

# SKRIPSI

**PEMBUATAN ALAT PENGATUR DAN PENAMPIL LAYAR  
DOT Matrik 16 KARAKTER YANG BISA DI-REMOTE  
SECARA WIRELESS BERBASIS MIKROKONTROLER  
AT89S8252**



Disusun Oleh :

**MUHAMMAD FITHRONI**

**NIM : 03.17.087**

**JURUSAN TEKNIK ELEKTRO S - 1  
KONSENTRASI ELEKTRONIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI NASIONAL MALANG  
MARET 2008**

WORLD BANK  
INTERNATIONAL DEVELOPMENT ASSOCIATION  
ECONOMIC DEVELOPMENT FINANCING  
RECONSTRUCTION ESTABLISHMENT  
WORLD BANK GROUP 2 - 1

WORLD BANK  
INTERNATIONAL DEVELOPMENT ASSOCIATION  
ECONOMIC DEVELOPMENT FINANCING

WORLD BANK  
INTERNATIONAL DEVELOPMENT ASSOCIATION  
ECONOMIC DEVELOPMENT FINANCING  
RECONSTRUCTION ESTABLISHMENT  
WORLD BANK GROUP 2 - 1

WORLD BANK

# LEMBAR PERSETUJUAN

**PEMBUATAN ALAT PENGATUR DAN PENAMPIL LAYAR DOT  
MATRIK 16 KARAKTER YANG BISA DI-REMOTE SECARA WIRELESS  
BERBASIS MIKROKONTROLER AT89S8252**

## SKRIPSI

*Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh  
Gelar Sarjana Teknik Elektronika Strata Satu (S-1)*

**Disusun Oleh :**

**MUHAMMAD FITHRONI**

**NIM : 03.17.087**

**Diperiksa dan Disetujui**

**Dosen Pembimbing I**

**Dosen Pembimbing II**

**Ir. F. Yudi Limpraptono, MT**  
**NIP.Y 1039500274**

**Joseph Dedy Irawan, ST. MT**  
**NIP.132315178**

**Mengetahui**

**Ketua Jurusan Teknik Elektro S-1**



**Ir. F. Yudi Limpraptono, MT**  
**NIP.Y 1039500274**



**JURUSAN TEKNIK ELEKTRO S-1  
KONSENTRASI TEKNIK ELEKTRONIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI NASIONAL MALANG**

**2008**





INSTITUT TEKNOLOGI NASIONAL  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO S-1  
KONSENTRASI TEKNIK ELEKTRONIKA

## BERITA ACARA UJIAN SKRIPSI FAKULTAS TEKNOLOGI INDUSTRI

Nama : Muhammad Fithroni  
NIM : 03.17.087  
Jurusan : Teknik Elektro S-1  
Konsentrasi : Teknik Elektronika  
Judul Skripsi : Pembuatan Alat Pengatur dan Penampil Layar Dot Matrik 16 Karakter yang Bisa Di-Remote Secara *Wireless* Berbasis Mikrokontroler AT89S8252

Dipertahankan di hadapan majelis penguji Skripsi jenjang Strata satu ( S-1 ) pada :

Hari : Sabtu

Tanggal : 15 Maret 2008

Dengan Nilai : 81,8 (A) *Fuf*



**Ketua Majelis Penguji**

**(Ir. Mochtar Asroni, MSME)**  
NIP.Y.1018100036

**Sekretaris Majelis Penguji**

**(Ir. F. Yudi Limpraptono, MT)**  
NIP.Y.1039500274

**Penguji I**

**(Ir. M. Abdul Hamid, MT)**  
NIP. Y. 1018800188

**Penguji II**

**(I Komang Somawirata, ST. MT)**  
NIP.P. 1030100361



## ABSTRAKSI

### PEMBUATAN ALAT PENGATUR DAN PENAMPIL LAYAR DOT Matrik 16 Karakter yang Bisa Di-Remote Secara Wireless Berbasis Mikrokontroler AT89S8252

Muhammad Fithroni

03.17.087

Jurusan Teknik Elektronika – Institut Teknologi Nasional Malang

Jln. Raya Karanglo Km 2 Malang

[onny\\_guk@yahoo.com](mailto:onny_guk@yahoo.com)

Dosen Pembimbing : I. Ir. F. Yudi Limpraptono MT.

II. Joseph Dedy Irawan ST, MT.

**Kata Kunci** : dot matrik, 16 karakter, *remote*, *wireless*, mikrokontroler, AT89S8252.

Pada umumnya Layar Dot Matrik diletakkan pada dinding, hal ini ditujukan agar jangkauan pembacaan informasi yang ditampilkan lebih luas. Posisi penempatan layar ini menimbulkan suatu masalah, yaitu adanya kesulitan apabila informasi tersebut akan diganti.

Alat yang dirancang ini merupakan salah satu alat penampil informasi yang bersifat fleksibel. Alat ini terdiri dari layar dot matrik dan remote sebagai pengendalinya. Layar dot matrik mampu menampilkan 16 buah karakter dalam satu kali tampil dengan 2 jenis animasi yaitu layar bergeser dan berkedip. Pengendalian jenis karakter dan animasi dioperasikan secara wireless dengan bantuan remote. Adapun masukan jenis karakter pada sistem remote didapatkan dari karakter keyboard PC PS/2.

Sistem komunikasi wireless yang digunakan menggunakan modul TLP dan RLP434A yang sudah tersedia di pasaran. Sedangkan pengendali sistem remote dan layar menggunakan mikrokontroler AT89S8252. Mikrokontroler ini memiliki on-chip EEPROM, sehingga semua jenis karakter dan animasi yang telah tersimpan di memori ini tidak akan hilang walaupun catu daya dihilangkan. Dari hasil pengujian, disimpulkan bahwa remote mampu mengendalikan layar dot matrik pada jarak maksimal 9 meter dan layar dot matrik mampu menampilkan semua karakter dan animasi sesuai dengan data yang dikirim oleh remote.

## **KATA PENGANTAR**

Alhamdulillah, puji syukur kehadiran-Mu Ya Allah yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat menyelesaikan skripsi ini dengan lancar.

Keberhasilan penyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terima kasih kepada :

1. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1 dan Dosen Pembimbing I.
2. Bapak Joseph Dedy Irawan, ST, MT selaku Dosen Pembimbing II dan Ka. Laboratorium Elektronika Digital.
3. Ayah dan Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
4. Rekan-rekan Instruktur di Laboratorium Perancangan Elektronika.
5. Semua yang telah membantu dalam penyelesaian penyusunan skripsi ini.

Harapan penyusun semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Maret 2008

Penyusun

## DAFTAR ISI

<b>LEMBAR PERSETUJUAN</b> .....	i
<b>ABSTRAKSI</b> .....	ii
<b>KATA PENGANTAR</b> .....	iii
<b>DAFTAR ISI</b> .....	iv
<b>DAFTAR GAMBAR</b> .....	viii
<b>DAFTAR TABEL</b> .....	x
<b>BAB I PENDAHULUAN</b> .....	1
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	2
1.3. Tujuan .....	2
1.4. Batasan Masalah .....	2
1.5. Metodologi.....	3
1.6. Sistematika Penulisan .....	4
<b>BAB II TEORI DASAR</b> .....	6
2.1.Keyboard.....	5
2.2. Mikrokontroler AT89S8252.....	7
2.2.1. Sistem Mikrokontroler AT89S8252 .....	6
2.2.2. Keluarga mikrokontroler AT89S8252 .....	7
2.2.3. Arsitektur AT89S8252.....	8
2.2.4. Fungsi Pin Mikrokontroller AT89S8252.....	10
2.2.5. Accumulator ( A atau ACC ).....	14



2.2.6. Register .....	14
2.2.7. SP ( Stack Pointer ) .....	15
2.2.8. DATA POINTER (DPTR).....	15
2.2.9. Power Control (PCON).....	15
2.2.10. SFR ( <i>Special Function Register</i> ) .....	16
2.2.11. TIMER .....	17
2.2.12. Program Status Word ( PSW ).....	22
2.2.13. Interrupt.....	23
2.2.14. Organisasi Memory.....	24
2.2.15. Data Memory (EEPROM) dan RAM .....	25
2.3. LCD M16x2 .....	26
2.4. TLP434A dan RLP434A .....	30
2.5. Transistor sebagai <i>switching</i> .....	31
2.6. IC LM358 .....	33
<b>BAB III PERENCANAAN ALAT .....</b>	<b>34</b>
3.1. Pendahuluan .....	34
3.2. Perencanaan Perangkat Keras.....	39
3.2.1. Perangkat keras sistem <i>remote</i> .....	40
3.2.2. Perangkat keras sistem layar dot matrik .....	47
3.3. Perencanaan Perangkat Lunak.....	54
<b>BAB IV PENGUJIAN ALAT .....</b>	<b>82</b>
4.1. Pendahuluan .....	58
4.2. Pengujian <i>interface</i> keyboard PC dan LCD M16x2 .....	58

4.2.1. Tujuan pengujian rangkaian .....	58
4.2.1. Peralatan yang digunakan .....	59
4.2.3. Prosedur Pengujian .....	59
4.2.4. Hasil pengujian dan analisa .....	61
4.3. Pengujian Rangkaian TLP434A dan RLP434A.....	62
4.3.1. Tujuan pengujian rangkaian .....	62
4.3.2. Peralatan yang digunakan .....	62
4.3.3. Prosedur pengujian .....	62
4.3.4. Hasil Pengujian dan analisa.....	63
4.4. Pengujian <i>driver</i> dot matrik .....	64
4.4.1. Tujuan .....	64
4.4.2. Peralatan yang digunakan .....	64
4.4.3. Prosedur pengujian .....	64
4.4.4. Hasil pengujian dan analisa .....	66
4.5. Pengujian Rangkaian Layar Dot Matrik .....	70
4.5.1. Tujuan .....	70
4.5.2. Peralatan yang digunakan .....	70
4.5.3. Prosedur pengujian .....	70
4.5.4. Hasil pengujian dan analisa .....	71
4.6. Pengujian Sinyal Gelombang RLP434A.....	72
4.6.1. Tujuan .....	72
4.6.2. Peralatan yang digunakan .....	72
4.6.3. Prosedur pengujian .....	72

4.6.4. Hasil pengujian dan analisa .....	73
4.7. Pengujian Frekuensi Pembawa TLP434A .....	75
4.7.1. Tujuan .....	75
4.7.2. Peralatan yang digunakan .....	75
4.7.3. Prosedur pengujian .....	75
4.7.4. Hasil pengujian dan analisa .....	75
4.8. Pengujian Jangkauan <i>Wireless</i> .....	76
4.8.1. Tujuan .....	76
4.8.2. Peralatan yang digunakan .....	76
4.8.3. Prosedur pengujian .....	76
4.8.4. Hasil pengujian dan analisa .....	77
4.9. Pengujian Alat Keseluruhan .....	77
4.8.1. Tujuan .....	77
4.8.2. Peralatan yang digunakan .....	77
4.8.3. Prosedur pengujian .....	77
4.8.4. Hasil pengujian dan analisa .....	78
<b>BAB V PENUTUP .....</b>	<b>80</b>
5.1. Kesimpulan .....	80
5.2. Saran .....	80
<b>DAFTAR PUSTAKA .....</b>	<b>81</b>
<b>LAMPIRAN-LAMPIRAN</b>	



## DAFTAR GAMBAR

Gambar 2.1.	<i>Female socket</i> PS2 keyboard .....	5
Gambar 2.2.	Kode ASCII pada setiap tombol keyboard PC.....	6
Gambar 2.3.	Sinyal pada jalur data dan clk keyboard .....	6
Gambar 2.4.	Diagram blok AT89S8252 .....	9
Gambar 2.5.	Pin/kaki IC AT89S8252.....	10
Gambar 2.6.	Osilator Eksternal AT89S8252 .....	14
Gambar 2.7.	Timer 2 pada mode CAPTURE .....	19
Gambar 2.8.	Pada Mode Auto Reload DCEN=0 .....	20
Gambar 2.9.	Pada Mode Auto Reload DCEN=1 .....	20
Gambar 2.10.	Sebagai Baud Rate Generator .....	22
Gambar 2.11.	Ultra Small Transmitter (TLP434A).....	30
Gambar 2.12.	RLP 434A SAW Based Receiver.....	31
Gambar 2.13.	Transistor dalam keadaan Saturasi.....	32
Gambar 2.14.	Transistor dalam keadaan Cut Off .....	33
Gambar 2.15.	Low Power Dual Operational Amplifier.....	33
Gambar 3.1.	Diagram Blok Sistem .....	34
Gambar 3.2.	Rangkaian <i>interface</i> keyboard PC.....	40
Gambar 3.3.	Rangkaian minimum sistem mikrokontroler AT89S8252 .....	41
Gambar 3.4.	Rangkaian <i>clock</i> minimum sistem mikrokontroler AT89S8252.....	42
Gambar 3.5.	Rangkaian <i>power on reset</i> mikrokontroler AT89S8252 .....	43
Gambar 3.6.	Rangkaian LCD dengan mikrokontroler AT89S8252 .....	45
Gambar 3.7.	Rangkaian <i>interface</i> TLP434A dengan AT89S8252 .....	46
Gambar 3.8.	Rangkaian <i>interface</i> RLP434A dengan AT89S8252 .....	48
Gambar 3.9.	Rangkaian <i>interface</i> dot matrik.....	53
Gambar 3.10.	Diagram Alir <i>software</i> sistem <i>remote</i> .....	55
Gambar 3.11.	Diagram Alir <i>software</i> sistem layar dot matrik.....	56
Gambar 4.1.	Rangkaian pengujian <i>interface</i> keyboard PC dan LCD M16x2 .....	59
Gambar 4.2.	Foto alat pengujian <i>interface</i> keyboard PC dan LCD .....	60
Gambar 4.3.	Rangkaian pengujian modul TLP434A.....	62
Gambar 4.4.	Rangkaian pengujian modul RLP434A .....	63
Gambar 4.5.	Rangkaian pengujian saklar baris dot matrik .....	65

Gambar 4.6.	Rangkaian pengujian dot matrik .....	66
Gambar 4.7.	Hasil pengujian saklar dot matrik pada kondisi <i>cut off</i> .....	67
Gambar 4.8.	Hasil pengujian saklar dot matrik pada kondisi saturasi.....	67
Gambar 4.9.	Hasil pengujian Ib .....	68
Gambar 4.10.	Hasil pengujian dot matrik pada kondisi <i>on</i> .....	69
Gambar 4.11.	Hasil pengujian dot matrik pada kondisi <i>off</i> .....	69
Gambar 4.12.	Rangkaian pengujian modul dot matrik.....	71
Gambar 4.13.	Foto hasil pengujian rangkaian layar dot matrik.....	71
Gambar 4.14.	Rangkaian pengujian gelombang RLP434A.....	73
Gambar 4.15.	Hasil gelombang RLP434A sebelum rangkaian RC.....	74
Gambar 4.16.	Hasil gelombang RLP434A setelah rangkaian RC .....	74
Gambar 4.17.	Hasil gelombang RLP434A setelah penguatan IC LM358.....	75
Gambar 4.18.	Frekuensi pembawa TLP434A.....	76
Gambar 4.19.	Rangkaian pengujian alat keseluruhan.....	78
Gambar 4.20.	Informasi <i>remote</i> yang dikirim .....	78
Gambar 4.21.	Hasil pengujian alat keseluruhan .....	78

1	...	...	...
2	...	...	...
3	...	...	...
4	...	...	...
5	...	...	...
6	...	...	...
7	...	...	...
8	...	...	...
9	...	...	...
10	...	...	...
11	...	...	...
12	...	...	...
13	...	...	...
14	...	...	...
15	...	...	...
16	...	...	...
17	...	...	...
18	...	...	...
19	...	...	...
20	...	...	...
21	...	...	...
22	...	...	...
23	...	...	...
24	...	...	...
25	...	...	...
26	...	...	...
27	...	...	...
28	...	...	...
29	...	...	...
30	...	...	...
31	...	...	...
32	...	...	...
33	...	...	...
34	...	...	...
35	...	...	...
36	...	...	...
37	...	...	...
38	...	...	...
39	...	...	...
40	...	...	...
41	...	...	...
42	...	...	...
43	...	...	...
44	...	...	...
45	...	...	...
46	...	...	...
47	...	...	...
48	...	...	...
49	...	...	...
50	...	...	...



## DAFTAR TABEL

No.	Judul	Halaman
Tabel 2.1.	Keluarga Mikrokontoller MCS- 51 .....	8
Tabel 2.2	Fungsi Khusus Pada Port 1 AT89S8252.....	11
Tabel 2.3	Fungsi Alternarif Port 3 .....	12
Tabel 2.4	Letak SFR pada alamat 80H sampai FFH.....	17
Tabel 2.5	Mode Operasi Timer 2 .....	18
Tabel 2.6	T2MOD – Timer 2 Mode Control Register .....	19
Tabel 2.7	Program Status Word ( PSW) .....	22
Tabel 2.8	Pemilihan Bank Memori .....	23
Tabel 2.9	Interrupt Enable ( IE ) Register.....	24
Tabel 2.10	Pin-Pin LCD dan Fungsinya .....	29
Tabel 4.1	Hasil pengujian <i>interface</i> keyboard PC dan LCD M16x2 .....	61
Tabel 4.2	Hasil pengujian modul RLP434A dan TLP434A .....	64
Tabel 4.3	Hasil pengujian saklar baris dot matrik.....	68
Tabel 4.4	Hasil pengujian Ib .....	68
Tabel 4.5	Hasil pengujian dot matrik.....	70
Tabel 4.6	Hasil pengujian jangkauan <i>wireless</i> .....	77

# BAB I

## PENDAHULUAN

### 1.1. Latar Belakang

Layar dot matrik adalah layar yang terdiri dari sekumpulan dioda led yang dirangkai secara teratur, sehingga mampu menampilkan beberapa jenis karakter berdasarkan konfigurasi led yang hidup dan mati.

Layar ini berfungsi sebagai media penampil informasi yang berupa tulisan. Adapun informasi yang ditampilkan bisa berupa pengumuman ataupun iklan. Jenis karakter pada layar mampu diubah sesuai dengan jenis informasi yang akan ditampilkan.

Umumnya Layar Dot Matrik diletakkan pada dinding bagian atas, hal ini ditujukan agar jangkauan pembacaan informasi yang ditampilkan lebih luas. Posisi penempatan layar ini menimbulkan suatu masalah, yaitu adanya kesulitan apabila informasi tersebut akan diganti. Dengan adanya masalah tersebut, maka diperlukannya suatu alat yang dapat mempermudah pekerjaan tersebut, salah satu alternatifnya yaitu pengiriman data tanpa kabel (*wireless*). Pengiriman data ini dapat dilakukan dengan menggunakan komunikasi serial.

Keunggulan lain dari alat yang akan dirancang yaitu mampu menampilkan informasi yang berupa angka ataupun tulisan sebanyak 16 karakter, mudah diganti sesuai dengan keinginan pengguna dan praktis (mudah dibawa).

## 1.2. Rumusan Masalah

Berdasarkan latar belakang tersebut, maka permasalahan yang timbul adalah bagaimana pembuatan layar dot matrik 16 karakter yang dapat dikendalikan secara *wireless* berbasis mikrokontroler AT89S8252. Adapun rumusan masalah dari permasalahan tersebut adalah sebagai berikut :

- a) Bagaimana sistem remote layar dot matrik berbasis mikrokontroler AT89S8252 ?
- b) Bagaimana sistem layar dot matrik yang bisa menampilkan 16 karakter berbasis mikrokontroler AT89S8252?
- c) Bagaimana menghubungkan sistem remote dengan layar dot matrik via *wireless* ?

## 1.3. Tujuan

Tujuan dari penulisan skripsi ini adalah untuk membuat prototipe layar dot matrik 16 karakter yang bisa dikendalikan secara *wireless* berbasis mikrokontroler AT89S8252.

## 1.4. Batasan Masalah

Penulis membatasi ruang lingkup skripsi ini dalam batasan masalah. Adapun batasan masalah tersebut adalah sebagai berikut :

- a) Karakter maksimal yang mampu ditampilkan dalam 1 layar dot matrik adalah 16.
- b) Jenis animasi yang ditampilkan maksimal 2 jenis yaitu karakter berjalan, dan karakter berkedip.



- c) Mikrokontroler yang digunakan pada sistem alat pengatur layar dot matrik menggunakan AT89S8252.
- d) Mikrokontroler yang digunakan pada sistem layar dot matrik adalah AT89S8252.
- e) Modul wireless yang digunakan adalah TLP434A dan RLP434A.

### **1.5. Metodologi**

Metodologi yang dipakai dalam pembuatan skripsi ini adalah:

#### **1. Studi Literatur**

Mencari referensi-referensi yang berhubungan dengan perencanaan dan pembuatan alat yang akan dibuat.

#### **2. *Field Research***

Melakukan penelitian secara langsung mengenai objek-objek yang berhubungan langsung dengan perencanaan alat yang akan dibuat.

#### **3. *Design* dan Pembuatan Alat**

Membuat PCB, merakit komponen dan membuat perangkat lunak.

#### **4. Pengujian Alat**

Melakukan pengujian per-blok rangkaian dan kerja seluruh sistem pada alat tersebut di laboratorium Elektronika ITN Malang.

## **1.6. Sistematika Penulisan**

Penyusunan laporan skripsi ini secara garis besar terdiri dari lima bab, dengan rincian sebagai berikut :

### **BAB I : PENDAHULUAN**

Berisi tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penulisan dan sistematika pembahasan.

### **BAB II : TEORI DASAR**

Membahas tentang teori penunjang yang diperlukan sebagai acuan dalam perencanaan dan pembuatan alat.

### **BAB III : PERENCANAAN DAN PEMBUATAN ALAT**

Meliputi penjelasan blok diagram serta perencanaan dan pembuatan *hardware* dan *software*.

### **BAB IV : PENGUJIAN ALAT**

Membahas pengujian peralatan secara keseluruhan dan analisa hasil pengujian.

### **BAB V : PENUTUP**

Berisi tentang kesimpulan dan saran.

## BAB II

### TEORI PENUNJANG

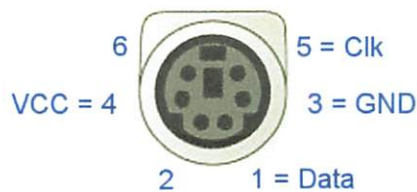
#### 2.1. Keyboard

Keyboard adalah alat input utama dari sebuah PC selain *mouse*. Melalui keyboard kita dapat memberikan instruksi kepada komputer untuk mengerjakan sesuatu yang kita inginkan.

Keyboard memuat tombol-tombol penting di antaranya :

- Tombol penulisan huruf.
- Tombol *enter* untuk mengeksekusi sebuah *action*.
- Tombol Esc (escape) untuk membatalkan sebuah *action*.
- Tombol-tombol fungsi khusus (F1 – F12).
- Tombol penulisan angka.
- Tombol kontrol sistem operasi (tombol windows, tombol right click menu).

Keyboard dikenal dalam 2 jenis, yaitu keyboard serial dan keyboard PS2 (baca PS Two). Yang membedakan hanyalah konektor kabelnya. Untuk yang serial berbentuk kotak, sedangkan yang PS2 berbentuk bulat.



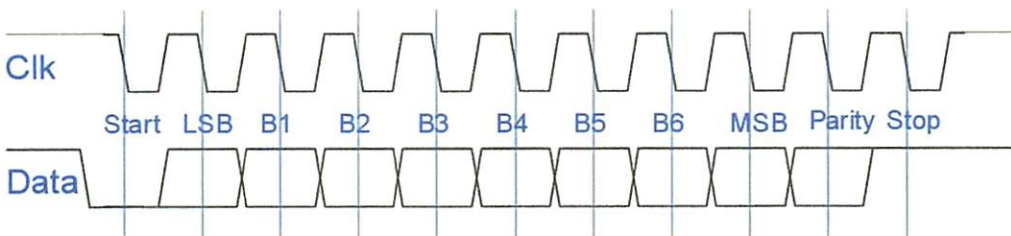
Gambar 2.1. *Female socket* PS2 keyboard<sup>[4]</sup>

Komputer bisa mengenal jenis tombol yang ditekan berdasarkan kode ASCII yang dikeluarkan dari jalur data keyboard. Setiap tombol tersebut memiliki kode ASCII yang berbeda-beda, sesuai dengan Gambar 2.2.



Gambar 2.2. Kode ASCII pada setiap tombol keyboard PC<sup>[4]</sup>

Komunikasi yang digunakan pada keyboard jenis PS2 adalah serial, adapun jumlah bit yang dikeluarkan tiap kali tombol yang ditekan adalah 9 bit. Bit pertama yang dikirim merupakan bit indikator dimulainya pengiriman kode ASCII. 8 bit berikutnya adalah angka biner yang dibentuk dari kode ASCII tiap tombol. Jalur “Clk” berfungsi sebagai sinyal indikator masuknya data, ketika sinyal “clk” berlogika “low” maka jenis sinyal yang keluar dari jalur data merupakan data sebenarnya. Adapun sinyal yang terbentuk pada jalur data dan clk keyboard sesuai dengan Gambar 2.3.



Gambar 2.3. Sinyal pada jalur data dan clk keyboard<sup>[4]</sup>

## **2.2. Mikrokontroler AT89S8252**

### **2.2.1. Sistem Mikrokontroler AT89S8252**

Mikrokontroler berbeda dengan mikroprosesor karena selain memiliki CPU juga dilengkapi dengan memori dan input-output yang merupakan kelengkapan sistem dalam mikrokomputer dalam keping tunggal (*Single Chip Mikrokomputer*) yang dapat berdiri sendiri.

### **2.2.2. Keluarga mikrokontroler AT89S8252**

Perbedaan mendasar antara mikrokontroler dan mikroprosesor adalah mikrokontroler selain memiliki CPU juga dilengkapi dengan memori input-output yang merupakan kelengkapan sebagai sistem minimum mikrokomputer sehingga sebuah mikrokontroler dapat dikatakan sebagai mikrokomputer dalam keping tunggal (*single chip Microcomputer*) yang dapat berdiri sendiri.

Mikrokontroler AT89S8252 adalah mikrokontroler ATMEL yang kompatibel penuh dengan mikrokontroler keluarga MCS-51, membutuhkan daya yang rendah, memiliki performa yang tinggi dan merupakan mikrokomputer 4 x 8 bit yang dilengkapi 8 Kbyte 2K EEPROM (*Erasable and Programable Read Only Memori*) dan 256 byte RAM internal. Program memori dapat diprogram ulang dalam sistem atau dengan menggunakan Program *Nonvolatelly Memory Konvensional*.

Dalam sistem mikrokontroler terdapat dua hal yang mendasar, yaitu: perangkat keras dan perangkat lunak yang keduanya saling terkait dan mendukung. Berikut ini adalah tabel keluarga mikrokontroler MCS- 51, dapat



dilihat bahwa mikrokontroller 8031 merupakan versi tanpa EPROM dari mikrokontroller 8051

**Tabel 2.1. Keluarga Mikrokontoller MCS- 51<sup>[1]</sup>**

PART NUMBER	ON- CHIP CODE MEMORY	ON CHIP DATA MEMORY	TIMER
8051	4K ROM	128 BYTES	2
8031	0K	128 BYTES	2
8751	4K EOROM	128 BYTES	2
8052	8KROM	256 BYTES	3
8032	0K	256 BYTES	3
8752	8KEPROM	256 BYTES	3
AT 89S8252	8K EPROM	256 BYTES	3

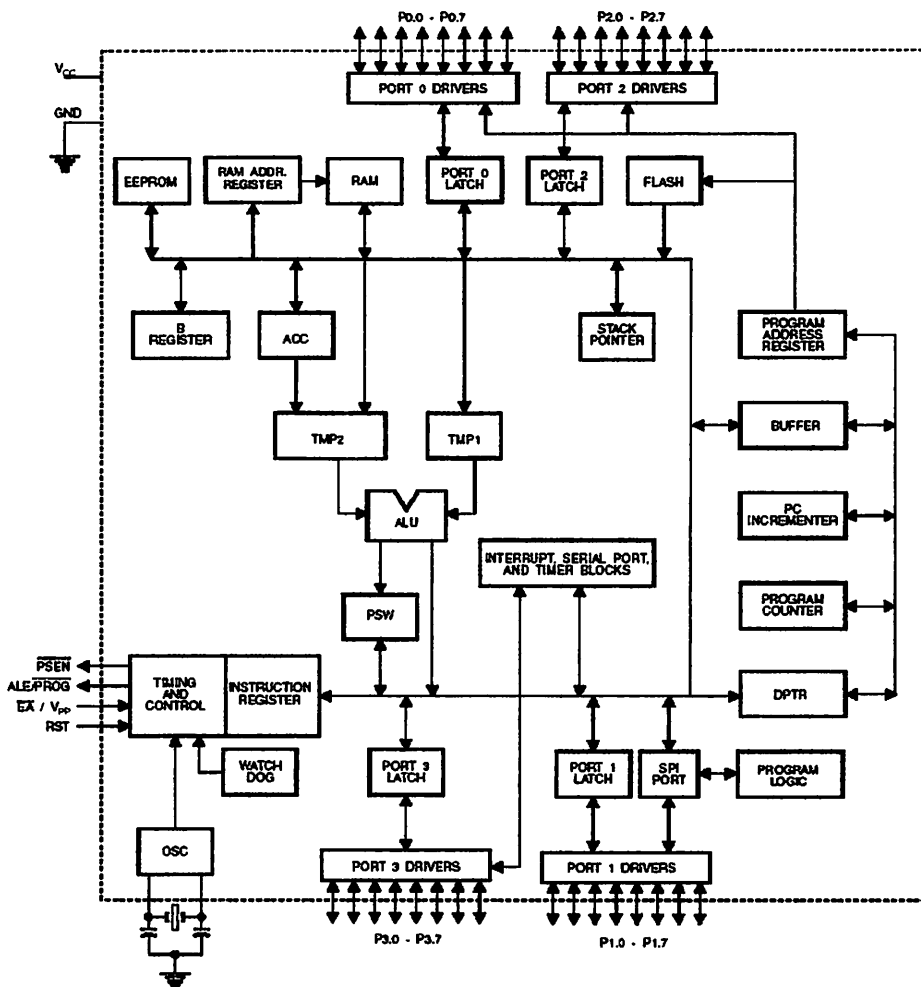
### 2.2.3. Arsitektur AT89S8252

Sebagai *single chip* yaitu suatu sistem mikroprosesor yang terintegrasi, mikrokontroller AT89S8252 mempunyai konfigurasi sebagai berikut:

1. Kompatibel dengan mikrokontroler MCS-51
2. 8 Kbyte Downloadable Flas Memory
3. 2 Kbyte EEPROM
4. 3 level program memori lock
5. 256 byte RAM internal
6. 32 pin I/O yang dapat dipakai semua

7. 3 buah timer/ counter 16 bit.
8. Programmable Watchdog Timer
9. Dual Data Pointer
10. Frekuensi kerja 0 sampai 24 MHz
11. Tegangan operasi 2,7 Volt sampai 6 Volt

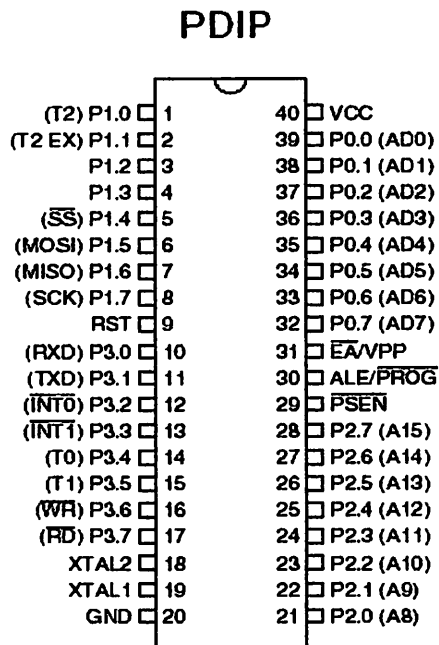
Arsitektur dasar dari mikrokontroler AT89S8252 seperti diagram blok berikut ini:



Gambar 2.4. Diagram blok AT89S8252<sup>[1]</sup>

## 2.2.4. Fungsi Pin Mikrokontroller AT89S8252

Susunan pin-pin mikrokontroller AT89S8952 diperlihatkan pada Gambar 2.2, dan penjelasan dari masing-masing pin adalah sebagai berikut:



Gambar 2.5. Pin/kaki IC AT89S8252<sup>[1]</sup>

### 1. Port 0 ( pin 32-39 )

Port 0 merupakan port 8 bit yang bersifat open drain dua arah. Sebagai port keluaran, tiap pena dapat menerima 8 masukan TTL. Saat logika 1 ditulis pada port, pena port dapat digunakan sebagai masukan dengan impedansi tinggi

### 2. Port 1 ( pin 1-8 )

Port 1 merupakan port I/O dua arah (bi-directional) yang telah dilengkapi dengan pull-up internal. Port ini dapat mendayai atau menerima 4 masukan TTL (Transistor-Transistor Logic). Jika suatu logika '1' dituliskan pada port ini, maka

port akan dibuat tinggi oleh pull-up internal dan dapat digunakan sebagai masukan, port ini akan dibuat rendah, (externally pulled low) dan port ini akan mendayai karena adanya pull-up internal.

**Tabel 2.2.** Fungsi Khusus Pada Port 1 AT89S8252<sup>[1]</sup>

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.4	$\overline{SS}$ (Slave port select input)
P1.5	MOSI (Master data output, slave data input pin for SPI channel)
P1.6	MISO (Master data input, slave data output pin for SPI channel)
P1.7	SCK (Master clock output, slave clock input pin for SPI channel)

### 3. Port 2 ( pin 21-28 )

Port 2 merupakan port paralel 8 bit yang bersifat arah dan memiliki pull-up internal. Penyangga pada port ini mampu menangani 4 masukan TTL. Jika logika 1 dituliskan pada port ini, maka port ini akan dibuat tinggi oleh pull-up internalnya. Port 2 mengirimkan byte tinggi dan alamat selama pengaksesan dari program memori luar selama penulisan ke data memori luar yang menggunakan alamat 16 bit seperti moxy@ dptr.

#### 4. Port 3 ( pin 10-17 )

Port 3 merupakan port 8 bit dua arah dengan pill-up internal. Keluaran dari port 3 ini dapat mendayai atau menerima masukan sebanyak 4 masukan TTL. Saat logika 1 dituliskan pada port ini, maka port ini akan dibuat tinggi oleh pull-up internalnya dan port ini dapat dipakai sebagai masukan. Selaian sebagai port paralel biasa, port 3 juga memiliki fungsi khusus. Fungsi khusus pada port 3 ini diperlihatkan pada table sbb :

**Tabel 2.3.** Fungsi Alternarif Port 3<sup>[1]</sup>

BIT	NAMA	BIT ADDRES	FUNGSI ALTERNATIF
P3.0	RXD	B0H	Penerima data pada port serial ( UART )
P3.1	TXD	B1H	Pemancar data pada port serial ( UART )
P3.2	INT0	B2H	Eksternal interupsi 0
P3.3	INT 1	B3H	Eksternal interuposi 1
P3.4	T0	B4H	Input Timer/ counter 0
P3.5	T1	B5H	Input Timer / counter 1
P3.6	WR	B6H	Sinyal pembacaan memori data eksternal
P3.7	RD	B7H	Sinyal penulisan memori data eksternal

#### 5. XTAL1 ( pin 18-19 )

Masukan untuk penguat inverting osilator dan masukan rangkaian clock internal.

#### 6. XTAL 2 ( pin 18-19 )

Keluaran dari penguat inverting osilator

### **7 RESET ( RST pin 9 )**

Masukan untuk RESET, suatu logika tinggi selama dua siklus pada pena RESET akan menyebabkan terjadinya proses reset.

### **8. GND ( pin 20 )**

Ground sistem

### **9. PSEN ( pin 29 )**

Program Store Enable adalah pulsa pengaktifan untuk membaca program memori luar. Saat mikrokontroler melaksanakan instruksi dari program memori memori luar, PSEN akan aktifkan dua kali tiap siklus mesin, kecuali saat mengakses data memori luar

### **10. ALE ( *Address Latch Enable pin 30* )**

Address Latch Enable merupakan suatu pulsa keluaran untuk mengaitkan ( Latch ) byte bawah dari alamat selama mengakses memori luar.

### **10. EA ( *External Acces pin 31* )**

External Access Enable, EA harus dihubungkan dengan ground jika ingin mengakses dari program memori luar dengan alamat 0000H, EA dihubungkan dengan Vcc jika menggunakan program memori internal.

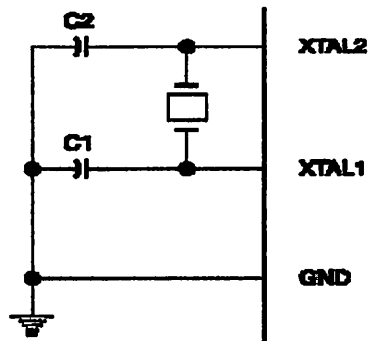
### **11.Power**

AT89S8252 dioperasikan dengan tegangan supply +5v, pin Vcc berada pada pin 40 dan Vss(*ground*) pada pin 20.

### **12. Osilator**

Pin ini dihubungkan dengan kristal bila menggunakan osilator internal. X-TAL 1 merupakan masukan ke rangkaian osilator internal sedangkan X-TAL 2

keluaran dari rangkaian osilator internal. Untuk keperluan ini diperlukan kapasitor penstabil sebesar 30 pF. Dan nilai dari X-TAL tersebut antara 4-24 Mhz. untuk lebih jelasnya dapat dilihat gambar pemasangan X-TAL serta kapasitor yang digunakannya.



Gambar 2.6. Osilator Eksternal AT89S8252<sup>[1]</sup>

### 2.2.5. Accumulator ( A atau ACC )

Dari namanya dapat diketahui bahwa fungsinya accumulator adalah pengumpulan. SFR ini merupakan register yang paling banyak dipakai. Untuk semua operasi aritmatika biasanya menggunakan / melibatkan accumulator. Alamat dari accumulator adalah E0H.

### 2.2.6. Register

B register di F0H. Register ini digunakan untuk operasi perkalian dan pembagian. Contoh instruksi yang menggunakan register ini :

Mul AB : kalikan isi pada accumulator dan pada B register

**MSB**

<b>SMOD</b>	-	-	-	<b>GF1</b>	<b>GF0</b>	<b>PD</b>	<b>IDL</b>
-------------	---	---	---	------------	------------	-----------	------------

**Keterangan**

Bit SMOD = digunakan untuk membuat dobel ( 2 kali ) baud rate pada timer

Bit - = tidak digunakan, untuk pengembangan selanjutnya

Bit GF1 = bit Flag serbaguna

Bit GF0 = bit serbaguna

Bit PD = bit power down. Bila berlogika 1 mode power down aktif

Bit IDL = bit idl mode. Aktif jika berlogika 1

**2.2.10. SFR (Special Function Register)**

SFR merupakan register dengan tugas khusus dengan alamat 80H sampai FFH sehingga terdapat 128 lokasi alamat untuk SFR. MCU AT 89S8252 selain memiliki SFR ( Special Finction Register ) seperti halnya pada mikrokontroler MCS-51, mikrokontroler Atmel AT 89S8252 memiliki tambahan SFR. Hal ini tak lain adalah karena terdapatnya tambahan fitur pada AT 89S8252.

SFR tambahan ini meliputi : T2CON ( Timer 2 Register dengan alamat 0C8H ), T2MOD ( Timer 2 Mode dengan alamat 0C9H ), WMCON ( Wacthdog and Memory Control Register dengan alamat 96H ), SPCR ( SPI Control Register dengan alamat D5H ), SPSR ( SPI Status Register dengan alamat AAH ), SPDR ( SPI Data Register dengan alamat 86H ).



Div AB : membagi isi accumulator dengan isi B register. Accumulator akan berisi hasil bagi dan B register akan berisi sisa pembagian.

### **2.2.7. SP (Stack Pointer)**

Stack pointer adalah penunjuk stack yang memiliki alamat di 81H. Isi register ini mengindikasikan dimana nilai selanjutnya yang harus diambil dari stack pointer pada RAM internal. Jika anda memasukkan (push) suatu nilai ke stack, nilai tersebut akan dituliskan pada alamat  $SP + 1$  (alamat SP ditambah satu). Jika SP berisi 07H kemudian suatu instruksi PUSH dijalankan, maka nilai yang dimasukkan akan dituliskan pada alamat 08H. Nilai default SP adalah 07H.

### **2.2.8. DATA POINTER (DPTR)**

Data pointer (DPTR) yang berukuran 16 bit terdiri dari dua register yaitu DPL (Data Pointer Low Byte) ber-alamat di 82H dan DPH (Data Pointer High Byte) ber-alamat di 83H. Data pointer digunakan untuk mengakses memori luar.

### **2.2.9. Power Control (PCON)**

Register Power Control ber-alamat di 87H berguna untuk mengatur kebutuhan daya mikrokontroler. Dengan adanya register pengaturan daya ini memungkinkan mikrokontroler ke mode "idle" atau "sleep" yang mana akan lebih menghemat pemakaian daya. Selain itu ada bit-bit pada register PCON ini untuk mengatur Baud Rate pada serial port. Bit-bit pada PCON adalah sebagai berikut :

**Table 2.4** Letak SFR pada alamat 80H sampai FFH<sup>[1]</sup>

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0E7H
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000					SPCR 000001XX		0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000		SPSR 00XXXXXX					0AFH
0A0H	P2 11111111							0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111					WMCN 00000010		97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000		8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR XXXXXXXX	PCON 0XX00000

### 2.2.11. TIMER

Pada mikrokontroler Atmel AT89S8252 terdapat tambahan timer 2. Timer lain adalah timer 0 dan timer 1. Timer 1 digunakan sebagai penghasil baud rate dan menjaga kompatibilitasi dengan MCS-51 juga dapat menggunakan timer 2 sebagai penghasil baud rate untuk serial port.

Timer 2 merupakan timer/counter yang berukuran 18 bit yang dapat beroperasi sebagai timer ( dengan detak dari system detak mikrokontroler ) atau dapat beroperasi sebagai penghitung kejadian ( event counter ). Timer memiliki 3

mode operasi yaitu capture, auto reload ( up dan down counting ) dan baud rate generator. Untuk memilih mode ini dilakukan dengan mengatur bit pada SFR T2CON.

**Tabel 2.5 Mode Operasi Timer 2<sup>[1]</sup>**

RCLK + TCLK	CP/RLZ	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud Rate Generator
X	X	0	(Off)

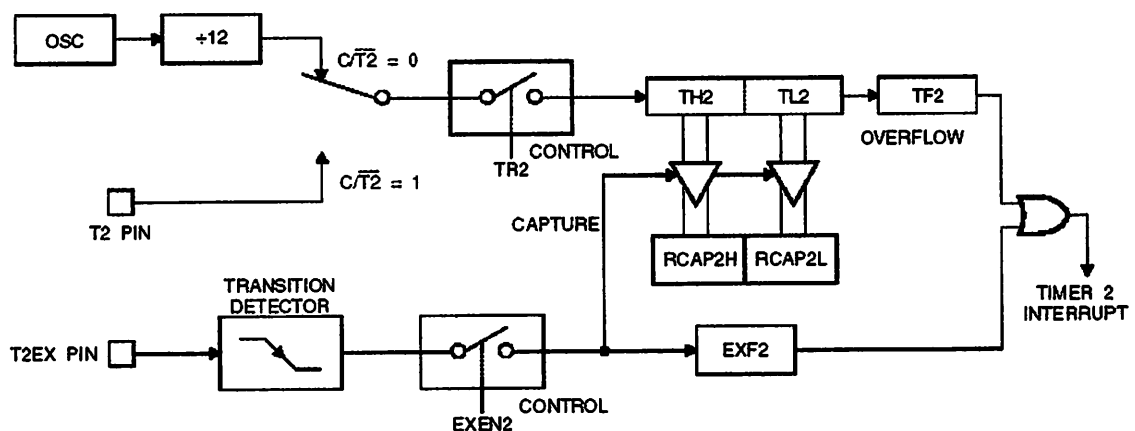
#### - Mode Capture

Pada mode ini dua pilihan dipilih oleh bit EXEN2 pada SFR T2CON. Jika EXEN2 = 0, timer 2 merupakan 16 bit timer atau counter yang jika telah overflow ( melimpah ) akan mengeset bit TF2 pada T2CON. Bit ini juga dapat digunakan untuk menghasilkan sela ( interup ). Jika EXEN2 = 1, timer 2 akan berlaku sama, tetapi suatu transisi tinggi ke rendah ( 1 to 0 ) pada pin T2EX ( P1.1 ) akan menyebabkan nilai sekarang pada TH2 dan TL2 untuk ditangkap dan disimpan ke RCAP2H dan RCAP2L

**Tabel 2.6 T2MOD – Timer 2 Mode Control Register<sup>[1]</sup>**

T2MOD Address = 0C9H						Reset Value = XXXX XX0B		
Not Bit Addressable								
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	T2OE	DCEN

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

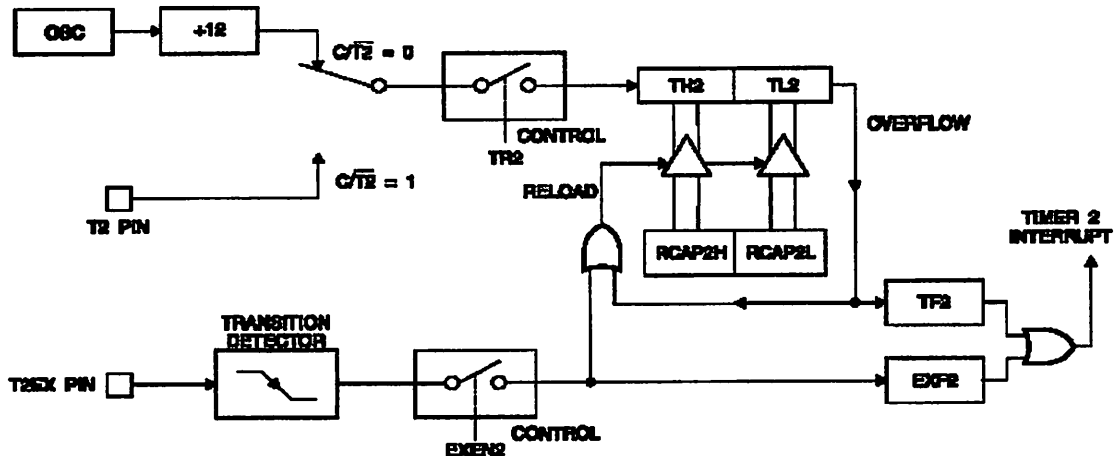


**Gambar 2.7. Timer 2 pada mode CAPTURE<sup>[1]</sup>**

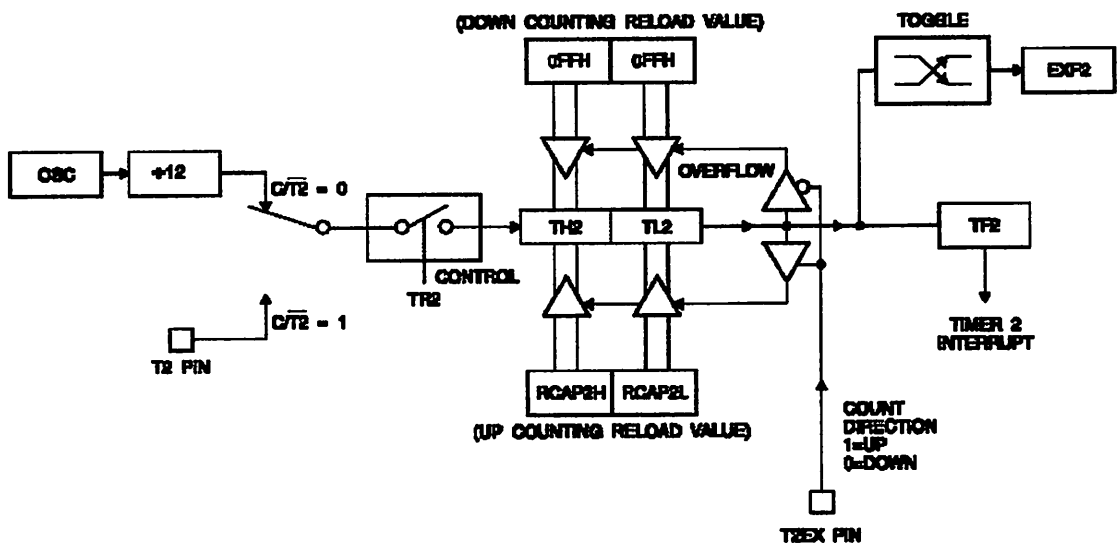
**- Auto Reload ( Up atau Down Counter )**

Timer ini dapat diprogram untuk menghitung maju atau menghitung mundur jika dikonfigurasi sebagai mode 16 bit auto reload. Fitur ini dapat dimatikan dengan mengatur bit DCEN pada SFR T2MOD. Pada saat reset DCEN = 0 timer 2 akan menghitung maju jika DCEN diset maka timer 2 menghitung maju dan mundur tergantung pada nilai logika pada pin T2EX.

Saat  $DCEN = 0$ , pada mode ini, 2 pilihan dipilih oleh bit  $WXWN2$  pada SFR  $T2 CON$ .  $EXEN2 = 0$  timer 2 menjadi  $0FFFFH$  dan kemudian akan mengeset bit  $TF2$  jika overflow yang dapat menyebabkan terjadinya register timer diisi kembali dengan nilai 16 bit dari  $RCAP2H$  dan  $RCAP2L$ .



Gambar 2.8. Pada Mode Auto Reload  $DCEN=0$ <sup>[1]</sup>



Gambar 2.9. Pada Mode Auto Reload  $DCEN=1$ <sup>[1]</sup>

### - **Baud Rate Generator**

Berfungsi sebagai pengirim dan penerima yang disetting pada TCLK pada SFR T2CON. Baud rate generator secara teknis sama dengan auto reload, dimana pelimpahan TH2 meyebabkan register timer2 diisi kembali dengan nilai 16 bit pada register RCAP2H dan RCAP2L yang telah diisi ( preset ) oleh program pemakai.

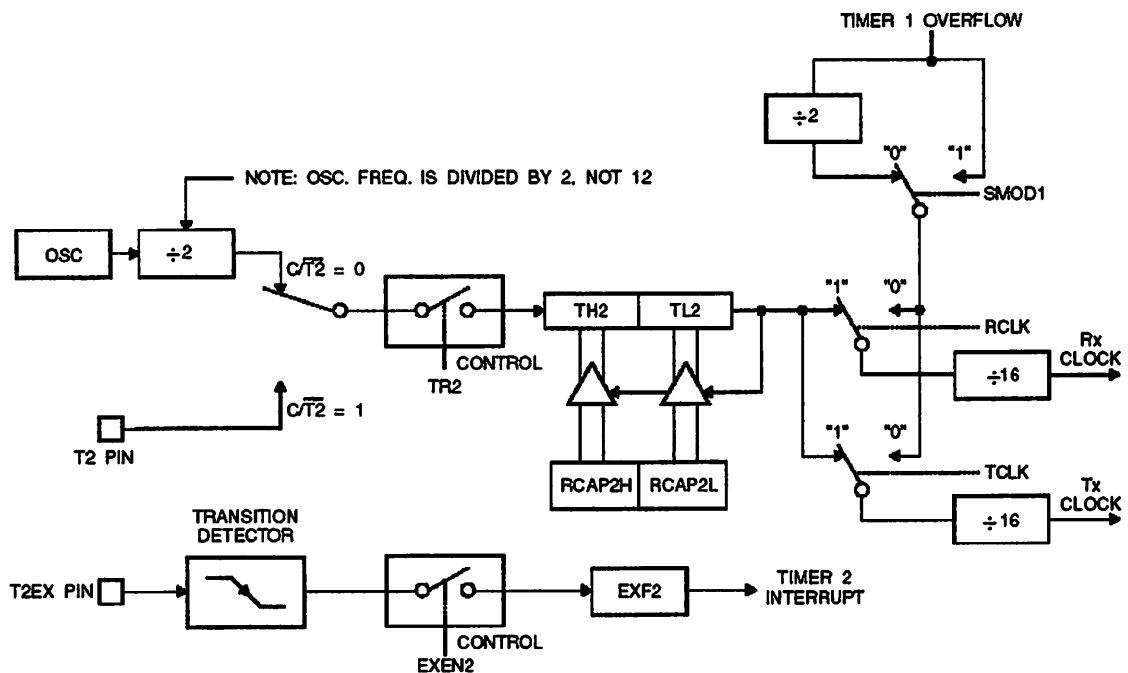
Baut rate pada mode 1 dan 3 ditentukan oleh rate overflow timer berdasarkan persamaan :

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

Timer dapat dikonfigurasi sebagai timer atau counter. Kebanyakan difungsikan sebagai timer dimana bit CP/T2 dibuat "0". Operasi timer adalah berbeda jika timer 2 difungsikan sebagai baud rate generator. Secara normal sebagai timer, akan naik setiap siklus mesin ( pada  $\frac{1}{2}$  frekuensi osilator ). Sebagai baud rate generator akan dinaikkan tiap state time ( pada  $\frac{1}{2}$  frekuensi osilator ). Rumus untuk baud rate sebagai berikut :

$$\frac{\text{Modes 1 and 3}}{\text{Baud Rate}} = \frac{\text{Oscillator Frequency}}{32 \times |65536 - (\text{RCAP2H}, \text{RCAP2L})|}$$

dimana (RCAP2H, RCAP2L ) adalah isi dari register RCAP2H dan RCAP2L yang diambil sebagai 16 bit integer tak bertanda



Gambar 2.10. Sebagai Baud Rate Generator<sup>[1]</sup>

### 2.2.12. Program Status Word ( PSW )

Program Status Word berguna untuk memilih bank memori yang aktif. Berikut adalah bit-bit penyusunan PSW. PSW ini bersifat bit-addressable artinya bit-bit nya masing-masing dapat dirubah tanpa harus merubah satu kesatuan byte

Tabel 2.7. Program Status Word ( PSW)<sup>[1]</sup>

BIT	SIMBOL	ADDRES	BIT DESCRIPTION
PSW. 7	CY	D7 H	Carry Flag
PSW. 6	AC	D6 H	Auxiliaricary Flaf
PSW. 5	F0	D5 H	Flag 0
PSW. 4	RS1	D4 H	Register bank select 1
PSW. 3	RS0	D3 H	Register bank select 0

			00 = bank 0; addresses 00H – 07H 01 = bank 1; addresses 08 H- 0FH 10 = bank 2; addresses 10 H- 17 H 11 = bank 3; addresses 18 H- 1FH
PSW. 2	0V	D2 H	Over Flow Flag
PSW. 1	-	D1 H	Reserved
PSW. 0	P	D0 H	Even Parity flag

PSW dapat dipakai untuk mengatur pemilihan bank memori ( bank0 – bank 3 ). Pemilihan dilakukan dengan mengatur bit RS0 dan bit RS1 menurut tabel berikut :

**Tabel 2.8** Pemilihan Bank Memori<sup>[1]</sup>

RS1	RS0	Bank terpilih	Lokasi memori
0	0	Bank 0	00H – 07H
0	1	Bank 1	08H – 0FH
1	0	Bank 2	10H – 17H
1	1	Bank 3	18H – 1FH

### 2.2.13. Interrupt

Pada AT89S8252 mempunyai 6 interrupt vector : dua interrupt eksternal ( INT0 dan INT1 ), tiga timer interrupt ( timer 0, 1, 2), dan serial port interrupt. Masing-masing dapat bekerja sendiri baik disable atau enable yang dapat diatur



dan dibersihkan pada bit SFR IE. IE ber-alamat A8H dan IP ( Interrupt Priority Control ) alamat B8H. IE digunakan untuk mengontrol interrupt mana saja yang akan diaktifkan, sedangkan IP akan menentukan interrupt mana yang memiliki prioritas tinggi dan prioritas rendah.

**Table 2.9** Interrupt Enable ( IE ) Register<sup>[1]</sup>

(MSB)(LSB)							
EA	—	ET2	ES	ET1	EX1	ET0	EX0
Enable Bit = 1 enables the interrupt.							
Enable Bit = 0 disables the interrupt.							

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
—	IE.6	Reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	SPI and UART interrupt enable bit.
ET1	IE.3	Timer 1 interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.

User software should never write 1s to unimplemented bits, because they may be used in future AT89 products.

### 2.2.14. Organisasi Memory

Organisasi memori pada mikrokontroler AT89S8252 dapat dibagi menjadi dua bagian besar yaitu memori program dan memori data. Pembagian tersebut didasarkan atas fungsi dari penyimpanan data maupun program. Memori program digunakan untuk menyimpan instruksi-instruksi yang akan dijalankan oleh

mikrokontroler, sedangkan memori data digunakan sebagai tempat yang sedang diolah mikrokontroler.

Program mikrokontroler disimpan dalam memori program berupa ROM. Mikrokontroler AT89S8252 dilengkapi dengan Rom internal, sehingga untuk menyimpan program tidak digunakan ROM Eksternal yang terpisah dari mikrokontroler. Agar tidak menggunakan memori program eksternal, penyemat  $\overline{EA}$  dihubungkan dengan Vcc ( logika 1 ).

Memori program mikrokontroler menggunakan alamat 16 Bit mulai dari  $0000_H - 0FFF_H$  sehingga kapasitas penyimpanan program maksimal adalah 8Kbyte. Sinyal  $\overline{PSEN}$  (*Program Store Enable* ) tidak digunakan jika menggunakan memori program internal.

Selain program mikrokontroler AT89S8252 juga memiliki data internal sebesar 256 Byte dan mampu mengakses memori data eksternal sebesar 64 Kbyte. Semua memori data internal dapat dialamati dengan data langsung atau tidak langsung. Ciri dari pengalamatan langsung adalah *operand* adalah alamat register yang berisis alamat data yang akan diolah. Sebagian memori tersebut dapat dialamati dengan pengalamatan register, dan sebagian lagi dapat dialamati dengan memori satu bit. Untuk membaca data digunakan sinyal  $\overline{RD}$  sedangkan untuk menulis digunakan sinyal  $\overline{WR}$ .

#### **2.2.15. Data Memory (EEPROM) dan RAM**

Berbeda dengan mikrokontroler standart MCS-51, mikroontroler ATmel AT89S8252 juga dilengkapi dengan data memori yang berupa EEPROM

(*Electrically Erasable Programmable Read Only Memory*). EEPROM yang ditanamkan ini besarnya 2 Kbyte dan dipakai untuk menyimpan data.

EEPROM ini diakses dengan mengeset bit EEMEN pada register WMCON pada alamat 96 H. Alamat EEPROM ini adalah 000 H – 7FF H. Instruksi MOVX digunakan untuk mengakses EEPROM internal ini. Namun jika ingin mengakses data memori luar (diluar mikrokontroler ATmel AT89S8252) dengan menggunakan instruksi MOVX ini maka bit EEMEN harus dibuat “0”.

Bit EEMWE pada register WMCON harus diset ke 1 sebelum sembarang lokasi pada EEPROM dapat ditulisi. Program pengguna harus mereset bit EEMWE ke “0” jika proses penulisan ke EEPROM tidak diperlukan lagi.

Proses penulisan pada EEPROM dapat dilihat dengan membaca bit RDY/BSY pada SFR WMCON. Jika bit ini berlogika rendah maka penulisan EEPROM sedang berlangsung, jika bit ini berlogika tinggi maka penulisan sudah selesai dan penulisan lain dapat dimulai lagi. Sedangkan RAM yang ada pada mikrokontroler ATMEL AT89S8252 adalah berkapasitas 256 Byte. Penjelasan mengenai RAM ini adalah sama dengan RAM yang ada pada mikrokontroler standart MCS-51.

### **2.3. LCD M16x2**

LCD 1632 yang digunakan adalah LCD produksi Seiko Instrument (M1632), kelebihan dari LCD M1632 ini antara lain adalah mampu dioperasikan dalam mode 4bit, dalam arti pengolahan data dari dan ke LCD dapat dilewatkan

lewat data dengan lebar 4bit saja. Ini cukup membantu menghemat port microcontroller yang dipakai.

Penjelasan pin-out pada LCD M1632 adalah sebagai berikut :

- VSS (pin 1) dan VCC (pin 2) adalah pin untuk power supply.
- VEE (pin 3), adalah pin untuk drive LCD, yaitu untuk mengatur intensitas tampilan pada LCD.
- RS (pin 4), adalah pin untuk pemilihan mode input data. Apabila RS diberi logic 0, maka data berupa data control dan bila RS diberi logic 1, maka data adalah data untuk ditampilkan di LCD.
- R/W (pin 5), pin ini merupakan pin untuk pemilihan proses pada LCD. Bila pin R/W berlogic 1, maka proses read (baca) data, kebalikannya bila pin R/W berlogic 0, proses write (tuliskan) data.
- E (pin 6), adalah pin enable untuk LCD. LCD akan enable bila pin ini berlogic high, kebalikannya bila pin ini berlogic 0, LCD akan disable..
- DB0-DB7 (pin 7 – pin 14), adalah pin untuk input/output data. Tetapi untuk mode pengoperasian 4 bit yang berfungsi hanya DB4 – DB7.
- V+ BL (pin15) dan V-BL (pin 16) adalah pin untuk supply lampu back light.

LCD (Liquid Crystal Display) adalah komponen display yang tidak memancar (nonemissive), sehingga tidak menghasilkan sumber cahaya seperti CRT (Cathode Ray Tube), dan berdaya sangat rendah (lebih rendah dari LED) yaitu dalam hitungan mikrowatt (LED dalam hitungan miliwatt). LCD menahan atau membiarkan cahaya yang dipantulkan dari sumber cahaya luar dan cahaya yang berasal dari belakang atau samping yang melewatinya. LCD dikontrol oleh

ROM/RAM generator karakter dan RAM data display. Semua fungsi display dikontrol dengan instruksi dan LCD dapat dengan mudah diinterfacekan dengan MPU (Mikroprosesor Unit).

Karakteristik dari LCD dot-matriks adalah sebagai berikut:

- 16X2 karakter dengan 5X7 dot matriks + kursor
- ROM generator karakter dengan 8 tipe karakter (untuk program write)
- 80X8 bit RAM data display
- Dapat diinterfacekan dengan 4 atau 8 bit MPU
- RAM data dan RAM generator karakter dapat dibaca dari MPU
- +5V single power supply
- Power-on reset
- Range temperature operasi 0-60°C
- Beberapa fungsi instruksi:

Display clear, Cursor home, Display ON/OFF, Cursor ON/OFF, Display character blink, Cursor Shift dan Display shift.

LCD disini dapat menampilkan karakter yang ada pada ROM generator karakter, yang sudah berisi 192 jenis karakter, dengan cara memberikan kode karakter untuk tiap-tiap karakter yang diinginkan pada bus data dengan menggunakan sinyal kontrol. Fungsi masing-masing pin dari LCD Dot Matriks ini dapat dilihat pada tabel 2.13 berikut ini:

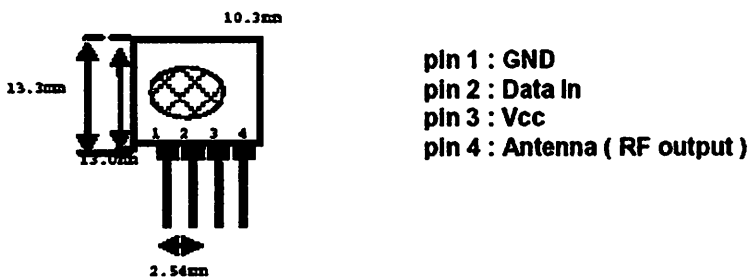
**Tabel 2.10** Pin-Pin LCD dan Fungsinya<sup>[6]</sup>

Nama sinyal	Jumlah terminal	I/O	Tujuan	Fungsi
DB0-DB7	8	I/O	MPU	4 bit data bus lower tristate dua arah, dapat dibaca atau ditulisi terhadap MPU melalui data tersebut. DB7 juga sebagai busy flag.
E	1	INPUT	MPU	Sinyal penanda operasi read/write
R/W	1	INPUT	MPU	0: Write 1: Read
RS	1	INPUT	MPU	Sinyal seleksi register 0: Register instruksi (write) Busy flag dan address Counter (read) 1: Data register (write dan read)
VLC	1	-	Power Supply	Power supply untuk mendrive LCD guna pengaturan contrast
VDD	1	-	Power Supply	+5V
VSS	1	-	Power Supply	Ground
V+BL	1	-	Power Supply	4-4,2 V 50-200mA
V-BL	1	-	Power Supply	0V (GND)

## 2.4. TLP434A dan RLP434A

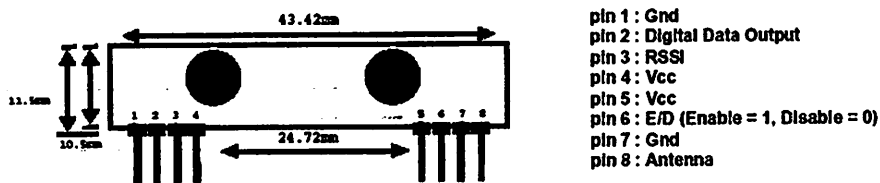
TLP 434 (pemancar) dan RLP 434 (penerima) merupakan suatu pasangan dalam suatu modul rangkaian yang mempunyai fungsi berbeda, yaitu TLP berfungsi mengirimkan data dan RLP berfungsi menerima sinyal yang dipancarkan dari TLP. Digunakannya TLP434A dan RLP 434A sebagai modul RF (*Radio Frekuensi*) pada skripsi ini karena kemampuannya di dalam pengiriman dan penerimaan data yang cukup baik. Biasanya kedua modul ini dihubungkan dengan mikrokontroler atau peralatan digital yang lainnya. Input data adalah serial dengan level TTL (*Transistor – Transistor Logic*). Jarak pancar maksimum dari modul RF ini adalah 100 meter tanpa halangan dan 30 meter di dalam gedung. Ukuran ini dapat dipengaruhi oleh faktor antena, kebisingan, dan tegangan kerja dari pemancar. Panjang antena yang digunakan adalah 17 cm, dan terbuat dari kawat besi. Berikut ini merupakan bentuk dari TLP 434 dan RLP 434:

### 1. Ultra Small Transmitter (TLP434A)



Gambar 2.11. Ultra Small Transmitter (TLP434A)<sup>[3]</sup>

## 2. RLP 434A SAW Based Receiver



Gambar 2.12. RLP 434A SAW Based Receiver<sup>[3]</sup>

## 2.5. Transistor sebagai *switching*

Dalam perancangan alat ini digunakan transistor C945 yang berfungsi sebagai saklar. Transistor ini merupakan transistor bipolar dengan tipe NPN.

Transistor yang digunakan sebagai saklar harus dikondisikan dalam dua kondisi, yaitu :

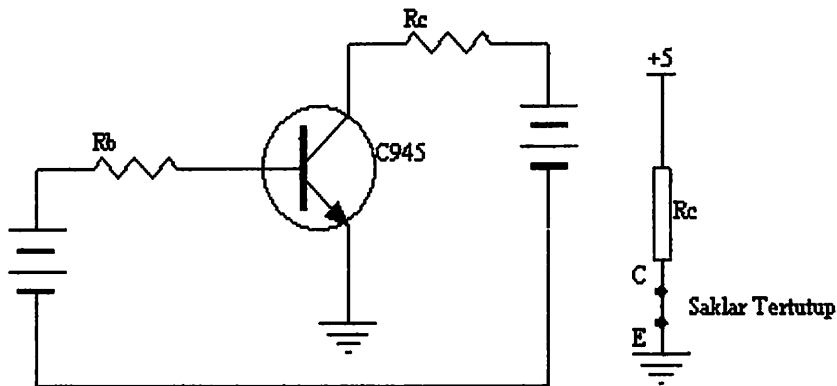
1. Transistor dalam keadaan jenuh (saturasi).

Transistor dalam keadaan jenuh (saturasi), maka berlaku :

- Kuat arus ( $I_c$ ) mencapai maksimum.
- Nilai  $V_{ce}$  sama dengan 0 volt.
- Tegangan pada beban sama dengan tegangan sumber ( $V_{cc}=V_{Rc}$ ).

Untuk lebih jelasnya dapat dilihat pada gambar dibawah ini :





Gambar 2.13. Transistor dalam keadaan Saturasi<sup>[8]</sup>

Untuk menghitung resistansi pada basis menggunakan rumus :

$$V_{cc} - I_c.R_c - V_{ce} = 0$$

Karena keadaan saturasi  $V_{ce} = 0$  maka rumusnya menjadi :

$$V_{cc} - I_c.R_c = 0$$

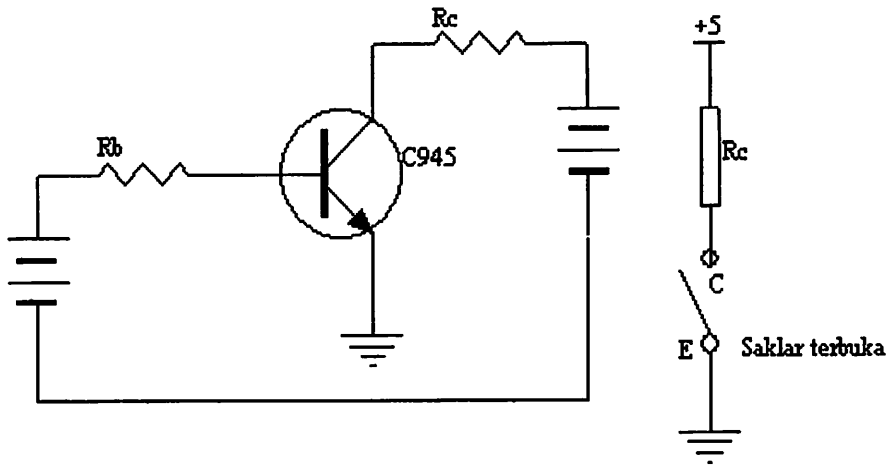
$$I_c = \beta_{dc}.I_b$$

$$I_b = \frac{I_c = V_{cc} - V_{be}}{R_b}$$

## 2. Transistor dalam keadaan *Cut Off*.

Transistor dalam keadaan cut off (sumbat) berlaku hal – hal sebagai berikut :

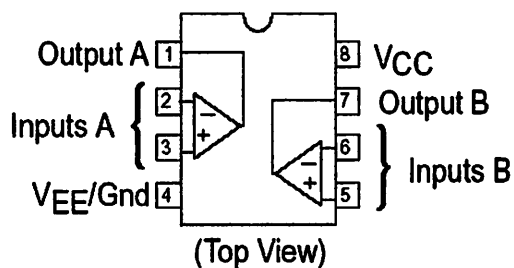
- Nilai arus  $I_b$  sama dengan 0 volt.
- Nilai arus  $I_c$  sangat kecil sekali sehingga dapat dabaikan.
- Besarnya  $V_{cc}$  sama dengan  $V_{ce}$ .



Gambar 2.14. Transistor dalam keadaan Cut Off<sup>[8]</sup>

## 2.6. IC LM 358

IC LM 358 ini digunakan untuk penguatan tegangan yang masuk melalui *receiver* RLP 434 karena tegangan yang diterima atau tegangan yang masuk melalui penerima sangat kecil. Namun bentuk gelombang yang dikeluarkan masih kurang bagus dan masih memerlukan penguatan lebih lanjut yang dalam perancangan alat ini menggunakan transistor C945, sehingga gelombang yang dikeluarkan sesuai dengan yang diharapkan. Berikut ini gambar dari IC LM 358 :



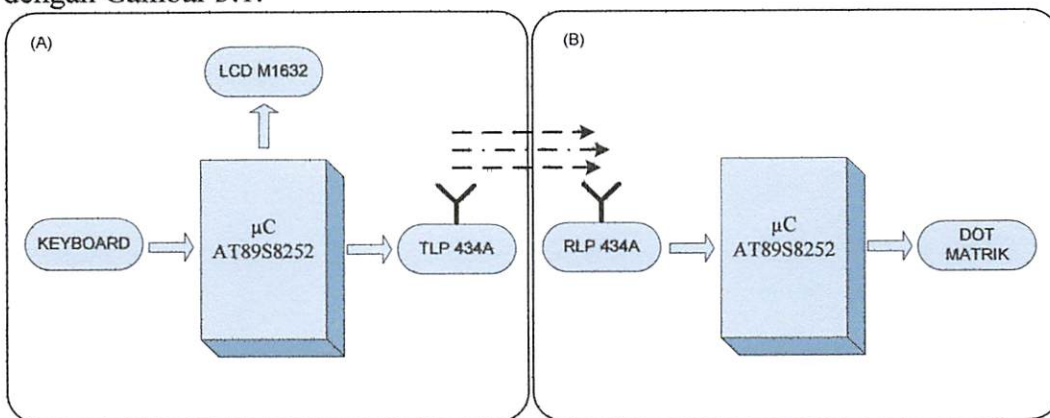
Gambar 2.15. Low Power Dual Operational Amplifier<sup>[7]</sup>

### BAB III PERENCANAAN ALAT

#### 3.1 Pendahuluan

Alat yang dirancang dalam skripsi ini adalah alat penampil informasi huruf dan angka melalui layar dot matrik yang mampu dikendalikan tanpa merubah posisi layar. Sumber informasi berasal dari keyboard PC yang terpasang di unit pengendali, terpisah dari layar dot matrik. Penghubung antara unit pengendali dengan layar dot matrik menggunakan sistem *wireless* (tanpa kabel) pada frekuensi 433,92MHz. Jenis informasi yang akan ditampilkan di layar dot matrik diatur terlebih dahulu di unit pengendali yang dilengkapi dengan LCD M16x2 sebagai alat bantu penampil 16 karakter. Keunggulan lain dari alat ini adalah mampu menyimpan informasi yang telah dibuat secara permanen, walaupun tegangan sumber diputus. Hal ini dikarenakan, informasi tersebut disimpan di memori EEPROM mikrokontroler AT89S8252.

Adapun diagram blok untuk memperjelas sistem alat yang akan dibuat sesuai dengan Gambar 3.1.



Gambar 3.1. Diagram Blok Sistem : (A) Sistem remote  
(B) Sistem layar dot matrik

Perencanaan alat yang dibahas dalam skripsi ini meliputi 2 sistem, yaitu sistem remote dan layar dot matrik 16 karakter. Kedua sistem tersebut terhubung dengan menggunakan sistem *wireless*.

Sistem remote berfungsi sebagai alat yang menghubungkan antara pengguna dengan layar dot matrik. Dengan alat ini, pengguna bisa mengatur jenis tulisan dan animasi yang akan ditampilkan di layar dot matrik. Adapun komponen utama yang terdapat dalam sistem ini adalah sebagai berikut :

- a) Keyboard PC, komponen ini berfungsi sebagai masukan Mikrokontroler AT89S8252 yang berupa jenis tulisan yang akan ditampilkan pada layar dot matrik. Hubungan antara komponen ini dengan mikrokontroler AT89S8252 menggunakan komunikasi serial. Adapun pin I/O AT89S8252 yang digunakan untuk hubungan ini terdiri dari 2 pin. 1 pin berfungsi sebagai pin data dan pin yang lainnya sebagai pembangkit *clock* keyboard PC.
- b) Mikrokontroler AT89S8252, IC ini berfungsi sebagai pengatur sinyal keyboard, LCD serta pengirim data jenis tulisan dan animasi yang akan ditampilkan di layar dot matrik. Selain fungsi tersebut, IC ini juga digunakan sebagai penyimpan jenis tulisan dan animasi yang telah dibuat oleh pengguna, memori penyimpan menggunakan *on-chip* EEPROM.
- c) LCD, komponen ini berfungsi sebagai penampil jenis tulisan sesuai dengan inputan keyboard. Selain itu, komponen ini juga dapat berfungsi sebagai simulasi penampil jenis tulisan dan animasi yang akan ditampilkan di layar dot matrik. Adapun jenis LCD yang digunakan adalah LCD M1632, LCD jenis ini terdiri dari 16 kolom dan 2 baris sehingga karakter maksimum yang

Handwritten notes at the top right of the page, possibly a date or reference.

Main body of handwritten text, appearing to be a letter or a report, with several lines of cursive script.

mampu ditampilkan per-baris adalah 16. Pada proses animasi, hanya baris I yang diaktifkan.

d) TLP434A, modul ini dapat memancarkan data serial dengan baudrate maksimum adalah 200 Kbps. Biasanya modul ini digunakan untuk aplikasi-aplikasi yang tidak membutuhkan kabel misalnya remote control, adapun spesifikasi teknis dari modul ini adalah:

- DC Power Supply : 2-12 Volt
- Modulation ASK
- Data Rate : 512 bps-200 Kbps
- Input : Serial Asynchronous
- Frequency transmitter : 433.92 MHz
- current Consumption : 10mA

Sistem layar dot matrik berfungsi sebagai penampil jenis tulisan dan animasi yang sesuai dengan keinginan. Jenis tulisan dan animasi yang akan ditampilkan di layar dot matrik tersimpan di dalam memori *on-chip* EEPROM AT89S8252. Dengan adanya memori ini, maka jenis tulisan ataupun animasi yang telah tersimpan tidak akan hilang walaupun sistem dimatikan. Adapun komponen utama dalam sistem ini adalah sebagai berikut :

a) RLP434A, Modul ini berguna untuk menerima data yang dipancarkan dari modul TLP434A. Modul ini mempunyai output digital sehingga mampu dihubungkan langsung ke mikroprosesor atau peralatan digital lainnya melalui *interface* serial. Adapun spesifikasi teknis dari modul ini adalah:

- DC Power Supply : 3.3-6.0 Volt

- Modulation : ASK
- Data Rate : 4.8 KHz
- Output : Digital dan Linier
- Frequency Receiver : 433.92 Mhz
- Current Consumption : 15 mA

- b) Mikrokontroler AT89S8252, IC ini berfungsi sebagai penerima data dari sistem alat pengatur layar dot matrik. Adapun data yang diterima berupa jenis tulisan dan jenis animasi yang akan ditampilkan di layar dot matrik. Data yang diterima akan disimpan di dalam *on-chip* EEPROM AT89S8252. Dengan adanya memori ini, jenis tulisan dan animasi yang ditampilkan di layar dot matrik tidak hilang walaupun sistem penampil layar dot matrik dalam kondisi mati.
- c) Layar dot matrik, layar ini terdiri dari beberapa led yang tersusun berdasarkan baris dan kolom. 1 karakter terdiri dari 8 baris dan 5 kolom led, sedangkan jumlah karakter maksimum yang mampu ditampilkan secara langsung adalah 16 karakter. Informasi tulisan yang lebih dari 16 karakter akan ditampilkan dengan menggunakan animasi karakter berjalan atau berkedip.

### **Cara Kerja :**

Prinsip kerja alat yang dirancang dibagi menjadi 2 bagian, berdasarkan fungsinya. Kedua bagian tersebut adalah sistem *remote* layar dot matrik, dan sistem layar dot matrik. Adapun prinsip kerja dari kedua bagian tersebut adalah sebagai berikut :

#### A) Sistem *remote* layar dot matrik

Sistem ini berfungsi sebagai alat pengatur jenis tampilan dan animasi dari informasi yang akan ditampilkan di layar dot matrik. Pembentukan jenis tampilan dilakukan oleh tombol-tombol keyboard yang terhubung secara serial ke mikrokontroler. Sinyal yang terjadi pada keyboard berupa konfigurasi 8 bit nilai biner yang membentuk kode ASCII. Setiap karakter yang dikirim oleh keyboard akan ditampilkan di LCD M16x2, melalui 4 jalur data. Oleh karena kode ASCII LCD terdiri dari 8 bit, maka pengiriman data dari mikrokontroler ke LCD dilakukan secara *multiplex*. Jenis animasi yang akan ditampilkan di layar dot matrik adalah layar berkedip dan layar berjalan dari kanan ke kiri. Kedua animasi tersebut akan disimulasikan di LCD M16x2 dengan cara memberikan logika “0” pada pin RS LCD, dan mengatur 8 bit data LCD sebagai berikut :

- animasi berkedip : - display off (00001000b)
  - display on (00001100b)
- animasi layar berjalan (00011000b)

Setelah jenis tampilan dan animasi dibuat di sistem ini, informasi tersebut akan dikirim ke sistem layar dot matrik secara *wireless* menggunakan modul TLP434A. Pengiriman data dari mikrokontroler ke modul ini dilakukan secara serial.

#### B) Sistem layar dot matrik

Sistem layar dot matrik berfungsi sebagai penampil informasi yang telah dibentuk di sistem *remote* layar dot matrik. Data yang dikirim dari sistem *remote*,



akan diterima modul RLP434A dan diteruskan ke mikrokontroler melalui komunikasi serial. Setelah pengiriman data selesai, mikrokontroler akan menampilkan data tersebut ke layar dot matrik. Jenis tulisan yang terjadi di layar ini, merupakan hasil konfigurasi mati/hidupnya dioda led yang tersusun secara baris dan kolom.

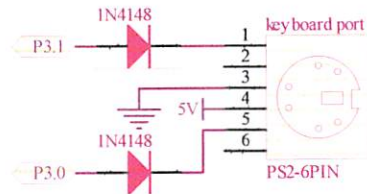
### 3.2 Perencanaan Perangkat Keras

Bagian ini menguraikan perencanaan perangkat keras yang meliputi perencanaan :

1. Sistem *remote*
  - a) *Interface* keyboard PC dengan mikrokontroler AT89S8252.
  - b) Minimum sistem mikrokontroler AT89S8252.
  - c) *Interface* modul LCD dengan mikrokontroler AT89S8252.
  - d) *Interface* modul TLP434A dengan mikrokontroler AT89S8252.
2. Sistem layar dot matrik
  - a) *Interface* modul RLP434A dengan Mikrokontroler AT89S8252.
  - b) *Interface* dot matrik 8x5 dengan Mikrokontroler AT89S8252.

### 3.2.1 Perangkat keras sistem *remote*

#### a) *Interface* keyboard PC dengan mikrokontroler AT89S8252



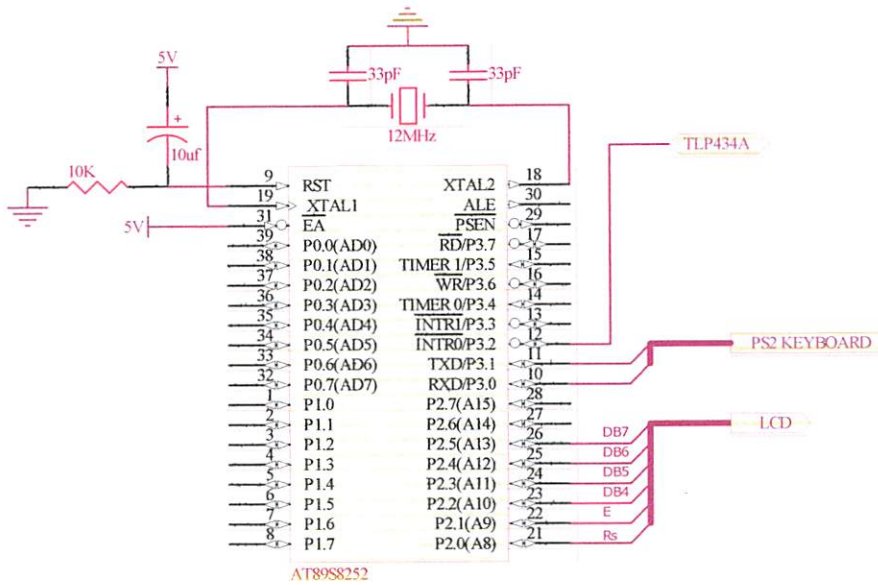
Gambar 3.2. Rangkaian *interface* keyboard PC

Sck pada keyboard adalah jalur pengiriman data yang fungsinya sebagai indikator bahwa data pada jalur Sda siap di kirim. Setiap ada pengiriman data pada Sda, jalur ini akan berlogika “0”.

Sda pada keyboard adalah jalur pengiriman data yang fungsinya untuk mengirimkan data yang berupa kode tombol ASCII (8-bit) yang terdapat pada keyboard ke mikrokontroler.

#### b) Minimum sistem mikrokontroler AT89S8252 sebagai *remote*

Suatu minimum sistem Mikrokontroler AT89S8252 harus dilengkapi oleh 2 rangkaian utama, yaitu rangkaian *clock* dan *reset* serta jalur I/O. Jalur I/O harus ditentukan dengan baik, agar pengaturan sinyal-sinyal yang digunakan baik sebagai masukan ataupun keluaran bisa optimal. Pada sistem *remote* layar dot matrik, mikrokontroler digunakan sebagai pengatur sinyal masukan dari keyboard PC, sinyal keluaran ke LCD, dan sinyal keluaran ke rangkaian TLP434A. Rangkaian pembentuk minimum sistem AT89S8252 *remote* layar dot matrik dan jalur I/O yang digunakan sesuai dengan Gambar 3.3.



Gambar 3.3. Rangkaian minimum sistem mikrokontroler AT89S8252

Adapun komponen-komponen pembentuk minimum sistem mikrokontroler AT89S8252 dan jalur-jalur I/O yang digunakan adalah sebagai berikut:

- **Rangkaian *clock***

Kecepatan proses yang dilakukan oleh mikrokontroler ditentukan oleh sumber *clock* (pewaktuan) yang mengendalikan mikrokontroler tersebut. Sistem yang dirancang akan menggunakan osilator internal yang sudah tersedia di dalam chip AT89S8252. Untuk menentukan frekuensi osilatornya, cukup dengan cara menghunungkan kristal pada pin XTAL1 dan XTAL2 serta dua buah kapasitor ke *ground*. Dengan menggunakan *crystal* maka dapat dihitung waktu yang diperlukan untuk satu siklus mesin.

$$f = 12 \text{ Mhz}$$

sehingga  $T = \frac{1}{f}$

$$T = \frac{1}{12.000.000\text{Hz}} = 0.08333 \mu\text{s}$$

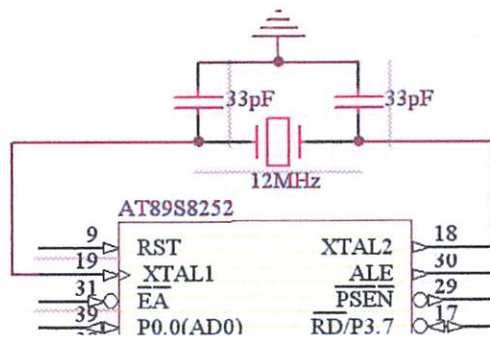
Maka untuk satu siklus mesin dari mikrokontroler besarnya adalah :

$$T_{me} = 12 \times T$$

$$T_{me} = 12 \times 0.08333 \mu\text{s}$$

$$T_{me} = 1 \mu\text{s}$$

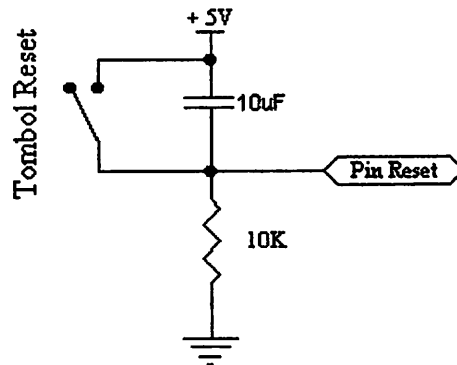
Rangkaian *clock* (pewaktu) yang digunakan sesuai dengan Gambar 3.4.



Gambar 3.4. Rangkaian *clock* minimum sistem mikrokontroler AT89S8252

- **Rangkaian Reset**

*Reset* pada mikrokontroler merupakan masukkan aktif High '1', pulsa transisi dari rendah '0' ke tinggi '1' akan mereset mikrokontroler menuju alamat 0000H. Pin *reset* dihubungkan dengan rangkaian *power on reset* seperti pada Gambar 3.5.



Gambar 3.5. Rangkaian *power on reset* mikrokontroler AT89S8252

Rangkaian reset bertujuan agar mikrokontroler dapat menjalankan proses dari awal. Rangkaian *reset* untuk mikrokontroler dirancang agar mempunyai kemampuan *power on reset*, yaitu *reset* yang terjadi pada saat sistem dinyalakan untuk pertama kalinya. *Reset* juga dapat dilakukan secara manual dengan menekan tombol reset yang berupa *switch push button*.

Rangkaian *Reset* terbentuk oleh komponen R dan C. Nilai R yang dipakai adalah  $10\text{ k}\Omega$  dan C  $10\text{ }\mu\text{F}$ .

Sedangkan untuk mencari frekuensi dari reset tersebut dengan menggunakan rumus sebagai berikut :

$$f_o = \frac{1}{1,1RC}$$

Sehingga dengan komponen resistor  $10\text{ Kohm}$  dan kapasitor  $10\text{ uF}$  akan dihasilkan frekuensi ( $f_o$ ) sebagai berikut :

$$\begin{aligned} f_o &= \frac{1}{1,1RC} \\ &= \frac{1}{1,1 \times 10 \cdot 10^3 \times 10 \cdot 10^{-6}} = 9.09\text{ Hz} \end{aligned}$$

Maka Periode Clock  $T = \frac{1}{f_0}$

$$T = \frac{1}{9.09\text{Hz}} = 0,11 \text{ detik} = 110\text{ms}$$

Waktu minimal yang dibutuhkan untuk *power on reset* adalah 2 kali siklus mesin ditambah *power on osilator*, sehingga :

$$\begin{aligned} \text{waktu minimal } \textit{power on reset} &= (1 \mu\text{s} \times 2) + 10 \text{ ms} \\ &= 0,001002 \text{ s} \end{aligned}$$

Berdasarkan hasil perhitungan, maka komponen resistor 10 Kohm dan kapasitor 10 uF dapat dijadikan komponen pembentuk rangkaian *power on reset*.

- **Pembagian Port**

Mikrokontroler AT89S8252 mempunyai 4 buah port yaitu *port 0* ( P0 ), *port 1* ( P1 ), *port 2* ( P2 ) dan *port 3* ( P3 ). Adapun pembagian *port-port* ini terhadap *device* di luar sistem AT89S8252 adalah sebagai berikut :

- LCD LM1632 :
  - RS = P2.0
  - E = P2.1
  - DB4 = P2.2
  - DB5 = P2.3
  - DB6 = P2.4
  - DB7 = P2.5
- TLP434 = P3.2

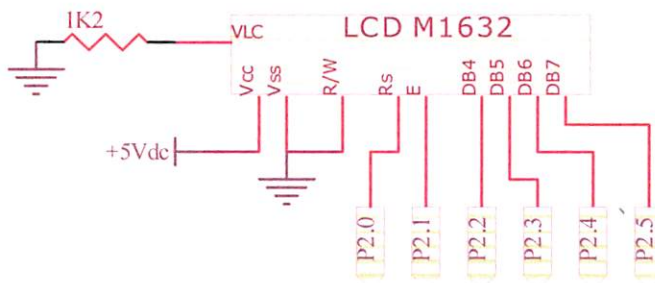
- o Keyboard :
  - sck = P3.0
  - sda = P3.1

c) *Interface* modul LCD dengan mikrokontroler AT89S8252

LCD yang digunakan dalam sistem *remote* adalah LCD LM1632, LCD ini terdiri dari dua baris layar dengan tiap barisnya memiliki 16 karakter.

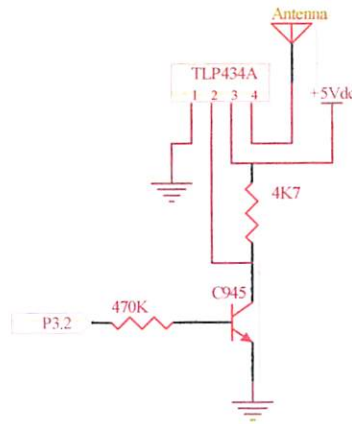
LCD ini membutuhkan 3 sinyal kontrol, R/W (*read/write*) untuk menentukan apakah data akan dibaca atau ditulis, E (*Enable*) yang merupakan sinyal untuk mengaktifkan dan RS (*Register Select*) untuk memilih register yang diakses. LCD LM1632 memiliki 2 register yaitu register data dan register instruksi.

Dalam sistem ini, pin R/W dihubungkan ke *ground* atau selalu berlogika 0 karena dalam perancangan LCD selalu dalam operasi tulis dan pin RS dihubungkan ke port P2.0 sistem mikrokontroler. Pengaktifan LCD ini selanjutnya tergantung pada pin E yang terhubung ke mikrokontroler AT89S8252 melalui P2.1. Adapun *interface* modul LCD dengan sistem mikrokontroler AT89S8252 pada sistem *remote* sesuai dengan Gambar 3.6.



Gambar 3.6. Rangkaian LCD dengan mikrokontroler AT89S8252

d) *Interface* modul TLP434A dengan mikrokontroler AT89S8252



Gambar 3.7. Rangkaian *interface* TLP434A dengan AT89S8252

Keluaran *default* P3.2 mikrokontroler AT89A8252 adalah high (5 Volt), sehingga diperlukannya transistor yang berfungsi sebagai saklar agar TLP434A tidak mengirimkan logika *high* secara terus menerus.

Tegangan  $V_{cc} = 5V$  dan  $I_c$  yang diinginkan untuk dapat mengaktifkan transistor sebesar  $1\text{ mA}$  ( $V_{be} = 0,65\text{ V}$ ), maka nilai resistor  $R_c$  adalah :

$$R_c = \frac{V_{cc}}{I_c}$$

$$R_c = \frac{5V}{1\text{mA}}$$

$$R_c = 5K\Omega$$

Diinginkan  $H_{fe}$  dari transistor adalah 100 dan karena keluaran P3.2 AT89SA8252 adalah 5V, maka untuk dapat mengaktifkan transistor diperlukan resistor ( $R_b$ ) :

$$I_b = \frac{I_c}{\beta_{dc}}$$



$$I_b = \frac{1mA}{100}$$

$$I_b = 10\mu A$$

Dari nilai  $I_b$  yang sudah diketahui maka nilai  $R_b$  adalah :

$$R_b = \frac{V_{bb} - V_{be}}{I_b}$$

$$R_b = \frac{5 - 0,65}{10\mu A}$$

$$R_b = 435 \text{ K}\Omega$$

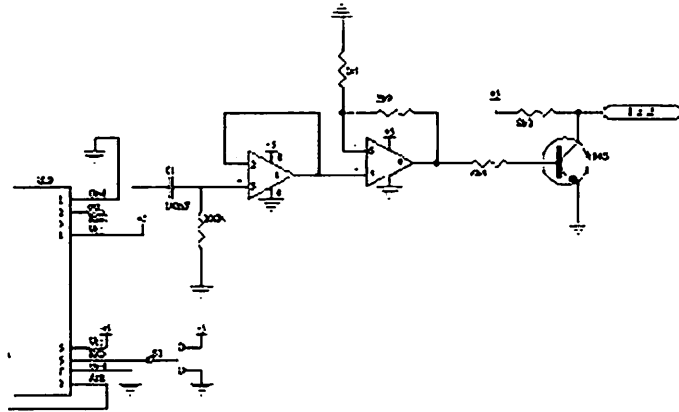
Oleh karena resistor dengan nilai  $5\text{K}\Omega$  dan  $453\text{K}\Omega$  tidak terdapat di pasaran maka digunakan resistor pengganti yaitu senilai  $4\text{K}7\Omega$  dan  $470\text{K}\Omega$ . Dan dengan karakteristik transistor yang diinginkan seperti diatas maka digunakanlah transistor C945.

### 3.2.2 Perangkat keras sistem layar dot matrik

#### a) *Interface* RLP434A dengan mikrokontroler AT89S8252

Sinyal data dari *remote* mikrokontroler AT89S8252 yang dikirimkan oleh rangkaian pengirim (TLP434A) akan diterima oleh modul penerima RLP434A, kedua modul ini bekerja pada frekwensi 434 MHZ.

Keluaran dari rangkaian RLP sudah berupa sinyal digital dan selanjutnya sinyal tersebut dikuatkan dengan rangkaian LM358 sehingga dapat mengaktifkan transistor SC945 untuk disesuaikan dengan tegangan mikrokontroler. Adapun gambar rangkaian *interface* RLP434 dengan mikrokontroler AT89S8252 sebagai pengatur layar dot matrik sesuai dengan Gambar 3.8.



Gambar 3.8. Rangkaian *interface* RLP434A dengan AT89S8252

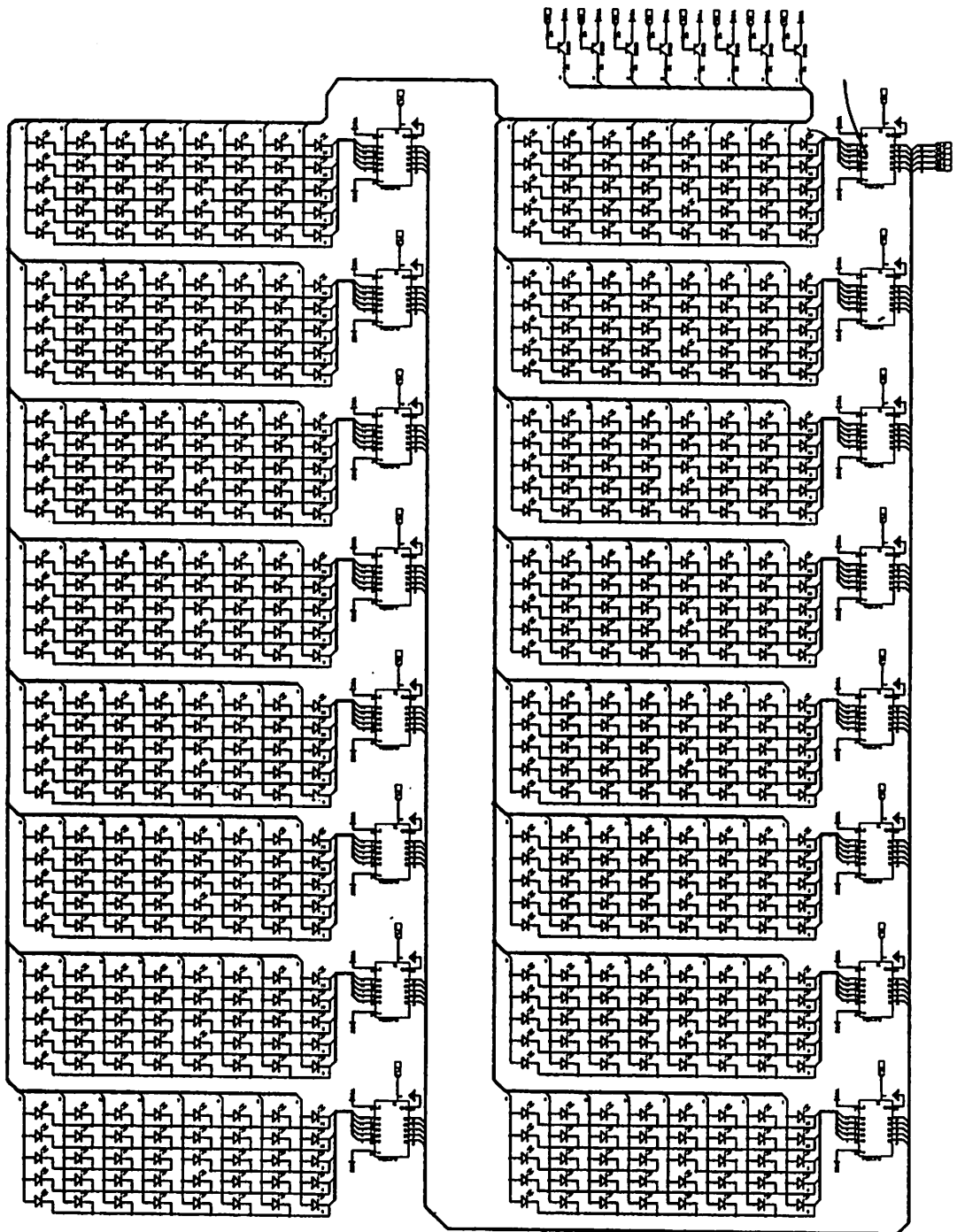
Pada saat RLP menerima data High =0.5V pada pengukuran menggunakan oscilloscope dapat dilihat bahwa gelombang keluaran dari RLP belum berada di level 0 volt hal ini disebabkan oleh adanya tegangan DC. Kapasitor 100nF dan resistor 100Kohm berfungsi untuk menghilangkan sinyal dc dan menurunkan level tersebut sehingga berada pada level 0 volt. RLP dihubungkan dengan rangkaian Buffer (IC LM 358) sebagai penyangga dan kemudian tegangan dikuatkan pada penguat non inverting sehingga tegangan tersebut dapat mengaktifkan transistor C945, maka dilakukan perhitungan :

**Perhitungan rangkaian buffer :**

$$V_{in} = V_{out}(\text{buffer}) = 0,5V$$

Rangkaian ini (buffer) berfungsi untuk menstabilkan frekuensi dan atau amplitud akibat dari pembebanan tingkat selanjutnya.

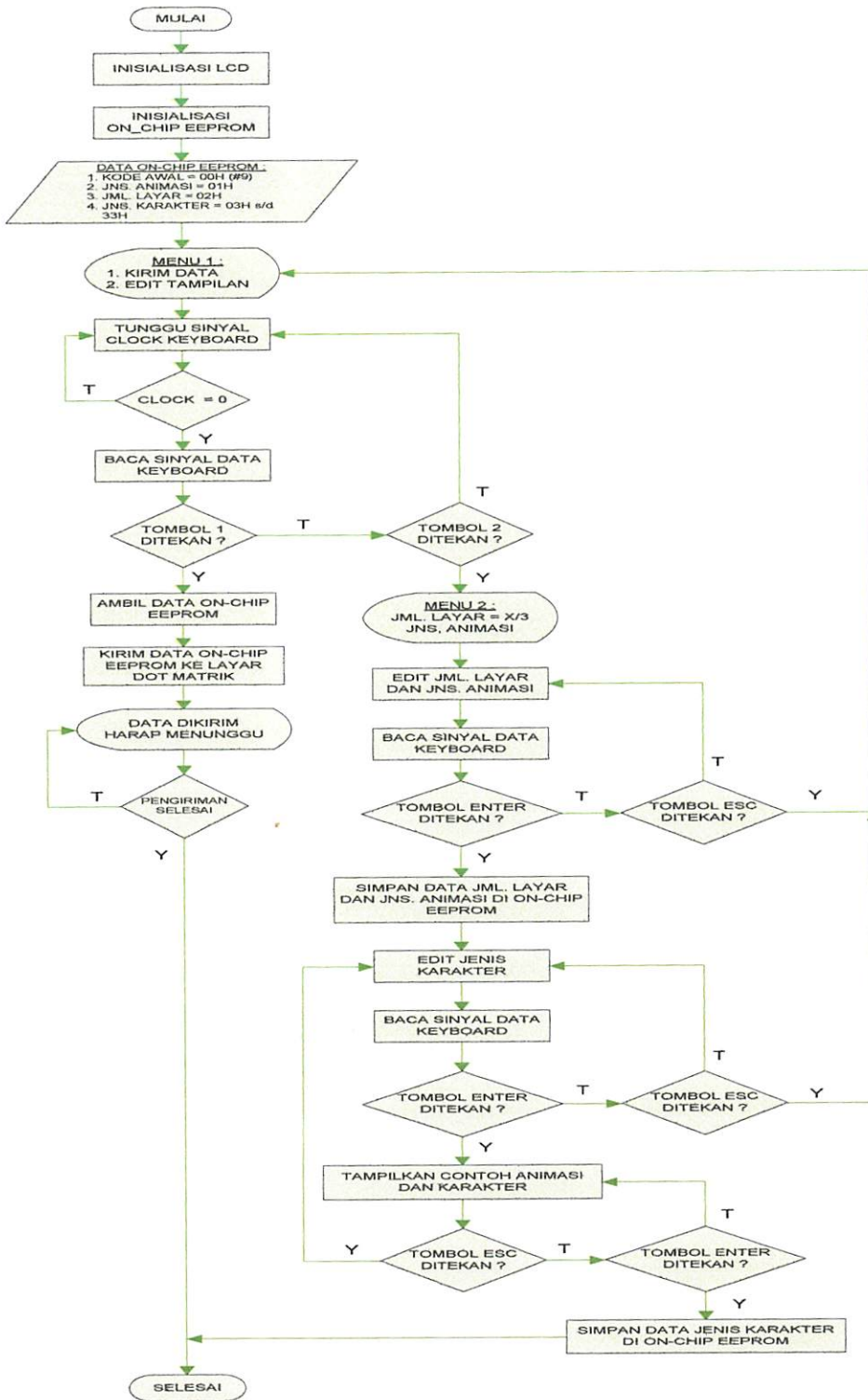
Berdasarkan perhitungan tersebut, maka untuk mendapatkan arus yang dibutuhkan led (0,015A) maka nilai  $R_B$  yang digunakan lebih kecil daripada  $19.130\Omega$ . Adapun nilai  $R_B$  yang digunakan yaitu  $4.700\Omega$ .



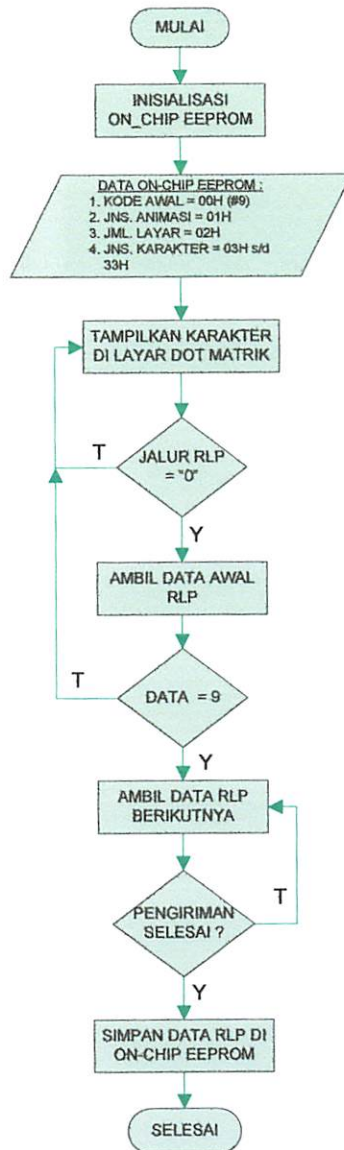
Gambar 3.9. Rangkaian *interface* dot matrik

### 3.3 Perencanaan Perangkat Lunak

Perencanaan alat yang akan dibuat terdiri dari 2 sistem yang saling terpisah, yaitu sistem *remote* dan layar dot matrik. Kedua sistem tersebut, masing-masing menggunakan mikrokontroler AT89S8252 sebagai pengatur peralatan lainnya. Jenis program yang digunakan untuk mengatur jenis perintah mikrokontroler tersebut adalah program bahasa *assembly*. Sistematika jalannya program yang dibuat didasarkan pada sistem *hardware* yang telah dirancang. Adapun urutan kerja program pada kedua alat yang dibuat sesuai dengan Gambar 3.10. dan Gambar 3.11.



Gambar 3.10. Diagram Alir *software* sistem remote Layar Dot Matrik



Gambar 3.11. Diagram Alir *software* sistem layar dot matrik

Salah satu keunggulan mikrokontroler AT89S8252 adalah memiliki *on-ship* memori EEPROM. Memori ini digunakan pada setiap sistem dengan tujuan sebagai berikut :

- a) sistem *remote*, memori EEPROM digunakan untuk menyimpan jenis karakter, jenis animasi, dan jumlah layar yang telah dibuat.

- b) sistem layar dot matrik, memori EEPROM digunakan untuk menyimpan jenis karakter, animasi, dan jumlah layar yang akan ditampilkan.

Dengan adanya memori ini, maka data yang sudah disimpan dalam memori ini tidak akan rusak atau hilang walaupun sumber tegangan dihilangkan.

Untuk mempermudah menggunakan memori ini, maka perlu dibentuknya *subroutine* program *assembly* yang terdiri dari 2 jenis, yaitu *subroutine* membaca data dalam memori EEPROM dan *subroutine* menulis/menyimpan data ke dalam memori EEPROM. Adapun *listing* kedua jenis *subroutine* tersebut adalah sebagai berikut :

- a) *subroutine* membaca data memori EEPROM

```

=====
;
WMCON EQU 96H
;
=====
BACA_EEPROM:
MOV WMCON,#00001010B
MOV DPH,#00H
MOV DPL,ALAMAT
MOVX A,@DPTR
MOV DATA_MASUK,A
RET
=====

```

WMCON adalah salah satu register memori SFR mikrokontroler AT89S8252 yang berfungsi untuk mengaktifkan memori EEPROM.

- b) *subroutine* menulis data memori EEPROM

```

TULIS_EEPROM:
MOV WMCON,#00011010B
MOV DPH,#00H
MOV DPL,ALAMAT
MOV A,DATA_KELUAR
MOVX @DPTR,A
TUNGGU_TULIS_EEPROM:
MOV A,WMCON
CJNE A,#00011010B,TUNGGU_TULIS_EEPROM
MOV WMCON,#00001010B
RET

```



## BAB IV

### PENGUJIAN ALAT

#### 4.1. Pendahuluan

Tujuan pengujian adalah untuk mengetahui kinerja alat dari tiap blok rangkaian yang direncanakan, sehingga dapat diketahui *performance* alat yang direncanakan. Adapun pengujian yang dilakukan meliputi beberapa rangkaian sebagai berikut :

- a) Pengujian rangkaian *interface* keyboard PC dan LCD M16x2
- b) Pengujian rangkaian TLP434A dan RLP434A
- c) Pengujian *driver* dot matrik
- d) Pengujian rangkaian layar dot matrik
- e) Pengujian sinyal RLP434A
- f) Pengujian frekuensi pembawa TLP434A
- g) Pengujian jangkauan *wireless*
- h) Pengujian alat keseluruhan

#### 4.2. Pengujian Rangkaian *interface* keyboard PC dan LCD M16x2

##### 4.2.1. Tujuan pengujian rangkaian

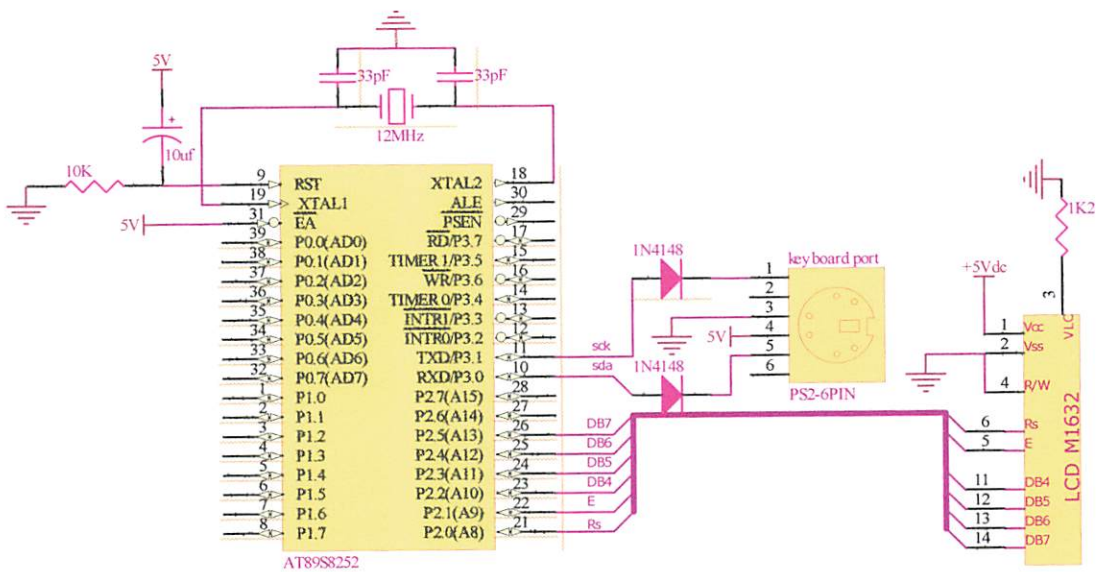
1. Untuk mengetahui apakah rangkaian *interface* keyboard PC dengan mikrokontroler AT89S8252 bisa berfungsi dengan baik.
2. Untuk mengetahui kode setiap tombol keyboard.
3. Untuk mengetahui apakah rangkaian *interface* LCD dengan mikrokontroler AT89S8252 bisa berfungsi dengan baik.

#### 4.2.2. Peralatan yang digunakan

- Rangkaian *interface* dan keyboard PC PS/2
- Rangkaian minimum sistem mikrokontroler AT89S8252
- Rangkaian LCD M16x2
- Power Supply 5 volt

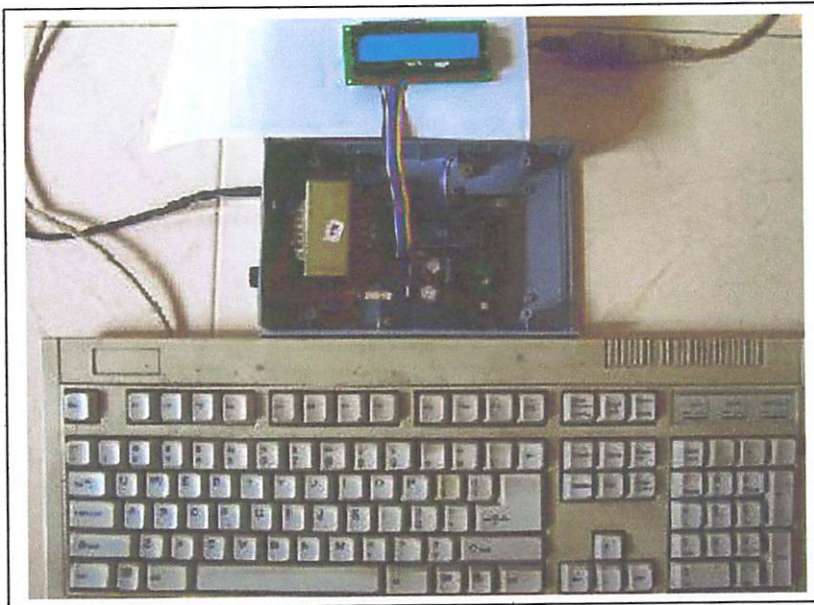
#### 4.2.3. Prosedur pengujian

- Merangkai peralatan yang sesuai dengan Gambar 4.1.

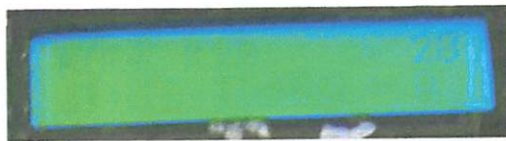


Gambar 4.1. Rangkaian pengujian *interface* keyboard PC dan LCD M16x2

Adapun foto rangkaian peralatan yang telah dibuat sesuai dengan Gambar 4.2.



(A)



(B)

Gambar 4.2.

- (A) Foto alat pengujian *interface* keyboard PC dan LCD ;  
(B) Kode dan jenis tombol

2. Menghidupkan catu daya.
3. Menekan salah satu tombol keyboard.
4. Mengamati dan mencatat kode tombol dan jenis tombol yang ditekan.

Mengulangi langkah 3 sampai 5 sebanyak 44 kali ulangan pada tombol yang berbeda.

#### 4.2.4. Hasil pengujian dan analisa

Hasil pengujian berupa kode data setiap tombol keyboard PC dan jenis tombol yang ditekan. Data tersebut ditampilkan sesuai dengan Tabel 4.1.

Tabel 4.1  
Hasil pengujian *interface* keyboard PC dan LCD M16x2

NO.	JENIS TOMBOL	KODE TOMBOL (HEXA)	NO.	JENIS TOMBOL	KODE TOMBOL (HEXA)
1	Esc	76	23	E	24
2	F1	5	24	F	2B
3	F2	6	25	G	34
4	F3	4	26	H	33
5	`	0E	27	I	43
6	1	16	28	J	3B
7	2	1E	29	K	42
8	3	26	30	L	4B
9	4	25	31	M	3A
10	5	2E	32	N	31
11	6	36	33	O	44
12	7	3D	34	P	4D
13	8	3E	35	Q	15
14	9	46	36	R	2D
15	0	45	37	S	1B
16	-	4E	38	T	2C
17	=	55	39	U	3C
18	←	66	40	V	2A
19	A	1C	41	W	1D
20	B	32	42	X	22
21	C	21	43	Y	35
22	D	23	44	Z	1A

Dari data hasil pengujian tersebut, dapat dilihat bahwa rangkaian *interface* keyboard PC mampu memberikan sinyal *clock* dan data yang sesuai dan dapat dibaca oleh rangkaian minimum sistem AT89S8252. Rangkaian LCD M16x2 juga dapat berfungsi dengan baik, karena LCD bisa menampilkan data yang sesuai dengan program.

### 4.3. Pengujian Rangkaian TLP434A dan RLP434A

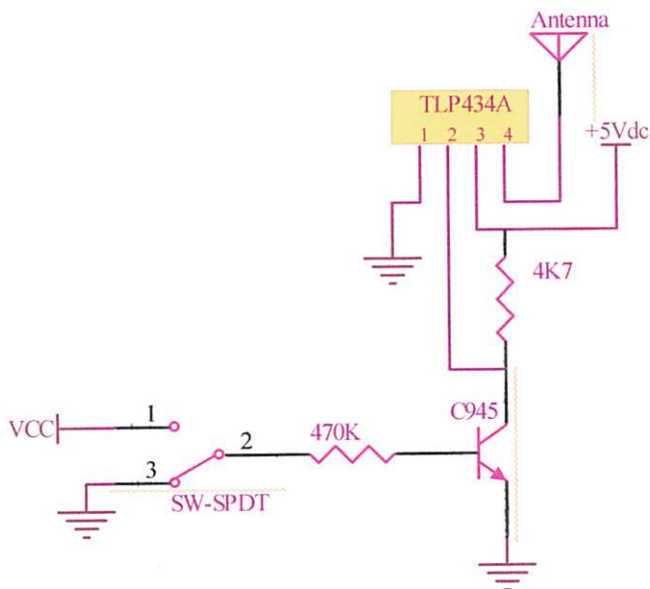
4.3.1. Tujuan pengujian rangkaian adalah untuk mengetahui apakah modul TLP dan RLP434A dapat bekerja dengan baik.

4.3.2. Peralatan yang digunakan

1. Rangkaian TLP dan RLP434A.
2. Rangkaian minimum sistem
3. Power Supply 5V

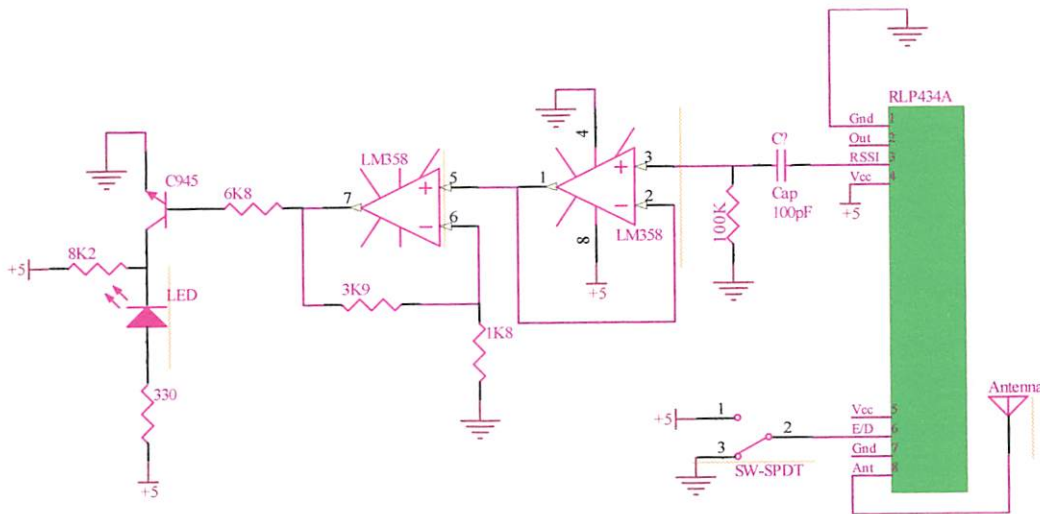
4.3.3. Prosedur pengujian

1. Merangkai peralatan sesuai dengan Gambar 4.3. dan Gambar 4.4.



Gambar 4.3.  
Rangkaian pengujian modul TLP434A





Gambar 4.4.  
Rangkaian pengujian modul RLP434A

2. Menghubungkan jalur input TLP434A ke Gnd dan +5Vdc, secara bergantian.
3. Mencatat kondisi led RLP434A pada kondisi TLP434A diberi sinyal Gnd dan +5vdc.

#### 4.3.4. Hasil dan analisa

Hasil pengujian rangkaian berupa nyala dan matinya lampu led yang terhubung ke jalur output RLP434A. Sinyal Gnd dari TLP434 diasumsikan sebagai logika “0” sedangkan sinyal +5Vdc sebagai logika “1”. Lampu led RLP434A akan hidup apabila modul ini menerima logika “0” dari TLP434A, demikian sebaliknya apabila led modul RLP434A menerima logika “1”. Adapun hasil dari pengujian rangkaian ini ditampilkan pada Tabel 4.2.

Tabel 4.2  
Hasil pengujian modul RLP434A dan TLP434A

NO.	SINYAL TLP434A	LED RLP434A
1	Ground	Mati
2	5V	Menyala

Dari data hasil pengujian dapat dilihat bahwa modul RLP434A mampu menerima sinyal digital dari TLP434A dengan baik.

#### 4.4. Pengujian *driver* dot matrik

4.4.1. Tujuan pengujian adalah untuk mengetahui kinerja transistor 9012 sebagai saklar pengatur logika baris layar dot matrik dan IC 74HC573 sebagai pengatur logika kolom layar dot matrik.

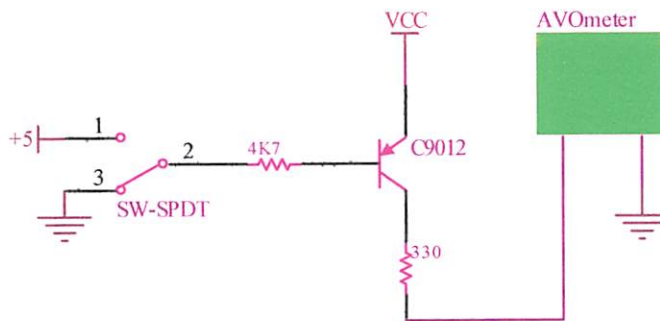
4.4.2. Peralatan yang digunakan

1. Rangkaian transistor 9012 sebagai saklar digital.
2. Rangkaian IC 74HC573.
3. Dot matrik.
4. AVOMeter.
5. *Power supply* 5Vdc.

4.4.3. Prosedur pengujian

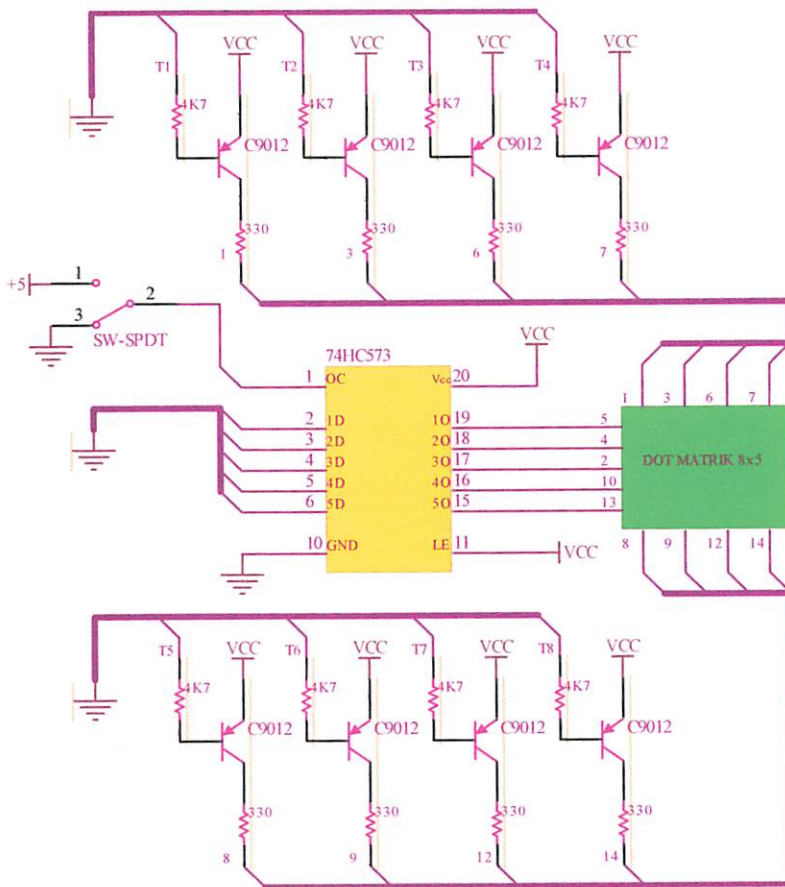
1. Merangkai saklar digital transistor 9012 sesuai dengan Gambar 4.5.
2. Menghidupkan *power supply*.
3. Mengamati dan mencatat nilai tegangan di AVOMeter.

4. Mengaktifkan saklar, kemudian mengamati dan mencatat perubahan nilai tegangan di AVOMeter.
5. Mematikan *power supply*.
6. Merangkai IC 74HC573, transistor dan dot matrik seperti Gambar 4.6.
7. Menghidupkan *power supply*.
8. Mengamati perubahan led pada dot matrik.
9. Mengaktifkan saklar, kemudian mengamati perubahan led pada dot matrik.



Gambar 4.5.  
Rangkaian pengujian saklar baris dot matrik

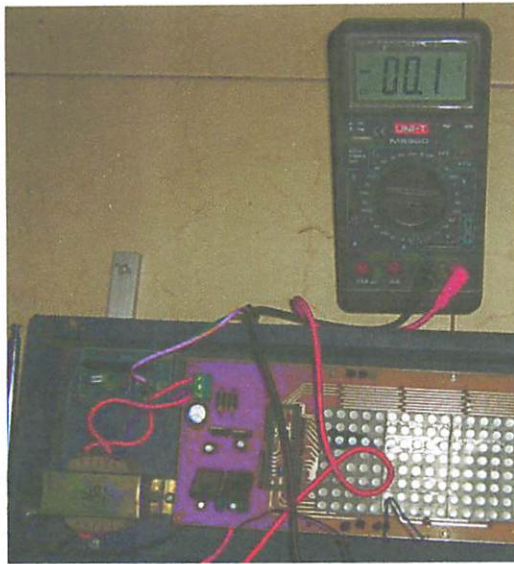




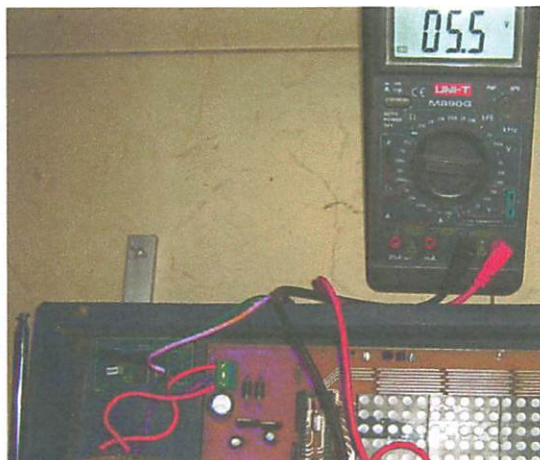
Gambar 4.6.  
Rangkaian pengujian dot matrik

#### 4.4.4. Hasil dan analisa

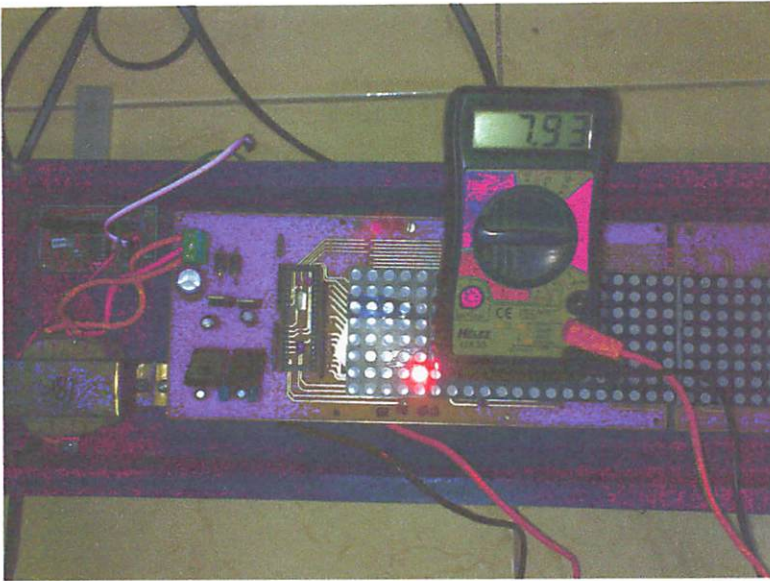
1. Hasil pengujian saklar baris dot matrik ditunjukkan pada Gambar 4.7 dan 4.8., sedangkan datanya ditampilkan sesuai Tabel 4.3.



Gambar 4.7.  
Hasil pengujian saklar dot matrik pada kondisi *cut off*



Gambar 4.8.  
Hasil pengujian saklar dot matrik pada kondisi saturasi



Gambar 4.9.  
Hasil pengujian Ib

Tabel 4.3.  
Hasil pengujian saklar baris dot matrik

No.	Saklar	Tegangan output (v)
1	"0"	5,5
2	"1"	0,01

Tabel 4.4.  
Hasil pengujian Ib

No	Sinyal Base	Arus Base (mA)	
		Pengukuran	Perhitungan
1	"0"	7,93	9
2	"1"	0	0

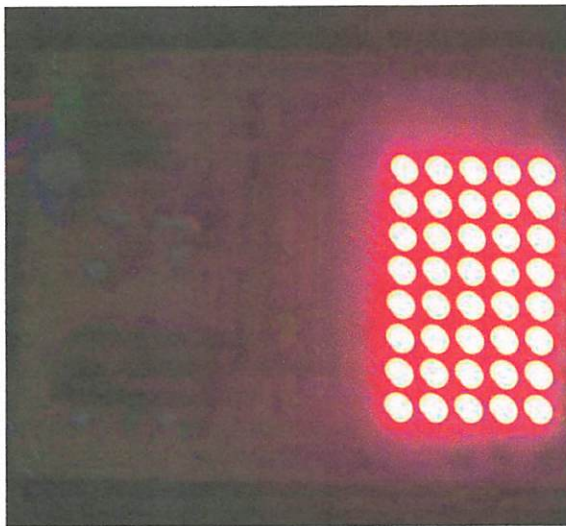
Dari Tabel 4.3. terlihat bahwa rangkaian transistor 9012 akan saturasi apabila basis diberi logika "0" dan *cut off* apabila basis berlogika "1". Sedangkan untuk perhitungan arus yang melewati Ib adalah sebagai berikut :

$$I_B = \frac{5 - 0,6}{4700}$$

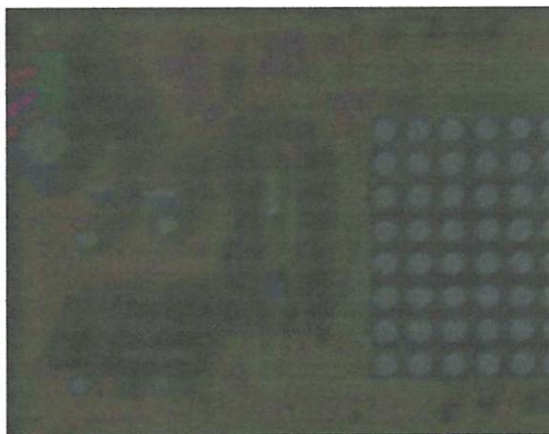
$$I_B = 0,0009 A = 9 mA$$

Dari perhitungan tersebut dan Tabel 4.4. terlihat bahwa perbedaan antara pengukuran dan perhitungan arus tidak terlalu besar.

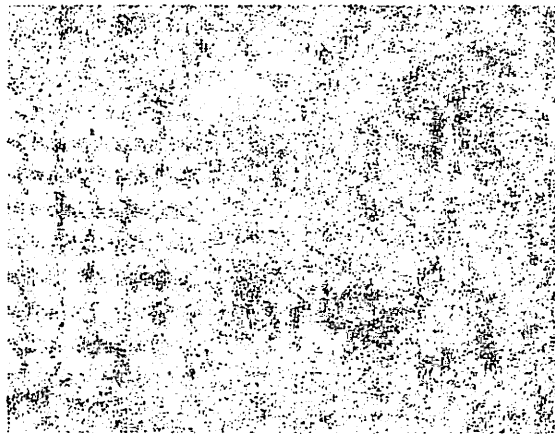
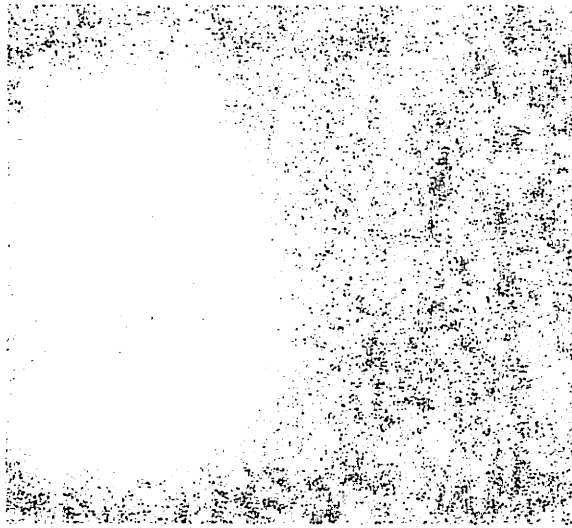
2. Hasil pengujian dot matrik ditunjukkan pada Gambar 4.10. dan 4.11., sedangkan datanya ditampilkan sesuai Tabel 4.4.



Gambar 4.10.  
Hasil pengujian dot matrik pada kondisi *on*



Gambar 4.11.  
Hasil pengujian dot matrik pada kondisi *off*



**Tabel 4.5.**  
**Hasil pengujian dot matrik**

<b>No.</b>	<b>Pin OC</b>	<b>DOT MATRIK</b>
1	"0"	ON
2	"1"	OFF

Dari Gambar 4.10 dan 4.11. serta Tabel 4.5. dapat diketahui bahwa dot matrik akan aktif apabila transistor 9012 dalam kondisi saturasi, jalur input IC 74HC573 berlogika "0" dan pin OC IC 74HC573 berlogika "0".

#### **4.5. Pengujian Rangkaian Layar Dot Matrik**

4.5.1. Tujuan pengujian adalah untuk mengetahui apakah modul dot matrik bisa menampilkan karakter tulisan dan angka yang sesuai.

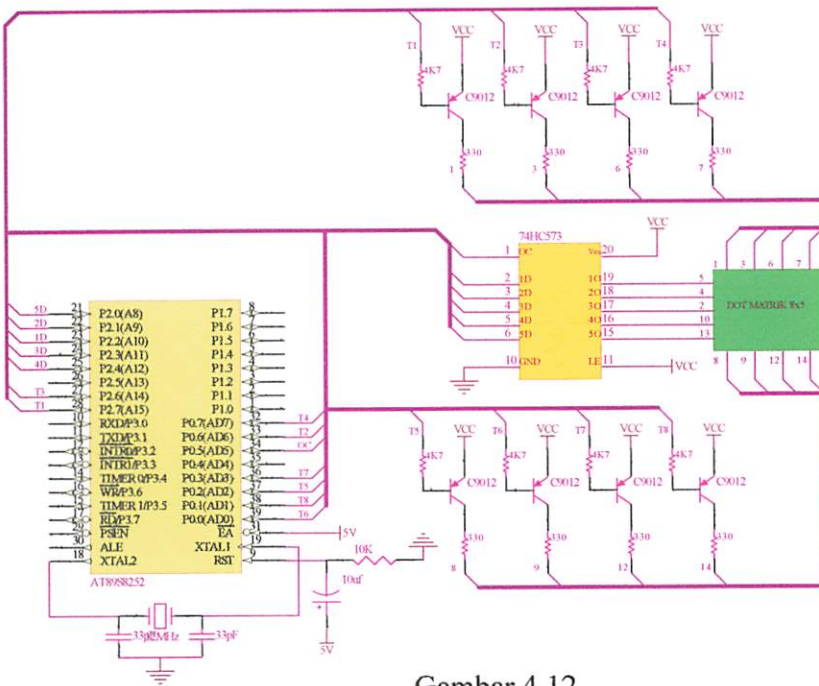
4.5.2. Peralatan yang digunakan

1. Rangkaian minimum sistem AT89S8252.
2. Modul led dot matrik
3. Power supply 5Vdc.

4.5.3. Prosedur pengujian

1. Merangkai peralatan sesuai dengan Gambar 4.12.



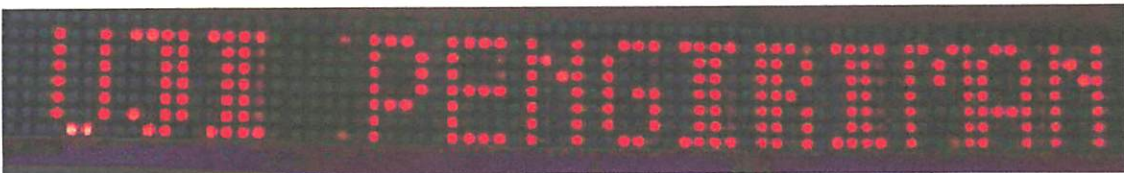


Gambar 4.12.  
Rangkaian pengujian modul dot matrik

2. Menghidupkan catu daya.
3. Membuat program assembly untuk menampilkan tulisan "UJI PENGIRIMAN", kemudian *download* program tersebut ke mikrontroler sistem layar dot matrik.
4. Mendokumentasikan foto hasil tampilan layar dot matrik

#### 4.5.4. Hasil dan analisa

Hasil pengujian berupa tampilan karakter di layar dot matrik, hasil ini dapat dilihat pada Gambar 4.13.



Gambar 4.13. Foto hasil pengujian rangkaian layar dot matrik

Dari Gambar 4.13. terlihat bahwa rangkaian layar dot matrik mampu menampilkan karakter tulisan dengan baik.

#### **4.6. Pengujian Sinyal Gelombang RLP434A**

##### **4.6.1. Tujuan pengujian adalah**

1. untuk mengetahui bentuk gelombang masukkan RLP sebelum melewati rangkaian RC.
2. untuk mengetahui bentuk gelombang masukkan RLP sesudah melewati rangkaian RC.
3. untuk mengetahui bentuk gelombang masukkan RLP setelah dikuatkan oleh IC LM358.

##### **4.6.2. Peralatan yang digunakan**

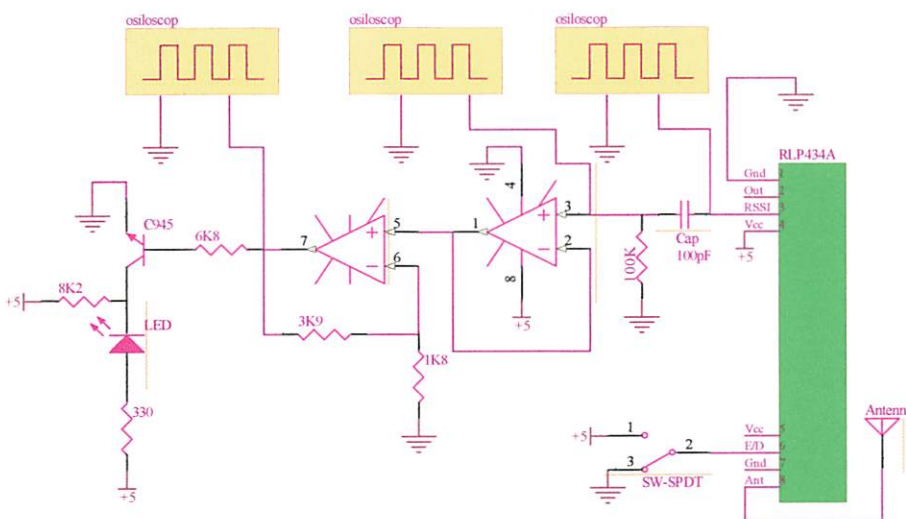
1. Rangkaian TLP434A dan RLP434A
2. *Osiloskop*

##### **4.6.3. Prosedur pengujian**

1. Menyusun rangkaian TLP434A seperti Gambar 4.3. dan rangkaian RLP434A seperti Gambar 4.14.
2. Memasang catu daya rangkaian 5 Vdc.
3. Menghubungkan input *osiloskop* pada pin 3 RLP, sebelum melewati rangkaian RC. Input ground pada *osiloskop* dihubungkan ke ground rangkaian RLP.
4. Menekan saklar TLP434A.
5. Mengamati gelombang pada *osiloskop*.



6. Melepaskan saklar TLP434A.
7. Menghubungkan input *osiloskop* pada jalur pin 3 RLP, setelah rangkaian RC. Input ground pada osiloskop dihubungkan ke ground rangkaian RLP.
8. Menekan saklar TLP434A.
9. Mengamati gelombang pada *osiloskop*.
10. Melepaskan saklar TLP434A.
11. Menghubungkan input *osiloskop* pada pin 7 IC LM358. Input ground pada osiloskop dihubungkan ke ground rangkaian RLP.
12. Menekan saklar TLP434A.
13. Mengamati gelombang pada *osiloskop*.

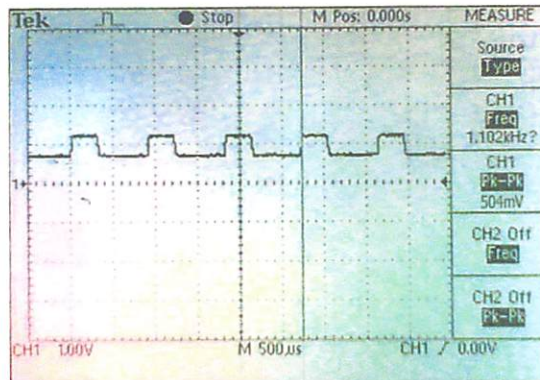


Gambar 4.14. Rangkaian pengujian gelombang RLP434A

#### 4.6.4. Hasil dan analisa

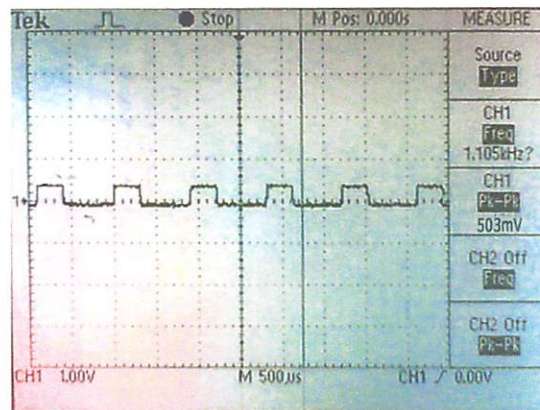
1. Hasil pengujian sinyal masukkan RLP434A sebelum rangkaian RC sesuai Gambar 4.15.

2. Hasil pengujian sinyal masukkan RLP434A setelah rangkaian RC sesuai Gambar 4.16.
3. Hasil pengujian sinyal masukkan RLP434A setelah melalui IC LM358 sesuai Gambar 4.17.

T/div = 500 $\mu$ s

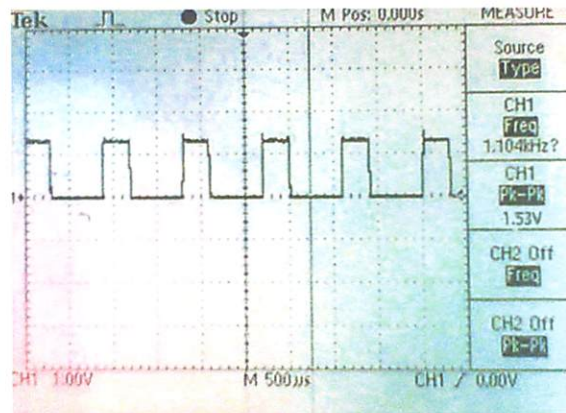
V/div = 1V

Gambar 4.15. Hasil gelombang RLP434A sebelum rangkaian RC

T/div = 500 $\mu$ s

V/div = 1V

Gambar 4.16. Hasil gelombang RLP434A setelah rangkaian RC



T/div = 500µs

V/div = 1V

Gambar 4.17. Hasil gelombang RLP434A setelah penguatan IC LM358

#### 4.7. Pengujian Frekuensi Pembawa TLP434A

4.7.1. Tujuan pengujian adalah untuk mengetahui frekuensi pembawa TLP434A.

4.7.2. Peralatan yang digunakan

1. Rangkaian TLP434A.
2. Rangkaian catu daya 5 Vdc.

4.7.3. Prosedur pengujian

1. Menghubungkan pin 2 TLP434A ke +5Vdc.
2. Menghubungkan input frekuensi counter pada antenna TLP434A dan menghubungkan *ground* pada frekuensi counter pada *ground* rangkaian TLP 434A.
3. Mengamati nilai frekuensi pada frekuensi *counter*.

4.7.4. Hasil dan analisa

Hasil pengujian dapat dilihat pada Gambar 4.18.



Gambar 4.18. Frekuensi pembawa TLP434A

dari hasil pengujian terlihat bahwa frekuensi yang dihasilkan oleh TLP434A adalah 433,9354 MHz.

#### 4.8. Pengujian Jangkauan *Wireless*

4.8.1. Tujuan pengujian adalah untuk mengetahui jarak maksimal pengoperasian alat.

4.8.2. Peralatan yang dibutuhkan

1. *Remote* dan layar dot matrik.
2. Catu daya 5Vdc.
3. Meteran.

4.8.3. Prosedur pengujian

1. Hidupkan *remote* dan layar dot matrik.
2. Edit jenis tulisan dan animasi di *remote*.
3. Kirim jenis tulisan dan animasi *remote* sesuai dengan jarak pengujian setiap 1 meter.
4. Amati jenis tulisan dan animasi layar dot matrik.

#### 4.8.4. Hasil dan analisa

Tabel 4.6  
Hasil pengujian jangkauan *wireless*

No.	Jarak (meter)	Hasil tampilan layar dot matrik
1	1	sukses
2	2	sukses
3	3	sukses
4	4	sukses
5	5	sukses
6	6	sukses
7	7	sukses
8	8	sukses
9	9	sukses
10	10	gagal
11	11	gagal

Dari Tabel 4.6. terlihat bahwa pada jarak  $< 10$  meter, *remote* dapat mengirimkan jenis tulisan dan animasi dengan baik.

#### 4.9. Pengujian Alat Keseluruhan

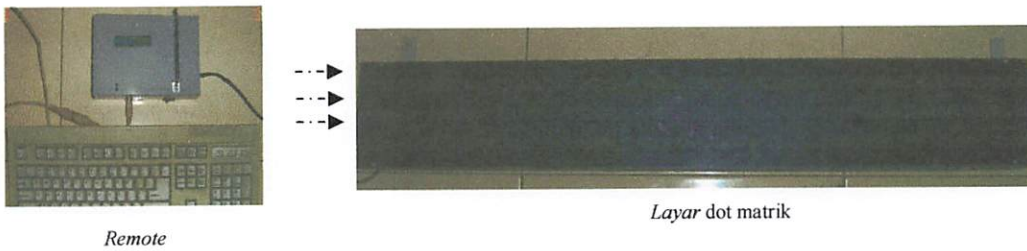
4.9.1. Tujuan pengujian adalah untuk mengetahui kinerja sistem *remote* dan layar dot matrik.

4.9.2. Peralatan yang dibutuhkan

1. *Remote* layar dot matrik dan keyboard PC PS/2.
2. Layar dot matrik.

4.9.3. Prosedur pengujian

1. Menyusun peralatan sesuai dengan Gambar 4.19.



Gambar 4.19.  
Rangkaian pengujian alat keseluruhan

2. Nyalakan *remote* dan layar dot matrik..
3. Edit informasi di *remote* sesuai dengan acuan sebagai berikut :
  - jumlah layar = 1,
  - jenis animasi = kedip,
  - Karakter = “ITN MALANG “



Gambar 4.20.  
Informasi *remote* yang dikirim

4. Pilih menu kirim data pada *remote*.
5. Tunggu dan amati hasil pengiriman data pada layar dot matrik.

#### 4.9.4. Hasil dan analisa

Hasil pengujian ditampilkan pada Gambar 4.20.



Gambar 4.21.  
Hasil pengujian alat keseluruhan

Dari Gambar 4.21. terlihat bahwa layar dot matrik mampu menampilkan informasi yang sesuai dengan *remote*.

## **BAB V**

### **PENUTUP**

#### **5.1. Kesimpulan**

Berdasarkan hasil perancangan dan pembuatan alat ini maka dapat diperoleh beberapa kesimpulan sebagai berikut :

1. Rangkaian TLP434A dan RLP434A dapat mengirim dan menerima data *remote* secara serial tanpa terjadi *error*
2. Frekuensi yang diterima oleh RLP434A berupa frekuensi digital dengan besar frekuensi pembawa TLP434A bernilai 433,9354MHz
3. Sistem *remote* dan layar dot matrik dapat berkomunikasi secara *wireless* tanpa *error* dalam jarak 9 meter.

#### **5.2. Saran**

Perlunya pengembangan terhadap layar dot matrik yang berupa penambahan jenis karakter yang bisa ditampilkan sesuai dengan kemampuan penampilan karakter LCD M1632.





## DAFTAR PUSTAKA

- [1]. Atmel Corporation. 2000. **AT89S8252 : 8 Bit Microcontroller with 8 K Bytes Flash**. <http://www.atmel.com> Tanggal akses 12 Juni 2006
- [2]. Budiharto, Widodo dan Sigit. 2005. **Elektronika Digital dan Mikroprosesor**. ANDI. Yogyakarta
- [3]. Digiware. 2006. **TLP434A & RLP434A : RF ASK Hybrid Modules for Radio Control (New Version)**. <http://www.Digi-Ware.com> Tanggal akses 10 November 2006
- [4]. Edward, Stephen A. 2007. **The PS/2 Keyboard and Mouse Interface**. Columbia University
- [5]. Fairchild. 2000. **MM74HC573 3-STATE Octal D-Type Latch**. <http://www.fairchildsemi.com> Tanggal akses 12 Juni 2006
- [6]. Seiko Instruments Inc. 1987. **Liquid Crystal Display Module M1632 User Manual**
- [7]. Semiconductor Components Industries. 2004. **LM358 : Single Supply Dual Operational Amplifier**. <http://onsemi.com> Tanggal akses 13 Juni 2006
- [8]. Widodo, Thomas. 2002. **Elektronika Dasar**. Salemba Teknika. Jakarta
- [9]. Wing Shing Computer Components Co., (H.K)Ltd. **9012 : PNP Silicon Transistor**. <http://www.datasheetarchive.com> Tanggal akses 12 Juni 2006
- [10]. Woollard, Barry. 2006. **Elektronika Praktis**. PT Pradnya Paramita. Jakarta

**LAMPIRAN**



FORMULIR BIMBINGAN SKRIPSI

Nama : Muhammad Fithroni  
Nim : 03.17.087  
Masa Bimbingan : 17-Juli-2007 s/d 17-Januari-2008  
Judul Skripsi : Pembuatan Alat Pengatur dan Penampil Layar Dot Matrik 16 Karakter Yang Bisa Diremote Secara Wireless Berbasis Mikrokontroler AT89S8252

No	Tanggal	Uraian	Paraf Pembimbing
1	24/08 11	Bab I, II, III Perbaiki	
2	30/08 11	Bab I - IV ke	
3	6/08 12	Bab IV + V perbaiki	
4	11/08 12	Bab IV + V OS	
5	9/08 13	ke Laporan	
6			
7			
8			
9			
10			

Malang,

Dosen pembimbing I

**Ir. F. Yudi Limpraptono. MT**  
NIP. Y. 1039500274





**FORMULIR BIMBINGAN SKRIPSI**

Nama : Muhammad Fithroni  
Nim : 03.17.087  
Masa Bimbingan : 17-Juli-2007 s/d 17-Januari-2008  
Judul Skripsi : Pembuatan Alat Pengatur dan Penampil Layar Dot Matrik 16 Karakter Yang Bisa Diremote Secara Wireless Berbasis Mikrokontroller AT89S8252

No	Tanggal	Uraian	Paraf Pembimbing
1	05/01 08	Bab I, II Perbaiki sumber gambar (indeks)	
2	15/01 08	Revisi blou diagram	
3	17/01 08	Bab III, Flow chart diperjelas	
4	11/02 08	Revisi Bab IV, Foto/gambar pengujian diperbaiki	
5	12/02 08	Revisi Bab V, Perbaiki kesimpulan	
6	20/02 08	Revisi makalah seminar hasil	
7	23/02 08	Acc seminar hasil	
8	04/03 08	Acc ujian kompre.	
9			
10			

Malang,

Dosen pembimbing II

**Joseph Dedy Irawan ST, MT**  
NIP. 132315178

Form S-4a



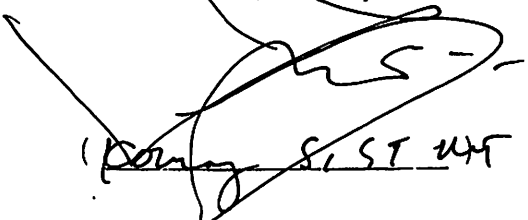
INSTITUT TEKNOLOGI NASIONAL  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO

## Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentasi T. Energi Listrik / T. Elektronika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : MUHAMMAD FITHRONI  
NIM : 03.17-087  
Perbaikan meliputi :

Malang, 05-03-2020

  
(Dony S. S. H)



INSTITUT TEKNOLOGI NASIONAL  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO

## Formulir Perbaikan Ujian Skripsi

Dalam pelaksanaan Ujian Skripsi Janjang Strata 1 Jurusan Teknik Elektro Konsentrasi T. Energi Listrik / T. Elektronika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA :  
N I M :  
Perbaikan meliputi :

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

---

Malang,

( \_\_\_\_\_ )

S\_TRAN

```

=====
; ALAMAT MEMORI INTERNAL YG DIPAKAI
=====

```

```

DataKeyboard      equ      30h
COUNT            EQU      31H
BOUNC             EQU      32H
MEMORI_DELAY      EQU      33H
DATA_AWAL         EQU      34H
JNS_ANIMASI       EQU      35H
JML_LAYAR         EQU      36H
CHAR_RAM          EQU      37H      ;37H S/D 67H UTK KARAKTER
KOLOM1            EQU      6AH
KOLOM2            EQU      6BH
KOLOM3            EQU      6CH
LAYAR             EQU      6DH
MEM_LOOP          EQU      6EH
TMP               EQU      6FH
MEMORI_DELAY_2    EQU      70H
KOLOM             EQU      71H
TMP2              EQU      72H
MEM_LOOP1         EQU      73H
MEM_LOOP2         EQU      74H
MEMORI_DELAY_1    EQU      75H
ALAMAT            EQU      76H
DATA_MASUK        EQU      77H
DATA_KELUAR       EQU      78H
WMCON             EQU      96H

```

```

=====
; posisi pin MK dengan LCD
=====

```

```

D7                bit      p2.5
D6                bit      p2.4
D5                bit      p2.3
D4                bit      p2.2
E                 bit      p2.1
RS                bit      p2.0

```

```

=====
; posisi pin MK dengan KEYBOARD
=====

```

```

KbdData           bit      P1.7
KbdClock          bit      P1.6
DATA_TLP          BIT      P3.1
ORG 00H
JMP MULAI

```

```

=====
; PROGRAM UTAMA
=====

```

```

MULAI:
CLR E
CLR RS
SETB 00H      ;INDIKATOR JENIS ANIMASI
SETB 01H      ;INDIKATOR CODE ENTER/ESC
SETB 02H      ;INDIKATOR MENU EDIT SELESAI
SETB 03H      ;INDIKATOR LAYAR 2
SETB 04H      ;INDIKATOR LAYAR 3
SETB 05H
SETB 06H
SETB 07H      ;INDIKATOR CAPS LOCK
CALL INIT_LCD
MOV ALAMAT,#0
CALL BACA_EEPROM
CJNE A,#0FFH,PAKAI_DATA_LAMA
CALL RESET_EEPROM
JMP MENU_UTAMA

PAKAI_DATA_LAMA:
MOV R0,#JNS_ANIMASI
MOV ALAMAT,#0

```



S\_TRAN

```
MOV     MEM_LOOP, #50
CALL   BACA_CHAR
MENU_UTAMA:
CALL   DISPLAY_CLEAR
MOV    DPTR, #DATA_UTAMA1
CALL   TAMPIL
CALL   BARIS2_POSISI_40H
MOV    DPTR, #DATA_UTAMA2
CALL   TAMPIL
LOOP_UTAMA:
JB     KBDLOCK, $
CALL   AMBIL_SCANCODE
CJNE  A, #0F0H, LOOP_UTAMA
JB     KBDLOCK, $
CALL   AMBIL_SCANCODE
MOV    DATAKEYBOARD, A
CJNE  A, #16H, CEK_MENU2      ;16H ASCII = 1 KYB
CALL   KIRIM_DATAX
JMP    MENU_UTAMA
CEK_MENU2:
CJNE  A, #1EH, LOOP_UTAMA    ;1EH ASCII = 2 KYB
MOV    R0, #JNS_ANIMASI
MOV    ALAMAT, #0
MOV    MEM_LOOP, #50
CALL   BACA_CHAR
CALL   DISPLAY_CLEAR
MOV    DPTR, #DATA_EDIT1
CALL   TAMPIL
MOV    A, #8DH
CALL   INTRUKSI
MOV    A, JML_LAYAR
CJNE  A, #1, CEK_JML_LAYAR2
MOV    A, #31H
CALL   TULIS_DATA
JMP    CEK_MENU2_1
CEK_JML_LAYAR2:
CJNE  A, #2, CEK_JML_LAYAR3
MOV    A, #32H
CALL   TULIS_DATA
CLR    05H
SETB  06H
JMP    CEK_MENU2_1
CEK_JML_LAYAR3:
CJNE  A, #3, CEK_MENU2_1
MOV    A, #33H
CALL   TULIS_DATA
CLR    05H
CLR    06H
JMP    CEK_MENU2_1
CEK_MENU2_1:
MOV    A, JNS_ANIMASI
CJNE  A, #1, CEK_MENU2_2
CALL   BARIS2_POSISI_40H
MOV    DPTR, #DATA_KEDIP
CALL   TAMPIL
MOV    A, #0CEH
CALL   INTRUKSI
MOV    A, #7FH                ;PANAH KE KIRI
CALL   TULIS_DATA
MOV    A, #8DH
CALL   INTRUKSI
CALL   CURSOR_ON
JMP    CEK_EDIT0
CEK_MENU2_2:
CALL   BARIS2_POSISI_40H
MOV    DPTR, #DATA_GESER
CALL   TAMPIL
MOV    A, #0CEH
```

```

S_TRAN
CALL INTRUKSI
MOV A,#7EH ;PANAH KE KANAN
CALL TULIS_DATA
MOV A,#8DH
CALL INTRUKSI
CALL CURSOR_ON
JMP CEK_EDIT0
CEK_EDIT0:
CALL KYB_MENU
CJNE A,#16H,CEK_EDIT1 ;1 LAYAR
MOV A,#8DH
CALL INTRUKSI
MOV A,#31H
CALL TULIS_DATA
MOV A,#8DH
CALL INTRUKSI
MOV JML_LAYAR,#1
SETB 05H
SETB 06H
JMP CEK_EDIT0
CEK_EDIT1:
CJNE A,#1EH,CEK_EDIT2 ;2 LAYAR
MOV A,#8DH
CALL INTRUKSI
MOV A,#32H
CALL TULIS_DATA
MOV A,#8DH
CALL INTRUKSI
MOV JML_LAYAR,#2
CLR 05H
SETB 06H
JMP CEK_EDIT0
CEK_EDIT2:
CJNE A,#26H,CEK_EDIT3 ;3 LAYAR
MOV A,#8DH
CALL INTRUKSI
MOV A,#33H
CALL TULIS_DATA
MOV A,#8DH
CALL INTRUKSI
MOV JML_LAYAR,#3
CLR 05H
CLR 06H
JMP CEK_EDIT0
CEK_EDIT3:
CJNE A,#74H,CEK_EDIT4 ;74h = KE KANAN
CALL CURSOR_OFF
CALL BARIS2_POSISI_40H
MOV DPTR,#DATA_KEDIP
CALL TAMPIL
MOV A,#0CEH
CALL INTRUKSI
MOV A,#7FH
CALL TULIS_DATA
MOV JNS_ANIMASI,#1
MOV A,#8DH
CALL INTRUKSI
CALL CURSOR_ON
JMP CEK_EDIT0
CEK_EDIT4:
CJNE A,#6BH,CEK_EDIT5 ;6Bh = KE KIRI
CALL CURSOR_OFF
CALL BARIS2_POSISI_40H
MOV DPTR,#DATA_GESER
CALL TAMPIL
MOV A,#0CEH
CALL INTRUKSI
MOV A,#7EH

```

## S\_TRAN

```

CALL    TULIS_DATA
MOV     JNS_ANIMASI, #2
MOV     A, #8DH
CALL    INTRUKSI
CALL    CURSOR_ON
JMP     CEK_EDIT0

CEK_EDIT5:
CJNE   A, #5AH, CEK_EDIT6           ;5AH = ENTER CODE
CALL   EDIT
JNB    01H, CEK_MENU2X
SETB   01H
SETB   02H
CALL   EKSEKUSI
JB     01H, MENU_UTAMAX             ;01H = "1" CODE ENTER

CEK_MENU2X:
SETB   01H
SETB   02H
MOV    A, #1EH
JMP    CEK_MENU2

CEK_EDIT6:
CJNE   A, #76H, CEK_EDIT0X         ;76H = ESC CODE
JMP    MENU_UTAMA

CEK_EDIT0X:
JMP    CEK_EDIT0

MENU_UTAMAX:
MOV    ALAMAT, #00
MOV    MEM_LOOP, #50
MOV    R0, #JNS_ANIMASI

LOOP_EDIT_EEPROM1:
MOV    DATA_KELUAR, @R0
CALL   TULIS_EEPROM
INC    ALAMAT
INC    R0
DJNZ   MEM_LOOP, LOOP_EDIT_EEPROM1
SETB   01H
SETB   02H
SETB   03H
SETB   04H
JMP    MENU_UTAMA

;=====:
KYB_MENU:
JB     KBDCLOCK, $
CALL   AMBIL_SCANCODE
CJNE   A, #0F0H, KYB_MENU
JB     KBDCLOCK, $
CALL   AMBIL_SCANCODE
MOV    DATAKEYBOARD, A
RET

;=====:
BR_SERIAL:
;BAUDRATE 9600bps, 1 BIT = 0,000104s
mov    TMOD, #01H
mov    TH0, #0FFH
mov    TLO, #9CH
clr    TF0
setb   TR0
jnb    TF0, $
clr    TR0
ret

KIRIM_DATAX:
CALL   DISPLAY_CLEAR
CALL   BARIS1_POSISI_00H
MOV    DPTR, #DATA_KIRIM_1
CALL   TAMPIL
CALL   BARIS2_POSISI_40H
MOV    DPTR, #DATA_KIRIM_2
CALL   TAMPIL
MOV    DATA_AWAL, #9
MOV    R0, #JNS_ANIMASI

```

S\_TRAN

```
MOV     ALAMAT, #0
MOV     MEM_LOOP, #50
CALL    BACA_CHAR
MOV     A, JML_LAYAR
CJNE   A, #1, DATA_2_LAYAR
MOV     BOUNC, #19
MOV     R0, #DATA_AWAL
JMP     LOOP_BYTE_DATA
DATA_2_LAYAR:
CJNE   A, #2, DATA_3_LAYAR
MOV     BOUNC, #35
MOV     R0, #DATA_AWAL
JMP     LOOP_BYTE_DATA
DATA_3_LAYAR:
MOV     BOUNC, #51
MOV     R0, #DATA_AWAL
LOOP_BYTE_DATA:
MOV     MEM_LOOP, #8
MOV     A, @R0
CALL    ULANG_PENGIRIMAN
SETB   DATA_TLP
CALL    DELAY_5MS
INC    R0
DJNZ   BOUNC, LOOP_BYTE_DATA
RET
ULANG_PENGIRIMAN:
RRC    A
JC     LOGIKA_1
JMP    LOGIKA_0
LOGIKA_0:
CLR    DATA_TLP
CALL   BR_SERIAL
SETB   DATA_TLP
CALL   BR_SERIAL
CALL   BR_SERIAL
CLR    DATA_TLP
CALL   BR_SERIAL
DJNZ   MEM_LOOP, ULANG_PENGIRIMAN
RET
LOGIKA_1:
CLR    DATA_TLP
CALL   BR_SERIAL
SETB   DATA_TLP
CALL   BR_SERIAL
CLR    DATA_TLP
CALL   BR_SERIAL
DJNZ   MEM_LOOP, ULANG_PENGIRIMAN
RET
;=====;
; SUBROUTINE ANIMASI 2 ;
;=====;
EKSEKUSI2:
CALL   DISPLAY_CLEAR
CALL   LCD_GESER_KANAN
CALL   CURSOR_OFF
SETB   03H
JMP    LOOP_EKS_ANIM1_4
;=====;
LOOP_EKS_ANIM1_4:
MOV    R6, #80H
MOV    R1, #37H
MOV    TMP, R1
MOV    MEM_LOOP, #16
MOV    MEM_LOOP1, #16
LOOP_EKS_ANIM1_4X:
MOV    MEM_LOOP, MEM_LOOP1
CALL   LOOP_ANIM2_0
INC    TMP
; RAM ALAMAT DATA LAYAR
```

```

                                S_TRAN
MOV     A,#16
CLR     C
SUBB   A, MEM_LOOP1             ;SISA POSISI TAMPIL DI LCD
MOV     MEM_LOOP2,A
CJNE   A,#0, LOOP_EKS_ANIM1_4XX
MOV     MEM_LOOP1,#15
CALL   DELAY_05S_TLP          ;DELAY ANIMASI
JNB    02H, ANIMASI_2_SLS
JMP    LOOP_EKS_ANIM1_4X
;=====
LOOP_EKS_ANIM1_4XX:
MOV     A, JML_LAYAR
CJNE   A,#1, LOOP_EKS_ANIM1_4XXX
MOV     R1,#37H
JMP    LOOP_EKS_ANIM1_4XXXXX
LOOP_EKS_ANIM1_4XXX:
CJNE   A,#2, LOOP_EKS_ANIM1_4XXXX
JB     03H, LOOP_EKS_ANIM1_4XXX0
MOV     R1,#37H
JMP    LOOP_EKS_ANIM1_4XXXXX
LOOP_EKS_ANIM1_4XXX0:
MOV     R1,#47H
SETB   03H
JMP    LOOP_EKS_ANIM1_4XXXXX
LOOP_EKS_ANIM1_4XXXX:
JB     03H, LOOP_EKS_ANIM1_4XXX00
JB     04H, LOOP_EKS_ANIM1_4XXX000
MOV     R1,#37H
JMP    LOOP_EKS_ANIM1_4XXXXX
LOOP_EKS_ANIM1_4XXX00:
MOV     R1,#47H
SETB   03H
SETB   04H
JMP    LOOP_EKS_ANIM1_4XXXXX
LOOP_EKS_ANIM1_4XXX000:
MOV     R1,#57H
JMP    LOOP_EKS_ANIM1_4XXXXX
;=====
ANIMASI_2_SLS:
RET
;=====
LOOP_EKS_ANIM1_4XXXXX:
MOV     MEM_LOOP, MEM_LOOP2    ;MEM_LOOP2 = RAM DATA SISA POSISI
LCD
CALL   LOOP_ANIM2_0X           ;POSISI R1 = 37H? WHY?
CALL   DELAY_05S_TLP          ;DELAY ANIMASI
JNB    02H, ANIMASI_2_SLS
DEC    MEM_LOOP1
MOV     A, MEM_LOOP2
CJNE   A,#15, LOOP_EKS_ANIM1_4X
MOV     A, JML_LAYAR
CJNE   A,#1, LOOP_ANIM2_X1
JMP    LOOP_EKS_ANIM1_4
LOOP_ANIM2_X1:
CJNE   A,#2, LOOP_ANIM2_X2
JNB    03H, LOOP_X1
MOV     R6,#80H
MOV     R1,#47H
MOV     TMP,R1
MOV     MEM_LOOP,#16
MOV     MEM_LOOP1,#16
CLR     03H
JMP    LOOP_EKS_ANIM1_4X
LOOP_X1:
SETB   03H
JMP    LOOP_EKS_ANIM1_4
;=====

```

## S\_TRAN

```

LOOP_ANIM2_X2:
    JNB     03H, LOOP_X2
    MOV     R6, #80H
    MOV     R1, #47H
    MOV     TMP, R1
    MOV     MEM_LOOP, #16
    MOV     MEM_LOOP1, #16
    CLR     03H
    JMP     LOOP_EKS_ANIM1_4X

LOOP_X2:
    JNB     04H, LOOP_X3
    MOV     R6, #80H
    MOV     R1, #57H
    MOV     TMP, R1
    MOV     MEM_LOOP, #16
    MOV     MEM_LOOP1, #16
    CLR     03H
    CLR     04H
    JMP     LOOP_EKS_ANIM1_4X

LOOP_X3:
    SETB    03H
    SETB    04H
    JMP     LOOP_EKS_ANIM1_4

;=====;
LOOP_ANIM2_0:
    MOV     A, #80H
    CALL    INTRUKSI
    MOV     R1, TMP

LOOP_ANIM2_0x:
    MOV     A, @R1
    CALL    TULIS_DATA
    INC     R1
    DJNZ   MEM_LOOP, LOOP_ANIM2_0X
    RET

;=====;
CEK_ANIM2_ENT:
    CJNE   A, #5AH, DELAY_EKS2_SLS           ;5AH = ENT CODE
    SETB   01H
    CLR    02H
    RET

;=====;
DELAY_EKS2_SLS:
    SETB   01H
    SETB   02H
    RET

;=====;
;=====;
EKSEKUSI2X:
    JMP    EKSEKUSI2

EKSEKUSI:
    SETB   01H
    SETB   02H
    MOV    A, JNS_ANIMASI
    CJNE   A, #1, EKSEKUSI2X
    MOV    B, JML_LAYAR

LOOP_EKS0:
    CALL   DISPLAY_CLEAR
    CALL   CURSOR_OFF
    MOV    R1, #37H
    CALL   TAMPIL_LAYAR
    DJNZ   JML_LAYAR, LOOP_EKS1

LOOP_EKS:
    CALL   KYB_MENU
    CJNE   A, #76H, LOOP_EKS_ENT           ;ESC CODE
    MOV    JML_LAYAR, B
    CLR    01H
    CLR    02H
    RET

```

## S\_TRAN

```

LOOP_EKS_ENT:
    CJNE    A, #5AH, LOOP_EKS           ; ENT CODE
    MOV     JML_LAYAR, B
    SETB    01H
    CLR     02H
    RET

LOOP_EKS1:
    CALL    LOOP_DELAY_EKS0
    JNB     02H, LOOP_EKS_SLS
    MOV     R1, #47H
    CALL    DISPLAY_CLEAR
    CALL    TAMPIL_LAYAR
    DJNZ    JML_LAYAR, LOOP_EKS2
    CALL    LOOP_DELAY_EKS0
    JNB     02H, LOOP_EKS_SLS
    MOV     JML_LAYAR, B
    JMP     LOOP_EKS0

LOOP_EKS2:
    CALL    LOOP_DELAY_EKS0
    JNB     02H, LOOP_EKS_SLS
    MOV     R1, #57H
    CALL    DISPLAY_CLEAR
    CALL    TAMPIL_LAYAR
    CALL    LOOP_DELAY_EKS0
    JNB     02H, LOOP_EKS_SLS
    MOV     JML_LAYAR, B
    JMP     LOOP_EKS0

LOOP_EKS_SLS:
    MOV     JML_LAYAR, B
    RET
;=====;
LOOP_DELAY_EKS0:
    MOV     MEMORI_DELAY, #200
    MOV     MEMORI_DELAY_1, #2

LOOP_DELAY_EKS1:
    mov     TMOD, #01H
    mov     TH0, #0EDH
    mov     TL0, #0FFH
    clr     TF0
    setb    TR0

LOOP_DELAY_ANIM1:
    jnb     TF0, CEK_KYB_SINYAL
    clr     TR0
    DJNZ    MEMORI_DELAY, LOOP_DELAY_EKS1
    MOV     MEMORI_DELAY, #200
    DJNZ    MEMORI_DELAY_1, LOOP_DELAY_EKS1
    RET

CEK_KYB_SINYAL:
    JB      KBDCLOCK, LOOP_DELAY_ANIM1
    CLR     TR0
    CALL    AMBIL_SCANCODE
    CJNE    A, #0F0H, CEK_KYB_SINYAL
    JB      KBDCLOCK, $
    CALL    AMBIL_SCANCODE
    MOV     DATAKEYBOARD, A
    CJNE    A, #76H, CEK_ANIM1_ENT      ; 76H = ESC CODE
    CLR     01H
    CLR     02H
    MOV     JML_LAYAR, B
    RET

CEK_ANIM1_ENT:
    CJNE    A, #5AH, DELAY_EKS_SLS      ; 5AH = ENT CODE
    SETB    01H
    CLR     02H
    MOV     JML_LAYAR, B
    RET

DELAY_EKS_SLS:
    SETB    01H

```

S\_TRAN

```
SETB    02H
MOV     JML_LAYAR,B
RET
```

```
=====
subroutine LCD
=====
```

```
init_lcd:
MOV     A,#00101111b      ;function set1
CLR     RS
CALL   kirim1
CALL   DELAY_5MS
MOV     A,#00101111b      ;function set2
CALL   intruksi
MOV     A,#00001000b      ;display off
CALL   intruksi
MOV     A,#00000001b      ;display clear
CALL   intruksi
MOV     A,#00000110b      ;entry mode
CALL   intruksi
MOV     A,#00001100b      ;display on,cursor off,blink off
CALL   intruksi
RET
```

```
=====
kirim1:
MOV     C,Acc.7
MOV     D7,C
MOV     C,Acc.6
MOV     D6,C
MOV     C,Acc.5
MOV     D5,C
MOV     C,Acc.4
MOV     D4,C
SETB    E
CLR     E
RET
```

```
=====
kirim2:
MOV     C,Acc.3
MOV     D7,C
MOV     C,Acc.2
MOV     D6,C
MOV     C,Acc.1
MOV     D5,C
MOV     C,Acc.0
MOV     D4,C
SETB    E
CLR     E
CALL   DELAY_5MS
RET
```

```
=====
intruksi:
CLR     RS
CALL   kirim1
CALL   kirim2
RET
```

```
=====
tulis_data:
SETB    RS
CALL   kirim1
CALL   kirim2
RET
```

```
=====
tampil:
MOV     1DH,#10H
loop1:
MOV     A,#00H
MOVC   A,@A+DPTR
CALL   tulis_data
```

;ada delay 5 ms



S\_TRAN

```
INC DPTR
DJNZ 1DH, loop1
RET
```

```
;=====
display_clear:
mov A, #01H
call intruksi
ret
```

```
;=====
cursor_home:
mov A, #03H
call intruksi
ret
```

```
;=====
BARIS1_posisi_00H:
MOV A, #080H
CALL intruksi
RET
```

```
;=====
BARIS2_posisi_40H:
MOV A, #0C0H
CALL intruksi
RET
```

;ada delay 5 ms

```
;=====
BARIS1_POSISI_16:
MOV A, #08FH
CALL INTRUKSI
RET
```

```
;=====
BARIS2_POSISI_16:
MOV A, #0CFH
CALL INTRUKSI
RET
```

```
;=====
CURSOR_ON:
MOV A, #00001111B
CALL INTRUKSI
RET
```

```
;=====
CURSOR_OFF:
MOV A, #00001100b
CALL INTRUKSI
RET
```

```
;=====
KONV_LCD:
MOV B, #10
DIV AB
MOV R7, B
MOV B, #10
DIV AB
MOV A, #30H
CALL TULIS_DATA
MOV A, R7
ADD A, #30H
CALL TULIS_DATA
RET
```

```
;=====
JUDUL:
MOV DPTR, #JUDUL1
CALL TAMPIL
CALL BARIS2_POSISI_40H
MOV DPTR, #JUDUL2
CALL TAMPIL
CALL DELAY_2S
CALL DISPLAY_CLEAR
RET
```

```
;=====
PIL_ANIMASI:
```

```

                                S_TRAN
MOV     DPTR,#P1_ANIMASI
CALL   TAMPIL
CALL   BARIS2_POSISI_40H
MOV     DPTR,#P2_ANIMASI
CALL   TAMPIL
RET

;=====
LCD_GESER_KIRI:
MOV     A,#00000100B
CALL   INTRUKSI
RET

;=====
LCD_GESER_KANAN:
MOV     A,#00000110B
CALL   INTRUKSI
RET

;=====
HAPUS_RAM:
MOV     R1,#37H
MOV     R0,#48
CLR     A
MOV     DPTR,#DATA_KOSONG
MOVC   A,@A+DPTR

LOOP_RAM:
MOV     @R1,A
INC     R1
DJNZ   R0,LOOP_RAM
RET

;=====
; ANIMASI 3 LAYAR
;=====
EDIT:
CALL   CURSOR_ON
MOV     R1,#37H
MOV     KOLOM1,#80H
MOV     KOLOM2,#80H
MOV     KOLOM3,#80H
MOV     LAYAR,#1

LOOP_A1:
MOV     A,LAYAR
CJNE   A,#1,CEK_LAYAR2
CALL   LAYAR_1
JNB    02H,EDIT_SLS
JMP    LOOP_A1

CEK_LAYAR2:
CJNE   A,#2,CEK_LAYAR3
CALL   LAYAR_2
JNB    02H,EDIT_SLS
JMP    LOOP_A1

CEK_LAYAR3:
CALL   LAYAR_3
JNB    02H,EDIT_SLS
JMP    LOOP_A1

EDIT_SLS:
RET

;=====
LAYAR_1:
CALL   DISPLAY_CLEAR
CALL   BARIS2_POSISI_40H
MOV     DPTR,#DATA_LAYAR1
CALL   TAMPIL
CALL   BARIS1_POSISI_00H
MOV     LAYAR,#1
MOV     R1,#37H
MOV     KOLOM,KOLOM1
CALL   TAMPIL_LAYAR
MOV     R1,#37H
MOV     KOLOM,#80H

```

## S\_TRAN

```

LOOP_L1:
    JB     KBDCLOCK,$
    CALL  AMBIL_SCANCODE
    CJNE  A,#0F0H,LOOP_L1
    JB     KBDCLOCK,$
    CALL  AMBIL_SCANCODE
    MOV   DATAKEYBOARD,A
    CJNE  A,#58H,OLD_CHAR1
    CPL   07H
    JMP   LOOP_L1
OLD_CHAR1:
    CALL  SCAN_KEY
    JNB   02H,L1_SLS           ;02H="0" INDIKATOR SELESAI
    MOV   A,LAYAR
    CJNE  A,#1,L2_SLS
    MOV   KOLOM1,KOLOM
    JMP   LOOP_L1
L1_SLS:
    RET
;=====
LAYAR_2:
    CALL  DISPLAY_CLEAR
    CALL  BARIS2_POSISI_40H
    MOV   DPTR,#DATA_LAYAR2
    CALL  TAMPIL
    CALL  BARIS1_POSISI_00H
    MOV   LAYAR,#2
    MOV   R1,#47H
    MOV   KOLOM,KOLOM2
    CALL  TAMPIL_LAYAR
    MOV   R1,#47H
    MOV   KOLOM,#80H
LOOP_L2:
    JB     KBDCLOCK,$
    CALL  AMBIL_SCANCODE
    CJNE  A,#0F0H,LOOP_L2
    JB     KBDCLOCK,$
    CALL  AMBIL_SCANCODE
    MOV   DATAKEYBOARD,A
    CJNE  A,#58H,OLD_CHAR2
    CPL   07H
    JMP   LOOP_L2
OLD_CHAR2:
    CALL  SCAN_KEY
    JNB   02H,L2_SLS           ;02H="0" INDIKATOR SELESAI
    MOV   A,LAYAR
    CJNE  A,#2,L2_SLS
    MOV   KOLOM2,KOLOM
    JMP   LOOP_L2
L2_SLS:
    RET
;=====
LAYAR_3:
    CALL  DISPLAY_CLEAR
    CALL  BARIS2_POSISI_40H
    MOV   DPTR,#DATA_LAYAR3
    CALL  TAMPIL
    CALL  BARIS1_POSISI_00H
    MOV   LAYAR,#3
    MOV   R1,#57H
    MOV   KOLOM,KOLOM3
    CALL  TAMPIL_LAYAR
    MOV   R1,#57H
    MOV   KOLOM,#80H
LOOP_L3:
    JB     KBDCLOCK,$
    CALL  AMBIL_SCANCODE
    CJNE  A,#0F0H,LOOP_L3

```

S\_TRAN

```
JB      KBDLOCK, $
CALL   AMBIL_SCANCODE
MOV    DATAKEYBOARD, A
CJNE  A, #58H, OLD_CHAR3
CPL   07H
JMP   LOOP_L3
```

OLD\_CHAR3:

```
CALL   SCAN_KEY
JNB   02H, L3_SLS           ;02H="0" INDIKATOR SELESAI
MOV   A, LAYAR
CJNE  A, #3, L3_SLS
MOV   KOLOM3, KOLOM
JMP   LOOP_L3
```

L3\_SLS:

RET

=====:

RESET\_RAM:

```
MOV   R1, #37H
MOV   MEM_LOOP, #48
CLR   A
MOV   DPTR, #DATA_KOSONG
MOVC  A, @A+DPTR
```

LOOP\_RESET\_RAM:

```
MOV   @R1, A
INC   R1
DJNZ  MEM_LOOP, LOOP_RESET_RAM
RET
```

=====:

TAMPIL\_LAYAR:

```
CALL   LCD_GESER_KANAN
MOV   KOLOM, #80H
```

LOOP\_TL:

```
MOV   A, @R1
CALL   TULIS_DATA
INC   R1
INC   KOLOM
MOV   A, KOLOM
CJNE  A, #8FH, LOOP_TL
MOV   A, @R1
CALL   TULIS_DATA
CALL   BARIS1_POSISI_00H
RET
```

=====:

SCAN\_KEY:

```
MOV   A, DATAKEYBOARD
```

BACKSPACE:

```
CJNE  A, #66H, ESC
MOV   A, KOLOM
CJNE  A, #80H, HAPUS_CHAR
RET
```

ESC:

```
CJNE  A, #76H, ENT
CLR   02H
CLR   01H
RET
```

ENT:

```
CJNE  A, #5AH, DEL
CLR   02H
SETB  01H
RET
```

DEL:

```
CJNE  A, #71H, F1
CLR   A
MOV   DPTR, #DATA_KOSONG
MOVC  A, @A+DPTR
MOV   @R1, A
CALL   TULIS_DATA
MOV   A, KOLOM
```

## S\_TRAN

```

CALL      INTRUKSI
RET
HAPUS_CHAR:
CALL      LCD_GESER_KANAN
DEC       R1
MOV       A, R1
MOV       R3, A
INC       R1
DEC       KOLOM
MOV       B, KOLOM
MOV       A, KOLOM
CALL      INTRUKSI
LOOP_HAPUS_CHAR:
MOV       A, @R1
DEC       R1
MOV       @R1, A
CALL      TULIS_DATA
INC       R1
INC       R1
INC       KOLOM
MOV       A, KOLOM
CJNE     A, #8FH, LOOP_HAPUS_CHAR
DEC       R1
CLR       A
MOV       DPTR, #DATA_KOSONG
MOVC     A, @A+DPTR
MOV       @R1, A
CALL      TULIS_DATA
MOV       A, R3
MOV       R1, A
MOV       KOLOM, B
MOV       A, KOLOM
CALL      INTRUKSI
RET

F1:
CJNE     A, #5H, F2
MOV       LAYAR, #1
SETB     02H
RET

F2:
CJNE     A, #6H, F3
JB       05H, F2_SLS
MOV       LAYAR, #2

F2_SLS:
SETB     02H
RET

F3:
CJNE     A, #4H, KIRI
JB       06H, F3_SLS
MOV       LAYAR, #3

F3_SLS:
SETB     02H
RET

KIRI:
CJNE     A, #6BH, KANAN
MOV       A, KOLOM
CJNE     A, #80H, KE_KIRI
RET

KE_KIRI:
DEC       KOLOM
DEC       R1
MOV       A, KOLOM
CALL      INTRUKSI
RET

KANAN:
CJNE     A, #74H, SPACEBAR
MOV       A, KOLOM
CJNE     A, #8FH, KE_KANAN

```

## S\_TRAN

```

RET
KE_KANAN:
  INC    KOLOM
  INC    R1
  MOV    A, KOLOM
  CALL   INTRUKSI
  RET

SPACEBAR:
  CJNE  A, #29H, ANGKA
  MOV   A, KOLOM
  CJNE  A, #8FH, SPASI_CHAR
  RET

SPASI_CHAR:
  MOV   A, R1
  MOV   R3, A
  MOV   A, KOLOM
  MOV   B, #8fh

LOOP_SPASI1:
  INC   R1
  INC   A
  CJNE  A, #8FH, LOOP_SPASI1
  MOV   TMP, @R1
  CLR   A
  MOV   DPTR, #DATA_KOSONG
  MOVC  A, @A+DPTR
  CJNE  A, TMP, SPASI_SLS
  CALL  LCD_GESER_KIRI
  CALL  BARIS1_POSISI_16

LOOP_SPASI2:
  DEC   R1
  MOV   A, @R1
  INC   R1
  MOV   @R1, A
  CALL  TULIS_DATA
  DEC   R1
  DEC   B
  MOV   A, B
  CJNE  A, KOLOM, LOOP_SPASI2
  CLR   A
  MOV   DPTR, #DATA_KOSONG
  MOVC  A, @A+DPTR
  MOV   @R1, A
  CALL  TULIS_DATA
  CALL  LCD_GESER_KANAN
  INC   KOLOM
  INC   R1
  MOV   A, KOLOM
  CALL  INTRUKSI
  RET

SPASI_SLS:
  MOV   A, R3
  MOV   R1, A
  RET

ANGKA:
  MOV   A, KOLOM
  CJNE  A, #8FH, NEXT_CHAR
  CALL  KONVERSI_SCANCODE
  MOV   @R1, DATAKEYBOARD
  MOV   A, @R1
  CALL  TULIS_DATA
  MOV   A, #8FH
  CALL  INTRUKSI
  MOV   KOLOM, #8FH
  RET

NEXT_CHAR:
  CALL  LCD_GESER_KANAN
  CALL  Konversi_Scancode
  MOV   @R1, DATAKEYBOARD

```

## S\_TRAN

```

MOV     A,@R1
CALL    TULIS_DATA
INC     KOLOM
INC     R1
RET

```

```

;=====
Konversi_ScanCode:

```

```

JB      07H,DATA_BAWAH
MOV     A,DATAKEYBOARD
MOV     DPTR,#CAPS
MOVC   A,@A+DPTR
MOV     DATAKEYBOARD,A
RET

```

```

DATA_BAWAH:

```

```

MOV     A,DATAKEYBOARD
MOV     DPTR,#TableKeyboard
MOVC   A,@A+DPTR
MOV     DataKeyboard,A
RET

```

```

;=====
Ambil_scanCode:

```

```

CLR     A
JNB    KbdClock,$
MOV     R0,#08h

```

```

Ambil_ScanCode2:

```

```

JB      KbdClock,$
MOV     C,KbdData
RR      A
MOV     acc.7,C
JNB    KbdClock,$
DJNZ   R0,Ambil_ScanCode2
JB      KbdClock,$
JNB    KbdClock,$
JB      KbdClock,$
JNB    KbdClock,$
RET

```

```

;=====
delay:

```

```

MOV     R5,#0

```

```

delay1:

```

```

MOV     R4,#02fh
DJNZ   R4,$
DJNZ   R5,delay1
RET

```

```

;=====
TableKeyboard:

```

```

DB 00
DB 0F9H
DB 00
DB 0F5H,0F3H,0F1H,0F2H,0FCH
DB 00H
DB 0FAH,0F8H,0F6H,0F4H
DB 0C0H
DB ' '
DB 00H
DB 00H
DB 0C1H
DB 0C2H
DB 00H
DB 0C3H
DB 'q1'
DB 00H,00H,00H
DB 'zsaw2'
DB 00H,00H
DB 'cxde43'
DB 00H,00H
DB 'vftr5'
DB 00H,00H

```

S\_TRAN

```

DB 'nbhgy6'
DB 00H,00H,00H
DB 'mju78'
DB 00H,00H
DB ',kio09'
DB 00H,00H
DB './!;p-'
DB 00H,00H,00H
DB 27H
DB 00H
DB '['
DB 00H,00H
DB 0C5H
DB 0C6H
DB 0AH
DB ']'
DB 00H
DB '\'
DB 00H,00H,00H,00H,00H,00H,00H,00H
DB 08H
DB 00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
DB 0DH
DB 00H
DB 0FBH
DB 00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
DB 0F7H

```

CAPS:

```

DB 00
DB 0F9H
DB 00
DB 0F5H,0F3H,0F1H,0F2H,0FCH
DB 00H
DB 0FAH,0F8H,0F6H,0F4H
DB 0C0H
DB 0F3H
DB 00H
DB 00H
DB 0C1H
DB 0C2H
DB 00H
DB 0C3H
DB 'Q!'
DB 00H,00H,00H
DB 'z5aw@'
DB 00H,00H
DB 'CXDE$#'
DB 00H,00H
DB ' VFTR'
DB 25H,00H,00H
DB 'NBHGYA'
DB 00H,00H,00H
DB 'MJU&*'
DB 00H,00H
DB '"KIO)('
DB 00H,00H
DB '>?L:P_'
DB 00H,00H,00H
DB 27H
DB 00H
DB '{+'
DB 00H,00H
DB 0C5H
DB 0C6H
DB 0AH
DB '}'
DB 00H
DB '|'
DB 00H,00H,00H,00H,00H,00H,00H,00H

```



S\_TRAN

```

DB 08H
DB 00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
DB 0DH
DB 00H
DB 0FBH
DB 00H,00H,00H,00H,00H,00H,00H,00H,00H,00H,00H
DB 0F7H

```

KOSONG:

```

DB ' '

```

```

;=====
; SUBROUTINE EEPROM
;=====

```

TULIS\_EEPROM:

```

MOV WMCON,#00011010B
MOV DPH,#00H
MOV DPL,ALAMAT
MOV A,DATA_KELUAR
MOVX @DPTR,A

```

TUNGGU\_TULIS\_EEPROM:

```

MOV A,WMCON
CJNE A,#00011010B,TUNGGU_TULIS_EEPROM
MOV WMCON,#00001010B
RET

```

```

;=====

```

BACA\_EEPROM:

```

MOV WMCON,#00001010B
MOV DPH,#00H
MOV DPL,ALAMAT
MOVX A,@DPTR
MOV DATA_MASUK,A
RET

```

```

;=====

```

BACA\_CHAR:

```

CALL BACA_EEPROM
MOV @R0,A
INC R0
INC ALAMAT
DJNZ MEM_LOOP,BACA_CHAR
RET

```

```

;=====

```

RESET\_EEPROM:

```

MOV ALAMAT,#0
MOV DATA_KELUAR,#1
CALL TULIS_EEPROM
INC ALAMAT
MOV DATA_KELUAR,#1
CALL TULIS_EEPROM
MOV MEM_LOOP,#48
MOV DPTR,#KOSONG
MOVC A,@A+DPTR

```

LOOP\_RESET\_EEPROM:

```

INC ALAMAT
MOV DATA_KELUAR,A
CALL TULIS_EEPROM
DJNZ MEM_LOOP,LOOP_RESET_EEPROM
RET

```

```

;=====
; subroutine waktu/delay
;=====

```

DELAY\_2S:

```

MOV BOUNC,#2

```

TUNGGU\_DELAY\_2S:

```

CALL DELAY_1S
DJNZ BOUNC,TUNGGU_DELAY_2S
RET

```

```

;-----
delay_1s:

```

```

S_TRAN
    mov     MEMORI_DELAY,#200
tunggu_1s:
    call    delay_5ms
    djnz   MEMORI_DELAY,tunggu_1s
    ret

;-----
DELAY_05S:
    MOV     MEMORI_DELAY,#100
TUNGGU_05S:
    CALL    DELAY_5MS
    DJNZ   MEMORI_DELAY,TUNGGU_05S
    RET

;-----
DELAY_05S_TLP:
    MOV     MEMORI_DELAY,#100
TUNGGU_05S_TLP:
    mov     TMOD,#01H
    mov     TH0,#0EDH
    mov     TLO,#0FFH
    clr     TFO
    setb    TRO
LOOP_DELAY_TLP:
    jnb     TFO,CEK_ENT_ANIM2
    clr     TRO
    DJNZ   MEMORI_DELAY,TUNGGU_05S_TLP
    RET
CEK_ENT_ANIM2:
    JB      KBDCLOCK,LOOP_DELAY_TLP
    CLR     TRO
    CALL    AMBIL_SCANCODE
    CJNE   A,#0F0H,CEK_ENT_ANIM2
    JB      KBDCLOCK,$
    CALL    AMBIL_SCANCODE
    MOV     DATAKEYBOARD,A
    CJNE   A,#76H,CEK_ANIM2_ENT_TLP      ;76H = ESC CODE
    CLR     01H
    CLR     02H
    RET
CEK_ANIM2_ENT_TLP:
    CJNE   A,#5AH,DELAY_EKS2_TLP_SLS      ;5AH = ENT CODE
    SETB   01H
    CLR    02H
    RET
DELAY_EKS2_TLP_SLS:
    SETB   01H
    SETB   02H
    RET

;-----
DELAY_01S:
    MOV     MEMORI_DELAY,#20
TUNGGU_DELAY_01S:
    CALL    DELAY_5MS
    DJNZ   MEMORI_DELAY,TUNGGU_DELAY_01S
    RET

;-----
delay_5ms:
    mov     TMOD,#01H
    mov     TH0,#0EDH
    mov     TLO,#0FFH
    clr     TFO
    setb    TRO
    jnb     TFO,$
    clr     TRO
    ret

;-----
DELAY_10MS:
    MOV     MEMORI_DELAY,#2
TUNGGU_DELAY_10MS:

```

```

                                S_TRAN
CALL      DELAY_5MS
DJNZ     MEMORI_DELAY, TUNGGU_DELAY_10MS
RET
;-----
DELAY_30MS:
MOV      MEMORI_DELAY, #6
TUNGGU_DELAY_30MS:
CALL     DELAY_5MS
DJNZ     MEMORI_DELAY, TUNGGU_DELAY_30MS
RET
;-----
DELAY_200us:
MOV      MEMORI_DELAY_1, #10
TUNGGU_DELAY_200us:
MOV      MEMORI_DELAY_2, #20
DJNZ     MEMORI_DELAY_2, $
DJNZ     MEMORI_DELAY_1, TUNGGU_DELAY_200us
RET
;-----
DELAY_100us:
MOV      MEMORI_DELAY_1, #10
TUNGGU_DELAY_100us:
MOV      MEMORI_DELAY_2, #10
DJNZ     MEMORI_DELAY_2, $
DJNZ     MEMORI_DELAY_1, TUNGGU_DELAY_100us
RET
;-----
JUDUL1:
DB       'REMOTE  KEYBOARD'
JUDUL2:
DB       '    BY ONNY    '
DATA_KOSONG:
DB       ' '
P1_ANIMASI:
DB       '1. ANIMASI 1  '
P2_ANIMASI:
DB       '2. ANIMASI 2  '
EKS_ANIMASI1:
DB       '    PROGRAM    '
EKS_ANIMASI2:
DB       '  DIJALANKAN  '
DATA_LAYAR1:
DB       '    LAYAR 1    '
DATA_LAYAR2:
DB       '    LAYAR 2    '
DATA_LAYAR3:
DB       '    LAYAR 3    '
DATA_UTAMA1:
DB       '1. KIRIM DATA  '
DATA_UTAMA2:
DB       '2. EDIT TAMPILAN'
DATA_EDIT1:
DB       'JML. LAYAR = 1/3'
DATA_KEDIP:
DB       'ANIMASI KEDIP( )'
DATA_GESER:
DB       'ANIMASI GESER( )'
DATA_ANIMASI:
DB       'ANIMASI GESER( )'
DATA_ANIMASI2:
DB       'ANIMASI KEDIP( )'
DATA_KIRIM_1:
DB       '    DATA DIKIRIM  '
DATA_KIRIM_2:
DB       '  HARAP MENUNGGU  '
END

```

terima\_X1

```

=====
; ALAMAT MEMORI INTERNAL YG DIPAKAI
;=====
MEMORI_DELAY EQU 33H
MEMORI_DELAY_1 EQU 34H
MEMORI_DELAY_2 EQU 35H
MEM_LOOP EQU 36H
MEM_CHAR EQU 37H
MEM_LAYAR EQU 38H
ALAMAT EQU 39H
DATA_KELUAR EQU 3AH
MEM_TMP1 EQU 3BH
JNS_ANIMASI EQU 3CH
JML_LAYAR EQU 3DH
CHAR_RAM EQU 3EH ;4DH = KARAKTER KE 16
CHAR_17 EQU 4EH ;5DH = KARAKTER KE 32
CHAR_33 EQU 5EH ;6DH = KARAKTER KE 48
CHAR_49 EQU 6EH
COUNT_1 EQU 6FH
COUNT_2 equ 70H
DATA_MASUK EQU 30H
MEM_TMP2 EQU 31H
LOOP_ASCII EQU 32H
;-----
;00H INDIKATOR BIT JENIS ANIMASI,"1"=ANIMASI 1,"0"=ANIMASI 2:
WMCON EQU 96H
;-----
; posisi pin MK dengan 74HC573
;-----
OC1 BIT P0.5
OC2 BIT P0.4
OC3 BIT P3.7
OC4 BIT P3.6
OC5 BIT P3.5
OC6 BIT P3.4
OC7 BIT P3.1
OC8 BIT P3.0
OC9 BIT P1.7
OC10 BIT P1.6
OC11 BIT P1.5
OC12 BIT P1.4
OC13 BIT P1.3
OC14 BIT P1.2
OC15 BIT P1.1
OC16 BIT P1.0
;-----
; posisi pin MK dengan LED
;-----
B6 BIT P0.0
B8 BIT P0.1
B5 BIT P0.2
B7 BIT P0.3
B2 BIT P0.6
B4 BIT P0.7

K5 BIT P2.0
K2 BIT P2.1
K1 BIT P2.2
K3 BIT P2.3
K4 BIT P2.4
B3 BIT p2.6
B1 BIT P2.7
;-----
ORG 00H
JMP MULAI_X
ORG 03H
JMP TERIMA_DATA
=====

```

```

                                                    terima_x1
;          PROGRAM UTAMA          :
;=====;
MULAI_X:
;
MULAI:
    SETB    EA
    SETB    EX0
    SETB    00H
    SETB    01H
    SETB    02H
    SETB    03H
    SETB    04H
    SETB    05H
    SETB    06H
    MOV     P0,#0FFH
    MOV     P1,#0FFH
    MOV     P3,#0FFH
    CALL    CEK_EEPROM
    MOV     ALAMAT,#0
    CALL    BACA_EEPROM
    MOV     A,DATA_MASUK
    CJNE    A,#1,CEK_ANIMASI
    MOV     ALAMAT,#1          ;EEPROM JML LAYAR
    CALL    BACA_EEPROM
    MOV     B,DATA_MASUK
    JMP     ANIMASI_1
CEK_ANIMASI:
    JMP     ANIMASI_2
ANIMASI_1:
    MOV     MEM_LAYAR,B
    MOV     R0,#CHAR_RAM
    MOV     ALAMAT,#2
    MOV     MEM_LOOP,#16
    CALL    BACA_CHAR
    MOV     MEM_LOOP,#200
    CALL    LAYAR_1
    JNB    03H,MULAI
    DJNZ   MEM_LAYAR,ANIM1_LAYAR2
    JMP     ANIMASI_1
ANIM1_LAYAR2:
    MOV     R0,#CHAR_RAM
    MOV     ALAMAT,#18
    MOV     MEM_LOOP,#16
    CALL    BACA_CHAR
    MOV     MEM_LOOP,#200
    CALL    LAYAR_1
    JNB    03H,MULAI
    DJNZ   MEM_LAYAR,ANIM1_LAYAR3
    JMP     ANIMASI_1
ANIM1_LAYAR3:
    MOV     R0,#CHAR_RAM
    MOV     ALAMAT,#34
    MOV     MEM_LOOP,#16
    CALL    BACA_CHAR
    MOV     MEM_LOOP,#200
    CALL    LAYAR_1
    JNB    03H,MULAI_
    JMP     ANIMASI_1
MULAI_:
    JMP     MULAI
;=====;
ANIMASI_2:
    MOV     ALAMAT,#1          ;ALAMAT = 1, JML_LAYAR
    CALL    BACA_EEPROM
CEK_ANIM2_LAYAR1:
    CJNE    A,#1,CEK_ANIM2_LAYAR2
    MOV     R0,#CHAR_RAM
    MOV     ALAMAT,#2          ;ALAMAT =2, CHAR_RAM

```

Table 1

Continued

Year	Number of cases	Rate per 100,000
1970	1,200	1.2
1971	1,300	1.3
1972	1,400	1.4
1973	1,500	1.5
1974	1,600	1.6
1975	1,700	1.7
1976	1,800	1.8
1977	1,900	1.9
1978	2,000	2.0
1979	2,100	2.1
1980	2,200	2.2
1981	2,300	2.3
1982	2,400	2.4
1983	2,500	2.5
1984	2,600	2.6
1985	2,700	2.7
1986	2,800	2.8
1987	2,900	2.9
1988	3,000	3.0
1989	3,100	3.1
1990	3,200	3.2
1991	3,300	3.3
1992	3,400	3.4
1993	3,500	3.5
1994	3,600	3.6
1995	3,700	3.7
1996	3,800	3.8
1997	3,900	3.9
1998	4,000	4.0
1999	4,100	4.1
2000	4,200	4.2
2001	4,300	4.3
2002	4,400	4.4
2003	4,500	4.5
2004	4,600	4.6
2005	4,700	4.7
2006	4,800	4.8
2007	4,900	4.9
2008	5,000	5.0
2009	5,100	5.1
2010	5,200	5.2
2011	5,300	5.3
2012	5,400	5.4
2013	5,500	5.5
2014	5,600	5.6
2015	5,700	5.7
2016	5,800	5.8
2017	5,900	5.9
2018	6,000	6.0
2019	6,100	6.1
2020	6,200	6.2
2021	6,300	6.3
2022	6,400	6.4
2023	6,500	6.5
2024	6,600	6.6
2025	6,700	6.7
2026	6,800	6.8
2027	6,900	6.9
2028	7,000	7.0
2029	7,100	7.1
2030	7,200	7.2

```

                                                    terima_X1
MOV     MEM_LOOP,#16
CALL   BACA_CHAR
TAMPIL_ANIM2_LAYAR1:
MOV     MEM_LOOP,#25
CALL   LAYAR_1
JNB    03H,MULAIX
CALL   ANIM2_LAYAR1
JMP    TAMPIL_ANIM2_LAYAR1
CEK_ANIM2_LAYAR2:
CJNE  A,#2,CEK_ANIM2_LAYAR3
MOV    R0,#CHAR_RAM
MOV    ALAMAT,#2
MOV    MEM_LOOP,#32
CALL   BACA_CHAR
TAMPIL_ANIM2_LAYAR2:
MOV    MEM_LOOP,#25
CALL   LAYAR_1
JNB    03H,MULAIX
CALL   ANIM2_LAYAR2
JMP    TAMPIL_ANIM2_LAYAR2
CEK_ANIM2_LAYAR3:
MOV    R0,#CHAR_RAM
MOV    ALAMAT,#2
MOV    MEM_LOOP,#48
CALL   BACA_CHAR
TAMPIL_ANIM2_LAYAR3:
MOV    MEM_LOOP,#25
CALL   LAYAR_1
JNB    03H,MULAIX
CALL   ANIM2_LAYAR3
JMP    TAMPIL_ANIM2_LAYAR3
MULAIX:
JMP    MULAI
;=====:
ANIM2_LAYAR1:
MOV    R0,#CHAR_RAM
MOV    CHAR_17,CHAR_RAM
MOV    R1,#3FH
MOV    MEM_LOOP,#16
LOOP_ANIM2_LAYAR1:
MOV    A,@R1
MOV    @R0,A
INC    R0
INC    R1
DJNZ  MEM_LOOP,LOOP_ANIM2_LAYAR1
RET
ANIM2_LAYAR2:
MOV    R0,#CHAR_RAM
MOV    CHAR_33,CHAR_RAM
MOV    R1,#3FH
MOV    MEM_LOOP,#32
JMP    LOOP_ANIM2_LAYAR1
ANIM2_LAYAR3:
MOV    R0,#CHAR_RAM
MOV    CHAR_49,CHAR_RAM
MOV    R1,#3FH
MOV    MEM_LOOP,#48
JMP    LOOP_ANIM2_LAYAR1
;=====:
BACA_CHAR:
CALL   BACA_EEPROM
MOV    @R0,A
INC    R0
INC    ALAMAT
DJNZ  MEM_LOOP,BACA_CHAR
RET
;=====:
DATA_EEPROM_TETAP:

```

terima\_x1

```

RET
CEK_EEPROM:
MOV     ALAMAT,#00H
CALL   BACA_EEPROM
CJNE   A,#0FFH,DATA_EEPROM_TETAP
MOV     ALAMAT,#00H
MOV     DATA_KELUAR,#2           ;JNS ANIMASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#1         ;JML LAYAR
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'D'      ;SPASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'A'      ;SPASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'T'      ;D
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'A'      ;A
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;T
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'K'      ;A
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'O'      ;SPASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'S'      ;K
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'O'      ;O
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'N'      ;S
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#'G'      ;O
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;N
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;G
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;SPASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;SPASI
CALL   TULIS_EEPROM
INC     ALAMAT
MOV     DATA_KELUAR,#32       ;SPASI
CALL   TULIS_EEPROM
RET

```

```

;=====
; SUBROUTINE DATA DITERIMA
;=====

```

```

BR_SERIAL:
mov     TMOD,#01H
mov     TH0,#0FFH
mov     TLO,#9CH
clr     TF0

```



```

setb    TR0
jnb     TF0,$
clr     TR0
ret

=====
TERIMA_DATA:
CLR     EX0
CLR     A
MOV     MEM_TMP2,#8
SETB    0H
SETB    01H
SETB    02H
START_BIT:
SETB    03H
SETB    04H
JNB     P3.2,$
WAIT_AWAL:
mov     TMOD,#01H
mov     TH0,#0FFH
mov     TLO,#0B0H
clr     TF0
setb    TR0
LOOP_WAIT_AWAL:
jnb     TF0,WAIT_AWAL_START
clr     TR0
JNB     04H,LOGIKA_0
CLR     04H
JMP     WAIT_AWAL
WAIT_AWAL_START:
JB      P3.2,LOOP_WAIT_AWAL
CLR     TR0
JNB     04H,LOGIKA_1
JMP     START_BIT
LOGIKA_1:
SETB    C
RRC     A
DJNZ    MEM_TMP2,START_BIT
JMP     CEK_BIT_AKHIR
LOGIKA_0:
CLR     C
RRC     A
CALL    STOP_BIT
JNB     02H,DATA_NOISE
DJNZ    MEM_TMP2,START_BIT
JMP     CEK_BIT_AKHIR
CEK_BIT_AKHIR:
MOV     MEM_TMP2,#8
call    BIT_AKHIR
JNB     02H,DATA_SLS
JB      0H,CEK_KODE_AWAL
BIT_STOP:
MOV     @R0,A
INC     R0
JMP     START_BIT
CEK_KODE_AWAL:
CLR     0H
CJNE    A,#9,BKN_DATA_REMOTE
MOV     R0,#JNS_ANIMASI
JMP     START_BIT
BKN_DATA_REMOTE:
CALL    DELAY_1S
JMP     DATA_NOISE
DATA_NOISE:
CALL    DELAY_1S
CLR     03H
SETB    EX0
RETI
=====

```

terima\_X1

ASCII\_DATA\_45:  
DB 251,251,251,253,251,247,251,239,255,254

ASCII\_DATA\_46:  
DB 255,251,245,253,245,247,255,239,255,254

ASCII\_DATA\_47:  
DB 247,251,254,253,251,247,127,239,255,190

ASCII\_DATA\_48:  
DB 114,187,188,253,185,247,61,239,114,190

ASCII\_DATA\_49:  
DB 255,251,253,189,48,183,253,239,255,254

ASCII\_DATA\_50:  
DB 245,187,188,253,185,247,125,175,255,254

ASCII\_DATA\_51:  
DB 189,251,185,253,185,247,118,175,255,254

ASCII\_DATA\_52:  
DB 251,251,123,253,48,183,251,239,255,254

ASCII\_DATA\_53:  
DB 57,187,61,253,61,247,178,239,255,254

ASCII\_DATA\_54:  
DB 114,187,185,253,185,247,246,175,255,254

ASCII\_DATA\_55:  
DB 180,251,187,253,63,247,191,175,255,254

ASCII\_DATA\_56:  
DB 118,187,185,253,185,247,118,175,255,254

ASCII\_DATA\_57:  
DB 119,187,185,253,185,247,114,175,255,254

ASCII\_DATA\_58:  
DB 255,251,182,189,182,183,255,239,255,254

ASCII\_DATA\_59:  
DB 255,251,188,189,182,183,255,239,255,254

ASCII\_DATA\_60:  
DB 251,251,126,253,247,183,189,239,255,254

ASCII\_DATA\_61:  
DB 126,251,126,253,126,247,126,239,126,254

ASCII\_DATA\_62:  
DB 189,251,247,189,126,247,251,239,255,254

ASCII\_DATA\_63:  
DB 255,187,191,253,188,247,187,239,127,190

ASCII\_DATA\_64:  
DB 246,187,185,253,176,247,189,239,114,190

ASCII\_DATA\_65:  
DB 112,187,190,253,190,247,190,239,112,190

ASCII\_DATA\_66:  
DB 48,187,185,253,185,247,185,239,118,190

ASCII\_DATA\_67:  
DB 114,187,189,253,189,247,189,239,247,190

ASCII\_DATA\_68:  
DB 48,187,189,253,189,247,247,175,122,254

ASCII\_DATA\_69:  
DB 48,187,185,253,185,247,185,239,189,254

ASCII\_DATA\_70:  
DB 48,187,187,253,187,247,187,239,191,254

ASCII\_DATA\_71:  
DB 114,187,189,253,185,247,185,239,240,190

ASCII\_DATA\_72:  
DB 48,187,251,253,251,247,251,239,48,190

ASCII\_DATA\_73:  
DB 255,251,189,253,48,183,189,239,255,254

ASCII\_DATA\_74:  
DB 247,251,253,253,189,247,50,175,191,254

ASCII\_DATA\_75:  
DB 48,187,251,253,126,247,247,175,189,254

ASCII\_DATA\_76:  
DB 48,187,253,253,253,247,253,239,253,254

ASCII\_DATA\_77:  
DB 48,187,255,189,123,247,255,175,48,190

ASCII\_DATA\_78:  
DB 48,187,127,253,251,247,254,239,48,190

The first part of the document discusses the importance of maintaining accurate records of all transactions. It emphasizes that every entry should be clearly documented, including the date, amount, and purpose of the transaction. This ensures transparency and allows for easy reconciliation of accounts.

In the second section, the author details the various methods used to collect and analyze data. This includes direct observation, interviews with key personnel, and the use of specialized software tools. The goal is to gather comprehensive information that can be used to identify trends and areas for improvement.

The third section focuses on the implementation of new procedures. It outlines the steps taken to train staff, update policies, and integrate new technology into the existing workflow. The author notes that while there were some initial challenges, the overall process went smoothly, leading to increased efficiency and productivity.

Finally, the document concludes with a summary of the findings and recommendations. It highlights the successful outcomes of the project and offers suggestions for future work, such as continuing to monitor performance and staying up-to-date with industry developments.

The following table provides a detailed breakdown of the data collected during the study. It shows the distribution of responses across different categories and over time, allowing for a more granular analysis of the results.

Category	Q1	Q2	Q3	Q4
Response A	15%	20%	18%	22%
Response B	30%	25%	28%	27%
Response C	45%	40%	42%	41%
Response D	10%	15%	12%	14%

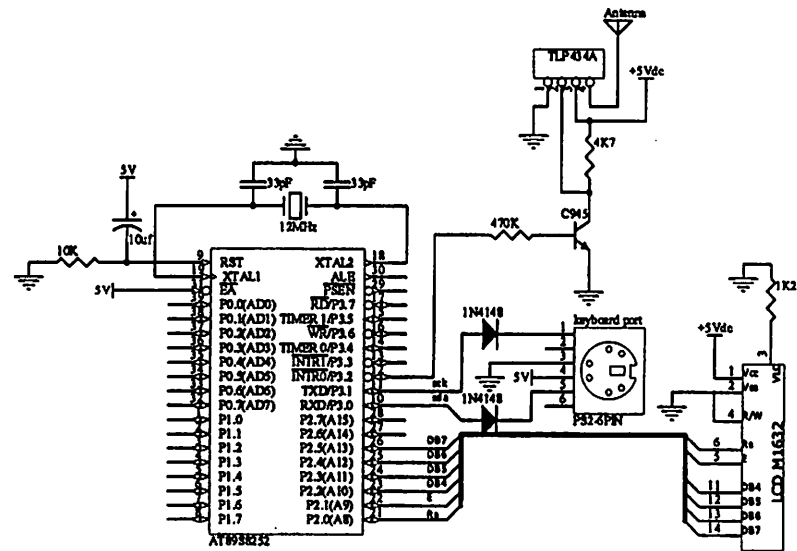
The data indicates that Response C is the most prevalent across all quarters, while Response D is the least. There are noticeable shifts in the proportions of other responses over time, suggesting external factors may be influencing the results.

terima\_x1

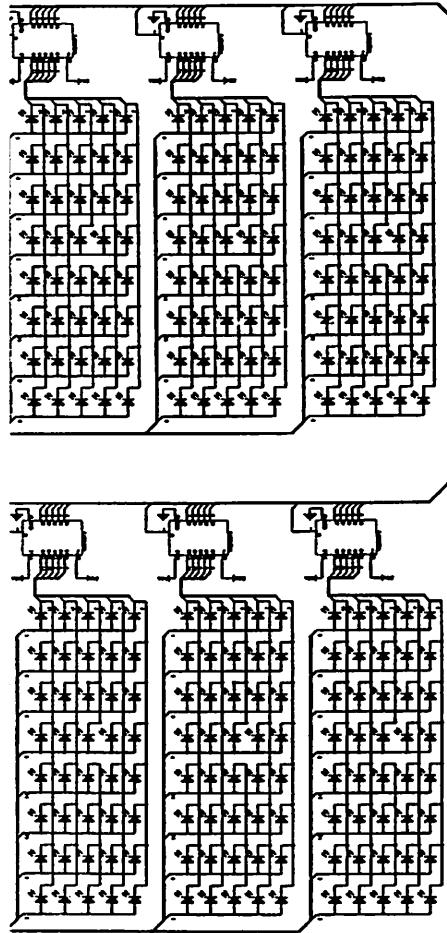
ASCII\_DATA\_79: DB 114,187,189,253,189,247,189,239,114,190  
 ASCII\_DATA\_80: DB 48,187,187,253,187,247,187,239,127,190  
 ASCII\_DATA\_81: DB 114,187,189,253,188,247,183,239,120,190  
 ASCII\_DATA\_82: DB 48,187,187,253,186,247,179,239,125,190  
 ASCII\_DATA\_83: DB 125,187,185,253,185,247,185,239,182,254  
 ASCII\_DATA\_84: DB 191,251,191,253,48,183,191,239,191,254  
 ASCII\_DATA\_85: DB 50,187,253,253,253,247,253,239,50,190  
 ASCII\_DATA\_86: DB 58,187,247,253,253,247,247,239,58,190  
 ASCII\_DATA\_87: DB 50,187,253,253,242,247,253,239,50,190  
 ASCII\_DATA\_88: DB 181,187,126,253,251,247,126,239,181,190  
 ASCII\_DATA\_89: DB 63,187,251,253,244,247,251,239,63,190  
 ASCII\_DATA\_90: DB 181,251,188,253,185,247,61,239,189,190  
 ASCII\_DATA\_91: DB 48,187,189,253,189,247,255,239,255,254  
 ASCII\_DATA\_92: DB 62,251,126,189,112,247,126,175,62,254  
 ASCII\_DATA\_93: DB 189,251,189,253,48,183,255,239,255,254  
 ASCII\_DATA\_94: DB 127,251,255,189,191,247,255,175,127,254  
 ASCII\_DATA\_95: DB 253,251,253,253,253,247,253,239,253,254  
 ASCII\_DATA\_96: DB 255,251,255,253,255,119,191,239,255,190  
 ASCII\_DATA\_97: DB 247,251,124,253,124,247,124,239,240,254  
 ASCII\_DATA\_98: DB 48,187,249,253,125,247,125,239,242,254  
 ASCII\_DATA\_99: DB 242,251,125,253,125,247,125,239,247,254  
 ASCII\_DATA\_100: DB 242,251,125,253,125,247,249,239,48,190  
 ASCII\_DATA\_101: DB 242,251,124,253,124,247,124,239,250,254  
 ASCII\_DATA\_102: DB 251,251,112,189,187,247,191,239,255,190  
 ASCII\_DATA\_103: DB 123,251,252,189,252,183,252,175,114,190  
 ASCII\_DATA\_104: DB 48,187,251,253,127,247,127,239,240,254  
 ASCII\_DATA\_105: DB 255,251,125,253,48,247,253,239,255,254  
 ASCII\_DATA\_106: DB 247,251,253,253,125,247,50,239,255,254  
 ASCII\_DATA\_107: DB 48,187,254,253,243,247,125,239,255,190  
 ASCII\_DATA\_108: DB 255,251,189,253,48,183,253,239,255,254  
 ASCII\_DATA\_109: DB 112,251,127,253,250,247,127,239,240,254  
 ASCII\_DATA\_110: DB 112,251,251,253,127,247,127,239,240,254  
 ASCII\_DATA\_111: DB 242,251,125,253,125,247,125,239,242,254  
 ASCII\_DATA\_112: DB 112,251,126,253,126,247,126,239,251,254

terima\_x1

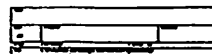
```
ASCII_DATA_113:
  DB 251,251,126,253,126,247,250,239,112,254
ASCII_DATA_114:
  DB 112,251,251,253,127,247,127,239,251,254
ASCII_DATA_115:
  DB 249,251,124,253,124,247,124,239,247,254
ASCII_DATA_116:
  DB 127,251,50,189,125,247,253,239,247,254
ASCII_DATA_117:
  DB 114,251,253,253,253,247,247,239,112,254
ASCII_DATA_118:
  DB 122,251,247,253,253,247,247,239,122,254
ASCII_DATA_119:
  DB 114,251,253,253,247,247,253,239,114,254
ASCII_DATA_120:
  DB 125,251,243,253,254,247,243,239,125,254
ASCII_DATA_121:
  DB 121,251,252,253,252,247,114,239,255,254
ASCII_DATA_122:
  DB 125,251,117,253,124,247,121,239,125,254
ASCII_DATA_123:
  DB 251,251,118,189,189,247,255,239,255,254
ASCII_DATA_124:
  DB 255,251,255,253,48,183,255,239,255,254
ASCII_DATA_125:
  DB 189,251,118,189,251,247,255,239,255,254
ASCII_DATA_126:
  DB 251,251,251,253,243,183,122,239,251,254
ASCII_DATA_127:
  DB 251,251,122,253,243,183,251,239,251,254
END
```



## SISTEM REMOTE



MATRIK



## Features

Compatible with MCS-51™ Products  
8K Bytes of In-System Reprogrammable Downloadable Flash Memory  
– SPI Serial Interface for Program Downloading  
– Endurance: 1,000 Write/Erase Cycles  
2K Bytes EEPROM  
– Endurance: 100,000 Write/Erase Cycles  
4V to 6V Operating Range  
Fully Static Operation: 0 Hz to 24 MHz  
Three-level Program Memory Lock  
256 x 8-bit Internal RAM  
32 Programmable I/O Lines  
Three 16-bit Timer/Counters  
Five Interrupt Sources  
Programmable UART Serial Channel  
SPI Serial Interface  
Low-power Idle and Power-down Modes  
Interrupt Recovery From Power-down  
Programmable Watchdog Timer  
Dual Data Pointer  
Power-off Flag

## Description

The AT89S8252 is a low-power, high-performance CMOS 8-bit microcomputer with 8K bytes of downloadable Flash programmable and erasable read only memory and 2K bytes of EEPROM. The device is manufactured using Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pinout. The on-chip downloadable Flash allows the program memory to be reprogrammed in-system through an SPI serial interface or by a conventional nonvolatile memory programmer. By combining a versatile 8-bit CPU with downloadable Flash on a monolithic chip, the Atmel AT89S8252 is a powerful microcomputer which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89S8252 provides the following standard features: 8K bytes of downloadable Flash, 2K bytes of EEPROM, 256 bytes of RAM, 32 I/O lines, programmable watchdog timer, two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S8252 is designed with static logic for operation down to zero frequency and supports two software selectable power saving modes. The Idle Mode suspends the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM contents but freezes the oscillator, disabling all other chip functions until the next interrupt or hardware reset.

The downloadable Flash can be changed a single byte at a time and is accessible through the SPI serial interface. Holding RESET active forces the SPI bus into a serial programming interface and allows the program memory to be written to or read from unless Lock Bit 2 has been activated.



## 8-bit Microcontroller with 8K Bytes Flash

### AT89S8252

Rev. 0401E-02/00





**SECRET**

SECRET

SECRET

SECRET

SECRET

SECRET

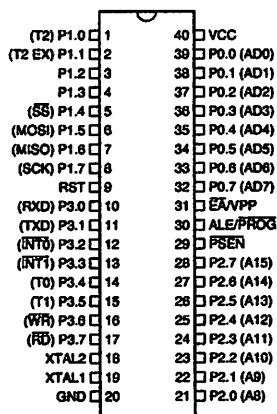
SECRET

SECRET

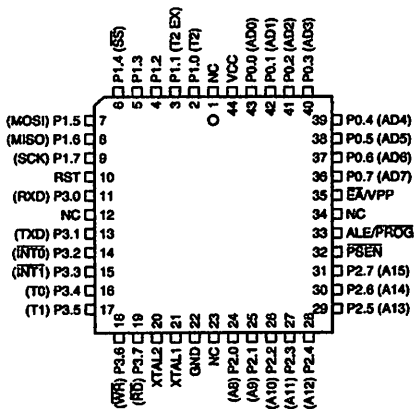
SECRET

## Pin Configurations

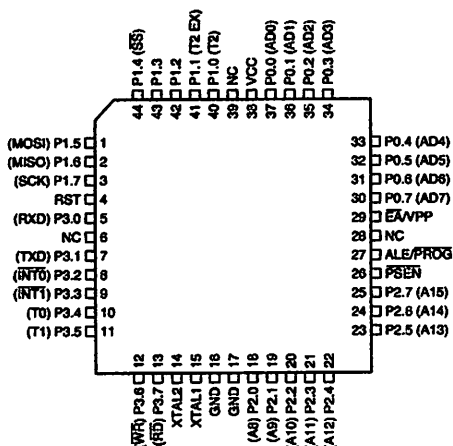
PDIP



PLCC



PQFP/TQFP



## Pin Description

**VCC**  
Supply voltage.

**EA/VPP**  
Program voltage.

**Port 0**  
Port 0 is an 8-bit open drain bi-directional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

Port 0 can also be configured to be the multiplexed lower-order address/data bus during accesses to external

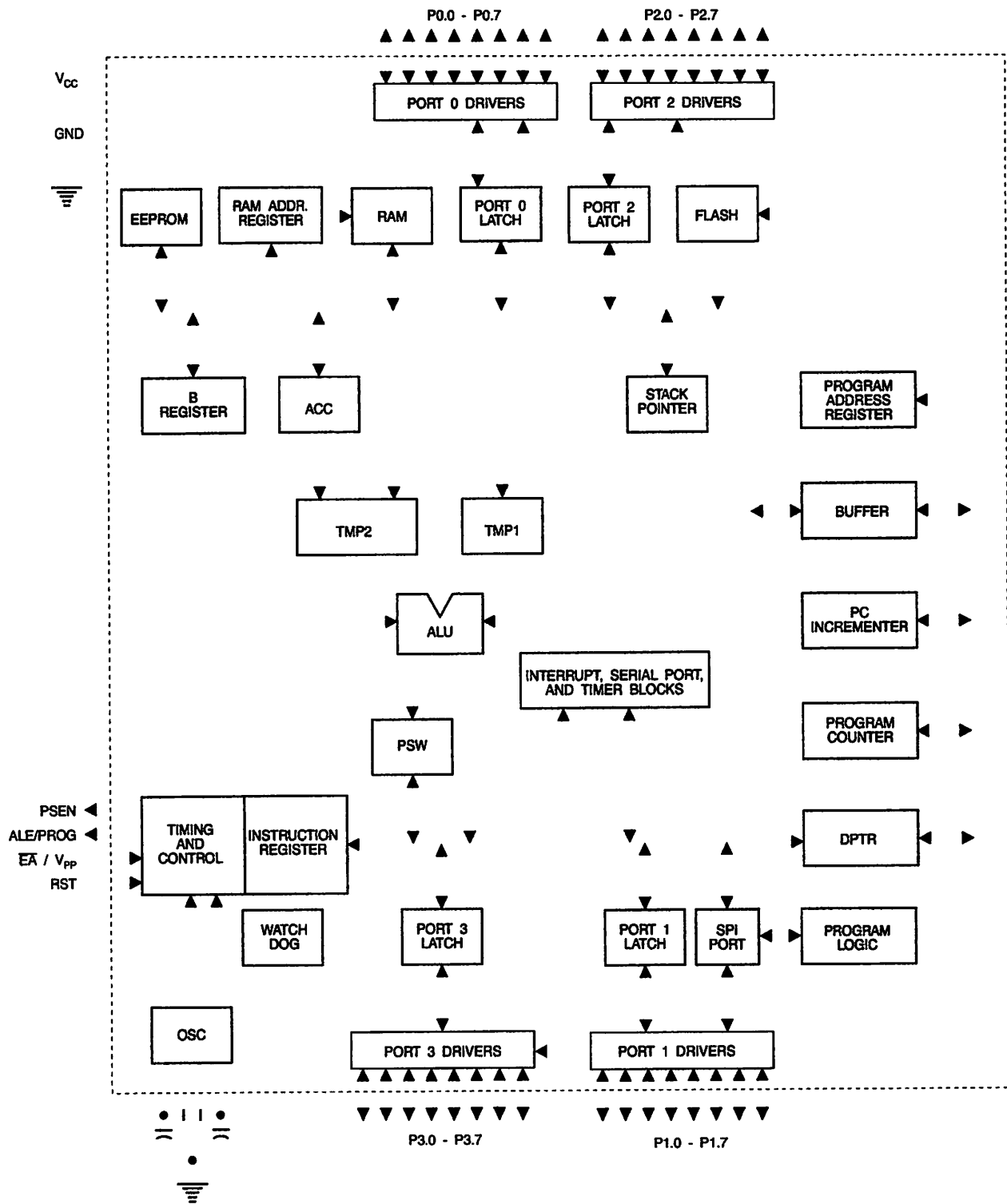
program and data memory. In this mode, P0 has internal pullups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. External pullups are required during program verification.

### Port 1

Port 1 is an 8-bit bi-directional I/O port with internal pullups. The Port 1 output buffers can sink/source four TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 1 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the internal pullups.

Block Diagram



Some Port 1 pins provide additional functions. P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively.

## In Description

Furthermore, P1.4, P1.5, P1.6, and P1.7 can be configured as the SPI slave port select, data input/output and shift clock input/output pins as shown in the following table.

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.4	$\overline{SS}$ (Slave port select input)
P1.5	MOSI (Master data output, slave data input pin for SPI channel)
P1.6	MISO (Master data input, slave data output pin for SPI channel)
P1.7	SCK (Master clock output, slave clock input pin for SPI channel)

Port 1 also receives the low-order address bytes during Flash programming and verification.

### Port 2

Port 2 is an 8-bit bi-directional I/O port with internal pullups. The Port 2 output buffers can sink/source four TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 2 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the internal pullups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ PTR). In this application, Port 2 uses strong internal pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 outputs the contents of the P2 Special Function Register.

Port 2 also receives the high-order address bits and some control signals during Flash programming and verification.

### Port 3

Port 3 is an 8 bit bi-directional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs,

Port 3 pins that are externally being pulled low will source current ( $I_{IL}$ ) because of the pullups.

Port 3 also serves the functions of various special features of the AT89S8252, as shown in the following table.

Port 3 also receives some control signals for Flash programming and verification.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

### RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device.

### ALE/ $\overline{PROG}$

Address Latch Enable is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input ( $\overline{PROG}$ ) during Flash programming.

In normal operation, ALE is emitted at a constant rate of 1/6 the oscillator frequency and may be used for external timing or clocking purposes. Note, however, that one ALE pulse is skipped during each access to external data memory.

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

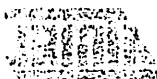
### $\overline{PSEN}$

Program Store Enable is the read strobe to external program memory.

When the AT89S8252 is executing code from external program memory,  $\overline{PSEN}$  is activated twice each machine cycle, except that two  $\overline{PSEN}$  activations are skipped during each access to external data memory.

### $\overline{EA}/VPP$

External Access Enable.  $\overline{EA}$  must be strapped to GND in order to enable the device to fetch code from external pro-



... ..  
... ..  
... ..

Alabama Functions	Part No.
Alabama Functions	100
Alabama Functions	101
Alabama Functions	102
Alabama Functions	103
Alabama Functions	104
Alabama Functions	105
Alabama Functions	106
Alabama Functions	107
Alabama Functions	108
Alabama Functions	109
Alabama Functions	110
Alabama Functions	111
Alabama Functions	112
Alabama Functions	113
Alabama Functions	114
Alabama Functions	115
Alabama Functions	116
Alabama Functions	117
Alabama Functions	118
Alabama Functions	119
Alabama Functions	120

... ..  
... ..

**ALABAMA**

... ..  
... ..

... ..  
... ..

... ..  
... ..

**ALABAMA**

... ..  
... ..

**ALABAMA**

... ..  
... ..

... ..  
... ..

**ALABAMA**

... ..  
... ..

Alabama Functions	Part No.
Alabama Functions	100
Alabama Functions	101
Alabama Functions	102
Alabama Functions	103
Alabama Functions	104
Alabama Functions	105
Alabama Functions	106
Alabama Functions	107
Alabama Functions	108
Alabama Functions	109
Alabama Functions	110
Alabama Functions	111
Alabama Functions	112
Alabama Functions	113
Alabama Functions	114
Alabama Functions	115
Alabama Functions	116
Alabama Functions	117
Alabama Functions	118
Alabama Functions	119
Alabama Functions	120

... ..  
... ..

**ALABAMA**

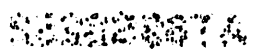
... ..  
... ..

... ..  
... ..

... ..  
... ..

**ALABAMA**

... ..  
... ..



memory locations starting at 0000H up to FFFFH. However, if lock bit 1 is programmed,  $\overline{EA}$  will be normally latched on reset.

$\overline{EA}$  should be strapped to  $V_{CC}$  for internal program executions. This pin also receives the 12-volt programming enable voltage ( $V_{PP}$ ) during Flash programming when 12-volt programming is selected.

### XTAL1

Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

### XTAL2

Output from the inverting oscillator amplifier.

Table 1. AT89S8252 SFR Map and Reset Values

0F8H								0FFH	
0F0H	B 00000000							0F7H	
0E8H								0EFH	
0E0H	ACC 00000000							0E7H	
0D8H								0DFH	
0D0H	PSW 00000000					SPCR 000001XX		0D7H	
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH	
0C0H								0C7H	
0B8H	IP XX000000							0BFH	
0B0H	P3 11111111							0B7H	
0A8H	IE 0X000000		SPSR 00XXXXXX					0AFH	
0A0H	P2 11111111							0A7H	
098H	SCON 00000000	SBUF XXXXXXXX						09FH	
090H	P1 11111111						WMCON 00000010	097H	
088H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000		08FH	
080H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR XXXXXXXX	PCON 0XX00000	087H



## Special Function Registers

map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 1.

Note that not all of the addresses are occupied, and unoccupied addresses may not be implemented on the chip. Read accesses to these addresses will in general return random data, and write accesses will have an indeterminate effect.

User software should not write 1s to these unlisted

locations, since they may be used in future products to invoke new features. In that case, the reset or inactive values of the new bits will always be 0.

**Timer 2 Registers** Control and status bits are contained in registers T2CON (shown in Table 2) and T2MOD (shown in Table 9) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture/Reload registers for Timer 2 in 16 bit capture mode or 16-bit auto-reload mode.

**Table 2. T2CON—Timer/Counter 2 Control Register**

T2CON Address = 0C8H				Reset Value = 0000 0000B			
Bit Addressable							
TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/T2	CP/RL2
7	6	5	4	3	2	1	0

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

**Watchdog and Memory Control Register** The WMCON register contains control bits for the Watchdog Timer (shown in Table 3). The EEMEN and EEMWE bits are used

to select the 2K bytes on-chip EEPROM, and to enable byte-write. The DPS bit selects one of two DPTR registers available.

**Table 3. WMCON—Watchdog and Memory Control Register**

WMCON Address = 96H				Reset Value = 0000 0010B			
PS2	PS1	PS0	EEMWE	EEMEN	DPS	WDTRST	WDTEN
7	6	5	4	3	2	1	0

Symbol	Function
PS2 PS1 PS0	Prescaler Bits for the Watchdog Timer. When all three bits are set to "0", the watchdog timer has a nominal period of 16 ms. When all three bits are set to "1", the nominal period is 2048 ms.
EEMWE	EEPROM Data Memory Write Enable Bit. Set this bit to "1" before initiating byte write to on-chip EEPROM with the MOVX instruction. User software should set this bit to "0" after EEPROM write is completed.
EEMEN	Internal EEPROM Access Enable. When EEMEN = 1, the MOVX instruction with DPTR will access on-chip EEPROM instead of external data memory. When EEMEN = 0, MOVX with DPTR accesses external data memory.
DPS	Data Pointer Register Select. DPS = 0 selects the first bank of Data Pointer Register, DP0, and DPS = 1 selects the second bank, DP1
WDTRST RDY/BSY	Watchdog Timer Reset and EEPROM Ready/Busy Flag. Each time this bit is set to "1" by user software, a pulse is generated to reset the watchdog timer. The WDTRST bit is then automatically reset to "0" in the next instruction cycle. The WDTRST bit is Write-Only. This bit also serves as the RDY/BSY flag in a Read-Only mode during EEPROM write. RDY/BSY = 1 means that the EEPROM is ready to be programmed. While programming operations are being executed, the RDY/BSY bit equals "0" and is automatically reset to "1" when programming is completed.
WDTEN	Watchdog Timer Enable Bit. WDTEN = 1 enables the watchdog timer and WDTEN = 0 disables the watchdog timer.

**SPI Registers** Control and status bits for the Serial Peripheral Interface are contained in registers SPCR (shown in Table 4) and SPSR (shown in Table 5). The SPI data bits are contained in the SPDR register. Writing the SPI data register during serial data transfer sets the Write Collision bit, WCOL, in the SPSR register. The SPDR is double buffered for writing and the values in SPDR are not changed by reset.

**Interrupt Registers** The global interrupt enable bit and the individual interrupt enable bits are in the IE register. In addition, the individual interrupt enable bit for the SPI is in the SPCR register. Two priorities can be set for each of the interrupt sources in the IP register.

**Dual Data Pointer Registers** To facilitate accessing both internal EEPROM and external data memory, two banks of 16 bit Data Pointer Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 in SFR WMCON selects DP0 and DPS = 1 selects DP1. The user should always initialize the DPS bit to the appropriate value before accessing the respective Data Pointer Register.

**Power Off Flag** The Power Off Flag (POF) is located at bit\_4 (PCON.4) in the PCON SFR. POF is set to "1" during power up. It can be set and reset under software control and is not affected by RESET.





TOP SECRET - FRODO BAGGINS

CONFIDENTIAL - GANDALF THE WHITE

SECRET - ARWEN UNBORN

CONFIDENTIAL - GALADRIEL

TOP SECRET - LEGOLAS

CONFIDENTIAL - SAMWISSE GAMBIT

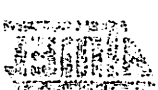
SECRET - GALADRIEL

CONFIDENTIAL - GALADRIEL

TOP SECRET - GALADRIEL

CONFIDENTIAL - GALADRIEL

TOP SECRET - GALADRIEL





#### Table 4. SPCR—SPI Control Register

SPCR Address = D5H

Reset Value = 0000 01XXB

Bit	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
	7	6	5	4	3	2	1	0

Symbol	Function
SPIE	SPI Interrupt Enable. This bit, in conjunction with the ES bit in the IE register, enables SPI interrupts: SPIE = 1 and ES = 1 enable SPI interrupts. SPIE = 0 disables SPI interrupts.
SPE	SPI Enable. SPI = 1 enables the SPI channel and connects $\overline{SS}$ , MOSI, MISO and SCK to pins P1.4, P1.5, P1.6, and P1.7. SPI = 0 disables the SPI channel.
DORD	Data Order. DORD = 1 selects LSB first data transmission. DORD = 0 selects MSB first data transmission.
MSTR	Master/Slave Select. MSTR = 1 selects Master SPI mode. MSTR = 0 selects Slave SPI mode.
CPOL	Clock Polarity. When CPOL = 1, SCK is high when idle. When CPOL = 0, SCK of the master device is low when not transmitting. Please refer to figure on SPI Clock Phase and Polarity Control.
CPHA	Clock Phase. The CPHA bit together with the CPOL bit controls the clock and data relationship between master and slave. Please refer to figure on SPI Clock Phase and Polarity Control.
SPR1 SPR0	SPI Clock Rate Select. These two bits control the SCK rate of the device configured as master. SPR1 and SPR0 have no effect on the slave. The relationship between SCK and the oscillator frequency, $F_{osc}$ , is as follows: SPR1SPR0 SCK = $F_{osc}$ divided by 0 0 4 0 1 16 1 0 64 1 1 128

#### Table 5. SPSR – SPI Status Register

SPSR Address = AAH

Reset Value = 00XX XXXXB

Bit	SPIF	WCOL	–	–	–	–	–	–
	7	6	5	4	3	2	1	0

Symbol	Function
SPIF	SPI Interrupt Flag. When a serial transfer is complete, the SPIF bit is set and an interrupt is generated if SPIE = 1 and ES = 1. The SPIF bit is cleared by reading the SPI status register with SPIF and WCOL bits set, and then accessing the SPI data register.
WCOL	Write Collision Flag. The WCOL bit is set if the SPI data register is written during a data transfer. During data transfer, the result of reading the SPDR register may be incorrect, and writing to it has no effect. The WCOL bit (and the SPIF bit) are cleared by reading the SPI status register with SPIF and WCOL set, and then accessing the SPI data register.

#### Table 6. SPDR – SPI Data Register

SPDR Address = 86H

Reset Value = unchanged

Bit	SPD7	SPD6	SPD5	SPD4	SPD3	SPD2	SPD1	SPD0
	7	6	5	4	3	2	1	0

**AT89S8252**

**Data Memory – EEPROM and RAM**

The AT89S8252 implements 2K bytes of on-chip EEPROM data storage and 256 bytes of RAM. The upper 128 bytes of RAM occupy a parallel space to the Special Function Registers. That means the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions that use direct addressing access SFR space.

For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV 0A0H, #data
```

Instructions that use indirect addressing access the upper 128 bytes of RAM. For example, the following indirect addressing instruction, where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

```
MOV @R0, #data
```

Note that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space.

The on-chip EEPROM data memory is selected by setting the EEMEN bit in the WMCON register at SFR address location 96H. The EEPROM address range is from 000H to 7FH. The MOVX instructions are used to access the EEPROM. To access off-chip data memory with the MOVX instructions, the EEMEN bit needs to be set to "0".

The EEMWE bit in the WMCON register needs to be set to 1 before any byte location in the EEPROM can be written. After software should reset EEMWE bit to "0" if no further EEPROM write is required. EEPROM write cycles in the normal programming mode are self-timed and typically take 5 ms. The progress of EEPROM write can be monitored by reading the RDY/BSY bit (read-only) in SFR WMCON. RDY/BSY = 0 means programming is still in progress and RDY/BSY = 1 means EEPROM write cycle is completed and another write cycle can be initiated.

In addition, during EEPROM programming, an attempted read from the EEPROM will fetch the byte being written with the MSB complemented. Once the write cycle is completed, true data are valid at all bit locations.

**Programmable Watchdog Timer**

The programmable Watchdog Timer (WDT) operates from an independent oscillator. The prescaler bits, PS0, PS1 and PS2 in SFR WMCON are used to set the period of the Watchdog Timer from 16 ms to 2048 ms. The available timer periods are shown in the following table and the

actual timer periods (at V<sub>CC</sub> = 5V) are within ±30% of the nominal.

The WDT is disabled by Power-on Reset and during Power-down. It is enabled by setting the WDTEN bit in SFR WMCON (address = 96H). The WDT is reset by setting the WDRST bit in WMCON. When the WDT times out without being reset or disabled, an internal RST pulse is generated to reset the CPU.

**Table 7. Watchdog Timer Period Selection**

WDT Prescaler Bits			Period (nominal)
PS2	PS1	PS0	
0	0	0	16 ms
0	0	1	32 ms
0	1	0	64 ms
0	1	1	128 ms
1	0	0	256 ms
1	0	1	512 ms
1	1	0	1024 ms
1	1	1	2048 ms

**Timer 0 and 1**

Timer 0 and Timer 1 in the AT89S8252 operate the same way as Timer 0 and Timer 1 in the AT89C51, AT89C52 and AT89C55. For further information, see the October 1995 Microcontroller Data Book, page 2-45, section titled, "Timer/Counters."

**Timer 2**

Timer 2 is a 16 bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/T2 in the SFR T2CON (shown in Table 2). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 8.

Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

In the Counter function, the register is incremented in response to a 1-to-0 transition at its corresponding external input pin, T2. In this function, the external input is sampled during S5P2 of every machine cycle. When the samples show a high in one cycle and a low in the next cycle, the count is incremented. The new count value appears in the register during S3P1 of the cycle following the one in which



transition was detected. Since two machine cycles (24 oscillator periods) are required to recognize a 1-to-0 transition, the maximum count rate is 1/24 of the oscillator frequency. To ensure that a given level is sampled at least once before it changes, the level should be held for at least one full machine cycle.

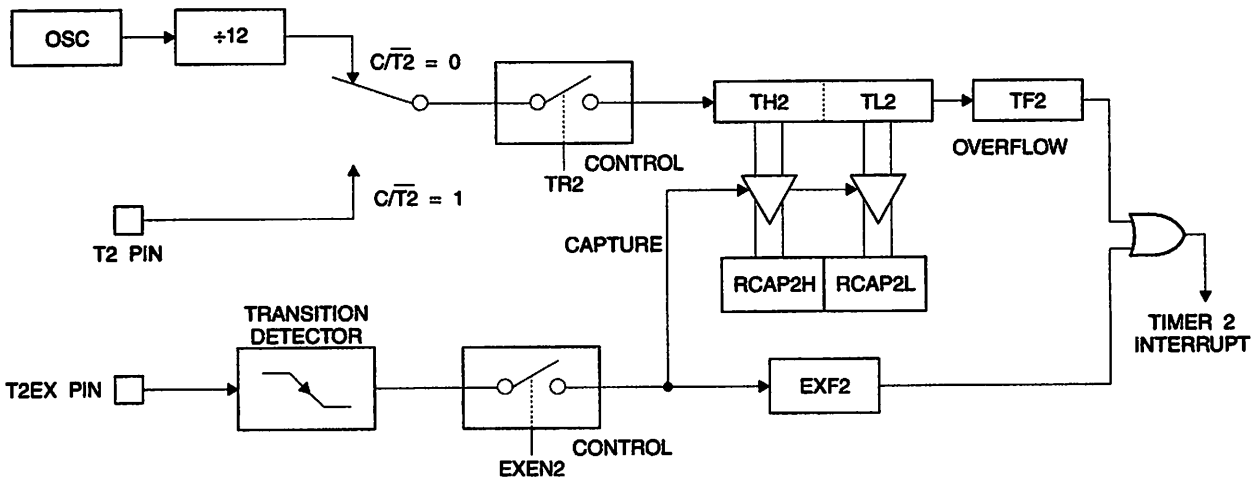
Table 8. Timer 2 Operating Modes

CLK + TCLK	CP/RL2	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud Rate Generator
X	X	0	(Off)

### Capture Mode

In the capture mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16 bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 performs the same operation, but a 1-to-0 transition at external input T2EX also causes the current value in TH2 and TL2 to be captured into RCAP2H and RCAP2L, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt. The capture mode is illustrated in Figure 1.

Figure 1. Timer 2 in Capture Mode



**Auto-reload (Up or Down Counter)**

Timer 2 can be programmed to count up or down when configured in its 16 bit auto-reload mode. This feature is controlled by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 9). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.

Figure 2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16 bit value in RCAP2H and RCAP2L. The values in RCAP2H and RCAP2L are preset by software. If EXEN2 = 1, a 16 bit reload can be triggered either by an overflow or

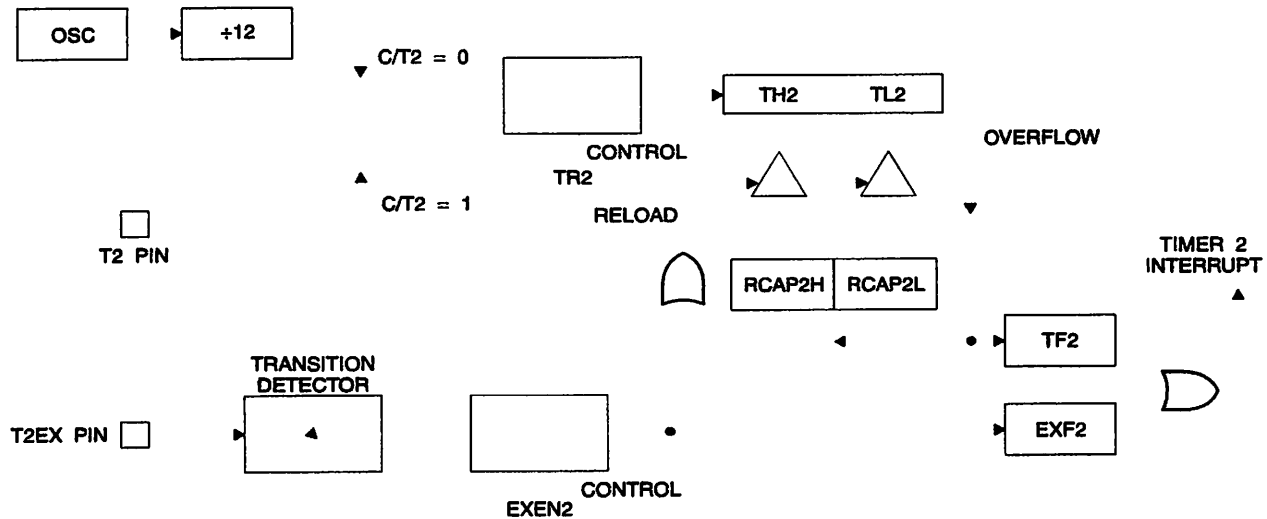
by a 1-to-0 transition at external input T2EX. This transition also sets the EXF2 bit. Both the TF2 and EXF2 bits can generate an interrupt if enabled.

Setting the DCEN bit enables Timer 2 to count up or down, as shown in Figure 3. In this mode, the T2EX pin controls the direction of the count. A logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit. This overflow also causes the 16 bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively.

A logic 0 at T2EX makes Timer 2 count down. The timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers.

The EXF2 bit toggles whenever Timer 2 overflows or underflows and can be used as a 17th bit of resolution. In this operating mode, EXF2 does not flag an interrupt.

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



**Table 9. T2MOD – Timer 2 Mode Control Register**

T2MOD Address = 0C9H							Reset Value = XXXX XX00B	
Not Bit Addressable								
Bit	7	6	5	4	3	2	1	0
	-	-	-	-	-	-	T2OE	DCEN

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



Figure 3. Timer 2 Auto Reload Mode (DCEN = 1)

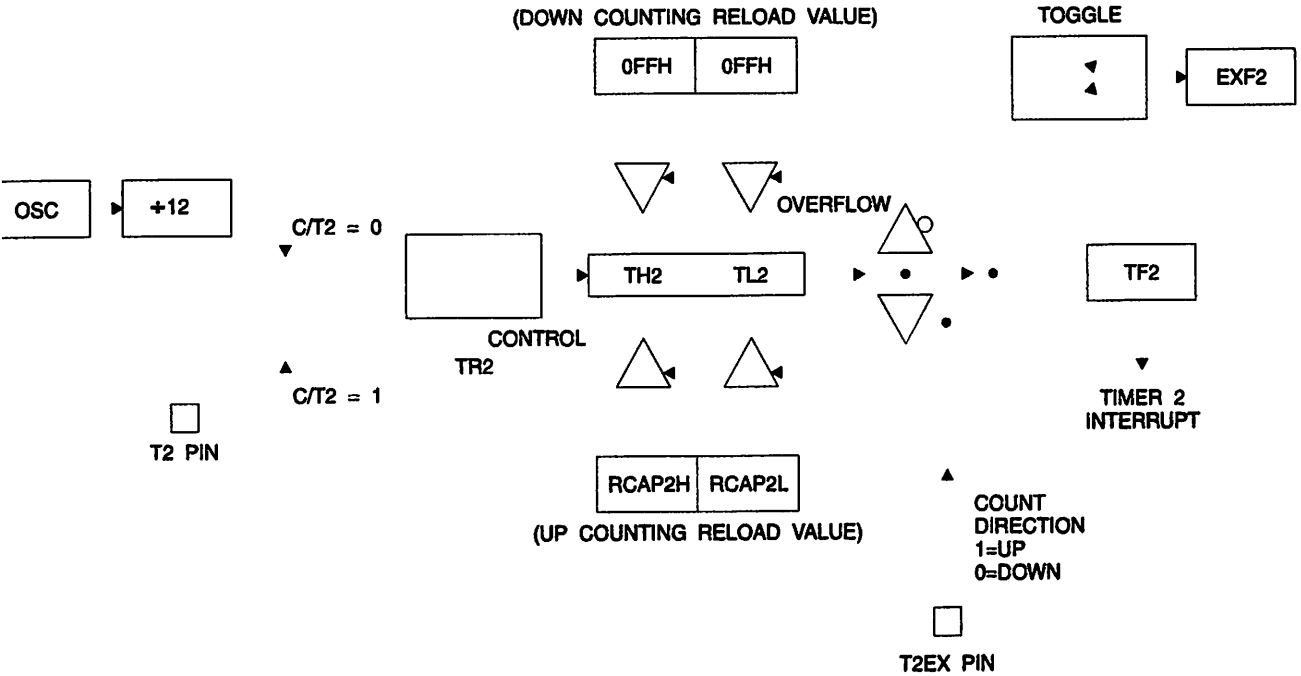
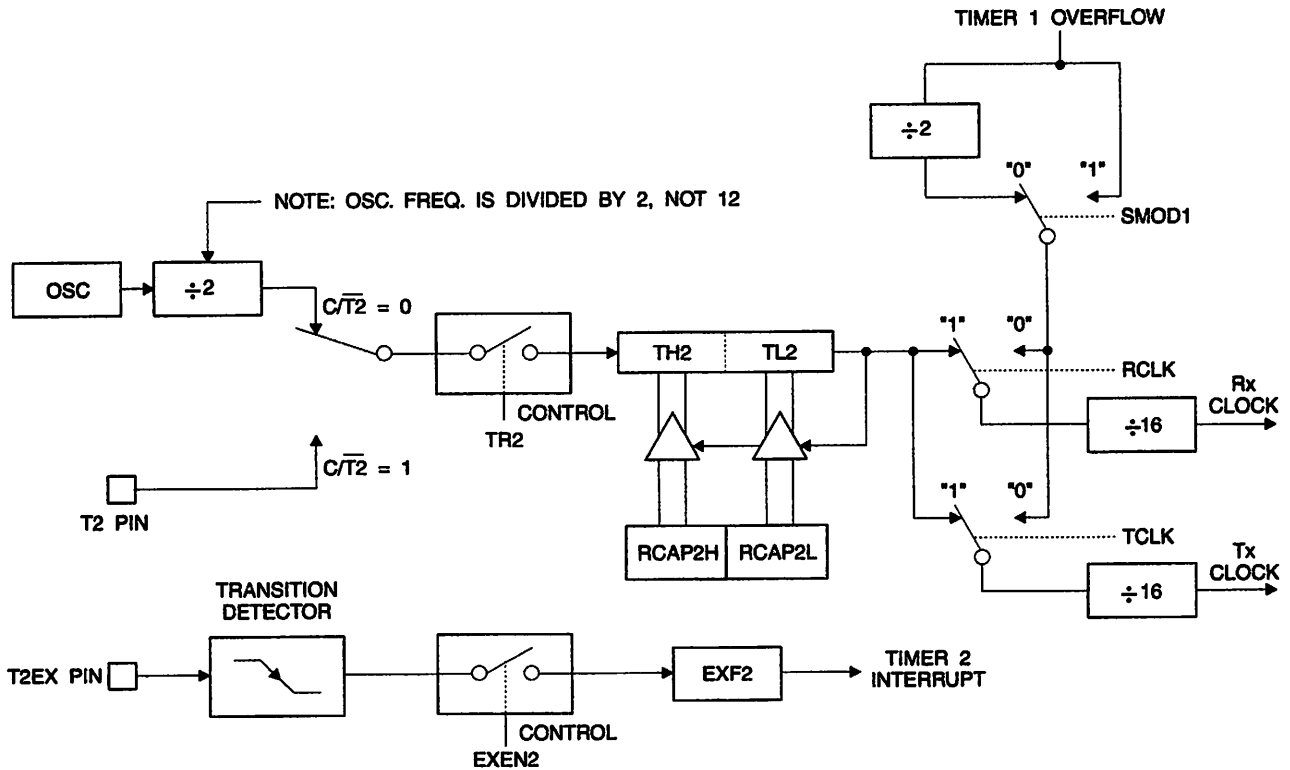


Figure 4. Timer 2 in Baud Rate Generator Mode



# LMV321 / LMV358 / LMV324 Single/Dual/Quad General Purpose, Low Voltage, Rail-to-Rail Output Operational Amplifiers

## General Description

The LMV358/324 are low voltage (2.7–5.5V) versions of the dual and quad commodity op amps, LM358/324, which currently operate at 5–30V. The LMV321 is the single version.

The LMV321/358/324 are the most cost effective solutions for the applications where low voltage operation, space saving and low price are needed. They offer specifications that meet or exceed the familiar LM358/324. The LMV321/358/324 have rail-to-rail output swing capability and the input common-mode voltage range includes ground. They all exhibit excellent speed-power ratio, achieving 1 MHz of bandwidth and 1 V/ $\mu$ s of slew rate with low supply current.

The LMV321 is available in space saving SC70-5, which is approximately half the size of SOT23-5. The small package saves space on pc boards, and enables the design of small portable electronic devices. It also allows the designer to place the device closer to the signal source to reduce noise pickup and increase signal integrity.

The chips are built with National's advanced submicron silicon-gate BiCMOS process. The LMV321/358/324 have bipolar input and output stages for improved noise performance and higher output current drive.

## Features

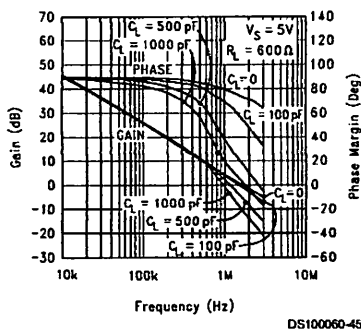
(For  $V^+ = 5V$  and  $V^- = 0V$ , Typical Unless Otherwise Noted)

- Guaranteed 2.7V and 5V Performance
- No Crossover Distortion
- Space Saving Package SC70-5 2.0x2.1x1.0mm
- Industrial Temp. Range  $-40^{\circ}C$  to  $+85^{\circ}C$
- Gain-Bandwidth Product 1MHz
- Low Supply Current
  - LMV321 130 $\mu$ A
  - LMV358 210 $\mu$ A
  - LMV324 410 $\mu$ A
- Rail-to-Rail Output Swing @ 10k $\Omega$  Load
  - $V^+ - 10mV$
  - $V^- + 65mV$
- $V_{CM}$   $-0.2V$  to  $V^+ - 0.8V$

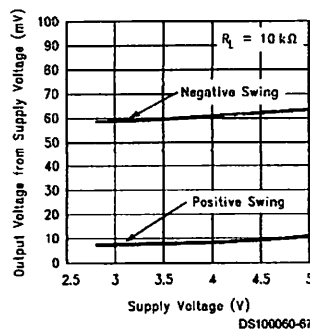
## Applications

- Active Filters
- General Purpose Low Voltage Applications
- General Purpose Portable Devices

**Gain and Phase vs Capacitive Load**



**Output Voltage Swing vs Supply Voltage**



**Absolute Maximum Ratings** (Note 1)

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

ESD Tolerance (Note 2)	
Machine Model	100V
Human Body Model	
LMV358/324	2000V
LMV321	900V
Differential Input Voltage	± Supply Voltage
Supply Voltage ( $V^+ - V^-$ )	5.5V
Output Short Circuit to $V^+$	(Note 3)
Output Short Circuit to $V^-$	(Note 4)
Soldering Information	
Infrared or Convection (20 sec)	235°C

Storage Temp. Range	-65°C to 150°C
Junction Temp. ( $T_j$ , max) (Note 5)	150°C

**Operating Ratings** (Note 1)

Supply Voltage	2.7V to 5.5V
Temperature Range	
LMV321, LMV358, LMV324	-40°C ≤ $T_j$ ≤ 85°C
Thermal Resistance ( $\theta_{JA}$ )(Note 10)	
5-pin SC70-5	478°C/W
5-pin SOT23-5	265°C/W
8-Pin SOIC	190°C/W
8-Pin MSOP	235°C/W
14-Pin SOIC	145°C/W
14-Pin TSSOP	155°C/W

**2.7V DC Electrical Characteristics**

Unless otherwise specified, all limits guaranteed for  $T_j = 25^\circ\text{C}$ ,  $V^+ = 2.7\text{V}$ ,  $V^- = 0\text{V}$ ,  $V_{CM} = 1.0\text{V}$ ,  $V_O = V^+/2$  and  $R_L > 1\text{M}\Omega$ .

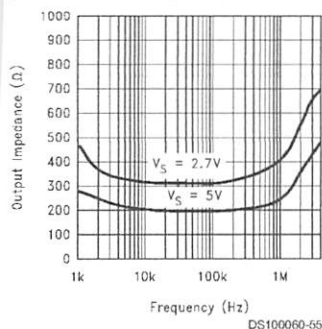
Symbol	Parameter	Conditions	Typ (Note 6)	Limit (Note 7)	Units
$V_{OS}$	Input Offset Voltage		1.7	7	mV max
$TCV_{OS}$	Input Offset Voltage Average Drift		5		$\mu\text{V}/^\circ\text{C}$
$I_B$	Input Bias Current		11	250	nA max
$I_{OS}$	Input Offset Current		5	50	nA max
CMRR	Common Mode Rejection Ratio	$0\text{V} \leq V_{CM} \leq 1.7\text{V}$	63	50	dB min
PSRR	Power Supply Rejection Ratio	$2.7\text{V} \leq V^+ \leq 5\text{V}$ $V_O = 1\text{V}$	60	50	dB min
$V_{CM}$	Input Common-Mode Voltage Range	For CMRR ≥ 50dB	-0.2	0	V min
			1.9	1.7	V max
$V_O$	Output Swing	$R_L = 10\text{k}\Omega$ to 1.35V	$V^+ - 10$	$V^+ - 100$	mV min
			60	180	mV max
$I_S$	Supply Current	LMV321	80	170	$\mu\text{A}$ max
		LMV358 Both amplifiers	140	340	$\mu\text{A}$ max
		LMV324 All four amplifiers	260	680	$\mu\text{A}$ max



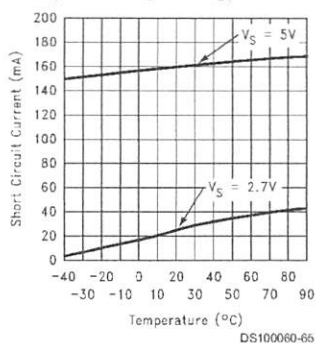
## Typical Performance Characteristics

Unless otherwise specified,  $V_S = +5V$ , single supply,  $T_A = 25^\circ C$ . (Continued)

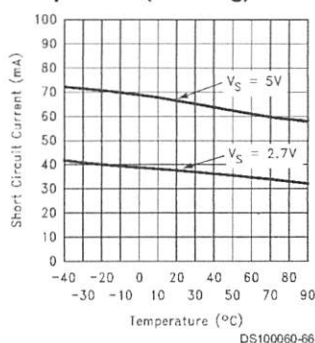
### Open Loop Output Impedance vs Frequency



### Short Circuit Current vs Temperature (Sinking)



### Short Circuit Current vs Temperature (Sourcing)



## Application Notes

### 10 Benefits of the LMV321/358/324

**Small Size.** The small footprints of the LMV321/358/324 packages save space on printed circuit boards, and enable the design of smaller electronic products, such as cellular phones, pagers, or other portable systems. The low profile of the LMV321/358/324 make them possible to use in PCMCIA type III cards.

**Signal Integrity.** Signals can pick up noise between the signal source and the amplifier. By using a physically smaller amplifier package, the LMV321/358/324 can be placed closer to the signal source, reducing noise pickup and increasing signal integrity.

**Simplified Board Layout.** These products help you to avoid long pc traces in your pc board layout. This means that additional components, such as capacitors and resistors, are not needed to filter out the unwanted signals due to the interference between the long pc traces.

**Supply Current.** These devices will help you to maximize battery life. They are ideal for battery powered systems.

**Supply Voltage.** National provides guaranteed performance at 2.7V and 5V. These guarantees ensure operation throughout the battery lifetime.

**Rail-to-Rail Output.** Rail-to-rail output swing provides maximum possible dynamic range at the output. This is particularly important when operating on low supply voltages.

**Includes Ground.** Allows direct sensing near GND in supply operation.

**Input Protection.** Differential input voltage may be larger than  $V^+$  without damaging the device. Protection should be provided to prevent the input voltages from going negative more than  $-0.3V$  ( $5^\circ C$ ). An input clamp diode with a resistor to the IC input terminal can be used.

**Use of Use & No Crossover Distortion.** The LMV321/358/324 offer specifications similar to the familiar LM324. In addition, the new LMV321/358/324 effectively eliminate the output crossover distortion. The scope photos in Figure 1 and Figure 2 compare the output swing of the LMV324 and the LM324 in a voltage follower configuration, with  $V_S = \pm 5V$  and  $R_L (= 2k\Omega)$  connected to GND. It is apparent that the crossover distortion has been eliminated in the new LMV324.

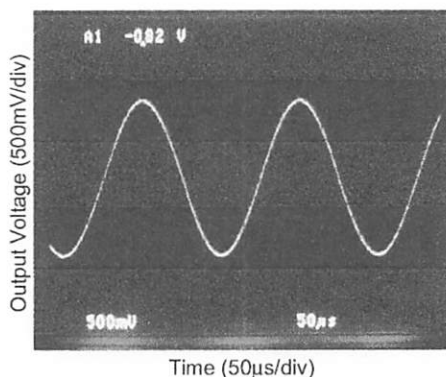


FIGURE 1. Output Swing of LMV324

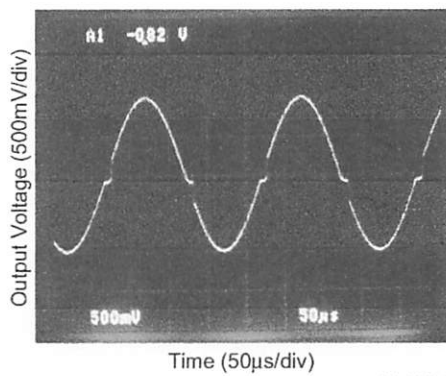
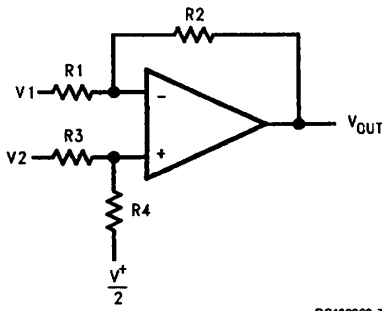


FIGURE 2. Output Swing of LM324

### 2.0 Capacitive Load Tolerance

The LMV321/358/324 can directly drive 200 pF in unity-gain without oscillation. The unity-gain follower is the most sensitive configuration to capacitive loading. Direct capacitive loading reduces the phase margin of amplifiers. The combination of the amplifier's output impedance and the capacitive load induces phase lag. This results in either an underdamped pulse response or oscillation. To drive a heavier capacitive load, circuit in Figure 3 can be used.

## Application Notes (Continued)



DS100060-7

$$V_{OUT} = \left( \frac{R1 + R2}{R3 + R4} \right) \frac{R4}{R1} V_2 - \frac{R2}{R1} V_1 + \left( \frac{R1 + R2}{R3 + R4} \right) \frac{R3}{R1} \cdot \frac{V^+}{2}$$

for  $R1 = R3$  and  $R2 = R4$

$$V_{OUT} = \frac{R2}{R1} (V_2 - V_1) + \frac{V^+}{2}$$

DS100060-19

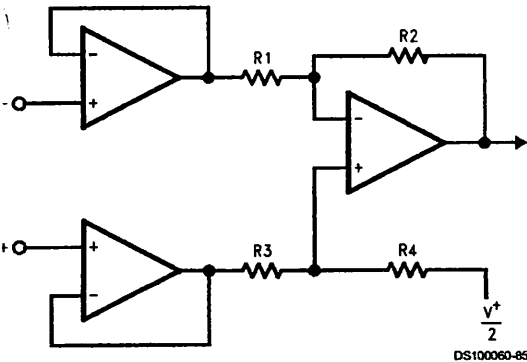
**FIGURE 7. Difference Amplifier**

### 2 Instrumentation Circuits

The input impedance of the previous difference amplifier is limited by the resistors  $R_1$ ,  $R_2$ ,  $R_3$ , and  $R_4$ . To eliminate the problems of low input impedance, one way is to use a voltage follower ahead of each input as shown in the following two instrumentation amplifiers.

#### 2.1 Three-op-amp Instrumentation Amplifier

The quad LMV324 can be used to build a three-op-amp instrumentation amplifier as shown in *Figure 8*.



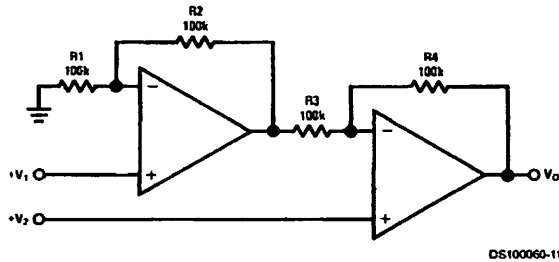
DS100060-85

**FIGURE 8. Three-op-amp Instrumentation Amplifier**

The first stage of this instrumentation amplifier is a differential-input, differential-output amplifier, with two voltage followers. These two voltage followers assure that the input impedance is over 100 MΩ. The gain of this instrumentation amplifier is set by the ratio of  $R_2/R_1$ .  $R_3$  should equal  $R_1$ , and  $R_4$  equal  $R_2$ . Matching of  $R_3$  to  $R_1$  and  $R_4$  to  $R_2$  affects the CMRR. For good CMRR over temperature, low drift resistors should be used. Making  $R_4$  slightly smaller than  $R_2$  and adding a trim pot equal to twice the difference between  $R_2$  and  $R_4$  will allow the CMRR to be adjusted for optimum.

#### 4.2.2 Two-op-amp Instrumentation Amplifier

A two-op-amp instrumentation amplifier can also be used to make a high-input-impedance dc differential amplifier (*Figure 9*). As in the three-op-amp circuit, this instrumentation amplifier requires precise resistor matching for good CMRR.  $R_4$  should equal to  $R_1$  and  $R_3$  should equal  $R_2$ .



DS100060-11

$$V_O = \left( 1 + \frac{R4}{R3} \right) (V_2 - V_1), \text{ where } R1 = R4 \text{ and } R2 = R3$$

As shown:  $V_O = 2(V_2 - V_1)$

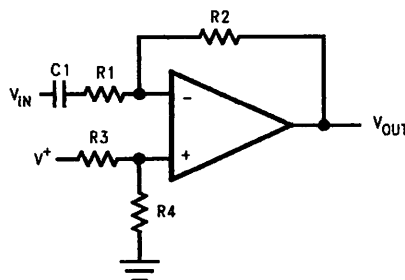
DS100060-36

**FIGURE 9. Two-Op-amp Instrumentation Amplifier**

#### 4.3 Single-Supply Inverting Amplifier

There may be cases where the input signal going into the amplifier is negative. Because the amplifier is operating in single supply voltage, a voltage divider using  $R_3$  and  $R_4$  is implemented to bias the amplifier so the input signal is within the input common-mode voltage range of the amplifier. The capacitor  $C_1$  is placed between the inverting input and resistor  $R_1$  to block the DC signal going into the AC signal source,  $V_{IN}$ . The values of  $R_1$  and  $C_1$  affect the cutoff frequency,  $f_c = 1/2\pi R_1 C_1$ .

As a result, the output signal is centered around mid-supply (if the voltage divider provides  $V^+/2$  at the non-inverting input). The output can swing to both rails, maximizing the signal-to-noise ratio in a low voltage system.



DS100060-13

$$V_{OUT} = -\frac{R2}{R1} V_{IN}$$

DS100060-20

**FIGURE 10. Single-Supply Inverting Amplifier**

#### 4.4 Active Filter

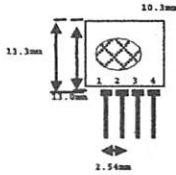
##### 4.4.1 Simple Low-Pass Active Filter

The simple low-pass filter is shown in *Figure 11*. Its low-frequency gain ( $\omega \rightarrow 0$ ) is defined by  $-R_2/R_1$ . This allows low-frequency gains other than unity to be obtained. The filter has a -20dB/decade roll-off after its corner frequency  $f_c$ .  $R_2$  should be chosen equal to the parallel combination of  $R_1$  and  $R_3$  to minimize errors due to bias current. The frequency response of the filter is shown in *Figure 12*.

# TLP434A & RLP434A RF ASK Hybrid Modules for Radio Control ( New Version )

## TLP434A Ultra Small Transmitter

**Easy-Link  
Wireless**

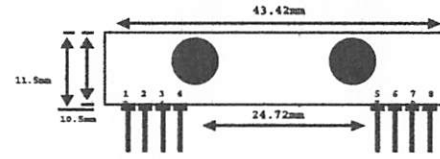


pin 1 : GND  
pin 2 : Data In  
pin 3 : Vcc  
pin 4 : Antenna ( RF output )

**Frequency 315, 418 and 433.92 Mhz**

Modulation : ASK  
Operation Voltage : 2 - 12 VDC

## RLP434A SAW Based Receiver



pin 1 : Gnd  
pin 2 : Digital Data Output  
pin 3 : RSSI  
pin 4 : Vcc  
pin 5 : Vcc  
pin 6 : E/D ( Enable = 1, Disable = 0 )  
pin 7 : Gnd  
pin 8 : Antenna

**Frequency 315, 418 and 433.92 Mhz**

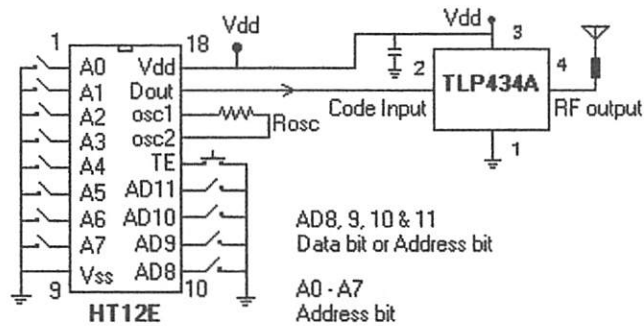
Modulation : ASK  
Supply Voltage : 3.3 - 6.0 VDC  
Output : Digital & Linear

Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Vcc	Operating supply voltage		2.0	-	12.0	V
Icc 1	Peak Current (2V)		-	-	1.64	mA
Icc 2	Peak Current (12V)		-	-	19.4	mA
Vh	Input High Voltage	Idata = 100uA (High)	Vcc-0.5	Vcc	Vcc+0.5	V
VI	Input Low Voltage	Idata = 0 uA (Low)	-	-	0.3	V
FO	Absolute Frequency	315Mhz module	314.8	315	315.2	MHz
PO	RF Output Power- 50ohm	Vcc = 9V-12V	-	16	-	dBm
		Vcc = 5V-6V	-	14	-	dBm
DR	Data Rate	External Encoding	512	4.8K	200K	bps

Notes : ( Case Temperature = 25°C +/- 2°C , Test Load Impedance = 50 ohm )

### Application Circuit :

Typical Key-chain Transmitter using HT12E-18DIP, a Binary 12 bit Encoder from Holtek Semiconductor Inc.



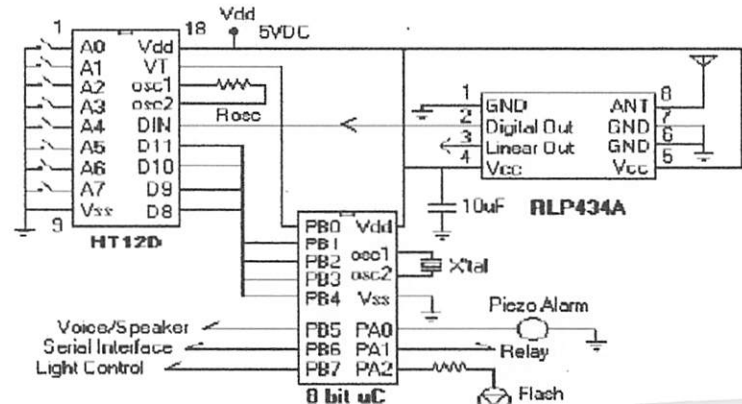
Symbol	Parameter	Conditions	Min	Typ	Max	Unit
Vcc	Operating supply voltage		3.3	5.0V	6.0	V
I <sub>tot</sub>	Operating Current		-	4.5		mA
V <sub>data</sub>	Data Out	I <sub>data</sub> = +200 uA ( High )	V <sub>cc</sub> -0.5	-	V <sub>cc</sub>	V
		I <sub>data</sub> = -10 uA ( Low )	-	-	0.3	V

### Electrical Characteristics

Characteristics	SYM	Min	Typ	Max	Unit
Operation Radio Frequency	FC		315, 418 and 433.92		MHz
Sensitivity	Pref		-110		dBm
Channel Width			+500		KHz
Noise Equivalent BW			4		KHz
Receiver Turn On Time			5		ms
Operation Temperature	Top	-20	-	80	C
Baseboard Data Rate			4.8		KHz

### Application Circuit :

Typical RF Receiver using HT12D-18DIP, a Binary 12 bit Decoder with 8 bit uC HT48RXX from Holtek Semiconductor Inc.





## FEATURES

Power dissipation

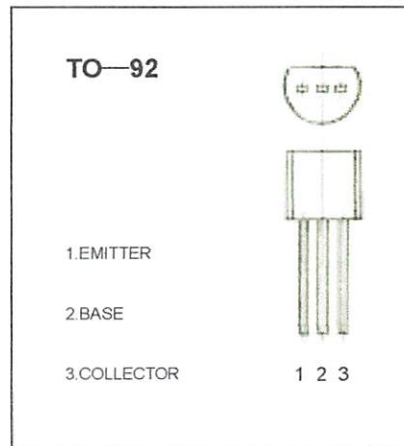
$$P_{CM} : 0.625 \text{ W (Tamb=25}^\circ\text{C)}$$

Collector current

$$I_{CM} : -0.5 \text{ A}$$

Collector-base voltage

$$V_{(BR)CBO} : -40 \text{ V}$$



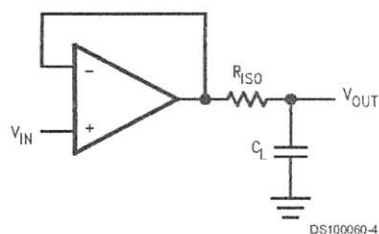
## ELECTRICAL CHARACTERISTICS (Tamb=25°C unless otherwise specified)

Parameter	Symbol	Test conditions	MIN	TYP	MAX	UNIT
Collector-base breakdown voltage	$V_{(BR)CBO}$	$I_C = -100 \mu\text{A}, I_E = 0$	-40			V
Collector-emitter breakdown voltage	$V_{(BR)CEO}$	$I_C = -0.1 \text{ mA}, I_B = 0$	-20			V
Emitter-base breakdown voltage	$V_{(BR)EBO}$	$I_E = -100 \mu\text{A}, I_C = 0$	-5			V
Collector cut-off current	$I_{CBO}$	$V_{CB} = -40 \text{ V}, I_E = 0$			-0.1	$\mu\text{A}$
Collector cut-off current	$I_{CEO}$	$V_{CE} = -20 \text{ V}, I_B = 0$			-0.2	$\mu\text{A}$
Emitter cut-off current	$I_{EBO}$	$V_{EB} = -5 \text{ V}, I_C = 0$			-0.1	$\mu\text{A}$
DC current gain(note)	$H_{FE(1)}$	$V_{CE} = -1 \text{ V}, I_C = -50 \text{ mA}$	64		300	
	$H_{FE(2)}$	$V_{CE} = -1 \text{ V}, I_C = -500 \text{ mA}$	40			
Collector-emitter saturation voltage	$V_{CE(sat)}$	$I_C = -500 \text{ mA}, I_B = -50 \text{ mA}$			-0.6	V
Base-emitter saturation voltage	$V_{BE(sat)}$	$I_C = -500 \text{ mA}, I_B = -50 \text{ mA}$			-1.2	V
Base-emitter voltage	$V_{EB}$	$I_E = -100 \text{ mA}$			-1.4	V
Transition frequency	$f_T$	$V_{CE} = -6 \text{ V}, I_C = -20 \text{ mA}$ $f = 30 \text{ MHz}$	150			MHz

## CLASSIFICATION OF $H_{FE(1)}$

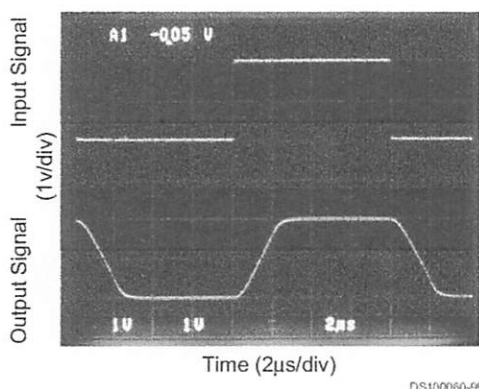
Rank	D	E	F	G	H	I
Range	64-91	78-112	96-135	112-166	144-202	190-300

## Application Notes (Continued)



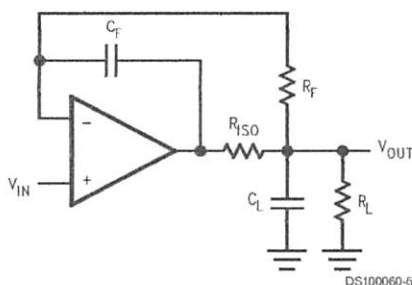
**FIGURE 3. Indirectly Driving A Capacitive Load Using Resistive Isolation**

In *Figure 3*, the isolation resistor  $R_{ISO}$  and the load capacitor  $C_L$  form a pole to increase stability by adding more phase margin to the overall system. The desired performance depends on the value of  $R_{ISO}$ . The bigger the  $R_{ISO}$  resistor value, the more stable  $V_{out}$  will be. *Figure 4* is an output waveform of *Figure 3* using  $620\Omega$  for  $R_{ISO}$  and  $510\text{ pF}$  for  $C_L$ .



**FIGURE 4. Pulse Response of the LMV324 Circuit in Figure 3**

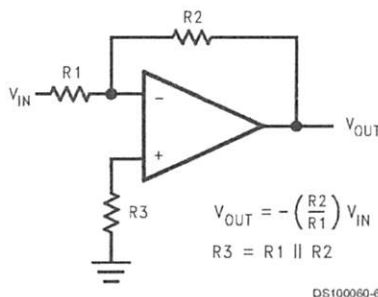
The circuit in *Figure 5* is an improvement to the one in *Figure 3* because it provides DC accuracy as well as AC stability. If there were a load resistor in *Figure 3*, the output would be voltage divided by  $R_{ISO}$  and the load resistor. Instead, in *Figure 5*,  $R_F$  provides the DC accuracy by using feed-forward techniques to connect  $V_{IN}$  to  $R_L$ . Caution is needed in choosing the value of  $R_F$  due to the input bias current of the LMV321/358/324.  $C_F$  and  $R_{ISO}$  serve to counteract the loss of phase margin by feeding the high frequency component of the output signal back to the amplifier's inverting input, thereby preserving phase margin in the overall feedback loop. Increased capacitive drive is possible by increasing the value of  $C_F$ . This in turn will slow down the pulse response.



**FIGURE 5. Indirectly Driving A Capacitive Load with DC Accuracy**

### 3.0 Input Bias Current Cancellation

The LMV321/358/324 family has a bipolar input stage. The typical input bias current of LMV321/358/324 is  $15\text{ nA}$  with  $5\text{V}$  supply. Thus a  $100\text{ k}\Omega$  input resistor will cause  $1.5\text{ mV}$  of error voltage. By balancing the resistor values at both inverting and non-inverting inputs, the error caused by the amplifier's input bias current will be reduced. The circuit in *Figure 6* shows how to cancel the error caused by input bias current.



**FIGURE 6. Cancelling the Error Caused by Input Bias Current**

### 4.0 Typical Single-Supply Application Circuits

#### 4.1 Difference Amplifier

The difference amplifier allows the subtraction of two voltages or, as a special case, the cancellation of a signal common to two inputs. It is useful as a computational amplifier, in making a differential to single-ended conversion or in rejecting a common mode signal.