H,DS6
            CLR     EXT_DT.5
            MOV     A,#00H
            SJMP    DDS6
DS6:        SETB    EXT_DT.5
            MOV     A,#10H
DDS6:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
DT_SL7:     MOV     DPTR,#PORTB_3
            MOV     A,#06H
            MOVX    @DPTR,A
            CJNE    R7,#00H,DS7
            CLR     EXT_DT.6
            MOV     A,#00H
            SJMP    DDS7
DS7:        SETB    EXT_DT.6
            MOV     A,#10H
DDS7:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
DT_SL8:     MOV     DPTR,#PORTB_3
            MOV     A,#0EH
            MOVX    @DPTR,A
            CJNE    R7,#00H,DS8
            CLR     EXT_DT.7
            MOV     A,#00H
            SJMP    DDS8
DS8:        SETB    EXT_DT.7
            MOV     A,#10H
DDS8:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
;************** SUB BUSY TONE FOR EXTS ***************
BT_SL1:     MOV     DPTR,#PORTA_3
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,BS1
            CLR     EXT_BT.0
            MOV     A,#00H
            SJMP    DBS1
BS1:        SETB    EXT_BT.0
            MOV     A,#08H
DBS1:       LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
BT_SL2:     MOV     DPTR,#PORTA_3
            MOV     A,#80H
            MOVX    @DPTR,A
            CJNE    R7,#00H,BS2
            CLR     EXT_BT.1
            MOV     A,#00H
            SJMP    DBS2
BS2:        SETB    EXT_BT.1
            MOV     A,#08H
DBS2:       LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
BT_SL3:     MOV     DPTR,#PORTA_3
            MOV     A,#40H
            MOVX    @DPTR,A
            CJNE    R7,#00H,BS3
            CLR     EXT_BT.2
            MOV     A,#00H
            SJMP    DBS3
BS3:        SETB    EXT_BT.2
            MOV     A,#08H
DBS3:       LCALL   DI
            MOV     A,#08H
            LCALL   STB
            RET
BT_SL4:     MOV     DPTR,#PORTA_3
            MOV     A,#0C0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,BS4
            CLR     EXT_BT.3
            MOV     A,#00H
            SJMP    DBS4
BS4:        SETB    EXT_BT.3
            MOV     A,#08H
```

```
DBS4:     LCALL   DI
          MOV     A,#08H
          LCALL   STB
          RET
BT_SL5:   MOV     DPTR,#PORTB_3
          MOV     A,#00H
          MOVX    @DPTR,A
          CJNE    R7,#00H,BS5
          CLR     EXT_BT.4
          MOV     A,#00H
          SJMP    DBS5
BS5:      SETB    EXT_BT.4
          MOV     A,#10H
DBS5:     LCALL   DI
          MOV     A,#10H
          LCALL   STB
          RET
BT_SL6:   MOV     DPTR,#PORTB_3
          MOV     A,#08H
          MOVX    @DPTR,A
          CJNE    R7,#00H,BS6
          CLR     EXT_BT.5
          MOV     A,#00H
          SJMP    DBS6
BS6:      SETB    EXT_BT.5
          MOV     A,#10H
DBS6:     LCALL   DI
          MOV     A,#10H
          LCALL   STB
          RET
BT_SL7:   MOV     DPTR,#PORTB_3
          MOV     A,#04H
          MOVX    @DPTR,A
          CJNE    R7,#00H,BS7
          CLR     EXT_BT.6
          MOV     A,#00H
          SJMP    DBS7
BS7:      SETB    EXT_BT.6
          MOV     A,#10H
DBS7:     LCALL   DI
          MOV     A,#10H
          LCALL   STB
          RET
BT_SL8:   MOV     DPTR,#PORTB_3
          MOV     A,#0CH
          MOVX    @DPTR,A
          CJNE    R7,#00H,BS8
          CLR     EXT_BT.7
          MOV     A,#00H
          SJMP    DBS8
BS8:      SETB    EXT_BT.7
          MOV     A,#10H
DBS8:     LCALL   DI
          MOV     A,#10H
          LCALL   STB
          RET
;************** SUB RING BACK TONE FOR EXTS ***************
RBT_SL1:  MOV     DPTR,#PORTA_3
          MOV     A,#10H
          MOVX    @DPTR,A
          CJNE    R7,#00H,R_S1
          CLR     EXT_RBT.0
          MOV     A,#00H
          SJMP    DRS1
R_S1:     SETB    EXT_RBT.0
          MOV     A,#08H
DRS1:     LCALL   DI
          MOV     A,#08H
          LCALL   STB
          RET
RBT_SL2:  MOV     DPTR,#PORTA_3
          MOV     A,#90H
          MOVX    @DPTR,A
          CJNE    R7,#00H,RS2
          CLR     EXT_RBT.1
          MOV     A,#00H
          SJMP    DRS2
RS2:      SETB    EXT_RBT.1
          MOV     A,#08H
DRS2:     LCALL   DI
          MOV     A,#08H
          LCALL   STB
          RET
RBT_SL3:  MOV     DPTR,#PORTA_3
          MOV     A,#50H
          MOVX    @DPTR,A
          CJNE    R7,#00H,RS3
          CLR     EXT_RBT.2
          MOV     A,#00H
          SJMP    DRS3
RS3:      SETB    EXT_RBT.2
          MOV     A,#08H
DRS3:     LCALL   DI
          MOV     A,#08H
          LCALL   STB
          RET
RBT_SL4:  MOV     DPTR,#PORTA_3
          MOV     A,#0D0H
          MOVX    @DPTR,A
          CJNE    R7,#00H,RS4
          CLR     EXT_RBT.3
          MOV     A,#00H
          SJMP    DRS4
RS4:      SETB    EXT_RBT.3
          MOV     A,#08H
DRS4:     LCALL   DI
```

```
            MOV     A,#08H
            LCALL   STB
            RET
RBT_SL5:    MOV     DPTR,#PORTB_3
            MOV     A,#01H
            MOVX    @DPTR,A
            CJNE    R7,#00H,RS5
            CLR     EXT_RBT.4
            MOV     A,#00H
            SJMP    DRS5
RS5:        SETB    EXT_RBT.4
            MOV     A,#10H
DRS5:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
RBT_SL6:    MOV     DPTR,#PORTB_3
            MOV     A,#09H
            MOVX    @DPTR,A
            CJNE    R7,#00H,RS6
            CLR     EXT_RBT.5
            MOV     A,#00H
            SJMP    DRS6
RS6:        SETB    EXT_RBT.5
            MOV     A,#10H
DRS6:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
RBT_SL7:    MOV     DPTR,#PORTB_3
            MOV     A,#05H
            MOVX    @DPTR,A
            CJNE    R7,#00H,RS7
            CLR     EXT_RBT.6
            MOV     A,#00H
            SJMP    DRS7
RS7:        SETB    EXT_RBT.6
            MOV     A,#10H
DRS7:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
RBT_SL8:    MOV     DPTR,#PORTB_3
            MOV     A,#0DH
            MOVX    @DPTR,A
            CJNE    R7,#00H,RS8
            CLR     EXT_RBT.7
            MOV     A,#00H
            SJMP    DRS8
RS8:        SETB    EXT_RBT.7
            MOV     A,#10H
DRS8:       LCALL   DI
            MOV     A,#10H
            LCALL   STB
            RET
;************** DTMF NO.3 CONNECT TO SLICS ***************
DF3_SL1:    MOV     DPTR,#PORTB_3
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S1
            CLR     DTMF3.0
            MOV     A,#00H
            SJMP    DD3S1
D3S1:       SETB    DTMF3.0
            MOV     A,#20H
DD3S1:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF3_SL2:    MOV     DPTR,#PORTB_3
            MOV     A,#80H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S2
            CLR     DTMF3.1
            MOV     A,#00H
            SJMP    DD3S2
D3S2:       SETB    DTMF3.1
            MOV     A,#20H
DD3S2:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF3_SL3:    MOV     DPTR,#PORTB_3
            MOV     A,#40H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S3
            CLR     DTMF3.2
            MOV     A,#00H
            SJMP    DD3S3
D3S3:       SETB    DTMF3.2
            MOV     A,#20H
DD3S3:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF3_SL4:    MOV     DPTR,#PORTB_3
            MOV     A,#0C0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S4
            CLR     DTMF3.3
            MOV     A,#00H
            SJMP    DD3S4
D3S4:       SETB    DTMF3.3
            MOV     A,#20H
DD3S4:      LCALL   DI
            MOV     A,#20H
```

```
            LCALL   STB
            RET
DF3_SL5:    MOV     DPTR,#PORTC_3
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S5
            CLR     DTMF3.4
            MOV     A,#00H
            SJMP    DD3S5
D3S5:       SETB    DTMF3.4
            MOV     A,#40H
DD3S5:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF3_SL6:    MOV     DPTR,#PORTC_3
            MOV     A,#08H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S6
            CLR     DTMF3.5
            MOV     A,#00H
            SJMP    DD3S6
D3S6:       SETB    DTMF3.5
            MOV     A,#40H
DD3S6:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF3_SL7:    MOV     DPTR,#PORTC_3
            MOV     A,#04H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S7
            CLR     DTMF3.6
            MOV     A,#00H
            SJMP    DD3S7
D3S7:       SETB    DTMF3.6
            MOV     A,#40H
DD3S7:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF3_SL8:    MOV     DPTR,#PORTC_3
            MOV     A,#0CH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D3S8
            CLR     DTMF3.7
            MOV     A,#00H
            SJMP    DD3S8
D3S8:       SETB    DTMF3.7
            MOV     A,#40H
DD3S8:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
;*************** DTMF NO.4 CONNECT TO SLICS ******************
DF4_SL1:    MOV     DPTR,#PORTB_3
            MOV     A,#20H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S1
            CLR     DTMF4.0
            MOV     A,#00H
            SJMP    DD4S1
D4S1:       SETB    DTMF4.0
            MOV     A,#20H
DD4S1:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF4_SL2:    MOV     DPTR,#PORTB_3
            MOV     A,#0A0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S2
            CLR     DTMF4.1
            MOV     A,#00H
            SJMP    DD4S2
D4S2:       SETB    DTMF4.1
            MOV     A,#20H
DD4S2:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF4_SL3:    MOV     DPTR,#PORTB_3
            MOV     A,#60H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S3
            CLR     DTMF4.2
            MOV     A,#00H
            SJMP    DD4S3
D4S3:       SETB    DTMF4.2
            MOV     A,#20H
DD4S3:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF4_SL4:    MOV     DPTR,#PORTB_3
            MOV     A,#0E0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S4
            CLR     DTMF4.3
            MOV     A,#00H
            SJMP    DD4S4
D4S4:       SETB    DTMF4.3
            MOV     A,#20H
DD4S4:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
```

```
            RET
DF4_SL5:    MOV     DPTR,#PORTC_3
            MOV     A,#02H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S5
            CLR     DTMF4.4
            MOV     A,#00H
            SJMP    DD4S5
D4S5:       SETB    DTMF4.4
            MOV     A,#40H
DD4S5:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF4_SL6:    MOV     DPTR,#PORTC_3
            MOV     A,#0AH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S6
            CLR     DTMF4.5
            MOV     A,#00H
            SJMP    DD4S6
D4S6:       SETB    DTMF4.5
            MOV     A,#40H
DD4S6:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF4_SL7:    MOV     DPTR,#PORTC_3
            MOV     A,#06H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S7
            CLR     DTMF4.6
            MOV     A,#00H
            SJMP    DD4S7
D4S7:       SETB    DTMF4.6
            MOV     A,#40H
DD4S7:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF4_SL8:    MOV     DPTR,#PORTC_3
            MOV     A,#0EH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D4S8
            CLR     DTMF4.7
            MOV     A,#00H
            SJMP    DD4S8
D4S8:       SETB    DTMF4.7
            MOV     A,#40H
DD4S8:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
```

```
;************* DTMF NO.5 CONNECT TO SLICS ***************
DF5_SL1:    MOV     DPTR,#PORTB_3
            MOV     A,#10H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S1
            CLR     DTMF5.0
            MOV     A,#00H
            SJMP    DD5S1
D5S1:       SETB    DTMF5.0
            MOV     A,#20H
DD5S1:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF5_SL2:    MOV     DPTR,#PORTB_3
            MOV     A,#90H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S2
            CLR     DTMF5.1
            MOV     A,#00H
            SJMP    DD5S2
D5S2:       SETB    DTMF5.1
            MOV     A,#20H
DD5S2:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF5_SL3:    MOV     DPTR,#PORTB_3
            MOV     A,#50H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S3
            CLR     DTMF5.2
            MOV     A,#00H
            SJMP    DD5S3
D5S3:       SETB    DTMF5.2
            MOV     A,#20H
DD5S3:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF5_SL4:    MOV     DPTR,#PORTB_3
            MOV     A,#0D0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S4
            CLR     DTMF5.3
            MOV     A,#00H
            SJMP    DD5S4
D5S4:       SETB    DTMF5.3
            MOV     A,#20H
DD5S4:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
```

```
DF5_SL5:    MOV     DPTR,#PORTC_3
            MOV     A,#01H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S5
            CLR     DTMF5.4
            MOV     A,#00H
            SJMP    DD5S5
D5S5:       SETB    DTMF5.4
            MOV     A,#40H
DD5S5:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF5_SL6:    MOV     DPTR,#PORTC_3
            MOV     A,#09H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S6
            CLR     DTMF5.5
            MOV     A,#00H
            SJMP    DD5S6
D5S6:       SETB    DTMF5.5
            MOV     A,#40H
DD5S6:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF5_SL7:    MOV     DPTR,#PORTC_3
            MOV     A,#05H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S7
            CLR     DTMF5.6
            MOV     A,#00H
            SJMP    DD5S7
D5S7:       SETB    DTMF5.6
            MOV     A,#40H
DD5S7:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF5_SL8:    MOV     DPTR,#PORTC_3
            MOV     A,#0DH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D5S8
            CLR     DTMF5.7
            MOV     A,#00H
            SJMP    DD5S8
D5S8:       SETB    DTMF5.7
            MOV     A,#40H
DD5S8:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
```

```
;************** DTMF NO.6 CONNECT TO SLICS ***************
DF6_SL1:    MOV     DPTR,#PORTB_3
            MOV     A,#30H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S1
            CLR     DTMF6.0
            MOV     A,#00H
            SJMP    DD6S1
D6S1:       SETB    DTMF6.0
            MOV     A,#20H
DD6S1:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF6_SL2:    MOV     DPTR,#PORTB_3
            MOV     A,#0B0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S2
            CLR     DTMF6.1
            MOV     A,#00H
            SJMP    DD6S2
D6S2:       SETB    DTMF6.1
            MOV     A,#20H
DD6S2:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF6_SL3:    MOV     DPTR,#PORTB_3
            MOV     A,#70H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S3
            CLR     DTMF6.2
            MOV     A,#00H
            SJMP    DD6S3
D6S3:       SETB    DTMF6.2
            MOV     A,#20H
DD6S3:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
DF6_SL4:    MOV     DPTR,#PORTB_3
            MOV     A,#0F0H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S4
            CLR     DTMF6.3
            MOV     A,#00H
            SJMP    DD6S4
D6S4:       SETB    DTMF6.3
            MOV     A,#20H
DD6S4:      LCALL   DI
            MOV     A,#20H
            LCALL   STB
            RET
```

```
DF6_SL5:    MOV     DPTR,#PORTC_3
            MOV     A,#03H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S5
            CLR     DTMF6.4
            MOV     A,#00H
            SJMP    DD6S5
D6S5:       SETB    DTMF6.4
            MOV     A,#40H
DD6S5:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF6_SL6:    MOV     DPTR,#PORTC_3
            MOV     A,#0BH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S6
            CLR     DTMF6.5
            MOV     A,#00H
            SJMP    DD6S6
D6S6:       SETB    DTMF6.5
            MOV     A,#40H
DD6S6:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF6_SL7:    MOV     DPTR,#PORTC_3
            MOV     A,#07H
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S7
            CLR     DTMF6.6
            MOV     A,#00H
            SJMP    DD6S7
D6S7:       SETB    DTMF6.6
            MOV     A,#40H
DD6S7:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
DF6_SL8:    MOV     DPTR,#PORTC_3
            MOV     A,#0FH
            MOVX    @DPTR,A
            CJNE    R7,#00H,D6S8
            CLR     DTMF6.7
            MOV     A,#00H
            SJMP    DD6S8
D6S8:       SETB    DTMF6.7
            MOV     A,#40H
DD6S8:      LCALL   DI
            MOV     A,#40H
            LCALL   STB
            RET
```

```
;************** TONE CONNECT TO TRUNK LINE **************
TK1_RB1:    MOV     DPTR,#PORTA_3
            MOV     A,#00H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TRB1
            CLR     TK_T.2
            MOV     A,#00H
            SJMP    DTRB1
TRB1:       SETB    TK_T.2
            MOV     A,#04H
DTRB1:      LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
TK2_RBT:    MOV     DPTR,#PORTA_3
            MOV     A,#02H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TRB2
            CLR     TK_T.3
            MOV     A,#00H
            SJMP    DTRB2
TRB2:       SETB    TK_T.3
            MOV     A,#04H
DTRB2:      LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
TK1_BT:     MOV     DPTR,#PORTA_3
            MOV     A,#08H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TB1
            CLR     TK_T.0
            MOV     A,#00H
            SJMP    DTB1
TB1:        SETB    TK_T.0
            MOV     A,#04H
DTB1:       LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
TK2_BT:     MOV     DPTR,#PORTA_3
            MOV     A,#0AH
            MOVX    @DPTR,A
            CJNE    R7,#00H,TB2
            CLR     TK_T.1
            MOV     A,#00H
            SJMP    DTB2
TB2:        SETB    TK_T.1
            MOV     A,#04H
DTB2:       LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
```

```
TK1_AS:     MOV     DPTR,#PORTA_3
            MOV     A,#04H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TAS1
            ANL     SOUND,#0FEH
            MOV     A,#00H
            SJMP    DTAS1
TAS1:       ORL     SOUND,#01H
            MOV     A,#04H
DTAS1:      LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
TK2_AS:     MOV     DPTR,#PORTA_3
            MOV     A,#06H
            MOVX    @DPTR,A
            CJNE    R7,#00H,TAS2
            ANL     SOUND,#0FDH
            MOV     A,#00H
            SJMP    DTAS2
TAS2:       ORL     SOUND,#02H
            MOV     A,#04H
DTAS2:      LCALL   DI
            MOV     A,#04H
            LCALL   STB
            RET
;************ DATA IN OF CROSSPOINT ************
DI:         MOV     DPTR,#PORTB_2
            MOVX    @DPTR,A
            RET
;*************** SUB CONNECTING ***************
CTK1_SL1:   MOV     R7,#0FFH
            LCALL   TK1_SL1
            ORL     TKE1,#01H
            MOV     EXT1,#09H
            RET
CTK1_SL2:   MOV     R7,#0FFH
            LCALL   TK1_SL2
            ORL     TKE1,#02H
            MOV     EXT2,#09H
            RET
CTK1_SL3:   MOV     R7,#0FFH
            LCALL   TK1_SL3
            ORL     TKE1,#04H
            MOV     EXT3,#09H
            RET
CTK1_SL4:   MOV     R7,#0FFH
            LCALL   TK1_SL4
            ORL     TKE1,#08H
            MOV     EXT4,#09H
            RET
CTK1_SL5:   MOV     R7,#0FFH
            LCALL   TK1_SL5
            ORL     TKE1,#10H
            MOV     EXT5,#09H
            RET
CTK1_SL6:   MOV     R7,#0FFH
            LCALL   TK1_SL6
            ORL     TKE1,#20H
            MOV     EXT6,#09H
            RET
CTK1_SL7:   MOV     R7,#0FFH
            LCALL   TK1_SL7
            ORL     TKE1,#40H
            MOV     EXT7,#09H
            RET
CTK1_SL8:   MOV     R7,#0FFH
            LCALL   TK1_SL8
            ORL     TKE1,#80H
            MOV     EXT8,#09H
            RET
CTK2_SL1:   MOV     R7,#0FFH
            LCALL   TK2_SL1
            ORL     TKE2,#01H
            MOV     EXT1,#0AH
            RET
CTK2_SL2:   MOV     R7,#0FFH
            LCALL   TK2_SL2
            ORL     TKE2,#02H
            MOV     EXT2,#0AH
            RET
CTK2_SL3:   MOV     R7,#0FFH
            LCALL   TK2_SL3
            ORL     TKE2,#04H
            MOV     EXT3,#0AH
            RET
CTK2_SL4:   MOV     R7,#0FFH
            LCALL   TK2_SL4
            ORL     TKE2,#08H
            MOV     EXT4,#0AH
            RET
CTK2_SL5:   MOV     R7,#0FFH
            LCALL   TK2_SL5
            ORL     TKE2,#10H
            MOV     EXT5,#0AH
            RET
CTK2_SL6:   MOV     R7,#0FFH
            LCALL   TK2_SL6
            ORL     TKE2,#20H
            MOV     EXT6,#0AH
            RET
CTK2_SL7:   MOV     R7,#0FFH
            LCALL   TK2_SL7
            ORL     TKE2,#40H
            MOV     EXT7,#0AH
            RET
```

```
CTK2_SL8:  MOV     R7,#0FFH
           LCALL   TK2_SL8
           ORL     TKE2,#80H
           MOV     EXT8,#0AH
           RET
C1SL1_SL2: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_2
           MOV     EXT1,#02H
           MOV     EXT2,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL3: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_3
           MOV     EXT1,#03H
           MOV     EXT3,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL4: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_4
           MOV     EXT1,#04H
           MOV     EXT4,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL5: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_5
           MOV     EXT1,#05H
           MOV     EXT5,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL6: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_6
           MOV     EXT1,#06H
           MOV     EXT6,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL7: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_7
           MOV     EXT1,#07H
           MOV     EXT7,#01H
           SETB    CROSS_EXT.0
           RET
C1SL1_SL8: MOV     R7,#0FFH
           LCALL   CROSS1_1
           LCALL   CROSS1_8
           MOV     EXT1,#08H
           MOV     EXT8,#01H
           SETB    CROSS_EXT.0
           RET
C2SL1_SL2: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_2
           MOV     EXT1,#02H
           MOV     EXT2,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL3: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_3
           MOV     EXT1,#03H
           MOV     EXT3,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL4: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_4
           MOV     EXT1,#04H
           MOV     EXT4,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL5: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_5
           MOV     EXT1,#05H
           MOV     EXT5,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL6: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_6
           MOV     EXT1,#06H
           MOV     EXT6,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL7: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_7
           MOV     EXT1,#07H
           MOV     EXT7,#01H
           SETB    CROSS_EXT.1
           RET
C2SL1_SL8: MOV     R7,#0FFH
           LCALL   CROSS2_1
           LCALL   CROSS2_8
           MOV     EXT1,#08H
           MOV     EXT8,#01H
           SETB    CROSS_EXT.1
           RET
C3SL1_SL2: MOV     R7,#0FFH
           LCALL   CROSS3_1
           LCALL   CROSS3_2
```

```
            MOV     EXT1,#02H
            MOV     EXT2,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL3:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_3
            MOV     EXT1,#03H
            MOV     EXT3,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL4:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_4
            MOV     EXT1,#04H
            MOV     EXT4,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL5:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_5
            MOV     EXT1,#05H
            MOV     EXT5,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL6:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_6
            MOV     EXT1,#06H
            MOV     EXT6,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL7:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_7
            MOV     EXT1,#07H
            MOV     EXT7,#01H
            SETB    CROSS_EXT.2
            RET
C3SL1_SL8:  MOV     R7,#0FFH
            LCALL   CROSS3_1
            LCALL   CROSS3_8
            MOV     EXT1,#08H
            MOV     EXT8,#01H
            SETB    CROSS_EXT.2
            RET
C1SL2_SL3:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_3
            MOV     EXT2,#03H
            MOV     EXT3,#02H
            SETB    CROSS_EXT.0
            RET
C1SL2_SL4:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_4
            MOV     EXT2,#04H
            MOV     EXT4,#02H
            SETB    CROSS_EXT.0
            RET
C1SL2_SL5:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_5
            MOV     EXT2,#05H
            MOV     EXT5,#02H
            SETB    CROSS_EXT.0
            RET
C1SL2_SL6:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_6
            MOV     EXT2,#06H
            MOV     EXT6,#02H
            SETB    CROSS_EXT.0
            RET
C1SL2_SL7:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_7
            MOV     EXT2,#07H
            MOV     EXT7,#02H
            SETB    CROSS_EXT.0
            RET
C1SL2_SL8:  MOV     R7,#0FFH
            LCALL   CROSS1_2
            LCALL   CROSS1_8
            MOV     EXT2,#08H
            MOV     EXT8,#02H
            SETB    CROSS_EXT.0
            RET
C2SL2_SL3:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_3
            MOV     EXT2,#03H
            MOV     EXT3,#02H
            SETB    CROSS_EXT.1
            RET
C2SL2_SL4:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_4
            MOV     EXT2,#04H
            MOV     EXT4,#02H
            SETB    CROSS_EXT.1
            RET
C2SL2_SL5:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_5
            MOV     EXT2,#05H
```

```
            MOV     EXT5,#02H
            SETB    CROSS_EXT.1
            RET
C2SL2_SL6:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_6
            MOV     EXT2,#06H
            MOV     EXT6,#02H
            SETB    CROSS_EXT.1
            RET
C2SL2_SL7:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_7
            MOV     EXT2,#07H
            MOV     EXT7,#02H
            SETB    CROSS_EXT.1
            RET
C2SL2_SL8:  MOV     R7,#0FFH
            LCALL   CROSS2_2
            LCALL   CROSS2_8
            MOV     EXT2,#08H
            MOV     EXT8,#02H
            SETB    CROSS_EXT.1
            RET
C3SL2_SL3:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_3
            MOV     EXT2,#03H
            MOV     EXT3,#02H
            SETB    CROSS_EXT.2
            RET
C3SL2_SL4:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_4
            MOV     EXT2,#04H
            MOV     EXT4,#02H
            SETB    CROSS_EXT.2
            RET
C3SL2_SL5:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_5
            MOV     EXT2,#05H
            MOV     EXT5,#02H
            SETB    CROSS_EXT.2
            RET
C3SL2_SL6:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_6
            MOV     EXT2,#06H
            MOV     EXT6,#02H
            SETB    CROSS_EXT.2
            RET
C3SL2_SL7:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_7
            MOV     EXT2,#07H
            MOV     EXT7,#02H
            SETB    CROSS_EXT.2
            RET
C3SL2_SL8:  MOV     R7,#0FFH
            LCALL   CROSS3_2
            LCALL   CROSS3_8
            MOV     EXT2,#08H
            MOV     EXT8,#02H
            SETB    CROSS_EXT.2
            RET
C1SL3_SL4:  MOV     R7,#0FFH
            LCALL   CROSS1_3
            LCALL   CROSS1_4
            MOV     EXT3,#04H
            MOV     EXT4,#03H
            SETB    CROSS_EXT.0
            RET
C1SL3_SL5:  MOV     R7,#0FFH
            LCALL   CROSS1_3
            LCALL   CROSS1_5
            MOV     EXT3,#05H
            MOV     EXT5,#03H
            SETB    CROSS_EXT.0
            RET
C1SL3_SL6:  MOV     R7,#0FFH
            LCALL   CROSS1_3
            LCALL   CROSS1_6
            MOV     EXT3,#06H
            MOV     EXT6,#03H
            SETB    CROSS_EXT.0
            RET
C1SL3_SL7:  MOV     R7,#0FFH
            LCALL   CROSS1_3
            LCALL   CROSS1_7
            MOV     EXT3,#07H
            MOV     EXT7,#03H
            SETB    CROSS_EXT.0
            RET
C1SL3_SL8:  MOV     R7,#0FFH
            LCALL   CROSS1_3
            LCALL   CROSS1_8
            MOV     EXT3,#08H
            MOV     EXT8,#03H
            SETB    CROSS_EXT.0
            RET
C2SL3_SL4:  MOV     R7,#0FFH
            LCALL   CROSS2_3
            LCALL   CROSS2_4
            MOV     EXT3,#04H
            MOV     EXT4,#03H
```

```
            SETB    CROSS_EXT.1
            RET
C2SL3_SL5:  MOV     R7,#0FFH
            LCALL   CROSS2_3
            LCALL   CROSS2_5
            MOV     EXT3,#05H
            MOV     EXT5,#03H
            SETB    CROSS_EXT.1
            RET
C2SL3_SL6:  MOV     R7,#0FFH
            LCALL   CROSS2_3
            LCALL   CROSS2_6
            MOV     EXT3,#06H
            MOV     EXT6,#03H
            SETB    CROSS_EXT.1
            RET
C2SL3_SL7:  MOV     R7,#0FFH
            LCALL   CROSS2_3
            LCALL   CROSS2_7
            MOV     EXT3,#07H
            MOV     EXT7,#03H
            SETB    CROSS_EXT.1
            RET
C2SL3_SL8:  MOV     R7,#0FFH
            LCALL   CROSS2_3
            LCALL   CROSS2_8
            MOV     EXT3,#08H
            MOV     EXT8,#03H
            SETB    CROSS_EXT.1
            RET
C3SL3_SL4:  MOV     R7,#0FFH
            LCALL   CROSS3_3
            LCALL   CROSS3_4
            MOV     EXT3,#04H
            MOV     EXT4,#03H
            SETB    CROSS_EXT.2
            RET
C3SL3_SL5:  MOV     R7,#0FFH
            LCALL   CROSS3_3
            LCALL   CROSS3_5
            MOV     EXT3,#05H
            MOV     EXT5,#03H
            SETB    CROSS_EXT.2
            RET
C3SL3_SL6:  MOV     R7,#0FFH
            LCALL   CROSS3_3
            LCALL   CROSS3_6
            MOV     EXT3,#06H
            MOV     EXT6,#03H
            SETB    CROSS_EXT.2
            RET
C3SL3_SL7:  MOV     R7,#0FFH
            LCALL   CROSS3_3
            LCALL   CROSS3_7
            MOV     EXT3,#07H
            MOV     EXT7,#03H
            SETB    CROSS_EXT.2
            RET
C3SL3_SL8:  MOV     R7,#0FFH
            LCALL   CROSS3_3
            LCALL   CROSS3_8
            MOV     EXT3,#08H
            MOV     EXT8,#03H
            SETB    CROSS_EXT.2
            RET
C1SL4_SL5:  MOV     R7,#0FFH
            LCALL   CROSS1_4
            LCALL   CROSS1_5
            MOV     EXT4,#05H
            MOV     EXT5,#04H
            SETB    CROSS_EXT.0
            RET
C1SL4_SL6:  MOV     R7,#0FFH
            LCALL   CROSS1_4
            LCALL   CROSS1_6
            MOV     EXT4,#06H
            MOV     EXT6,#04H
            SETB    CROSS_EXT.0
            RET
C1SL4_SL7:  MOV     R7,#0FFH
            LCALL   CROSS1_4
            LCALL   CROSS1_7
            MOV     EXT4,#07H
            MOV     EXT7,#04H
            SETB    CROSS_EXT.0
            RET
C1SL4_SL8:  MOV     R7,#0FFH
            LCALL   CROSS1_4
            LCALL   CROSS1_8
            MOV     EXT4,#08H
            MOV     EXT8,#04H
            SETB    CROSS_EXT.0
            RET
C2SL4_SL5:  MOV     R7,#0FFH
            LCALL   CROSS2_4
            LCALL   CROSS2_5
            MOV     EXT4,#05H
            MOV     EXT5,#04H
            SETB    CROSS_EXT.1
            RET
C2SL4_SL6:  MOV     R7,#0FFH
            LCALL   CROSS2_4
            LCALL   CROSS2_6
            MOV     EXT4,#06H
            MOV     EXT6,#04H
            SETB    CROSS_EXT.1
```

```
            RET
C2SL4_SL7:  MOV     R7,#0FFH
            LCALL   CROSS2_4
            LCALL   CROSS2_7
            MOV     EXT4,#07H
            MOV     EXT7,#04H
            SETB    CROSS_EXT.1
            RET
C2SL4_SL8:  MOV     R7,#0FFH
            LCALL   CROSS2_4
            LCALL   CROSS2_8
            MOV     EXT4,#08H
            MOV     EXT8,#04H
            SETB    CROSS_EXT.1
            RET
C3SL4_SL5:  MOV     R7,#0FFH
            LCALL   CROSS3_4
            LCALL   CROSS3_5
            MOV     EXT4,#05H
            MOV     EXT5,#04H
            SETB    CROSS_EXT.2
            RET
C3SL4_SL6:  MOV     R7,#0FFH
            LCALL   CROSS3_4
            LCALL   CROSS3_6
            MOV     EXT4,#06H
            MOV     EXT6,#04H
            SETB    CROSS_EXT.2
            RET
C3SL4_SL7:  MOV     R7,#0FFH
            LCALL   CROSS3_4
            LCALL   CROSS3_7
            MOV     EXT4,#07H
            MOV     EXT7,#04H
            SETB    CROSS_EXT.2
            RET
C3SL4_SL8:  MOV     R7,#0FFH
            LCALL   CROSS3_4
            LCALL   CROSS3_8
            MOV     EXT4,#08H
            MOV     EXT8,#04H
            SETB    CROSS_EXT.2
            RET
C1SL5_SL6:  MOV     R7,#0FFH
            LCALL   CROSS1_5
            LCALL   CROSS1_6
            MOV     EXT5,#06H
            MOV     EXT6,#05H
            SETB    CROSS_EXT.0
            RET
C1SL5_SL7:  MOV     R7,#0FFH
            LCALL   CROSS1_5
            LCALL   CROSS1_7
            MOV     EXT5,#07H
            MOV     EXT7,#05H
            SETB    CROSS_EXT.0
            RET
C1SL5_SL8:  MOV     R7,#0FFH
            LCALL   CROSS1_5
            LCALL   CROSS1_8
            MOV     EXT5,#08H
            MOV     EXT8,#05H
            SETB    CROSS_EXT.0
            RET
C2SL5_SL6:  MOV     R7,#0FFH
            LCALL   CROSS2_5
            LCALL   CROSS2_6
            MOV     EXT5,#06H
            MOV     EXT6,#05H
            SETB    CROSS_EXT.1
            RET
C2SL5_SL7:  MOV     R7,#0FFH
            LCALL   CROSS2_5
            LCALL   CROSS2_7
            MOV     EXT5,#07H
            MOV     EXT7,#05H
            SETB    CROSS_EXT.1
            RET
C2SL5_SL8:  MOV     R7,#0FFH
            LCALL   CROSS2_5
            LCALL   CROSS2_8
            MOV     EXT5,#08H
            MOV     EXT8,#05H
            SETB    CROSS_EXT.1
            RET
C3SL5_SL6:  MOV     R7,#0FFH
            LCALL   CROSS3_5
            LCALL   CROSS3_6
            MOV     EXT5,#06H
            MOV     EXT6,#05H
            SETB    CROSS_EXT.2
            RET
C3SL5_SL7:  MOV     R7,#0FFH
            LCALL   CROSS3_5
            LCALL   CROSS3_7
            MOV     EXT5,#07H
            MOV     EXT7,#05H
            SETB    CROSS_EXT.2
            RET
C3SL5_SL8:  MOV     R7,#0FFH
            LCALL   CROSS3_5
            LCALL   CROSS3_8
            MOV     EXT5,#08H
            MOV     EXT8,#05H
            SETB    CROSS_EXT.2
            RET
```

```
C1SL6_SL7: MOV     R7,#0FFH
           LCALL   CROSS1_6
           LCALL   CROSS1_7
           MOV     EXT6,#07H
           MOV     EXT7,#06H
           SETB    CROSS_EXT.0
           RET
C1SL6_SL8: MOV     R7,#0FFH
           LCALL   CROSS1_6
           LCALL   CROSS1_8
           MOV     EXT6,#08H
           MOV     EXT8,#06H
           SETB    CROSS_EXT.0
           RET
C2SL6_SL7: MOV     R7,#0FFH
           LCALL   CROSS2_6
           LCALL   CROSS2_7
           MOV     EXT6,#07H
           MOV     EXT7,#06H
           SETB    CROSS_EXT.1
           RET
C2SL6_SL8: MOV     R7,#0FFH
           LCALL   CROSS2_6
           LCALL   CROSS2_8
           MOV     EXT6,#08H
           MOV     EXT8,#06H
           SETB    CROSS_EXT.1
           RET
C3SL6_SL7: MOV     R7,#0FFH
           LCALL   CROSS3_6
           LCALL   CROSS3_7
           MOV     EXT6,#07H
           MOV     EXT7,#06H
           SETB    CROSS_EXT.2
           RET
C3SL6_SL8: MOV     R7,#0FFH
           LCALL   CROSS3_6
           LCALL   CROSS3_8
           MOV     EXT6,#08H
           MOV     EXT8,#06H
           SETB    CROSS_EXT.2
           RET
C1SL7_SL8: MOV     R7,#0FFH
           LCALL   CROSS1_7
           LCALL   CROSS1_8
           MOV     EXT7,#08H
           MOV     EXT8,#07H
           SETB    CROSS_EXT.0
           RET
C2SL7_SL8: MOV     R7,#0FFH
           LCALL   CROSS2_7
           LCALL   CROSS2_8
           MOV     EXT7,#0βH
           MOV     EXT8,#07H
           SETB    CROSS_EXT.1
           RET
C3SL7_SL8: MOV     R7,#0FFH
           LCALL   CROSS3_7
           LCALL   CROSS3_8
           MOV     EXT7,#08H
           MOV     EXT8,#07H
           SETB    CROSS_EXT.2
           RET
;******* SUB CHECK LINES THAT DON'T HAVE DTMF DECODER. *******
CHK_NLDF:  MOV     A,WAIT_DF
           CJNE    A,#00H,$+4
           RET
           JNB     WAIT_DF.0,C_NS2DF
           MOV     A,DTMF3
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF3_SL1
           CLR     WAIT_DF.0
           RET
           MOV     A,DTMF4
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF4_SL1
           CLR     WAIT_DF.0
           RET
           MOV     A,DTMF5
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF5_SL1
           CLR     WAIT_DF.0
           RET
           MOV     A,DTMF6
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF6_SL1
           CLR     WAIT_DF.0
           RET
C_NS2DF:   JNB     WAIT_DF.1,C_NS3DF
           MOV     A,DTMF3
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF3_SL2
           CLR     WAIT_DF.1
           RET
           MOV     A,DTMF4
           CJNE    A,#00H,$+11
           MOV     R7,#0FFH
           LCALL   DF4_SL2
           CLR     WAIT_DF.1
           RET
           MOV     A,DTMF5
```

```
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF5_SL2
        CLR     WAIT_DF.1
        RET
        MOV     A,DTMF6
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF6_SL2
        CLR     WAIT_DF.1
        RET
C_NS3DF: JNB    WAIT_DF.2,C_NS4DF
        MOV     A,DTMF3
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF3_SL3
        CLR     WAIT_DF.2
        RET
        MOV     A,DTMF4
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF4_SL3
        CLR     WAIT_DF.2
        RET
        MOV     A,DTMF5
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF5_SL3
        CLR     WAIT_DF.2
        RET
        MOV     A,DTMF6
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF6_SL3
        CLR     WAIT_DF.2
        RET
C_NS4DF: JNB    WAIT_DF.3,C_NS5DF
        MOV     A,DTMF3
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF3_SL4
        CLR     WAIT_DF.3
        RET
        MOV     A,DTMF4
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF4_SL4
        CLR     WAIT_DF.3
        RET
        MOV     A,DTMF5
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF5_SL4
        CLR     WAIT_DF.3
        RET
        MOV     A,DTMF6
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF6_SL4
        CLR     WAIT_DF.3
        RET
C_NS5DF: JNB    WAIT_DF.4,C_NS6DF
        MOV     A,DTMF3
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF3_SL5
        CLR     WAIT_DF.4
        RET
        MOV     A,DTMF4
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF4_SL5
        CLR     WAIT_DF.4
        RET
        MOV     A,DTMF5
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF5_SL5
        CLR     WAIT_DF.4
        RET
        MOV     A,DTMF6
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF6_SL5
        CLR     WAIT_DF.4
        RET
C_NS6DF: JNB    WAIT_DF.5,C_NS7DF
        MOV     A,DTMF3
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF3_SL6
        CLR     WAIT_DF.5
        RET
        MOV     A,DTMF4
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF4_SL6
        CLR     WAIT_DF.5
        RET
        MOV     A,DTMF5
        CJNE    A,#00H,$+11
        MOV     R7,#0FFH
        LCALL   DF5_SL6
        CLR     WAIT_DF:5
        RET
        MOV     A,DTMF6
```

```
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF6_SL6
            CLR     WAIT_DF.5
            RET
C_NS7DF:    JNB     WAIT_DF.6,C_NS8DF
            MOV     A,DTMF3
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF3_SL7
            CLR     WAIT_DF.6
            RET
            MOV     A,DTMF4
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF4_SL7
            CLR     WAIT_DF.6
            RET
            MOV     A,DTMF5
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF5_SL7
            CLR     WAIT_DF.6
            RET
            MOV     A,DTMF6
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF6_SL7
            CLR     WAIT_DF.6
            RET
C_NS8DF:    JNB     WAIT_DF.7,$-1
            MOV     A,DTMF3
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF3_SL8
            CLR     WAIT_DF.7
            RET
            MOV     A,DTMF4
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF4_SL8
            CLR     WAIT_DF.7
            RET
            MOV     A,DTMF5
            CJNE    A,#00H,$+11
            MOV     R7,#0FFH
            LCALL   DF5_SL8
            CLR     WAIT_DF.7
            RET
            MOV     A,DTMF6
            CJNE    A,#00H,$+10
            MOV     R7,#0FFH
            LCALL   DF6_SL8
            CLR     WAIT_DF.7
            RET
;********* SEND EXT.NO. FOR COUNT RINGING **********
C_LINK:     MOV     DPTR,#PORTA_5
            MOV     A,#01H
            MOVX    @DPTR,A
            MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,#10H,$-4
            MOV     DPTR,#PORTA_5
            MOV     A,#00H
            MOVX    @DPTR,A
            LCALL   DELAY90
            RET
;*** STORE NUMS.AND SEND DATA TO 2'nd CONTROL UNIT ***
STR_NUM1:   MOV     A,R5
            CJNE    A,#0BH,$+16
            MOV     A,FNC4
            ANL     A,#01H
            CJNE    A,#00H,$+6
            ORL     FNC4,#01H
            LJMP    INT_1
            CJNE    A,#0AH,$+7
            MOV     A,#00H
            SJMP    REC_N1
            CJNE    A,#0CH,REC_N1
            MOV     R1,PP1
            MOV     @R1,A
            MOV     R5,#0F0H
            LJMP    S_LINK
REC_N1:     MOV     R1,PP1
            MOV     @R1,A
            INC     R1
            MOV     PP1,R1
            DJNZ    SEQ_N1,$+10
            MOV     @R1,#0CH
            MOV     R5,#0F0H
            LJMP    S_LINK
            LJMP    INT_1
S_LINK:     CJNE    R5,#0F0H,$-3
            MOV     R6,LROOM1
            MOV     PP1,#AD_LN1
            SJMP    S_ST
            LJMP    INT_1
S_ST:       MOV     DPTR,#PORTA_5
            MOV     A,#03H
            MOVX    @DPTR,A
            MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,#30H,$-4
            MOV     DPTR,#PORTB_5
            MOV     A,#01H
            MOVX    @DPTR,A
```

```
            SWAP    A
            MOV     KP_DA,A
            MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,KP_DA,$-4

            MOV     PP2,#00H
            MOV     R2,#0CH
            MOV     R1,PP1
SEND_N:     DJNZ    R2,$+4
            SJMP    END_SE
            MOV     DPTR,#PORTB_5
            MOV     A,@R1
            INC     R1
            INC     PP2
            MOVX    @DPTR,A
WAIT_ACK:   MOV     DPTR,#PORTC_5
            MOVX    A,@DPTR
            CJNE    A,PP2,$+5
            SJMP    SEND_N
            CJNE    A,#0F0H,WAIT_ACK
            SJMP    END_SE
END_SE:     MOV     DPTR,#PORTA_5
            MOV     A,#00H
            MOVX    @DPTR,A
            MOV     DPTR,#PORTB_5
            MOVX    @DPTR,A
            CJNE    R5,#0F0H,LLIN
            ANL     FNC4,#0FEH
            MOV     SEQ_N2,#0BH
            MOV     PP2,#AD_LN2
            MOV     RA1,#00H
            ORL     TK_DF,#01H
            MOV     R6,#10111000B
            LCALL   AUTO_ANS1
            LCALL   DELAY1
            LCALL   DELAY90
            LCALL   AUT1
            MOV     R7,#00H
            LCALL   TK1_HK
LLIN:       LJMP    INT_1
;******** RECEIVE DATA FROM 2'nd CONTROL UNIT ********
R_LINK:     MOV     R5,A
            SWAP    A
            MOV     DPTR,#PORTA_5
            MOVX    @DPTR,A
            LCALL   DELAY90
            LCALL   DELAY90
            MOV     A,#00H
            MOVX    @DPTR,A
            MOV     A,R5
            CJNE    A,#01H,CHK_R2 ;FIND TK THAT HAVE RINGED
            MOV     R6,#01H
            MOV     R5,TKE1
            MOV     C_TK,#05H
            SJMP    CHK_TRE
            LJMP    MAIN_SYS
CHK_R2:     LJMP    MAIN_SYS
CHK_TRE:    MOV     A,R_EXT1
            CJNE    A,C_TK,$-5
            MOV     A,R5
            ANL     A,#01H
            CJNE    A,#01H,$+6      ;IF EXT.1 HK ON BEFORE COUNT
            LJMP    MAIN_SYS        ;10 sec.WHEN COUNT COMPLETE
            LCALL   OFF_R1          ;WILL JUMP TO MAIN SYSTEM...
            MOV     R4,#01H
            LJMP    AUTO_LM
AUTO_LM:    MOV     R7,#00H
            LCALL   TK1_RBT
            MOV     R6,#11001000B
            LCALL   AUTO_ANS1
            LCALL   DELAY4
            LCALL   DELAY3
            LCALL   DELAY90
            LCALL   AUT1
            ANL     TK_DF,#0FEH
            MOV     RA1,#0FFH
            MOV     LROOM1,#01H
            LJMP    MAIN_SYS
AUTO2:      LJMP    MAIN_SYS
;************ SUB AUTO ANSWER *************
AUTO_ANS1:  MOV     R7,#0FFH
            LCALL   TK1_AS
            MOV     DPTR,#PORTC_4
            MOV     A,R6
            LCALL   DELAY500
            MOVX    @DPTR,A
            CLR     ACC.7
            MOVX    @DPTR,A
            RET
AUT1:       MOV     A,#80H
            MOVX    @DPTR,A
            MOV     R7,#00H         ;** OPEN SW AND STOP SOUND
            LCALL   TK1_AS
            RET
AUTO_ANS2:  MOV     R7,#0FFH
            LCALL   TK2_AS
            MOV     DPTR,#PORTC_4
            MOV     A,R6
            LCALL   DELAY500
            MOVX    @DPTR,A
            CLR     ACC.7
            MOVX    @DPTR,A
            RET
AUT2:       MOV     A,#80H
            MOVX    @DPTR,A
```

```
            MOV     R7,#00H         ;** OPEN SW AND STOP SOUND
            LCALL   TK2_AS
            RET
RESET_S:    MOV     DPTR,#PORTC_4
            MOV     A,#80H
            MOVX    @DPTR,A
            RET
;*************** SUB SEND BUSY TONE ******************
ONBT_SL1:   MOV     R7,#0FFH
            LCALL   BT_SL1
            RET
ONBT_SL2:   MOV     R7,#0FFH
            LCALL   BT_SL2
            RET
ONBT_SL3:   MOV     R7,#0FFH
            LCALL   BT_SL3
            RET
ONBT_SL4:   MOV     R7,#0FFH
            LCALL   BT_SL4
            RET
ONBT_SL5:   MOV     R7,#0FFH
            LCALL   BT_SL5
            RET
ONBT_SL6:   MOV     R7,#0FFH
            LCALL   BT_SL6
            RET
ONBT_SL7:   MOV     R7,#0FFH
            LCALL   BT_SL7
            RET
ONBT_SL8:   MOV     R7,#0FFH
            LCALL   BT_SL8
            RET
;********** SUB OFF DTMF FOR SLIC **********
OFF_DFS1:   MOV     R7,#00H
            LCALL   DF3_SL1
            LCALL   DF4_SL1
            LCALL   DF5_SL1
            LCALL   DF6_SL1
            LCALL   CHK_NLDF
            RET
OFF_DFS2:   MOV     R7,#00H
            LCALL   DF3_SL2
            LCALL   DF4_SL2
            LCALL   DF5_SL2
            LCALL   DF6_SL2
            LCALL   CHK_NLDF
            RET
OFF_DFS3:   MOV     R7,#00H
            LCALL   DF3_SL3
            LCALL   DF4_SL3
            LCALL   DF5_SL3
            LCALL   DF6_SL3
            LCALL   CHK_NLDF
            RET
OFF_DFS4:   MOV     R7,#00H
            LCALL   DF3_SL4
            LCALL   DF4_SL4
            LCALL   DF5_SL4
            LCALL   DF6_SL4
            LCALL   CHK_NLDF
            RET
OFF_DFS5:   MOV     R7,#00H
            LCALL   DF3_SL5
            LCALL   DF4_SL5
            LCALL   DF5_SL5
            LCALL   DF6_SL5
            LCALL   CHK_NLDF
            RET
OFF_DFS6:   MOV     R7,#00H
            LCALL   DF3_SL6
            LCALL   DF4_SL6
            LCALL   DF5_SL6
            LCALL   DF6_SL6
            LCALL   CHK_NLDF
            RET
OFF_DFS7:   MOV     R7,#00H
            LCALL   DF3_SL7
            LCALL   DF4_SL7
            LCALL   DF5_SL7
            LCALL   DF6_SL7
            LCALL   CHK_NLDF
            RET
OFF_DFS8:   MOV     R7,#00H
            LCALL   DF3_SL8
            LCALL   DF4_SL8
            LCALL   DF5_SL8
            LCALL   DF6_SL8
            LCALL   CHK_NLDF
            RET
;*************** DATA_IN ***********************
DATA_IN:    MOV     DPTR,#PORTB_2
            MOV     A,#00H
            MOVX    @DPTR,A
            LCALL   DELAY500
            LCALL   DELAY500
            RET
;*************** STROBE ************************
STB:        MOV     DPTR,#PORTA_2
            MOVX    @DPTR,A
            LCALL   DELAY500
            LCALL   DELAY500
            MOV     A,#00H
            MOVX    @DPTR,A
            RET
;**********DELAY (4 SEC)***********
DELAY4:     MOV     R0,#20H
```

```
DE1:        MOV     R1,#00H
DE2.        MOV     R2,#00H
            DJNZ    R2,$
            DJNZ    R1,DE2
            DJNZ    R0,DE1
            RET
;**********DELAY (3 SEC)***********
DELAY3:     MOV     R0,#17H
D31:        MOV     R1,#00H
D32:        MOV     R2,#00H
            DJNZ    R2,$
            DJNZ    R1,D32
            DJNZ    R0,D31
            RET
;**********DELAY (2 SEC)***********
DELAY2:     MOV     R0,#10H
D21:        MOV     R1,#00H
D22:        MOV     R2,#00H
            DJNZ    R2,$
            DJNZ    R1,D22
            DJNZ    R0,D21
            RET
;**********DELAY (1 SEC)***********
DELAY1:     MOV     R0,#08H
D11:        MOV     R1,#00H
D12:        MOV     R2,#00H
            DJNZ    R2,$
            DJNZ    R1,D12
            DJNZ    R0,D11
            RET
;**********DELAY (0.5SEC)***********
DELAY05:    MOV     R0,#04H
D051:       MOV     R1,#00H
D052:       MOV     R2,#00H
            DJNZ    R2,$
            DJNZ    R1,D052
            DJNZ    R0,D051
            RET
;******** DELAY (90.8 mSEC)***********
DELAY90:    MOV     R0,#0B1H
D9:         MOV     R1,#0FFH
            DJNZ    R1,$
            DJNZ    R0,D9
            RET
;********DELAY (5.15 msec)**********
DELAY5M:    MOV     R0,#0AH
D5M:        MOV     R1,#00H
            DJNZ    R1,$
            DJNZ    R0,D5M
            RET
;********DELAY (518 uSEC)**********
DELAY500:   MOV     R0,#01H
D500:       MOV     R1,#00H
            DJNZ    R1,$
            DJNZ    R0,D500
            RET
END
```

;****************** FINISH **********************

```
;FILENAME      KEYBOARD.ASM
;DESCRIPTION   MK48T08 RTC DEMO PROGRAM
; ASSEMBLER    SXA51
; ENGINEER     TAWEE  KUSUMAWALEE
; THIS SOFTWARE SUPPORT  PABX FUNCTION TO
; DISPLAY  TELEPHONE  NUMBER, TIME ,YEAR
;***********VARIABLE**************
PORTA_1       EQU     0F800H  ;PORTA 8255_1
PORTB_1       EQU     0F801H  ;PORTB 8255_1
PORTC_1       EQU     0F802H  ;PORTC 8255_1
CONPORT_1     EQU     0F803H  ;CONTROL PORT 8255_1
;*********************************
PORTA_2       EQU     0FC00H  ;PORTA 8255_2
PORTB_2       EQU     0FC01H  ;PORTB 8255_2
PORTC_2       EQU     0FC02H  ;PORTC 8255_2
CONPORT_2     EQU     0FC03H  ;CONTROL PORT 8255_2
;*********************************
;SET_SEC   EQU    12H
;SET_MIN   EQU    11H
;SET_HOUR  EQU    10H
;SET_DAY   EQU    13H
;SET_MONTH EQU    14H
;SET_YEAR  EQU    15H
SEC           EQU    62H
MIN           EQU    61H
HOUR          EQU    60H
DAY           EQU    63H
MONTH         EQU    64H
YEAR          EQU    65H
TIM_HOUR      EQU    70H
TIM_MIN       EQU    71H
TIM_SEC       EQU    72H
DATE_D        EQU    73H
DATE_M        EQU    74H
DATE_Y        EQU    75H
BUF_TIM       EQU    40H
BUF_DATE      EQU    30H
BUF_WEEK      EQU    50H
TIMBUF        EQU    1FF8H  ;RTC BUFFER
CHARBUF       EQU    1FF9H  ;CHAR BUFFER
COM_PORT      EQU    0FA00H ;LCD WRITE CONTROL (ANT-31PJ)
RD_BUSY       EQU    0FA01H ;LCD READ CONTROL
DAT_PORT      EQU    0FA02H ;LCD WRITE DATA
RD_DATA       EQU    0FA03H ;LCD READ DATA
BUFFER        EQU    76H ;8 BIT
BUFFER1       EQU    77H ;8 BIT
BUFFER2       EQU    78H ;8 BIT
POINTER       EQU    79H ;8 BIT
POINTER1      EQU    7AH ;8 BIT
POINTER2      EQU    7BH ;8 BIT
POINTER3      EQU    7CH ;8 BIT
POINTER4      EQU    7DH ;8 BIT
POINTER5      EQU    7EH ;8 BIT
HAP1          EQU    2DH ;8 BIT
HAP2          EQU    2EH ;8 BIT
HAP3          EQU    23H ;8 BIT
HAP4          EQU    24H ;8 BIT
HAP5          EQU    25H ;8 BIT
HAP6          EQU    26H ;8 BIT
HAP7          EQU    27H ;8 BIT
HAP8          EQU    28H ;8 BIT
HAP9          EQU    29H ;8 BIT
HAP10         EQU    2AH ;8 BIT
HAP11         EQU    2BH ;8 BIT
HAP12         EQU    2CH ;8 BIT
;**********************RESERVE  ADDRESS**********************
ROOM1         EQU    0100H
ROOM2         EQU    016EH
ROOM3         EQU    01DCH
ROOM4         EQU    024AH
ROOM5         EQU    02B8H
ROOM6         EQU    0326H
ROOM7         EQU    0394H
ROOM8         EQU    0402H
CALENDAR1     EQU    0410H
CALENDAR2     EQU    047EH
CALENDAR3     EQU    04ECH
CALENDAR4     EQU    055AH
CALENDAR5     EQU    05C8H
CALENDAR6     EQU    0636H
CALENDAR7     EQU    06A4H
CALENDAR8     EQU    0712H
RECTIME1      EQU    0780H
RECTIME2      EQU    0820H
RECTIME3      EQU    08C0H
RECTIME4      EQU    0960H
RECTIME5      EQU    0A00H
RECTIME6      EQU    0AA0H
RECTIME7      EQU    0B40H
RECTIME8      EQU    0BE0H
R_LV          EQU      20H
RR6           EQU      21H
RR7           EQU      22H
;*********** INITIAL PORT *****************
              ORG    0000H
              LJMP   MAIN_SYS
;**************** MAIN PROGRAM ******************
              ORG    0100H
MAIN_SYS:     MOV    P1,#00H
              MOV    DPTR,#017FH
```

```
        MOV    A,#18H
        MOVX   @DPTR,A
        MOV    IE,#00H
        MOV    R2,#02H
        LCALL  DELAY
        MOV    DPTR,#CONPORT_1     ;PA=IN PB=IN PC=OUT
        MOV    A,#92H
        MOVX   @DPTR,A
        MOV    DPTR,#CONPORT_2     ;PA=IN PB=OUT PC=OUT
        MOV    A,#90H
        MOVX   @DPTR,A
        SJMP   RES
; ********** RESET **********
; TEST LCD PORT
RES:    MOV    SP,#80H
        MOV    R2,#01H
        LCALL  DELAY
        MOV    A,#00111000B        ;FUNCTION SET
        LCALL  WR_COMM
        MOV    A,#00001100B        ;DISPLAY ON/OFF
        LCALL  WR_COMM
        MOV    A,#00H
        MOV    DPTR,#PORTC_1
        MOVX   @DPTR,A             ;SEND 00H OUT OF PORTC_1
        MOV    A,#01H              ;CLEAR
        LCALL  WR_COMM
START:  MOV    DPTR,#MAINT1
        LCALL  DISPLAY             ;WRITE LCD MAINT1
        MOV    R2,#01H
        LCALL  DELAY
CONFIG: MOV    POINTER3,#0FFH
MAIN1:  LCALL  CHECK
MAIN11: LCALL  RTIME          ;DISPLAY HH.MM.SS
        LCALL  DECODE
        MOV    R1,#BUF_TIM
        LCALL  RUN_8R1
MAIN2:  LCALL  RDATE          ;DISPLAY DD.MM.YY
        LCALL  DECODE2
        MOV    R1,#BUF_DATE
        LCALL  RUN_8R2
        MOV    A,R5
        ANL    A,#07H
        MOV    R0,A
        CJNE   R0,#01H,A2
        LJMP   DDAY1
A2:     CJNE   R0,#02H,A3
        LJMP   DDAY2
A3:     CJNE   R0,#03H,A4
        LJMP   DDAY3
A4:     CJNE   R0,#04H,A5
        LJMP   DDAY4
A5:     CJNE   R0,#05H,A6
        LJMP   DDAY5
A6:     CJNE   R0,#06H,A7
        LJMP   DDAY6
A7:     CJNE   R0,#07H,MAIN1X
        LJMP   DDAY7
MAIN1X: LJMP   CHK_CO
DDAY1:  MOV    BUF_WEEK,#'S'
        MOV    BUF_WEEK+1,#'U'
        MOV    BUF_WEEK+2,#'N'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY
        LJMP   CHK_CO
DDAY2:  MOV    BUF_WEEK,#'M'
        MOV    BUF_WEEK+1,#'O'
        MOV    BUF_WEEK+2,#'N'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY
        LJMP   CHK_CO
DDAY3:  MOV    BUF_WEEK,#'T'
        MOV    BUF_WEEK+1,#'U'
        MOV    BUF_WEEK+2,#'E'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY            ;THIS LINE WRITE MESSAGE ON LCD
        LJMP   CHK_CO
DDAY4:  MOV    BUF_WEEK,#'W'
        MOV    BUF_WEEK+1,#'E'
        MOV    BUF_WEEK+2,#'D'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY            ;THIS LINE WRITE MESSAGE ON LCD
        LJMP   CHK_CO
DDAY5:  MOV    BUF_WEEK,#'T'
        MOV    BUF_WEEK+1,#'H'
        MOV    BUF_WEEK+2,#'U'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY            ;THIS LINE WRITE MESSAGE ON LCD
        LJMP   CHK_CO
DDAY6:  MOV    BUF_WEEK,#'F'
        MOV    BUF_WEEK+1,#'R'
        MOV    BUF_WEEK+2,#'I'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY            ;THIS LINE WRITE MESSAGE ON LCD
        LJMP   CHK_CO
DDAY7:  MOV    BUF_WEEK,#'S'
        MOV    BUF_WEEK+1,#'A'
        MOV    BUF_WEEK+2,#'T'
        MOV    R1,#BUF_WEEK
        LCALL  DISPLAYY            ;THIS LINE WRITE MESSAGE ON LCD
        LJMP   CHK_CO
```

```
DISPLAYY:MOV   A,#90H          ;ADDRESS LINE 3
         LCALL RUN_WEEK
         RET
;*CHECK SIGNAL FROM CONTROLLER1 TO INSTRUCT TAKE TIME =10 SEC*****
CHK_CO:  MOV   DPTR,#PORTA_1   ;WAIT SIGNAL FROM UC1 PASS PORTA_1
         MOVX  A,@DPTR
         CJNE  A,#01H,BB8      ;A='1' FOR CO 1 ONLY
         SWAP  A           ;CHANGE 4 BIT UPPER AND 4 BIT LOWER =10 OR 20H
         MOV   POINTER5,A
BB6:     MOV   DPTR,#PORTC_1
         MOVX  @DPTR,A         ;SEND TO UC1 PASS PORTC_1 = #10H OUT
         LCALL DELAY3M
         MOV   A,#00H
         MOV   DPTR,#PORTC_1
         MOVX  @DPTR,A         ;SEND TO UC1 PASS PORTC_1 = #00H OUT
         MOV   POINTER2,TIM_SEC
         ANL   POINTER2,#0FH
         MOV   POINTER3,POINTER2
         LCALL DELAY3          ;DELAY 1 SEC
         LJMP  MAIN1
BB8:     MOV   POINTER2,TIM_SEC ;|
         ANL   POINTER2,#0FH    ;|
         MOV   A,POINTER2       ; > COUNTER 10 SEC
         CJNE  A,POINTER3,BB9   ;|
         LCALL ACK1             ;| SEND 01H  TO UC1
BB10:    MOV   DPTR,#PORTA_1
         MOVX  A,@DPTR
         CJNE  A,#10H,BB10     ;WAIT PORTA_1 FROM UC1 SEND 10H COME
;*UC1 SEND MESSAGE NOBODY RECEIVE TEL PLEASE PRESS * TEL NUMBER AND # **
DISK5:   MOV   DPTR,#PORTA_1
         MOVX  A,@DPTR
         CJNE  A,#0AH,BB35
         LCALL DELAY3M
         LCALL DELAY3M
         SWAP  A
         MOV   DPTR,#PORTC_1
         MOVX  @DPTR,A
         LCALL DELAY3M
         LCALL DELAY3M
         MOV   A,#00H
         MOVX  @DPTR,A
         LJMP  MAIN1           ;[3]
BB6:     LJMP  MAIN1           ;
BB35:    MOV   RR6,#0FFH       ;RR6=#0FFH
         MOV   DPTR,#PORTA_1
         MOVX  A,@DPTR
         CJNE  A,#01H,BB7   ;A=01H GO TO NEXT LINE(PRESS EXTENSION 1)
         MOV   A,#10H
         LCALL RTIME           ;DISPLAY HH.MM.SS
         LCALL DECODE
         MOV   R1,#BUF_TIM
         LCALL RUN_8R1
         LJMP  CHK_CO
BB7:     LCALL RTIME           ;DISPLAY HH.MM.SS
         LCALL DECODE
         MOV   R1,#BUF_TIM
         LCALL RUN_8R1
         DJNZ  RR6,BB7
         MOV   DPTR,#PORTA_1
         MOVX  A,@DPTR
         CJNE  A,#03H,DISK5    ;A=#03H  WILL READ TIME CONTINUE
         LCALL RTIME           ;DISPLAY HH.MM.SS
         LCALL DECODE
         MOV   R1,#BUF_TIM
         LCALL RUN_8R1
         LJMP  RECE_N
;********* CHECK  SUB *********************
CHECK:   MOV   DPTR,#PORTA_2
         MOVX  A,@DPTR         ;READ KEYBOARD SIGNAL FROM PORTA_2
         JB    ACC.4,CHECK1    ;IF DATA AVAILABLE ='1' GO CHECK1
         RET
;***********CHECK1 SUB************
;CHECK  KEYBOARD THAT IS KEY 1,2,3,4,5,6,7,8
CHECK1:  MOVX  A,@DPTR
         JNB   ACC.4,CHECK4
         SJMP  CHECK1
CHECK4:  ANL   A,#00001111B
         CJNE  A,#00H,B1       ;A NOT EQUAL 0 (KEY 1) GO B1
         MOV   BUFFER,A
         LJMP  DETECT
B1:      CJNE  A,#01H,B2       ;A NOT EQUAL 1 (KEY 2) GO B2
         MOV   BUFFER,A
         LJMP  DETECT
B2:      CJNE  A,#02H,B3       ;A NOT EQUAL 2 (KEY 3) GO B3
         MOV   BUFFER,A
         LJMP  DETECT
B3:      CJNE  A,#04H,B4       ;A NOT EQUAL 4 (KEY 4) GO B4
         MOV   BUFFER,A
         LJMP  DETECT
B4:      CJNE  A,#05H,B5       ;A NOT EQUAL 5 (KEY 5) GO B5
         MOV   BUFFER,A
         LJMP  DETECT
B5:      CJNE  A,#06H,B6       ;A NOT EQUAL 6 (KEY 6) GO B6
         MOV   BUFFER,A
         LJMP  DETECT
B6:      CJNE  A,#08H,B7       ;A NOT EQUAL 8 (KEY 7) GO B7
         MOV   BUFFER,A
         LJMP  DETECT
B7:      CJNE  A,#09H,B8       ;A NOT EQUAL 9 (KEY 8) GO B8
         MOV   BUFFER,A
```

```
        LJMP   DETECT
B8:     RET          ;THIS IS SHOW YOU PRESS ANY KEY THAT NOT KEY 1-8
DETECT: MOV    A,#01H          ;CLEAR  DISPLAY
        LCALL  WR_COMM
        LJMP   MAIN3           ;BEGIN WRITE  MESSAGE FACE 2
MAIN3:  MOV    DPTR,#MAINT2
        LCALL  DISPLAY         ;WRITE  MESSAGE LCD 2
        MOV    A,#10001110B    ;TELL POSITION TO WRITE NUMBER ROOM
        LCALL  WR_COMM
        MOV    A,BUFFER        ;MOVE NUMBER  ROOM GO TO A
        CJNE   A,#10,NOTEQ1
        SJMP   GRTR1
NOTEQ1: JC     ASCZ1
GRTR1:  ADD    A,#'A'-'9'-1
ASCZ1:  ADD    A,#'0'          ;CONVERT HEX TO ASCII SUCCESS
        CJNE   A,#30H,INDE1    ;PRESS 1
        ADD    A,#1
        LCALL  WRITE
        LJMP   INDE8
INDE1:  CJNE   A,#31H,INDE2    ;PRESS 2
        ADD    A,#1
        LCALL  WRITE
        LJMP   INDE8
INDE2:  CJNE   A,#32H,INDE3    ;PRESS 3
        ADD    A,#1
        LCALL  WRITE
        LJMP   INDE8
INDE3:  CJNE   A,#34H,INDE4    ;PRESS 4
        LCALL  WRITE
        LJMP   INDE8
INDE4:  CJNE   A,#35H,INDE5    ;PRESS 5
        LCALL  WRITE
        LJMP   INDE8
INDE5:  CJNE   A,#36H,INDE6    ;PRESS 6
        LCALL  WRITE
        LJMP   INDE8
INDE6:  CJNE   A,#38H,INDE7    ;PRESS 7
        SUBB   A,#1
        LCALL  WRITE
        SJMP   INDE8
INDE7:  CJNE   A,#39H,INDE8    ;PRESS 8
        SUBB   A,#1
        LCALL  WRITE
        SJMP   INDE8
INDE8:  LJMP   M3
M3:     MOV    DPTR,#PORTA_2
        MOVX   A,@DPTR          ;READ KEYBOARD SIGNAL FROM KEYBOARD
        JB     ACC.4,CHECK3
        SJMP   M3
;************ CHECK3 SUB *****************
```

```
;CHECK IT THAT KEY SIGNAL IS =1,2,3
CHECK3: MOV    DPTR,#PORTA_2
        MOVX   A,@DPTR
        JB     ACC.4,CHECK3
        ANL    A,#00001111B
        CJNE   A,#00H,B9        ;KEY 1
        LJMP   S_LEAVE
B9:     CJNE   A,#01H,B10       ;KEY 2
        LJMP   SUM_USE
B10:    CJNE   A,#02H,B11       ;KEY 3
        MOV    A,#01H
        LCALL  WR_COMM          ;CLEAR DISPLAY
        MOV    DPTR,#MAINT1
        LCALL  DISPLAY          ;WRITE  PROJECT PABX ,4A
        MOV    DPTR,#PORTA_2
        MOVX   A,@DPTR
        JB     ACC.4,$-1
        RET
B11:    LJMP   CHECK3
S_LEAVE: MOV    A,#01H
        LCALL  WR_COMM          ;CLEAR DISPLAY
        MOV    DPTR,#ROOM1
        MOV    A,BUFFER
        SJMP   RT1
        LJMP   ZONE1
RT1:    CJNE   A,#00H,$-3       ;IF A ='1' GO NEXT LINE
        ;**************BEGIN WRITE LEAVE NUMBER ON LCD************
        LCALL  DISPLAYYY        ;SHOW DATA OF NUMBER 1
        MOV    0F0H,#01H        ;B=1
        ;**************SUB PROGRAM CHECK '*' BUTTON *************
CH1:    MOV    DPTR,#PORTA_2
        MOVX   A,@DPTR          ;READ KEYBOARD SIGNAL
        JB     ACC.4,CH2        ;DATA AVAILIABLE = 1 GO CH2
        SJMP   CH1
CH2:    MOVX   A,@DPTR
        JNB    ACC.4,CH3
        SJMP   CH2
CH3:    ANL    A,#00001111B
        CJNE   A,#0CH,DISTRICT1 ;A NOT EQUAL BUTTON '*' GO DISTRICT1 LINE
        MOV    A,0F0H
        CJNE   A,#01H,NEXT1
        LJMP   CH4              ;BEGIN WRITE 2 LINE
NEXT1:  CJNE   A,#02H,NEXT2
        LJMP   CH8              ;BEGIN WRITE 3 LINE
NEXT2:  CJNE   A,#03H,NEXT3
        LJMP   CH9              ;BEGIN WRITE 4 LINE
NEXT3:  CJNE   A,#04H,NE_XT3
        LJMP   C_H9             ;BEGIN WRITE 5 LINE
NE_XT3: CJNE   A,#05H,NE_XT4
        LJMP   C_H10            ;BEGIN WRITE 6 LINE
```

```
NE_XT4:  CJNE   A,#06H,NE_XT5
         LJMP   C_H11              ;BEGIN WRITE 7 LINE
NE_XT5:  CJNE   A,#07H,NE_XT6
         LJMP   C_H12              ;BEGIN WRITE 8 LINE
NE_XT6:  CJNE   A,#08H,NE_XT7
         LJMP   C_H13              ;BEGIN WRITE 9 LINE
NE_XT7:  CJNE   A,#09H,NE_XT8
         LJMP   C_H14              ;BEGIN WRITE 10 LINE
NE_XT8:  LJMP   CH1                ;WAIT PRESS ANY KEY
;**************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT1: CJNE   A,#0EH,DISTRICT2;A =/ BUTTON'#'GO DISTRICT2 LINE
           MOV    A,B
           CJNE   A,#01H,HEN1    ;A NOT EQUAL 0 GO TO HEN1 LINE
           LJMP   CH1
HEN1:      CJNE   A,#02H,HEN2    ;A NOT EQUAL 1 GO TO HEN2 LINE PASS 2
           MOV    BUFFER,A
           LCALL  DISPLAYYYY     ;BEGIN WRITE 1
           MOV    0F0H,#01H
           LJMP   CH1
HEN2:      CJNE   A,#03H,HEN3    ;A NOT EQUAL 2 GO TO HEN3 LINE PASS 3
           MOV    BUFFER,A
           LCALL  DISPLAYYYY     ;BEGIN WRITE 2
           MOV    0F0H,#02H
           LJMP   CH1
HEN3:      CJNE   A,#04H,HE_N4   ;PASS WRITING LINE 4
           MOV    BUFFER,A
           LCALL  DISPLAYYYY     ;BEGIN WRITE 3
           MOV    0F0H,#03H
           LJMP   CH1
HE_N4:     CJNE   A,#05H,HE_N5      ;PASS WRITING LINE 5
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 4
           MOV    0F0H,#04H
           LJMP   CH1
HE_N5:     CJNE   A,#06H,HE_N6      ;PASS WRITING LINE 6
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 5
           MOV    0F0H,#05H
           LJMP   CH1
DISTRICT2: CJNE   A,#02H,DISTRICT3 ;A NOT EQUAL BUTTON '3' GO CH1 LINE
           MOV    A,#01H
           LCALL  WR_COMM           ;CLEAR DISPLAY
           MOV    BUFFER,#00H       ;ROOM1
           LJMP   MAIN3             ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT3: LJMP   CH1
HE_N6:     CJNE   A,#07H,HE_N7      ;PASS WRITING LINE 7
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 6
           MOV    0F0H,#06H
           LJMP   CH1
HE_N7:     CJNE   A,#08H,HE_N8      ;PASS WRITING LINE 8
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 7
           MOV    0F0H,#07H
           LJMP   CH1
HE_N8:     CJNE   A,#09H,HE_N9      ;PASS WRITING LINE 9
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 8
           MOV    0F0H,#08H
           LJMP   CH1
HE_N9:     CJNE   A,#0AH,HE_N10     ;PASS WRITING LINE 10
           MOV    BUFFER,A
           LCALL  DISPLAYYYY        ;BEGIN WRITE 9
           MOV    0F0H,#09H
           LJMP   CH1
HE_N10:    NOP
           ;********** WRITE NEXT NUMBER ON LCD ***************
CH4:       LCALL  DISPLAY5          ;NUMBER 2
           MOV    0F0H,#02H         ;B=2
           LJMP   CH1
CH8:       LCALL  DISPLAY5          ;NUMBER 3
           MOV    0F0H,#03H         ;B=3
           LJMP   CH1
CH9:       LCALL  DISPLAY5          ;NUMBER 4
           MOV    0F0H,#04H         ;B=4
           LJMP   CH1
C_H9:      LCALL  DISPLAY5          ;NUMBER 5
           MOV    B,#05H            ;B=5
           LJMP   CH1
C_H10:     LCALL  DISPLAY5          ;NUMBER 6
           MOV    B,#06H            ;B=6
           LJMP   CH1
C_H11:     LCALL  DISPLAY5          ;NUMBER 7
           MOV    B,#07H            ;B=7
           LJMP   CH1
C_H12:     LCALL  DISPLAY5          ;NUMBER 8
           MOV    B,#08H            ;B=8
           LJMP   CH1
C_H13:     LCALL  DISPLAY5          ;NUMBER 9
           MOV    B,#09H            ;B=9
           LJMP   CH1
C_H14:     LCALL  DISPLAY5          ;NUMBER 10
           MOV    B,#0AH            ;B=10
           LJMP   CH1
           LJMP   ZONE2
ZONE1:     CJNE   A,#01H,$-3        ;IF A ='2' GO TO NEXT LINE
           ;**************BEGIN WRITE LEAVE NUMBER ON LCD************
           LCALL  STAGE1            ;SHOW DATA OF NUMBER 2
           MOV    0F0H,#01H         ;B=1
           ;*************SUB PROGRAM CHECK '*' BUTTON *************
```

```
CH10:    MOV   DPTR,#PORTA_2
         MOVX  A,@DPTR          ;READ KEYBOARD SIGNAL
         JB    ACC.4,CH11       ;DATA AVAILIABLE = 1 GO CH2
         SJMP  CH10
CH11:    MOVX  A,@DPTR
         JNB   ACC.4,CH12
         SJMP  CH11
CH12:    ANL   A,#00001111B
         CJNE  A,#0CH,DISTRICT4;A NOT EQUAL BUTTON'*'GO CH1 LINE
         MOV   A,0F0H
         CJNE  A,#01H,NEXT4
         LJMP  CH13             ;BEGIN WRITE 2 LINE
NEXT4:   CJNE  A,#02H,NEXT5
         LJMP  CH14             ;BEGIN WRITE 3 LINE
NEXT5:   CJNE  A,#03H,NEX_T6
         LJMP  CH15             ;BEGIN WRITE 4 LINE
NEX_T6:  CJNE  A,#04H,NEX_T7
         LJMP  C_H16            ;BEGIN WRITE 5 LINE
NEX_T7:  CJNE  A,#05H,NEX_T8
         LJMP  C_H17            ;BEGIN WRITE 6 LINE
NEX_T8:  CJNE  A,#06H,NEX_T9
         LJMP  C_H18            ;BEGIN WRITE 7 LINE
NEX_T9:  CJNE  A,#07H,NEX_T10
         LJMP  C_H19            ;BEGIN WRITE 8 LINE
NEX_T10: CJNE  A,#08H,NEX_T11
         LJMP  C_H20            ;BEGIN WRITE 9 LINE
NEX_T11: CJNE  A,#09H,NEX_T12
         LJMP  C_H21            ;BEGIN WRITE 10 LINE
NEX_T12: LJMP  CH10
;**************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT4: CJNE  A,#0EH,DISTRICT5  ;A =/ BUTTON'#'GO DISTRICT2 LINE
         MOV   A,B
         CJNE  A,#01H,HEN5        ;A NOT EQUAL 0 GO TO HEN1 LINE
         LJMP  CH10
HEN5:    CJNE  A,#02H,HEN6        ;A NOT EQUAL 1 GO TO HEN2 LINE
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 1
         MOV   0F0H,#01H
         LJMP  CH10
HEN6:    CJNE  A,#03H,HEN7  ;A NOT EQUAL 2 GO TO HEN3 LINE PASS 7-9
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 2
         MOV   0F0H,#02H
         LJMP  CH10
HEN7:    CJNE  A,#04H,HEN_8       ;PASS WRITING LINE 4
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 3
         MOV   0F0H,#03H
         LJMP  CH10
HEN_8:   CJNE  A,#05H,HEN_9       ;PASS WRITING LINE 5
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 4
         MOV   0F0H,#04H
         LJMP  CH10
HEN_9:   CJNE  A,#06H,HEN_10      ;PASS WRITING LINE 6
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 5
         MOV   0F0H,#05H
         LJMP  CH10
DISTRICT5: CJNE  A,#02H,DISTRICT6  ;A NOT EQUAL BUTTON '3' GO CH1 LINE
         MOV   A,#01H
         LCALL WR_COMM            ;CLEAR DISPLAY
         MOV   BUFFER,#01H        ;ROOM2
         LJMP  MAIN3              ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT6: LJMP  CH10
HEN_10:  CJNE  A,#07H,HE_N11      ;PASS WRITING LINE 7
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 6
         MOV   0F0H,#06H
         LJMP  CH10
HE_N11:  CJNE  A,#08H,HE_N12      ;PASS WRITING LINE 8
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 7
         MOV   0F0H,#07H
         LJMP  CH10
HE_N12:  CJNE  A,#09H,HE_N13      ;PASS WRITING LINE 9
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 8
         MOV   0F0H,#08H
         LJMP  CH10
HE_N13:  CJNE  A,#0AH,HE_N14      ;PASS WRITING LINE 10
         MOV   BUFFER,A
         LCALL WLCD1              ;BEGIN WRITE 9
         MOV   0F0H,#09H
         LJMP  CH10
HE_N14:  NOP
;************** WRITE NEXT NUMBER ON LCD ****************
CH13:    LCALL STAGE2             ;NUMBER 2
         MOV   0F0H,#02H          ;B=2
         LJMP  CH10
CH14:    LCALL STAGE2             ;NUMBER 3
         MOV   0F0H,#03H          ;B=3
         LJMP  CH10
CH15:    LCALL STAGE2             ;NUMBER 4
         MOV   0F0H,#04H          ;B=4
         LJMP  CH10
C_H16:   LCALL STAGE2             ;NUMBER 5
         MOV   B,#05H             ;B=5
         LJMP  CH10
C_H17:   LCALL STAGE2             ;NUMBER 6
```

```
        MOV     B,#06H          ;B=6
        LJMP    CH10
C_H18:  LCALL   STAGE2          ;NUMBER 7
        MOV     B,#07H          ;B=7
        LJMP    CH10
C_H19:  LCALL   STAGE2          ;NUMBER 8
        MOV     B,#08H          ;B=8
        LJMP    CH10
C_H20:  LCALL   STAGE2          ;NUMBER 9
        MOV     B,#09H          ;B=9
        LJMP    CH10
C_H21:  LCALL   STAGE2          ;NUMBER 10
        MOV     B,#0AH          ;B=10
        LJMP    CH10
        LJMP    ZONE3
ZONE2:  CJNE    A,#02H,$-3      ;IF A ='3' GO TO NEXT LINE
;***************BEGIN WRITE LEAVE NUMBER ON LCD************
        LCALL   STAGE3          ;SHOW DATA OF NUMBER 1
        MOV     0F0H,#01H       ;B=1
;*************SUB PROGRAM CHECK '*' BUTTON *************
CH16:   MOV     DPTR,#PORTA_2
        MOVX    A,@DPTR         ;READ KEYBOARD SIGNAL
        JB      ACC.4,CH17      ;DATA AVAILIABLE = 1 GO CH2
        SJMP    CH16
CH17:   MOVX    A,@DPTR
        JNB     ACC.4,CH18
        SJMP    CH17
CH18:   ANL     A,#00001111B
        CJNE    A,#0CH,DISTRICT7  ; A =/ BUTTON '*' GO DISTRICT7 LINE
        MOV     A,0F0H
        CJNE    A,#01H,NEXT7
        LJMP    CH19            ;BEGIN WRITE 2 LINE
NEXT7:  CJNE    A,#02H,NEXT8
        LJMP    CH20            ;BEGIN WRITE 3 LINE
NEXT8:  CJNE    A,#03H,NEXT_9
        LJMP    CH21            ;BEGIN WRITE 4 LINE
NEXT_9: CJNE    A,#04H,NEXT_10
        LJMP    C_H22           ;BEGIN WRITE 5 LINE
NEXT_10:CJNE    A,#05H,NEXT_11
        LJMP    C_H23           ;BEGIN WRITE 6 LINE
NEXT_11:CJNE    A,#06H,NEXT_12
        LJMP    C_H24           ;BEGIN WRITE 7 LINE
NEXT_12:CJNE    A,#07H,NEXT_13
        LJMP    C_H25           ;BEGIN WRITE 8 LINE
NEXT_13:CJNE    A,#08H,NEXT_14
        LJMP    C_H26           ;BEGIN WRITE 9 LINE
NEXT_14:CJNE    A,#09H,NEXT_15
        LJMP    C_H27           ;BEGIN WRITE 10 LINE
NEXT_15:LJMP    CH16
;***************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT7:CJNE  A,#0EH,DISTRICT8  ;A =/ BUTTON '#' GO DISTRICT2  LINE
        MOV     A,B
        CJNE    A,#01H,HEN9     ;A NOT EQUAL 0 GO TO HEN1 LINE
        LJMP    CH16
HEN9:   CJNE    A,#02H,HEN10    ;A NOT EQUAL 1 GO TO HEN2 LINE PASS 2
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 1
        MOV     0F0H,#01H
        LJMP    CH16
HEN10:  CJNE    A,#03H,HEN11    ;A NOT EQUAL 2 GO TO HEN3 LINE PASS 3
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 2
        MOV     0F0H,#02H
        LJMP    CH16
HEN11:  CJNE    A,#04H,HEN_12   ;PASS WRITING LINE 4
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 3
        MOV     0F0H,#03H
        LJMP    CH16
HEN_12: CJNE    A,#05H,HEN_13   ;PASS WRITING LINE 5
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 4
        MOV     0F0H,#04H
        LJMP    CH16
HEN_13: CJNE    A,#06H,HEN_14   ;PASS WRITING LINE 6
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 5
        MOV     0F0H,#05H
        LJMP    CH16
DISTRICT8:CJNE  A,#02H,DISTRICT9  ;A NOT EQUAL BUTTON '3' GO CH1 LINE
        MOV     A,#01H
        LCALL   WR_COMM         ;CLEAR DISPLAY
        MOV     BUFFER,#02H     ;ROOM3
        LJMP    MAIN3           ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT9:LJMP  CH16
HEN_14: CJNE    A,#07H,HEN_15   ;PASS WRITING LINE 7
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 6
        MOV     0F0H,#06H
        LJMP    CH16
HEN_15: CJNE    A,#08H,HEN_16   ;PASS WRITING LINE 8
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 7
        MOV     0F0H,#07H
        LJMP    CH16
HEN_16: CJNE    A,#09H,HEN_17   ;PASS WRITING LINE 9
        MOV     BUFFER,A
        LCALL   WLCD2           ;BEGIN WRITE 8
        MOV     0F0H,#08H
        LJMP    CH16
```

```
HEN_17:  CJNE   A,#0AH,HEN_18     ;PASS WRITING LINE 10
         MOV    BUFFER,A
         LCALL  WLCD2              ;BEGIN WRITE 9
         MOV    0F0H,#09H
         LJMP   CH16
HEN_18:  NOP
;************** WRITE NEXT NUMBER ON LCD ****************
CH19:    LCALL  STAGE4             ;NUMBER 2
         MOV    0F0H,#02H          ;B=2
         LJMP   CH16
CH20:    LCALL  STAGE4             ;NUMBER 3
         MOV    0F0H,#03H          ;B=3
         LJMP   CH16
CH21:    LCALL  STAGE4             ;NUMBER 4
         MOV    0F0H,#04H          ;B=4
         LJMP   CH16
C_H22:   LCALL  STAGE4             ;NUMBER 5
         MOV    0F0H,#05H          ;B=5
         LJMP   CH16
C_H23:   LCALL  STAGE4             ;NUMBER 6
         MOV    0F0H,#06H          ;B=6
         LJMP   CH16
C_H24:   LCALL  STAGE4             ;NUMBER 7
         MOV    0F0H,#07H          ;B=7
         LJMP   CH16
C_H25:   LCALL  STAGE4             ;NUMBER 8
         MOV    0F0H,#08H          ;B=8
         LJMP   CH16
C_H26:   LCALL  STAGE4             ;NUMBER 9
         MOV    0F0H,#09H          ;B=9
         LJMP   CH16
C_H27:   LCALL  STAGE4             ;NUMBER 10
         MOV    0F0H,#0AH          ;B=10
         LJMP   CH16
         LJMP   ZONE4
ZONE3:   CJNE   A,#04H,$-3         ;IF  A= '4' GO TO NEXT LINE
;**************BEGIN WRITE LEAVE NUMBER ON LCD***********
         LCALL  STAGE5
         MOV    0F0H,#03H          ;B=3
;**************SUB PROGRAM CHECK '*' BUTTON *************
CH22:    MOV    DPTR,#PORTA_2
         MOVX   A,@DPTR            ;READ KEYBOARD SIGNAL
         JB     ACC.4,CH23         ;DATA AVAILIABLE = 1 GO CH2
         SJMP   CH22
CH23:    MOVX   A,@DPTR
         JNB    ACC.4,CH24
         SJMP   CH23
CH24:    ANL    A,#00001111B
         CJNE   A,#0CH,DISTRICT10 ; A =/ BUTTON '*' GO CH1 LINE
         MOV    A,0F0H
         CJNE   A,#03H,NEXT10
         LJMP   CH25
NEXT10:  CJNE   A,#06H,NEXT11
         LJMP   CH26
NEXT11:  CJNE   A,#09H,NEXT12
         LJMP   CH27
NEXT12:  LJMP   CH22
;        LJMP   ZONE4
;**************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT10: CJNE   A,#0EH,DISTRICT11 ;A =/ BUTTON'#'GO DISTRICT11 LINE
            MOV    A,B
            CJNE   A,#03H,HEN13      ;A =/ 3 GO TO HEN13 LINE
            LJMP   CH22
HEN13:      CJNE   A,#06H,HEN14      ;A =/ 6 GO TO HEN14 LINE PASS 4-6
            MOV    BUFFER,A
            LCALL  WLCD3             ;BEGIN WRITE 1-3
            MOV    0F0H,#03H
            LJMP   CH22
HEN14:      CJNE   A,#09H,HEN15      ;A =/ 9 GO TO HEN15 LINE PASS 7-9
            MOV    BUFFER,A
            LCALL  WLCD3             ;BEGIN WRITE 4-6
            MOV    0F0H,#06H
            LJMP   CH22
HEN15:      CJNE   A,#0CH,HEN16      ;PASS WRITING LINE 10
            MOV    BUFFER,A
            LCALL  WLCD3             ;BEGIN WRITE 7-9
            MOV    0F0H,#09H
            LJMP   CH22
DISTRICT11: CJNE   A,#02H,DISTRICT12 ;A =/ BUTTON '3' GO CH1 LINE
            MOV    A,#01H
            LCALL  WR_COMM           ;CLEAR DISPLAY
            MOV    BUFFER,#04H       ;ROOM4
            LJMP   MAIN3             ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT12: LJMP   CH22
HEN16:      LJMP   CH22
;************** WRITE NEXT NUMBER ON LCD ****************
CH25:    LCALL  STAGE6
         MOV    0F0H,#06H          ;B=6
         LJMP   CH22
CH26:    LCALL  STAGE6
         MOV    0F0H,#09H          ;B=9
         LJMP   CH22
CH27:    LCALL  STAGE6
         MOV    0F0H,#0CH
         LJMP   CH22
         LJMP   ZONE5
ZONE4:   CJNE   A,#05H,$-3         ;IF A ='5' GO TONEXT LINE
;**************BEGIN WRITE LEAVE NUMBER ON LCD***********
         LCALL  STAGE7
         MOV    0F0H,#03H          ;B=3
```

```
;*************SUB PROGRAM CHECK '*' BUTTON *************
CH28:   MOV     DPTR,#PORTA_2
        MOVX    A,@DPTR                 ;READ KEYBOARD SIGNAL
        JB      ACC.4,CH29              ;DATA AVAILIABLE = 1 GO CH2
        SJMP    CH28
CH29:   MOVX    A,@DPTR
        JNB     ACC.4,CH30
        SJMP    CH29
CH30:   ANL     A,#00001111B
        CJNE    A,#0CH,DISTRICT13  ; A =/ BUTTON '*' GO CH1 LINE
        MOV     A,0F0H
        CJNE    A,#03H,NEXT13
        LJMP    CH31
NEXT13: CJNE    A,#06H,NEXT14
        LJMP    CH32
NEXT14: CJNE    A,#09H,NEXT15
        LJMP    CH33
NEXT15: LJMP    CH28
;***************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT13:CJNE   A,#0EH,DISTRICT14  ;A =/ BUTTON '#' GO DISTRICT2 LINE
           MOV    A,B
           CJNE   A,#03H,HEN17       ;A =/ 3 GO TO HEN1 LINE
           LJMP   CH28
HEN17:     CJNE   A,#06H,HEN18       ;A =/ 6 GO TO HEN2 LINE PASS 4-6
           MOV    BUFFER,A
           LCALL  WLCD4              ;BEGIN WRITE 1-3
           MOV    0F0H,#03H
           LJMP   CH28
HEN18:     CJNE   A,#09H,HEN19       ;A =/ 66 GO TO HEN3 LINE PASS 7-9
           MOV    BUFFER,A
           LCALL  WLCD4              ;BEGIN WRITE 4-6
           MOV    0F0H,#06H
           LJMP   CH28
HEN19:     CJNE   A,#0CH,HEN20       ;PASS WRITING LINE 10
           MOV    BUFFER,A
           LCALL  WLCD4              ;BEGIN WRITE 7-9
           MOV    0F0H,#09H
           LJMP   CH28
DISTRICT14:CJNE   A,#02H,DISTRICT15  ;A =/ BUTTON '3' GO CH1 LINE
           MOV    A,#01H
           LCALL  WR_COMM            ;CLEAR DISPLAY
           MOV    BUFFER,#05H        ;ROOM5
           LJMP   MAIN3              ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT15:LJMP   CH28
HEN20:     LJMP   CH28
;************** WRITE NEXT NUMBER ON LCD ***************
CH31:   LCALL   STAGE8
        MOV     0F0H,#06H           ;B=6
        LJMP    CH28
CH32:   LCALL   STAGE8
        MOV     0F0H,#09H           ;B=9
        LJMP    CH28
CH33:   LCALL   STAGE8
        MOV     0F0H,#0CH           ;B=12
        LJMP    CH28
        LJMP    ZONE6
ZONE5:  CJNE    A,#06H,$-3          ;IF A='6' GO TO NEXT LINE
;**************BEGIN WRITE LEAVE NUMBER ON LCD***********
        LCALL   STAGE9
        MOV     0F0H,#03H           ;B=3
;*************SUB PROGRAM CHECK '*' BUTTON *************
CH34:   MOV     DPTR,#PORTA_2
        MOVX    A,@DPTR             ;READ KEYBOARD SIGNAL
        JB      ACC.4,CH35          ;DATA AVAILIABLE = 1 GO CH2
        SJMP    CH34
CH35:   MOVX    A,@DPTR
        JNB     ACC.4,CH36
        SJMP    CH35
CH36:   ANL     A,#00001111B
        CJNE    A,#0CH,DISTRICT16 ; A =/ BUTTON '*' GO CH1 LINE
        MOV     A,0F0H
        CJNE    A,#03H,NEXT16
        LJMP    CH37
NEXT16: CJNE    A,#06H,NEXT17
        LJMP    CH38
NEXT17: CJNE    A,#09H,NEXT18
        LJMP    CH39
NEXT18: LJMP    CH34
;***************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT16:CJNE   A,#0EH,DISTRICT17 ;A =/ BUTTON '#' GO DISTRICT2  LINE
           MOV    A,B
           CJNE   A,#03H,HEN21       ;A =/ 3 GO TO HEN1 LINE
           LJMP   CH34
HEN21:     CJNE   A,#06H,HEN22       ;A =/ 6 GO TO HEN2 LINE PASS 4-6
           MOV    BUFFER,A
           LCALL  WLCD5              ;BEGIN WRITE 1-3
           MOV    0F0H,#03H
           LJMP   CH34
HEN22:     CJNE   A,#09H,HEN23       ;A =/ 66 GO TO HEN3 LINE PASS 7-9
           MOV    BUFFER,A
           LCALL  WLCD5              ;BEGIN WRITE 4-6
           MOV    0F0H,#06H
           LJMP   CH34
HEN23:     CJNE   A,#0CH,HEN24       ;PASS WRITING LINE 10
           MOV    BUFFER,A
           LCALL  WLCD5              ;BEGIN WRITE 7-9
           MOV    0F0H,#09H
           LJMP   CH34
DISTRICT17:CJNE   A,#02H,DISTRICT18 ;A =/ BUTTON '3' GO CH1 LINE
           MOV    A,#01H
```

```
          LCALL   WR_COMM          ;CLEAR DISPLAY
          MOV     BUFFER,#06H      ;ROOM6
          LJMP    MAIN3            ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT18:LJMP   CH34
HEN24:    LJMP    CH34
;************* WRITE NEXT NUMBER ON LCD ****************
CH37:     LCALL   STAGE10
          MOV     0F0H,#06H        ;B=6
          LJMP    CH34
CH38:     LCALL   STAGE10
          MOV     0F0H,#09H        ;B=9
          LJMP    CH34
CH39:     LCALL   STAGE10
          MOV     0F0H,#0CH
          LJMP    CH34
          LJMP    ZONE7
ZONE6:    CJNE    A,#08H,$-3       ;IF A ='7' GO TO NEXT LINE
;**************BEGIN WRITE LEAVE NUMBER ON LCD***********
          LCALL   STAGE11
          MOV     0F0H,#03H        ;B=3
;*************SUB PROGRAM CHECK '*' BUTTON *************
CH40:     MOV     DPTR,#PORTA_2
          MOVX    A,@DPTR          ;READ KEYBOARD SIGNAL
          JB      ACC.4,CH41       ;DATA AVAILIABLE = 1 GO CH2
          SJMP    CH40
CH41:     MOVX    A,@DPTR
          JNB     ACC.4,CH42
          SJMP    CH41
CH42:     ANL     A,#00001111B
          CJNE    A,#0CH,DISTRICT19   ; A =/ BUTTON '*' GO CH1 LINE
          MOV     A,0F0H
          CJNE    A,#03H,NEXT19
          LJMP    CH43
NEXT19:   CJNE    A,#06H,NEXT20
          LJMP    CH44
NEXT20:   CJNE    A,#09H,NEXT21
          LJMP    CH45
NEXT21:   LJMP    CH40
;**************BEGIN WRITE GO BACK 3 LINE ******************
DISTRICT19:CJNE   A,#0EH,DISTRICT20 ;A =/ BUTTON '#' GO DISTRICT2  LINE
          MOV     A,B
          CJNE    A,#03H,HEN25     ;A =/ 3 GO TO HEN1 LINE
          LJMP    CH40
HEN25:    CJNE    A,#06H,HEN26     ;A =/ 6 GO TO HEN2 LINE PASS 4-6
          MOV     BUFFER,A
          LCALL   WLCD6            ;BEGIN WRITE 1-3
          MOV     0F0H,#03H
          LJMP    CH40
HEN26:    CJNE    A,#09H,HEN27     ;A =/ 66 GO TO HEN3 LINE PASS 7-9
          MOV     BUFFER,A
          LCALL   WLCD6            ;BEGIN WRITE 4-6
          MOV     0F0H,#06H
          LJMP    CH40
HEN27:    CJNE    A,#0CH,HEN28     ;PASS WRITING LINE 10
          MOV     BUFFER,A
          LCALL   WLCD6            ;BEGIN WRITE 7-9
          MOV     0F0H,#09H
          LJMP    CH40
DISTRICT20:CJNE   A,#02H,DISTRICT21 ;A =/ BUTTON '3' GO CH1 LINE
          MOV     A,#01H
          LCALL   WR_COMM          ;CLEAR DISPLAY
          MOV     BUFFER,#08H      ;ROOM7
          LJMP    MAIN3            ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT21:LJMP   CH40
HEN28:    LJMP    CH40
;************* WRITE NEXT NUMBER ON LCD ****************
CH43:     LCALL   STAGE12
          MOV     0F0H,#06H        ;B=6
          LJMP    CH40
CH44:     LCALL   STAGE12
          MOV     0F0H,#09H        ;B=9
          LJMP    CH40
CH45:     LCALL   STAGE12
          MOV     0F0H,#0CH
          LJMP    CH40
          LJMP    ZONE8
ZONE7:    CJNE    A,#09H,$-3       ;IF A ='8' GO TO NEXT LINE
;**************BEGIN WRITE LEAVE NUMBER ON LCD***********
          LCALL   STAGE13
          MOV     0F0H,#03H        ;B=3
;*************SUB PROGRAM CHECK '*' BUTTON *************
CH46:     MOV     DPTR,#PORTA_2
          MOVX    A,@DPTR          ;READ KEYBOARD SIGNAL
          JB      ACC.4,CH47       ;DATA AVAILIABLE = 1 GO CH2
          SJMP    CH46
CH47:     MOVX    A,@DPTR
          JNB     ACC.4,CH48
          SJMP    CH47
CH48:     ANL     A,#00001111B
          CJNE    A,#0CH,DISTRICT22 ; A =/ BUTTON '*' GO CH1 LINE
          MOV     A,0F0H
          CJNE    A,#03H,NEXT22
          LJMP    CH49
NEXT22:   CJNE    A,#06H,NEXT23
          LJMP    CH50
NEXT23:   CJNE    A,#09H,NEXT24
          LJMP    CH51
NEXT24:   LJMP    CH46
;**************BEGIN WRITE GO BACK 3 LINE ******************
```

```
DISTRICT22:CJNE   A,#0EH,DISTRICT23 ;A =/ BUTTON '#' GO DISTRICT2  LINE
           MOV    A,B
           CJNE   A,#03H,HEN29       ;A =/ 3 GO TO HEN1 LINE
           LJMP   CH46
HEN29:     CJNE   A,#06H,HEN30       ;A =/ 6 GO TO HEN2 LINE PASS 4-6
           MOV    BUFFER,A
           LCALL  WLCD7              ;BEGIN WRITE 1-3
           MOV    0F0H,#03H
           LJMP   CH46
HEN30:     CJNE   A,#09H,HEN31       ;A =/ 66 GO TO HEN3 LINE PASS 7-9
           MOV    BUFFER,A
           LCALL  WLCD7              ;BEGIN WRITE 4-6
           MOV    0F0H,#06H
           LJMP   CH46
HEN31:     CJNE   A,#0CH,HEN32       ;PASS WRITING LINE 10
           MOV    BUFFER,A
           LCALL  WLCD7              ;BEGIN WRITE 7-9
           MOV    0F0H,#09H
           LJMP   CH46
DISTRICT23:CJNE   A,#02H,DISTRICT24 ;A =/ BUTTON '3' GO CH1 LINE
           MOV    A,#01H
           LCALL  WR_COMM            ;CLEAR DISPLAY
           MOV    BUFFER,#09H        ;ROOM8
           LJMP   MAIN3              ;BEGIN WRITE  MESSAGE FACE 2
DISTRICT24:LJMP   CH46
HEN32:     LJMP   CH46
;************* WRITE NEXT NUMBER ON LCD ****************
CH49:      LCALL  STAGE14
           MOV    0F0H,#06H          ;B=6
           LJMP   CH46
CH50:      LCALL  STAGE14
           MOV    0F0H,#09H          ;B=9
           LJMP   CH46
CH51:      LCALL  STAGE14
           MOV    0F0H,#0CH
           LJMP   CH46
ZONE8:     SJMP   $
DISPLAYYY: MOV    A,#80H             ;SET ADDRESS LINE 1
           LCALL  RUN_TT1
           MOV    A,#0C0H            ;SET ADDRESS LINE 2
           LCALL  WR_COMM
           MOV    R6,#11
           MOV    DPTR,#CALENDAR1
           LCALL  RUN_XY1
           MOV    A,#90H             ;SET ADDRESS LINE 3
           LCALL  WR_COMM
           MOV    R6,#16
           MOV    R1,#10010000B
           MOV    DPTR,#RECTIME1
           LCALL  RUN_XZ1
           MOV    A,#0D0H            ;SET ADDRESS LINE 4
           MOV    DPTR,#MAINT4+48
           LCALL  RUN_TT
           MOV    A,#11010011B       ;TELL POSITION TO WRITE CHARACTER
           LCALL  WR_COMM
           MOV    A,#7EH             ;WRITE ARROW UP ON WANTED POSITION
           LCALL  WRITE
           MOV    A,#11011000B       ;TELL POSITION TO WRITE CHARACTER
           LCALL  WR_COMM
           MOV    A,#7FH             ;WRITE ARROW DOWN ON WANTED POSITION
           LCALL  WRITE
           RET
DISPLAY5:  CLR    C
           MOV    A,#01H
           LCALL  WR_COMM            ;CLEAR DISPLAY
           MOV    A,#80H             ;ADDRESS LINE 1
           LCALL  WR_COMM
           MOV    R6,#11
           CLR    A
           MOV    A,#0BH             ;A=11
           MUL    AB          ;A*B=11,22,33,44,55,66,77,88,99 THEN
;A=11,22,33,44,55,66,77,88,99 B=00
           MOV    DPTR,#ROOM1
           ADD    A,DPL              ;A=A+DPL
           MOV    DPL,A              ;MOVE  DATA IN A GO TO DPL
           LCALL  RUN_XX1            ;WRITE NEXT LINE ON LCD
           MOV    A,#0C0H            ;ADDRESS LINE 2
           LCALL  WR_COMM
           MOV    R6,#11
           MOV    A,DPL              ;MOVE DPL IN A
           CJNE   A,#16H,ZZ1         ;A=#16H GO TO NEXT LINE
           MOV    B,#01H             ;B=1
           SJMP   SUM
ZZ1:       CJNE   A,#21H,ZZ2         ;A=#21H GO TO NEXT LINE
           MOV    B,#02H             ;B=2
           SJMP   SUM
ZZ2:       CJNE   A,#2CH,ZZ3         ;A=#2CH GO TO NEXT LINE
           MOV    B,#03H             ;B=3
           SJMP   SUM
ZZ3:       CJNE   A,#37H,ZZ4         ;A=#37H GO TO NEXT LINE
           MOV    B,#04H             ;B=4
           SJMP   SUM
ZZ4:       CJNE   A,#42H,ZZ5         ;A=#42H GO TO NEXT LINE
           MOV    B,#05H             ;B=5
           SJMP   SUM
ZZ5:       CJNE   A,#4DH,ZZ6         ;A=#4DH GO TO NEXT LINE
           MOV    B,#06H             ;B=6
           SJMP   SUM
ZZ6:       CJNE   A,#58H,ZZ7         ;A=#58H GO TO NEXT LINE
           MOV    B,#07H             ;B=7
```

```
        SJMP   SUM
ZZ7:    CJNE   A,#63H,ZZ8          ;A=#63H GO TO NEXT LINE
        MOV    B,#08H              ;B=8
        SJMP   SUM
ZZ8:    CJNE   A,#6EH,ZZ9          ;A=#6EH GO TO NEXT LINE
        MOV    B,#09H              ;B=9
        SJMP   SUM
SUM:    MOV    A,#0BH              ;A=11
        MUL    AB          ;A*B=11,22,33,44,55,66,77,88,99 THEN
;A=11,22,33,44,55,66,77,88,99 B=00
        MOV    DPTR,#CALENDAR1
        ADD    A,DPL               ;A=A+DPL
        MOV    DPL,A
        LCALL  RUN_XY1             ;WRITE NEXT LINE ON LCD
        MOV    A,#90H              ;ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#16
        MOV    A,DPL
ZZ9:    CJNE   A,#26H,ZZ10         ;A=#26H GO TO NEXT LINE
        MOV    B,#01H              ;B=1
        SJMP   SUM1
ZZ10:   CJNE   A,#31H,ZZ11         ;A=#31H GO TO NEXT LINE
        MOV    B,#02H              ;B=2
        SJMP   SUM1
ZZ11:   CJNE   A,#3CH,ZZ12         ;A=#3CH GO TO NEXT LINE
        MOV    B,#03H              ;B=3
        SJMP   SUM1
ZZ12:   CJNE   A,#47H,ZZ13         ;A=#47H GO TO NEXT LINE
        MOV    B,#04H              ;B=4
        SJMP   SUM1
ZZ13:   CJNE   A,#52H,ZZ14         ;A=#52H GO TO NEXT LINE
        MOV    B,#05H              ;B=5
        SJMP   SUM1
ZZ14:   CJNE   A,#5DH,ZZ15         ;A=#5DH GO TO NEXT LINE
        MOV    B,#06H              ;B=6
        SJMP   SUM1
ZZ15:   CJNE   A,#68H,ZZ16         ;A=#68H GO TO NEXT LINE
        MOV    B,#07H              ;B=7
        SJMP   SUM1
ZZ16:   CJNE   A,#73H,ZZ17         ;A=#73H GO TO NEXT LINE
        MOV    B,#08H              ;B=8
        SJMP   SUM1
ZZ17:   CJNE   A,#7EH,ZZ18         ;A=#7EH GO TO NEXT LINE
        MOV    B,#09H              ;B=9
        SJMP   SUM1
ZZ18:   LJMP   CH1
SUM1:   MOV    A,#10H              ;A=16
        MUL    AB          ;AB=16,32,48,64,80,96,108,126,142,156 THEN
;A=16,32,48,64,80,96,108,126,142,156 B=00
        MOV    DPTR,#RECTIME1

        CLR    C                   ;CLEAR CARRY FLAG = 0
        ADD*   A,DPL               ;A=A+DPL, IT HAS OPPURITY CARRY FLAG = 1
        MOV    DPL,A               ;MOVE  DATA IN A GO TO DPL
        JC     HUW1                ;IF CARRY FLAG = 1 GO HUW1
        SJMP   HUW2
HUW1:   INC    DPH
HUW2:   MOV    R1,#10010000B
        LCALL  RUN_XZ1             ;WRITE NEXT LINE
        MOV    A,#0D0H             ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B        ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH              ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B        ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH              ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
RUN_TT1: LCALL WR_COMM             ;LOAD ONE LINE
        MOV    R6,#11              ;11 CHAR.
        MOV    A,BUFFER
        CJNE   A,#00H,ROM1
        MOV    DPTR,#ROOM1
        SJMP   RUN_XX1
ROM1:   CJNE   A,#01H,ROM2
        MOV    DPTR,#ROOM2
        SJMP   RUN_XX1
ROM2:   CJNE   A,#02H,ROM3
        MOV    DPTR,#ROOM3
        SJMP   RUN_XX1
ROM3:   CJNE   A,#04H,ROM4
        MOV    DPTR,#ROOM4
        SJMP   RUN_XX1
ROM4:   CJNE   A,#05H,ROM5
        MOV    DPTR,#ROOM5
        SJMP   RUN_XX1
ROM5:   CJNE   A,#06H,ROM6
        MOV    DPTR,#ROOM6
        SJMP   RUN_XX1
ROM6:   CJNE   A,#08H,ROM7
        MOV    DPTR,#ROOM7
        SJMP   RUN_XX1
ROM7:   CJNE   A,#09H,ROM8
        MOV    DPTR,#ROOM8
        SJMP   RUN_XX1
ROM8:   LJMP   CH1
RUN_XX1: CLR   A
        MOVX   A,@DPTR             ;READ VALUE IN ROOM1 TO PUT IN A
```

```
        CJNE   A,#0FFH,WR1     ;A NOT EQUAL #0FFH JUMP WR1
        INC    DPTR
        MOV    BUFFER1,DPH
        MOV    BUFFER2,DPL
        DJNZ   R6,RUN_XX1   ;(-) R6 BY 1 THEN =/ 0 JUMP RUN_XX1
        LJMP   CHECK6
WR1:    CJNE   A,#10,NOTEQ
        SJMP   GRTR
NOTEQ:  JC     ASCZ
GRTR:   ADD    A,#'A'-'9'-1
ASCZ:   ADD    A,#'0'
        LCALL  WRITE
        INC    DPTR
        MOV    BUFFER1,DPH
        MOV    BUFFER2,DPL
        DJNZ   R6,RUN_XX1
CHECK6: RET
RUN_XY1: CLR   A
        MOVX   A,@DPTR         ;READ VALUE IN ROOM1 TO PUT IN A
        CJNE   A,#0FFH,WR15    ;A NOT EQUAL #0FFH JUMP WR1
        INC    DPTR
        MOV    BUFFER1,DPH
        MOV    BUFFER2,DPL
        DJNZ   R6,RUN_XY1      ;(-) R6 BY 1 THEN =/ 0 JUMP RUN_XX1
        LJMP   CHECK15
WR15:   CJNE   A,#47,WR16      ;IF A =/ '/' ( / = 47.) GO WR15
        LCALL  WRITE
        INC    DPTR
        DJNZ   R6,RUN_XY1
WR16:   CJNE   A,#10,NOTEQ2
        SJMP   GRTR2
NOTEQ2: JC     ASCZ2
GRTR2:  ADD    A,#'A'-'9'-1
ASCZ2:  ADD    A,#'0'
        LCALL  WRITE
        INC    DPTR
        DJNZ   R6,RUN_XY1
CHECK15: RET
RUN_XZ1: CLR   A
        MOVX   A,@DPTR         ;READ VALUE IN RECTIME1 TO PUT IN A
        CJNE   A,#0FFH,WR17    ;A =/ #0FFH JUMP WR17
        INC    DPTR
        MOV    BUFFER1,DPH
        MOV    BUFFER2,DPL
        INC    R1
        DJNZ   R6,RUN_XZ1      ;(-) R6 BY 1 THEN =/ 0 JUMP RUN_XX1
        LJMP   CHECK16
WR17:   CJNE   A,#58,WR18      ;IF A =/ ':' ( : = 58 ) GO WR17
        LCALL  WRITE
        INC    DPTR
WR18:   INC    R1
        DJNZ   R6,RUN_XZ1
        MOV    BUFFER,A
        ANL    A,#11110000B    ;CHECK A IS CHARACTER OF DAY YES OR NO
        CJNE   A,#40H,WR19     ;A IS CHARACTER GO TO NEXT LINE
        MOV    A,R1            ;TELL POSITION OF CHARACTER
        LCALL  WR_COMM
        MOV    A,BUFFER
        LCALL  WRITE
        INC    DPTR
        INC    R1
        DJNZ   R6,RUN_XZ1
WR19:   CJNE   A,#50H,WR20     ;A IS CHARACTER GO TO NEXT LINE
        MOV    A,DPL
        SUBB   A,#20H          ; A = A-#20H
        ORL    A,#10000000B    ;A = A V #10000000B TELL POSITION OF CHARACTER
        MOV    A,R1            ;TELL  POSITION OF CHARACTER
        LCALL  WR_COMM
        MOV    A,BUFFER
        LCALL  WRITE
        INC    DPTR
        INC    R1
        DJNZ   R6,RUN_XZ1
WR20:   MOV    A,BUFFER
        CJNE   A,#10,NOTEQ3
        SJMP   GRTR3
NOTEQ3: JC     ASCZ3
GRTR3:  ADD    A,#'A'-'9'-1
ASCZ3:  ADD    A,#'0'
        LCALL  WRITE
        INC    DPTR
        INC    R1
        DJNZ   R6,RUN_XZ1
CHECK16: RET
STAGE1: MOV    A,#80H          ;SET ADDRESS LINE 1
        LCALL  RUN_TT1
        MOV    A,#0C0H         ;SET ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        MOV    DPTR,#CALENDAR2
        LCALL  RUN_XY1
        MOV    A,#90H          ;SET ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#16
        MOV    R1,#10010000B
        MOV    DPTR,#RECTIME2
        LCALL  RUN_XZ1
        MOV    A,#0D0H         ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
```

```
        MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
STAGE2: CLR   C
        MOV   A,#01H
        LCALL WR_COMM         ;CLEAR DISPLAY
        MOV   A,#80H          ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        CLR   A
        MOV   A,#0BH          ;A=11
        MUL   AB        ;A*B=11,22,33,44,55,66,77,88,99 THEN
;A=11,22,33,44,55,66,77,88,99 B=00
        MOV   DPTR,#ROOM2
        ADD   A,DPL           ;A=A+DPL
        MOV   DPL,A           ;MOVE  DATA IN A GO TO DPL
        LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV   A,#0C0H         ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        MOV   A,DPL           ;MOVE DPL IN A
        CJNE  A,#84H,ZZ19     ;A=#84H GO TO NEXT LINE
        MOV   B,#01H          ;B=1
        SJMP  SUM6
ZZ19:   CJNE  A,#8FH,ZZ20     ;A=#8FH GO TO NEXT LINE
        MOV   B,#02H          ;B=2
        SJMP  SUM6
ZZ20:   CJNE  A,#9AH,ZZ21     ;A=#9AH GO TO NEXT LINE
        MOV   B,#03H          ;B=3
        SJMP  SUM6
ZZ21:   CJNE  A,#0A5H,ZZ22    ;A=#A5H GO TO NEXT LINE
        MOV   B,#04H          ;B=4
        SJMP  SUM6
ZZ22:   CJNE  A,#0B0H,ZZ23    ;A=#B0H GO TO NEXT LINE
        MOV   B,#05H          ;B=5
        SJMP  SUM6
ZZ23:   CJNE  A,#0BBH,ZZ24    ;A=#BBH GO TO NEXT LINE
        MOV   B,#06H          ;B=6
        SJMP  SUM6
ZZ24:   CJNE  A,#0C6H,ZZ25    ;A=#C6H GO TO NEXT LINE
        MOV   B,#07H          ;B=7
        SJMP  SUM6
ZZ25:   CJNE  A,#0D1H,ZZ26    ;A=#D1H GO TO NEXT LINE
        MOV   B,#08H          ;B=8
        SJMP  SUM6
ZZ26:   CJNE  A,#0DCH,ZZ27    ;A=#DCH GO TO NEXT LINE
        MOV   B,#09H          ;B=9
        SJMP  SUM6
SUM6:   MOV   A,#0BH          ;A=11
        MUL   AB        ;A*B=11,22,33,44,55,66,77,88,99 THEN
;A=11,22,33,44,55,66,77,88,99 B=00
        MOV   DPTR,#CALENDAR2
        ADD   A,DPL           ;A=A+DPL
        MOV   DPL,A
        LCALL RUN_XY1         ;WRITE NEXT LINE ON LCD
        MOV   A,#90H          ;ADDRESS LINE 3
        LCALL WR_COMM
        MOV   R6,#16
        MOV   A,DPL
ZZ27:   CJNE  A,#94H,ZZ28     ;A=#94H GO TO NEXT LINE
        MOV   B,#01H          ;B=1
        SJMP  SUM7
ZZ28:   CJNE  A,#9FH,ZZ29     ;A=#9FH GO TO NEXT LINE
        MOV   B,#02H          ;B=2
        SJMP  SUM7
ZZ29:   CJNE  A,#0AAH,ZZ30    ;A=#94H GO TO NEXT LINE
        MOV   B,#03H          ;B=3
        SJMP  SUM7
ZZ30:   CJNE  A,#0B5H,ZZ31    ;A=#94H GO TO NEXT LINE
        MOV   B,#04H          ;B=4
        SJMP  SUM7
ZZ31:   CJNE  A,#0C0H,ZZ32    ;A=#94H GO TO NEXT LINE
        MOV   B,#05H          ;B=5
        SJMP  SUM7
ZZ32:   CJNE  A,#0CBH,ZZ33    ;A=#94H GO TO NEXT LINE
        MOV   B,#06H          ;B=6
        SJMP  SUM7
ZZ33:   CJNE  A,#0D6H,ZZ34    ;A=#94H GO TO NEXT LINE
        MOV   B,#07H          ;B=7
        SJMP  SUM7
ZZ34:   CJNE  A,#0E1H,ZZ35    ;A=#94H GO TO NEXT LINE
        MOV   B,#08H          ;B=8
        SJMP  SUM7
ZZ35:   CJNE  A,#0ECH,ZZ36    ;A=#94H GO TO NEXT LINE
        MOV   B,#09H          ;B=9
        SJMP  SUM7
ZZ36:   LJMP  CH10
SUM7:   MOV   A,#10H          ;A=16
        MUL   AB              ;AB=16,32,48,64,80,96,108,126,142,156
;THEN A= 16,32,48,64,80,96,108,126,142,156 B=00
        MOV   DPTR,#RECTIME2
        CLR   C               ;CLEAR CARRY FLAG = 0
        ADD   A,DPL           ;A=A+DPL ,IT HAS OPPURITY CARRY FLAG=1
        MOV   DPL,A
```

```
        JC      HUW15
        SJMP    HUW16
HUW15:  INC     DPH
HUW16:  MOV     R1,#10010000B
        LCALL   RUN_XZ1         ;WRITE NEXT LINE ON LCD
        MOV     A,#0D0H         ;SET ADDRESS LINE 4
        MOV     DPTR,#MAINT4+48
        LCALL   RUN_TT
        MOV     A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL   WR_COMM
        MOV     A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL   WRITE
        MOV     A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL   WR_COMM
        MOV     A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL   WRITE
        RET
STAGE3: MOV     A,#80H          ;SET ADDRESS LINE 1
        LCALL   RUN_TT1
        MOV     A,#0C0H         ;SET ADDRESS LINE 2
        LCALL   WR_COMM
        MOV     R6,#11
        MOV     DPTR,#CALENDAR3
        LCALL   RUN_XY1
        MOV     A,#90H          ;SET ADDRESS LINE 3
        LCALL   WR_COMM
        MOV     R6,#16
        MOV     R1,#10010000B
        MOV     DPTR,#RECTIME3
        LCALL   RUN_XZ1
        MOV     A,#0D0H         ;SET ADDRESS LINE 4
        MOV     DPTR,#MAINT4+48
        LCALL   RUN_TT
        MOV     A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL   WR_COMM
        MOV     A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL   WRITE
        MOV     A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL   WR_COMM
        MOV     A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL   WRITE
        RET
STAGE4: CLR     C
        MOV     A,#01H
        LCALL   WR_COMM         ;CLEAR DISPLAY
        MOV     A,#80H          ;ADDRESS LINE 1
        LCALL   WR_COMM
        MOV     R6,#11
        CLR     A
        MOV     A,#0BH          ;A=11
        MUL     AB              ;A*B=11,22,33,44,55,66,77,88,99 THEN
                                ;A=11,22,33,44,55,66,77,88,99 B=00
        MOV     DPTR,#ROOM3
        ADD     A,DPL           ;A=A+DPL
        MOV     DPL,A           ;MOVE  DATA IN A GO TO DPL
        JC      TEST1
        SJMP    TEST2
TEST1:  INC     DPH
TEST2:  LCALL   RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV     A,#0C0H         ;ADDRESS LINE 2
        LCALL   WR_COMM
        MOV     R6,#11
        MOV     A,DPL           ;MOVE DPL IN A
        CJNE    A,#0F2H,ZZ37    ;A=#F2H GO TO NEXT LINE
        MOV     B,#01H          ;B=1
        SJMP    SUM8
ZZ37:   CJNE    A,#0FDH,ZZ38    ;A=#FDH GO TO NEXT LINE
        MOV     B,#02H          ;B=2
        SJMP    SUM8
ZZ38:   CJNE    A,#08H,ZZ39     ;A=#08H GO TO NEXT LINE
        MOV     B,#03H          ;B=3
        SJMP    SUM8
ZZ39:   CJNE    A,#13H,ZZ40     ;A=#13H GO TO NEXT LINE
        MOV     B,#04H          ;B=4
        SJMP    SUM8
ZZ40:   CJNE    A,#1EH,ZZ41     ;A=#1EH GO TO NEXT LINE
        MOV     B,#05H          ;B=5
        SJMP    SUM8
ZZ41:   CJNE    A,#29H,ZZ42     ;A=#29H GO TO NEXT LINE
        MOV     B,#06H          ;B=6
        SJMP    SUM8
ZZ42:   CJNE    A,#34H,ZZ43     ;A=#34H GO TO NEXT LINE
        MOV     B,#07H          ;B=7
        SJMP    SUM8
ZZ43:   CJNE    A,#3FH,ZZ44     ;A=#3FH GO TO NEXT LINE
        MOV     B,#08H          ;B=8
        SJMP    SUM8
ZZ44:   CJNE    A,#4AH,ZZ45     ;A=#4AH GO TO NEXT LINE
        MOV     B,#09H          ;B=9
        SJMP    SUM8
SUM8:   MOV     A,#0BH          ;A=11
        MUL     AB              ;A*B=11,22,33,44,55,66,77,88,99 THEN
                                ;A=11,22,33,44,55,66,77,88,99 B=00
        MOV     DPTR,#CALENDAR3
        CLR     C
        ADD     A,DPL           ;A=A+DPL
        MOV     DPL,A
        JC      HUW17
        SJMP    HUW18
HUW17:  INC     DPH
```

```
HUW18:  LCALL  RUN_XY1       ;WRITE NEXT LINE ON LCD
        MOV    A,#90H        ;ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#16
        MOV    A,DPL
ZZ45:   CJNE   A,#02H,ZZ46   ;A=#02H GO TO NEXT LINE
        MOV    B,#01H        ;B=1
        SJMP   SUM9
ZZ46:   CJNE   A,#0DH,ZZ47   ;A=#0DH GO TO NEXT LINE
        MOV    B,#02H        ;B=2
        SJMP   SUM9
ZZ47:   CJNE   A,#18H,ZZ48   ;A=#1BH GO TO NEXT LINE
        MOV    B,#03H        ;B=3
        SJMP   SUM9
ZZ48:   CJNE   A,#23H,ZZ49   ;A=#26H GO TO NEXT LINE
        MOV    B,#04H        ;B=4
        SJMP   SUM9
ZZ49:   CJNE   A,#2EH,ZZ50   ;A=#31H GO TO NEXT LINE
        MOV    B,#05H        ;B=5
        SJMP   SUM9
ZZ50:   CJNE   A,#39H,ZZ51   ;A=#3CH GO TO NEXT LINE
        MOV    B,#06H        ;B=6
        SJMP   SUM9
ZZ51:   CJNE   A,#44H,ZZ52   ;A=#47H GO TO NEXT LINE
        MOV    B,#07H        ;B=7
        SJMP   SUM9
ZZ52:   CJNE   A,#4FH,ZZ53   ;A=#52H GO TO NEXT LINE
        MOV    B,#08H        ;B=8
        SJMP   SUM9
ZZ53:   CJNE   A,#5AH,ZZ54   ;A=#5DH GO TO NEXT LINE
        MOV    B,#09H        ;B=9
        SJMP   SUM9
ZZ54:   LJMP   CH16
SUM9:   MOV    A,#10H        ;A=16
        MUL    AB            ;A*B=16,32,48,64,80,96,108,126,142,156 THEN
;A=16,32,48,64,80,96,108,126,142,156 B=00
        MOV    DPTR,#RECTIME3
        CLR    C             ;CLEAR CARRY FLAG =0
        ADD    A,DPL         ;A=A+DPL ,IT HAS OPPURITY CARRY FLAG =1
        MOV    DPL,A
        JC     HUW19         ;IF CARRY FLAG=1 GO HUW19
        SJMP   HUW20
HUW19:  INC    DPH
HUW20:  MOV    R1,#10010000B
        LCALL  RUN_XZ1       ;WRITE NEXT LINE ON LCD
        MOV    A,#0D0H       ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
STAGE5: MOV    A,#80H        ;SET ADDRESS LINE 1
        LCALL  RUN_TT1
        MOV    A,#0C0H       ;SET ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#90H        ;SET ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#0D0H       ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
STAGE6: MOV    A,#01H
        LCALL  WR_COMM       ;CLEAR DISPLAY
        MOV    A,#80H        ;ADDRESS LINE 1
        LCALL  WR_COMM
        MOV    R6,#11
        CLR    A
        MOV    A,#0BH        ;A=11
        MUL    AB            ;A*B=66 THEN  A=66 B=00
        MOV    DPTR,#ROOM4
        ADD    A,DPL         ;A=A+DPL
        MOV    DPL,A         ;MOVE  DATA IN A GO TO DPL
        LCALL  RUN_XX1       ;WRITE NEXT LINE ON LCD
        MOV    A,DPL
        CJNE   A,#0B8H,CONTINUE3
        SJMP   BREAK3
CONTINUE3:MOV  A,#0C0H       ;ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1       ;WRITE NEXT LINE ON LCD
        MOV    A,#90H        ;ADDRESS LINE 3
        LCALL  WR_COMM
```

```
          MOV   R6,#11
          LCALL RUN_XX1          ;WRITE NEXT LINE ON LCD
BREAK3:   MOV   A,#0D0H          ;SET ADDRESS LINE 4
          MOV   DPTR,#MAINT4+48
          LCALL RUN_TT
          MOV   A,#11010011B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7EH           ;WRITE ARROW UP ON WANTED POSITION
          LCALL WRITE
          MOV   A,#11011000B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7FH           ;WRITE ARROW DOWN ON WANTED POSITION
          LCALL WRITE
          RET
STAGE7:   MOV   A,#80H           ;SET ADDRESS LINE 1
          LCALL RUN_TT1
          MOV   A,#0C0H          ;SET ADDRESS LINE 2
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1
          MOV   A,#90H           ;SET ADDRESS LINE 3
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1
          MOV   A,#0D0H          ;SET ADDRESS LINE 4
          MOV   DPTR,#MAINT4+48
          LCALL RUN_TT
          MOV   A,#11010011B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7EH           ;WRITE ARROW UP ON WANTED POSITION
          LCALL WRITE
          MOV   A,#11011000B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7FH           ;WRITE ARROW DOWN ON WANTED POSITION
          LCALL WRITE
          RET
STAGE8:   MOV   A,#01H
          LCALL WR_COMM          ;CLEAR DISPLAY
          MOV   A,#80H           ;ADDRESS LINE 1
          LCALL WR_COMM
          MOV   R6,#11
          CLR   A
          MOV   A,#0BH           ;A=11
          MUL   AB               ;A*B=66 THEN  A=66 B=00
          MOV   DPTR,#ROOM5
          ADDC   A,DPL           ;A=A+DPL
          MOV   DPL,A            ;MOVE  DATA IN A GO TO DPL
          JC    TEST3
          CLR   PSW.7
          SJMP  TEST4
TEST3:    INC   DPH
TEST4:    LCALL RUN_XX1          ;WRITE NEXT LINE ON LCD
          MOV   A,DPL
          CJNE  A,#26H,CONTINUE4
          SJMP  BREAK4
CONTINUE4:MOV   A,#0C0H          ;ADDRESS LINE 2
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1          ;WRITE NEXT LINE ON LCD
          MOV   A,#90H           ;ADDRESS LINE 3
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1          ;WRITE NEXT LINE ON LCD
BREAK4:   MOV   A,#0D0H          ;SET ADDRESS LINE 4
          MOV   DPTR,#MAINT4+48
          LCALL RUN_TT
          MOV   A,#11010011B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7EH           ;WRITE ARROW UP ON WANTED POSITION
          LCALL WRITE
          MOV   A,#11011000B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7FH           ;WRITE ARROW DOWN ON WANTED POSITION
          LCALL WRITE
          RET
STAGE9:   MOV   A,#80H           ;SET ADDRESS LINE 1
          LCALL RUN_TT1
          MOV   A,#0C0H          ;SET ADDRESS LINE 2
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1
          MOV   A,#90H           ;SET ADDRESS LINE 3
          LCALL WR_COMM
          MOV   R6,#11
          LCALL RUN_XX1
          MOV   A,#0D0H          ;SET ADDRESS LINE 4
          MOV   DPTR,#MAINT4+48
          LCALL RUN_TT
          MOV   A,#11010011B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7EH           ;WRITE ARROW UP ON WANTED POSITION
          LCALL WRITE
          MOV   A,#11011000B     ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7FH           ;WRITE ARROW DOWN ON WANTED POSITION
          LCALL WRITE
          RET
STAGE10:  MOV   A,#01H
          LCALL WR_COMM          ;CLEAR DISPLAY
          MOV   A,#80H           ;ADDRESS LINE 1
```

```
        LCALL  WR_COMM
        MOV    R6,#11
        CLR    A
        MOV    A,#0BH          ;A=11
        MUL    AB              ;A*B=66 THEN  A=66 B=00
        MOV    DPTR,#ROOM6
        ADD    A,DPL           ;A=A+DPL
        MOV    DPL,A           ;MOVE  DATA IN A GO TO DPL
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV    A,DPL
        CJNE   A,#94H,CONTINUE5
        SJMP   BREAK5
CONTINUE5:MOV  A,#0C0H         ;ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV    A,#90H          ;ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
BREAK5: MOV    A,#0D0H         ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
STAGE11: MOV   A,#80H          ;SET ADDRESS LINE 1
        LCALL  RUN_TT1
        MOV    A,#0C0H         ;SET ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#90H          ;SET ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#0D0H         ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
STAGE12: MOV   A,#01H
        LCALL  WR_COMM         ;CLEAR DISPLAY
        MOV    A,#80H          ;ADDRESS LINE 1
        LCALL  WR_COMM
        MOV    R6,#11
        CLR    A
        MOV    A,#0BH          ;A=11
        MUL    AB              ;A*B=66 THEN  A=66 B=00
        MOV    DPTR,#ROOM7
        ADD    A,DPL           ;A=A+DPL
        MOV    DPL,A           ;MOVE  DATA IN A GO TO DPL
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV    A,DPL
        CJNE   A,#02H,CONTINUE6
        SJMP   BREAK6
CONTINUE6:MOV  A,#0C0H         ;ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
        MOV    A,#90H          ;ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1         ;WRITE NEXT LINE ON LCD
BREAK6: MOV    A,#0D0H         ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
        RET
STAGE13: MOV   A,#80H          ;SET ADDRESS LINE 1
        LCALL  RUN_TT1
        MOV    A,#0C0H         ;SET ADDRESS LINE 2
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#90H          ;SET ADDRESS LINE 3
        LCALL  WR_COMM
        MOV    R6,#11
        LCALL  RUN_XX1
        MOV    A,#0D0H         ;SET ADDRESS LINE 4
```

```
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
STAGE14: MOV  A,#01H
        LCALL WR_COMM       ;CLEAR DISPLAY
        MOV   A,#80H        ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        CLR   A
        MOV   A,#0BH        ;A=11
        MUL   AB            ;A*B=66 THEN  A=66 B=00
        MOV   DPTR,#ROOM8
        ADD   A,DPL         ;A=A+DPL
        MOV   DPL,A         ;MOVE  DATA IN A GO TO DPL
        LCALL RUN_XX1       ;WRITE NEXT LINE ON LCD
        MOV   A,DPL
        CJNE  A,#70H,CONTINUE7
        SJMP  BREAK7
CONTINUE7:MOV A,#0C0H       ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1       ;WRITE NEXT LINE ON LCD
        MOV   A,#90H        ;ADDRESS LINE 3
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1       ;WRITE NEXT LINE ON LCD
BREAK7:  MOV  A,#0D0H       ;SET ADDRESS LINE 4
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
DISPLAYYYY: MOV  A,#01H
        LCALL WR_COMM       ;CLEAR DISPLAY
        MOV   A,#0D0H       ;SET ADDRESS LINE 4
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        MOV   A,#90H        ;ADDRESS LINE 3
        LCALL WR_COMM
        MOV   R6,#16
        CLR   C
        MOV   A,BUFFER
        MOV   A,#01H
        MOV   B,#20H
        MUL   AB            ;A*B=32
        MOV   BUFFER,A      ;STORE RESULT IN BUFFER
        MOV   A,BUFFER2     ;BUFFER2 IS DPL OF RECTIME1
        CLR   C
        SUBB  A,BUFFER      ;A=A(DPL)-BUFFER(32)
        MOV   DPL,A         ;RETURN DPL
        MOV   DPH,BUFFER1
        JC    HUW3
        SJMP  HUW4
HUW3:   DEC   DPH
HUW4:   SJMP  OPERATE
OPERATE: MOV  R1,#10010000B
        LCALL RUN_XZ1       ;WRITE LINE 3
        MOV   A,#0C0H       ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        MOV   A,BUFFER2     ;BUFFER2 IS DPL
        MOV   R1,A          ;FOR USE IN PRESS '#' AGAIN
        MOV   R0,BUFFER1    ;FOR USE IN PRESS '#' AGAIN
        CJNE  A,#90H,HENN1  ;WRITE NUMBER 1
        MOV   B,#00H
        SJMP  SUM2
HENN1:  CJNE  A,#0A0H,HENN2 ;WRITE NUMBER 2
        MOV   B,#01H
        SJMP  SUM2
HENN2:  CJNE  A,#0B0H,HEN_N3 ;WRITE NUMBER 3
        MOV   B,#02H
        SJMP  SUM2
HEN_N3: CJNE  A,#0C0H,HEN_N4 ;WRITE NUMBER 4
        MOV   B,#03H
        SJMP  SUM2
HEN_N4: CJNE  A,#0D0H,HEN_N5 ;WRITE NUMBER 5
        MOV   B,#04H
        SJMP  SUM2
```

```
HEN_N5:   CJNE  A,#0E0H,HEN_N6 ;WRITE NUMBER 6
          MOV   B,#05H
          SJMP  SUM2
HEN_N6:   CJNE  A,#0F0H,HEN_N7 ;WRITE NUMBER 7
          MOV   B,#06H
          SJMP  SUM2
HEN_N7:   CJNE  A,#00H,HEN_N8  ;WRITE NUMBER 8
          MOV   B,#07H
          SJMP  SUM2
HEN_N8:   CJNE  A,#10H,HEN_N9  ;WRITE NUMBER 9
          MOV   B,#08H
          SJMP  SUM2
HEN_N9:   NOP
SUM2:     MOV   DPTR,#CALENDAR1
          MOV   A,#11
          MUL   AB              ;A*B=00,11,22,33,44,55,66,77,88,99
          ADD   A,DPL           ;A=A+DPL
          MOV   DPL,A
          JC    HUW5
          SJMP  HUW6
HUW5:     INC   DPH
HUW6:     LCALL RUN_XY1         ;WRITE LINE 2
          MOV   A,#80H          ;ADDRESS LINE 1
          LCALL WR_COMM
          MOV   R6,#11
          MOV   A,BUFFER2       ;BUFFER2 IS DPL
          CJNE  A,#1BH,NE_XT9   ;WRITE NUMBER 1
          MOV   B,#00H
          SJMP  SUM3
NE_XT9:   CJNE  A,#26H,NE_XT10;WRITE NUMBER 2
          MOV   B,#01H
          SJMP  SUM3
NE_XT10:  CJNE  A,#31H,NE_XT11;WRITE NUMBER 3
          MOV   B,#02H
          SJMP  SUM3
NE_XT11:  CJNE  A,#3CH,NE_XT12;WRITE NUMBER 4
          MOV   B,#03H
          SJMP  SUM3
NE_XT12:  CJNE  A,#47H,NE_XT13;WRITE NUMBER 5
          MOV   B,#04H
          SJMP  SUM3
NE_XT13:  CJNE  A,#52H,NE_XT14;WRITE NUMBER 6
          MOV   B,#05H
          SJMP  SUM3
NE_XT14:  CJNE  A,#5DH,NE_XT15;WRITE NUMBER 7
          MOV   B,#06H
          SJMP  SUM3
NE_XT15:  CJNE  A,#68H,NE_XT16;WRITE NUMBER 8
          MOV   B,#07H
          SJMP  SUM3
NE_XT16:  CJNE  A,#73H,NE_XT17;WRITE NUMBER 9
          MOV   B,#08H
          SJMP  SUM3
NE_XT17:  NOP
SUM3:     MOV   DPTR,#ROOM1
          MOV   A,#11
          MUL   AB              ;A*B=00,11,22,33,44,55,66,,77,88,99
          ADD   A,DPL           ;A=A+DPL
          MOV   DPL,A
          JC    HUW7
          SJMP  HUW8
HUW7:     INC   DPH
HUW8:     LCALL RUN_XX1         ;WRITE LINE 1
          MOV   BUFFER2,R1
          MOV   BUFFER1,R0
          RET
;NE_XT28:  SJMP  $
WLCD1:    MOV   A,#01H
          LCALL WR_COMM         ;CLEAR DISPLAY
          MOV   A,#0D0H         ;SET ADDRESS LINE 4
          MOV   DPTR,#MAINT4+48
          LCALL RUN_TT
          MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
          LCALL WRITE
          MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
          LCALL WR_COMM
          MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
          LCALL WRITE
          MOV   A,#90H          ;ADDRESS LINE 3
          LCALL WR_COMM
          MOV   R6,#16
          CLR   C
          MOV   A,BUFFER
          MOV   A,#01H
          MOV   B,#20H
          MUL   AB              ;A*B=32
          MOV   BUFFER,A        ;STORE RESULT IN BUFFER
          MOV   A,BUFFER2
          CLR   C
          SUBB  A,BUFFER        ;A=A(DPL)-BUFFER(32)
          MOV   DPL,A           ;RETURN DPL
          MOV   DPH,BUFFER1
          JC    HUW9
          SJMP  HUW10
HUW9:     DEC   DPH
HUW10:    SJMP  OPERATE1
OPERATE1: MOV   R1,#10010000B
          LCALL RUN_XZ1         ;WRITE NEXT LINE 3
```

```
        MOV   A,#0C0H       ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        MOV   A,BUFFER2     ;BUFFER2 IS DPL
        MOV   R1,A          ;FOR USE IN PRESS '#' AGAIN
        MOV   R0,BUFFER1    ;FOR USE IN PRESS '#' AGAIN
        CJNE  A,#30H,HENN3  ;WRITE NUMBER 1
        MOV   B,#00H
        SJMP  SUM4
HENN3:  CJNE  A,#40H,HENN4  ;WRITE NUMBER 2
        MOV   B,#01H
        SJMP  SUM4
HENN4:  CJNE  A,#50H,HE_NN5 ;WRITE NUMBER 3
        MOV   B,#02H
        SJMP  SUM4
HE_NN5: CJNE  A,#60H,HE_NN6 ;WRITE NUMBER 4
        MOV   B,#03H
        SJMP  SUM4
HE_NN6: CJNE  A,#70H,HE_NN7 ;WRITE NUMBER 5
        MOV   B,#04H
        SJMP  SUM4
HE_NN7: CJNE  A,#80H,HE_NN8 ;WRITE NUMBER 6
        MOV   B,#05H
        SJMP  SUM4
HE_NN8: CJNE  A,#90H,HE_NN9 ;WRITE NUMBER 7
        MOV   B,#06H
        SJMP  SUM4
HE_NN9: CJNE  A,#0A0H,HE_NN10 ;WRITE NUMBER 8
        MOV   B,#07H
        SJMP  SUM4
HE_NN10:CJNE  A,#0B0H,HE_NN11 ;WRITE NUMBER 9
        MOV   B,#08H
        SJMP  SUM4
HE_NN11:NOP
SUM4:   MOV   DPTR,#CALENDAR2
        MOV   A,#11
        MUL   AB            ;A*B=00,11,22,33,44,55,66,77,88,99
        ADD   A,DPL         ;A=A+DPL
        MOV   DPL,A
        JC    HUW11
        SJMP  HUW12
HUW11:  INC   DPH
HUW12:  LCALL RUN_XY1       ;WRITE LINE 2
        MOV   A,#80H        ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        MOV   A,BUFFER2     ;BUFFER2 IS DPL
        CJNE  A,#89H,NE_XT18 ;WRITE NUMBER 1
        MOV   B,#00H
        SJMP  SUM5
NE_XT18:CJNE  A,#94H,NE_XT19 ;WRITE NUMBER 2
        MOV   B,#01H
        SJMP  SUM5
NE_XT19:CJNE  A,#9FH,NE_XT20 ;WRITE NUMBER 3
        MOV   B,#02H
        SJMP  SUM5
NE_XT20:CJNE  A,#0AAH,NE_XT21 ;WRITE NUMBER 4
        MOV   B,#03H
        SJMP  SUM5
NE_XT21:CJNE  A,#0B5H,NE_XT22 ;WRITE NUMBER 5
        MOV   B,#04H
        SJMP  SUM5
NE_XT22:CJNE  A,#0C0H,NE_XT23 ;WRITE NUMBER 6
        MOV   B,#05H
        SJMP  SUM5
NE_XT23:CJNE  A,#0CBH,NE_XT24 ;WRITE NUMBER 7
        MOV   B,#06H
        SJMP  SUM5
NE_XT24:CJNE  A,#0D6H,NE_XT25 ;WRITE NUMBER 8
        MOV   B,#07H
        SJMP  SUM5
NE_XT25:CJNE  A,#0E1H,NE_XT26 ;WRITE NUMBER 9
        MOV   B,#08H
        SJMP  SUM5
NE_XT26:NOP
SUM5:   MOV   DPTR,#ROOM2
        MOV   A,#11
        MUL   AB            ;A*B=00,11,22,33,44,55,66,77,88,99
        ADD   A,DPL         ;A=A+DPL
        MOV   DPL,A
        JC    HUW13
        SJMP  HUW14
HUW13:  INC   DPH
HUW14:  LCALL RUN_XX1       ;WRITE LINE 1
        MOV   BUFFER2,R1
        MOV   BUFFER1,R0
        RET
WLCD2:  MOV   A,#01H
        LCALL WR_COMM       ;CLEAR DISPLAY
        MOV   A,#0D0H       ;SET ADDRESS LINE 4
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH        ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B  ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH        ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
```

```
         MOV   A,#90H          ;ADDRESS LINE 3
         LCALL WR_COMM
         MOV   R6,#16
         CLR   C
         MOV   A,BUFFER
         MOV   A,#01H
         MOV   B,#20H
         MUL   AB              ;A*B=32
         MOV   BUFFER,A        ;STORE RESULT IN BUFFER
         MOV   A,BUFFER2       ;BUFFER2 IS DPL OF RECTIME3
         CLR   C
         SUBB  A,BUFFER        ;A=A(DPL)-BUFFER(32)
         MOV   DPL,A           ;RETURN DPL
         MOV   DPH,BUFFER1
         JC    HUW21
         SJMP  HUW22
HUW21:   DEC   DPH
HUW22:   SJMP  OPERATE2
OPERATE2: MOV  R1,#10010000B
         LCALL RUN_XZ1         ;WRITE  LINE 3
         MOV   A,#0C0H         ;ADDRESS LINE 2
         LCALL WR_COMM
         MOV   R6,#11
         MOV   A,BUFFER2       ;BUFFER2 IS DPL
         MOV   R1,A            ;FOR USE IN PRESS '#' AGAIN
         MOV   R0,BUFFER1      ;FOR USEIN PRESS  '#' AGAIN
         CJNE  A,#0D0H,HE_NN12 ;WRITE NUMBER 1
         MOV   B,#00H
         SJMP  SUM10
HE_NN12: CJNE  A,#0E0H,HE_NN13 ;WRITE NUMBER 2
         MOV   B,#01H
         SJMP  SUM10
HE_NN13: CJNE  A,#0F0H,HE_NN14 ;WRITE NUMBER 3
         MOV   B,#02H
         SJMP  SUM10
HE_NN14: CJNE  A,#00H,HE_NN15 ;WRITE NUMBER 4
         MOV   B,#03H
         SJMP  SUM10
HE_NN15: CJNE  A,#10H,HE_NN16 ;WRITE NUMBER 5
         MOV   B,#04H
         SJMP  SUM10
HE_NN16: CJNE  A,#20H,HE_NN17 ;WRITE NUMBER 6
         MOV   B,#05H
         SJMP  SUM10
HE_NN17: CJNE  A,#30H,HE_NN18 ;WRITE NUMBER 7
         MOV   B,#06H
         SJMP  SUM10
HE_NN18: CJNE  A,#40H,HE_NN19 ;WRITE NUMBER 8
         MOV   B,#07H
         SJMP  SUM10
HE_NN19: CJNE  A,#50H,HE_NN20 ;WRITE NUMBER 9
         MOV   B,#08H
         SJMP  SUM10
HE_NN20: NOP
SUM10:   MOV   DPTR,#CALENDAR3
         MOV   A,#11
         MUL   AB              ;A*B=00,11,22,33,44,55,66,77,88,99
         ADD   A,DPL           ;A=A+DPL
         MOV   DPL,A
         JC    HUW23
         SJMP  HUW24
HUW23:   INC   DPH
HUW24:   LCALL RUN_XY1         ;WRITE LINE 2
         MOV   A,#80H          ;ADDRESS LINE 1
         LCALL WR_COMM
         MOV   R6,#11
         MOV   A,BUFFER2       ;BUFFER2 IS DPL
         CJNE  A,#0F7H,NE_XT28 ;WRITE NIMBER 1
         MOV   B,#00H
         SJMP  SUM11
NE_XT28: CJNE  A,#02H,NE_XT29  ;WRITE NUMBER 2
         MOV   B,#01H
         SJMP  SUM11
NE_XT29: CJNE  A,#0DH,NE_XT30  ;WRITE NUMBER 3
         MOV   B,#02H
         SJMP  SUM11
NE_XT30: CJNE  A,#18H,NE_XT31  ;WRITE NUMBER 4
         MOV   B,#03H
         SJMP  SUM11
NE_XT31: CJNE  A,#23H,NE_XT32  ;WRITE NUMBER 5
         MOV   B,#04H
         SJMP  SUM11
NE_XT32: CJNE  A,#2EH,NE_XT33  ;WRITE NUMBER 6
         MOV   B,#05H
         SJMP  SUM11
NE_XT33: CJNE  A,#39H,NE_XT34  ;WRITE NUMBER 7
         MOV   B,#06H
         SJMP  SUM11
NE_XT34: CJNE  A,#44H,NE_XT35  ;WRITE NUMBER 8
         MOV   B,#07H
         SJMP  SUM11
NE_XT35: CJNE  A,#4FH,NE_XT36  ;WRITE NUMBER 9
         MOV   B,#08H
         SJMP  SUM11
NE_XT36: NOP
SUM11:   MOV   DPTR,#ROOM3
         MOV   A,#11
         MUL   AB              ;A*B=00,11,22,33,44,55,66,77,88,99
         ADD   A,DPL           ;A=A+DPL
         MOV   DPL,A
```

```
        JC    HUW25
        SJMP  HUW26
HUW25:  INC   DPH
HUW26:  LCALL RUN_XX1        ;WRITE LINE 1
        MOV   BUFFER2,R1
        MOV   BUFFER1,R0
        RET
WLCD3:  MOV   A,#01H
        LCALL WR_COMM        ;CLEAR DISPLAY
        MOV   A,#80H         ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        CLR   A
        MOV   A,BUFFER
        CJNE  A,#06H,HENN7   ;PASS  4-6
        MOV   DPTR,#ROOM4
        MOV   DPL,#4AH
        SJMP  OPERATE3
HENN7:  CJNE  A,#09H,HENN8   ;PASS  7-9
        MOV   DPTR,#ROOM4
        MOV   DPL,#6BH
        SJMP  OPERATE3
HENN8:  MOV   DPTR,#ROOM4    ;PASS  10
        MOV   DPL,#8CH
        SJMP  OPERATE3
OPERATE3: LCALL RUN_XX1      ;WRITE NEXT LINE ON LCD
        MOV   A,DPL
        MOV   A,#0C0H        ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1        ;WRITE NEXT LINE ON LCD
        MOV   A,#90H         ;ADDRESS LINE 3
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1        ;WRITE NEXT LINE ON LCD
        MOV   A,#0D0H         ;SET ADDRESS LINE 4
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
WLCD4:  MOV   A,#01H
        LCALL WR_COMM        ;CLEAR DISPLAY
        MOV   A,#80H         ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        CLR   A
        MOV   A,BUFFER
        CJNE  A,#06H,HENN9   ;PASS  4-6
        MOV   DPTR,#ROOM5
        MOV   DPL,#0B8H
        SJMP  OPERATE4       ;WRITE 1-3
HENN9:  CJNE  A,#09H,HENN10  ;PASS  7-9
        MOV   DPH,#02H
        MOV   DPL,#0D9H
        SJMP  OPERATE4       ;WRITE 4-6
HENN10: MOV   DPH,#02H       ;PASS 10
        MOV   DPL,#0FAH
        SJMP  OPERATE4       ;WRITE 7-9
OPERATE4: LCALL RUN_XX1      ;WRITE NEXT LINE ON LCD
        MOV   A,DPL
        MOV   A,#0C0H        ;ADDRESS LINE 2
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1        ;WRITE NEXT LINE ON LCD
        MOV   A,#90H         ;ADDRESS LINE 3
        LCALL WR_COMM
        MOV   R6,#11
        LCALL RUN_XX1        ;WRITE NEXT LINE ON LCD
        MOV   A,#0D0H         ;SET ADDRESS LINE 4
        MOV   DPTR,#MAINT4+48
        LCALL RUN_TT
        MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
WLCD5:  MOV   A,#01H
        LCALL WR_COMM        ;CLEAR DISPLAY
        MOV   A,#80H         ;ADDRESS LINE 1
        LCALL WR_COMM
        MOV   R6,#11
        CLR   A
        MOV   A,BUFFER
        CJNE  A,#06H,HENN11  ;PASS  4-6
        MOV   DPTR,#ROOM6
        MOV   DPL,#26H
        SJMP  OPERATE5       ;WRITE 1-3
HENN11: CJNE  A,#09H,HENN12  ;PASS  7-9
        MOV   DPTR,#ROOM6
```

```
            MOV   DPL,#47H
            SJMP  OPERATE5          ;WRITE 4-6
HENN12:     MOV   DPTR,#ROOM6       ;PASS 10
            MOV   DPL,#68H
            SJMP  OPERATE5          ;WRITE 7-9
OPERATE5:   LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,DPL
            MOV   A,#0C0H         ;ADDRESS LINE 2
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#90H          ;ADDRESS LINE 3
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#0D0H          ;SET ADDRESS LINE 4
            MOV   DPTR,#MAINT4+48
            LCALL RUN_TT
            MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
            LCALL WR_COMM
            MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
            LCALL WRITE
            MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
            LCALL WR_COMM
            MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
            LCALL WRITE
            RET
WLCD6:      MOV   A,#01H
            LCALL WR_COMM         ;CLEAR DISPLAY
            MOV   A,#80H          ;ADDRESS LINE 1
            LCALL WR_COMM
            MOV   R6,#11
            CLR   A
            MOV   A,BUFFER
            CJNE  A,#06H,HENN13   ;PASS  4-6
            MOV   DPTR,#ROOM7
            MOV   DPL,#94H
            SJMP  OPERATE6        ;WRITE 1-3
HENN13:     CJNE  A,#09H,HENN14   ;PASS  7-9
            MOV   DPTR,#ROOM7
            MOV   DPL,#0B5H
            SJMP  OPERATE6        ;WRITE 4-6
HENN14:     MOV   DPTR,#ROOM7     ;PASS 10
            MOV   DPL,#0D6H
            SJMP  OPERATE6        ;WRITE 7-9
OPERATE6:   LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,DPL
            MOV   A,#0C0H         ;ADDRESS LINE 2
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#90H          ;ADDRESS LINE 3
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#0D0H          ;SET ADDRESS LINE 4
            MOV   DPTR,#MAINT4+48
            LCALL RUN_TT
            MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
            LCALL WR_COMM
            MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
            LCALL WRITE
            MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
            LCALL WR_COMM
            MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
            LCALL WRITE
            RET
WLCD7:      MOV   A,#01H
            LCALL WR_COMM         ;CLEAR DISPLAY
            MOV   A,#80H          ;ADDRESS LINE 1
            LCALL WR_COMM
            MOV   R6,#11
            CLR   A
            MOV   A,BUFFER
            CJNE  A,#06H,HENN15   ;PASS  4-6
            MOV   DPTR,#ROOM8
            MOV   DPL,#02H
            SJMP  OPERATE7        ;WRITE 1-3
HENN15:     CJNE  A,#09H,HENN16   ;PASS  7-9
            MOV   DPTR,#ROOM8
            MOV   DPL,#23H
            SJMP  OPERATE7        ;WRITE 4-6
HENN16:     MOV   DPTR,#ROOM8     ;PASS 10
            MOV   DPL,#44H
            SJMP  OPERATE7        ;WRITE 7-9
OPERATE7:   LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,DPL
            MOV   A,#0C0H         ;ADDRESS LINE 2
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#90H          ;ADDRESS LINE 3
            LCALL WR_COMM
            MOV   R6,#11
            LCALL RUN_XX1         ;WRITE NEXT LINE ON LCD
            MOV   A,#0D0H          ;SET ADDRESS LINE 4
            MOV   DPTR,#MAINT4+48
            LCALL RUN_TT
            MOV   A,#11010011B    ;TELL POSITION TO WRITE CHARACTER
            LCALL WR_COMM
```

```
        MOV   A,#7EH          ;WRITE ARROW UP ON WANTED POSITION
        LCALL WRITE
        MOV   A,#11011000B    ;TELL POSITION TO WRITE CHARACTER
        LCALL WR_COMM
        MOV   A,#7FH          ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL WRITE
        RET
;************SUM_USE SUB **************
;SHOW QUALITY USAGE
SUM_USE: MOV   A,#01H         ;CLEAR DISPLAY
        LCALL  WR_COMM
        MOV    DPTR,#MAINT7
        LCALL  DISPLAY
        MOV    A,#0D0H        ;SET ADDRESS LINE 4
        MOV    DPTR,#MAINT4+48
        LCALL  RUN_TT
        MOV    A,#11010011B   ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7EH         ;WRITE ARROW UP ON WANTED POSITION
        LCALL  WRITE
        MOV    A,#11011000B   ;TELL POSITION TO WRITE CHARACTER
        LCALL  WR_COMM
        MOV    A,#7FH         ;WRITE ARROW DOWN ON WANTED POSITION
        LCALL  WRITE
CHECKA:  MOV   DPTR,#PORTA_2
        MOVX   A,@DPTR
        JB     ACC.4,CHECKA
        ANL    A,#00001111B
CHECKA1: CJNE  A,#02H,CHECKA  ;KEY 3
        LJMP   START
; ********** RTIME SUB **********
; READ MK48T08 TIME
; OUT = R2,R3,R4  (HH,MM,SS)
; REG = A,R2,R3,R4
RTIME:   PUSH  DPH
        PUSH   DPL
        MOV    DPTR,#TIMBUF     ;SET R BIT
        MOVX   A,@DPTR
        SETB   ACC.6
        MOVX   @DPTR,A
        INC    DPTR             ;DPTR =1FF9H CHARBUF
        MOVX   A,@DPTR          ;READ SECOND
        MOV    R4,A
        MOV    TIM_SEC,R4
        INC    DPTR
        MOVX   A,@DPTR          ;READ MINUTE
        MOV    R3,A
        MOV    TIM_MIN,R3
        INC    DPTR
        MOVX   A,@DPTR          ;READ HOUR
        MOV    R2,A
        MOV    TIM_HOUR,R2
        MOV    DPTR,#TIMBUF     ;CLR R BIT
        MOVX   A,@DPTR
        CLR    ACC.6
        MOVX   @DPTR,A
        POP    DPL
        POP    DPH
        RET
; ********** RDATE SUB **********
; READ MK48T08 DATE
; OUT = R2,R3,R4,R5 (DD,MM,YY,WK)
; REG = A,R2,R3,R4,R5
RDATE:   PUSH  DPH
        PUSH   DPL
        MOV    DPTR,#TIMBUF     ;SET R BIT
        MOVX   A,@DPTR
        SETB   ACC.6
        MOVX   @DPTR,A
        MOV    DPTR,#TIMBUF+4
        MOVX   A,@DPTR          ;READ WEEK-DAY
        MOV    R5,A             ;
        INC    DPTR
        MOVX   A,@DPTR          ;READ DATE
        MOV    R2,A
        MOV    DATE_D,R2
        INC    DPTR
        MOVX   A,@DPTR          ;READ MONTH
        MOV    R3,A
        MOV    DATE_M,R3
        INC    DPTR
        MOVX   A,@DPTR          ;READ YEAR
        MOV    R4,A
        MOV    DATE_Y,R4
        MOV    DPTR,#TIMBUF     ;CLR R BIT
        MOVX   A,@DPTR
        CLR    ACC.6
        MOVX   @DPTR,A
        POP    DPL
        POP    DPH
        RET
; ********** WTIME SUB **********
; WRITE MK48T08 TIME
; IN  = R2,R3,R4  (HH,MM,SS)
; REG = A,R2,R3,R4
WTIME:   PUSH  DPH
        PUSH   DPL
        MOV    DPTR,#TIMBUF     ;SET W BIT
        MOVX   A,@DPTR
        SETB   ACC.7
```

```
        MOVX    @DPTR,A
        POP     DPL
        INC     DPTR
        MOV     R4,SEC
        MOV     A,R4
        CLR     ACC.7
        MOVX    @DPTR,A             ;SECOND
        INC     DPTR
        MOV     R3,MIN
        MOV     A,MIN
        MOV     A,R3
        CLR     ACC.7
        MOVX    @DPTR,A             ;MINUTE
        INC     DPTR
        MOV     R2,HOUR
        MOV     A,R2
        ANL     A,#3FH
        MOVX    @DPTR,A             ;HOUR
        MOV     DPTR,#TIMBUF        ;CLR W BIT
        MOVX    A,@DPTR
        CLR     ACC.7
        POP     DPH
        RET
; ********** WDATE SUB **********
; WRITE MK48T08 DATE
; IN  = R2,R3,R4,R5 (DD,MM,YY,WK)
; REG = A,R2,R3,R4,R5
WDATE:  PUSH    DPH
        PUSH    DPL
        MOV     DPTR,#TIMBUF        ;SET W BIT
        MOVX    A,@DPTR
        SETB    ACC.7
        MOVX    @DPTR,A
        MOV     DPTR,#TIMBUF+4
        MOV     A,R5
        ANL     A,#07H
        MOVX    @DPTR,A             ;WEEK-DAY
        INC     DPTR
        MOV     R2,DAY
        MOV     A,R2
        ANL     A,#3FH
        MOVX    @DPTR,A             ;DATE
        INC     DPTR
        MOV     R3,MONTH
        MOV     A,R3
        ANL     A,#3FH
        MOVX    @DPTR,A             ;MONTH
        INC     DPTR
        MOV     R4,YEAR
        MOV     A,R4
        MOVX    @DPTR,A             ;YEAR
        MOV     DPTR,#TIMBUF        ;CLR W BIT
        MOVX    A,@DPTR
        CLR     ACC.7
        MOVX    @DPTR,A
        POP     DPL
        POP     DPH
        RET
WCHAR:  PUSH    DPH
        PUSH    DPL
        MOV     DPTR,#CHARBUF       ;SET W BIT
        MOVX    A,@DPTR
        SETB    ACC.7
        MOVX    @DPTR,A
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        INC     DPTR
        MOVX    A,@DPTR
        LCALL   WRITE
        MOV     DPTR,#CHARBUF
        MOVX    A,@DPTR
        CLR     ACC.7           ;   CLR W BIT
        MOVX    @DPTR,A
        POP     DPL
        POP     DPH
        RET
MAINT1: DB      " PROJECT  PABX  "
        DB      "            4A  "
        DB      "                "
        DB      "ENTER ROOM NO:  "
MAINT2: DB      "MAIN_MENU(ROOM )"
```

```
         DB      "1.SEE LEAVE NO  "
         DB      "2.SEE SUM USE   "
         DB      "3.BACK 1ST PAGE "
MAINT4:  DB      "                "
         DB      "                "
         DB      "                "
         DB      "* = ,# = ,3=MAIN"
MAINT5:  DB      " LEAVING NUMBER "
         DB      "      ZONE      "
         DB      " SYSTEM BEGIN   "
         DB      "    LINKING     "
MAINT6:  DB      " LEAVING NUMBER "
         DB      "   OPERATING    "
         DB      "     ROOM1      "
         DB      "                "
MAINT7:  DB      "  THIS ZONE IS  "
         DB      "      UNDER     "
         DB      "  CONSTRUCTION  "
         DB      "                "
; ********** WR_COMM SUB ***** (ANT-31PJ)
; LCD WRITE INSTRUCTION (RS=0)
; IN  = A
; REG = A
WR_COMM: PUSH    DPH
         PUSH    DPL
         MOV     DPTR,#COM_PORT
         MOVX    @DPTR,A
         MOV     DPTR,#RD_BUSY
BUSY1:   MOVX    A,@DPTR          ;WAIT FOR BF=0
         JB      ACC.7,BUSY1
         POP     DPL
         POP     DPH
         RET
; ********** WRITE SUB ****** (ANT-31PJ)
; LCD WRITE DATA (RS=1)
; IN  = A
; REG = A
WRITE:   PUSH    DPH
         PUSH    DPL
         MOV     DPTR,#DAT_PORT
         MOVX    @DPTR,A          ;WRITE SHOW ON LCD
         MOV     DPTR,#RD_BUSY
BUSY2:   MOVX    A,@DPTR          ;WAIT FOR BF=0
         JB      ACC.7,BUSY2
         POP     DPL
         POP     DPH
         RET
; ********** DISPLAY SUB ***** (ANT-31PJ)
; LOAD PMEM TO LCD-MODULE (DMC202)
; IN  = DPTR  START BLOCK
DISPLAY: MOV     A,#80H  ;SET ADDRESS LINE 1
         LCALL   RUN_TT
         MOV     A,#0C0H ;SET ADDRESS LINE 2
         LCALL   RUN_TT
         MOV     A,#90H  ;SET ADDRESS LINE 3
         LCALL   RUN_TT
         MOV     A,#0D0H ;SET ADDRESS LINE 4
         LCALL   RUN_TT
         RET
RUN_TT:  LCALL   WR_COMM     ;LOAD ONE LINE
         MOV     R6,#16      ;16 CHAR.
RUN_XX:  CLR     A
         LCALL   WRITE       ;WRITE DATA
         INC     DPTR
         RET
;******** SUB RUN_8C *************
RUN_8C1: LCALL   WR_COMM
         MOV     R6,#8       ;8*2 CHAR
RUN_8C2: CLR     A
         MOV     A,@R1
         LCALL   WRITE
BB4:     INC     R1
BB2:     DJNZ    R6,RUN_8C2
         RET
RUN_8C3: LCALL   WR_COMM
         MOV     R6,#8
RR1:     CLR     A
         MOV     A,@R1
         LCALL   WRITE
BB28:    INC     R1
BB34:    DJNZ    R6,RR1
         RET
RUN_8C4: PUSH    DPH
         PUSH    DPL
         LCALL   WR_COMM
         MOV     R6,#11
RR2:     CLR     A
         LCALL   WRITE
         POP     DPL
         POP     DPH
         INC     DPTR
         MOVC    A,@A+DPTR
         MOV     R5,A
         DJNZ    R6,RR2
         RET
RUN_W:   LCALL   WR_COMM
         MOV     R6,#3
RW1:     CLR     A
         MOV     A,@R1
         LCALL   WRITE
```

```
        INC   R1
        DJNZ  R6,RW1
        RET
;******** SUB RUN_8R1 ************
RUN_8R1: MOV   A,#0C0H       ;SET ADDRESS LINE 2
        LCALL RUN_8C1
        RET
;******** SUB RUN_8R2 ************
RUN_8R2: MOV   A,#90H
        LCALL RUN_8C3
        RET
;******** SUB RUN_8R3 ************
RUN_8R3: MOV   A,#90H        ;SET ADDRESS LINE 2
        LCALL RUN_8C4
        RET
RUN_WEEK: MOV   A,#10011011B  ;TELL POSITION TO WRITE CHARACTER
        LCALL RUN_W
        RET
;******** SUB DECODE *************
DECODE:  MOV   BUF_TIM+2,#':'
        MOV   BUF_TIM+5,#':'
        MOV   A,TIM_HOUR
        LCALL HTOA
        MOV   BUF_TIM,R2
        MOV   BUF_TIM+1,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP1,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP2,A
        MOV   A,TIM_MIN
        LCALL HTOA
        MOV   BUF_TIM+3,R2
        MOV   BUF_TIM+4,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP3,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP4,A
        MOV   A,TIM_SEC
        LCALL HTOA
        MOV   BUF_TIM+6,R2
        MOV   BUF_TIM+7,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP5,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP6,A
        RET
;******** SUB DECODE2 ************
DECODE2: MOV   BUF_DATE+2,#'/'
        MOV   BUF_DATE+5,#'/'
        MOV   A,DATE_D
        LCALL HTOA
        MOV   BUF_DATE,R2
        MOV   BUF_DATE+1,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP7,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP8,A
        MOV   A,DATE_M
        LCALL HTOA
        MOV   BUF_DATE+3,R2
        MOV   BUF_DATE+4,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP9,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP10,A
        MOV   A,DATE_Y
        LCALL HTOA
        MOV   BUF_DATE+6,R2
        MOV   BUF_DATE+7,R3
        MOV   A,R2
        LCALL ATOHEX
        MOV   HAP11,A
        MOV   A,R3
        LCALL ATOHEX
        MOV   HAP12,A
        RET
;******** HTOA SUB ************
;CONVERT HEX TO ASCII
;IN  = A
;OUT = R2,R3
;REG = A,R2,R3
HTOA:    PUSH  ACC
        SWAP  A              ;EXCHANGE  4 BIT LOWWER AND 4 BIT UPPER
        LCALL HTOAS
        MOV   R2,A
        POP   ACC
        LCALL HTOAS
        MOV   R3,A
        RET
HTOAS:   ANL   A,#0FH
```

```
        CJNE  A,#0AH,$+3
        JNC   HTOAS1
        ORL   A,#30H
        RET
HTOAS1: SUBB  A,#9
        ORL   A,#40H
        RET
;*FUNTION LINK TO CONTROLLER1 TO RECEIVE LEAVE NUMBER*
RECE_N: MOV     RR7,#00H
        MOV     RR6,#0BH          ;
        LCALL   DELAY3M           ;DELAY  3 MSEC
        MOV     A,#01H
        LCALL   WR_COMM           ;CLEAR DISPLAY
        MOV     DPTR,#MAINT5
        LCALL   DISPLAY      ;WRITE MESSAGE ON LCD DISPLAY (MAINT5)
        LCALL   DELAY3M
        LCALL   DELAY3M
        MOV     DPTR,#PORTC_1
        MOV     A,#30H
        MOVX    @DPTR,A      ;SEND ACKNOWEDGE = 30H OUT OF PORT C TO UC1
        LCALL   DELAY3M
        MOV     DPTR,#PORTB_1
        MOVX    A,@DPTR      ;RECEIVE  NUMBER ROOM(1-8) FROM CONTROLLER 1
        MOV     BUFFER,A     ;BUFFER STORE NUMBER ROOM
        SJMP    R_LV0
;*CHECK A IS ANY NUMBER ROOM WITHIN 1-8*
R_LV0:  CJNE    A,#01H,R_LV1
        MOV     DPTR,#ROOM1       ;A=01H IS ROOM1
        SJMP    RECE
R_LV1:  CJNE    A,#02H,R_LV2
        MOV     DPTR,#ROOM2       ;A=02H IS ROOM2
        SJMP    RECE
R_LV2:  CJNE    A,#03H,R_LV3
        MOV     DPTR,#ROOM3       ;A=03H IS ROOM3
        SJMP    RECE
R_LV3:  CJNE    A,#04H,R_LV4
        MOV     DPTR,#ROOM4       ;A =04H IS ROOM4
        SJMP    RECE
R_LV4:  CJNE    A,#05H,R_LV5
        MOV     DPTR,#ROOM5       ;A=05H IS ROOM5
        SJMP    RECE
R_LV5:  CJNE    A,#06H,R_LV6
        MOV     DPTR,#ROOM6       ;A=06H IS ROOM6
        SJMP    RECE
R_LV6:  CJNE    A,#07H,R_LV7
        MOV     DPTR,#ROOM7       ;A=07H IS ROOM7
        SJMP    RECE
R_LV7:  CJNE    A,#08H,R_LV8
        MOV     DPTR,#ROOM8       ;A=08H IS ROOM8
        SJMP    RECE
R_LV8:  NOP
;*RECEIVE ,STORE INDIVIDUAL LEAVE NUMBER IN MEMORY POSITION *
RECE:   PUSH    DPH
        PUSH    DPL
        SWAP    A         ;CHANGE 4 BIT UPPER ,4 BIT LOWER (10H - 80H)
        MOV     DPTR,#PORTC_1
        MOVX    @DPTR,A           ;SEND 10H - 80H GO TO UC1
        LCALL   DELAY3M
        POP     DPL
        POP     DPH
;*BEGIN FIND POSITION THAT IS 0FFH AND SLIDE NUMBER *
        MOV     R1,#00
        MOV     R6,#0AH
RECE1:  MOVX    A,@DPTR
        MOV     R0,#11
        CJNE    A,#0FFH,R_LV9     ;A NOT EQUAL  #0FFH GO TO R_LV9
        SJMP    R_LV10
R_LV9:  MOV     A,DPL             ;|
        CLR     C                 ;|
        ADD     A,R0              ;|
        MOV     DPL,A             ; > SLIDE DPTR TO NEXT NUMBER IN SAME ROOM
        JC      YAHOO1            ;|
        INC     R1                ;|
        MOV     A,R1              ;|
        CJNE    A,#0AH,YAHOO2     ;|
        MOV     DPTR,#ROOM1       ;|  A = #0AH THAT IS WRITE  TO SEQUENCE 10 AN
;PREPARE WRITE SEQUENCE 1 NEW
        SJMP    R_LV10            ;|
YAHOO1: INC     DPH               ;|
        INC     R1
        MOV     A,R1
        CJNE    A,#0AH,YAHOO2
        MOV     DPTR,#ROOM1
        SJMP    R_LV10
YAHOO2: DEC     R6
        SJMP    RECE1
R_LV10: LCALL   DELAY3M           ;BEGIN SAVE NUMBER
        LCALL   DELAY3M
        LCALL   DELAY3M
        LCALL   DELAY3M
        LCALL   DELAY3M
        LJMP    LD_NU             ;RECEIVE INDIVIDUAL NUMBER FORM PORTB
YAHOO3: MOVX    @DPTR,A           ;STORE INDIVIDUAL NUMBER IN MEMORY POSITION
        INC     RR7
        INC     DPTR
        DJNZ    RR6,$+9           ;RR6 =/ 00 JUMP TO ACK: [3]
        MOV     R5,DPH            ;[2]
        MOV     R2,DPL            ;[2]
        SJMP    END_REC2          ;[2]
ACK:    PUSH    DPH
```

```
        PUSH    DPL
        MOV     DPTR,#PORTC_1
        MOV     A,RR7
        MOVX    @DPTR,A    ;SEND ACKNOWLEDGE INFORM TO CONTROLLER1
;THAT RECEIVE NUMBER
        LCALL   DELAY3M
        POP     DPL
        POP     DPH
        SJMP    R_LV10
END_REC:SJMP    DISP
END_REC2:MOV    DPTR,#PORTC_1
        MOV     A,RR7
        MOVX    @DPTR,A            ;SEND ACKNOWLEDGE =0BH (LEAVE NUMBER =11 )
DISP:   MOV     A,#00H
        MOV     DPTR,#PORTC_1
        MOVX    @DPTR,A             ;SEND 00H GO TO PORTC_1 TO UC1
        MOV     A,#01H
        LCALL   WR_COMM            ;CLEAR DISPLAY
        MOV     DPTR,#MAINT6
        LCALL   DISPLAY            ;WRITE MESSAGE MAINT6 ON LCD DISPLAY
        LCALL   DELAY3
        LCALL   DELAY3
        LCALL   DELAY3
        LCALL   DELAY3M
        LCALL   DELAY3M          ;    ___
        MOV     A,R2             ;| |
        ADD     A,#05H           ;|  > DPL=DPL+#05H
        MOV     DPL,A            ;| |___
        MOV     P1,A             ;<-----------------------------
        MOV     A,R5             ; >| SLIDE FROM ROOM TO CALENDAR
        ADD     A,#03H           ;|  > DPH=DPH+#03H
        MOV     DPH,A            :| |___
        MOV     A,HAP7
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,HAP8
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,#'/'
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP9
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,HAP10
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,#'/'
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP11
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,HAP12
        MOVX    @DPTR,A
        LCALL   DELAY3
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR          ;DPTR =041BH ,0426H,0431H ,043CH ,0447H, 0452H ,045
                              ;,0468H ,0473H ,047EH
        MOV     R5,DPH
        MOV     R2,DPL
        CJNE    R2,#1BH,HELLO5  ;R2=1BH RUN NEXT LINE
        MOV     R4,#00H
        LJMP    HELLO6
HELLO5: CJNE    R2,#26H,HELLO7  ;R0=26H RUN NEXT LINE
        MOV     R4,#01H
        LJMP    HELLO6
HELLO7: CJNE    R2,#31H,HELLO8  ;R0=31H RUN NEXT LINE
        MOV     R4,#02H
        LJMP    HELLO6
HELLO8: CJNE    R2,#3CH,HELLO9
        MOV     R4,#03H
        LJMP    HELLO6
HELLO9: CJNE    R2,#47H,HELLO10
        MOV     R4,#04H
        LJMP    HELLO6
HELLO10:CJNE    R2,#52H,HELLO11
        MOV     R4,#05H
        LJMP    HELLO6
HELLO11:CJNE    R2,#5DH,HELLO12
        MOV     R4,#06H
        LJMP    HELLO6
HELLO12:CJNE    R2,#68H,HELLO13
        MOV     R4,#07H
        LJMP    HELLO6
HELLO13:CJNE    R2,#73H,HELLO14
        MOV     R4,#08H
```

```
        LJMP    HELLO6
HELLO14: CJNE   R2,#7EH,HELLO15
        MOV     R4,#09H
        LJMP    HELLO6
HELLO15: NOP
HELLO6: MOV     A,DPL           ;| |
        CLR     C               ;| |
        ADD     A,#65H          ;|  > DPL=DPL+#65H
        MOV     DPL,A           ;| |___
        MOV     A,DPH           ;| |
        ADDC    A,#03H          ;|  > DPH=DPH+#03H
        MOV     DPH,A           ;| |___
        MOV     A,#5            ;  > SLIDE CALENDAR TO RECTIME
        MOV     B,R4            ;|
        MUL     AB              ;| A=0,5,10,15,20,25,30,35,40,45
        CLR     C               ;|
        ADD     A,DPL           ;||
        MOV     DPL,A           ;|| DPL=DPL+0,5,10,15,20,25,30,35,40,45
        JC      DISK3           ;|
        SJMP    DISK4           ;|
DISK3:  INC     DPH
DISK4:  MOV     A,HAP1      ;DPTR=0780H,790H,7A0H,7B0H,7C0H,
;7D0H,7E0H,7F0H,800H,810H
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP2
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#':'
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP3
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP4
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#':'
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP5
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,HAP6
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR
        MOV     A,#0FFH
        MOVX    @DPTR,A
        INC     DPTR            ;DPTR=0790H
        LJMP    START
LD_NU:  PUSH    DPH
        PUSH    DPL
        MOV     DPTR,#PORTB_1
        MOVX    A,@DPTR         ;RECEIVE INDIVIDUAL NUMBER FROM PORTB
        CJNE    A,#0CH,LOOP2    ;A NOT EQUAL'#' GO TO LOOP2
        POP     DPL
        POP     DPH
HELLO:  MOV     A,#0FFH
        MOVX    @DPTR,A         ;STORE A=0FFH IN MEMORY   DPTR=ROOM1
        INC     DPTR
        DJNZ    RR6,HELLO
        MOV     R5,DPH
        MOV     R2,DPL
        MOV     DPTR,#PORTC_1
        MOV     A,#0F0H
        MOVX    @DPTR,A         ;SEND ACKNOWEDGE =F0H TO UC1
        LJMP    END_REC
LOOP2:  POP     DPL
        POP     DPH
        LJMP    YAHOO3
ACK1:   MOV     A,POINTER5
        SWAP    A               ;CHANGE 4 BIT UPPER 4 BIT LOWER =01 ,02H
        MOV     DPTR,#PORTC_1
        MOVX    @DPTR,A         ;SEND 01H OR 02H TO UC1
        LCALL   DELAY90
        LCALL   DELAY90
        MOV     A,#00H
        MOV     DPTR,#PORTC_1
        MOVX    @DPTR,A         ;SEND  00H TO UC1
```

```
          RET
ATOHEX:   CLR   C
          SUBB  A,#'0'
          JC    ERR
          CJNE  A,#9+1,NOTEQQ
          SJMP  ERR
NOTEQQ:   JC    DONE
ERR:      MOV   A,#0FFH
DONE:     RET
;******* SUB DELAYS **********
; DELAY SUBROUTINE
; IN  = R2
; REG = R2,R3,R4
DELAY:    MOV   R3,#0
DELAY1:   MOV   R4,#0
          DJNZ  R4,$
          DJNZ  R3,DELAY1
          DJNZ  R2,DELAY
          RET
DELAY5:   MOV   R3,#40H
DELAY55:  MOV   R4,#00H
          DJNZ  R4,$
          DJNZ  R3,DELAY55
          RET
;**********DELAY (2 SEC)***********
DELAY2:   MOV   R0,#10H    ;[1]
D21:      MOV   R1,#00H    ;[1]
D22:      MOV   R2,#00H    ;[1]
          DJNZ  R2,$       ;[2]
          DJNZ  R1,D22     ;[2]
          DJNZ  R0,D21     ;[2]
          RET
;********DELAY (3 msec)**********
DELAY3M:  MOV   R0,#01H
D3M:      MOV   R1,#0F5H
          DJNZ  R1,$
          DJNZ  R0,D3M
          RET
;*********DELAY(1 SEC)************
DELAY3:   MOV   R0,#09H     ;[1]
          MOV   R1,#00H     ;[1]
D23:      MOV   R3,#00H     ;[1]
          DJNZ  R3,$        ;[2]
          DJNZ  R1,D23      ;[2]
          DEC   R0          ;[1]
          MOV   A,R0        ;[1]
          CJNE  A,#00H,D23  ;[2]
          RET               ;[2]
;******** DELAY (90 mSEC)***********
DELAY90:  MOV   R0,#0B1H
D9:       MOV   R1,#0FFH
          DJNZ  R1,$
          DJNZ  R0,D9
          RET
END
```

โปรแกรมในการตรวจจับสัญญาณไม่ว่าง (busy tone) และสัญญาณเรียกกลับ (ring back tone)

โปรแกรมลงใน PIC12C509

```
;FILENAME    DETECT_RBT&BT.ASM
;HARDWARE    PIC12C509 (CRYSTAL 32.768 kHz)
;PROGRAMER   MR. DECHO INTARARUJIKUL
;DATE        10 FEBRUARY 1998
;
GPIO        EQU    06H  ; PORT OF PIC12C509
W           EQU    00H
COUNT1      EQU    0CH
COUNT2      EQU    0DH

;*****************************************
            ORG    00H
START
            MOVLW  04H
            TRIS   GPIO      ;GPIO='0000 0100'
            BCF    GPIO,0    ;OUT GPIO.0='0' INITIAL RBT DETECTION OUTPUT.
            BCF    GPIO,1    ;OUT GPIO.1='0' INITIAL BT DETECTION OUTPUT.

;********* SUB DETECT RING BACK TONE ************
CHECK
            BTFSC  GPIO,2    ;loop for checking tone from trunk lines
            GOTO   CHECK
            GOTO   FD_TONE

;************** SUB TRUNK1  *****************
FD_TONE
            BTFSC  GPIO,2    ;found tone
            GOTO   CHECK
C_BT        CALL   DELAY0.8
            BTFSS  GPIO,2
            GOTO   D_RBT
            GOTO   D_BT

;************* SUB DETECT BUSY TONE   ******************
D_BT        CALL   DELAY0.5
            BTFSC  GPIO,2
            GOTO   CHECK
            CALL   DELAY0.5
            BTFSS  GPIO,2
            GOTO   CHECK
ON_BT       BSF    GPIO,1    ;found busy tone
            CALL   DELAY0.5
CHK_BT      BTFSC  GPIO,2
            GOTO   OFF_BT
            CALL   DELAY0.5
            BTFSS  GPIO,2
            GOTO   OFF_BT
            CALL   DELAY0.5
            GOTO   CHK_BT
OFF_BT      BCF    GPIO,1    ;busy tone disappear
            GOTO   CHECK
;************* SUB DETECT RING BACK TONE   *****************
D_RBT       CALL   DELAY0.5
            BTFSS  GPIO,2
            GOTO   CHECK
            CALL   DELAY3.4
            BTFSS  GPIO,2
            GOTO   CHECK
            CALL   DELAY1.1
            BTFSC  GPIO,2
            GOTO   CHECK
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

```
          CALL      DELAY0.5
          BTFSS     GPIO,2
          GOTO      CHECK

ON_RBT    BSF       GPIO,0    ;found ring back tone
          CALL      DELAY3.4
CHK_RBT   BTFSS     GPIO,2
          GOTO      OFF_RBT
          CALL      DELAY1.1
          BTFSC     GPIO,2
          GOTO      OFF_RBT
          CALL      DELAY0.5
          BTFSS     GPIO,2
          GOTO      OFF_RBT
          CALL      DELAY3.4
          GOTO      CHK_RBT
OFF_RBT   BCF       GPIO,0      ;ring back tone disappear
          GOTO      CHECK

;*******  SUB DELAY 3.4 SEC  *******
DELAY3.4
          MOVLW     .83           ; 125usec * {4+83(108(3)+4)+2}
          MOVWF     COUNT1
L31       MOVLW     .108
          MOVWF     COUNT2
L32       DECFSZ    COUNT2,1
          GOTO      L32
          DECFSZ    COUNT1,1
          GOTO      L31
          RETLW     00

;*********  SUB DELAY 1.1 SEC  *********
DELAY1.1
          MOVLW     .71           ; 125usec * {4+71(40(3)+4)+2}
          MOVWF     COUNT1
L11       MOVLW     .40
          MOVWF     COUNT2
L12       DECFSZ    COUNT2,1
          GOTO      L12
          DECFSZ    COUNT1,1
          GOTO      L11
          RETLW     00

;*********  SUB DELAY 0.8 SEC  *********
DELAY0.8
          MOVLW     .20           ; 125usec * {4+20(105(3)+4)+2}
          MOVWF     COUNT1
L81       MOVLW     .105
          MOVWF     COUNT2
L82       DECFSZ    COUNT2,1
          GOTO      L82
          DECFSZ    COUNT1,1
          GOTO      L81
          RETLW     00
;*********  SUB DELAY 0.5 SEC  *********
DELAY0.5
          MOVLW     .13           ; 125usec * {4+13(101(3)+4)+2}
          MOVWF     COUNT1
LP51      MOVLW     .101
          MOVWF     COUNT2
LP52      DECFSZ    COUNT2,1
          GOTO      LP52
          DECFSZ    COUNT1,1
          GOTO      LP51
          RETLW     00
END
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

โปรแกรมในการตรวจจับสัญญาณกระดิ่ง (Ringing Signal)

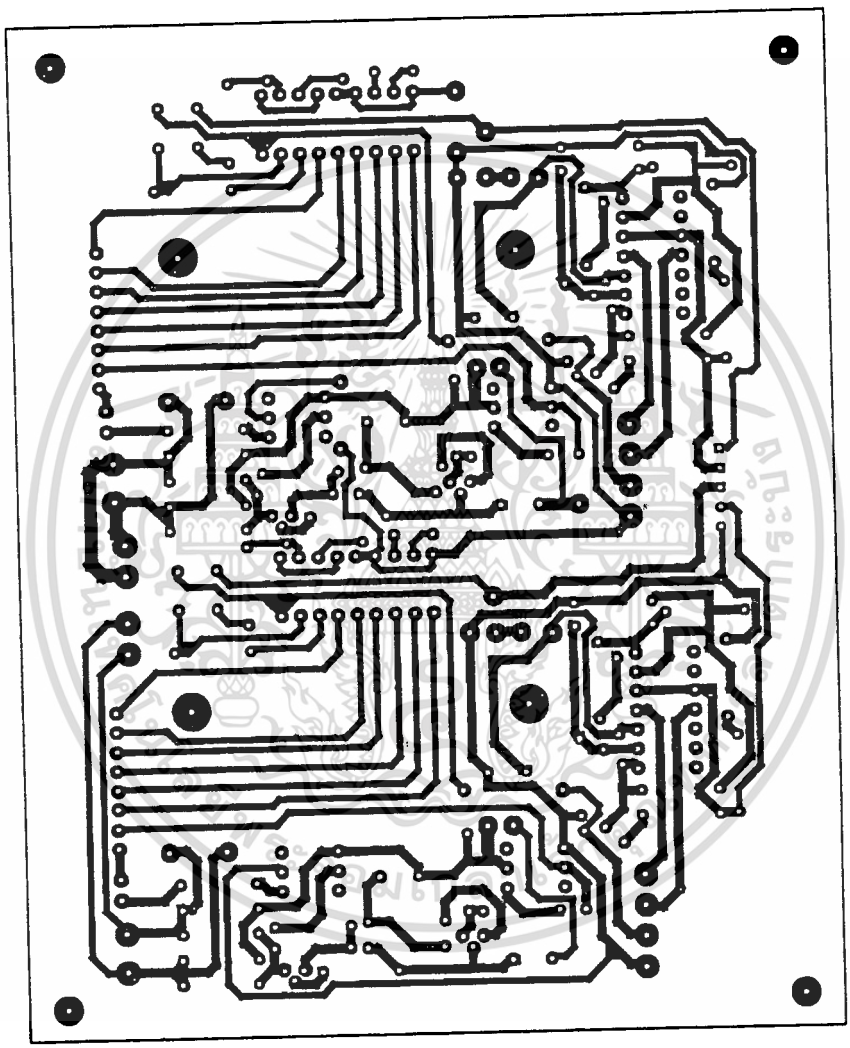
โปรแกรมลงใน PIC12C509

```
;FILENAME    DETECT_RING.ASM
;HARDWARE    PIC12C509 (CRYSTAL 32.768 kHz)
;PROGRAMER   MR. DECHO INTARARUJIKUL
;DATE        19 OCTOBER 1997
;
GPIO        EQU   06H  ; PORT OF PIC12C509
W           EQU   00H
COUNT1      EQU   0CH
COUNT2      EQU   0DH
DINT        EQU   10H
;*****************************************
        ORG     00H
START
        MOVLW   06H
        TRIS    GPIO    ;GPIO='0000 0110'
        BSF     GPIO,0  ;SET GPIO.0='1'  FOR PIN INTERRUPT

;********* LOOP DETECT RINGING ************
CHECK
        BTFSC   GPIO,2
        GOTO    CHECK1
        GOTO    SUB_TRUNK1
CHECK1  BTFSC   GPIO,1
        GOTO    CHECK
        GOTO    SUB_TRUNK2
;************** SUB TRUNK1  *****************
SUB_TRUNK1
        BTFSC   GPIO,2
        GOTO    CHECK
        CALL    DELAY2.6
        BTFSS   GPIO,2
        GOTO    CHECK
        CALL    DELAY2.6
        BTFSC   GPIO,2        ;ENSURE IT IS REAL RINGING SIGNAL (CHECK 2nd)
        GOTO    CHECK
        CALL    SUB_INT1
        GOTO    CHECK
;************  SUB_INT1  ******************
SUB_INT1
        BCF     GPIO,0
        MOVLW   .100
        MOVWF   DINT1
SUB1    DECFSZ  DINT,1         ;DELAY FOR INT PULSE 40 msec FOR TRUNK 1
        GOTO    SUB1
        BSF     GPIO,0
        RETLW   00
;************** SUB TRUNK2  ******************
SUB_TRUNK2
        BTFSC   GPIO,1
        GOTO    CHECK
S2      CALL    DELAY2.6
        BTFSS   GPIO,1
        GOTO    CHECK
        CALL    DELAY2.6
        BTFSC   GPIO,1        ;ENSURE IT IS REAL RINGING SIGNAL (CHECK 2nd)
        GOTO    CHECK
        DECFSZ  N_RING1,1
        GOTO    S2
        CALL    SUB_INT2
        GOTO    CHECK
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

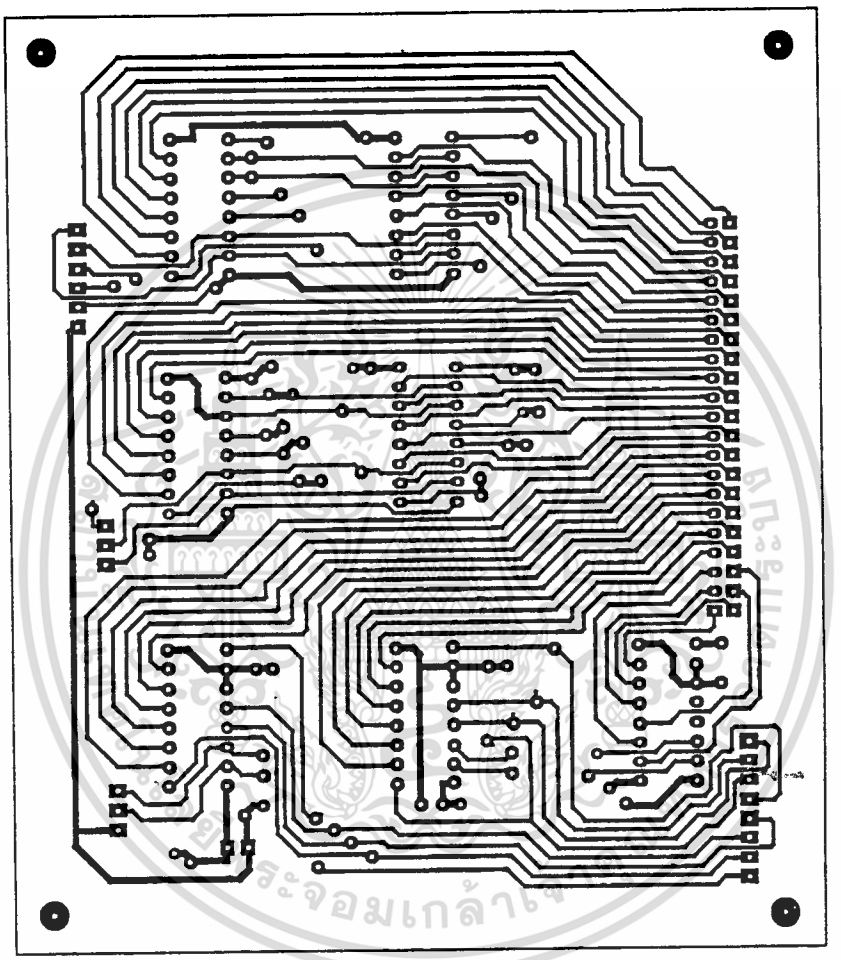
```
;************  SUB_INT2  *****************
SUB_INT2
        BCF     GPIO,0
        MOVLW   .10
        MOVWF   DINT
SUB2    DECFSZ  DINT,1          ;DELAY FOR INT PULSE 4 msec FOR TRUNK 2
        GOTO    SUB2
        BSF     GPIO,0
        RETLW   00

;*******  SUB DELAY 2.6 SEC  *******
DELAY2.6
        MOVLW   .69
        MOVWF   COUNT1
L21     MOVLW   .100
        MOVWF   COUNT2
L22     DECFSZ  COUNT2,1
        GOTO    L22
        DECFSZ  COUNT1,1
        GOTO    L21
        RETLW   00

END
```

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



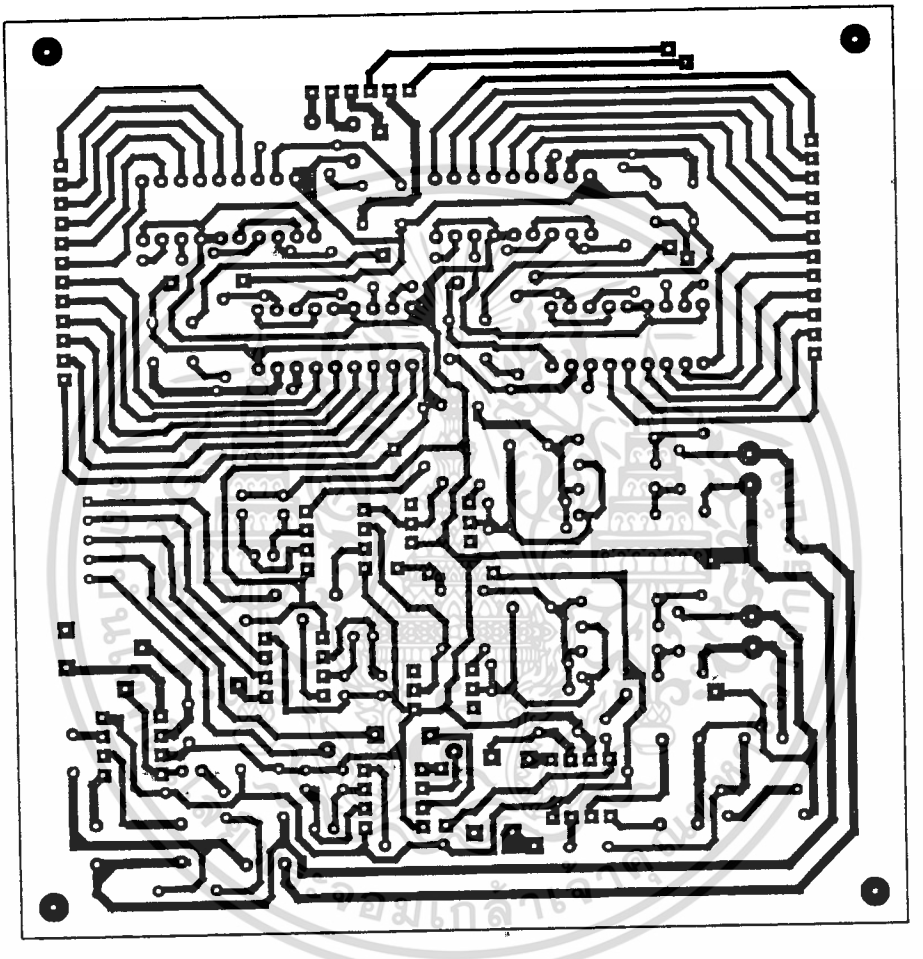
ลายวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายนอก

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



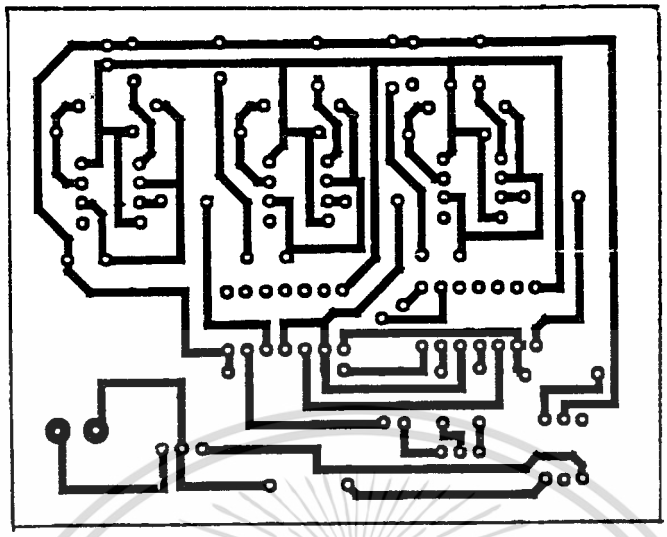
ลายวงจรส่วนคีรอสพอยท์สวิทช์

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

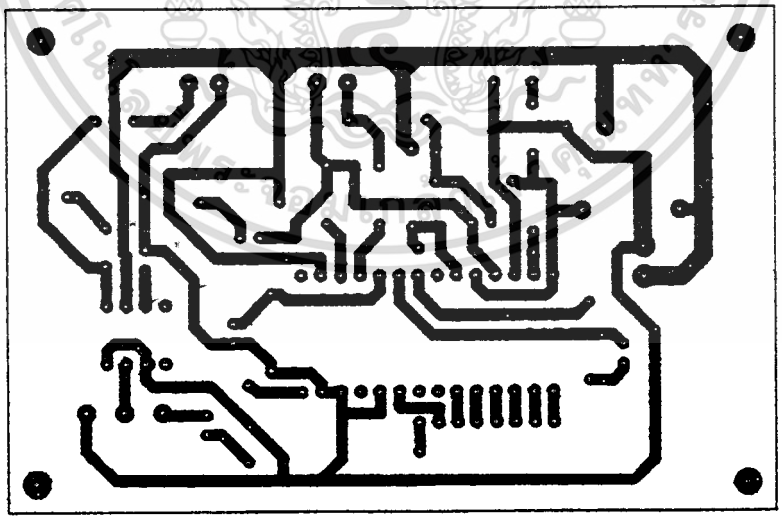


ลายวงจรส่วนตรวจจับสัญญาณเสียงและถอดรหัสสัญญาณ DTMF

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

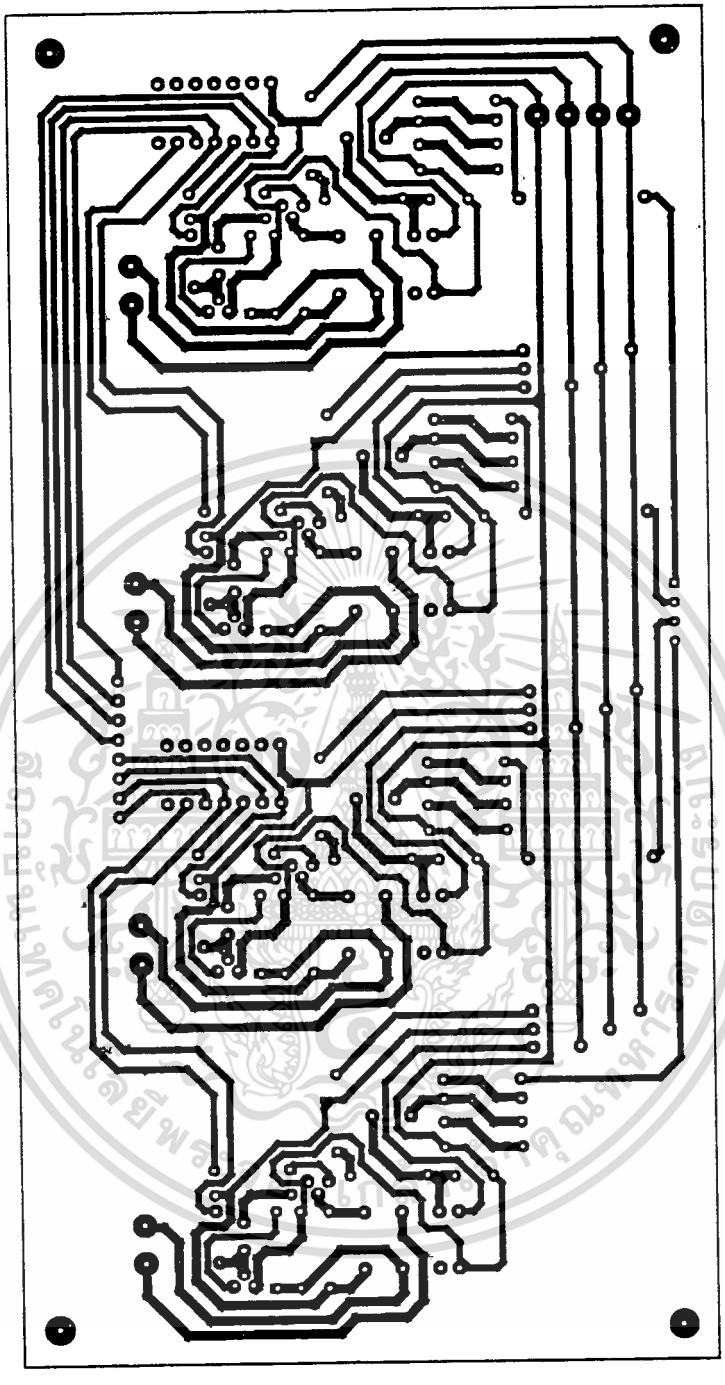


ลายวงจรส่วนกำเนิดสัญญาณเสียง



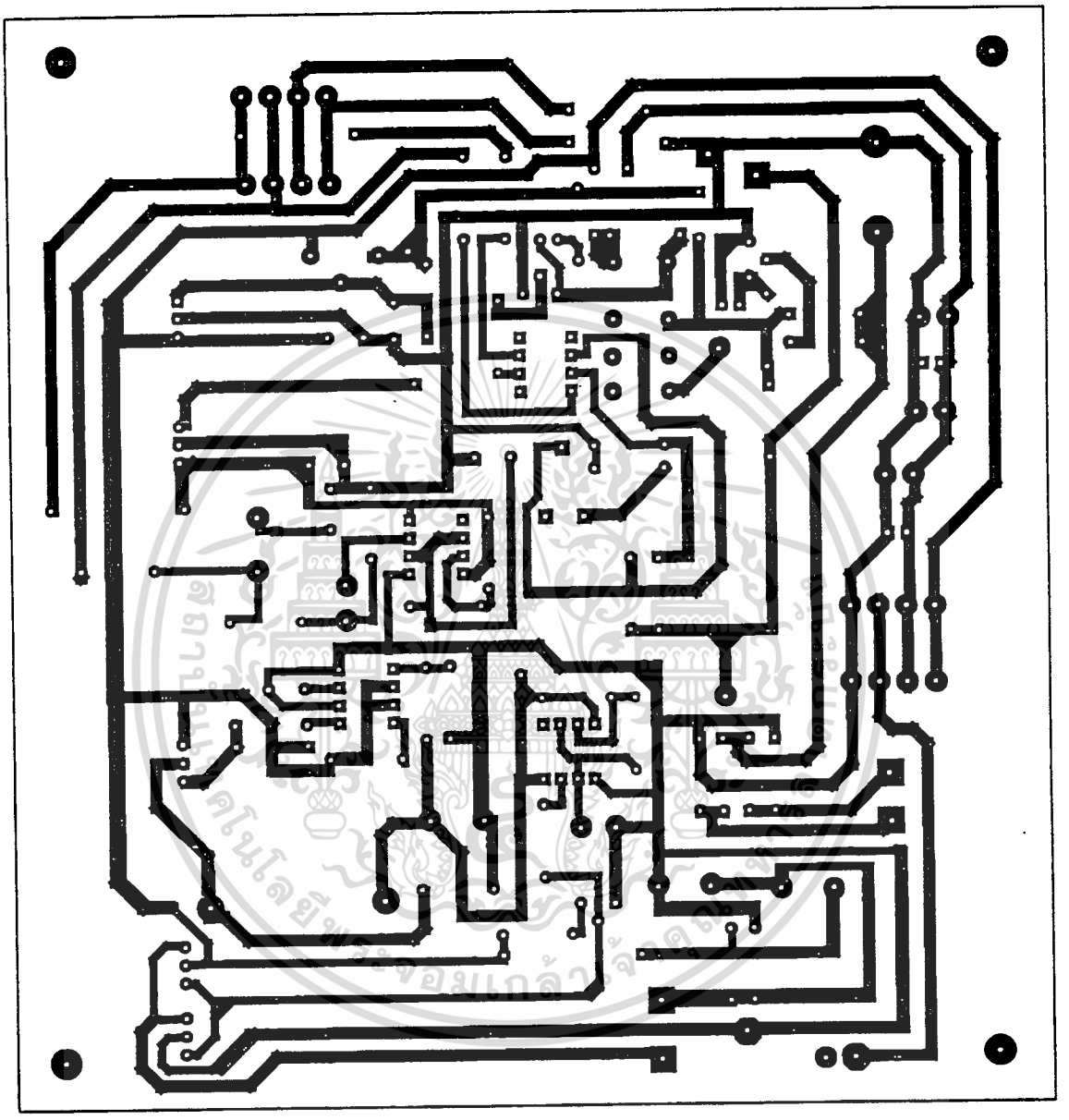
ลายวงจรส่วนตอบรับโทรศัพท์อัตโนมัติ

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ลายวงจรส่วนอินเตอร์เฟสกับคู่สายโทรศัพท์ภายใน

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



ลายวงจรส่วนแหล่งจ่ายไฟเลี้ยง

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MOTOROLA
SEMICONDUCTOR
TECHNICAL DATA

MC142100
MC145100

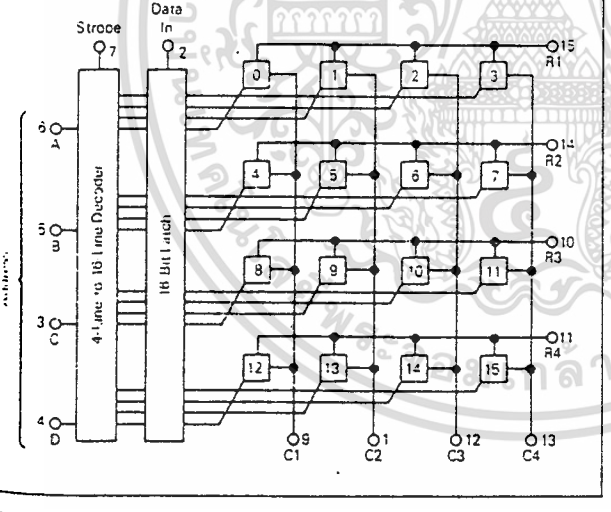
CMOS MSI
(LOW-POWER COMPLEMENTARY MOS)

4 × 4 CROSSPOINT SWITCH WITH CONTROL MEMORY

4 × 4 CROSSPOINT SWITCH WITH CONTROL MEMORY

The MC142100 and MC145100 consist of 16 crosspoint switches (analog transmission gates) organized in 4 rows and 4 columns. Both devices have 16 latches, each of which controls the state of a particular switch. Any of the 16 switches can be selected by applying its address to the device and a pulse to the strobe input. The selected crosspoint will turn on if during strobe, Data In was a one and will turn off if during strobe, Data In was a zero. In addition the MC145100 will reset all non-selected switches in the same row as the selected switch. Other switches are unaffected in the MC145100, an internal power-on reset turns off all switches as power is applied.

- Internal Latches Control State of Switches
- Power-On Reset (MC145100 Only)
- Low On Resistance — Typically on 110 Ω @ 10 Vdc
- Large Analog Range ($V_{DD} - V_{SS}$)
- All Pins Are Diode Protected
- Matched Switch Characteristics
- High CMOS Noise Immunity
- MC142100 Pin-for-Pin Replacement for CD22100



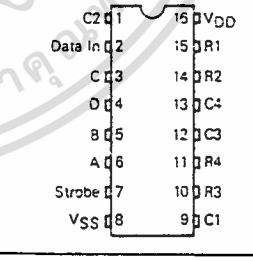
L SUFFIX
CERAMIC PACKAGE
CASE 620

P SUFFIX
PLASTIC PACKAGE
CASE 648

ORDERING INFORMATION

MC14XXXX — Suffix Denotes
- L — Ceramic Package
- P — Plastic Package
- A — Extended Operating Temperature Range
- C — Limited Operating Temperature Range

PIN ASSIGNMENTS



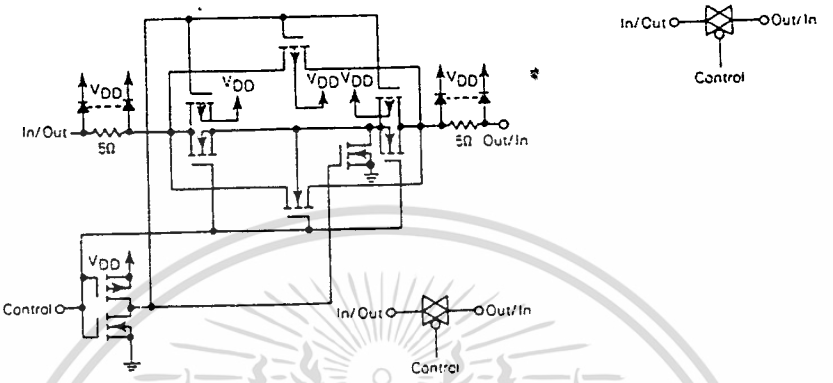
MAXIMUM RATINGS (Voltages referenced to V_{SS}, Pin 8)

Rating	Symbol	Value	Unit
DC Supply Voltage	V_{DD}	−0.5 to +18	Vdc
Input Voltage, All Inputs	V_{in}	−0.5 to V_{DD}+0.5	Vdc
DC Input Current	I	25	mAdc
Operating Temperature Range AL Device CL/CP Device	T_A	 −55 to +125 −40 to +85	°C
Storage Temperature Range	T_{stg}	−65 to +150	°C

This device contains circuitry to protect the inputs against damage due to high static voltages or electric fields; however, it is advised that normal precautions be taken to avoid application of any voltage higher than maximum rated voltages to this high impedance circuit. For proper operation it is recommended that V_{in} and V_{out} be constrained to the range $V_{SS} \le (V_{in}$ or $V_{out}) \le V_{DD}$.
Unused control inputs must always be tied to an appropriate logic voltage level (e.g., either V_{SS} or V_{DD}).

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ANALOG TRANSMISSION GATE
(CROSSPOINT) SCHEMATIC



ELECTRICAL CHARACTERISTICS (V_{SS} = 0 V)

Characteristic	Symbol	V_{DD} Vdc	T_{low}* Min	T_{low}* Max	25°C Min	25°C Typ	25°C Max	T_{high}* Min	T_{high}* Max	Unit
Operating Voltage MC145100	V_{DD}	–	4.25	18	4.25	–	18	4.25	18	Vdc
MC142100			3	18	3	–	18	3	18	
Input Voltage (Logic) "0" Level Control Input	V_{IL}	5	–	1.5	–	2.25	1.5	–	1.5	Vdc
		10	–	3.0	–	4.50	3.0	–	3.0	
		15	–	4.0	–	6.75	4.0	–	4.0	
"1" Level	V_{IH}	5	3.5	–	3.5	2.75	–	3.5	–	Vdc
		10	7.0	–	7.0	5.50	–	7.0	–	
See Figure 1		15	11.0	–	11.0	8.25	–	11.0	–	
Input Current AL	I_{in}	15	–	±0.1	–	±0.00001	±0.1	–	±1.0	μA
Pins 2, 3, 4, 5, 6, 7 CL, CP	I_{in}	15	–	±0.3	–	±0.00001	±0.3	–	±1.0	
Input Capacitance (V_{in} = 0)	C_{in}									pF
Digital Inputs		10	–	–	–	7	15	–	–	
Switch Inputs/Outputs		10	–	–	–	50	75	–	–	
Feedthrough Capacitance	$C_{in/out}$	–	–	–	–	0.4	–	–	–	pF
Quiescent Current (AL) MC145100	I_{DD}	5	–	200	–	55	110	–	70	μA
		10	–	400	–	115	230	–	100	
		15	–	600	–	170	340	–	200	
MC142100	I_{DD}	5	–	5	–	0.003	5	–	150	μA
		10	–	10	–	0.004	10	–	300	
		15	–	20	–	0.005	20	–	600	
Quiescent Current MC145100 (CL, CP Device)	I_{DD}	5	–	250	–	55	150	–	90	μA
		10	–	500	–	115	300	–	150	
		15	–	800	–	170	600	–	300	
MC142100	I_{DD}	5	–	5	–	0.003	5	–	150	μA
		10	–	10	–	0.004	10	–	300	
		15	–	20	–	0.005	20	–	600	
On-State Resistance See Figures 8-10	R_{on}	5	–	270	–	250	300	–	375	Ω
		10	–	140	–	110	170	–	220	
$V_{in} = \frac{V_{DD} - V_{SS}}{2}$		15	–	90	–	85	115	–	145	
On-State Resistance Difference Between Any Two Switches	ΔR_{on}	5	–	–	–	25	30	–	–	Ω
		10	–	–	–	15	25	–	–	
$V_{in} = \frac{V_{DD} - V_{SS}}{2}$ See Figure 6		15	–	–	–	15	20	–	–	
Input/Output Leakage AL	$I_{in/out}$	15	–	±100	–	±0.4	±100	–	±1000	nA
Current, Switch Off CL, CP		15	–	±300	–	±0.4	±300	–	±1000	

* T_{low} = 55°C for AL Device, −40°C for CL/CP Device.
T_{high} = +125°C for AL Device, ±85°C for CL/CP Device.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

SWITCHING CHARACTERISTICS (V_{SS} = 0, T_A = 25°C, C_L = 50 pF)

Characteristics		Symbol	V_{DD} Vdc	Min	Typ	Max	Unit
Propagation Delay Times Input to Output	V_{SS} = 0 Vdc	tPLH, tPHL	5 10 15	– – –	30 15 10	60 30 20	ns
Strobe to Output Output "1" to High Impedance Output "0" to High Impedance	MC142100	tPLZ, tPHZ	5 10 15	– – –	350 175 125	700 350 250	ns
Output "1" to High Impedance Output "0" to High Impedance	MC145100	tPLZ, tPHZ	5 10 15	– – –	520 215 140	1040 430 280	ns
High Impedance to Output "1" High Impedance to Output "0"	MC142100	tPZH, tPZL	5 10 15	– – –	300 150 80	600 250 160	ns
High Impedance to Output "1" High Impedance to Output "0"	MC145100	tPZH, tPZL	5 10 15	– – –	550 200 130	1100 400 260	ns
Data In to Output	MC142100	tPZH, tPHZ tPZL, tPLZ	5 10 15	– – –	300 110 75	600 220 150	ns
Data In to Output	MC145100	tPZH, tPHZ tPZL, tPLZ	5 10 15	– – –	500 200 120	1000 400 240	ns
Address to Output	MC142100	tPZH, tPHZ tPZL, tPLZ	5 10 15	– – –	350 135 90	700 270 180	ns
Address to Output See Figure 2	MC145100	tPZL, tPLZ tPZH, tPHZ	5 10 15	– – –	500 180 115	1000 360 230	ns
Minimum Setup Time Data In to Strobe	MC142100	t_{su}	5 10 15	– – –	50 10 0	190 50 30	ns
Data In to Strobe	MC145100	t_{su}	5 10 15	– – –	100 40 25	200 80 50	ns
Minimum Hold Time Data In to Strobe	MC142100	t_h	5 10 15	– – –	50 20 10	250 150 50	ns
Data In to Strobe	MC145100	t_h	5 10 15	– – –	40 10 0	400 200 90	ns
Minimum Set Up Time Address to Strobe	MC142100 MC145100	t_{su}	5 10 15	– – –	0 0 0	160 50 30	ns ns
Minimum Hold Time Address to Strobe	MC142100 MC145100	t_h	5 10 15	– – –	0 0 0	110 45 30	ns
Minimum Strobe Pulse Width	MC142100 MC145100	tWH	5 10 15	– – –	150 50 40	320 160 80	ns

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

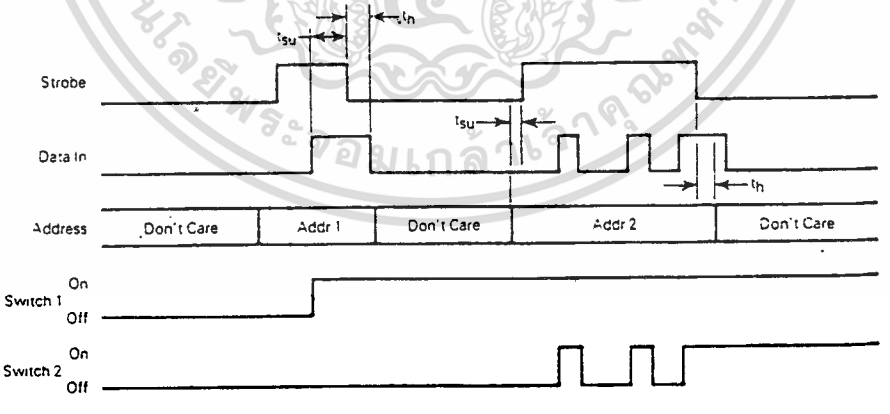
SWITCHING CHARACTERISTICS (continued) (V_{SS} = 0, T_A = 25°C, C_L = 50 pF)

Characteristics		Symbol	V_{DD} Vdc	Min	Typ	Max	Unit
Sine Wave Distortion (R_L = 1 kΩ, f = 1 kHz)	See Figure 3	–	10	–	0.5	–	%
Frequency Response (Switch On) (R_L = 1kΩ, 20 Log_{10} V_{out}/V_{in} = −3.0 dB)	See Figure 3	–	10	–	15	–	MHz
Feedthrough Attenuation (Switch Off) (V_{in} = 10 V_{pp}, F = 1.6 kHz, R_L = 1 kΩ, C_L = 15 pF)	See Figure 3	–	10	–	−80	–	dB
Frequency for Signal Crosstalk (V_{in} = 10 V_{pp}, Switch A On, Switch B Off, R_L = 1 kΩ, C_L = 15 pF)	−40 dB	–	10	–	1500	–	kHz
	−110 dB	–	–	–	0.1	–	kHz
	See Figure 4						
Crosstalk Controls to Output (R_L = 10 kΩ)	See Figure 5	–	10	–	70	–	mV

Address				Switch Selected		MC145100 Only Switches Cleared		
A	B	C	D					
0	0	0	0	C1R1	0	1	2	3
1	0	0	0	C2R1	1	0	2	3
0	1	0	0	C3R1	2	0	1	3
1	1	0	0	C4R1	3	0	1	2
0	0	1	0	C1R2	4	5	6	7
1	0	1	0	C2R2	5	4	6	7
0	1	1	0	C3R2	6	4	5	7
1	1	1	0	C4R2	7	4	5	6

Address				Switch Selected		MC145100 Only Switches Cleared		
A	B	C	D					
0	0	0	1	C1R3	9	9	10	11
1	0	0	1	C2R3	9	8	10	11
0	1	0	1	C3R3	10	8	9	11
1	1	0	1	C4R3	11	8	9	10
0	0	1	1	C1R4	12	13	14	15
1	0	1	1	C2R4	13	12	14	15
0	1	1	1	C3R4	14	12	13	15
1	1	1	1	C4R4	15	12	13	14

TIMING DIAGRAM
MC145100/MC142100



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TEST CIRCUITS

FIGURE 1 — INPUT VOLTAGE

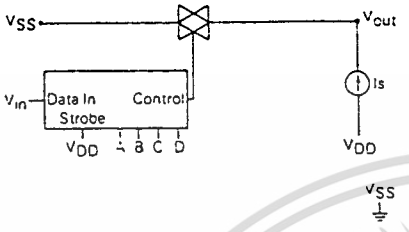


FIGURE 2 — PROPAGATION DELAY TIME

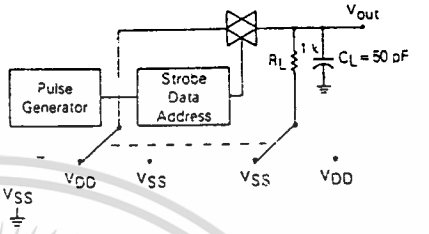


FIGURE 3 — BANDWIDTH AND FEEDTHROUGH ATTENUATION

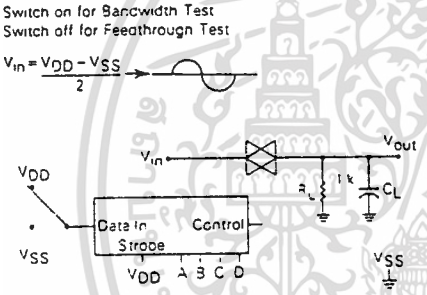


FIGURE 4 — CROSSTALK BETWEEN ANY TWO SWITCHES

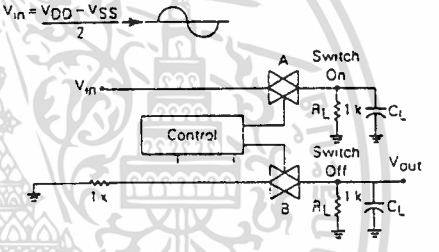
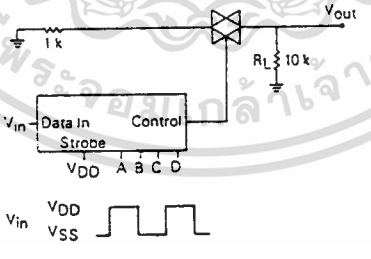
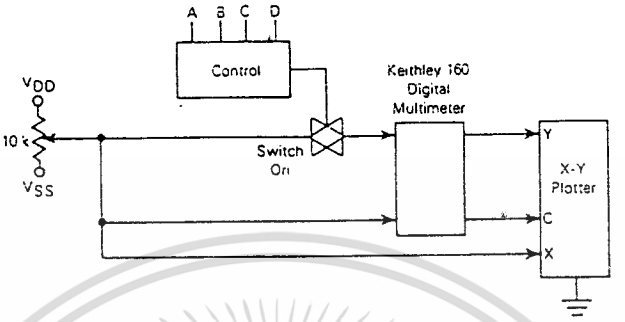


FIGURE 5 — CROSSTALK CONTROL TO OUTPUT



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FIGURE 6 — CHANNEL RESISTANCE (R_{ON}) TEST CIRCUIT



TYPICAL RESISTANCE CHARACTERISTICS

FIGURE 7 — COMPARISON AT 25°C

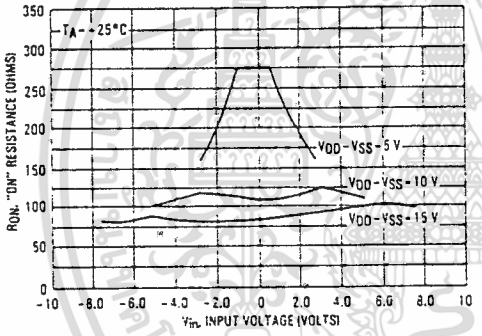


FIGURE 8 — V_{DD} = 2.5 V, V_{SS} = – 2.5 V

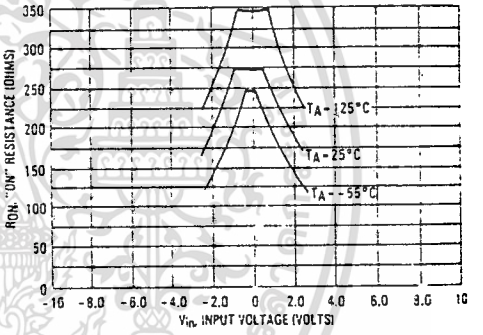


FIGURE 9 — V_{DD} = 5.0 V, V_{SS} = – 5.0 V

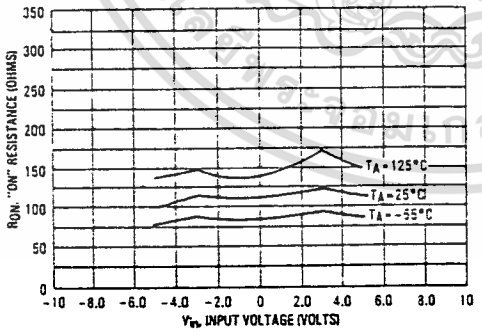
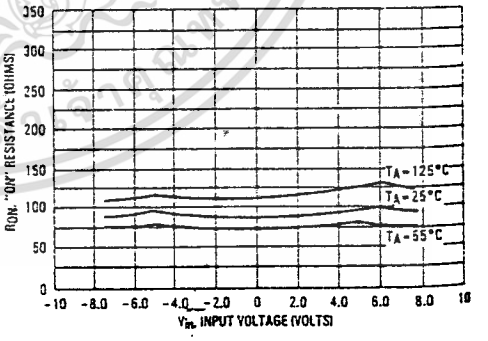


FIGURE 10 — V_{DD} = 7.5 V, V_{SS} = – 7.5 V



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

National Semiconductor

July 1993

MM54C922/MM74C922 16-Key Encoder
MM54C923/MM74C923 20-Key Encoder

General Description

These CMOS key encoders provide all the necessary logic to fully encode an array of SPST switches. The keyboard scan can be implemented by either an external clock or external capacitor. These encoders also have on-chip pull-up devices which permit switches with up to 50 kΩ on resistance to be used. No diodes in the switch array are needed to eliminate ghost switches. The internal debounce circuit needs only a single external capacitor and can be defeated by omitting the capacitor. A Data Available output goes to a high level when a valid keyboard entry has been made. The Data Available output returns to a low level when the entered key is released, even if another key is depressed. The Data Available will return high to indicate acceptance of the new key after a normal debounce period; this two-key roll-over is provided between any two switches.

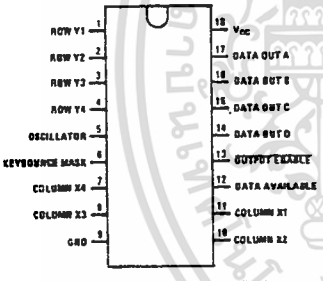
An internal register remembers the last key pressed even after the key is released. The TRI-STATE® outputs provide for easy expansion and bus operation and are LPTTL compatible.

Features

- 50 kΩ maximum switch on resistance
- On or off chip clock
- On-chip row pull-up devices
- 2 key roll-over
- Keybounce elimination with single capacitor
- Last key register at outputs
- TRI-STATE outpust LPTTL compatible
- Wide supply range 3V to 15V
- Low power consumption

Connection Diagrams

Pin Assignment for Dual-In-Line Package

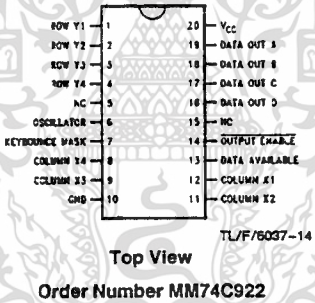


TL/F/6037–1

Top View

Order Number MM54C922 or MM74C922

Pin Assignment for SOIC

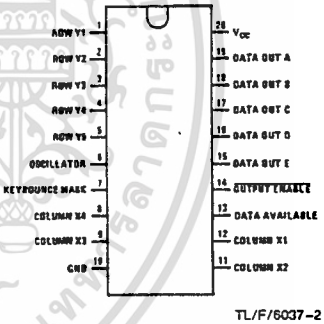


TL/F/6037–14

Top View

Order Number MM74C922

Pin Assignment for DIP and SOIC Package



TL/F/6037–2

Top View

Order Number MM54C923 or MM74C923

TRI-STATE® is a registered trademark of National Semiconductor Corporation.

©1995 National Semiconductor Corporation

RRD-B30M105/Printed in U. S. A.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings (Note 1)

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

Voltage at Any Pin	V_{CC} − 0.3V to V_{CC} + 0.3V
Operating Temperature Range	
MM54C922, MM54C923	−55°C to +125°C
MM74C922, MM74C923	−40°C to +85°C
Storage Temperature Range	−65°C to +150°C
Power Dissipation (P_D)	
Dual-In-Line	700 mW
Small Outline	500 mW
Operating V_{CC} Range	3V to 15V
V_{CC}	18V
Lead Temperature	
(Soldering, 10 seconds)	260°C

DC Electrical Characteristics Min/Max limits apply across temperature range unless otherwise specified

Symbol	Parameter	Conditions	Min	Typ	Max	Units
CMOS TO CMOS						
V_{T+}	Positive-Going Threshold Voltage at Osc and KBM Inputs	V_{CC} = 5V, I_{IN} ≥ 0.7 mA V_{CC} = 10V, I_{IN} ≥ 1.4 mA V_{CC} = 15V, I_{IN} ≥ 2.1 mA	3.0 6.0 9.0	3.6 6.8 10	4.3 8.6 12.9	V V V
V_{T-}	Negative-Going Threshold Voltage at Osc and KBM Inputs	V_{CC} = 5V, I_{IN} ≥ 0.7 mA V_{CC} = 10V, I_{IN} ≥ 1.4 mA V_{CC} = 15V, I_{IN} ≥ 2.1 mA	0.7 1.4 2.1	1.4 3.2 5	2.0 4.0 6.0	V V V
$V_{IN(1)}$	Logical "1" Input Voltage, Except Osc and KBM Inputs	V_{CC} = 5V V_{CC} = 10V V_{CC} = 15V	3.5 8.0 12.5	4.5 9 13.5		V V V
$V_{IN(0)}$	Logical "0" Input Voltage, Except Osc and KBM Inputs	V_{CC} = 5V V_{CC} = 10V V_{CC} = 15V		0.5 1 1.5	1.5 2 2.5	V V V
I_{rp}	Row Pull-Up Current at Y1, Y2, Y3, Y4 and Y5 Inputs	V_{CC} = 5V, V_{IN} = 0.1 V_{CC} V_{CC} = 10V V_{CC} = 15V		−2 −10 −22	−5 −20 −45	μA μA μA
$V_{OUT(1)}$	Logical "1" Output Voltage	V_{CC} = 5V, I_O = −10 μA V_{CC} = 10V, I_O = −10 μA V_{CC} = 15V, I_O = −10 μA	4.5 9 13.5			V V V
$V_{OUT(0)}$	Logical "0" Output Voltage	V_{CC} = 5V, I_O = 10 μA V_{CC} = 10V, I_O = 10 μA V_{CC} = 15V, I_O = 10 μA			0.5 1 1.5	V V V
R_{on}	Column "ON" Resistance at X1, X2, X3 and X4 Outputs	V_{CC} = 5V, V_O = 0.5V V_{CC} = 10V, V_O = 1V V_{CC} = 15V, V_O = 1.5V		500 300 200	1400 700 500	Ω Ω Ω
I_{CC}	Supply Current · Osc at 0V, (one Y low)	V_{CC} = 5V V_{CC} = 10V V_{CC} = 15V		0.55 1.1 1.7	1.1 1.9 2.6	mA mA mA
$I_{IN(1)}$	Logical "1" Input Current at Output Enable	V_{CC} = 15V, V_{IN} = 15V		0.005	1.0	μA
$I_{IN(0)}$	Logical "0" Input Current at Output Enable	V_{CC} = 15V, V_{IN} = 0V	−1.0	−0.005		μA
CMOS/LPTTL INTERFACE						
$V_{IN(1)}$	Logical "1" Input Voltage, Except Osc and KBM Inputs	54C, V_{CC} = 4.5V 74C, V_{CC} = 4.75V	V_{CC} − 1.5 V_{CC} − 1.5			V V
$V_{IN(0)}$	Logical "0" Input Voltage, Except Osc and KBM Inputs	54C, V_{CC} = 4.5V 74C, V_{CC} = 4.75V			0.8 0.8	V V
$V_{OUT(1)}$	Logical "1" Output Voltage	54C, V_{CC} = 4.5V I_O = −360 μA 74C, V_{CC} = 4.75V I_O = −360 μA	2.4 2.4			V V
$V_{OUT(0)}$	Logical "0" Output Voltage	54C, V_{CC} = 4.5V I_O = −360 μA 74C, V_{CC} = 4.75V I_O = −360 μA			0.4 0.4	V V

Note 1: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. Except for "Operating Temperature Range" they are not meant to imply that the devices should be operated at these limits. The table of "Electrical Characteristics" provides conditions for actual device operation.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

DC Electrical Characteristics

Min/Max limits apply across temperature range unless otherwise specified (Continued)

Symbol	Parameter	Conditions	Min	Typ	Max	Units
OUTPUT DRIVE (See 54C/74C Family Characteristics Data Sheet) (Short Circuit Current)						
I_{SOURCE}	Output Source Current (P-Channel)	V_{CC} = 5V, V_{OUT} = 0V, T_A = 25°C	−1.75	−3.3		mA
I_{SOURCE}	Output Source Current (P-Channel)	V_{CC} = 10V, V_{OUT} = 0V, T_A = 25°C	−8	−15		mA
I_{SINK}	Output Sink Current (N-Channel)	V_{CC} = 5V, V_{OUT} = V_{CC}, T_A = 25°C	1.75	3.6		mA
I_{SINK}	Output Sink Current (N-Channel)	V_{CC} = 10V, V_{OUT} = V_{CC}, T_A = 25°C	8	16		mA

AC Electrical Characteristics* T_A = 25°C, C_L = 50 pF, unless otherwise noted

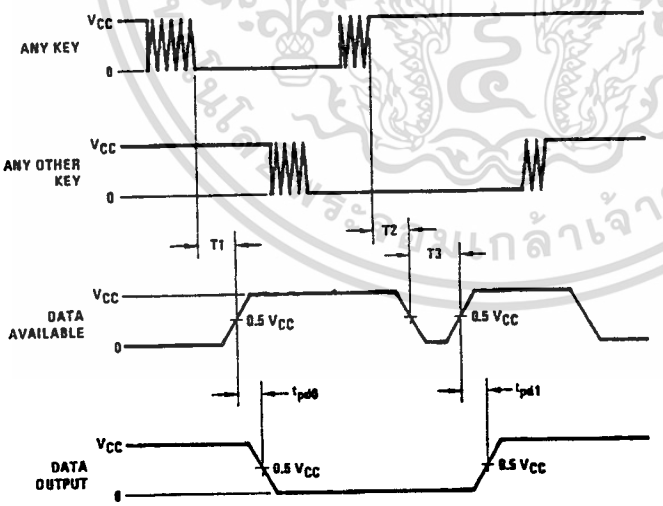
Symbol	Parameter	Conditions	Min	Typ	Max	Units
t_{pd0}, t_{pd1}	Propagation Delay Time to Logical "0" or Logical "1" from D.A.	C_L = 50 pF *(Figure 1)* V_{CC} = 5V V_{CC} = 10V V_{CC} = 15V		 60 35 25	 150 80 60	 ns ns ns
t_{0H}, t_{1H}	Propagation Delay Time from Logical "0" or Logical "1" into High Impedance State	R_L = 10k, C_L = 10 pF *(Figure 2)* V_{CC} = 5V, R_L = 10k V_{CC} = 10V, C_L = 10 pF V_{CC} = 15V		 80 65 50	 200 150 110	 ns ns ns
t_{H0}, t_{H1}	Propagation Delay Time from High Impedance State to a Logical "0" or Logical "1"	R_L = 10k, C_L = 50 pF *(Figure 2)* V_{CC} = 5V, R_L = 10k V_{CC} = 10V, C_L = 50 pF V_{CC} = 15V		 100 55 40	 250 125 90	 ns ns ns
C_{IN}	Input Capacitance	Any Input (Note 2)		5	7.5	pF
C_{OUT}	TRI-STATE Output Capacitance	Any Output (Note 2)		10		pF

*AC Parameters are guaranteed by DC correlated testing.

Note 1: "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. Except for "Operating Temperature Range" they are not meant to imply that the devices should be operated at these limits. The table of "Electrical Characteristics" provides conditions for actual device operation.

Note 2: Capacitance is guaranteed by periodic testing.

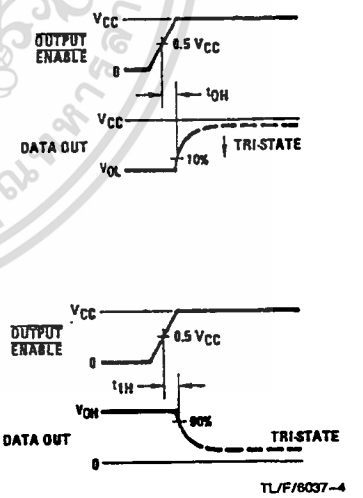
Switching Time Waveforms



TL/F/6037–3

T1 ≈ T2 ≈ RC, T3 ≈ 0.7 RC, where R ≈ 10k and C is external capacitor at KBM input.

FIGURE 1

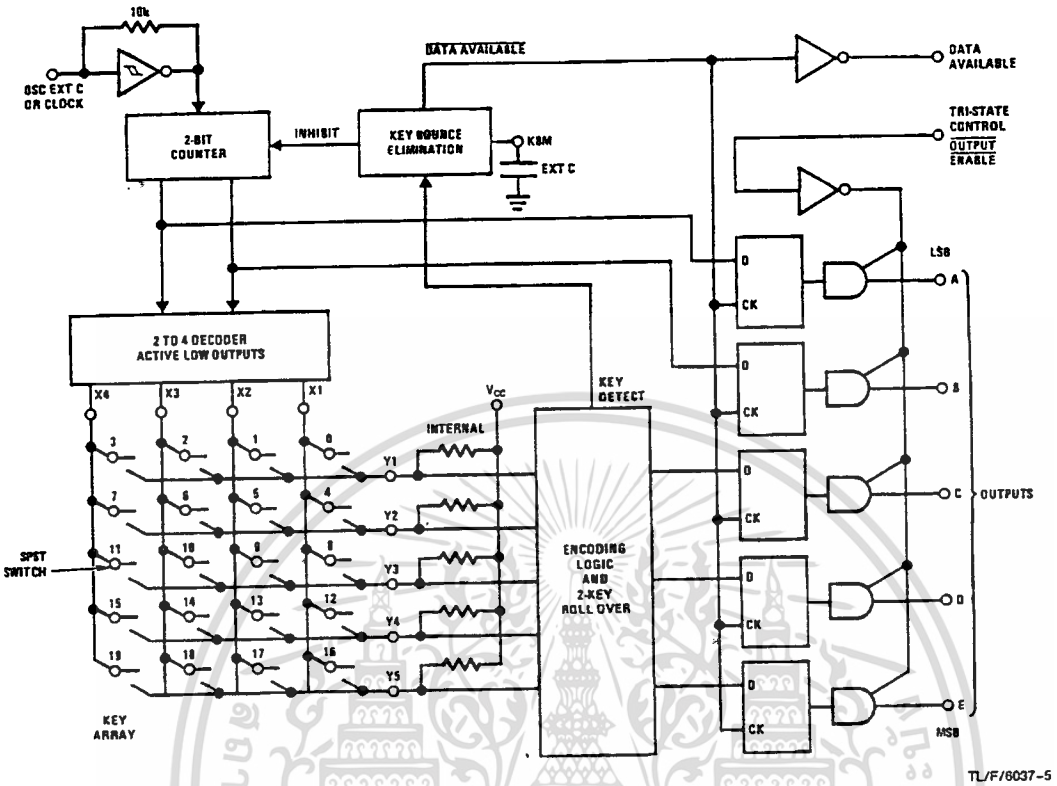


TL/F/6037–4

FIGURE 2

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Block Diagram



TL/F/6037–5

Truth Table

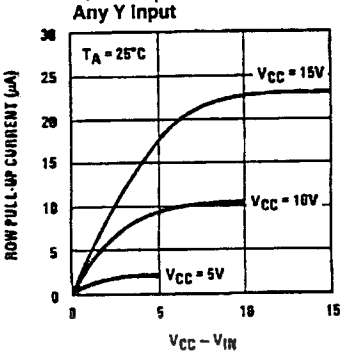
Switch Position	0 Y1,X1	1 Y1,X2	2 Y1,X3	3 Y1,X4	4 Y2,X1	5 Y2,X2	6 Y2,X3	7 Y2,X4	8 Y3,X1	9 Y3,X2	10 Y3,X3	11 Y3,X4	12 Y4,X1	13 Y4,X2	14 Y4,X3	15 Y4,X4	16 Y5*,X1	17 Y5*,X2	18 Y5*,X3	19 Y5*,X4
DATA OUT A	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
DATA OUT B	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
DATA OUT C	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
DATA OUT D	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
DATA OUT E*	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1

*Omit for MM54C922/MM74C922

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

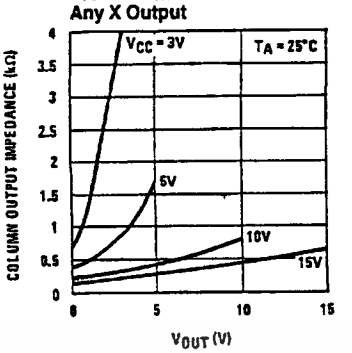
Typical Performance Characteristics

Typical I_{rp} vs V_{IN} at Any Y Input



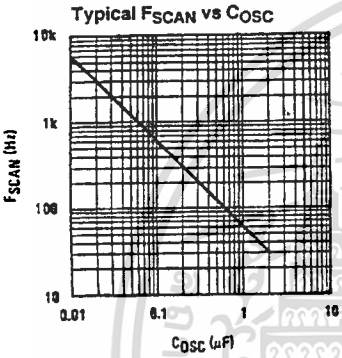
TL/F/6037–6

Typical R_{on} vs V_{OUT} at Any X Output



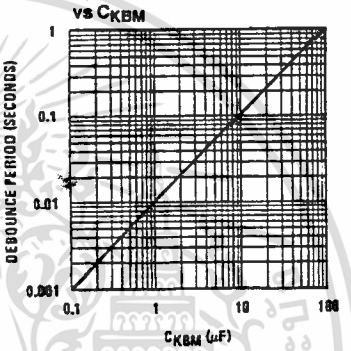
TL/F/6037–7

Typical F_{SCAN} vs C_{OSC}



TL/F/6037–8

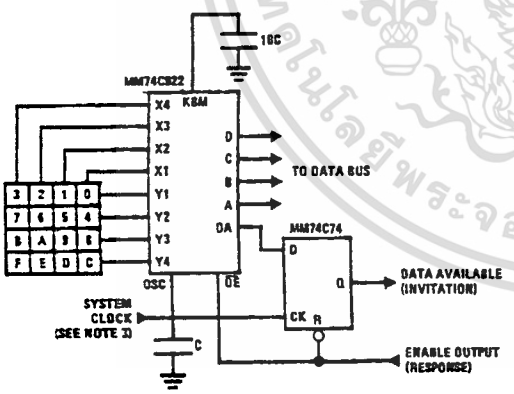
Typical Debounce Period vs C_{KBM}



TL/F/6037–9

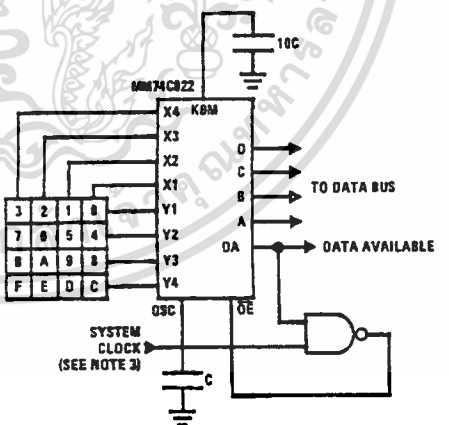
Typical Applications

Synchronous Handshake (MM74C922)



TL/F/6037–10

Synchronous Data Entry Onto Bus (MM74C922)



TL/F/6037–11

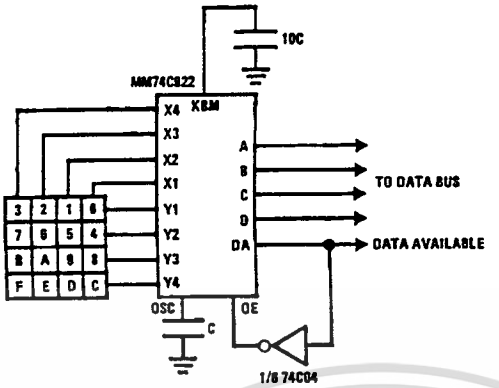
Outputs are enabled when valid entry is made and go into TRI-STATE when key is released.

Note 3: The keyboard may be synchronously scanned by omitting the capacitor at osc. and driving osc. directly if the system clock rate is lower than 10 kHz.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Typical Applications (Continued)

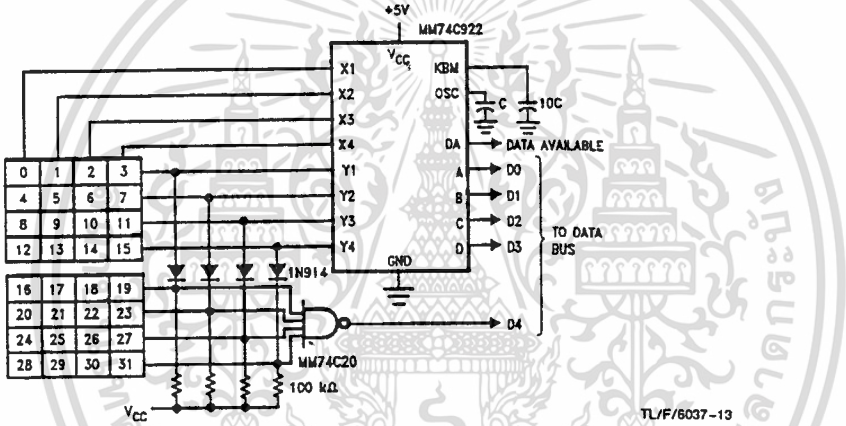
Asynchronous Data Entry Onto Bus (MM74C922)



TL/F/6037–12

Outputs are in TRI-STATE until key is pressed, then data is placed on bus. When key is released, outputs return to TRI-STATE.

Expansion to 32 Key Encoder (MM74C922)



TL/F/6037–13

Theory of Operation

The MM74C922/MM74C923 Keyboard Encoders implement all the logic necessary to interface a 16 or 20 SPST key switch matrix to a digital system. The encoder will convert a key switch closer to a 4(MM74C922) or 5(MM74C923) bit nibble. The designer can control both the keyboard scan rate and the key debounce period by altering the oscillator capacitor, C_{OSE}, and the key bounce mask capacitor, C_{MSK}. Thus, the MM74C922/MM74C923's performance can be optimized for many keyboards.

The keyboard encoders connect to a switch matrix that is 4 rows by 4 columns (MM74C922) or 5 rows by 4 columns (MM74C923). When no keys are depressed, the row inputs are pulled high by internal pull-ups and the column outputs sequentially output a logic "0". These outputs are open drain and are therefore low for 25% of the time and otherwise off. The column scan rate is controlled by the oscillator input, which consists of a Schmitt trigger oscillator, a 2-bit counter, and a 2–4-bit decoder.

When a key is depressed, key 0, for example, nothing will happen when the X1 input is off, since Y1 will remain high. When the X1 column is scanned, X1 goes low and Y1 will go low. This disables the counter and keeps X1 low. Y1 going low also initiates the key bounce circuit timing and locks out the other Y inputs. The key code to be output is a combination of the frozen counter value and the decoded Y inputs. Once the key bounce circuit times out, the data is latched, and the Data Available (DAV) output goes high.

If, during the key closure the switch bounces, Y1 input will go high again, restarting the scan and resetting the key bounce circuitry. The key may bounce several times, but as soon as the switch stays low for a debounce period, the closure is assumed valid and the data is latched.

A key may also bounce when it is released. To ensure that the encoder does not recognize this bounce as another key closure, the debounce circuit must time out before another closure is recognized.

The two-key roll-over feature can be illustrated by assuming a key is depressed, and then a second key is depressed. Since all scanning has stopped, and all other Y inputs are disabled, the second key is not recognized until the first key is lifted and the key bounce circuitry has reset.

The output latches feed TRI-STATE, which is enabled when the Output Enable ($\overline{OE}$) input is taken low.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MICROCHIP

PIC12C5XX

8-Pin, 8-Bit CMOS Microcontroller

Devices included in this Data Sheet:

PIC12C508 and PIC12C509 are 8-bit microcontrollers packaged in 8-lead packages. They are based on the Enhanced PIC16C5X family.

High-Performance RISC CPU:

- Only 33 single word instructions to learn
- All instructions are single cycle (1 µs) except for program branches which are two-cycle
- Operating speed: DC - 4 MHz clock input
  DC - 1 µs instruction cycle

Device	EPROM	RAM
PIC12C508	512 x 12	25
PIC12C509	1024 x 12	41

- 12-bit wide instructions
- 8-bit wide data path
- Seven special function hardware registers
- Two-level deep hardware stack
- Direct, indirect and relative addressing modes for data and instructions
- Internal 4 MHz RC oscillator with programmable calibration
- In-circuit serial programming

Peripheral Features:

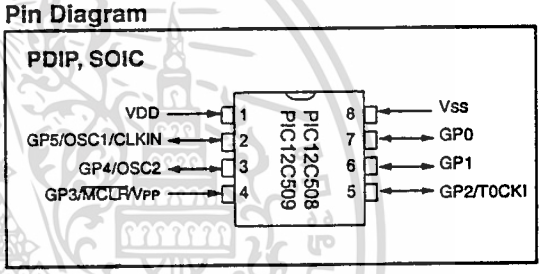
- 8-bit real time clock/counter (TMR0) with 8-bit programmable prescaler
- Power-On Reset (POR)
- Device Reset Timer (DRT)
- Watchdog Timer (WDT) with its own on-chip RC oscillator for reliable operation
- Programmable code-protection
- Power saving SLEEP mode
- Wake-up from SLEEP on pin change
- Internal pull-ups on I/O pins
- Selectable oscillator options:
  - INTRC: Internal 4 MHz RC oscillator
  - EXTRC: External low-cost RC oscillator
  - XT: Standard crystal/resonator
  - LP: Power saving, low frequency crystal
- Internal pull-up on $\overline{MCLR}$ pin

CMOS Technology:

- Low power, high speed CMOS EPROM technology
- Fully static design
- Wide operating voltage range:
  - Commercial: 2.5V to 5.5V
  - Industrial: 2.5V to 5.5V
- Low power consumption
  - < 2 mA @ 5V, 4 MHz
  - 15 µA typical @ 3V, 32 KHz
  - < 1 µA typical standby current

Pin Diagram

PDIP, SOIC



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1.0 GENERAL DESCRIPTION

The PIC12C5XX from Microchip Technology is a family of low-cost, high performance, 8-bit, fully static, EPROM/ROM-based CMOS microcontrollers. It employs a RISC architecture with only 33 single word/single cycle instructions. All instructions are single cycle (1 µs) except for program branches which take two cycles. The PIC12C5XX delivers performance an order of magnitude higher than its competitors in the same price category. The 12-bit wide instructions are highly symmetrical resulting in 2:1 code compression over other 8-bit microcontrollers in its class. The easy to use and easy to remember instruction set reduces development time significantly.

The PIC12C5XX products are equipped with special features that reduce system cost and power requirements. The Power-On Reset (POR) and Device Reset Timer (DRT) eliminate the need for external reset circuitry. There are four oscillator configurations to choose from, including INTRC internal oscillator mode and the power-saving LP (Low Power) oscillator. Power saving SLEEP mode, Watchdog Timer and code protection features improve system cost, power and reliability.

The PIC12C5XX are available in the cost-effective One-Time-Programmable (OTP) versions which are suitable for production in any volume. The customer can take full advantage of Microchip's price leadership in OTP microcontrollers while benefiting from the OTP's flexibility.

The PIC12C5XX products are supported by a full-featured macro assembler, a software simulator, an in-circuit emulator, a 'C' compiler, fuzzy logic support tools, a low-cost development programmer, and a full featured programmer. All the tools are supported on IBM® PC and compatible machines.

1.1 Applications

The PIC12C5XX series fits perfectly in applications ranging from personal care appliances and security systems to low-power remote transmitters/receivers. The EPROM technology makes customizing application programs (transmitter codes, appliance settings, receiver frequencies, etc.) extremely fast and convenient. The small footprint packages, for through hole or surface mounting, make this microcontroller series perfect for applications with space limitations. Low-cost, low-power, high performance, ease of use and I/O flexibility make the PIC12C5XX series very versatile even in areas where no microcontroller use has been considered before (e.g., timer functions, replacement of "glue" logic and PLD's in larger systems, coprocessor applications).

เอกสารนี้จัดทำขึ้นสำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TABLE 1-1: PIC12C5XX FAMILY OF DEVICES

	Clock	Memory			Peripherals	Features						
	Maximum Frequency of Operation (MHz)	EPROM Program Memory (words)	Data Memory (bytes)	Timer Module(s)	Wake-up from SLEEP on pin change	I/O Pins	Input Pins	Internal Pull-ups	Voltage Range (Volts)	In-Circuit Serial Programming	Number of Instructions	Packages
PIC12C508	4	512	25	TMR0	Yes	5	1	Yes	2.5-5.5	Yes	33	8-pin PDIP, 8-pin SOIC
PIC12C509	4	1024	41	TMR0	Yes	5	1	Yes	2.5-5.5	Yes	33	8-pin PDIP, 8-pin SOIC

All PIC12C5XX devices have Power-on Reset, selectable Watchdog Timer, selectable code protect and high I/O current capability.
All PIC12C5XX devices use serial programming with data pin GP0 and clock pin GP1.

© 1996 Microchip Technology Inc.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

2.0 PIC12C5XX DEVICE VARIETIES

A variety of packaging options are available. Depending on application and production requirements, the proper device option can be selected using the information in this section. When placing orders, please use the PIC12C5XX Product Identification System at the back of this data sheet to specify the correct part number.

2.1 One-Time-Programmable (OTP) Devices

The availability of OTP devices is especially useful for customers expecting frequent code changes and updates.

The OTP devices, packaged in plastic packages, permit the user to program them once. In addition to the program memory, the configuration bits must be programmed.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.0 ARCHITECTURAL OVERVIEW

The high performance of the PIC12C5XX family can be attributed to a number of architectural features commonly found in RISC microprocessors. To begin with, the PIC12C5XX uses a Harvard architecture in which program and data are accessed on separate buses. This improves bandwidth over traditional von Neumann architecture where program and data are fetched on the same bus. Separating program and data memory further allows instructions to be sized differently than the 8-bit wide data word. Instruction opcodes are 12-bits wide making it possible to have all single word instructions. A 12-bit wide program memory access bus fetches a 12-bit instruction in a single cycle. A two-stage pipeline overlaps fetch and execution of instructions. Consequently, all instructions (33) execute in a single cycle (1µs @ 4MHz) except for program branches.

The PIC12C508 address 512 x 12 of program memory, the PIC12C509 addresses 1K x 12 of program memory. All program memory is internal.

The PIC12C5XX can directly or indirectly address its register files and data memory. All special function registers including the program counter are mapped in the data memory. The PIC12C5XX has a highly orthogonal (symmetrical) instruction set that makes it possible to carry out any operation on any register using any addressing mode. This symmetrical nature and lack of 'special optimal situations' make programming with the PIC12C5XX simple yet efficient. In addition, the learning curve is reduced significantly.

The PIC12C5XX device contains an 8-bit ALU and working register. The ALU is a general purpose arithmetic unit. It performs arithmetic and Boolean functions between data in the working register and any register file.

The ALU is 8-bits wide and capable of addition, subtraction, shift and logical operations. Unless otherwise mentioned, arithmetic operations are two's complement in nature. In two-operand instructions, typically one operand is the W (working) register. The other operand is either a file register or an immediate constant. In single operand instructions, the operand is either the W register or a file register.

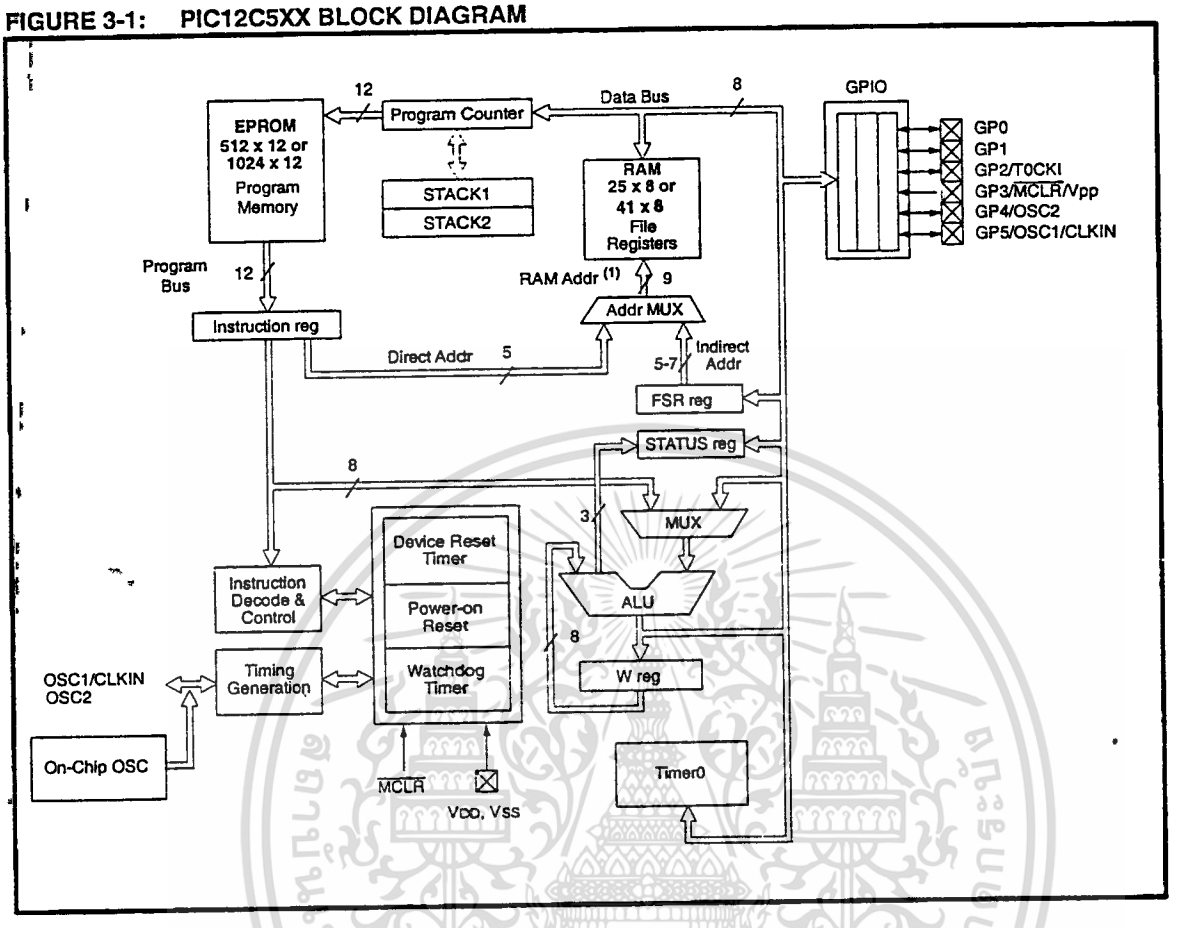
The W register is an 8-bit working register used for ALU operations. It is not an addressable register.

Depending on the instruction executed, the ALU may affect the values of the Carry (C), Digit Carry (DC), and Zero (Z) bits in the STATUS register. The C and DC bits operate as a $\overline{\text{borrow}}$ and $\overline{\text{digit borrow}}$ out bit, respectively, in subtraction. See the SUBWF and ADDWF instructions for examples.

A simplified block diagram is shown in Figure 3-1, with the corresponding device pins described in Table 3-1.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

FIGURE 3-1: PIC12C5XX BLOCK DIAGRAM



© 1996 Microchip Technology Inc.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TABLE 3-1: PIC12C5XX PINOUT DESCRIPTION

Name	DIP Pin #	SOIC Pin #	I/O/P Type	Buffer Type	Description
GP0	7	7	I/O	TTL/ST	Bi-directional I/O port/ serial programming clock. Can be software programmed for internal weak pull-up and wake-up from SLEEP on pin change. This buffer is a Schmitt Trigger input when used in serial programming mode.
GP1	6	6	I/O	TTL/ST	Bi-directional I/O port/ serial programming data. Can be software programmed for internal weak pull-up and wake-up from SLEEP on pin change. This buffer is a Schmitt Trigger input when used in serial programming mode.
GP2/T0CKI	5	5	I/O	ST	Bi-directional I/O port. Can be configured as T0CKI.
GP3/$\overline{MCLR}$/VPP	4	4	I	TTL	Input port/master clear (reset) input/programming voltage input. When configured as $\overline{MCLR}$, this pin is an active low reset to the device. Voltage on $\overline{MCLR}$/VPP must not exceed VDD during normal device operation. Can be software programmed for internal weak pull-up and wake-up from SLEEP on pin change. Weak pull-up always on if configured as $\overline{MCLR}$
GP4/OSC2	3	3	I/O	TTL	Bi-directional I/O port/oscillator crystal output. Connections to crystal or resonator in crystal oscillator mode (XT and LP modes only, GPIO in other modes).
GP5/OSC1/CLKIN	2	2	I/O	TTL/ST	Bidirectional IO port oscillator crystal input/external clock source input (GPIO in Internal RC mode only, OSC1 in all other oscillator modes).
VDD	1	1	P	—	Positive supply for logic and I/O pins
VSS	8	8	P	—	Ground reference for logic and I/O pins

Legend: I = input, O = output, I/O = input/output, P = power, — = not used, TTL = TTL input, ST = Schmitt Trigger input

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

3.1 Clocking Scheme/Instruction Cycle

The clock input (OSC1/CLKIN pin) is internally divided by four to generate four non-overlapping quadrature clocks namely Q1, Q2, Q3 and Q4. Internally, the program counter is incremented every Q1, and the instruction is fetched from program memory and latched into instruction register in Q4. It is decoded and executed during the following Q1 through Q4. The clocks and instruction execution flow is shown in Figure 3-2 and Example 3-1.

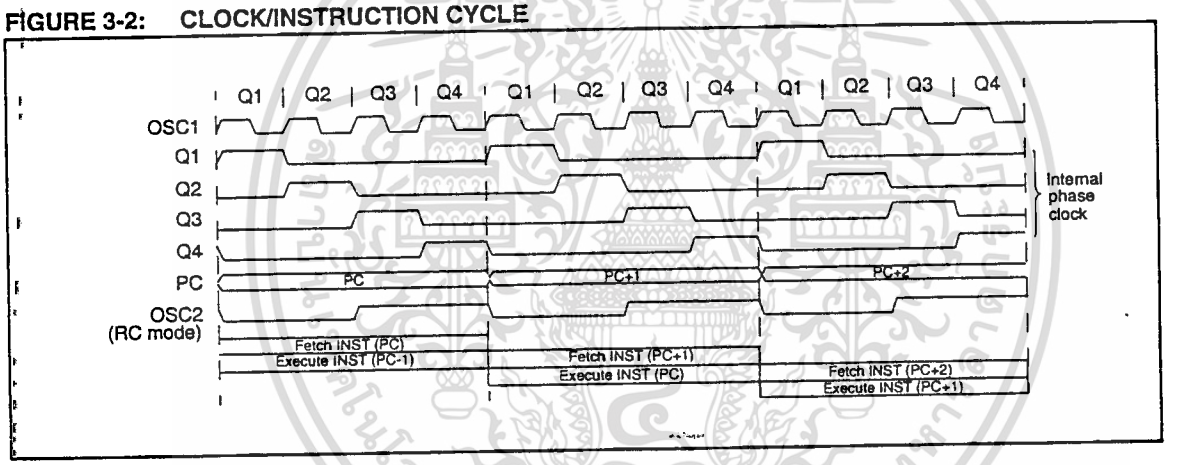
3.2 Instruction Flow/Pipelining

An Instruction Cycle consists of four Q cycles (Q1, Q2, Q3 and Q4). The instruction fetch and execute are pipelined such that fetch takes one instruction cycle while decode and execute takes another instruction cycle. However, due to the pipelining, each instruction effectively executes in one cycle. If an instruction causes the program counter to change (e.g., GOTO) then two cycles are required to complete the instruction (Example 3-1).

A fetch cycle begins with the program counter (PC) incrementing in Q1.

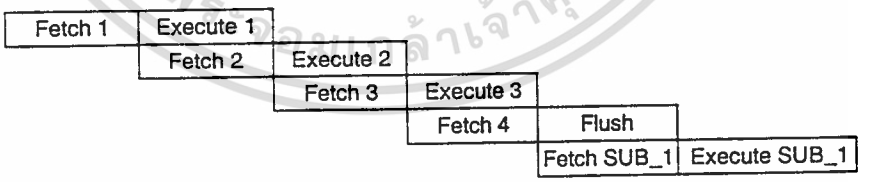
In the execution cycle, the fetched instruction is latched into the Instruction Register (IR) in cycle Q1. This instruction is then decoded and executed during the Q2, Q3, and Q4 cycles. Data memory is read during Q2 (operand read) and written during Q4 (destination write).

FIGURE 3-2: CLOCK/INSTRUCTION CYCLE



EXAMPLE 3-1: INSTRUCTION PIPELINE FLOW

```
1. MOVLW 03H
2. MOVWF GPIO
3. CALL  SUB_1
4. BSF   GPIO, BIT1
```



All instructions are single cycle, except for any program branches. These take two cycles since the fetch instruction is "flushed" from the pipeline while the new instruction is being fetched and then executed.

© 1996 Microchip Technology Inc.
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.0 MEMORY ORGANIZATION

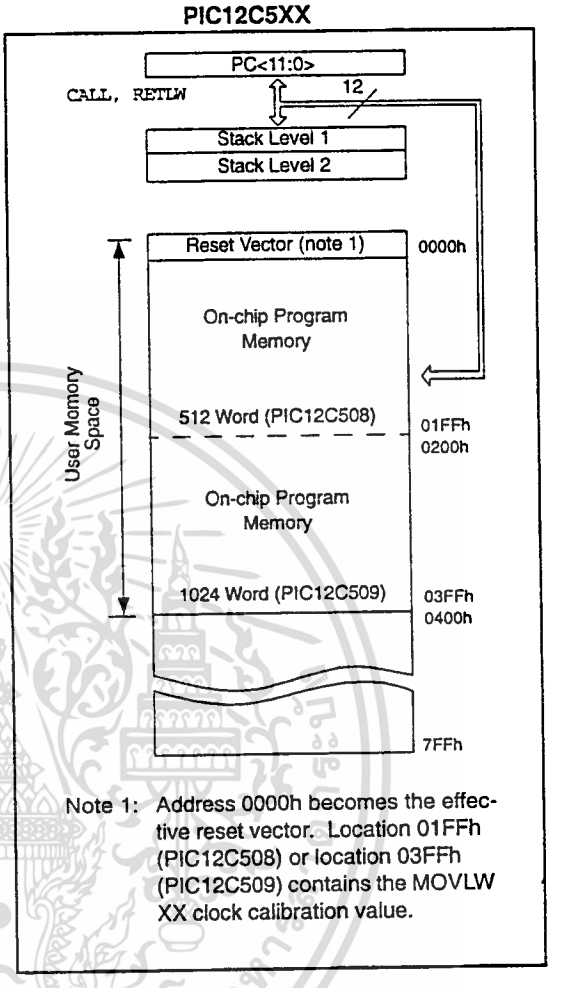
PIC12C5XX memory is organized into program memory and data memory. For devices with more than 512 bytes of program memory, a paging scheme is used. Program memory pages are accessed using one STATUS register bit. For the PIC12C509 with a data memory register file of more than 32 registers, a banking scheme is used. Data memory banks are accessed using the File Select Register (FSR).

4.1 Program Memory Organization

The PIC12C508 and PIC12C509 each have a 12-bit Program Counter (PC) capable of addressing a 2K x 12 program memory space.

Only the first 512 x 12 (0000h-01FFh) for the PIC12C508 and 1K x 12 (0000h-03FFh) for the PIC12C509 are physically implemented. Refer to Figure 4-1. Accessing a location above these boundaries will cause a wrap-around within the first 512 x 12 space (PIC12C508) or 1K x 12 space (PIC12C509). The reset vector is at 0000h. Location 01FFh (PIC12C508) or location 03FFh (PIC12C509) contains the internal clock oscillator calibration value. This value should never be overwritten.

FIGURE 4-1: PROGRAM MEMORY MAP AND STACK FOR THE PIC12C5XX



Note 1: Address 0000h becomes the effective reset vector. Location 01FFh (PIC12C508) or location 03FFh (PIC12C509) contains the MOVLW XX clock calibration value.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2 Data Memory Organization

Data memory is composed of registers, or bytes of RAM. Therefore, data memory for a device is specified by its register file. The register file is divided into two functional groups: special function registers and general purpose registers.

The special function registers include the TMR0 register, the Program Counter (PC), the Status Register, the I/O registers (ports), and the File Select Register (FSR). In addition, special purpose registers are used to control the I/O port configuration and prescaler options.

The general purpose registers are used for data and control information under command of the instructions.

For the PIC12C508, the register file is composed of 7 special function registers and 25 general purpose registers (Figure 4-2).

For the PIC12C509, the register file is composed of 7 special function registers, 25 general purpose registers, and 16 general purpose registers that may be addressed using a banking scheme (Figure 4-3).

4.2.1 GENERAL PURPOSE REGISTER FILE

The general purpose register file is accessed either directly or indirectly through the file select register FSR (Section 4.7).

FIGURE 4-2: PIC12C508 REGISTER FILE MAP

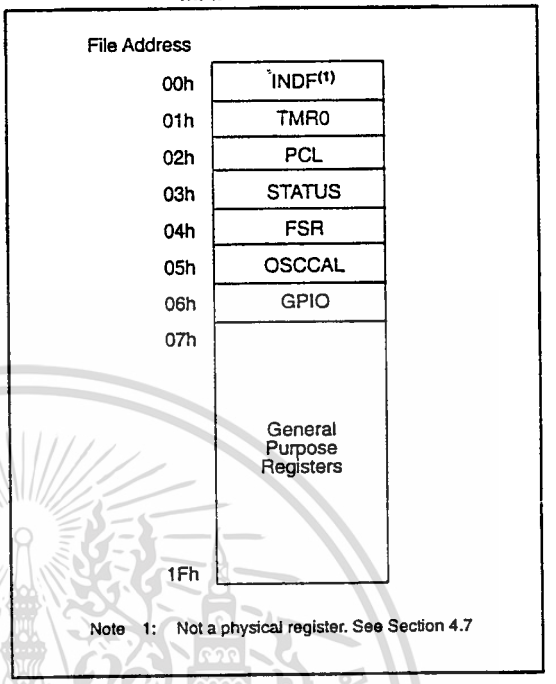
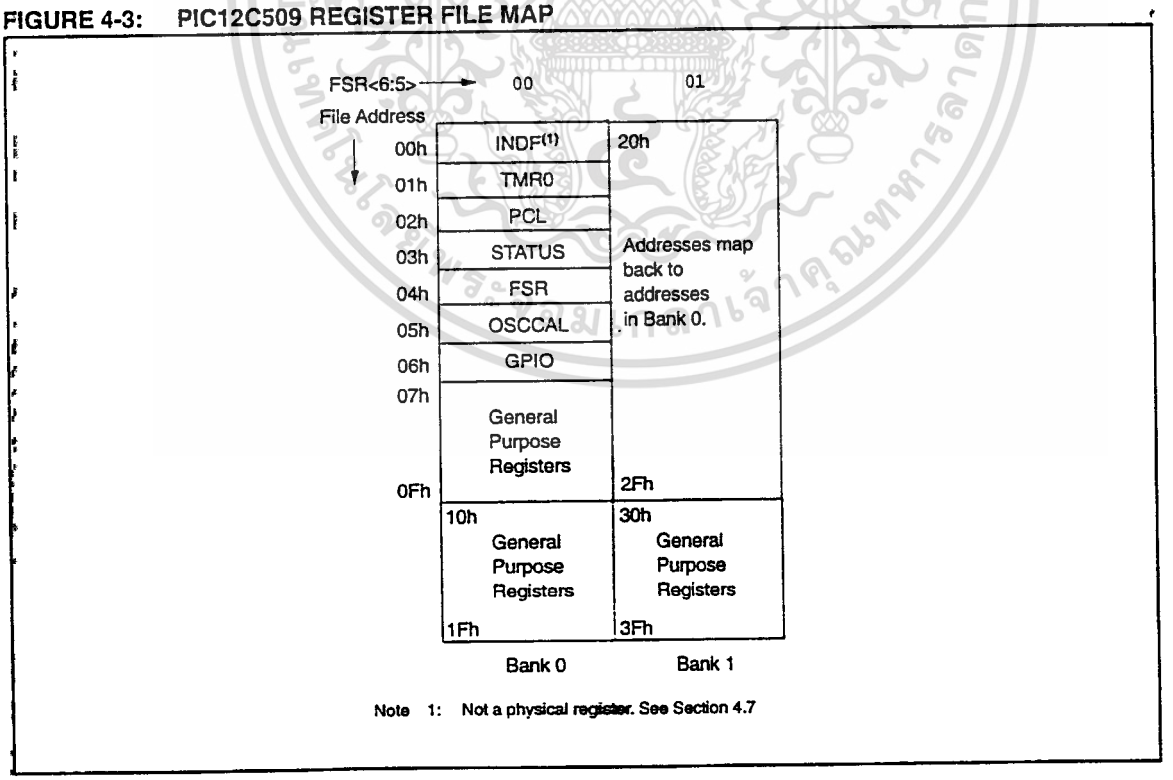


FIGURE 4-3: PIC12C509 REGISTER FILE MAP



© 1996 Microchip Technology Inc.
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

4.2.2 SPECIAL FUNCTION REGISTERS

The Special Function Registers are registers used by the CPU and peripheral functions to control the operation of the device (Table 4-1).

The special registers can be classified into two sets. The special function registers associated with the "core" functions are described in this section. Those related to the operation of the peripheral features are described in the section for each peripheral feature.

TABLE 4-1: SPECIAL FUNCTION REGISTER SUMMARY

Address	Name	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Value on Power-On Reset	Value on $\overline{MCLR}$ and WDT Reset	Value on Wake-up on Pin Change
N/A	TRIS	I/O control registers								--11 1111	--11 1111	--11 1111
N/A	OPTION	Contains control bits to configure Timer0, Timer0/WDT prescaler, interrupt on change, and weak pull-ups								1111 1111	1111 1111	1111 1111
00h	INDF	Uses contents of FSR to address data memory (not a physical register)								xxxx xxxx	uuuu uuuu	uuuu uuuu
01h	TMR0	8-bit real-time clock/counter								xxxx xxxx	uuuu uuuu	uuuu uuuu
02h(1)	PCL	Low order 8 bits of PC								1111 1111	1111 1111	1111 1111
03h	STATUS	GPWUF	—	PA0	$\overline{TO}$	$\overline{PD}$	Z	DC	C	0001 1xxx	000q quuu	100q quuu
04h	FSR (12C508)	Indirect data memory address pointer								111x xxxx	111u uuuu	111u uuuu
04h	FSR (12C509)	Indirect data memory address pointer								110x xxxx	11uu uuuu	11uu uuuu
04h	FSR	Indirect data memory address pointer								1xxx xxxx	1uuu uuuu	1uuu uuuu
05h	OSCCAL	CAL7	CAL6	CAL5	CAL4	—	—	—	—	0111 ----	uuuu ----	uuuu ----
06h	GPIO	—	—	GP5	GP4	GP3	GP2	GP1	GP0	--xx xxxx	--uu uuuu	--uu uuuu

Legend: Shaded boxes = unimplemented or unused, – = unimplemented, read as '0' (if applicable)
x = unknown, u = unchanged, q = see the tables in Section 7.7 for possible values.

Note 1: The upper byte of the Program Counter is not directly accessible. See Section 4.5 for an explanation of how to access these bits.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ISO²-CMOS MT8870D/MT8870D-1
Integrated DTMF Receiver

ISSUE 4 August 1996

Features

- Complete DTMF Receiver
- Low power consumption
- Internal gain setting amplifier
- Adjustable guard time
- Central office quality
- Power-down mode
- Inhibit mode
- Backward compatible with MT8870C/MT8870C-1

Applications

- Receiver system for British Telecom (BT) or CEPT Spec (MT8870D-1)
- Paging systems
- Repeater systems/mobile radio
- Credit card systems
- Remote control
- Personal computers
- Telephone answering machine

Ordering Information

MT8870DE/DE-1	18 Pin Plastic DIP
MT8870DC/DC-1	18 Pin Ceramic DIP
MT8870DS/DS-1	18 Pin SOIC
MT8870DN/DN-1	20 Pin SSOP

-40 °C to +85 °C

Description

The MT8870D/MT8870D-1 is a complete DTMF receiver integrating both the bandsplit filter and digital decoder functions. The filter section uses switched capacitor techniques for high and low group filters; the decoder uses digital counting techniques to detect and decode all 16 DTMF tone pairs into a 4-bit code. External component count is minimized by on chip provision of a differential input amplifier, clock oscillator and latched three-state bus interface.

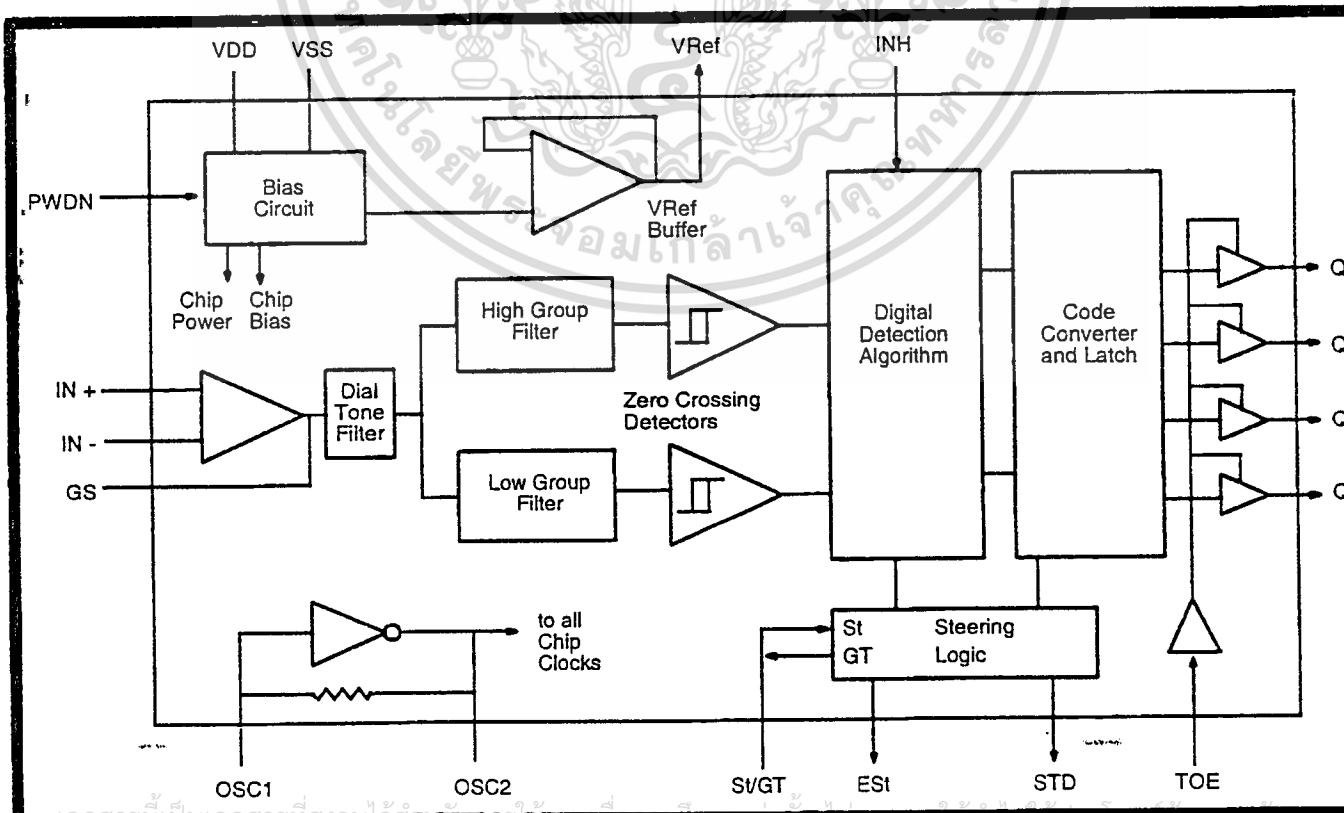
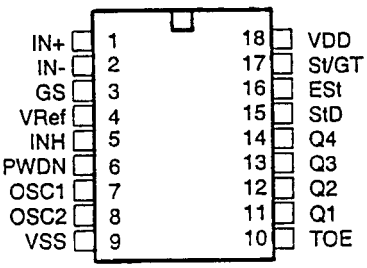
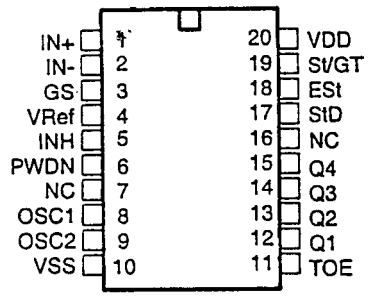


Figure 1 - Functional Block Diagram

ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



18 PIN CERDIP/PLASTIC DIP/SOIC



20 PIN SSOP

Figure 2 - Pin Connections

Pin Description

Pin #		Name	Description
18	20		
1	1	IN+	**Non-Inverting Op-Amp (Input).**
2	2	IN-	**Inverting Op-Amp (Input).**
3	3	GS	**Gain Select.** Gives access to output of front end differential amplifier for connection of feedback resistor.
4	4	V_{Ref}	**Reference Voltage (Output).** Nominally $V_{DD}/2$ is used to bias inputs at mid-rail (see Fig. 6 and Fig. 10).
5	5	INH	**Inhibit (Input).** Logic high inhibits the detection of tones representing characters A, B, C and D. This pin input is internally pulled down.
6	6	PWDN	**Power Down (Input).** Active high. Powers down the device and inhibits the oscillator. This pin input is internally pulled down.
7	8	OSC1	**Clock (Input).**
8	9	OSC2	**Clock (Output).** A 3.579545 MHz crystal connected between pins OSC1 and OSC2 completes the internal oscillator circuit.
9	10	V_{SS}	**Ground (Input).** 0V typical.
10	11	TOE	**Three State Output Enable (Input).** Logic high enables the outputs Q1-Q4. This pin is pulled up internally.
11-14	12-15	Q1-Q4	**Three State Data (Output).** When enabled by TOE, provide the code corresponding to the last valid tone-pair received (see Table 1). When TOE is logic low, the data outputs are high impedance.
15	17	StD	**Delayed Steering (Output).**Presents a logic high when a received tone-pair has been registered and the output latch updated; returns to logic low when the voltage on St/GT falls below V_{TSt}.
16	18	ESt	**Early Steering (Output).** Presents a logic high once the digital algorithm has detected a valid tone pair (signal condition). Any momentary loss of signal condition will cause ESt to return to a logic low.
17	19	St/GT	**Steering Input/Guard time (Output) Bidirectional.** A voltage greater than V_{TSt} detected at St causes the device to register the detected tone pair and update the output latch. A voltage less than V_{TSt} frees the device to accept a new tone pair. The GT output acts to reset the external steering time-constant; its state is a function of ESt and the voltage on St.
18	20	V_{DD}	**Positive power supply (Input).** +5V typical.
	7, 16	NC	No Connection.

เอกสารนี้เผยแพร่เพื่อสนับสนุนการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า

Functional Description

The MT8870D/MT8870D-1 monolithic DTMF receiver offers small size, low power consumption and high performance. Its architecture consists of a bandsplit filter section, which separates the high and low group tones, followed by a digital counting section which verifies the frequency and duration of the received tones before passing the corresponding code to the output bus.

Filter Section

Separation of the low-group and high group tones is achieved by applying the DTMF signal to the inputs of two sixth-order switched capacitor bandpass filters, the bandwidths of which correspond to the low and high group frequencies. The filter section also incorporates notches at 350 and 440 Hz for exceptional dial tone rejection (see Figure 3). Each filter output is followed by a single order switched capacitor filter section which smooths the signals prior to limiting. Limiting is performed by high-gain comparators which are provided with hysteresis to prevent detection of unwanted low-level signals. The outputs of the comparators provide full rail logic swings at the frequencies of the incoming DTMF signals.

Decoder Section

Following the filter section is a decoder employing digital counting techniques to determine the frequencies of the incoming tones and to verify that they correspond to standard DTMF frequencies. A complex averaging algorithm protects against tone simulation by extraneous signals such as voice while providing tolerance to small frequency deviations and variations. This averaging algorithm has been developed to ensure an optimum combination of immunity to talk-off and tolerance to the presence of interfering frequencies (third tones) and noise. When the detector recognizes the presence of two valid tones (this is referred to as the "signal condition" in some industry specifications) the "Early Steering" (ESt) output will go to an active state. Any subsequent loss of signal condition will cause ESt to assume an inactive state (see "Steering Circuit").

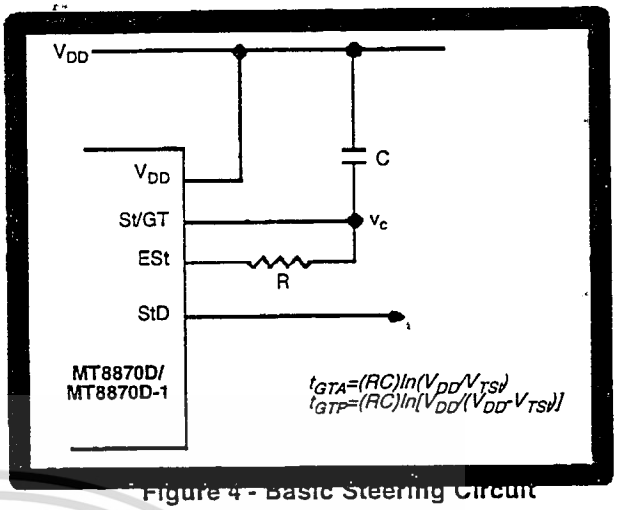


Figure 4 - Basic Steering Circuit

Steering Circuit

Before registration of a decoded tone pair, the receiver checks for a valid signal duration (referred to as character recognition condition). This check is performed by an external RC time constant driven by ESt. A logic high on ESt causes v_c (see Figure 4) to rise as the capacitor discharges. Provided signal

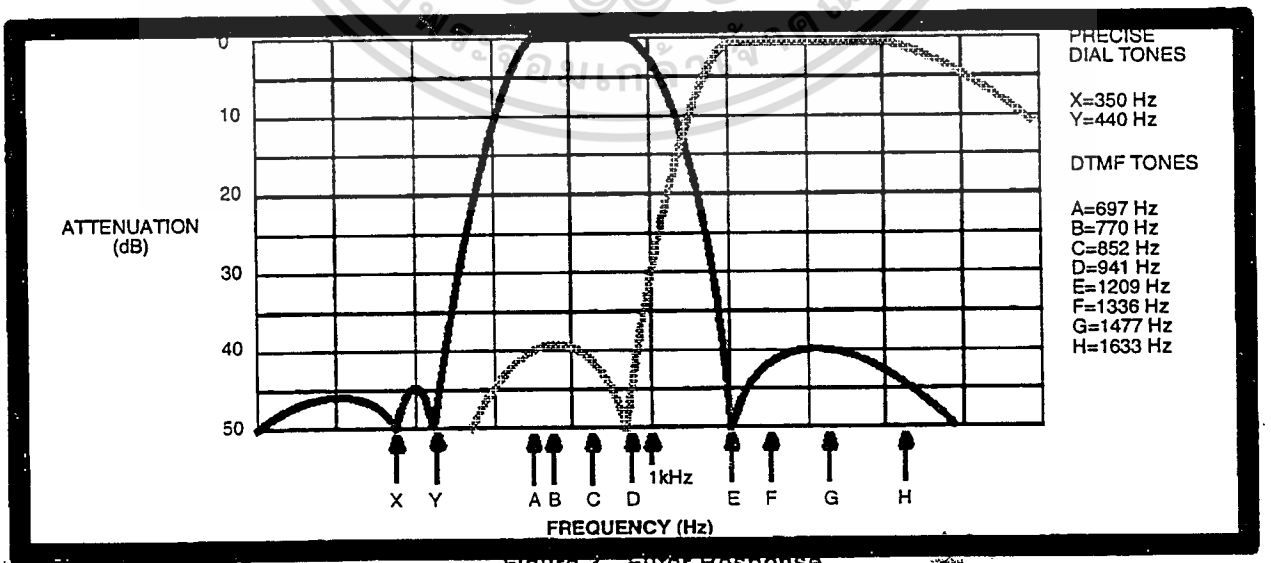


Figure 3 - Filter Response

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

condition is maintained (ESt remains high) for the validation period (t_{GTP}), v_c reaches the threshold (V_{TSt}) of the steering logic to register the tone pair, latching its corresponding 4-bit code (see Table 1) into the output latch. At this point the GT output is activated and drives v_c to V_{DD}. GT continues to drive high as long as ESt remains high. Finally, after a short delay to allow the output latch to settle, the delayed steering output flag (StD) goes high, signalling that a received tone pair has been registered. The contents of the output latch are made available on the 4-bit output bus by raising the three state control input (TOE) to a logic high. The steering circuit works in reverse to validate the interdigit pause between signals. Thus, as well as rejecting signals too short to be considered valid, the receiver will tolerate signal interruptions (dropout) too short to be considered a valid pause. This facility, together with the capability of selecting the steering time constants externally, allows the designer to tailor performance to meet a wide variety of system requirements.

Digit	TOE	INH	ESt	Q_4	Q_3	Q_2	Q_1
ANY	L	X	H	Z	Z	Z	Z
1	H	X	H	0	0	0	1
2	H	X	H	0	0	1	0
3	H	X	H	0	0	1	1
4	H	X	H	0	1	0	0
5	H	X	H	0	1	0	1
6	H	X	H	0	1	1	0
7	H	X	H	0	1	1	1
8	H	X	H	1	0	0	0
9	H	X	H	1	0	0	1
0	H	X	H	1	0	1	0
*	H	X	H	1	0	1	1
#	H	X	H	1	1	0	0
A	H	L	H	1	1	0	1
B	H	L	H	1	1	1	0
C	H	L	H	1	1	1	1
D	H	L	H	0	0	0	0
A	H	H	L	undetected, the output code will remain the same as the previous detected code			
B	H	H	L				
C	H	H	L				
D	H	H	L				

Table 1. Functional Decode Table

L=LOGIC LOW, H=LOGIC HIGH, Z=HIGH IMPEDANCE
X = DON'T CARE

Guard Time Adjustment

In many situations not requiring selection of tone duration and interdigital pause, the simple steering circuit shown in Figure 4 is applicable. Component values are chosen according to the formula:

$$t_{REC}=t_{DP}+t_{GTP}$$
$$t_{ID}=t_{DA}+t_{GTA}$$

The value of t_{DP} is a device parameter (see Figure 11) and t_{REC} is the minimum signal duration to be recognized by the receiver. A value for C of 0.1 µF is recommended for most applications, leaving R to be selected by the designer.

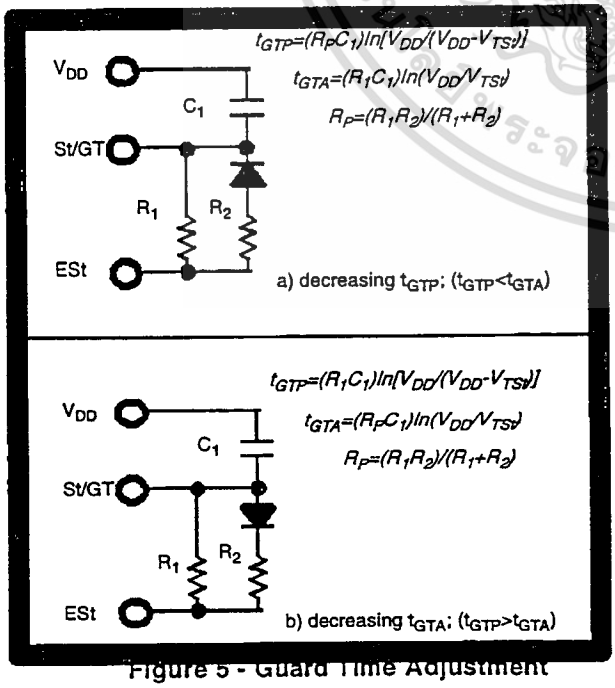


Figure 5 - Guard Time Adjustment

Different steering arrangements may be used to select independently the guard times for tone present (t_{GTP}) and tone absent (t_{GTA}). This may be necessary to meet system specifications which place both accept and reject limits on both tone duration and interdigital pause. Guard time adjustment also allows the designer to tailor system parameters such as talk off and noise immunity. Increasing t_{REC} improves talk-off performance since it reduces the probability that tones simulated by speech will maintain signal condition long enough to be registered. Alternatively, a relatively short t_{REC} with a long t_{DO} would be appropriate for extremely noisy environments where fast acquisition time and immunity to tone drop-outs are required. Design information for guard time adjustment is shown in Figure 5.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Power-down and Inhibit Mode

A logic high applied to pin 6 (PWDN) will power down the device to minimize the power consumption in a standby mode. It stops the oscillator and the functions of the filters.

Inhibit mode is enabled by a logic high input to the pin 5 (INH). It inhibits the detection of tones representing characters A, B, C, and D. The output code will remain the same as the previous detected code (see Table 1).

Differential Input Configuration

The input arrangement of the MT8870D/MT8870D-1 provides a differential-input operational amplifier as well as a bias source (V_{Ref}) which is used to bias the inputs at mid-rail. Provision is made for connection of a feedback resistor to the op-amp output (GS) for adjustment of gain. In a single-ended configuration, the input pins are connected as shown in Figure 10 with the op-amp connected for unity gain and V_{Ref} biasing the input at $^1/_2V_{DD}$. Figure 6 shows the differential configuration, which permits the adjustment of gain with the feedback resistor R_5.

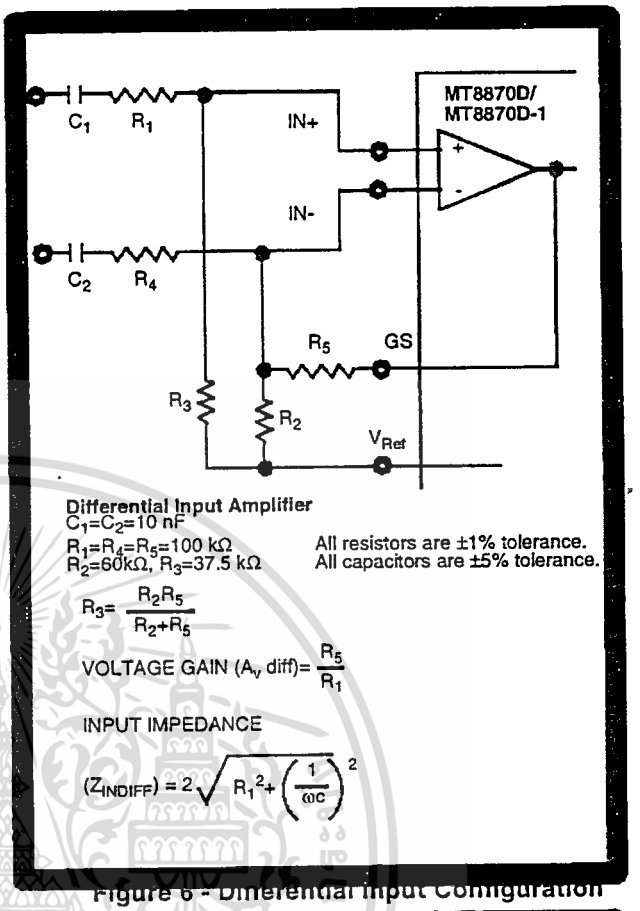


Figure 6 - Differential Input Configuration

Crystal Oscillator

The internal clock circuit is completed with the addition of an external 3.579545 MHz crystal and is normally connected as shown in Figure 10 (Single-Ended Input Configuration). However, it is possible to configure several MT8870D/MT8870D-1 devices employing only a single oscillator crystal. The oscillator output of the first device in the chain is coupled through a 30 pF capacitor to the oscillator input (OSC1) of the next device. Subsequent devices are connected in a similar fashion. Refer to Figure 7 for details. The problems associated with unbalanced loading are not a concern with the arrangement shown, i.e., precision balancing capacitors are not required.

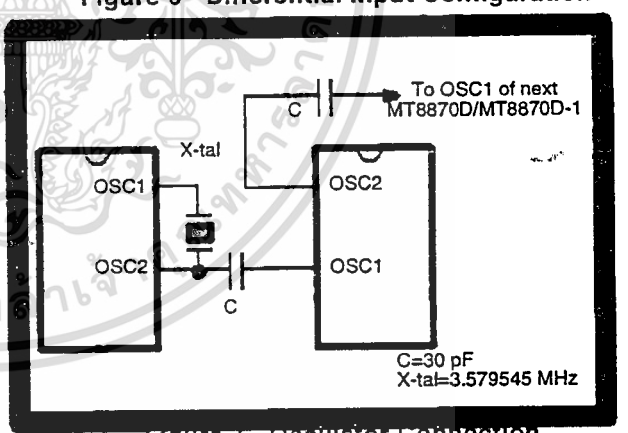


Figure 7 - Oscillator Connection

Parameter	Unit	Resonator
R1	Ohms	10.752
L1	mH	.432
C1	pF	4.984
C0	pF	37.915
Qm	-	896.37
Δf	%	±0.2%

Table 2. Recommended Resonator Specifications

Note: Qm=quality factor of RLC model, i.e., $1/2\Pi fR1C1$.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Applications

RECEIVER SYSTEM FOR BRITISH TELECOM SPEC POR 1151

The circuit shown in Fig. 9 illustrates the use of MT8870D-1 device in a typical receiver system. BT Spec defines the input signals less than -34 dBm as the non-operate level. This condition can be attained by choosing a suitable values of R_1 and R_2 to provide 3 dB attenuation, such that -34 dBm input signal will correspond to -37 dBm at the gain setting pin GS of MT8870D-1. As shown in the diagram, the component values of R_3 and C_2 are the guard time requirements when the total component tolerance is 6%. For better performance, it is recommended to use the non-symmetric guard time circuit in Fig. 8.

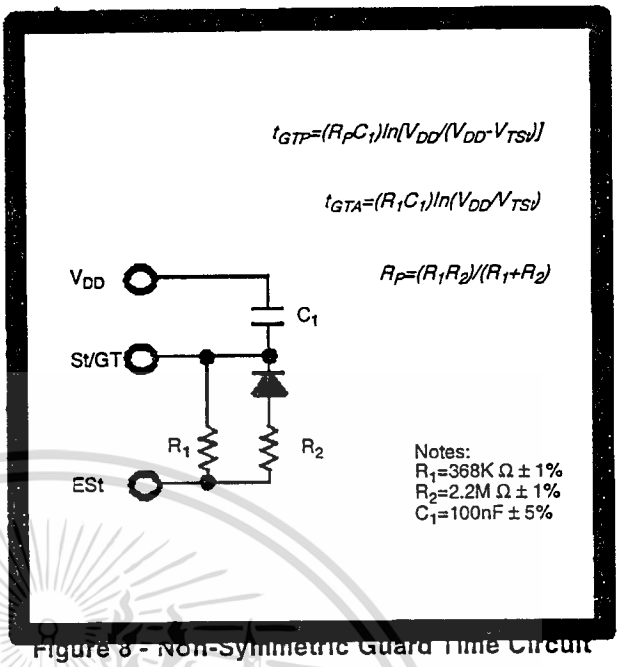


Figure 8 - Non-Symmetric Guard Time Circuit

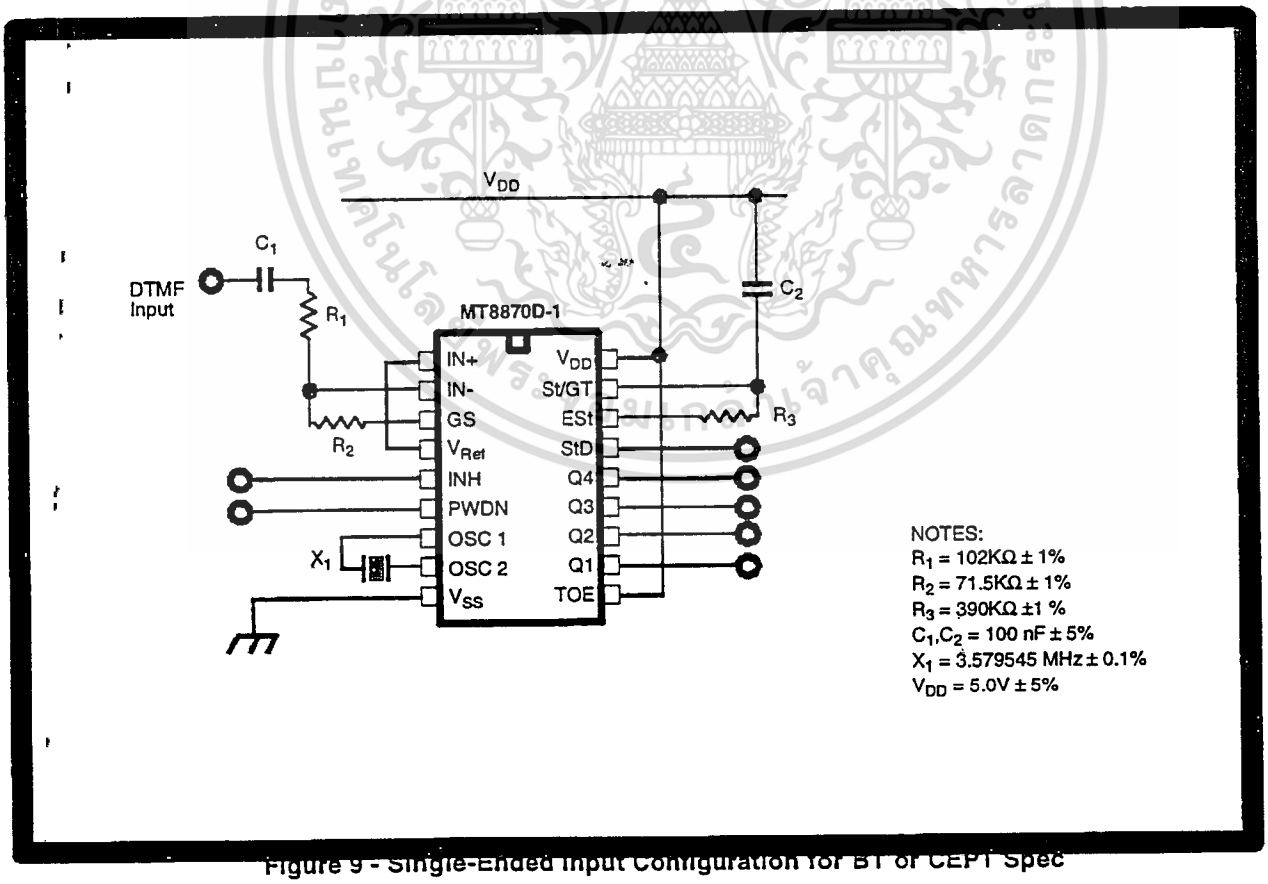


Figure 9 - Single-Ended Input Configuration for BT or CEPT Spec

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Absolute Maximum Ratings†

	Parameter	Symbol	Min	Max	Units
1	DC Power Supply Voltage	V_{DD}		7	V
2	Voltage on any pin	V_I	V_{SS}-0.3	V_{DD}+0.3	V
3	Current at any pin (other than supply)	I_I		10	mA
4	Storage temperature	T_{STG}	-65	+150	°C
5	Package power dissipation	P_D		500	mW

† Exceeding these values may cause permanent damage. Functional operation under these conditions is not implied.
Derate above 75 °C at 16 mW / °C. All leads soldered to board.

Recommended Operating Conditions - Voltages are with respect to ground (V_{SS}) unless otherwise stated.

	Parameter	Sym	Min	Typ‡	Max	Units	Test Conditions
1	DC Power Supply Voltage	V_{DD}	4.75	5.0	5.25	V	
2	Operating Temperature	T_O	-40		+85	°C	
3	Crystal/Clock Frequency	fc		3.579545		MHz	
4	Crystal/Clock Freq.Tolerance	Δfc		±0.1		%	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

DC Electrical Characteristics - V_{DD}=5.0V± 5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C, unless otherwise stated.

		Characteristics	Sym	Min	Typ‡	Max	Units	Test Conditions
1	SUPPLY	Standby supply current	I_{DDQ}		10	25	µA	PWDN=V_{DD}
2		Operating supply current	I_{DD}		3.0	9.0	mA	
3		Power consumption	P_O		15		mW	fc=3.579545 MHz
4	INPUTS	High level input	V_{IH}	3.5			V	V_{DD}=5.0V
5		Low level input voltage	V_{IL}			1.5	V	V_{DD}=5.0V
6		Input leakage current	I_{IH}/I_{IL}		0.1		µA	V_{IN}=V_{SS} or V_{DD}
7		Pull up (source) current	I_{SO}		7.5	20	µA	TOE (pin 10)=0, V_{DD}=5.0V
8		Pull down (sink) current	I_{SI}		15	45	µA	INH=5.0V, PWDN=5.0V, V_{DD}=5.0V
9		Input impedance (IN+, IN-)	R_{IN}		10		MΩ	@ 1 kHz
10		Steering threshold voltage	V_{TSt}	2.2	2.4	2.5	V	V_{DD} = 5.0V
11	OUTPUTS	Low level output voltage	V_{OL}			V_{SS}+0.03	V	No load
12		High level output voltage	V_{OH}	V_{DD}-0.03			V	No load
13		Output low (sink) current	I_{OL}	1.0	2.5		mA	V_{OUT}=0.4 V
14		Output high (source) current	I_{OH}	0.4	0.8		mA	V_{OUT}=4.6 V
15		V_{Ref} output voltage	V_{Ref}	2.3	2.5	2.7	V	No load, V_{DD} = 5.0V
16		V_{Ref} output resistance	R_{OR}		1		kΩ	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

Operating Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C ,unless otherwise stated.

Gain Setting Amplifier

	Characteristics	Sym	Min	Typ‡	Max	Units	Test Conditions
1	Input leakage current	I_{IN}			100	nA	$V_{SS} \le V_{IN} \le V_{DD}$
2	Input resistance	R_{IN}	10			MΩ	
3	Input offset voltage	V_{OS}			25	mV	
4	Power supply rejection	PSRR	50			dB	1 kHz
5	Common mode rejection	CMRR	40			dB	0.75 V ≤ V_{IN} ≤ 4.25 V biased at V_{Ref} =2.5 V
6	DC open loop voltage gain	A_{VOL}	32			dB	
7	Unity gain bandwidth	f_C	0.30			MHz	
8	Output voltage swing	V_O	4.0			V_{pp}	Load ≥ 100 kΩ to V_{SS} @ GS
9	Maximum capacitive load (GS)	C_L			100	pF	
10	Resistive load (GS)	R_L			50	kΩ	
11	Common mode range	V_{CM}	2.5			V_{DD}	No Load

MT8870D AC Electrical Characteristics - V_{DD}=5.0V ±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C , using Test Circuit shown in Figure 10.

	Characteristics	Sym	Min	Typ‡	Max	Units	Notes*
1	Valid input signal levels (each tone of composite signal)		-29		+1	dBm	1,2,3,5,6,9
			27.5		869	mV_{RMS}	1,2,3,5,6,9
2	Negative twist accept				8	dB	2,3,6,9,12
3	Positive twist accept				8	dB	2,3,6,9,12
4	Frequency deviation accept		±1.5% ± 2 Hz				2,3,5,9
5	Frequency deviation reject		±3.5%				2,3,5,9
6	Third tone tolerance			-16		dB	2,3,4,5,9,10
7	Noise tolerance			-12		dB	2,3,4,5,7,9,10
8	Dial tone tolerance			+22		dB	2,3,4,5,8,9,11

‡ Typical figures are at 25 °C and are for design aid only: not guaranteed and not subject to production testing.

***NOTES**

1. dBm= decibels above or below a reference power of 1 mW into a 600 ohm load.
2. Digit sequence consists of all DTMF tones.
3. Tone duration= 40 ms, tone pause= 40 ms.
4. Signal condition consists of nominal DTMF frequencies.
5. Both tones in composite signal have an equal amplitude.
6. Tone pair is deviated by ±1.5 %± 2 Hz.
7. Bandwidth limited (3 kHz) Gaussian noise.
8. The precise dial tone frequencies are (350 Hz and 440 Hz) ± 2 %.
9. For an error rate of better than 1 in 10,000.
10. Referenced to lowest level frequency component in DTMF signal.
11. Referenced to the minimum valid accept level.
12. Guaranteed by design and characterization.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

MT8870D-1 AC Electrical Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ T_O ≤ +85°C , using Test Circuit shown in Figure 10.

	Characteristics	Sym	Min	Typ‡	Max	Units	Notes*
1	Valid input signal levels (each tone of composite signal)		-31		+1	dBm	Tested at V_{DD}=5.0V 1,2,3,5,6,9
			21.8		869	mV_{RMS}	
2	Input Signal Level Reject		-37			dBm	Tested at V_{DD}=5.0V 1,2,3,5,6,9
			10.9			mV_{RMS}	
3	Negative twist accept				8	dB	2,3,6,9,13
4	Positive twist accept				8	dB	2,3,6,9,13
5	Frequency deviation accept		±1.5%± 2 Hz				2,3,5,9
6	Frequency deviation reject		±3.5%				2,3,5,9
7	Third zone tolerance			-18.5		dB	2,3,4,5,9,12
8	Noise tolerance			-12		dB	2,3,4,5,7,9,10
9	Dial tone tolerance			+22		dB	2,3,4,5,8,9,11

‡ Typical figures are at 25 °C and are for design aid only: not guaranteed and not subject to production testing.

*NOTES
1. dBm= decibels above or below a reference power of 1 mW into a 600 ohm load.
2. Digit sequence consists of all DTMF tones.
3. Tone duration= 40 ms, tone pause= 40 ms.
4. Signal condition consists of nominal DTMF frequencies.
5. Both tones in composite signal have an equal amplitude.
6. Tone pair is deviated by ±1.5 %± 2 Hz.
7. Bandwidth limited (3 kHz) Gaussian noise.
8. The precise dial tone frequencies are (350 Hz and 440 Hz) ± 2 %.
9. For an error rate of better than 1 in 10,000.
10. Referenced to lowest level frequency component in DTMF signal.
11. Referenced to the minimum valid accept level.
12. Referenced to Fig. 10 input DTMF tone level at -25dBm (-28dBm at GS Pin) interference frequency range between 480-3400Hz.
13. Guaranteed by design and characterization.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

AC Electrical Characteristics - V_{DD}=5.0V±5%, V_{SS}=0V, -40°C ≤ To ≤ +85°C , using Test Circuit shown in Figure 10.

		Characteristics	Sym	Min	Typ‡	Max	Units	Conditions
1	TIMING	Tone present detect time	t_{DP}	5	11	14	ms	Note 1
2		Tone absent detect time	t_{DA}	0.5	4	8.5	ms	Note 1
3		Tone duration accept	t_{REC}			40	ms	Note 2
4		Tone duration reject	$t_{\overline{REC}}$	20			ms	Note 2
5		Interdigit pause accept	t_{ID}			40	ms	Note 2
6		Interdigit pause reject	t_{DO}	20			ms	Note 2
7	OUTPUTS	Propagation delay (St to Q)	t_{PQ}		8	11	µs	TOE=V_{DD}
8		Propagation delay (St to StD)	t_{PStD}		12	16	µs	TOE=V_{DD}
9		Output data set up (Q to StD)	t_{QStD}		3.4		µs	TOE=V_{DD}
10		Propagation delay (TOE to Q ENABLE)	t_{PTE}		50		ns	load of 10 kΩ, 50 pF
11		Propagation delay (TOE to Q DISABLE)	t_{PTD}		300		ns	load of 10 kΩ, 50 pF
12	PDWN	Power-up time	t_{PU}		30		ms	Note 3
13		Power-down time	t_{PD}		20		ms	
14	CLOCK	Crystal/clock frequency	f_C	3.5759	3.5795	3.5831	MHz	
15*		Clock input rise time	t_{LHCL}			110	ns	Ext. clock
16		Clock input fall time	t_{HLCL}			110	ns	Ext. clock
17		Clock input duty cycle	DC_{CL}	40	50	60	%	Ext. clock
18		Capacitive load (OSC2)	C_{LO}			30	pF	

‡ Typical figures are at 25°C and are for design aid only: not guaranteed and not subject to production testing.

*NOTES:
1. Used for guard-time calculation purposes only.
2. These, user adjustable parameters, are not device specifications. The adjustable settings of these minimums and maximums are recommendations based upon network requirements.
3. With valid tone present at input, t_{PU} equals time from PDWN going low until ESt going high.

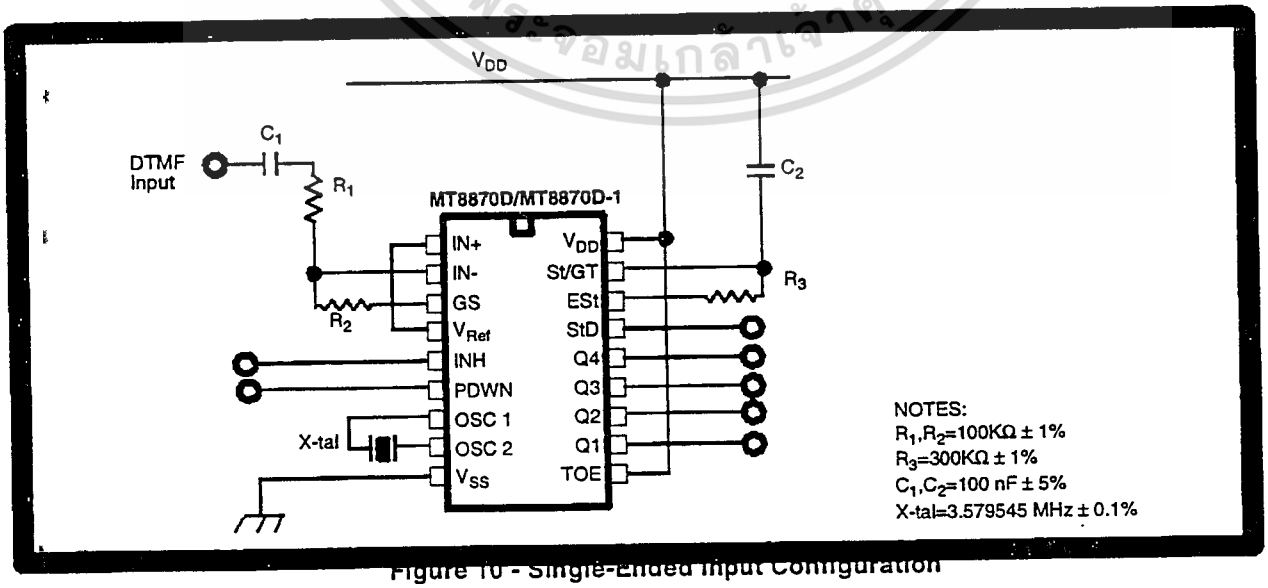
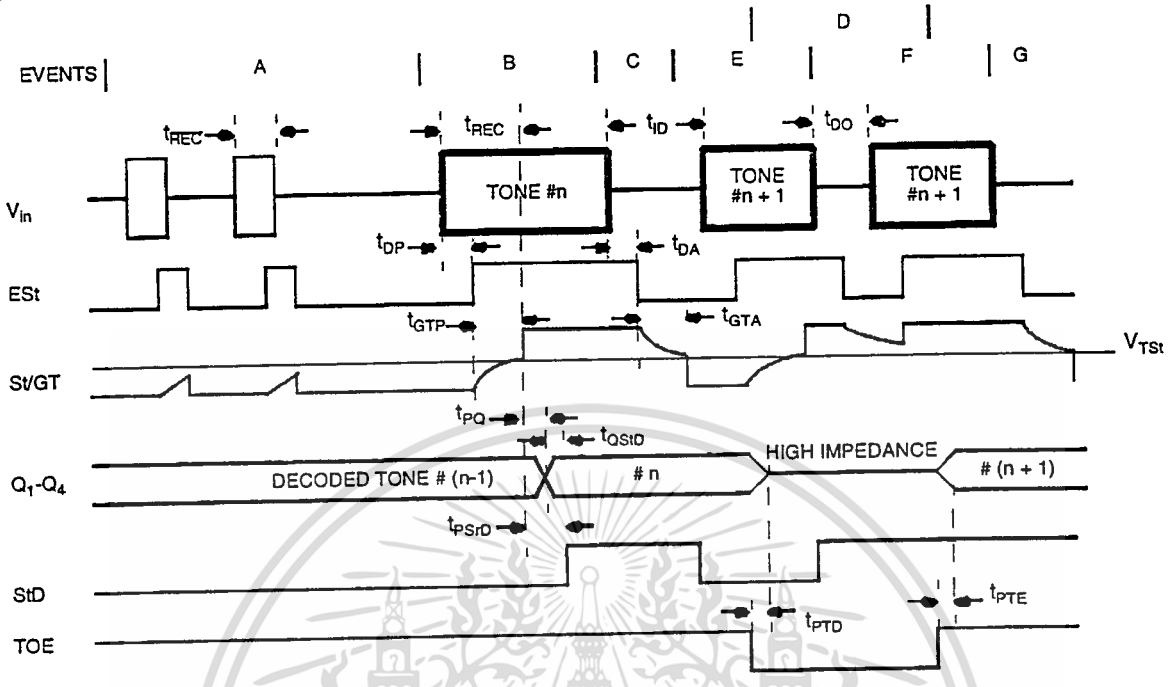


Figure 10 - Single-Ended Input Configuration

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้



EXPLANATION OF EVENTS

A) TONE BURSTS DETECTED, TONE DURATION INVALID, OUTPUTS NOT UPDATED.

B) TONE #n DETECTED, TONE DURATION VALID, TONE DECODED AND LATCHED IN OUTPUTS

C) END OF TONE #n DETECTED, TONE ABSENT DURATION VALID, OUTPUTS REMIAN LATCHED UNTIL NEXT VALID TONE.

D) OUTPUTS SWITCHED TO HIGH IMPEDANCE STATE.

E) TONE #n + 1 DETECTED, TONE DURATION VALID, TONE DECODED AND LATCHED IN OUTPUTS (CURRENTLY HIGH IMPEDANCE).

F) ACCEPTABLE DROPOUT OF TONE #n + 1, TONE ABSENT DURATION INVALID, OUTPUTS REMAIN LATCHED.

G) END OF TONE #n + 1 DETECTED, TONE ABSENT DURATION VALID, OUTPUTS REMAIN LATCHED UNTIL NEXT VALID TONE.

EXPLANATION OF SYMBOLS

V_{in} DTMF COMPOSITE INPUT SIGNAL.

ESt EARLY STEERING OUTPUT. INDICATES DETECTION OF VALID TONE FREQUENCIES.

St/GT STEERING INPUT/GUARD TIME OUTPUT. DRIVES EXTERNAL RC TIMING CIRCUIT.

Q_1-Q_4 4-BIT DECODED TONE OUTPUT.

StD DELAYED STEERING OUTPUT. INDICATES THAT VALID FREQUENCIES HAVE BEEN PRESENT/ABSENT FOR THE REQUIRED GUARD TIME THUS CONSTITUTING A VALID SIGNAL.

TOE TONE OUTPUT ENABLE (INPUT). A LOW LEVEL SHIFTS Q_1-Q_4 TO ITS HIGH IMPEDANCE STATE.

$t_{\overline{REC}}$ MAXIMUM DTMF SIGNAL DURATION NOT DETECED AS VALID

t_{REC} MINIMUM DTMF SIGNAL DURATION REQUIRED FOR VALID RECOGNITION

t_{ID} MAXIMUM TIME BETWEEN VALID DTMF SIGNALS.

t_{DO} MAXIMUM ALLOWABLE DROP OUT DURING VALID DTMF SIGNAL.

t_{DP} TIME TO DETECT THE PRESENCE OF VALID DTMF SIGNALS.

t_{DA} TIME TO DETECT THE ABSENCE OF VALID DTMF SIGNALS.

t_{GTP} GUARD TIME, TONE PRESENT.

t_{GTA} GUARD TIME, TONE ABSENT.

Figure 11 - Timing Diagram

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ISD1400 Series

Single-Chip Voice Record/Playback Devices 16- and 20-Second Durations

FEATURES

- Easy-to-use single-chip voice Record/ Playback solution
- High-quality, natural voice/audio reproduction
- Push-button interface
  - Playback can be edge- or level-activated
- Single-chip durations of 16 and 20 seconds
- Automatic power-down mode
  - Enters standby mode immediately following a Record or Playback cycle
  - Standby current 0.5 µA (typical)
- Zero-power message storage
  - Eliminates battery backup circuits
- Fully addressable to handle multiple messages
- 100-year message retention (typical)
- 100K record cycles (typical)
- On-chip clock source
- No programmer or development system needed
- Single +5 volt power supply
- Available in die form, DIP, and SOIC packaging
- Industrial temperature (–40°C to +85°C) versions available

ISD1400 SERIES SUMMARY

Part Number	Minimum Duration (Seconds)	Input Sample Rate (KHz)	Typical Filter Pass Band (KHz)
ISD1416	16	8.0	3.3
ISD1420	20	6.4	2.6

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1400_DS.FM Page 60 Monday, August 18, 1997 2:52 PM

GENERAL DESCRIPTION

Information Storage Devices' ISD1400 ChipCorder® Series provides high-quality, single-chip record/playback solutions to short-duration messaging applications. The CMOS devices include an on-chip oscillator, microphone preamplifier, automatic gain control, antialiasing filter, smoothing filter, and speaker amplifier. A minimum record/playback subsystem can be configured with a microphone, a speaker, several passives, two push-buttons, and a power source.

Recordings are stored in on-chip nonvolatile memory cells, providing zero-power message storage. This unique, single-chip solution is made possible through ISD's patented multilevel storage technology. Voice and audio signals are stored directly into memory in their natural form, providing high-quality, solid-state voice reproduction.

DETAILED DESCRIPTION

Speech/Sound Quality

The ISD1400 Series includes devices offered at 6.4 and 8.0 KHz sampling frequencies, allowing the user a choice of speech quality options. The speech samples are stored directly into on-chip nonvolatile memory without the digitization and compression associated with other solutions. Direct analog storage provides a very true, natural sounding reproduction of voice, music, tones, and sound effects not available with most solid-state digital solutions.

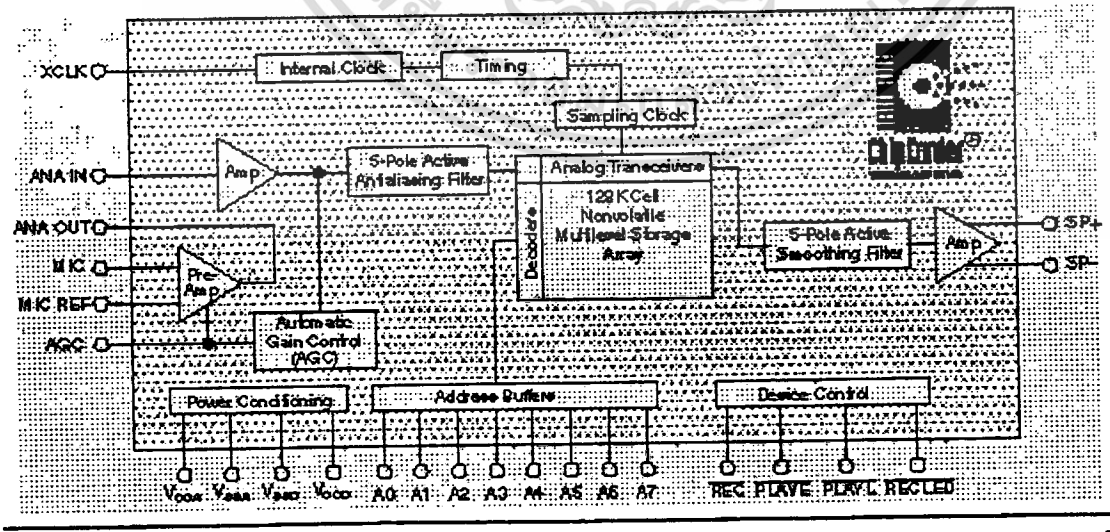
Duration

To meet end system requirements, the ISD1400 Series offers single-chip solutions at 16 and 20 seconds.

EEPROM Storage

One of the benefits of ISD's ChipCorder technology is the use of on-chip nonvolatile memory, providing zero-power message storage. The message is retained for up to 100 years typically without power. In addition, the device can be re-recorded typically over 100,000 times.

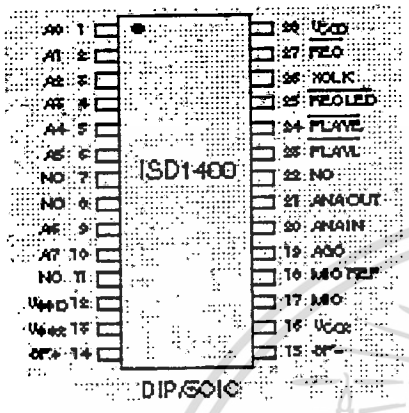
ISD1400 SERIES BLOCK DIAGRAM



เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1400_DS.FM Page 61 Monday, August 18, 1997 2:52 PM

ISD1400 SERIES PINOUTS



NOTE: NC means Must Not Connect.

The ISD1400 Series storage array has 160 distinct addressable segments, providing the following resolutions. See Chapter 5, Application Information for ISD1400 address tables.

Part Number	Minimum Duration (Seconds)
ISD1416	100 ms
ISD1420	125 ms

PIN DESCRIPTION

NOTE

The $\overline{REC}$ signal is debounced for 50 ms on the rising edge to prevent a false retriggering from a push-button switch.

Basic Operation

The ISD1400 ChipCorder Series devices are controlled by a single Record signal, $\overline{REC}$, and either of two push-button control playback signals, $\overline{PLAYE}$ (edge-activated playback), and $\overline{PLAYL}$ (level-activated playback). The ISD1400 parts are configured for simplicity of design in a single-message application. Using the address lines will allow multiple message applications. Device operation is explained on page 2-75.

Automatic Power-Down Mode

At the end of a Playback or Record cycle, the ISD1400 Series devices automatically return to a low-power standby mode, consuming typically 0.5 µA. During a Playback cycle, the device powers down automatically at the end of the message. During a Record cycle, the device powers down immediately after $\overline{REC}$ is released HIGH.

Addressing (optional)

In addition to providing simple message playback, the ISD1400 Series provides a full addressing capability.

Voltage Inputs (V_{CCA}, V_{CCD})

Analog and digital circuits internal to the ISD1400 Series use separate power buses to minimize noise on the chip. These power buses are brought out to separate pins on the package and should be tied together as close to the supply as possible. It is important that the power supply be decoupled as close as possible to the package.

Ground Inputs (V_{SSA}, V_{SSD})

Similar to V_{CCA} and V_{CCD}, the analog and digital circuits internal to the ISD1400 Series use separate ground buses to minimize noise. These pins should be tied together as close as possible to the device.

Record ($\overline{REC}$)

The $\overline{REC}$ input is an active-LOW Record signal. The device records whenever $\overline{REC}$ is LOW. This signal must remain LOW for the duration of the Recording. $\overline{REC}$ takes precedence over either Playback ($\overline{PLAYE}$ or $\overline{PLAYL}$) signal. If $\overline{REC}$ is pulled LOW during a Playback cycle, the Playback immediately ceases and Recording begins.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1400_DS.FM Page 62 Monday, August 18, 1997 2:52 PM

A Record cycle is completed when $\overline{REC}$ is pulled HIGH or the memory space is filled.

An end-of-message marker (EOM) is internally recorded, enabling a subsequent Playback cycle to terminate appropriately. The device automatically powers down to standby mode when $\overline{REC}$ goes HIGH.

Playback, Edge-Activated (PLA $\overline{YE}$)

When a LOW-going transition is detected on this input signal, a Playback cycle begins. Playback continues until an end-of-message (EOM) is encountered or the end of the memory space is reached. Upon completion of the Playback cycle, the device automatically powers down into standby mode. Taking $\overline{PLAYE}$ HIGH during a Playback cycle will not terminate the current cycle.

Playback, Level-Activated (PLA $\overline{YL}$)

When this input signal transitions from HIGH to LOW, a Playback cycle is initiated. Playback continues until $\overline{PLAYL}$ is pulled HIGH, an EOM marker is detected, or the end of the memory space is reached. The device automatically powers down to standby mode upon completion of the Playback cycle.

NOTE

In Playback, if either $\overline{PLAYE}$ or $\overline{PLAYL}$ is held LOW during EOM or OVERFLOW, the device will still enter standby and the internal oscillator and timing generator will stop. However, the rising edge of $\overline{PLAYE}$ and $\overline{PLAYL}$ are not debounced and any subsequent falling edge (particularly switch bounce) present on the input pins will initiate another Playback.

Record LED Output ($\overline{RECLED}$)

The output $\overline{RECLED}$ is LOW during a Record cycle. It can be used to drive an LED to provide feedback that a Record cycle is in progress. In addition, $\overline{RECLED}$ pulses LOW momentarily when an EOM is encountered in a Playback cycle.

Microphone Input (MIC)

The microphone input transfers its signal to the on-chip preamplifier. An on-chip Automatic Gain Control (AGC) circuit controls the gain of this preamplifier from -15 to 24 dB. An external microphone should be AC coupled to this pin via a series capacitor. The capacitor value, together with the internal 10 K Ohm resistance on this pin, determine the low-frequency cutoff for the ISD1400 Series passband. See Chapter 5, Application Information for additional information on low-frequency cutoff calculations.

Microphone Reference (MIC REF)

The MIC REF input is the inverting input to the microphone preamplifier. This provides a noise-canceling or common-mode rejection input to the device when connected differentially to a microphone.

Automatic Gain Control (AGC)

The AGC dynamically adjusts the gain of the preamplifier to compensate for the wide range of microphone input levels. The AGC allows the full range of sound, from whispers to loud sounds, to be recorded with minimal distortion. The "attack" time is determined by the time constant of a 5 KΩ internal resistance and an external capacitor (C6 on the schematic on page 2-75) connected from the AGC pin to V_{SSA} analog ground. The "release" time is determined by the time constant of an external resistor (R5) and an external capacitor (C6) connected in parallel between the AGC Pin and V_{SSA} analog ground. Nominal values of 470 KΩ and 4.7 µF give satisfactory results in most cases.

Analog Output (ANA OUT)

This pin provides the preamplifier output to the user. The voltage gain of the preamplifier is determined by the voltage level at the AGC pin.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1400_DS.FM Page 63 Monday, August 18, 1997 2:52 PM

Analog Input (ANA IN)

The ANA IN pin transfers the input signal to the chip for recording. For microphone inputs, the ANA OUT pin should be connected via an external capacitor to the ANA IN pin. This capacitor value, together with the 3.0 KΩ input impedance of ANA IN, is selected to give additional cutoff at the low-frequency end of the voice passband. If the desired input is derived from a source other than a microphone, the signal can be fed, capacitively coupled, into the ANA IN pin directly.

External Clock Input (XCLK)

The external clock input for the ISD1400 devices has an internal pull-down device. The ISD1400 is configured at the factory with an internal sampling clock frequency that guarantees its minimum nominal record/playback time. For instance, an ISD1420 operating within specification will be observed to always have a minimum of 20 seconds of recording time. The sampling frequency is then maintained to a variation of ±2.25% over the commercial temperature and operating voltage ranges, while still maintaining the minimum specified recording duration. This will result in some devices having a few percent more than nominal recording time.

The internal clock has a ±5% tolerance over the industrial temperature and voltage range. A regulated power supply is recommended for industrial temperature parts. If greater precision is required, the device can be clocked through the XCLK pin as follows:

Part Number	Sample Rate	Required Clock
ISD1416	8.0 KHz	1024 KHz
ISD1420	6.4 KHz	819.2 KHz

These recommended clock rates should not be varied because the antialiasing and smoothing filters are fixed, and aliasing problems can occur if the sample rate differs from the one recommended. The duty cycle on the input clock is not critical, as the clock is immediately divided by two internally. ***IF THE XCLK IS NOT USED, THIS INPUT SHOULD BE CONNECTED TO GROUND.***

Speaker Outputs (SP+, SP–)

The SP+ and SP– pins provide direct drive for loudspeakers with impedances as low as 16 ohms. A single output may be used, but, for direct-drive loudspeakers, the two opposite-polarity outputs provide an improvement in output power of up to four times over a single-ended connection. Furthermore, when SP+ and SP– are used, a speaker-coupling capacitor is not required. A single-ended connection will require an AC-coupling capacitor between the SP pin and the speaker. The speaker outputs are in a high-impedance state during a record cycle, and held at V_{SSA} during Power Down.

Address Inputs (A0–A7)

The Address Inputs have two functions, depending upon the level of the two Most Significant Bits (MSB) of the address.

If either of the two MSBs is LOW, the inputs are ***ALL*** interpreted as address bits and are used as the start address for the current Record or Playback cycle. The address pins are inputs only and do not output internal address information as the operation progresses. Address inputs are latched by the falling edge of $\overline{PLAYE}$, $\overline{PLAYL}$, or $\overline{REC}$.

OPERATIONAL MODES

The ISD1400 Series is designed with several built-in operational modes provided to allow maximum functionality with a minimum of additional components, described in detail below. The operational modes use the address pins on the ISD1400 devices, but are mapped outside the valid address range. When the two Most Significant Bits

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

(MSBs) are HIGH (A6 and A7), the remaining address signals are interpreted as mode bits and not as address bits. Therefore, operational modes and direct addressing are ***NOT*** compatible and cannot be used simultaneously.

There are two important considerations for using operational modes. First, all operations begin initially at address 0, which is the beginning of the ISD1400 address space. Later operations can begin at other address locations, depending on the operational mode(s) chosen. In addition, the address pointer is reset to 0 when the device is changed from Record to Playback but not from Playback to Record when A4 is HIGH in Operational Mode.

Second, an Operational Mode is executed when any of the control inputs, $\overline{\text{PLAYE}}$, $\overline{\text{PLAYL}}$, or $\overline{\text{REC}}$, go LOW and the two MSBs are HIGH. This Operational Mode remains in effect until the next LOW-going control input signal, at which point the current address/mode levels are sampled and executed.

NOTE

The two MSBs are on pins 9 and 10 for each ISD1400 Series device.

OPERATIONAL MODES DESCRIPTION

The Operational Modes can be used in conjunction with a microcontroller, or they can be hard-wired to provide the desired system operation.

A0 — Message Cueing

Message Cueing allows the user to skip through messages, without knowing the actual physical addresses of each message. Each control input LOW pulse causes the internal address pointer to skip to the next message. This mode should be used for Playback only, and is typically used with the A4 Operational Mode.

A1 — Delete EOM Markers

The A1 Operational Mode allows sequentially recorded messages to be combined into a single message with only one EOM marker set at the end of the final message. When this operational mode is configured, messages recorded sequentially are played back as one continuous message.

OPERATIONAL MODES TABLE

Address Ctrl (HIGH)	Function	Typical Use	Jointly Compatible*
A0	Message cueing	Fast-forward through messages	A4
A1	Delete EOM markers	Position EOM marker at the end of the last message	A3, A4
A2	Unused		
A3	Looping	Continuous playback from Address 0	A1
A4	Consecutive addressing	Record/Play multiple consecutive messages	A0, A1
A5	Unused		

NOTE: An asterisk (*) indicates additional operational modes which can be used simultaneously with the given mode.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

1400_DS.FM Page 65 Monday, August 18, 1997 2:52 PM

A2 — Unused

A3 — Message Looping

The A3 Operational Mode allows for the automatic, continuously repeated playback of the message located at the beginning of the address space.

A message can completely fill the ISD1400 device and will loop from beginning to end. Pulsing $\overline{\text{PLAYE}}$ will start the Playback and pulsing $\overline{\text{PLAYL}}$ will end the Playback.

A4 — Consecutive Addressing

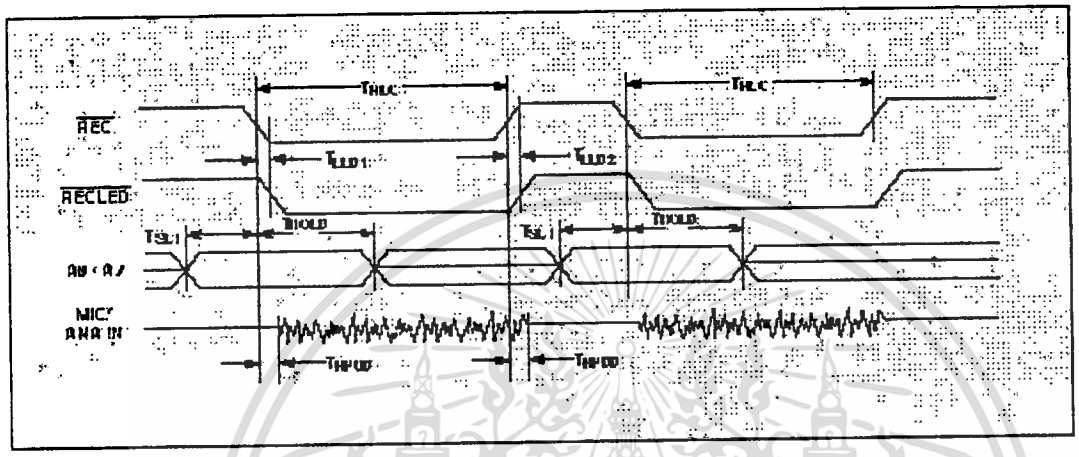
During normal operations, the address pointer will reset when a message is played through to an EOM marker. The A4 Operational Mode inhibits the address pointer reset, allowing messages to be recorded or played back consecutively. When the device is in a static state; i.e., not recording or playing back, momentarily taking this pin LOW will reset the address counter to zero.

A5 — Unused

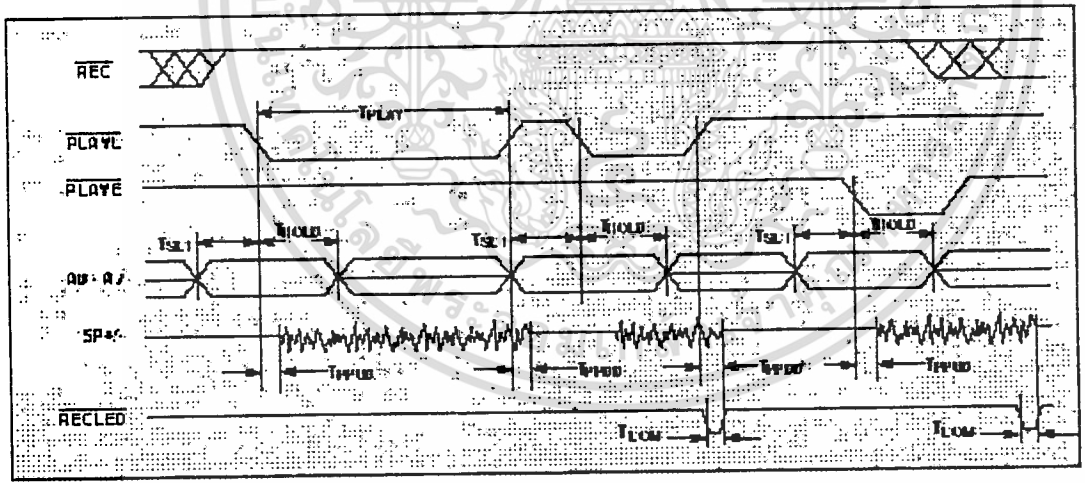
เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการ
ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

TIMING DIAGRAMS

Record



Playback



NOTES: 1. $\overline{\text{REC}}$ must be HIGH for the entire duration of a Playback cycle.
2. $\overline{\text{RECLED}}$ functions as an EOM during Playback.

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

ABSOLUTE MAXIMUM RATINGS (PACKAGED PARTS)

Condition	Value
Junction temperature	150° C
Storage temperature range	–65° C to +150° C
Voltage applied to any pin	(V_{SS} –0.3 V) to (V_{CC} + 0.3 V)
Voltage applied to any pin (Input current limited to ±20 mA)	(V_{SS} –1.0 V) to (V_{CC} + 1.0 V)
Lead temperature (soldering – 10 seconds)	300° C
V_{CC} - V_{SS}	–0.3 V to + 7.0 V

NOTE: Stresses above those listed may cause permanent damage to the device. Exposure to the absolute maximum ratings may affect device reliability. Functional operation is not implied at these conditions.

OPERATING CONDITIONS (PACKAGED PARTS)

Condition	Value
Commercial operating temperature range(1)	0° C to +70° C
Industrial operating temperature(1)	–40° C to +85° C
Supply voltage (V_{CC})(2)	+4.5 V to +5.5 V
Ground voltage (V_{SS})(3)	0 V

NOTES: 1. Case temperature.
2. $V_{CC} = V_{CCA} = V_{CCD}$.
3. $V_{SS} = V_{SSA} = V_{SSD}$.

DC PARAMETERS (PACKAGED PARTS)

Symbol	Parameters	Min(2)	Typ(1)	Max(2)	Units	Conditions
V_{IL}	Input Low Voltage			0.8	V	
V_{IH}	Input High Voltage	2.4			V	
V_{OL}	Output Low Voltage			0.4	V	I_{OL} = 4.0 mA
V_{OH}	Output High Voltage	2.4			V	I_{OH} = – 1.6 mA
I_{CC}	V_{CC} Current (Operating)		15	30	mA	V_{CC} = 5.5 V (3), R_{EXT} = ∞
I_{SB}	V_{CC} Current (Standby)		0.5	10	µA	(3) (4)
I_{IL}	Input Leakage Current			±1	µA	
I_{ILPD}	Input Current HIGH w/Pull Down			130	µA	Force V_{CC} (5)
R_{EXT}	Output Load Impedance	16			Ω	Speaker Load
R_{MIC}	Preamp In Input Resistance	4	9	17	KΩ	Pins 17, 18

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

กิตติกรรมประกาศ

โครงงานนี้สำเร็จได้ด้วยความช่วยเหลือจากบุคคลดังต่อไปนี้

ขอขอบคุณ

1. ผศ.ดร. ไกรสิน ส่งวัฒนา ที่ให้คำปรึกษา, เอื้อเฟื้ออุปกรณ์และสถานที่ทำงาน
2. คุณ เสกสรร ชื่นเคียนคต ที่ให้ความเอื้อเฟื้อใน การยืมอุปกรณ์และให้คำแนะนำ
3. คุณ จิราภรณ์ หลอดแก้วทองแดง ที่ให้ความช่วยเหลือในการบันทึกเสียงพูด
4. คุณ กอบกุญ โชติเรืองประเสริฐ ที่ให้ความช่วยเหลือในการพิสูจน์อักษร

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้

บรรณานุกรม

[1] ธวัชชัย เลื่อนฉวี " เทคโนโลยีโทรศัพท์ " สำนักพิมพ์บันเทิงการพิมพ์ พิมพ์ครั้งที่ 3,พ.ศ. 2533

[2] " โทรศัพท์แบบกดปุ่ม " วารสารเซมิคอนดัคเตอร์ อิเลคทรอนิคส์ ปีที่ 3 ฉบับที่ 6 , มิถุนายน พ.ศ. 2521

[3] ผศ.สมยศ จุณณะปิยะ , การใช้งานไมโครคอนโทรลเลอร์ ตระกูล MCS-51,พิมพ์ครั้งที่1 ,พ.ศ.2537

[4] สุนทร วิทูสุรพจน์,การใช้งานไมโครคอนโทรลเลอร์ตระกูล8051,สำนักพิมพ์ซีเอ็ดยูเคชั่นจำกัด(มหาชน) , พ.ศ.2537

[5] สุนทร วิทูสุรพจน์ , การโปรแกรมภาษาแอสเซมบลี ของ ไมโครคอนโทรลเลอร์ตระกูล8051,สำนักพิมพ์ซีเอ็ดยูเคชั่นจำกัด(มหาชน) , พ.ศ.2537

[6] ปรเมษฐ์ ประนยานันทน์ , ปิยพงศ์ เผ่าวณิช, คู่มือและการประยุกต์ใช้งานไมโครคอนโทรลเลอร์ MCS-51 , สำนักพิมพ์ซีเอ็ดยูเคชั่นจำกัด(มหาชน), พ.ศ.2536

[7] อ.บัณฑิต จามรภูติ , คู่มือการใช้ Protel for Windows , สำนักพิมพ์ซีเอ็ดยูเคชั่นจำกัด(มหาชน) , พิมพ์ครั้งที่1 , พ.ศ.2540

[8] ศิริชัย บุญบางยาง , วิโรจน์ พลสูงเนิน , ถวิล พึ่งมา , มนูญ สุขเกษม , " เครื่องบันทึกการใช้งานโทรศัพท์อัตโนมัติ " การประชุมวิชาการทางวิศวกรรมไฟฟ้าครั้งที่ 18 , หน้า 548 - 555 , พฤศจิกายน พ.ศ. 2538

เอกสารนี้เป็นเอกสารที่สงวนไว้สำหรับการใช้งานเพื่อการศึกษาเท่านั้น ไม่อนุญาตให้นำไปใช้ประโยชน์ด้านการค้า ไม่ว่ากรณีใดๆ ทั้งสิ้น อีกทั้งห้ามมิให้ดัดแปลงเนื้อหา และต้องอ้างอิงถึงเจ้าของเอกสารทุกครั้งที่มีการนำไปใช้