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

SKRIPSI

PERANCANGAN DAN PEMBUATAN ALAT PENGATUR KEBERANGKATAN BUS SECARA OTOMATIS DI TERMINAL MENGGUNAKAN MIKROKONTROLLER DAN PC

Disusun Oleh :

A. KADIR

01. 17. 138

MARET, 2006

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN ALAT PENGATUR KEBERANGKATAN BUS SECARA OTOMATIS DI TERMINAL MENGGUNAKAN MIKROKONTROLLER DAN PC

SKRIPSI

Disusun dan Diajukan untuk Melengkapi dan Memenuhi Syarat Guna Mencapai Gelar Sarjana Teknik

Disusun Oleh :

A. KADIR

01.17.138

Mengetahui	Diperiksa dan Disetujui
Ketua Jurusan Elektronika S-1	Dosen Pembimbing
(Ir. F. Yudi Limpraptono, MT)	(Ir. Yusuf Ismail Nakhoda, MT)
NIP. Y. 1039500274	NIP. Y. 1018800189

KONSENTRASI TEKNIK ELEKTRONIKA

JURUSAN TEKNIK ELEKTRO S-1

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2006

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSRTI
JURUSAN TEKNIK ELEKTRO S – I
KONSENTRASI ELEKTRONIKA

BERITA ACARA UJIAN SKRIPSI

Nama : A. Kadir

Nim : 01.17.138

Judul Skripsi : Perancangan dan Pembuatan Alat Pengatur Keberangkatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC

Dipertahankan dihadapan team penguji skripsi jenjang Strata Satu (S-I) pada :

Hari : Kamis

Tanggal : 23 Maret 2006

Dengan Nilai : 86,9

Panitia Ujian Skripsi :

Ir. Mochtar Asroni, MSME
Ketua

Ir.F. Yudi Limpraptono, MT
Sekretaris

Anggota Penguji :

Ir.F. Yudi Limpraptono, MT
Dosen Penguji I

Ir. Eko Nurcahyo
Dosen Penguji II

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

BERITA ACARA UJIAN SKRIPSI

Nama : A. Kadir

Nim : 01.17.139

Judul Skripsi : Pemancangan dan Pembuatan Alat Pengatur Kecepatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC

Dipersilahkan dilaksanakan ujian pengujji skripsi jenjang Strata Satu (S-1) pada :

Hari : Kamis

Tanggal : 23 Maret 2006

Dengan Nilai : 80,9

Panitia Ujian Skripsi :

Ir. Mochtar Asroni, MSME
Ketua

Ir.F. Yudi Limpraptono, MT
Sekretaris

Anggota Penguji :

Ir.F. Yudi Limpraptono, MT
Dosen Penguji I

Ir. Eko Nurcahyo
Dosen Penguji II

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSRTI
JURUSAN TEKNIK ELEKTRO S – I
KONSENTRASI ELEKTRONIKA

LEMBAR BIMBINGAN SKRIPSI

1. Nama : A. Kadir
2. Nim : 01.17.138
3. Judul Skripsi : Perancangan Dan Pembuatan Alat Pengatur Keberangkatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC

4. Tanggal Pengajuan : 15 September 2005
5. Selesai Menulis : 18 Maret 2006
6. Dosen Pembimbing : Ir. Yusuf Ismail Nakhoda, MT
7. Telah Dievaluasi Dengan Nilai : 86,9

Mengetahui
Ketua Jurusan

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

Diperiksa dan Disetujui
Dosen Pembimbing

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSRTI
JURUSAN TEKNIK ELEKTRO S – 1
KONSENTRASI ELEKTRONIKA

LEMBAR BIMBINGAN SKRIPSI

1. Nama : A. Kadir
2. Nim : 01.17.138
3. Judul Skripsi : Perancangan Dan Pembuatan Alat Pengatur Keberangkatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC

4. Tanggal Pengajuan : 15 September 2005
5. Selesai Menulis : 18 Maret 2006
6. Dosen Pembimbing : Ir. Yusuf Ismail Nakhoda, MT
7. Telah Dievaluasi Dengan Nilai : 86.9

Mengetahui Ketua Jurusan	Diperiksa dan Disetujui Dosen Pembimbing
Ir. F. Yudi Limpraptono, MT NIP.Y.1039500274	Ir. Yusuf Ismail Nakhoda, MT NIP.Y. 1018800189

بسم الله الرحمن الرحيم

Al Alhamdulillahi Robbil Alamiin

Dengan segala ridho, rahmad dan pertolongan-Mu telah kulewati segala cobaan dan rintangan ini.

Kegelisahan, ketakutan, dan kecemasan hatiku sirna ketika kuangkat tangan ini untuk-Mu.

Menghadap-Mu dengan segala kelemahan dan kekurangan diri.

Robbi...ampunilah hamba atas segala khilaf dan dosa.

Takbir, syahadat, dan sujud syukur akan selalu terpanjat untuk-Mu

Maka berikan kepadaku kesempurnaan iman, ketulusan, sabar, dan kekuatan untuk mengemban hidayah-Mu yang suci ini.

Ya... Rohman... Ya... Rohim...jangan tinggalkan hamba-Mu yang lemah ini walaupun sedetikpun saja **Ya Allah.....**

Ya Allah.....

Limpahkanlah semulia-mulianya sholawat dan salam, atas junjungan kami **Nabi Besar Muhammad SAW**.

Kekasih dari para kekasih, di bumi dan di langit, yang amat santun dan sayang kepada kami.

Allah akan meninggikan orang-orang yang beriman dan
berilmu penmgetahuan itu beberapa derajat,
dan Allah Maha Mengetahui apa-apa yang kamu kerjakan.
(QS. Al-Mujaadalah : 11)

Jazakumullah Khairal Jaza

Walet dan Ummi'.....

Yang telah mengasuh, membesarkan, mendidik, membimbing dan memberikan do'a restu serta kasih sayang yang tak ternilai harganya, sehingga ananda bisa menyelesaikan skripsi ini. Mungkin sampai saat ini merupakan bagian kecil dari harapan Walet dan Ummi' yang bisa ananda wujudkan, tapi ananda akan tetap berusaha untuk memberikan yang terbaik.

Tante.....

Yang selama ini selalu memperhatikan, menyayangi dan membimbing ananda, dan ananda berharap semua itu tidak akan putus dan senantiasa menyertai langka-langkah ananda yang masih panjang ini, terima kasih atas segalanya mungkin hanya Tuhan yang akan membalas semuanya.

Bang Dullah, Kak Uyach (Ary,Ayip, Farish, Nurma), Bang Segaf, Kak Wayya (Hasyimy), Kak Ida.....

Yang selalu menasehati ananda (Jangan Grogi) waktu kompre...terima kasih atas semua bantuannya selama ini, semoga ananda dapat mewujudkan semua harapan-harapan kakanda semua, mungkin hanya Allah SWT yang dapat membalas semuanya.

Terima kasih juga buat :

Adit, Jeky dan juga Motor GL MAX-nya (N 4326 AH) .....

Yang selalu menemani ananda baik suka maupun duka dan tak lupa juga masykur atas segala bantuannya selama ini, semoga kalian nantinya cepat nyusul jadi sarjana juga.

Mas udik, bak reni, bak yanda, Mama, Nadia, Rania.....

Terima kasih atas bantuan do'anya selama ini mungkin hanya Allah AWT yang bisa membalasnya.

Cakanca Sadhaja :

Phoharin Clubs .....

Aviv, Didik (Gajah Seksi), Andis (landak), Bagus (Babi), Erick (Pete'), Ali (Onta), Machmut (Tum), Paidi (Ulat Bulu), Iwan (Awi), Bo im, Sendhy, Joko (Tarzan X), dan laen-laen.....maaf kalau gak bisa kutulis semuanya, susah senang kita lewati bersama terima kasih atas bantuannya selama ini.

Bladhus Raung 03 Zhomenep.....

Ebo' , Dodyk (Mollon), Rico, Ayi' (Telor), Joko (Bilu'), Dayat (Tomo), Ami', Erwin (Ya'),Firman (Kacer), Joni (Mbut), dan yang lain.....maaf kalau ada yang gak disebutin, terima kasih atas segala bantuannya baik do'a atau yang lain, semoga persahabatan kita abadi selalu.

Nur Suryaningrat (Sinyo Edan) & Beny (Bendot).....

Yang selalu bersama baik suka maupun duka berjalan bersama, bekerja bersama ,bermain PS bersama dan akhirnya lulus bersama, Mator Sakalangkong ya kanca.

ABSTRAKSI

A. Kadir, 0117138, 2006, **Perancangan dan Penbuatan Alat Pengatur keberangkatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC.** Teknik Elektro / Konsentrasi Teknik Elektronika S1, 66 halaman. Dosen Pembimbing : Ir. Yusuf Ismail Nakhoda, MT.

Kata kunci: Pengatur Keberangkatan Bus, Mikrokontroller dan PC.

Guna pengaplikasikan suatu perkembangan teknologi mikrokontroller dan pemprograman komputer diperlukan suatu eksperimen aplikasi baru yang dapat membantu kelancaran untuk malaksanakan suatu pekerjaan, sesuai dangan ilmu masing-masing, dalam hal ini salah satu aplikasinya akan diwujudkan menjadi suatu alat pengatur keberangkatan bus yang dilengkapi dengan penampilan suara. Jadi alat ini mampu untuk mengatur keberangkatan dan mengatur antrian bus di terminal secara otomatis dengan keluaran suara.

Dalam perencanaan dan pembuatan alat pengatur keberangkatan bus secara otomatis di terminal menggunakan mikrokontroller dan PC. Modul-modul pendukung antara lain: komputer sebagai pengatur keseluruhan sistem alat ini, modul rangkaian mikrokontroller, modul penampil pada LCD, rangkaian komunikasi serial RS232 .

Setelah melakukan pengukuran dan perhitungan pada alat pengatur keberangkatan bus secara otomatis di terminal ini, serta melakukan perhitungan dalam algoritma pemrograman, maka prosentase dari kesalahan pada alat yang dibuat adalah pada I laserdioda sebesar 1,06 % dan V laserdioda 2,4 % sedangkan pada V photodioda sebesar 3,9 %.

ABSTRAKSI

A. Kadir. 0117133. 2006. **Perancangan dan Pembuatan Alat Pengatur Kecepatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC**. Teknik Elektro / Konsentrasi Teknik Elektronika ST. 66 halaman. 1Dosen Pembimbing : ir. Yusuf Ismail Nakhoda, MT.

Kata kunci: Pengatur Kecepatan Bus, Mikrokontroller dan PC.

Dunia pengaplikasikan suatu perkembangan teknologi mikrokontroller dan pemprograman komputer dipergunakan suatu diperoleh aplikasi baru yang dapat membantu kelancaran untuk mudahsanakan suatu pekerjaan, sesuai dengan ilmu masing-masing, dalam hal ini salah satu aplikasinya akan diwujudkan menjadi suatu alat pengatur kecepatan bus yang dilengkapi dengan penampilan suara. Jadi alat ini maupun untuk mengatur kecepatan dan mengatur antrian bus di terminal secara otomatis dengan keluaran suara.

Dalam perancangan dan pembuatan alat pengatur kecepatan bus secara otomatis di terminal menggunakan mikrokontroller dan PC. Modul-modul pendukung antara lain: komputer sebagai pengatur keseluruhan sistem alat ini, modul rangkaian mikrokontroller, modul percampil pada LCD, rangkaian kommunikasi serial RS232 .

Setelah melakukan pengukuran dan pengujian pada alat pengatur kecepatan bus secara otomatis di terminal ini, serta melakukan perhitungan dalam algoritma pemrograman, maka prosentase dari kesalahan pada alat yang dibuat adalah pada 1 fotodioda sebesar 1.06 % dan 7 fotodioda 2.4 % sedangkan pada 7 fotodioda sebesar 3.7 %.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah melimpahkan rahmat dan petunjuk-Nya sehingga penulis dapat menyelesaikan laporan skripsi yang berjudul Perancangan dan Pembuatan Alat Pengatur Keberangkatan Bus Secara Otomatis Di Terminal Menggunakan Mikrokontroller Dan PC. Penyusunan skripsi ini merupakan salah satu syarat dalam rangka menyelesaikan studi di Fakultas Teknologi Industri Institut Teknologi Nasional Malang.

Dalam kesempatan ini penulis ingin menyampaikan banyak terima kasih kepada :

- Bapak Dr. Ir. Abraham Lomi, MSEE selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. Mochtar Asroni, MSME selaku Dekan Fakultas Teknologi Industri Institut Teknologi Nasional Malang
- Ir. F Yudi Limpraptono, MT. selaku ketua jurusan teknik elektro ITN Malang.
- Ir. Yusuf Ismail Nakhoda, MT. selaku dosen pembimbing yang telah memberikan pengarahan dalam penyusunan laporan ini.
- Seluruh pihak yang telah membantu penulis untuk menyelesaikan laporan skripsi ini.

Penulis menyadari bahwa dalam laporan ini masih banyak terdapat kekurangan dan kesalahan. Oleh karena itu saran dan kritik guna memperbaiki laporan ini sangat diharapkan.

Akhir kata semoga laporan ini dapat berguna bagi semua pihak yang membutuhkan.

Malang, Maret 2006

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

Gambar Halaman

DAFTAR GAMBAR

DAFTAR TABEL

Tabel Halaman

BAB I
PENDAHULUAN

1.1. Latar Belakang

Perkembangan ilmu pengetahuan dan teknologi saat ini terus berkembang pesat seiring dengan perkembangan dunia industri terutama dibidang elektronika. Salah satu perkembangan yang paling menonjol saat ini adalah perkembangan di bidang komputerisasi, selain itu perkembangan teknologi robotika. Suatu sistem yang ditangani oleh komputer , semuanya akan terasa lebih cangih, lebih pintar, lebih otomatis, dan lebih praktis serta efisien.

Dalam kehidupan sehari-hari kita sering bepergian jauh dengan menggunakan bus yang tentunya kita pasti pergi ke terminal untuk naik bus. Di terminal proses pemberangkatan bus ini masih dilakukan secara manual oleh petugas jaga dari Dinas Perhubungan yang bertugas di terminal. Proses pemberangkatan secara manual tersebut masih terasa kurang efisien dan kurang maksimal karena proses pemberangkatan bus masih sering terlambat dan keterlambatan ini akan merugikan bus yang dibelakangnya dan juga masih membutuhkan banyak tenaga dari petugas-petugas di terminal, dan sering kali terjadi kesalahan jadwal pemberangkatan bus yang disebabkan kesalahan pengaturan atau pencatatan jadwal dari petugasnya sehingga mengakibatkan keterlambatan atau kacaunya jadwal pemberangkatan bus.

Dalam skripsi ini akan direncanakan dan dibuat suatu alat yang yang akan memadukan antara teknologi Mikrokontroller dan pemprograman komputer, dimana alat ini nantinya akan diaplikasikan di terminal bus. Seperti kita amati di terminal proses pengaturan atrian dan pengaturan jadwal keberangkatan bus tidak tertata dengan baik dan masih membutuhkan banyak petugas jaga. Alat ini berfungsi sebagai pengganti petugas jaga di terminal yang bertugas mengatur atrian dan pengatur jadwal keberangkatan bus di terminal. Alat ini nantinya akan diatur secara otomatis oleh *Personal Commputer* (PC) sehingga atrian dan pengaturan jadwal keberangkatan bus di terminal dapat berjalan teratur dan lancar.

1.2. Permasalahan

Permasalahan yang diangkat dalam skripsi ini adalah Bagaimana merencanakan dan membuat instalasi *hardware input* dan *output* dari Mikrokontroller, Bagaimana merencanakan dan membuat *software* pada mikrokontroller, serta bagaimana merencanakan dan membuat *software* dengan bahasa pemprograman *Delphi* pada *Personal Commputer* (PC)

1.3. Batasan Masalah

1. Jumlah jalur bus yang dipakai dibatasi tiga jalur saja.
2. Tidak membahas tentang catu daya
3. Menggunakan *Software* dan *Hardware* pendukung AT89S51.
4. Tidak membahas modul penampil suara

1.4. Tujuan

Tujuan dari skripsi ini adalah merancang dan membuat alat pengatur keberangkatan bus secara otimatis di terminal menggunakan Mikrokontroller dan *Personal Commputer* (PC).

1.5. Metodologi

Metodologi penelitian yang dipakai dalam pembuatan skripsi ini adalah :

1. Studi Literatur yang mempelajari teori-teori yang berkaitan mengenai cara kerja komponen-komponen yang digunakan dalam perancangan dan pembuatan alat pengatur keberangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan *Personal Computer* (PC).
2. Perancangan dan Pembuatan alat pengatur keberangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan *Personal Computer* (PC).
3. Pelaksanaan uji coba alat pengatur keberangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan *Personal Computer* (PC).

1.2. Permasalahan

Permasalahan yang dibahas dalam skripsi ini adalah bagaimana merancang dan membuat instalasi hardware input dan output dari Mikrokontroller, bagaimana merancang dan membuat software pada mikrokontroller, serta bagaimana merancang dan membuat software dengan bahasa pemprograman Delphi pada Personal Computer (PC).

1.3. Batasan Masalah

1. Jumlah jalur bus yang dipakai dibatasi tiga jalur saja.
2. Tidak membahas rentang catu daya
3. Menggunakan Software dan Hardware pendukung AT89S51.
4. Tidak membahas modul pemampil suara

1.4. Tujuan

Tujuan dari skripsi ini adalah merancang dan membuat alat pengatur kecerangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan Personal Computer (PC).

1.5. Metodologi

Metodology penelitian yang dipakai dalam pembuatan skripsi ini adalah :

1. Studi Literatur yang mempelajari teori-teori yang berkaitan mengenai cara kerja komponen-komponen yang digunakan dalam perancangan dan pembuatan alat pengatur kecerangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan Personal Computer (PC).
2. Perancangan dan Pembuatan alat pengatur kecerangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan Personal Computer (PC).
3. Pelaksanaan uji coba alat pengatur kecerangkatan bus secara otomatis di terminal menggunakan Mikrokontroller dan Personal Computer (PC).

4. Penyusunan laporan skripsi dan menyimpulkan hasil perancangan dan pembuatan alat

1.6. Sistematika Penulisan

Sistem penulisan yang akan digunakan untuk membahas masalah dalam Tugas Akhir ini diperlukan gambaran susunan alat secara keseluruhan yang selanjutnya ditentukan komponen-komponen utama dan pendukung yang digunakan dan kemungkinan untuk disederhanakan baik untuk bentuk, biaya pembuatanya agar didapatkan susunan yang seefisien dan seefektif mungkin.

Adapun pembahasan Tugas Akhir ini dibagi menjadi beberapa bab sebagai berikut:

BAB I Pendahuluan

Pada pendahuluan ini dibahas mengenai latar belakang perancangan dan pembuatan alat pengatur keberangkatan bus secara otimatis di terminal menggunakan mikrokontroller dan *Personal Commputer* (PC), rumusan masalah, tujuan pembuatan alat, pentingnya alat, dan ruang lingkup perancangan dan pembuatan alat.

BAB II Kajian Pustaka

Pada kajian pustaka ini akan dibahas mengenai teori-teori yang mendasari pembuatan alat ini.

BAB III Desain Proyek dan Implementasi

Memuat perancangan dan pembuatan alat pengatur keberangkatan bus secara otimatis di terminal menggunakan Mikrokontroller dan *Personal Commputer* (PC)

BAB IV Hasil Pengujian dan Analisisa

Memuat hasil pengujian pengatur keberangkatan bus secara otimatis di terminal menggunakan Mikrokontroller dan *Personal Commputer* (PC). Pengujian dilakukan dengan menguji tiap blok rangkaian dan pengujian rangkaian secara keseluruhan.

BAB V Penutup

Berisi kesimpulan dari perancangan dan pembuatan alat pengatur keberangkatan bus secara otimatis di terminal menggunakan Mikrokontroller dan *Personal Commputer* (PC). serta saran-saran yang dapat menyempurnakan alat yang telah dibuat.

BAB II
LANDASAN TEORI

2.1. Mikrokontroller AT89S51

2.1.1. Uraian Singkat

Mikrokontroler Atmel AT89S51 merupakan pengembangan dari Mikrokontroler MCS-51. Mikrokontroler AT89S51 memiliki beberapa kelebihan dan tambahan fitur yang tidak dimiliki oleh mikrokontroler MCS-51,sehingga Mikrokontroler AT89S51 dapat menggantikan mikrokontroler MCS-51.

Mikrokontroler AT89S51 memiliki kelengkapan sebagai berikut:

a. Kompatibel dengan Mikrokontroller MCS-51
b. 4K *Byte downloadable flas memory*
c. Tegangan operasi 4V sampai 5,5V
d. 128 x 8 bit *internal* RAM
e. 32 *programable I/O*
f. 2 buah *timer/counter* 16 bit
g. 3 *level memory clock*
h. Frekuensi osilator 0 hz sampai 33 Mhz
i. *Full duplex UART* (serial port)
j. *Programmable watchdog timer*
k. *Dual data pointer*
l. *Flexsibel ISP programing (byte & page mode)*

BAB II

LANDASAN TEORI

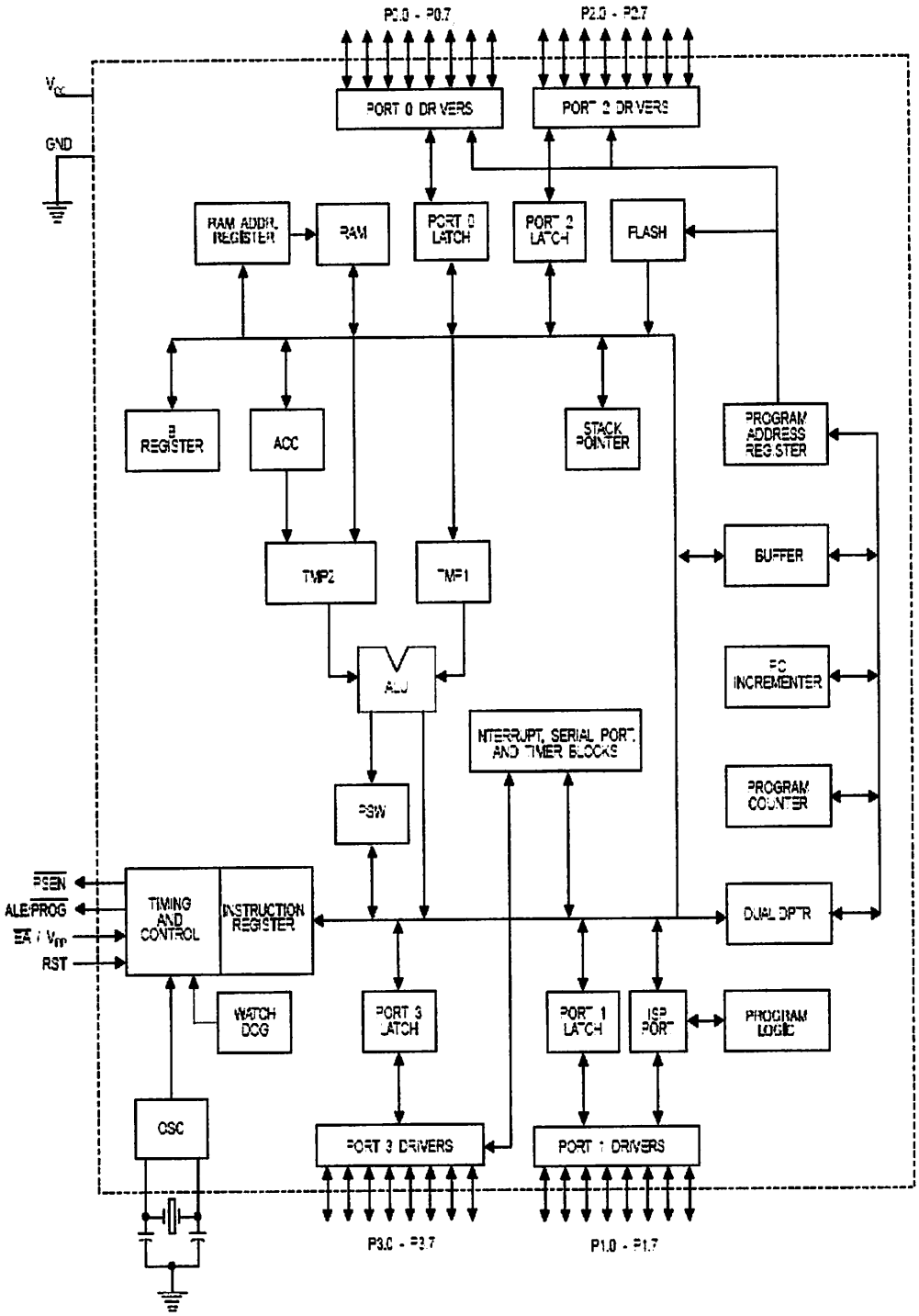
2.1. Mikrokontroller AT89S51

2.1.1. Uraian Singkat

Mikrokontroler Atmel AT89S51 merupakan pengembangan dari Mikrokontroler MCS-51. Mikrokontroler AT89S51 memiliki beberapa kelebihan dan tambahan fitur yang tidak dimiliki oleh mikrokontroler MCS-51sehingga Mikrokontroler AT89S51 dapat menggantikan mikrokontroler MCS-51.

Mikrokontroler AT89S51 memiliki kelengkapan sebagai berikut:

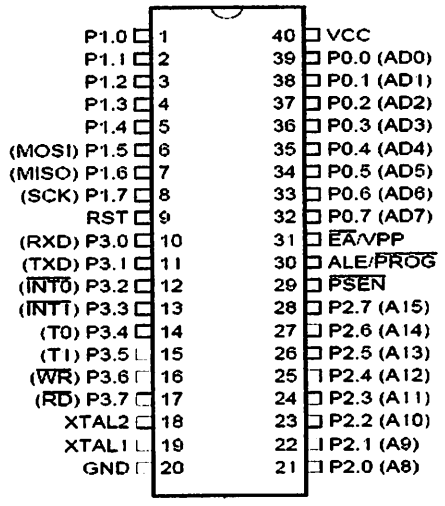
a. Kompatibel dengan Mikrokontroller MCS-51
b. 4K Byte downloadable flas memory
c. Tegangan operasi 4V sampai 5,5V
d. 128 x 8 bit internal RAM
e. 32 programmable I/O
f. 2 buah timer/counter 16 bit
g. 3 level memory clock
h. Frekuensi osilator 0 hz sampai 33 Mhz
i. Full duplex UART (serial port)
j. Programmable watchdog timer
k. Dual data pointer
l. Flexsibel ISP programming (byte & page mode)



Gambar 2.1 Blok Diagram Mikrokontroller Atmel AT89S51[1]

2.1.2. PIN-PIN AT89S51

Gambar menunjukkan penampang pin-pin IC AT89S51. 32 pin dari 40 pin yang dimiliki oleh AT89S51 difungsikan sebagai jalur *I/O.*



Gambar 2.2 Konfigurasi Pin Mikrokontroler AT89S51[1]

- ***Port* 0**

  Port 0 mempunyai 2 fungsi yang berada pada pin 32-39 dari IC AT89S51. didalam rancangan sistem sederhana *port* ini digunakan sebagai *port* I/O serbaguna. Untuk rancangan yang lebih komplek dengan melibatkan memori *eksternal* jalur ini dimultiplek untuk *bus* data dan *bus* alamat.

- ***Port* 1**

  Port 1 disediakan sebagai *port* I/O dan menempati pin 1-8. *port* ini juga mempunyai fungsi tambahan seperti pada tabel berikut:

Tabel 2.1 Fungsi Tambahan Pada *Port* 1[1]

Port Pin	Alternative Functions
P1.5	MOSI (*used for In-system Programming*)
P1.6	MOSI (*used for In-system Programming*)
P1.7	SCK (*used for In-Programming*)

- ***Port* 2**

  Port 2 terletak pada pin 21-28 yang merupakan *port* dua fungsi yaitu sebagai I/O serbaguna, atau sebagai *bus* alamat *byte* tinggi untuk rancangan yang melibatkan memori *eksterna*l.

- ***Port* 3**

  Port 3 adalah *port* dua fungsi yang berada pada pin 10-17, *port* ini memiliki multifungsi, seperti yang terdapat pada tabel berikut:

Tabel 2.2 Fungsi Alternatif *Port* 3[1]

Port Pin	Alternative Funtions
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	$\overline{WR}$ (*internal data memory write strobe*)
P3.7	$\overline{RD}$ (*external memory read strobe*)

➢ ***PSEN* (*program strobe enable*)**

PSEN adalah sebuah sinyal keluaran yang terdapat pada pin 29. Fungsinya adalah sinyal kontrol untuk memungkinkan mikrokontroller membaca program (*kode*) dari memori eksternal. Biasanya pin ini dihubungkan ke pin *OE EPROM*. Jika eksekusi program dari ROM *internal* atau dari *flash* memori, maka *PSEN* berada pada kondisi tidak aktif (*high*).

➢ **ALE (*address latch enable*)**

Sinyal *output* ALE yang berada pada pin 30 fungsinya sama dengan ALE pada mikroprosesor INTEL 8085, 8088, atau 8086. sinyal ALE dipergunakan untuk *demultiplex bus* alamat dan *bus* data.

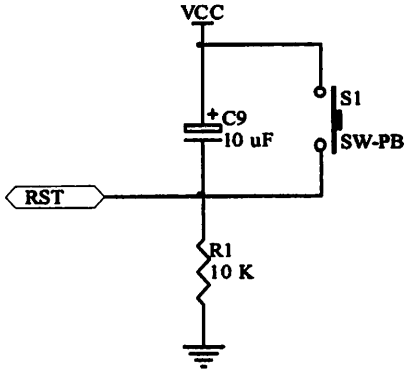
Sinyal ALE membangkitkan pulsa sebesar 1/6 frekwensi *oscillator* dan dapat dipakai sebagai *clock* yang dapat dipergunakan secara umum. Jika *clock* menggunakan XTAl 12 Mhz maka sinyal yang *oscillator* ALE sebesar 2 Mhz.

- ***EA* (*external access*)**

  Masukan sinyal *EA* terdapat pada pin 31 yang dapat diberikan logika rendah (*ground*) atau logika tinggi (+5V). Jika EA diberikan logika tinggi maka mikrokontroler akan mengakses program dari ROM *internal* (EPROM/*Flash* memori). Jika *EA* diberikan logika rendah maka mikrokontroler akan mengakses program dari memori eksternal.

- **RST (*reset*)**

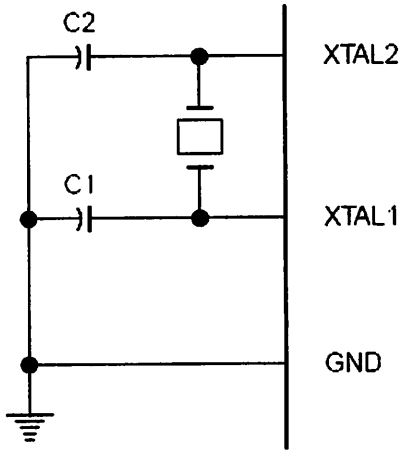
  Input reset pada pin 9 adalah *reset master* untuk AT89S51.



Gambar 2.3 Pemasangan *Oscillator*[4]

➢ **XTAL 1 dan XTAL 2**

Oscillator yang disediakan pada *chip* dikemudikan dengan XTAL yang dihubungkan pada pin 18 dan pin 19. Diperlukan kapasitor penstabil sebesar 30pF. Besar nilai XTAL sekitar 12 Mhz.



Gambar 2.4 Rangkaian *Clock*[1]

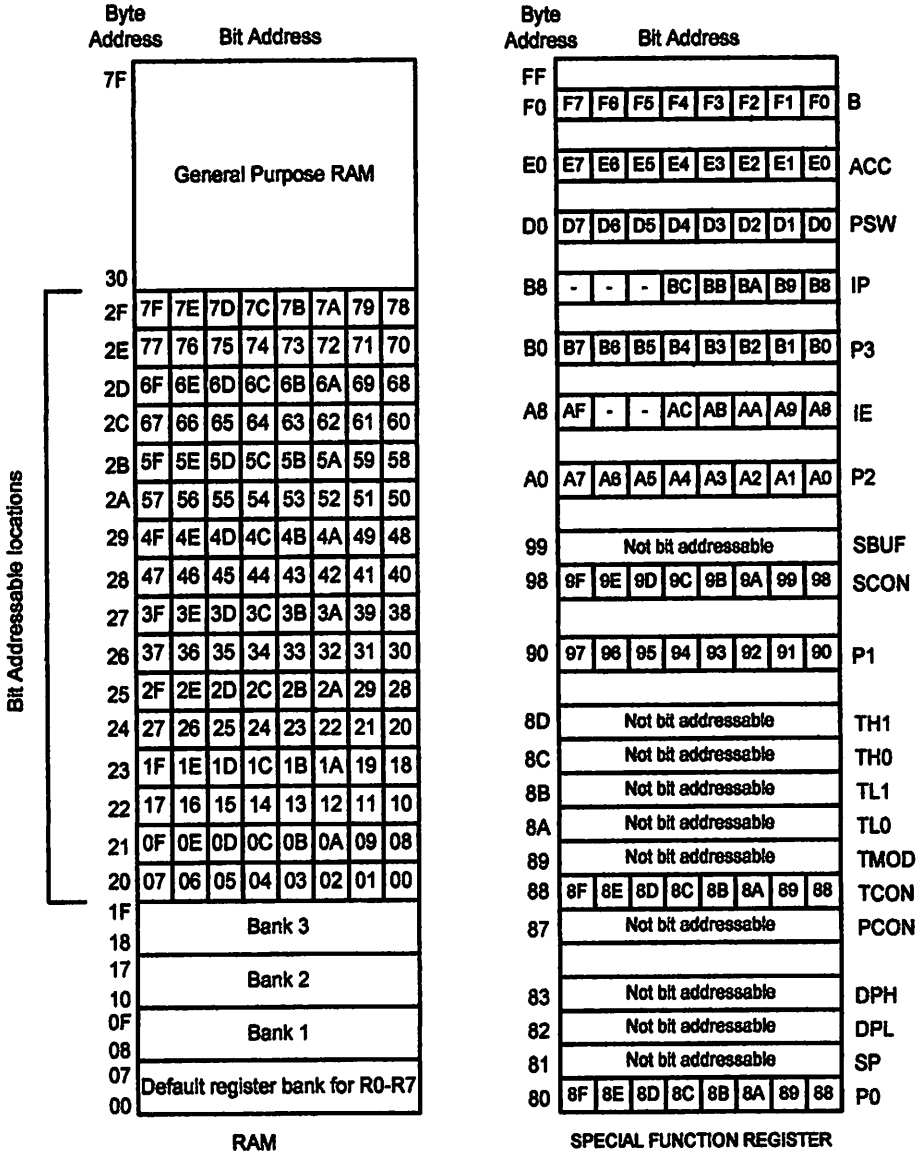
➢ **VCC**

AT89S51 dioperasikan dengan tegangan supplay +5V, pin VCC berada pada nomor 40 dan VSS (*ground*) pada pin 20.

2.1.3. Organisasi Memori

AT89S51 mengimplementasikan ruang memori yang terpisah antara program (*kode*) dan data. Keduanya bisa merupakan memori *internal*, tetapi keduanya dapat diperluas dengan memori eksternal sampai dengan 64K memori program dan 64K memori data.

Memori internal terdiri dari ROM/*Flash Memory* dan RAM data didalam *chip*. RAM berisi susunan *general purpose strorage*, *bit addressable strorage*, *register bank* dan *special function register*. Gambar 2.5 menampilkan secara detail memori data didalam *chip* mikrokontoller MCS-51.



Gambar 2.5 Memori Data Pada Mikrokontroler 8051[4]

Memori internal terdiri dari ROM/Flash Memory dan RAM data didalam chip. RAM berisi susunan general purpose storage, bit addressable storage, register bank dan special function register. Gambar 2.5 menampilkan secara detail memori dan didalam chip mikrokontroller MCS-51.

Gambar 2.5 Memori Data Pada Mikrokontroler 8051[11]

Ruang memori *internal* dibagi menjadi

a. *Register bank* (00H-1FH)

b. *Bit addressable* RAM (20H-2FH)

c. *General purpose* RAM (30H-7FH)

d. *Special function register* (80H-FFH)

➢ ***General Purpose* RAM**

Lokasi memori pada *general-purpose* RAM dapat diakses secara langsung dengan *mode direct* maupun *indirect addressing*. Sebagai contoh, untuk membaca dan mengkopi isi RAM *internal* dengan alamat 5FH kedalam *accumulator* dapat menggunakan instruksi:

MOV A,5FH

Instruksi ini memindahkan *byte* data secara *direct addressing* dengan sumber yang beralamat 5FH. RAM *internal* dapat juga diakses menggunakan *indirect addessing* melalui R0 atau R1. sebagai contoh, 2 instruksi berikut ini:

MOV R0,#5FH

MOV A,@R0

Instruksi pertama dipakai sebagai *immediate addressing* yang memindahkan nilai 5FH kedalam *register* R0,instruksi kedua dipakai sebagai *indirect addressing* untuk memindahkan data yang ditunjuk oleh R0 kedalam *accumulator*.

➢ ***Bit Addressable* RAM**

AT89S51 memiliki 210 lokasi bit *adderssable*, dimana 128 lokasi mempunyai alamat *byte* mulai dari 20H-2FH dan selebihnya merupakan *special funcion*

Ruang memori internal dibagi menjadi

a. Register bank (00H-1FH)

b. Bit addressable RAM (20H-2FH)

c. General purpose RAM (30H-7FH)

d. Special function register (80H-FFH)

- **General Purpose RAM**

Lokasi memori pada general-purpose RAM dapat diakses secara langsung dengan mode direct maupun indirect addressing. Sebagai contoh, untuk membaca dan menyalin isi RAM internal dengan alamat 5FH kedalam accumulator dapat menggunakan instruksi:

MOV A,5FH

Instruksi ini memindahkan byte data secara direct addressing dengan sumber yang beralamat 5FH. RAM internal dapat juga diakses menggunakan indirect addressing melalui R0 atau R1. sebagai contoh, 2 instruksi berikut ini:

MOV R0,#5FH

MOV A,@R0

Instruksi pertama dipakai sebagai immediate addressing yang memindahkan nilai 5FH kedalam register R0.instruksi kedua dipakai sebagai indirect addressing untuk memindahkan data yang ditunjuk oleh R0 kedalam accumulator.

- **Bit Addressable RAM**

AT89S51 memiliki 210 lokasi bit addressable, dimana 128 lokasi menpunyai alamat byte mulai dari 20H-2FH dan selebihnya merupakan special function

register. Adapun pengaksesan bit-bit secara individu dengan *software* adalah fasilitas utama yang dimiliki oleh kebanyakan mikrokontroler. Bit-bit dapat di *set*, di *reset*, di AND kan, di OR kan dan sebagainya, hanya dengan instruksi tungal.

Port I/O AT89S51 merupakan bit *addressable*, sehingga akan menyederhanakan program untuk *interface* I/O satu bit.

Ada 128 lokasi *general purpose* bit *addressable* pada alamat 20H-2FH (8 *bit/byte* X 16 Byte = 128 bit). Alamat-alamat ini dapat diakses sebagai *byte* atau bit, tergantung dari instruksi yang digunakan. Sebagai contoh, untuk mengeset bit pada alamat 67H, instruksi yang digunakan adalah:

SETB 67H

Instruksi diatas tidak berpengaruh pada bit-bit lain pada alamat yang bersangkutan.

➢ ***Register Bank***

32 *register* paling bawah dari memori *internal* berisi *register bank*. Instruksi AT89S51 mendukung 8 *register* yaitu R0-R7,dan kondisi default (setelah sistem reset), *register* ini beralamat 00H-07H. sebagai contoh instruksi berikut ini untuk membaca isi alamat 05H dan meletakkannya kedalam akumulator:

MOV A,R5

Instruksi ini adalah merupakan 1 *byte* instruksi yang menggunakan *register addressing*. Operasi diatas akan sama jika dibentuk dengan 2 *byte* instruksi menggunakan *direct address*

MOV A,05H

Instruksi menggunakan *register* R0-R7 merupakan instruksi yang pendek dan lebih cepat dari pada menggunakan *direct address*.

2.1.4. SFR (*Special Funktion Register*)

Register internal 8051 tersusun sebagai bagian dari RAM *internal* mikrokontroler. *Special Function Registers* (SFR) berjumlah 21 yang terletak pada bagian atas RAM *internal*, yaitu antara alamat 80H hingga FFH, yang ditunjukkan pada gambar dibawah ini.

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000								0D7H
0C8H									0CFH
0C0H									0C7H
0B8H	IP XX000000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0X000000								0AFH
0A0H	P2 11111111		AUXR1 XXXXXXX0				WDTRST XXXXXXXX		0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111								97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0		8FH
80H	P0 11111111	SP 00000111	DPOL 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

Gambar 2.6 Letak SFR Pada Alamat 80H Sampai FFH[1]

➢ **PSW (*Program Status Word*)**

PSW terletak pada alamat D0H yang berisi bit status, selengkapnya terdapat pada tabel 2.3.

Tabel 2.3 *Program Status Word*[4]

BIT	SYMBOL	ADDRESS	BIT DESKRIPTIN
PSW.7	CY	D7H	*Carry flag*
PSW.6	AC	D6H	*Auxiliary carry flag*
PSW.5	F0	D5H	*Flag* 0
PSW.4	RS1	D4H	*Register bank select* 1
PSW.3	RS0	D3H	*Register bank select* 0
			00 = *bank* 0;*address* 00H-07H
			01 = *bank* 1;*address* 08H-0FH
			10 = *bank* 2;*address* 10H-17H
			11 = *bank* 3;*address* 18H-1FH
PSW.2	OV	D2H	*Overflow flag*
PSW.1	-	D1H	*Reserved*
PSW.0	P	D0H	*Even parity flag*

- ***Carry Flag***

  Carry flag (CY) mempunyai dua fungsi, untuk fungsi tradisional CY dipakai untuk operasi matematika. Bit ini akan di *set* apabila selama proses penambahan terdapat bawaan (carry out), atau di *set* juga apabila dalam proses pengurangan terdapat *borrow* pada bit 7.

  Sebagai contoh, jika *accumulator* berisi FFH, lalu diberikan instruksi

  ADD A#,1

  Maka isi *accumulator* adalah 00H dan CY pada PSW akan di *set*.

- ***Auxiliary Carry Flag***

  Jika melakukan penambahan angka BCD (*binary-coded-decimal*), maka *Auxiliary Carry Flag* (CY) akan diset jika bawaan dihasilkan dari bit 3 ke bit 4 atau jika hasil dari *lower nibble* diantara 0AH-0FH.

- ***Flag* 0**

  Flag 0 (F0) adalah *general purpose flag* bit untuk aplikasi pemakai.

- ***Register Bank Select Bits***

  Register bank select bits (RS0 dan RS1) menunjukkan *register bank* yang aktif. *Register* ini dihapus setelah sistem *reset* dan untuk mengubahnya harus melaui *sofware*. Sebagai contoh, 3 instruksi berikut ini mengaktifkan *register bank* 3 dan kemudian memindahkan isi *register* R7 (dengan alamat *byte* 1FH) ke *accumulator*.

  SETB RS1

  SETB RS0

  MOV A,R7

- ***Overflow Flag***

  Overflow flag (OV) diset jika terdapat *arithmatic overflow* setelah operasi penambahan atau pengurangan. Apabila bilangan bertanda ditambahkan atau dikurangkan, *software* dapat memeriksa bit ini untuk menentukan apakah hasil terletak pada *range* yang sebenarnya. Apabila bilangan tak bertanda ditambahkan, bit OV bisa diabaikan. Proses aritmatik dengan hasil-hasil yang lebih besar dari +27 atau lebih kecil dari -128 akan mengeset bit OV.

- ***Parity Bit* (P)**

  Bit paritas (P) secara otomatis di *set* atau di *clear* setiap sirklus mesin untuk membentuk paritas genap dengan akumulator. Jumlah bit-bit 1 didalam akumulator ditambah bit P selalu bernilai genap.

  Sebagai contoh, akulmulator berisi 10101101B, P akan berisi 1 (untuk membentuk bit-bit 1 yang berjumlah 6). Bit paritas biasanya digunakan pada rutin *port* komunikasi serial untuk menyertakan bit paritas sebelum ditransmisikan atau memeriksa bit paritas setelah mengirim data.

- ***Register* B**

  Register B terletak pada alamat F0H digunakan bersama-sama dengan akumulator untuk operasi perkalian dan pembagian.

Instruksi MUL AB mengalikan suatu nilai 8 bit tak bertanda (*unsigned*) didalam *register* A dan B dan memberikan hasil 16 bit di A (*low-byte*) dan di B (*high-byte*).

Instruksi DIV AB membagi A dengan B memberikan hasil integer di A dan sisa di B. register B dapat juga diberikan sebagai *general purpose scratch pad register*. Register ini merupakan *bit-addressable* dengan alamat F0H-F7H.

- ***Stack Pointer***

Stack pointer (SP) adalah *register* 8 bit yang berada pada alamat 81H. register ini berisi alamat yang datanya terletak pada *stack* teratas. Operasi *stack* terdiri dari *pushing* data ke *stack* dan *popping* data dari *stack*.

Pushing data ke *stack* akan menaikkan isi *register* SP sebelum menuliskan data, dan *popping* dari *stack* akan menurunkan isi *register* SP.

Stack AT89S51 berada pada RAM *internal* dan pengaksesan dengan cara *indirect addressing*.

- ***Data Pointer***

Data pointer (DPTR), digunakan untuk mengakses memori kode program atau memori data. *Register* ini merupakan *register* 16 bit pada alamat 82H (DPL, *low byte)* dan 83H (DPH, *high byte*).

Sebagai contoh 3 baris instruksi berikut ini untuk menuliskan data 55H kedalam RAM *eksternal* yang berlokasi pada alamat 1000H:

```
MOV     A,#55H
MOV     DPTR,#1000H
MOVX    @DPTR,A
```

Instruksi pertama digunakan sebagai *immediate addessing* untuk meletakkan data 55H ke *akumulator*. Instruksi kedua juga digunakan juga sebagai *immediate addressing*, untuk meletakkan data 16 bit (10000H) kedalam *data pointer*. A (55H) ke lokasi RAM *eksternal* dimana lokasi alamatnya ditunjuk oleh data yang ada pada DPTR (1000H).

- ***Port Register***

Port I/O AT89S51 terdiri dari *port* 0 pada alamat 80H, *port* 1 pada alamat 90H, *port* 2 pada alamat A0H, dan *port* 3 pada alamat B0H. *port* 0, 2 dan 3 tidak dapat dipakai untuk I/O jika memori eksternal digunakan. Seluruh *port* bersifat *bit-addressable*.

Sebagai contoh, jika sebuah motor dihubungkan dengan suatu *relay* dan transistor *driver relay* dikendalikan melalui *port* 1 bit 7, maka motor akan hidup atau mati hanya menngunakan sebuah instruksi:

```
SETB P1.7
```

Dengan instruksi diatas motor akan hidup, dan motor akan mati dengan instruksi berikut:

```
CLR P1.7
```

➢ ***Timer Register***

AT89S51 mempunyai dua buah *timer/counter* 16 bit yang dapat dipakai sebagai pewaktuan dan penghitungan. *Timer* 0 berada pada alamat 8AH (TL0, *low byte*) dan 8CH (TH0, *high byte*), dan *timer* 1 berada pada alamat 8BH (TL1, *low byte*) dan 8DH (TH1, *high byte*). *Setting* operasi *timer* berada pada *timer mode register* (TMOD) yang beralamat pada 89H dan *timer control register* (TCON) yang berada pada alamat 88H. hanya TCON yang bersifat *bit-addressable*.

➢ ***Serial Port Register***

AT89S51 memiliki *port* serial untuk komunikasi *port* secara *serial* dengan peralatan lain. Register untuk penangan komunikasi *serial* adalah *serial data buffer* (SBUF) pada alamat 99H yang akan menangani pengiriman (*transmit*) dan penerimaan (*receive*) data. Menuliskan data ke SBUF berarti menyiapkan data yanng akan dikirimkan dan untuk membaca data dari SBUF berarti mengakses data yang diterima. Macam-macam *mode* operasi dapat diprogram melalui serial *port control register* (SCON) pada alamat 98H.

➢ ***Interrupt Register***

AT89S51 memiliki 5 sumber *interrupt*. *Interrupt* tidak aktif setelah sistem reset dan diaktifkan melalui interrup *enable register* (IE) pada alamat

A8H. *level* prioritas diset melalui *interrupt periority register* (IP) pada alamat B8H. kedua *register* bersifat *bit-addessable*.

➢ ***Power Control Register***

Power control register (PCON) terletak pada alamat 87H yang berisi beberapa *bit control* dan dirangkum pada tabel berikut ini:

Tabel 2.4 *Power Control Register*[4]

BIT	SIMBOL	DISKRIPSI
7	SMOD	*Double-boud rate bit*: jika diset maka *boud rate didouble* dan berlaku pada *mode serial* 1,2 dan 3.
6	-	Tidak didefinisikan
5	-	Tidak didefinisikan
4	-	Tidak didefinisikan
3	GF1	*General purpose flag bit* 1
2	GF0	*General purpose flag bit* 0
1*	PD	*Power down*: kondisi set untuk mengaktifkan *mode power down*, keluar dari *mode* ini hanya dengan *reset.*
0*	IDL	*Mode idle*; kondisi *set* untuk mengaktifkan *mode idle*; keluar dari *mode* ini hanya dengan *interrupt* atau sistem *reset*

➢ ***Mode Idle***

Pada *mode idle* sinyal *clock internal* untuk CPU akan dihentikan, tetapi *interrupt timer* dan *port serial* tetap aktif. Status CPU masih terpelihara dan semua isi *register* masih terjaga, *Mode idle* dihentikan oleh *interrupt* atau mereset sistem.

➢ ***Power Down Mode***

Pada *mode power down* maka akan terjadi:

a. *Oscillator* akan dihentikan

b. Semua fungsi berhenti bekerja

c. Semua isi RAM *internal* dipertahankan

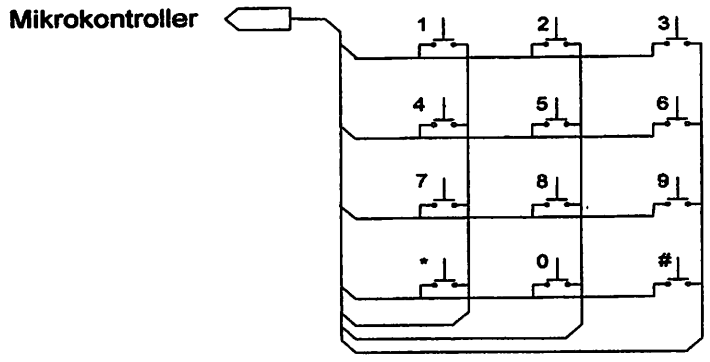
d. Pin *port* dipertahankan pada *level* logikanya

e. Sinyal ALE dan *PSEN* ditahan pada kondisi rendah.

Untuk keluar dari *mode* ini melalui *reset* sistem. Selama *mode power down*, VCC dapat lebih rendah dari 2V.

2.2. *Keypad*

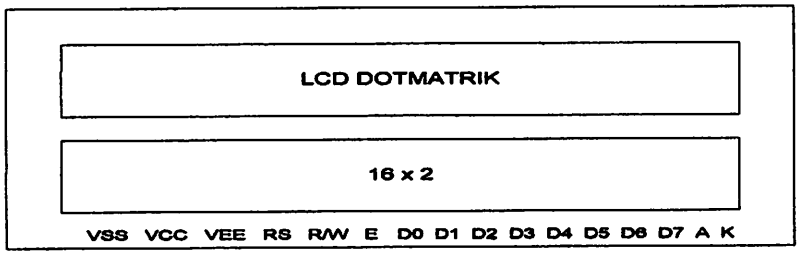
Pada intinya terdiri dari beberapa *switch* yang berhubungan berhubuingan secara matrik. Dimana jumlah dari *switch* tersebut adalah perkalian antara jumlah baris dan jumlah kolom. Fungsi dari *keypad* ini adalah untuk memasukkan *kode* dari tiap bus dengan menekan tombol dari *keypad* ini yang merupakan kombinasi angka yang nantinya akan ditampilkan di LCD dan dikirimkan ke Mikrokontroller .



Gambar 2.7 *Keypad* 3 × 4

2.3. LCD (*Liquit Cristal Display*)

LCD (*Liquid Crystal Display*) digunakan untuk menampilkan *kode* yang dimasukkan melalui *keypad*, kemudian ditampilkan pada LCD yang berupa kombinasi angka.

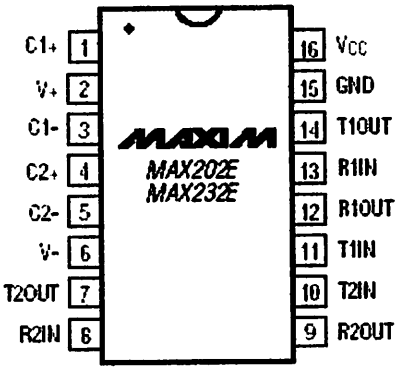


Gambar 2.8 Konfigurasi *Pin LCD Dot Matrik* 16 x 2

2.4. Sistem Komunikasi *Serial RS 232*

Dalam skripsi ini menggunakan komukasi serial antara Mikrokontroller dan Komputer karena antara keduanya tidak dapat langsung dihubungkan. Untuk mengirimkan data dari AT89S51 ke PC *(Personal Computer)*, digunakan *port serial* RS-232 yang terdapat pada PC, dimana pada *port* ini terdapat fungsi-fungsi untuk Tx (pengiriman data), Rx (penerimaan data) dan Tx/Rx (pemilihan mode

Tx atau Rx). Untuk melakukan *transfer* data dari mikrokontroller ke PC digunakan IC MAX232, yang merupakan rangkaian terpadu untuk antarmuka komunikasi serial.



Gambar 2.9 Konfigurasi *Pin-Pin MAX232*[2]

2.5. PC (Personal Computer)

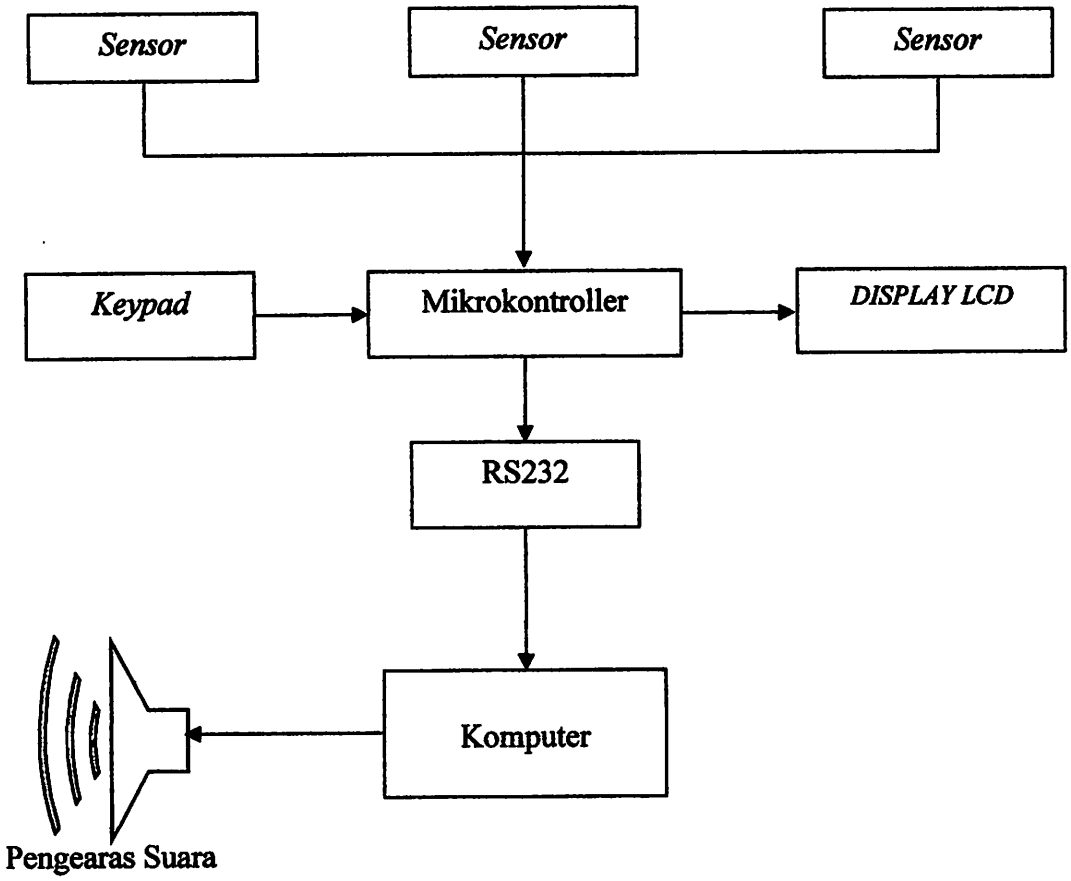
PC (*Personal Computer*) atau lebih dikenal dengan komputer dalam skipsi ini mempunyai peran yang sangat penting karena sebagian besar kerja dari alat yang dibuat ini dikendalikan oleh komputer dengan mernggunakan bahasa pemprograman *Delphi 6.*

BAB III
PERANCANGAN DAN PEMBUATAN ALAT

Pada bab ini akan dibahas mengenai peralatan yang direncanakan dan akan direalisasikan sebagaimana fungsinya. Adapun perencanaan dan pembuatan alat meliputi perencanaan dan pembuatan perangkat keras serta perencanaan dan pembuatan perangkat lunak secara garis besarnya.

3.1. Perencanaan Perangkat Keras

3.1.1. Diagram Blok



Gambar 3.1 Blok Diagram Perancangan Alat Keseluruhan

Fungsi dari tiap blok adalah sebagai berikut :

1. *Sensor* Berfungsi untuk mendeteksi ada tidaknya bus di jalur dimana sensor itu berada.
2. *Keypad* berfungsi sebagai alat atau sarana untuk memasukkan *kode* ke mikrokontroller yang berupa kombinasi angka .
3. Mikrokontroller berfungsi sebagai pembaca masukan dari *keypad* dan menampilkannya di *LCD* kemudian akan meneruskannya ke *server* (PC).
4. RS 232 berfungsi sebagai pengkonversi *level* TTL ke *level* tegangan RS 232 agar bisa dibaca oleh PC (*Personal Commputer*).
5. Komputer (*server)* berfungsi sebagai pengatur komukasi keseluruhan sistem dari alat ini dan menampilkan data-data yang masuk.
6. Pengeras Suara berfungsi untuk menyuarakan keluaran dari komputer yang berupa panggilan bus untuk masuk jalur dan pemberangkatan bus.

3.1.2. *Laser Dioda*

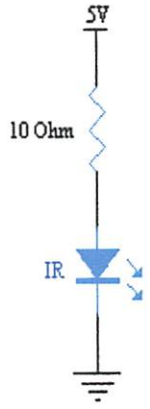
Pada umumnya tegangan kerja dari *Laserdioda* adalah 3 - 5 volt dan arusnya kurang dari 60mA. Pada perancangan alat ini *Laserdioda* diberi tegangan 4,5V dan arus 50ma, sehingga untuk mencari nilai *resistansi* yang digunakan adalah :

$$R = \frac{Vcc - Vled}{Iled}$$

$$= \frac{5 - 4{,}5}{50.10^{-3}}$$

$$= 10\Omega$$

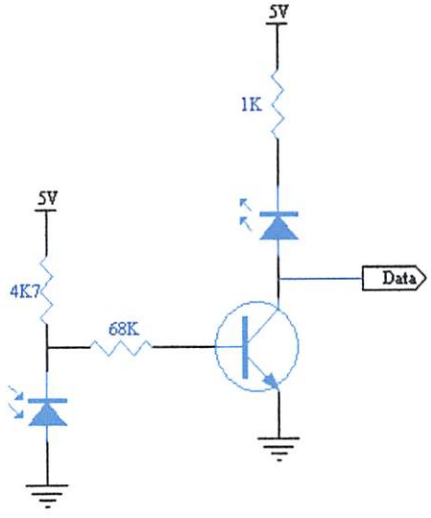
Gambar dari Rangkaian sensor laserdioda dapat dilihat pada gambar dibawah ini :



Gambar 3.2 Rangkaian Sensor Laserdioda

3.1.3. Rangkaian *Detektor Photodioda*

Rangkaian dari sensor *Photodioda* adalah sebagai berikut :



Gambar 3.3 Rangkaian *Detektor Photodioda*

Prinsip kerja dari rangkaian diatas adalah:

1. Kondisi saat *Photodioda* tidak mendapat cahaya dari *Laserdioda*

   Pada saat kondisi ini *Photodioda* mempunyai hambatan besar sekali sehingga tegangan dari Vcc dibias ke transitor. Arus yang melewati *Photodioda* max 10mA, untuk rangkaian diberikan arus sebesar 0,5mA supaya tidak cepat rusak karena rangkaian ini aktif dalam waktu yang lama. Dengan V_{Photo} sebesar 2,3V.

   maka :

   $Vcc = V_B + V_{Photo}$

   $V_R = Vcc - V_{photo}$

   $V_R = 5 - 2,3$

   $V_R = 2,7$ Volt

   $V_R = I_{Phpto.} R_1$

   $2,7 = 5.10^{-4}.R_1$

   $$R_1 = \frac{2,7}{5.10^{-4}}$$

   $R_1 = 5K4$

   Karena dipasaran tidak ada nilai resistor sebesar 5K4 maka diganti dengan nilai terdekat yaitu 4K7

   Dengan tegangan sebesar 2,7V ini, agar dapat mengkondisikan transistor C9014 berada dalam kondisi saturasi ($Vce = 0$). Dengan menentukan nilai R_c sebesar 1K maka dapat dicari nilai tahanan Basis (R_2):

   $$Ic = \frac{Vcc - Vce}{Rc}$$

   $$Ic = \frac{5-0}{1000}$$

   $Ic = 5mA$

   Dari *data sheet* diketahui nilai Bdc dari transistor C9014 =186, maka :

   $$Ib = \frac{Ic}{Bdc}$$

$$Ib = \frac{5mA}{186}$$

Ib = 27uA

Sehingga Nilai R_2 dapat dicari :

$$R_2 = \frac{V_B - Vbe}{Ib}$$

$$R_2 = \frac{(2{,}7 - 0.7)V}{27uA}$$

R_2 = 74,07K

Karena nilai tahanan 74,07K tidak ada dipasaran maka dapat diganti dengan nilai yang terdekat yaitu 68K.

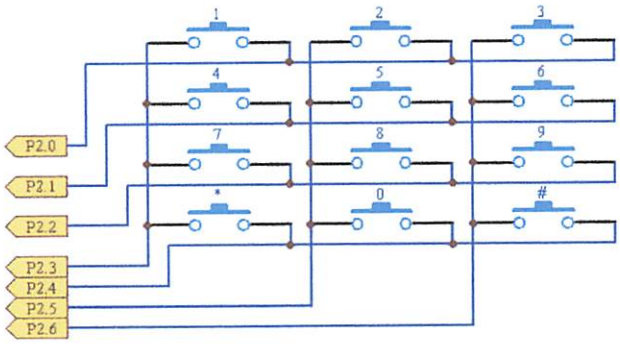
Dengan transistor berada dalam keadan saturasi maka tegangan Vcc akan mengalir ke ground, sehingga praktis dapat dikatakan tegangan yang mengalir menuju mikrokontroller adalah 0V.

2. Kondisi saat *Photodioda* mendapat cahaya dari *Laserdioda*

   Pada saat kondisi ini *Photodioda* mempunyai hambatan kecil, sehigga tegangan tidak dibias, melainkan mengalir ke *ground.* Akibatnya transistor berada dalam kondisi *cutt off*, sehingga tegangan Vcc transistor akan mengalir ke mikrokontroller.

   Jadi dapat disimpulkan bahwa photodioda tidak mendapat cahaya maka logika ke mikrokontroller adalah 0(*low*). Pada saat *Photodioda* mendapat cahaya maka logika ke mokrokontroller adalah 1(*high*).

3.1.4. Rangkaian *Keypad*



Gambar 3.4 Rangkaian *Keypad* 3 X 4

Pada perancangan skripsi ini diguanakan *Keypad 3 X 4* dimana fungsinya yaitu sebagai media untuk memasukkan data kode bus yang terdiri dari kombinasi angka. Pin-pin pada *Keypad* ini akan langsung dihubungkan ke Mikrokontroller yaitu pada *Port* 2.0 sampai *port* 2.6.

3.1.5. LCD (*Liquit Cristal Display*)

Sebagai penampil (*display*) dari masukan kombinasi angka melalui *Keypad* yang merupakan Kode tiap bus pada alat ini digunakan LCD jenis TM162 yang membutuhkan arus maksimal 3 mA dan tegangan 5V. LCD jenis TM162 dapat menampilkan 16 karakter dalam 2 baris. Dalam perencanaan skripsi ini untuk penyambungan antara *LCD* ke Mikrokontroller AT89S51 digunakan IC 74LS164 yang merupakan IC *converter* yang berfungsi sebagai register geser dan juga untuk memudahkan proses pengkabelan antara *LCD* dan Mikrokontroller AT89S51.

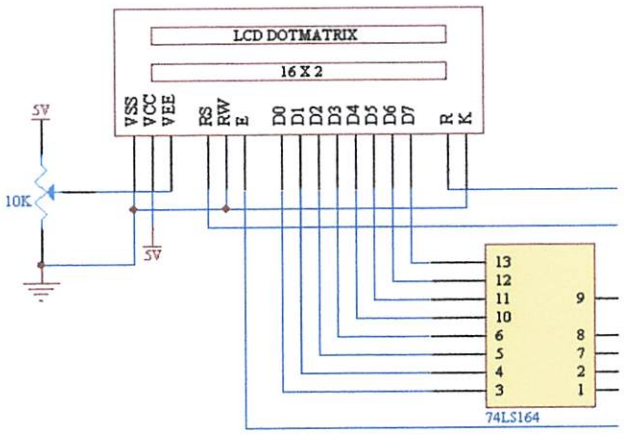
3.1.4. Rangkaian Keypad

Gambar 3.4 Rangkaian Keypad 3 X 4

Pada perancangan skripsi ini digunakan Keypad 3 X 4 dimana fungsinya yaitu sebagai media untuk memasukkan data kode bus yang terdiri dari kombinasi angka. Pin-pin pada Keypad ini akan langsung dihubungkan ke Mikrokontroller yaitu pada Port 2.0 sampai port 2.6.

3.1.5. LCD (Liquid Crystal Display)

Sebagai penampil (display) dari masukan kombinasi angka melalui Keypad yang mempakan Kode tiap bus pada alat ini digunakan LCD jenis TM162 yang membutuhkan arus maksimal 3 mA dan tegangan 5V. LCD jenis TM162 dapat menampilkan 16 karakter dalam 2 baris. Dalam perancangan skripsi ini untuk penyambungan antara LCD ke Mikrokontroller AT89S51 digunakan IC 74LS164 yang merupakan IC converter yang bertungsi sebagai register geser dan juga untuk memudahkan proses pengkabelan antara LCD dan Mikrokontroller AT89S51.



Gambar 3.5 Rangkaian LCD (*Liquit Cristal Display*)

Ket:

- Vcc : Tegangan +5V
- Vss : *Ground*
- Vee : LCD *Drive*
- RS : Instruksi/data register yang dihubungkan ke P1.1
- R/W : *Signal* pemilih *mode* baca atau tulis yang dihubungkan *Ground*
- E : *Signal* untuk mengaktifkan tulis data atau baca datayang dihubungkan ke P1.0
- DB0-DB7 : *Bus* data 8 bit, untuk membaca atau menulis data. DB7 juga digunakan untuk *busy flag.*
- R, K (*Back Light*) : Bila LCD yang digunakan tanpa *back light*, diode IN4001 tidak perlu dipasang.

3.1.6. Minimum Sistem AT89S51

Rangkaian sistem minimum dari mikrokontroller AT89S51 terdiri dari rangkaian *clock* dan *reset*. Rangkaian tersebut tersusun dari komponen-komponen 3 buah kapasitor, 1 buah IC mikrokontroller, sebuah resistor dan sebuah kristal atau resonator keramik. Rangkaian kapasitor dan kristal atau resonator keramik digunakan sebagai rangkaian pembangkit *internal clock generato*r yang terdapat

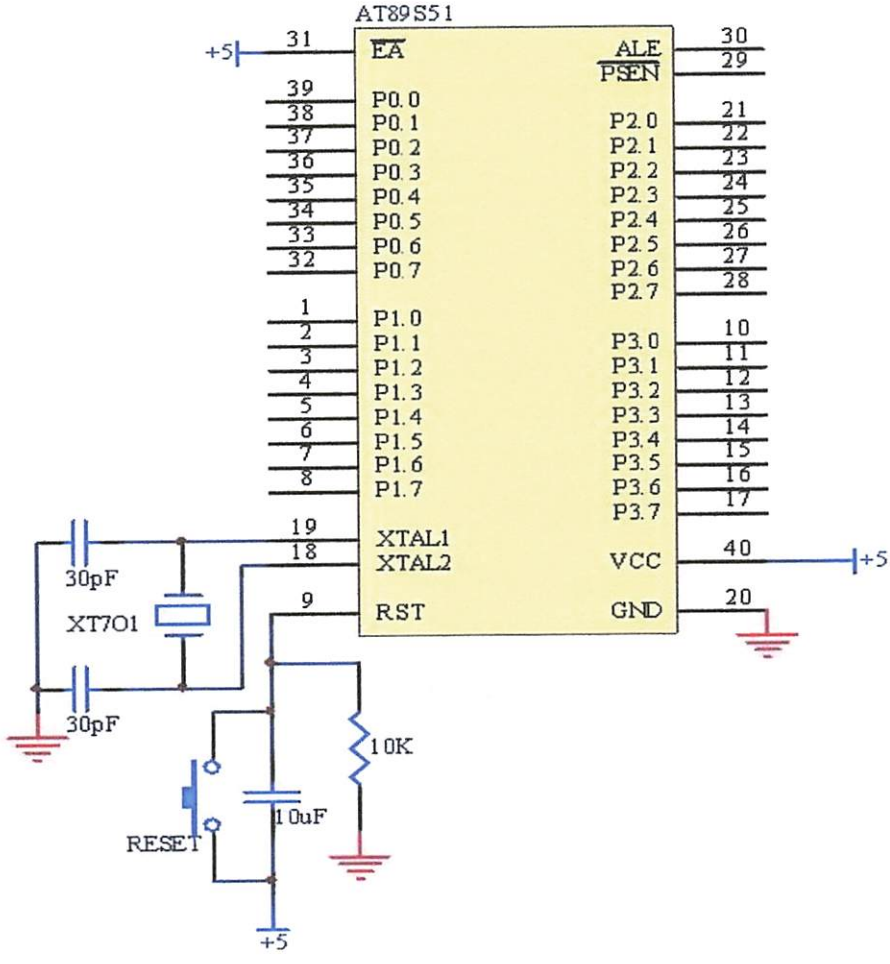
pada AT89S51. Nilai kapasitansi ditentukan sesuai dengan jenis *oscillator* yang digunakan, yaitu:

C1 dan C2 = 20pF – 40pF untuk kristal

C1 dan C2 = 30pF – 50pF untuk resonator keramik.

Karena dalam rancangan digunakan *oscillator* kristal maka harga kapasitor yang penulis gunakan adalah sebesar 30pF.

Mikrokontroller AT89S51 mempunyai frekwensi maksimal 33 MHz, dimana 1 siklus mesin = 12 clock. Dalam rangkaian digunakan kristal dengan harga 12 MHz, maka program akan dijalankan pada setiap langkahnya selama 1μs.

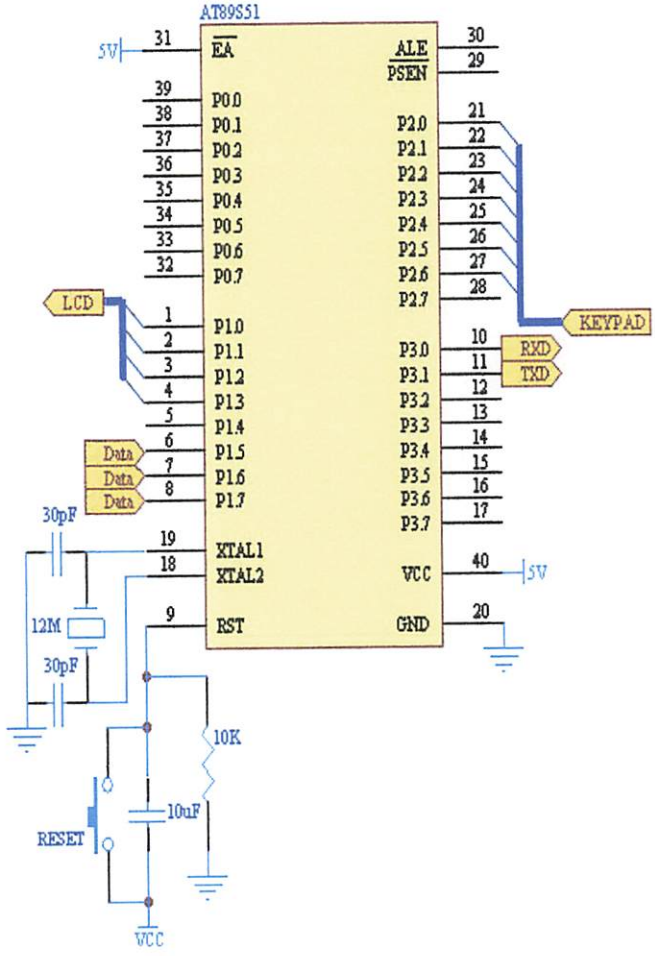


Gambar 3.6 Rangkaian Minimum Sistem Mikrokontroller AT89S51

3.1.7. Perancangan Penggunaan Port-Port Pada Mikrokontroller AT89S51

Pada skripsi ini IC Mikrokontroller AT89S51 digunakan sebagai pembaca inputan data dari keypad yang kemudian mikrokontroller ini akan menampilkannya ke LCD dan selanjutnya akan mengirimkan data tersebut ke Komputer. Untuk melaksanakan fungsi tersebut diatas maka perlu dirancang *port-port* I/O serta sinyal-sinyal yang akan digunakan dengan seksama.

Gambar 3.7 menunjukkan rancangan *port-port* I/O serta sinyal-sinyal pada IC mikrokontroller AT89S51 yang dimanfaatkan pada skripsi ini.



Gambar 3.7 Rancangan Pemakaian *Port-Port* Mikrokontroller AT89S51

- ***Port* 0**

  Port 0 merupakan *port* dua fungsi yang berada pada pin 32-39 dari IC AT89S51. dalam perancangan skripsi ini *port* 0 tidak digunakan.

- ***Port* 1**

  Port 1 disediakan sebagai *port* I/O dan menempati pin 1-8,dalam perancangan P1.0 dihubungkan ke E (*Enable Signal*), dan P1.1 dihubungkan ke RS (*Register Selection Signal*) pada LCD,*Port* 1.2 dihubungkan ke *pin* 8 pada IC 74LS164, sedangkan *pin* 1 dan 2 pada *IC* 74LS164 digabung dan dihubungkan dengan *port* 1.3, *port* 1.5, 1.6, 1.7 dihubungkan ke *sensor*.

- ***Port* 2**

  Port 2 yang berada pada pin 21-28 merupakan *port* dua fungsi yaitu sebagai I/O serbaguna.

  P 2.0 sampai P2.6 dihubungkan dengan *keypad* yang nantinya akan memberikan masukan data ke Mikrokontroller.

- ***Port* 3**

  Port 3 merupakan *port* dua fungsi yang berada pada pin 10-17 dari IC AT89S51. Pada perancangan alat ini *port* 3 yang digunakan hanya *port* 3.0 (RXD) dan *port* 3.1 (TXD).

- $\overline{PSEN}$ **(*Program Strobe Enable*)**

  $\overline{PSEN}$ adalah suatu sinyal keluaran yang terdapat pada pin 29. Fungsinya adalah sebagai sinyal kontrol untuk memungkinkan mikrokontroler membaca program (*code*) dari memori eksternal. Jika eksekusi program dari ROM internal (8051/8052) atau dari flash memori AT89S51 , maka $\overline{PSEN}$ berada pada kondisi tidak aktif (*high*).

- ***ALE* (*Address Latch Enable*)**

  Sinyal *output* ALE yang berada pada pin 30 fungsinya untuk demultipleks *bus* alamat dan *bus* data. Sinyal ALE membangkitkan pulsa sebesar 1/6 frekwensi *oscillator* dan dapat dipakai sebagai *clock* yang dipergunakan secara umum.

- ***$\overline{EA}$ (External Access)***

  Masukan sinyal $\overline{EA}$ terdapat pada pin 31 yang dapat diberikan logika rendah (pin terhubung *ground*) atau logika tinggi (pin terhubung Vcc). Jika $\overline{EA}$ diberikan logika tinggi, maka mikrokontroller akan mengakses program dari ROM *internal* (EPROM/*flash memory*). Jika $\overline{EA}$ diberikan logika rendah, maka mikrokontroller akan mengakses program dari memori eksternal. Pada skripsi ini $\overline{EA}$ dihubungkan ke Vcc

- **RST *(Reset)***

  Input reset pada pin 9 adalah *reset master* untuk AT89S51.

- ***Oscilator***

  Oscillator yang disediakan pada *chip* dikemudikan dengan XTAL yang dihubungkan pada pin 18 dan pin 19. Besar nilai XTAL yang digunakan sebesar 12 MHz untuk keluarga MCS-51, dan diperlukan dua buah kapasitor penstabil sebesar 30pF.

- **Vcc**

  AT89S51 dioperasikan dengan tegangan *supply* +5V. Pin Vcc berada pada pin nomor 40 sedangkan Vss (*ground*) berada pada pin 20.

- ***Timer***

  Didalam mikrokontroler AT89S51 terdapat 2 buah *timer/counter* masing-massing diberi nama *timer/counter* 0 dan *timer/counter* 1atau disingkat T0 dan T1. *Clock input* dari kedua *timer* ini didapat dari frekwensi X-TAL yang

dipasang dibagi dengan 12. Kedua *timer* bisa dipakai untuk meng-*interupt* program dengan selang waktu yang bias kita tentukan dengan memakai rumus:

1. Bila dipakai sebagai *timer/counter* 8 bit

$$T = (256 - TLx) * \frac{1}{\left(\frac{F.osc}{12}\right)}$$

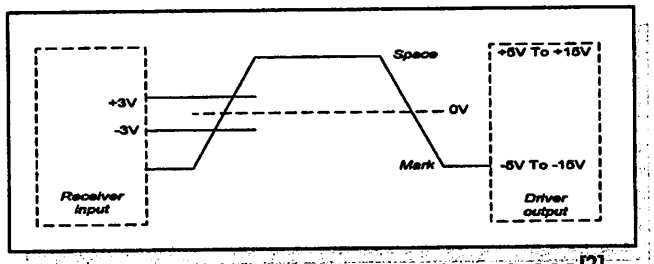
2. Bila dipakai sebagai *timer/counter*16bit

$$T = (65536 - THx\,TLx) * \frac{1}{\left(\frac{F.osc}{12}\right)}$$

3.1.8. Perancangan Komunikasi Serial RS232

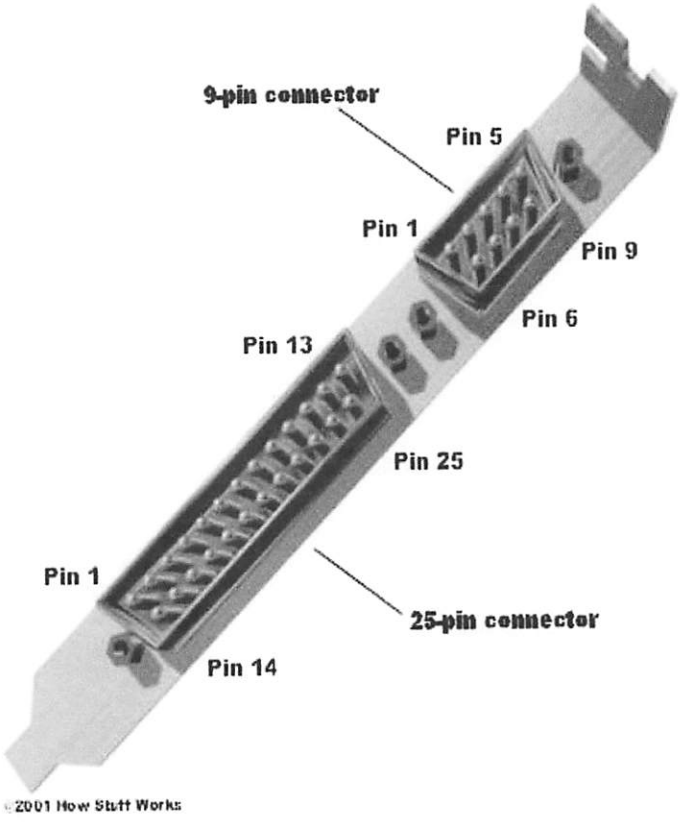
RS-232 merupakan seperangkat alat yang berfungsi sebagai *interface* dalam proses *transfer* data secara *serial*. Metode pengiriman secara serial RS-232 adalah *Asinkron*. Pengiriman *Asinkron* berarti tidak membutuhkan pewaktu sebagai Sinkronisasi. Dalam pengiriman serial *asinkron*, *clock* tidak dikirimkan, tetapi dikondisikan oleh *timing start bit* yang merupakan isyarat dari sumber ke tujuan untuk mengkodekan adanya pengiriman karakter sudah selesai dikirim.

Karakteristik elektris dari sistem RS-232 adalah mempunyai tegangan keluaran antara –15 Volt sampai dengan +15 Volt. Teganan +5 sampai +15 volt untuk mewakili level rendah (logika '0'/ *spacing*) dan tegangan -5 sampai -15 volt untuk mewakili level tinggi (logika '1'/ *marking*). Hal tersebut seperti ditunjukkan dalam Gambar 3.8.



Gambar 3.8 Level Logika Standar RS-232[2]

Di dalam komputer terdapat fasilitas komunikasi serial yang menggunakan standar RS-232, yaitu terletak pada COM1 dan COM2. Kedua fasilitas ini menggunakan konektor DB9 atau DB25 sebagai penghubung dengan piranti luar. Gambar konektor DB9 seperti terdapat dalam Gambar 3.9.



Gambar 3.9 Konektor DB-9 dan DB-25[7]

Fungsi masing-masing pin pada DB-9 seperti terdapat dalam Tabel 3.1.

Tabel 3.1. Fungsi Pin RS-232 dalam DB-9[7]

Pin	Nama	Fungsi
1	DCD (*Data Carrier Detect*)	Mendeteksi sinyal *carrier* dari modem lain
2	RD (*Receive Data Line*)/ (R_XD)	Pengiriman data serial dari DCE ke DTE
3	TD (*Transmit Data Line*)/(T_XD)	Pengiriman data serial dari DTE ke DCE
4	DTR (*Data Terminal Ready*)	Memberitahu DCE bahwa DTE telah aktif dan siap untuk bekerja
5	*Ground*	Referensi semua tegangan antarmuka
6	DSR (*Data Set Ready*)	Memberitahu DTE bahwa DCE telah aktif dan siap untuk bekerja
7	RTS (*Request To Send*)	Memberitahu DCE bahwa DTE akan mengirim data
8	CTS (*Clear To Send*)	Memberitahu DTE bahwa DCE siap menerima data
9	RI (*Ring Indikator*)	Aktif jika modem menerima sinyal ring pada jalur telepon

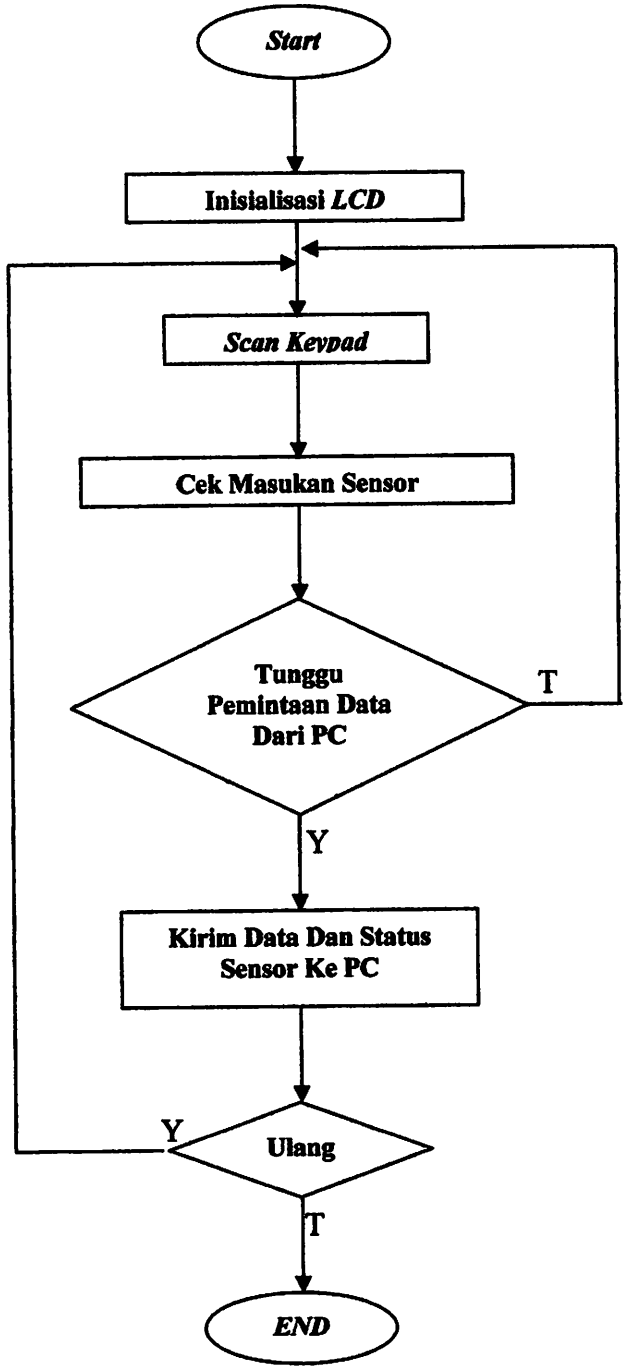
Jalur data (TxD dan RxD) untuk transport data, TxD adalah jalur *output* pada komputer, data dikirim dari pin ini. Sedangkan RxD adalah penerima untuk komputer, data yang datang akan diterima oleh pin ini. Pin ke empat adalah *output* (RTS) di mana sebuah sinyal akan diberikan pada alat yang dihubungkan dengan maksud meminta kiriman data. CTS adalah sinyal masukan yang menunggu sinyal dari alat yang terhubung. Ketika alat tersebut menerima sinyal

RTS dan bisa menerima data maka ia akan mengirimkan sinyal balik yang merupakan CTS. DTR adalah sinyal keluaran yang memberi tanda bahwa ada alat yang terhubung dan akan mengirimkan data. DSR merupakan sinyal *input* yang mana jika alat yang terhubung manerima sinyal DTR ia akan memberi sinyal balik kemudian diterima sebagai sinyal DSR.

3.2. Perancangan Perangkat lunak

3.2.1. Perancangan Perangkat Lunak Pada Mikrokontrller

Cara kerja dari perangkat lunak (*sofware*) pada Mikrokontroller secara umum sebagai berikut:



Gambar 3.10 *Folwcart* Perangkat Lunak Pada Mikrokontroller

3.2. Perancangan Perangkat lunak

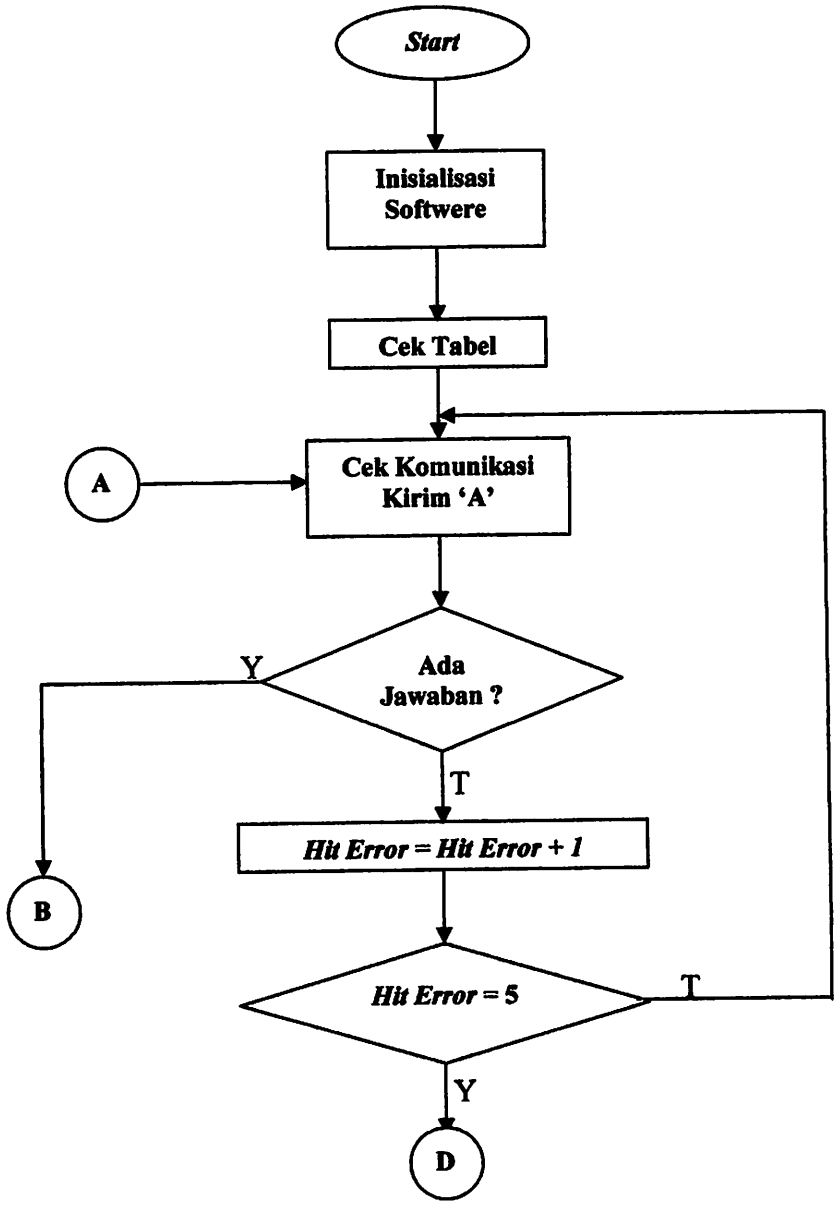
3.2.1. Perancangan Perangkat Lunak Pada Mikrokontroller

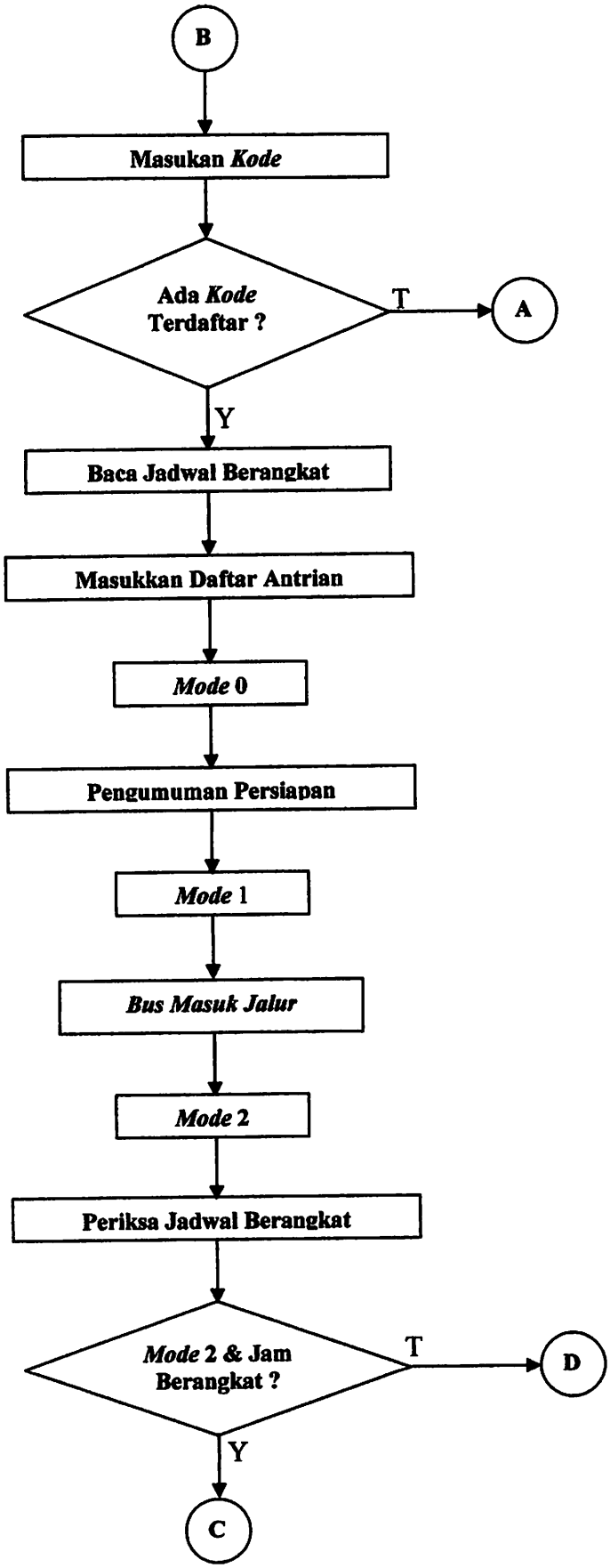
Cara kerja dari perangkat lunak (software) pada Mikrokontroller secara umum sebagai berikut:

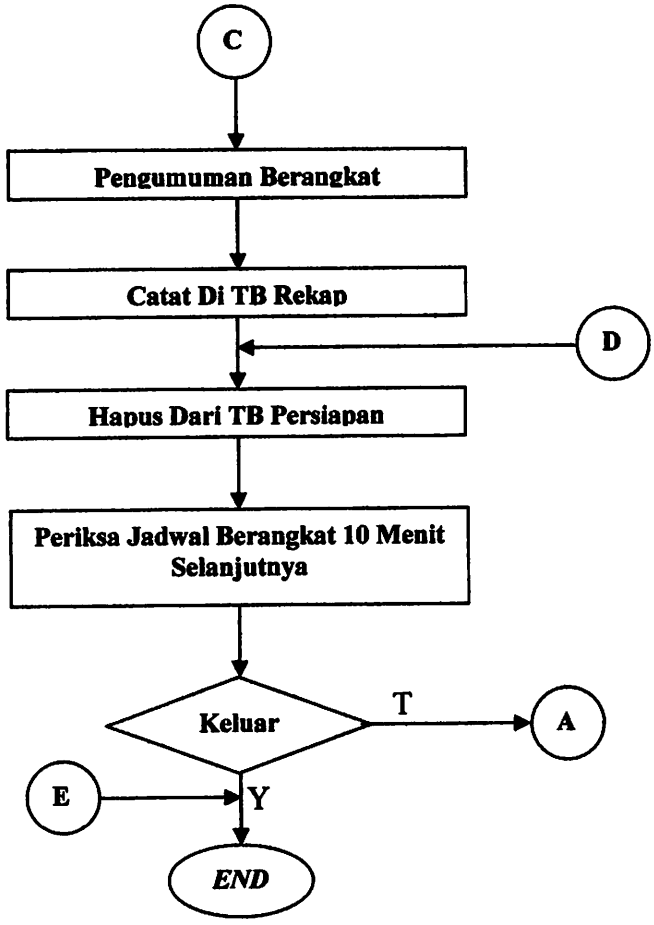
Gambar 3.10 Flowchart Perangkat Lunak Pada Mikrokontroller

3.2.2. Perancangan Perangkat Lunak (*Softwere*) Pada PC

Berikut ini merupakan *Flow Cart* perancangan perangkat lunak (*Softwere*) pada PC.




B
Masukan *Kode*
Ada *Kode* Terdaftar ?
T
A
Y
Baca Jadwal Berangkat
Masukkan Daftar Antrian
Mode 0
Pengumuman Persiapan
Mode 1
Bus Masuk Jalur
Mode 2
Periksa Jadwal Berangkat
Mode 2 & Jam Berangkat ?
T
D
Y
C



Gambar 3.11 *Folwcart* Perangkat Lunak Pada Komputer

3.3. Prinsip Kerja Alat

Prinsip kerja alat pengatur keberangkatan bis secara otomatis di terminal ini yaitu pertama-tama bis harus bis harus mendaftarkan data-data bisnya ke petugas yang mengoperasikan komputer sebagai *server* dari alat ini, selanjutnya setelah mendaftar masing-masing bis akan mendapatkan *ID* bis. Selanjutnya apabila bis tersebut telah siap untuk masuk daftar antrian maka supir bis memasukkan *ID* nya melalui *Keypad* yang selanjutnya *Keypad* akan meneruskannya ke Mikrokontroller dan mikrokontroller akan menampilkannya ke *LCD (Liquit Cristal Display)*, apabila data yang dimasukkan telah benar maka untuk mengirimkan data tersebut ke komputer maka ditekan tombol kirim yang berada di *Keypad.* Setelah datanya terkirim maka *Softwere* di komputer akan mencocokkannya dengan data *Base* yang ada di komputer apabila data *ID* yang dimasukkan ada yang cocok maka data dari *ID* tersebut akan dimasukkan ke daftar antrian dengan *Mode 0*, 10 menit sebelum keberangkatan bis yang bersangkutan akan dipanggil untuk masuk jalur sesuai dengan data jurusannya dan *Mode*nya akan berubah menjadi *Mode 1*,Apabila bis telah masuk ke jalurnya maka *Sensor* yang berada di jalur akan memberikan informasi ke komputer sehingga *Mode*nya berubah menjadi *Mode 2*,tepat pada jam keberangkatannya maka bis akan secara otomatis akan diberangkatkan, dan selanjutnya program akan memanggil daftar antrian 10 menit selanjutnya.

BAB IV
PENGUJIAN ALAT

4.1. Umum

Pengujian alat ini dilakukan untuk mengetahui kinerja dari keseluruhan sistem rangkaian. Jadi pada tahap ini akan diketahui nilai-nilai serta parameter-parameter dari setiap bagian yang menyusun sistem secara keseluruhan.

4.2. Pengujian Perangkat Keras

4.2.1. Pengujian Sistem Mikrokontroller

1. Tujuan

Untuk mengetahui kondisi awal dari mikrokontroller apakah sudah sesuai dengan yang direncanakan.

2. Peralatan Yang Dibutuhkan

1. Komputer.
2. Sistem mikrokontroller.

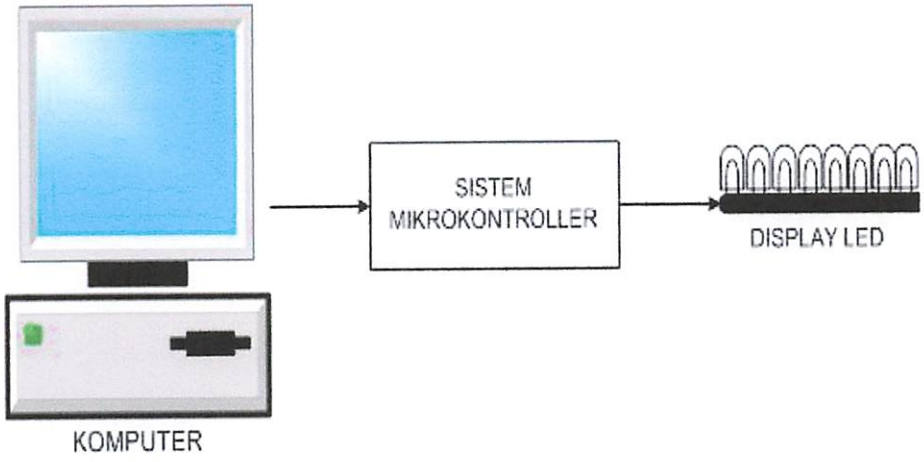
3. Prosedur Pengujian

1. Membuat program yang digunakan dalam pengujian mikrokontroller.
2. Program yang digunakan dalam pengujian mikrokontroller ini merupakan program sederhana dengan memasukkan bilangan 0FH dan F0H ke *port* 2.
3. *Port* 2 AT89C51. Program yang dibuat adalah sebagai berikut :

```
        ORG     0H
MULAI:  MOV     P2,#0FH         ;kondisi satu
        CALL    DELAY
        MOV     P2,# F0H        :kondisi dua
        CALL    DELAY
```

```
        JMP     MULAI
DELAY:MOV       R0,#0
DELAY1: MOV     R5,#50H
        DJNZ    R5,$
        DJNZ    R0,DELAY1
        END
```

4. Membuat rangkaian seperti Gambar 4.1:



Gambar 4.1 Diagram Blok Pengujian Mikrokontroller

5. Memasang catu rangkaian sebesar 5V
6. Men*download* program diatas .
7. Mengamati keluaran pada LED *Display* .
8. Hasil pengujian

   Hasil pengujian mikrokontroller adalah sebagai berikut :

Tabel 4.1 Hasil Pengujian Sistem Mikrokontroller

KONDISI	KELUARAN PADA *LED DISPLAY*							
I	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Satu	0	0	0	0	1	1	1	1
Dua	1	1	1	1	0	0	0	0

Pada saat *port* 2 diberi bilangan 0FH, LED pada P2.0 sampai dengan P2.3 padam, dan LED pada P2.4 sampai P2.7 nyala. Ketika *port* 2 diberi bilangan F0H, P2.0 sampai P2.3 nyala dan P2.4 sampai P2.7 padam. Kemudian program terus mengulang sehingga LED akan menyala dan padam secara bergantian.

4.2.2. Pengujian Rangkain *Sensor Laserdioda*

1.Tujuan :

Mengetahui nilai tegangan dan arus dari *sensor laserdioda.*

2. Peralatan yang dibutuhkan :

- Catu daya 5V DC.
- *Digital Multimeter.*
- Rangkaian sensor laserdioda.

3. Prosedur pengujian :

- Alat-alat dirangkai seperti ditunjukkan pada gambar 4.2 dibawah ini :

Tabel 4.1 Hasil Pengujian Sistem Mikrokontroller

KONDISI	KELUARAN PADA LED DISPLAY							
	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Satu	0	0	0	0	1	1	1	1
Dua	1	1	1	1	0	0	0	0

Pada saat port 2 diberi bilangan 0FFH, LED pada P2.0 sampai dengan P2.3 padam, dan LED pada P2.4 sampai P2.7 nyala. Ketika port 2 diberi bilangan F0H, P2.0 sampai P2.3 nyala dan P2.4 sampai P2.7 padam. Kemudian program terus mengulang sehingga LED akan menyala dan padam secara bergantian.

4.2.2. Pengujian Rangkaian Sensor Laserdioda

1.Tujuan :

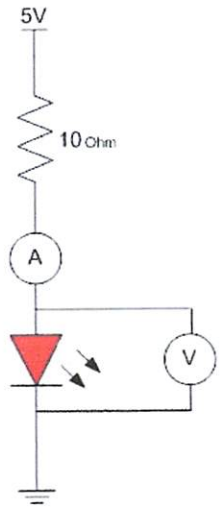
Mengetahui nilai tegangan dan arus dari sensor laserdioda.

2. Peralatan yang dibutuhkan :

- Catu daya 5V DC.
- Digital Multimeter.
- Rangkaian sensor laserdioda.

3. Prosedur pengujian :

- Alat-alat dirangkai seperti ditunjukkan pada gambar 4.2 dibawah ini :



Gambar 4.2 Rangkaian Pengujian Laserdioda

- Rangkaian diberi catu daya 5V DC.
- Dilakukan pengukuran arus dan tegangan pada rangkaian pengujian sensor *Laserdioda*.

4. Hasil Pengujian

Hasil pengujian rangkaian sensor laserdioda adalah sebagai berikut

Tabel 4.2 Hasil Perbandingan Antara Pengukuran Dan Perhitungan Rangkaian *Laserdioda*

PENGUKURAN		**PERHITUNGAN**	
$I_{laserdioda}$ (mA)	**$V_{laserdioda}$ (*Volt*)**	**$I_{laserdioda}$ (mA)**	**$V_{laserdioda}$ (*Volt*)**
49,47	4,39	50	4,5

$$Error\ \mathrm{I}_{laserdioda} = \frac{I\,\mathrm{inf}(perhitungan) - I\,\mathrm{inf}(pengukuran)}{I\,\mathrm{inf}(perhitungan)} \times 100\%$$

$$Error\ \mathrm{I}_{laserdioda} = \frac{50 - 49{,}47}{50} \times 100\%$$

$$Error\ \mathrm{I}_{laserdioda} = 1{,}06\ \%$$

$$Error\ \mathrm{V}_{laserdioda} = \frac{V\,\mathrm{inf}(perhitungan) - V\,\mathrm{inf}(pengukuran)}{V\,\mathrm{inf}(perhitungan)} \times 100\%$$

$$Error\ \mathrm{V}_{laserdioda} = \frac{4{,}5 - 4{,}39}{4{,}5} \times 100\%$$

$$Error\ \mathrm{V}_{laserdioda} = 2{,}4\%$$

4.2.3. Pengujian Rangkaian Sensor *Photodioda*

1. Tujuan

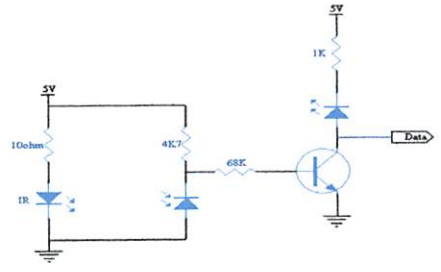
Mengetahui nilai tegangan dan arus pada rangkaian sensor *photodioda*

2. Peralatan yang dibutuhkan

- Catu daya 5V DC.
- *Digital Multimeter.*
- Rangkaian sensor *Photodioda.*

3. Prosedur pengujian

- Alat-alat dirangkai seperti ditunjukkan pada gambar 4.3 dibawah ini :



Gambar 4.3. Rangkaian Pengujian *Sensor Photodioda*

- Rangkaian diberi catu daya 5V DC
- Dilakukan pengukuran arus dan tegangan pada rangkaian pengujian *sensor photodioda.*

4. Hasil Pengujian

Hasil pengujian rangkaian *Sensor Photodioda* adalah sebagai berikut :

Tabel 4.3 Hasil Perbandingan Antara Pengukuran Dan Perhitungan Rangkaian *Photodioda*

$I_{Photodioda}$ (mA)	Pengukuran	Perhitungan
	$V_{Photodioda}$ *(Volt)*	$V_{Photodioda}$ *(Volt)*
0,5	2,21	2,3

- Kondisi Sensor Photodioda terhalang :

$$Error\ V_{Photodioda} = \frac{Vphoto(perhitungan) - Vphoto(pengukuran)}{Vphoto(perhitungan)} \times 100\%$$

$$Error\ V_{Photodioda} = \frac{2,3 - 2,21}{2,3} \times 100\%$$

$$Error\ V_{Photodioda} = 3,9\%$$

4.2.4. Pengujian *LCD*

1. Tujuan

Mengetahui apakah rangkaian LCD dapat menampilkan data karakter yang sesuai dengan data yang dikirimkan.

- Rangkaian diberi catu daya 5V DC
- Dilakukan pengukuran arus dan tegangan pada rangkaian penguji sensor photodioda.

4. Hasil Pengujian

Hasil pengujian rangkaian Sensor Photodioda adalah sebagai berikut :

Tabel 4.3 Hasil Perbandingan Antara Pengukuran Dan Perhitungan Rangkaian Photodioda

$I_{photodioda}$ (mA)	Pengukuran	Perhitungan
	$V_{photodioda}$ (Volt)	$V_{photodioda}$ (Volt)
0,5	2,21	2,3

- Kondisi Sensor Photodioda terhalang :

$$Error\ V_{photodioda} = \frac{Vphoto(perhitungan) - Vphoto(pengukuran)}{Vphoto(perhitungan)} \times 100\%$$

$$Error\ V_{photodioda} = \frac{2,3 - 2,21}{2,3} \times 100\%$$

$$Error\ V_{photodioda} = 3,9\%$$

4.2.4. Pengujian LCD

1. Tujuan

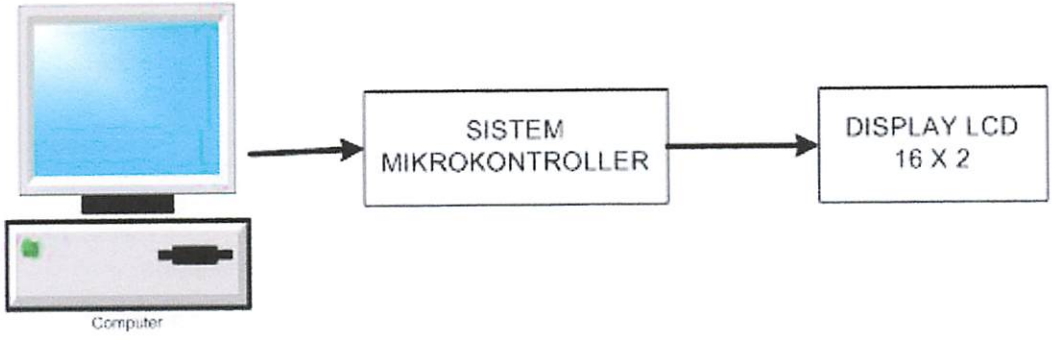
Mengetahui apakah rangkaian LCD dapat menampilkan data karakter yang sesuai dengan data yang dikirimkan.

2. Peralatan Yang Dibutuhkan

- Komputer.
- Sistem mikrokontroller AT8S51.
- Catu daya 5V DC.

3. Prosedur Pengujian

- Merangkai peralatan seperti dalam Gambar 4.4.
- Membuat perangkat lunak pengujian rangkaian LCD. Program ini berisi inisialisasi Mikrokontroller, dan LCD.
- Mengaktifkan catu daya
- Mengoperasikan program, hasil keluaran akan ditunjukkan pada layar penampil kristal cair.



Gambar 4.4 Diagram Blok Pengujian *LCD*

4. Hasil Pengujian

Dari hasil pengujian didapatkan bahwa rangkaian *LCD* dapat menampilkan karakter-karakter, sesuai dengan data yang dikirimkan. Tampilan terdiri atas 2 baris yang masing-masing mempunyai 11 karakter.

Gambar 4.5 Hasil Pengujian *LCD*

4.2.5. Pengujian Rangkaian *Keypad*

1. Tujuan

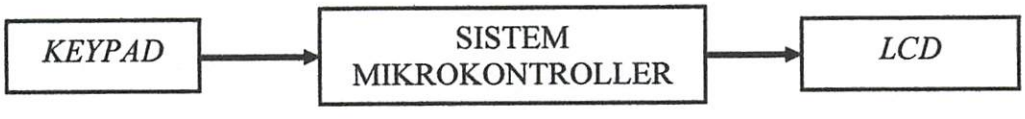
Untuk mengetahui keluaran dari unit papan tombol saat tombol ditekan apakah sesuai dengan tampilan yang ada di *LCD*.

2. Peralatan yang dibutuhkan

- Catu daya 5V DC.
- Rangkaian *keypad.*
- *Digital Multimeter.*
- MCU
- Rangkaian *LCD*

3. Prosedur pengujian :

- Alat-alat dirangkai seperti ditunjukkan pada gambar diagram blok dibawah ini



Gambar 4.6 Diagram Blok Pengujian *Keypad*

Gambar 4.5 Hasil Pengujian LCD

4.2.5. Pengujian Rangkaian Keypad

1. Tujuan

Untuk mengetahui keluaran dari unit papan tombol saat tombol ditekan apakah sesuai dengan tampilan yang ada di LCD.

2. Peralatan yang dibutuhkan

- Catu daya 5V DC.
- Rangkaian keypad
- Digital Multimeter.
- MCU
- Rangkaian LCD

3. Prosedur pengujian :

- Alat-alat dirangkai seperti ditunjukkan pada gambar diagram blok dibawah ini

Gambar 4.6 Diagram Blok Pengujian Keypad

4. Hasil Pengujian

Dari hasil pengujian dapat diketahui bahwa rangkaian *Keypad* dapat berfungsi dengan baik hal ini dapat diketahui dari tombol yang ditekan sesuai dengan tampilan yang ada di *LCD,* berikut ini table perbandingan tombol yang ditekan dengan tampilan di *LCD* :

Tabel 4.4 Hasil Pengujian Rangkaian *Keypad*

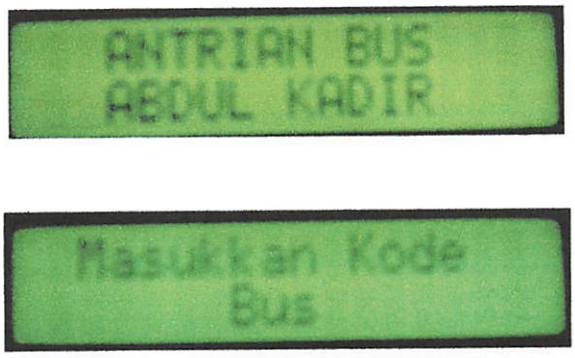
SWITCH	Tampilan *LCD*
1	1
2	2
3	3
4	4
5	5
6	6
7	7
8	8
9	9
*	*
0	0
#	#

4.2.6. Pengujian Perangkat Keras Secara Keseluruhan

1. Tujuan

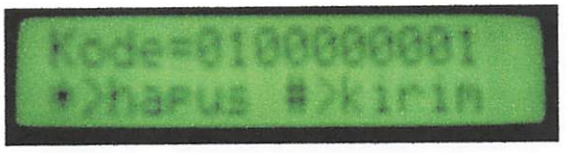
   Tujuan dari pengujian perangkat keras secara keseluruhan ini adalah untuk mengetahui kinerja perangkat keras secara keseluruhan apabila dijalankan sesuai perintah yang kita jalankan.

2. Peralatan yang dibutuhkan :
   - Catu daya 12 V
3. Prosedur pengujian
   - Merangkai semua perangkat keras menjadi satu seperti yang tampak pada diagram blok.
   - Mengaktifkan catu daya



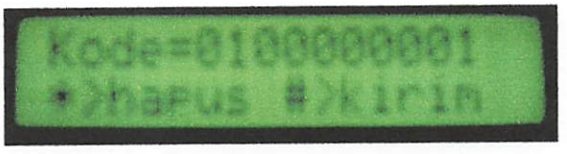
Gambar 4.7 Tampilan Awal *LCD*

- Memasukkan *kode* bis



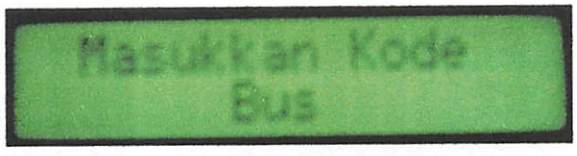
Gambar 4.8 Tampilan *LCD* Pada Saat Memasukkan *Kode* Bis

- Mengamati *kode* yang ada di *LCD*



Gambar 4.9 Tampilan *LCD* Pada saat Kode Bis Lengkap

- Mengirimkkan data ke komputer



Gambar 4.10 Tampilan *LCD* Setelah Data Dikirimkan

Gambar 4.7 Tampilan Awal (2)

- Memasukkan kode bis

Gambar 4.8 Tampilan (2) Pada Saat Memasukkan Kode Bis

- Memgamasi kode yang ada di (2)

Gambar 4.9 Tampilan (2) Pada saat Kode Bis Lengkap

- Mengirimkan data ke komputer

Gambar 4.10 Tampilan (2) Setelah Data Dikirimkan

4.3. Pengujian Perangkat Lunak

4.3.1. Pengujian Perangkat Lunak Pada Mikrokontroller

Program yang digunakan ialah program *assembly*. Pengujian program *assembly* ini dilakukan pada saat mengkompile program sumber *assembly*, yang merupakan kumpulan baris-baris perintah dan telah disimpan dengan *extention* .ASM. Program ini ditulis menggunakan HB2000W *Programmer & Evaluation Board For* AT89S/C/51/52. Pada bagian proses *assembly*, program .ASM akan dikompile menjadi dua bagian, yaitu *listing assembly* *.LST dan program obyek *.HEX yang berisikan kode-kode yang hanya dikenali mikrokontroller. Program inilah yang akan di*download*kan ke mikrokontroller.

4.3.2. Pengujian Perangkat Lunak Pada Komputer

Program yang digunakan adalah bahasa pemprograman *Borland Delphy 6* jadi di komputer ini dibuat suatu program yang nantinya akan memproses data yang dikirimkan oleh Mikrokontroller.Berikut ini tampilan utama dari Program pengatur keberangkatan bus ini :

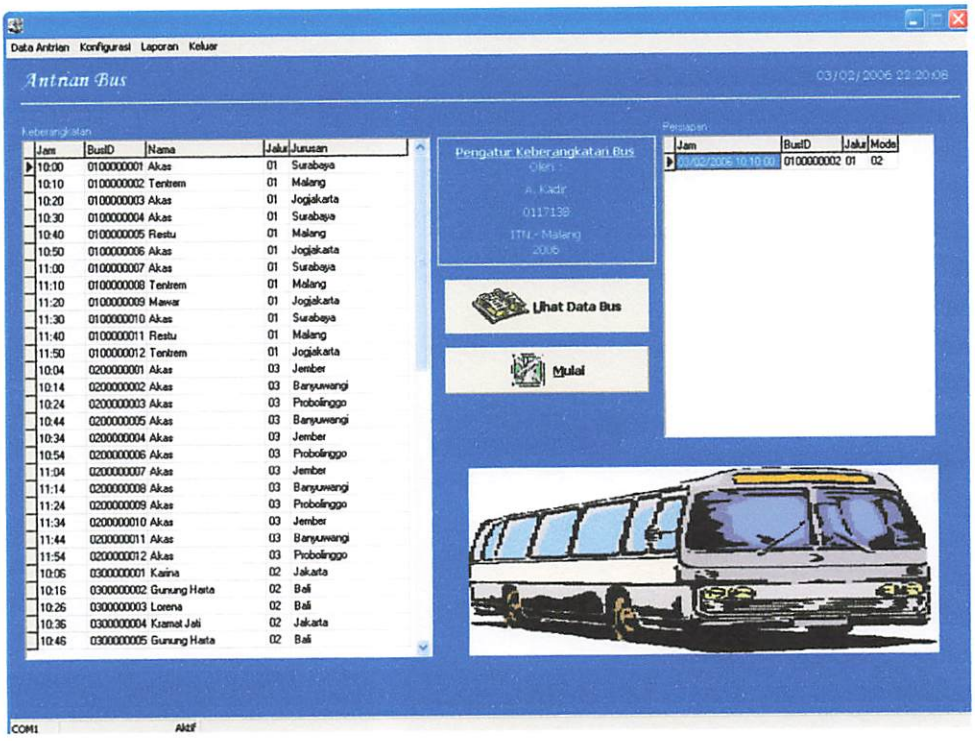
4.3. Pengujian Perangkat Lunak

4.3.1. Pengujian Perangkat Lunak Pada Mikrokontroller

Program yang digunakan ialah program assembly. Pengujian program assembly ini dilakukan pada saat mengkompile program sumber assembly yang merupakan kumpulan baris-baris perintah dan telah disimpan dengan extention .ASM. Program ini ditulis menggunakan HB2000W Programmer & Evaluation Board For AT89S(C)51/52. Pada bagian proses assembly program .ASM akan dikompile menjadi dua bagian, yaitu having assembly *.LST dan program object *.HEX yang berisikan kode-kode yang hanya dikenali mikrokontroller. Program inilah yang akan didownloadkan ke mikrokontroller.

4.3.2. Pengujian Perangkat Lunak Pada Komputer

Program yang digunakan adalah bahasa pemprograman Borland Delphi 6 jadi di komputer ini dibuat suatu program yang manfaya akan memproses data yang dikirimkan oleh MikrokontrollerBerikut ini tampilan utama dari Program pengatur keberangkatan bus ini :

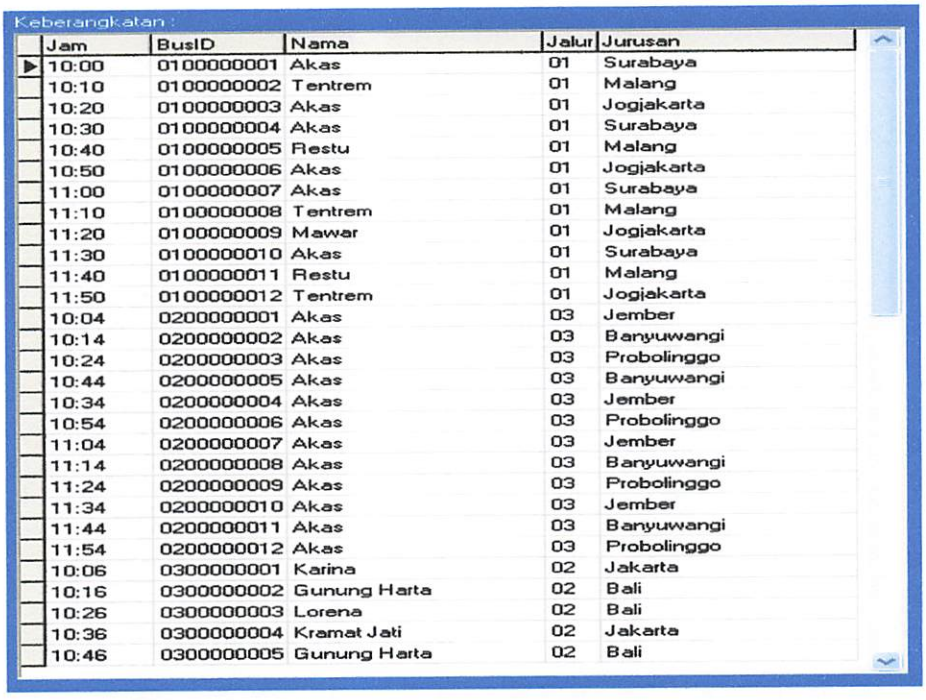


Gambar 4.11 Gambar Tampilan Menu Utama Pengatur Keberangkatan Bus

Dari gambar diatas dapat kita lihat bahwa terdapat dua tabel yaitu tabel keberangkatan dan tabel persiapan, dalam tabel keberangkatan terdapat 5 kolom yaitu :

- Jam keberangkatan
- *ID* bis
- Nama bis
- Jalur keberangkatan
- Jurusan.

Lebih jelasnya mengenai tabel keberangkatan ini dapat dilihat pada gambar dibawah ini :


Keberangkatan :

Jam	BusID	Nama	Jalur	Jurusan
10:00	0100000001	Akas	01	Surabaya
10:10	0100000002	Tentrem	01	Malang
10:20	0100000003	Akas	01	Jogjakarta
10:30	0100000004	Akas	01	Surabaya
10:40	0100000005	Restu	01	Malang
10:50	0100000006	Akas	01	Jogjakarta
11:00	0100000007	Akas	01	Surabaya
11:10	0100000008	Tentrem	01	Malang
11:20	0100000009	Mawar	01	Jogjakarta
11:30	0100000010	Akas	01	Surabaya
11:40	0100000011	Restu	01	Malang
11:50	0100000012	Tentrem	01	Jogjakarta
10:04	0200000001	Akas	03	Jember
10:14	0200000002	Akas	03	Banyuwangi
10:24	0200000003	Akas	03	Probolinggo
10:44	0200000005	Akas	03	Banyuwangi
10:34	0200000004	Akas	03	Jember
10:54	0200000006	Akas	03	Probolinggo
11:04	0200000007	Akas	03	Jember
11:14	0200000008	Akas	03	Banyuwangi
11:24	0200000009	Akas	03	Probolinggo
11:34	0200000010	Akas	03	Jember
11:44	0200000011	Akas	03	Banyuwangi
11:54	0200000012	Akas	03	Probolinggo
10:06	0300000001	Karina	02	Jakarta
10:16	0300000002	Gunung Harta	02	Bali
10:26	0300000003	Lorena	02	Bali
10:36	0300000004	Kramat Jati	02	Jakarta
10:46	0300000005	Gunung Harta	02	Bali

Gambar 4.12 Gambar Tampilan Tabel Keberangkatan

Sedangkan tabel kedua yaitu tabel persiapan dalam tabel ini dimasukkan data-data bis yang telah siap untuk diberangkatkan dan telah masuk dalam daftar antrian. Dalam tabel ini terdapat terdapat 4 kolom yaitu :

- Jam keberangkatan
- *ID* bis
- Jalur keberangkatan
- Mode, untuk mode disini dibagi menjadi 4 mode yaitu
  - Mode 0 berarti bis telah memasukkan *ID* nya dan telah dimasukkan dalam daftar antrian.
  - Mode 1 berarti bis telah dipanggil untuk persiapan di jalur yang telah ditentukan.
  - Mode 2 berarti menandakan bahwa bis telah berada di jalur.
  - Mode 3 berarti bis telah diberangkatkan.

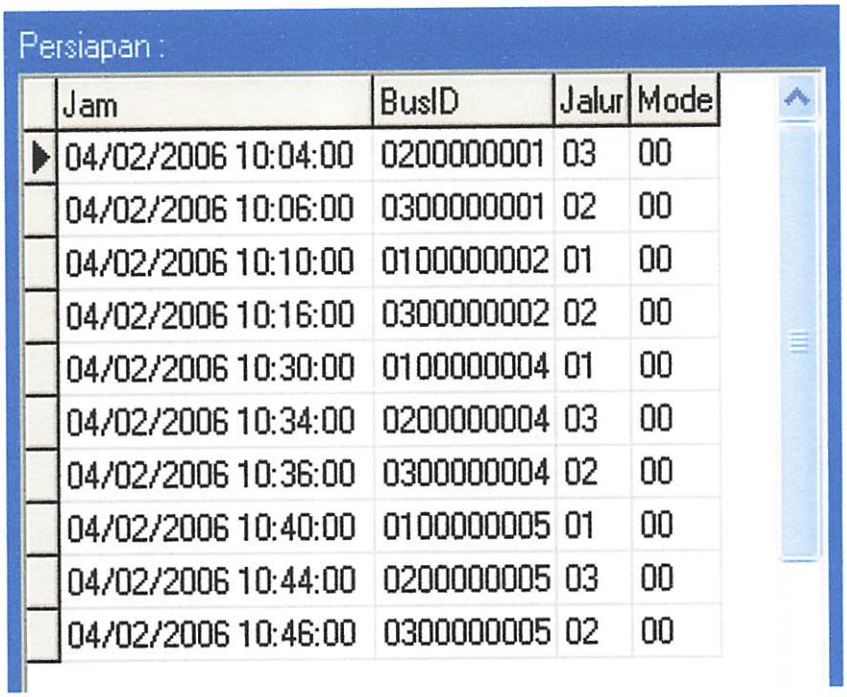
Lebih jelas tentang isi dari tabel persiapan ini dapat dilihat dari gambar dibawah ini :

Gambar 4.12 Gambar Tampilan Tabel Keberangkatan

Sedangkan tabel kedua yaitu tabel persiapan dalam tabel ini dimasukkan data-data bis yang telah siap untuk diberangkatkan dan telah masuk dalam daftar antrian. Dalam tabel ini terdapat terdapat 4 kolom yaitu :

- Jam keberangkatan
- ID bis
- Jalur keberangkatan
- Mode, untuk mode disini dibagi menjadi 4 mode yaitu
  - Mode 0 berarti bis telah memasukkan ID nya dan telah dimasukkan dalam daftar antrian.
  - Mode 1 berarti bis telah dipanggil untuk persiapan di jalur yang telah ditentukan.
  - Mode 2 berarti menandakan bahwa bis telah berada di jalur.
  - Mode 3 berarti bis telah diberangkatkan.

Lebih jelas tentang isi dari tabel persiapan ini dapat dilihat dari gambar dibawah ini :


Persiapan :

Jam	BusID	Jalur	Mode
04/02/2006 10:04:00	0200000001	03	00
04/02/2006 10:06:00	0300000001	02	00
04/02/2006 10:10:00	0100000002	01	00
04/02/2006 10:16:00	0300000002	02	00
04/02/2006 10:30:00	0100000004	01	00
04/02/2006 10:34:00	0200000004	03	00
04/02/2006 10:36:00	0300000004	02	00
04/02/2006 10:40:00	0100000005	01	00
04/02/2006 10:44:00	0200000005	03	00
04/02/2006 10:46:00	0300000005	02	00

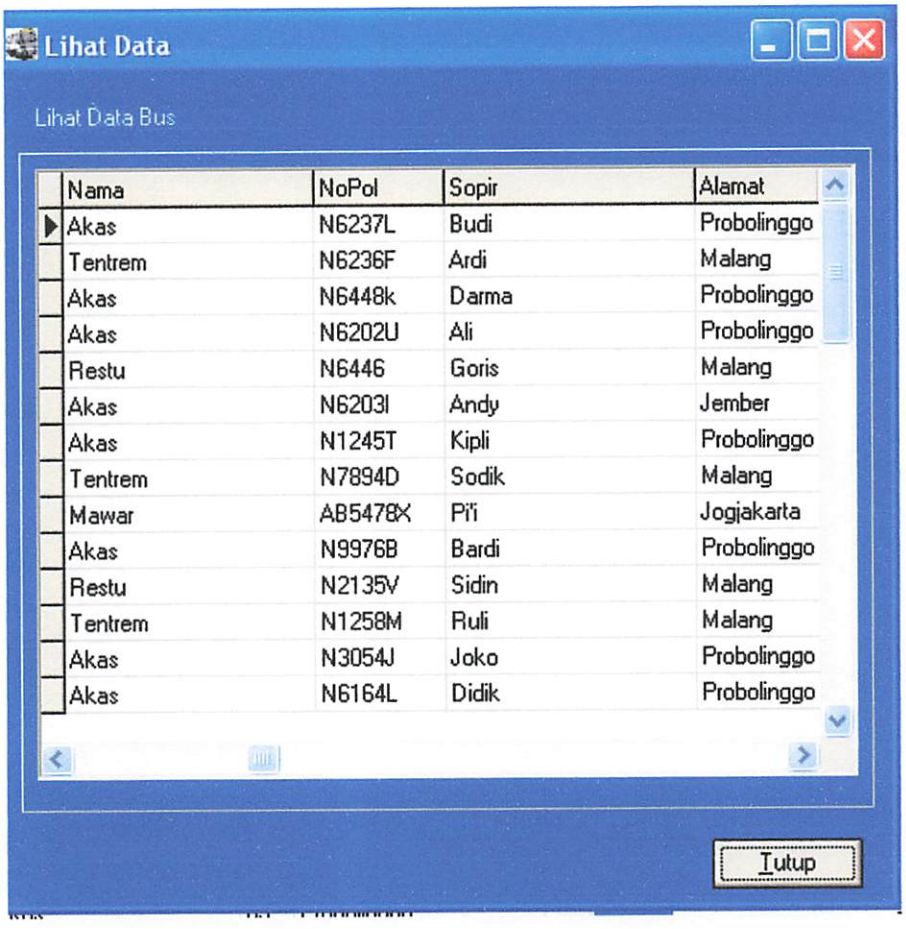
Gambar 4.13 Tampilan Tabel Persiapan

Pada menu utama pengatur keberangkatan bis ini terdapat pula dua *Button* yaitu :

- Lihat data bis

Gambar 4.14 *Button* Lihat Data Bis

Apabila kita mengklik button ini maka akan tampil data bis lengkap seperti gambar dibawah ini :



Gambar 4.15 Tabel Data Bis

- Mulai

Gambar 4.16 Gambar *Button* Mulai

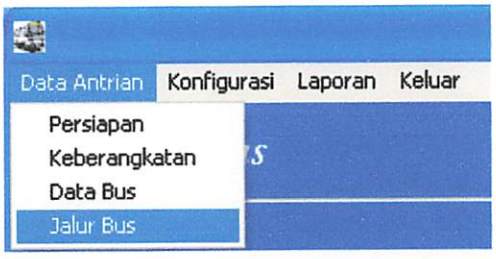
Pada saat kita mengklik *Button* mulai ini maka program pengatur keberangkatan bus ini akan dijalankan.

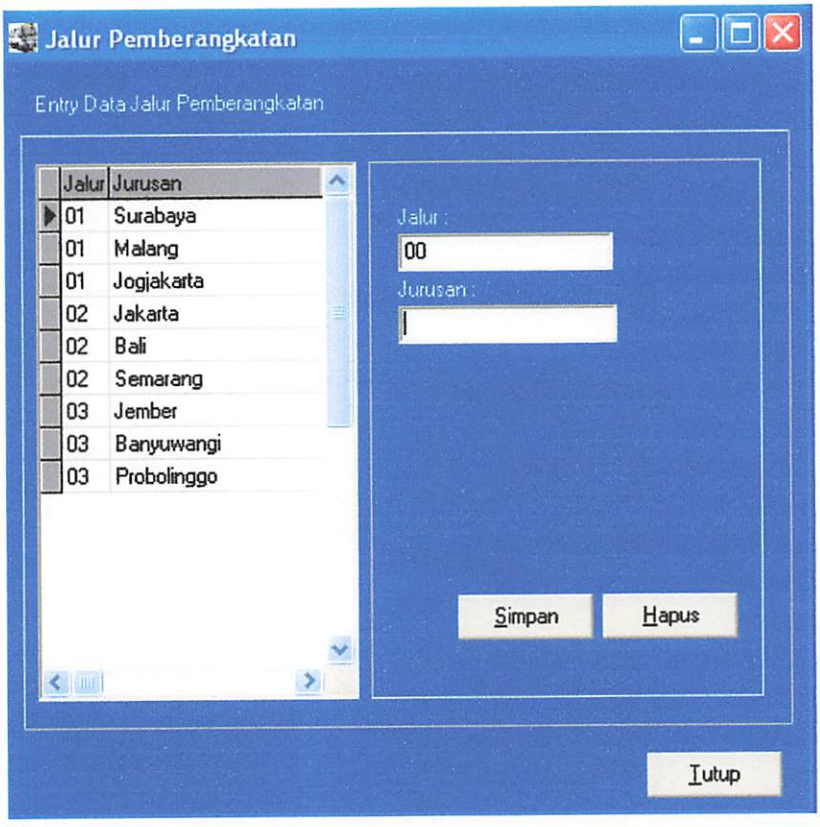
4.4. Langkah Langkah Menjalankan Program Pengatur Keberangkatan Bis

1. Pertama-tama kita harus memastikan keseluruhan perangkat keras telah terhubung dengan benar dan dialiri catu daya 12 V
2. Menghubungkan perangkat keras dengan komputer melalui *Port Serial 1*.
3. Mengklik *Button* mulai yang terdapat pada menu utama.
4. Mengatur konfigurasi *Port Serial* dengan cara mengklik tulisan *konfigurasi* yang terdapat pada bagian atas tampilan utama kemudian akan tampil kotak *konfigurasi komunikasi serial*. Lebih jelasnya dapat dilihat pada gambar dibawah ini :

Gambar 4.17 Gambar Tampilan *Konfigurasi Komunikasi Serial*

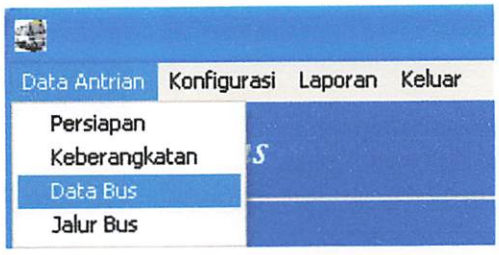
5. Untuk mendaftarkan jalur dan jurusan kita dapat masuk di data antrian dan klik di jalur bis, lebih jelasnya tampak pada gambar dibawah ini :

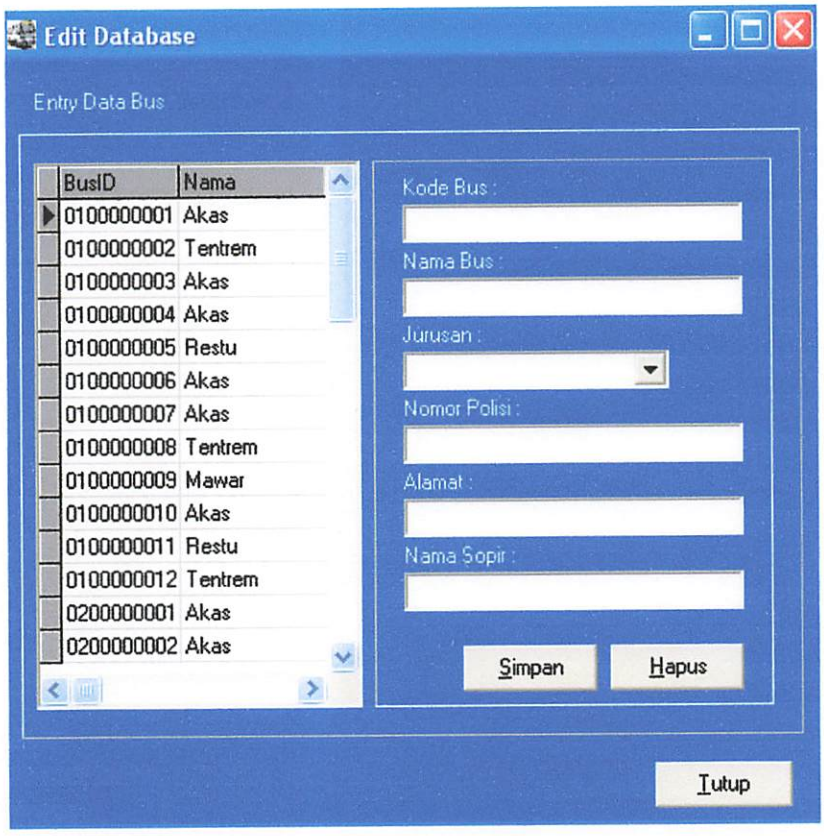




Gambar 4.18 *Form* Untuk Mengisikan Daftar Jalur Dan Jurusan

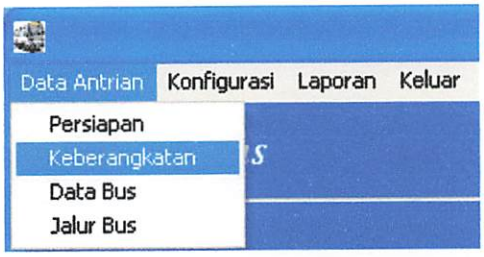
6. Untuk mengisi data bis maka kita dapat membuka data antrian dan klik data bus maka akan tampil kotak untuk memasukkan data bus. Lebih jelasnya mengenai pengisian data bus dapat dilihat di gambar dibawah ini :

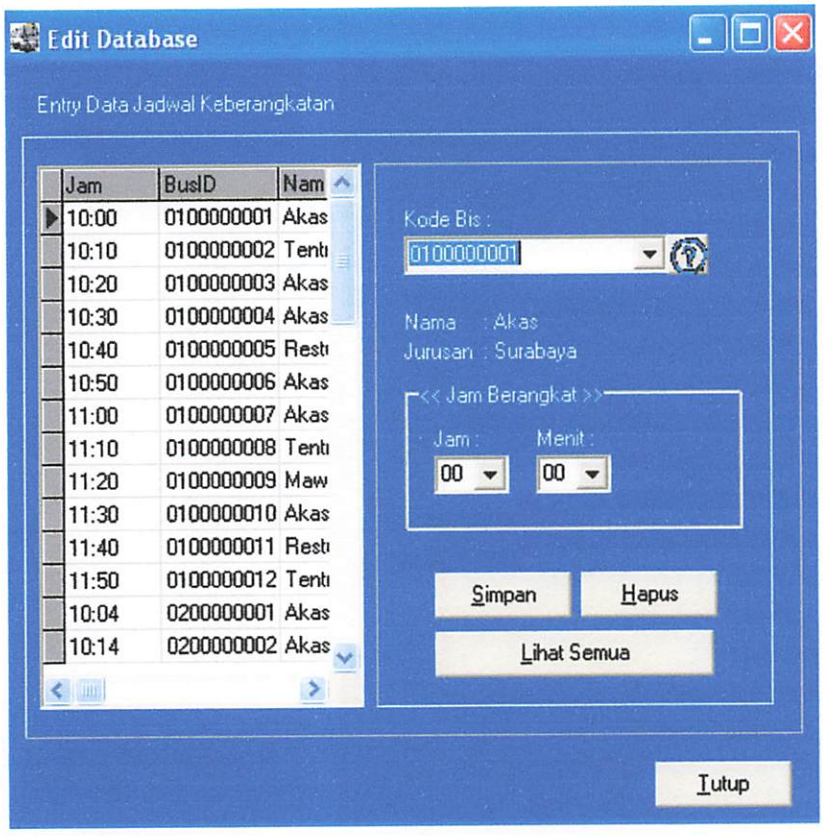




Gambar 4.19 Gambar *Form* Untuk Mengisikan Data Bus

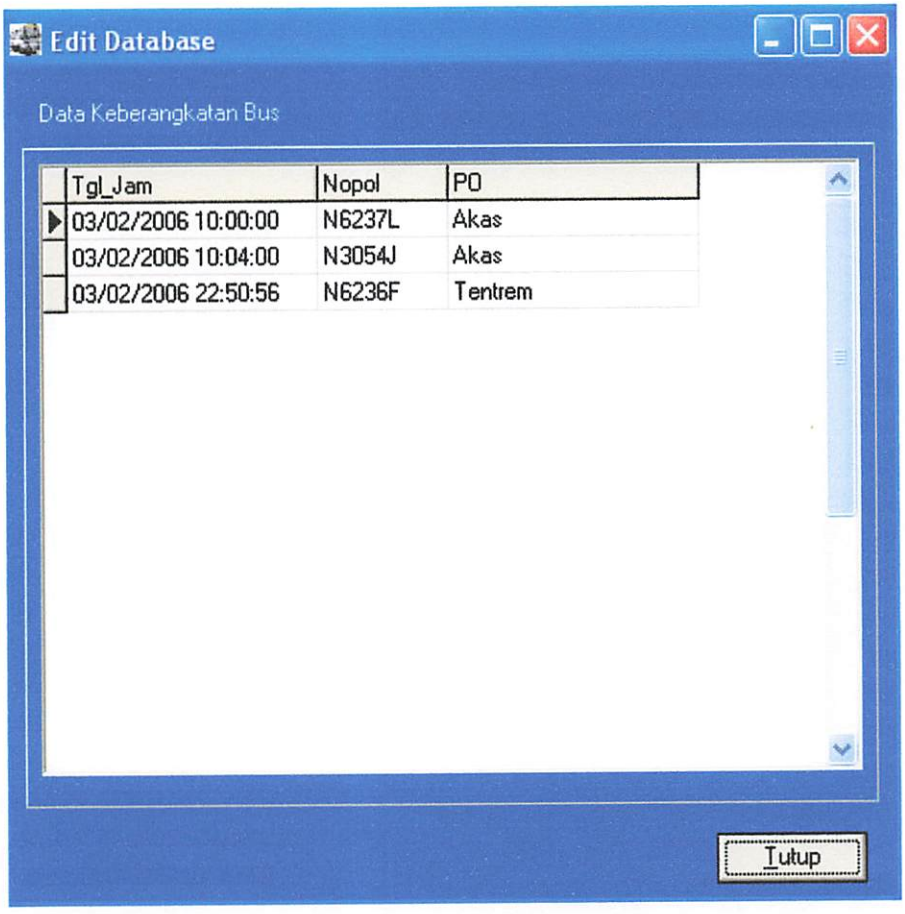
7. Langkah selanjutnya yaitu mengisikan jam keberangkatan bis yaitu dengan masuk di data antrian kemudian klik di keberangkatan, kita tinggal memasukkan *ID* bis dan menentukan jam keberangkatannya. Lebih jelasnya mengenai cara memasukkan jam keberangkatan ini dapat dilihat pada gambar dibawah ini :





Gambar 4.20 *Form* Untuk Mengisikan Jam Keberangkatan

8. Setelah komukasi antara perangkat keras dan computer telah berjalan maka kita dapat mulai mengirimkan *ID* bis melalui *keypad*. *ID* yang dikirimkan apabila datanya cocok dengan yang ada di data *base* computer maka akan langsung dimasukkan ke daftar antrian dan ditampilkan di tabel persiapan dan dengan mode 0. 10 menit sebelum keberangkatan bus yang telah masuk daftar antrian akan dipanggil untuk masuk jalur maka *mode*nya akan menjadi *mode 1*, apabila bis telah berada di jalur maka *mode*nya akan menjadi *mode 2*, tepat pada jam keberangkatannya maka bis akan diberangkatkan.
9. Setelah bis diberangkatkan maka data bis tadi akan dimasukkan kedalam data rekap keberangkatan. Untuk melihat data rekap keberangkatan kita dapat melihatnya di laporan kemudian klik di lihat rekap pemberangkatan. Untuk lebih jelasnya dapat kita lihat di gambar dibawah ini :



Gambar 4.21 Gambar Laporan Rekap Keberangkatan Bis

BAB V
PENUTUP

5.1. Kesimpulan

1. a. Pada perencanaan penggunaan *port-port* pada mikrokontroller, *port* 0 dalam perancangan alat ini tidak digunakan, *port* 1 digunakan untuk keluaran ke *LCD*, *port* 2 dihubungkan ke *Keypad*, sedangkan *Port* 3 yang digunakan hanya pin 3.0 dan 3.1 yang digunakan untuk komunikasi dengan komputer.
   b. Pada perencanaan rangkaian *Laserdioda* ini digunakan *Resistor* sebesar 10 Ω.
   c. Pada perencanaan rangkaian *Photodioda* dari hasil perhitungan didapatkan nilai R1 dan R2 sebesar 5K4 dan 74,07K karena dipasaran tidak ada nilai *resistor* sebesar ini maka diganti dengan nilai yang terdekat yaitu *resistor* dengan nilai 4K7 dan 68K.
2. a. Pada pengujian mikrokontroller didapatkan hasil bahwa mikrokontroller berjalan sesuai dengan yang direncanakan yaitu mengambil masukan data dari *keypad* melalui *port* 2 kemudian menampilkannya di *LCD* melalui *port* 1, dan mengirimkan data ke komputer melalui *port* 3.
   b. Pada pengujian rangkaian sensor *laserdioda* terdapat adanya *error* yaitu $I_{Laserdioda}$ sebesar 1,06% dan $V_{Laserdioda}$ sebesar 2,4%.
   c. Pada pengujian rangkaian *photodioda* terdapat adanya *error* yaitu $V_{photodioda}$ sebesar 3,9%.

5.2. Saran

1. Dengan segala keterbatasan yang dimiliki penulis maka, terbatasnya jangkauan operasional alat ini, seperti alat hanya mampu memakai 3 jalur saja, sebaiknya alat ini dapat dikembangkan lagi.
2. Hendaknya alat ini dipergunakan sebagaimana fungsinya yaitu mengatur keberangkatan bis agar supaya dapat berguna bagi masyarakat luas.
3. Sebaiknya alat diterapkan dan digunakan sesuai dengan prosedur dan ketentuan yang telah disebutkan.

BAB V

PENUTUP

5.1. Kesimpulan

1. a. Pada percobaan pengujian port-port pada mikrokontroller port 0 dalam pencahayaan alat ini tidak digunakan. port 1 digunakan untuk keluaran ke LCD. port 2 dihubungkan ke Keypad. sedangkan Port 3 yang digunakan hanya pin 3.0 dan 3.1 yang digunakan untuk komunikasi dengan komputer.

   b. Pada percobaan rangkaian Laservilode ini digunakan Resistor sebesar 10 Ω.

   c. Pada percobaan rangkaian Photodiode dan basil pertimbangan didapatkan nilai R1 dan R2 sebesar 5K4 dan 74.07K karena dipasaran tidak ada nilai resistor sebesar ini maka diganti dengan nilai yang terdekat yaitu resistor dengan nilai 4K7 dan 68K.

2. a. Pada pengujian mikrokontroller didapatkan basil bahwa mikrokontroller berjalan sesuai dengan yang diprogramkan yaitu mengambil masukan data dari Keypad melalui port 2 kemudian memampilkannya di LCD melalui port 1. dan mengirimkan data ke komputer melalui port 3.

   b. Pada pengujian rangkaian sensor Laservilode terdapat adanya error yaitu $L_{aservilode}$ sebesar 1.06% dan $V_{aservilode}$ sebesar 2.4%.

   c. Pada pengujian rangkaian photodiode terdapat adanya error yaitu $V_{photodiode}$ sebesar 3.9%.

5.2. Saran

1. Dengan segala keterbatasan yang dimiliki penulis maka. terbatasnya jangkauan operasional alat ini. sepert alat hanya mampu memakai 3 jalur saja, sebaiknya alat ini dapat dikembangkan lagi.

2. Hendaknya alat ini dipergunakan sebagaimana fungsinya yaitu mengatur kebersihkan bis agar supaya dapat berguna bagi masyarakat luas.

3. Sebaiknya alat diterapkan dan digunakan secara dengan prosedur dan ketentuan yang telah disebutkan.

DAFTAR PUSTAKA

[1] ATMEL, Data Sheet Book

[2] Dallas Semiconductor MAXIM MAX232E Data Sheet

[3] Dinas Perhubungan Kabupaten Sumenep

[4] Hafindo Electronic & Education, "*Modul Pelatihan Mikrokontroller MCS51*" Malang, 2005.

[5] *http://www.atmel.com*

[6] *http://www.wcs.com*

[7] *http://www.howstuffworks.com*

[8] Ibnu M.& Anistardi, "*Bereksperimen Dengan Mikrokontroller 8031*" Cetakan Pertama, Penerbit PT Elex Media Komputindo, Jakarta,1997.

LAMPIRAN

INSTITUT TEKNOLOGI NASIONAL
JL. BENDUNGAN SIGURA-GURA 2
MALANG

Lampiran : 1 (satu) berkas
Pembimbing Skripsi

Kepada : Yth. **Ir. Yusuf Ismail Nakhoda, MT**
Dosen Institut Teknologi Nasional
Malang

Yang bertanda tangan di bawah ini :

Nama	: A. Kadir
Nim	: 0117138
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Elektronika

Dengan ini mengajukan permohonan, kiranya Bapak/Ibu bersedia menjadi Dosen Pembimbing Utama / ~~Pendamping~~ dari (1), 2 orang pembimbing *), untuk penyusunan Skripsi dengan judul (Proposl terlampir) :

Perancangan dan pembuatan alat pengatur keberangkatan bus secara otomatis diterminal menggunakan Mikrokontroller dan PC (Personal Computer)

Adapun tugas tersebut sebagai salah satu syarat untuk menempuh Ujian Akhir Sarjana Teknik.
Demikian permohonan kami dan atas kesediaan Bapak/Ibu kami ucapkan terima kasih.

Malang, 18-Jun-2005

Mengetahui	**Hormat kami**
Ketua Jurusan T. Elektro S-1	
Ir. F. Yudi Limpraptono,MT	**A. Kadir**
NIP. Y. 1039500274	**0117138**

*) coret yang tidak perlu

Form S-3a

INSTITUT TEKNOLOGI NASIONAL
JL. BENDUNGAN SIGURA-GURA 2
MALANG

PERNYATAAN KESEDIAAN DALAM PEMBIMBINGAN SKRIPSI

Sesuai permohonan dari mahasiswa/i :

Nama : A. Kadir

Nim : 0117138

Semester : VIII

Jurusan : Teknik Elektro S-1

Konsentrasi : Teknik Elektronika

Dengan ini menyatakan bersedia/tidak bersedia *) membimbing skripsi dari mahasiswa tersebut. dengan judul :

Perancangan dan pembuatan alat pengatur keberangkatan bus secara otomatis diterminal menggunakan Mikrokontroller dan PC (Personal Computer)

Demikian surat pernyataan ini kami buat agar dapat digunakan seperlunya.

Malang, 18-Jun-2005

Hormat kami

Ir. Yusuf Ismail Nakhoda, M.T

NIP. P 10188 00789

Catatan:
Setelah disetujui agar formulir ini
Diserahkan mahasiswa/I yang bersangkutan
Kepada jurusan untuk diproses lebih lanjut
*) Coret yang tidak perlu

Form S-3b

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK

(PERSERO) MALANG
NK NIAGA MALANG

Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting) Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

Malang 18-Jun-2005

Nomor : ITN-773/7/TA.GJL/2005
Lampiran :
Perihal : ***Bimbingan Skripsi***

Kepada : Yth. Sdr. **Ir. Yusuf Ismail Nakhoda, MT**
Dosen Pembimbing
Jurusan Teknik Elektro S-1
di
Malang

Dengan hormat.
Sesuai dengan permohonan dan persetujuan dalam proposal skipsi untuk mahasiswa:

Nama	: A. Kadir
Nim	: 0117138
Semester	: IX
Fakultas	: Teknologi Industri
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Elektronika

Maka dengan ini pembimbingan tersebut kami serahkan sepenuhnya kepada Saudara/i selama masa waktu 6 (enam) bulan, terhitung mulai tanggal:

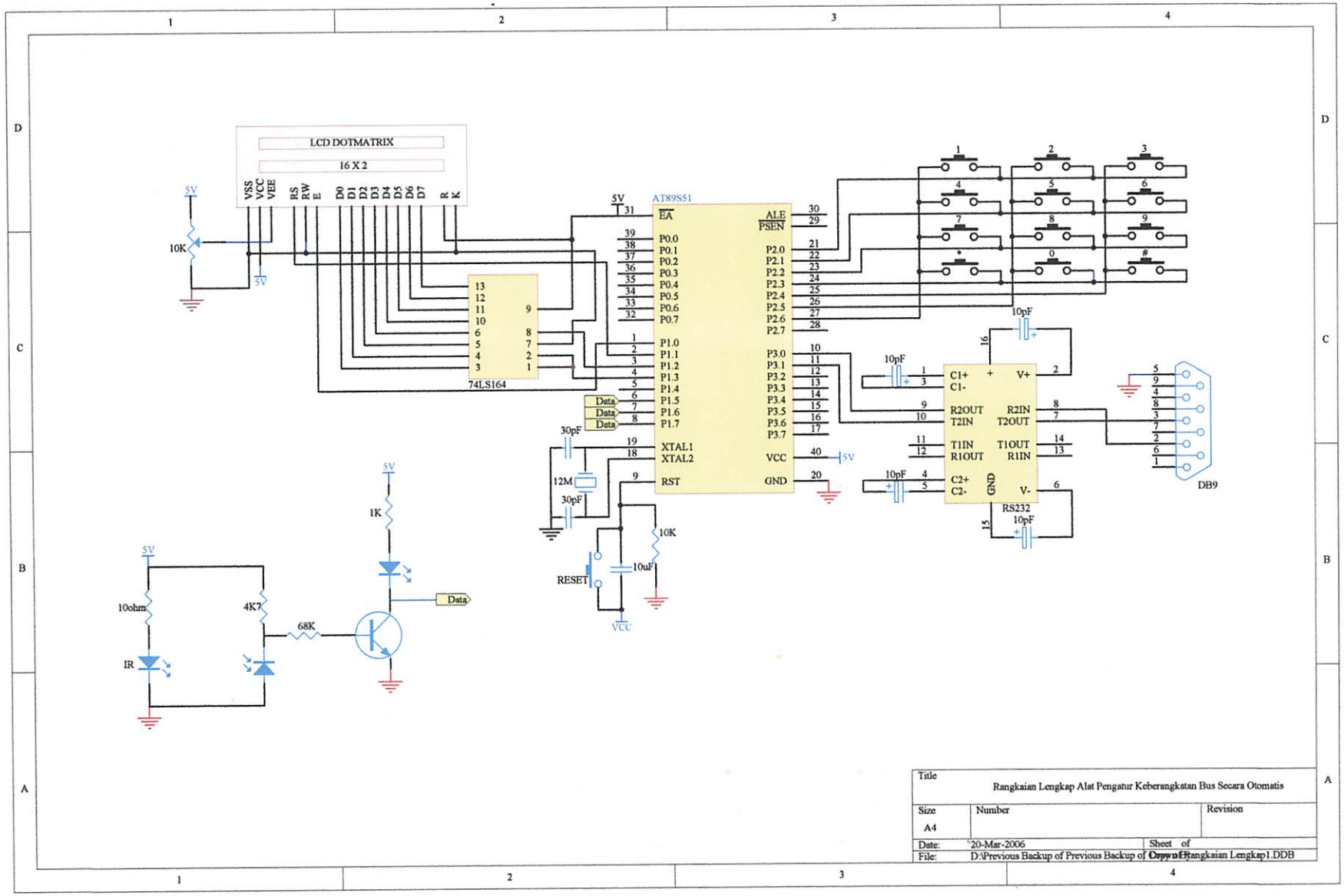
15-Sep-2005 s/d 18-Mar-2006

Sebagai satu syarat untuk menempuh Ujian Akhir Sarjana.
Demikian agar maklum, atas perhatian dan bantuannya kami ucapkan banyak terima kasih.

Ketua Jurusan
Teknik Elektro S-1

Ir. F. Yudi Limpraptono,MT
NIP. P. 1039500274

Form S-4a



Title	Rangkaian Lengkap Alat Pengatur Keberangkatan Bus Secara Otomatis	
Size A4	Number	Revision
Date: 20-Mar-2006	Sheet of	
File: D:\Previous Backup of Previous Backup of Copy of Rangkaian Lengkap1.DDB		

INSTITUT TEKNOLOGI NASIONAL
JL. BENDUNGAN SIGURA-GURA 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : A. Kadir
Nim : 0117138
Masa Bimbingan : 15-Sep-2005 s/d 18-Mar-2006
Judul Skripsi : Perancangan dan pembuatan alat pengatur keberangkatan bus secara otomatis di terminal menggunkan mikrokontroller dan PC

NO	Tanggal	Uraian	Paraf Pembimbing
1.	01-02-2006	Revisi penulisan pada Bab I	
2.	03-02-2006	Revisi penulisan letak judul tabel 2.3 pada Bab II	
3.	08-02-2006	Revisi gambar dan lebar *paragraph* penulisan pada tabel 3.3 Bab III	
4.	11-02-2006	Revisi hasil *error* pengujian alat pada Bab IV	
5.	15-02-2006	Revisi isi kesimpulan dan saran pada Bab V	
6.	19-02-2006	Revisi lebar *paragraph* penulisan Pada daftar pustaka	
7.	22-02-2006	Revisi makalah seminar hasil skripsi	
8.	14- 03-2006	ACC makalah seminar hasil Skripsi	
9.	19-03-2006	ACC Laporan Skripsi	

Malang, 17 Maret 2006
Dosen Pembimbing

Ir. Yusuf Ismail Nakhoda, MT

Form S-4a

DATA SHEET

ures

- ıpatible with MCS-51® Products
- ytes of In-System Programmable (ISP) Flash Memory
- Endurance: 1000 Write/Erase Cycles
- to 5.5V Operating Range
- y Static Operation: 0 Hz to 33 MHz
- ıe-level Program Memory Lock
- x 8-bit Internal RAM
- rogrammable I/O Lines
- 16-bit Timer/Counters
- Interrupt Sources
- Duplex UART Serial Channel
- -power Idle and Power-down Modes
- rrupt Recovery from Power-down Mode
- chdog Timer
- l Data Pointer
- er-off Flag
- t Programming Time
- ible ISP Programming (Byte and Page Mode)

cription

T89S51 is a low-power, high-performance CMOS 8-bit microcontroller with 4K of in-system programmable Flash memory. The device is manufactured using 's high-density nonvolatile memory technology and is compatible with the indus- ındard 80C51 instruction set and pinout. The on-chip Flash allows the program ory to be reprogrammed in-system or by a conventional nonvolatile memory pro- ner. By combining a versatile 8-bit CPU with in-system programmable Flash on a lithic chip, the Atmel AT89S51 is a powerful microcontroller which provides a -flexible and cost-effective solution to many embedded control applications.

T89S51 provides the following standard features: 4K bytes of Flash, 128 bytes of 32 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a five- r two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and circuitry. In addition, the AT89S51 is designed with static logic for operation to zero frequency and supports two software selectable power saving modes. dle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and ıpt system to continue functioning. The Power-down mode saves the RAM con- but freezes the oscillator, disabling all other chip functions until the next external ıpt or hardware reset.

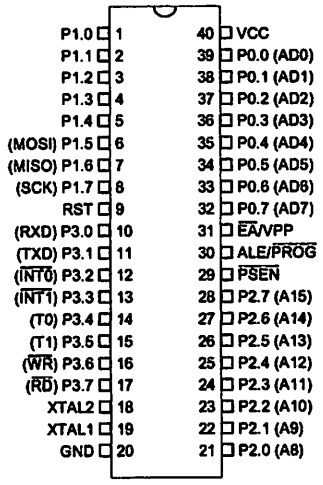
8-bit Microcontroller with 4K Bytes In-System Programmable Flash

AT89S51

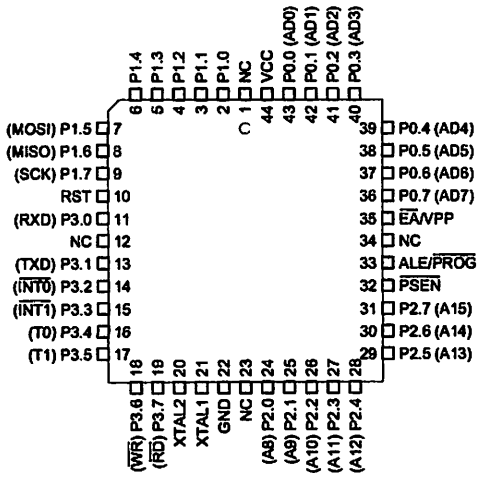
Rev. 2487A–10/01

Configurations

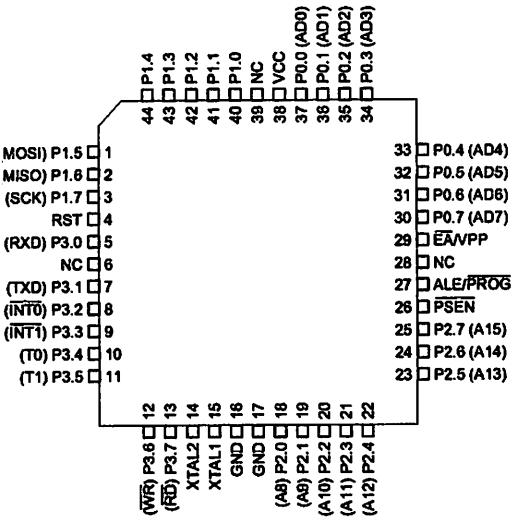
PDIP



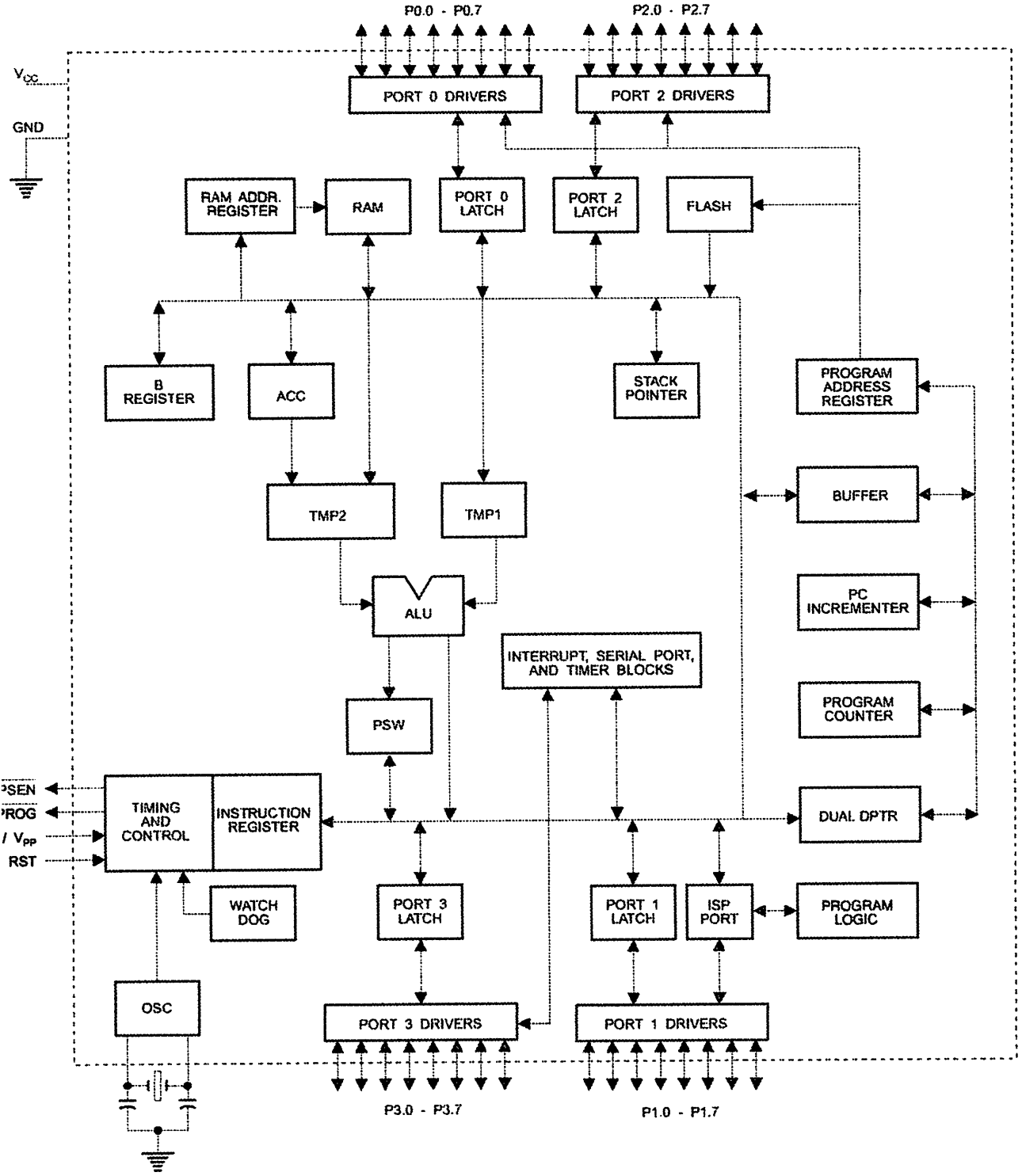
PLCC



TQFP



k Diagram


P0.0 - P0.7
P2.0 - P2.7
V_CC
GND
PORT 0 DRIVERS
PORT 2 DRIVERS
RAM ADDR. REGISTER
RAM
PORT 0 LATCH
PORT 2 LATCH
FLASH
B REGISTER
ACC
STACK POINTER
PROGRAM ADDRESS REGISTER
BUFFER
TMP2
TMP1
ALU
PC INCREMENTER
INTERRUPT, SERIAL PORT, AND TIMER BLOCKS
PSW
PROGRAM COUNTER
PSEN
PROG
/ V_PP
RST
TIMING AND CONTROL
INSTRUCTION REGISTER
DUAL DPTR
WATCH DOG
PORT 3 LATCH
PORT 1 LATCH
ISP PORT
PROGRAM LOGIC
OSC
PORT 3 DRIVERS
PORT 1 DRIVERS
P3.0 - P3.7
P1.0 - P1.7

Description

Supply voltage.

Ground.

0

Port 0 is an 8-bit open drain bidirectional 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 low-order address/data bus during accesses to external program and data memory. In this mode, P0 has internal pull-ups.

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. **External pull-ups are required during program verification.**

1

Port 1 is an 8-bit bidirectional I/O port with internal pull-ups. 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 pull-ups 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 pull-ups.

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

Port Pin	Alternate Functions
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

2

Port 2 is an 8-bit bidirectional I/O port with internal pull-ups. 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 pull-ups 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 pull-ups.

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits 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.

3

Port 3 is an 8-bit bidirectional I/O port with internal pull-ups. 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 pull-ups 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 pull-ups.

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

Port 3 also serves the functions of various special features of the AT89S51, as shown in the following table.

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)

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

/PROG Address Latch Enable (ALE) 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.

N Program Store Enable ($\overline{PSEN}$) is the read strobe to external program memory.

When the AT89S51 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.

/PP External Access Enable. $\overline{EA}$ must be strapped to GND in order to enable the device to fetch code from external program memory locations starting at 0000H up to FFFFH. Note, however, that if lock bit 1 is programmed, $\overline{EA}$ will be internally 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.

L1 Input to the inverting oscillator amplifier and input to the internal clock operating circuit.

L2 Output from the inverting oscillator amplifier

cial ction isters

A 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.

1. AT89S51 SFR Map and Reset Values

H								0FFH
H	B 00000000							0F7H
H								0EFH
H	ACC 00000000							0E7H
H								0DFH
H	PSW 00000000							0D7H
H								0CFH
H								0C7H
H	IP XX000000							0BFH
H	P3 11111111							0B7H
H	IE 0X000000							0AFH
H	P2 11111111		AUXR1 XXXXXXX0				WDTRST XXXXXXXX		0A7H
H	SCON 00000000	SBUF XXXXXXXX							9FH
H	P1 11111111								97H
H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0		8FH
H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

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.

Interrupt Registers: The individual interrupt enable bits are in the IE register. Two priorities can be set for each of the five interrupt sources in the IP register.

Table 2. AUXR: Auxiliary Register

AUXR Address = 8EH Reset Value = XXX00XX0B

Not Bit Addressable

	–	–	–	WDIDLE	DISRTO	–	–	DISALE
Bit	7	6	5	4	3	2	1	0

–	Reserved for future expansion
DISALE	Disable/Enable ALE
	DISALE Operating Mode
	0 ALE is emitted at a constant rate of 1/6 the oscillator frequency
	1 ALE is active only during a MOVX or MOVC instruction
DISRTO	Disable/Enable Reset out
	DISRTO
	0 Reset pin is driven High after WDT times out
	1 Reset pin is input only
WDIDLE	Disable/Enable WDT in IDLE mode
WDIDLE	
0	WDT continues to count in IDLE mode
1	WDT halts counting in IDLE mode

Dual Data Pointer Registers: To facilitate accessing both internal 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 AUXR1 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 rest under software control and is not affected by reset.

Table 3. AUXR1: Auxiliary Register 1

AUXR1

Address = A2H

Reset Value = XXXXXXX0B

Not Bit Addressable

	–	–	–	–	–	–	–	DPS
Bit	7	6	5	4	3	2	1	0

– Reserved for future expansion

DPS Data Pointer Register Select

DPS

0 Selects DPTR Registers DP0L, DP0H

1 Selects DPTR Registers DP1L, DP1H

ıory anization

MCS-51 devices have a separate address space for Program and Data Memory. Up to 64K bytes each of external Program and Data Memory can be addressed.

ram Memory

If the $\overline{EA}$ pin is connected to GND, all program fetches are directed to external memory.

On the AT89S51, if $\overline{EA}$ is connected to V_{CC}, program fetches to addresses 0000H through FFFH are directed to internal memory and fetches to addresses 1000H through FFFFH are directed to external memory.

Memory

The AT89S51 implements 128 bytes of on-chip RAM. The 128 bytes are accessible via direct and indirect addressing modes. Stack operations are examples of indirect addressing, so the 128 bytes of data RAM are available as stack space.

chdog er e-time bled with et-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 14-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running. The WDT timeout period is dependent on the external clock frequency. There is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST pin.

ıg the WDT

To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, the user needs to service it by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH), and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycles. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is 98xTOSC, where TOSC=1/FOSC. To make the best use of the WDT, it

should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

r During er-down Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt, which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S51 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

To ensure that the WDT does not overflow within a few states of exiting Power-down, it is best to reset the WDT just before entering Power-down mode.

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S51 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode.

With WDIDLE bit enabled, the WDT will stop to count in IDLE mode and resumes the count upon exit from IDLE.

RT

The UART in the AT89S51 operates the same way as the UART in the AT89C51. For further information on the UART operation, refer to the ATMEL Web site (http://www.atmel.com). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

er 0 and 1

Timer 0 and Timer 1 in the AT89S51 operate the same way as Timer 0 and Timer 1 in the AT89C51. For further information on the timers' operation, refer to the ATMEL Web site (http://www.atmel.com). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

rrupts

The AT89S51 has a total of five interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), two timer interrupts (Timers 0 and 1), and the serial port interrupt. These interrupts are all shown in Figure 1.

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

Note that Table 4 shows that bit position IE.6 is unimplemented. In the AT89S51, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products.

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle

Table 4. Interrupt Enable (IE) Register

(MSB) (LSB)

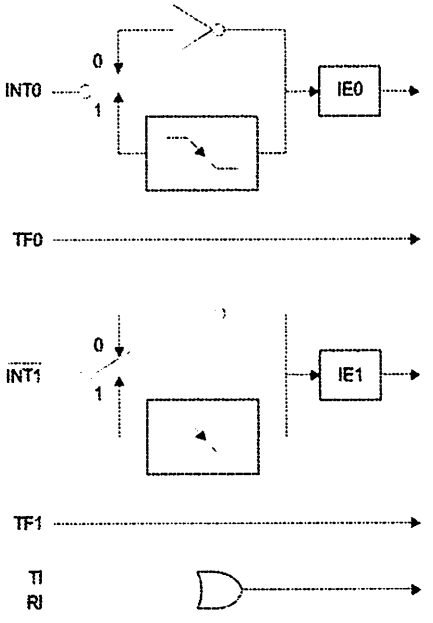
EA	–	–	ES	ET1	EX1	ET0	EX0

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
–	IE.5	Reserved
ES	IE.4	Serial Port 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 reserved bits, because they may be used in future AT89 products.		

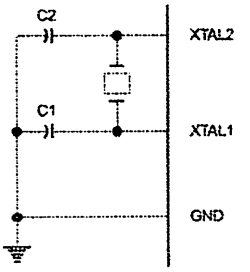
Figure 1. Interrupt Sources



illator racteristics

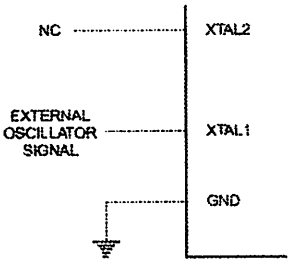
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 3. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 2. Oscillator Connections



Note: C1, C2 = 30 pF ±10 pF for Crystals = 40 pF ±10 pF for Ceramic Resonators

Figure 3. External Clock Drive Configuration



Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special function registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

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

er-down le

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by activation of an enabled external interrupt into $\overline{INT0}$ or $\overline{INT1}$. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Table 5. Status of External Pins During Idle and Power-down Modes

Mode	Program Memory	ALE	$\overline{PSEN}$	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

ıram ıory Lock

The AT89S51 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the following table.

Table 6. Lock Bit Protection Modes

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No program lock features
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further programming of the Flash memory is disabled
3	P	P	U	Same as mode 2, but verify is also disabled
4	P	P	P	Same as mode 3, but external execution is also disabled

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value and holds that value until reset is activated. The latched value of $\overline{EA}$ must agree with the current logic level at that pin in order for the device to function properly.

ıramming Flash – ıllel Mode

The AT89S51 is shipped with the on-chip Flash memory array ready to be programmed. The programming interface needs a high-voltage (12-volt) program enable signal and is compatible with conventional third-party Flash or EPROM programmers.

The AT89S51 code memory array is programmed byte-by-byte.

Programming Algorithm: Before programming the AT89S51, the address, data, and control signals should be set up according to the Flash programming mode table and Figures 13 and 14. To program the AT89S51, take the following steps:

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12V.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 50 µs. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

$\overline{Data}$ Polling: The AT89S51 features $\overline{Data}$ Polling to indicate the end of a byte write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P0.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{Data}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The progress of byte programming can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. P3.0 is pulled low after ALE goes high during programming to indicate $\overline{\text{BUSY}}$. P3.0 is pulled high again when programming is done to indicate READY.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed code data can be read back via the address and data lines for verification. The status of the individual lock bits can be verified directly by reading them back.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 000H, 100H, and 200H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows.

(000H) = 1EH indicates manufactured by Atmel
(100H) = 51H indicates 89S51
(200H) = 06H

Chip Erase: In the parallel programming mode, a chip erase operation is initiated by using the proper combination of control signals and by pulsing ALE/$\overline{\text{PROG}}$ low for a duration of 200 ns - 500 ns.

In the serial programming mode, a chip erase operation is initiated by issuing the Chip Erase instruction. In this mode, chip erase is self-timed and takes about 500 ms.

During chip erase, a serial read from any address location will return 00H at the data output.

ɟramming Flash – al Mode

The Code memory array can be programmed using the serial ISP interface while RST is pulled to V_{CC}. The serial interface consists of pins SCK, MOSI (input) and MISO (output). After RST is set high, the Programming Enable instruction needs to be executed first before other operations can be executed. Before a reprogramming sequence can occur, a Chip Erase operation is required.

The Chip Erase operation turns the content of every memory location in the Code array into FFH.

Either an external system clock can be supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2. The maximum serial clock (SCK) frequency should be less than 1/16 of the crystal frequency. With a 33 MHz oscillator clock, the maximum SCK frequency is 2 MHz.

ıl ıramming ıithm

To program and verify the AT89S51 in the serial programming mode, the following sequence is recommended:

1. Power-up sequence:
   Apply power between VCC and GND pins.
   Set RST pin to "H".
   If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.
2. Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less than the CPU clock at XTAL1 divided by 16.
3. The Code array is programmed one byte at a time in either the Byte or Page mode. The write cycle is self-timed and typically takes less than 0.5 ms at 5V.
4. Any memory location can be verified by using the Read instruction that returns the content at the selected address at serial output MISO/P1.6.
5. At the end of a programming session, RST can be set low to commence normal device operation.

Power-off sequence (if needed):

Set XTAL1 to "L" (if a crystal is not used).

Set RST to "L".

Turn V_{CC} power off.

$\overline{\text{Data}}$ Polling: The $\overline{\text{Data}}$ Polling feature is also available in the serial mode. In this mode, during a write cycle an attempted read of the last byte written will result in the complement of the MSB of the serial output byte on MISO.

ai
ramming
ruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 8 on page 18.

gramming
rface –
allel Mode

Every code byte in the Flash array can be programmed by using the appropriate combination of control signals. The write operation cycle is self-timed and once initiated, will automatically time itself to completion.

All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

7. Flash Programming Modes

e	V_{CC}	RST	$\overline{\text{PSEN}}$	ALE/$\overline{\text{PROG}}$	$\overline{\text{EA}}$/V_{PP}	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.3-0 Address	P1.7-0 Address
Code Data	5V	H	L	(2)	12V	L	H	H	H	H	D_{IN}	A11-8	A7-0
Code Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A11-8	A7-0
Lock Bit 1	5V	H	L	(3)	12V	H	H	H	H	H	X	X	X
Lock Bit 2	5V	H	L	(3)	12V	H	H	H	L	L	X	X	X
Lock Bit 3	5V	H	L	(3)	12V	H	L	H	H	L	X	X	X
Lock Bits 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Erase	5V	H	L	(1)	12V	H	L	H	L	L	X	X	X
Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	0000	00H
Device ID	5V	H	L	H	H	L	L	L	L	L	51H	0001	00H
Device ID	5V	H	L	H	H	L	L	L	L	L	06H	0010	00H

1. Each $\overline{\text{PROG}}$ pulse is 200 ns - 500 ns for Chip Erase.
2. Each $\overline{\text{PROG}}$ pulse is 200 ns - 500 ns for Write Code Data.
3. Each $\overline{\text{PROG}}$ pulse is 200 ns - 500 ns for Write Lock Bits.
4. RDY/$\overline{\text{BSY}}$ signal is output on P3.0 during programming.
5. X = don't care.

Figure 4. Programming the Flash Memory (Parallel Mode)

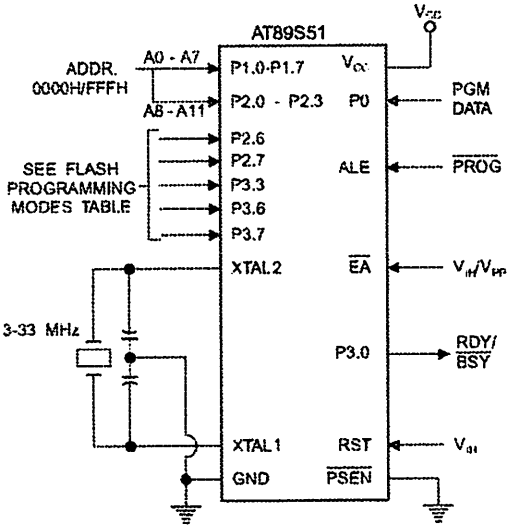
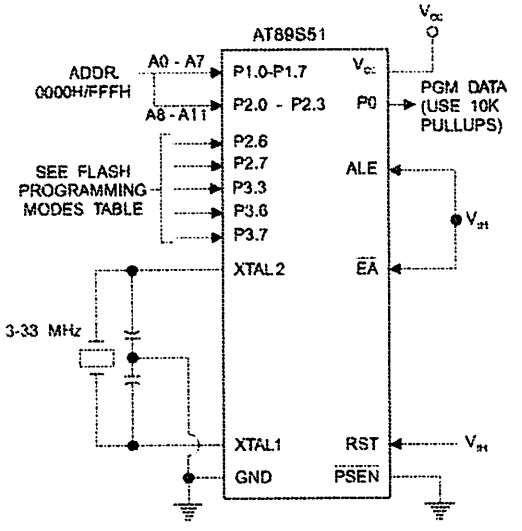


Figure 5. Verifying the Flash Memory (Parallel Mode)

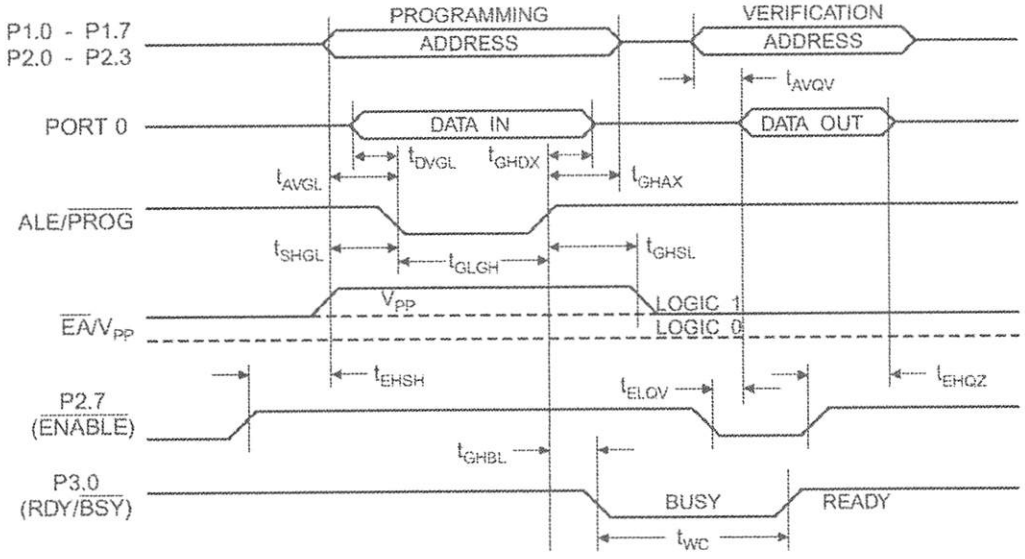


h Programming and Verification Characteristics (Parallel Mode)

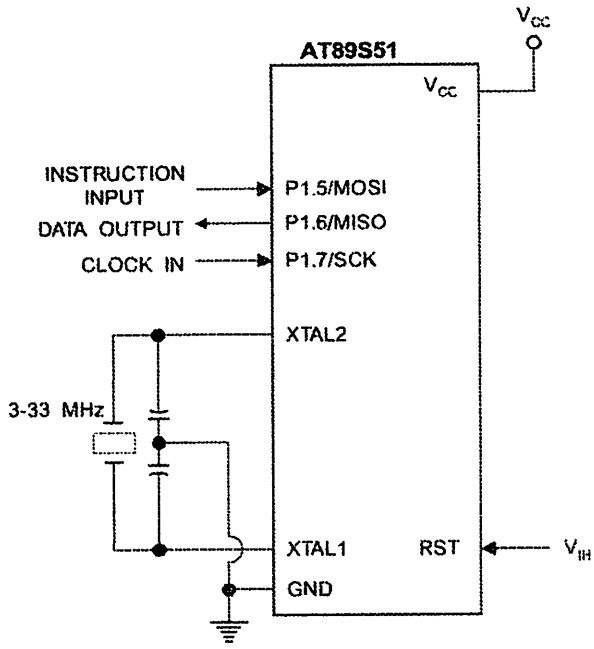
0°C to 30°C, V_{CC} = 4.5 to 5.5V

ol	Parameter	Min	Max	Units
	Programming Supply Voltage	11.5	12.5	V
	Programming Supply Current		10	mA
	V_{CC} Supply Current		30	mA
CL	Oscillator Frequency	3	33	MHz
	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
X	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
X	Data Hold After $\overline{PROG}$	$48t_{CLCL}$		
H	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
	V_{PP} Hold After $\overline{PROG}$	10		µs
H	$\overline{PROG}$ Width	0.2	1	µs
V	Address to Data Valid		$48t_{CLCL}$	
V	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
Z	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
	Byte Write Cycle Time		50	µs

re 6. Flash Programming and Verification Waveforms – Parallel Mode

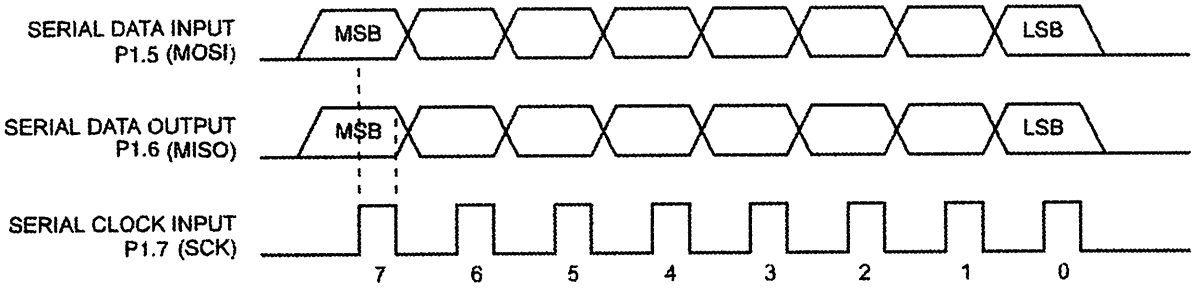


e 7. Flash Memory Serial Downloading



sh Programming and Verification Waveforms – Serial Mode

e 8. Serial Programming Waveforms



3. Serial Programming Instruction Set

ction	Instruction Format Byte 1	Byte 2	Byte 3	Byte 4	Operation
amming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output)	Enable Serial Programming while RST is high
Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
Program Memory Mode)	0010 0000	xxxx A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Read data from Program memory in the byte mode
Program Memory Mode)	0100 0000	xxxx A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Write data to Program memory in the byte mode
Lock Bits[2]	1010 1100	1110 00 B1 B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (2).
Lock Bits	0010 0100	xxxx xxxx	xxxx xxxx	xx LB3 LB2 LB1 xx	Read back current status of the lock bits (a programmed lock bit reads back as a "1")
Signature Bytes[1]	0010 1000	xxx A5 A4 A3 A2 A1	A0 xxx xxxx	Signature Byte	Read Signature Byte
Program Memory Mode)	0011 0000	xxxx A11 A10 A9 A8	Byte 0	Byte 1... Byte 255	Read data from Program memory in the Page Mode (256 bytes)
Program Memory Mode)	0101 0000	xxxx A11 A10 A9 A8	Byte 0	Byte 1... Byte 255	Write data to Program memory in the Page Mode (256 bytes)

1. The signature bytes are not readable in Lock Bit Modes 3 and 4.
2. B1 = 0, B2 = 0 → Mode 1, no lock protection
   B1 = 0, B2 = 1 → Mode 2, lock bit 1 activated
   B1 = 1, B2 = 0 → Mode 3, lock bit 2 activated
   B1 = 1, B1 = 1 → Mode 4, lock bit 3 activated

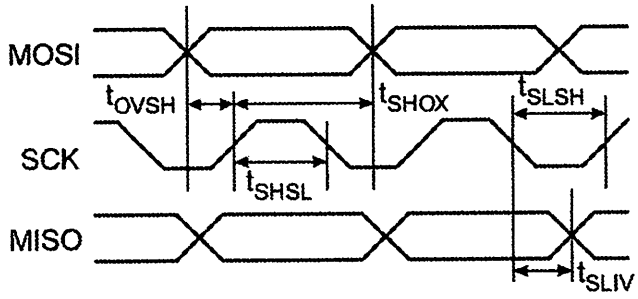
   } Each of the lock bits needs to be activated sequentially before Mode 4 can be executed.

After Reset signal is high, SCK should be low for at least 64 system clocks before it goes high to clock in the enable data bytes. No pulsing of Reset signal is necessary. SCK should be no faster than 1/16 of the system clock at XTAL1.

For Page Read/Write, the data always starts from byte 0 to 255. After the command byte and upper address byte are latched, each byte thereafter is treated as data until all 256 bytes are shifted in/out. Then the next instruction will be ready to be decoded.

ial Programming Characteristics

ıre 9. Serial Programming Timing



le 9. Serial Programming Characteristics, T_A = -40° C to 85° C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

mbol	Parameter	Min	Typ	Max	Units
CLCL	Oscillator Frequency	0		33	MHz
.CL	Oscillator Period	30			ns
ISL	SCK Pulse Width High	8 t_{CLCL}			ns
.SH	SCK Pulse Width Low	8 t_{CLCL}			ns
/SH	MOSI Setup to SCK High	t_{CLCL}			ns
IOX	MOSI Hold after SCK High	2 t_{CLCL}			ns
.IV	SCK Low to MISO Valid	10	16	32	ns
RASE	Chip Erase Instruction Cycle Time			500	ms
WC	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	µs

olute Maximum Ratings*

ating Temperature	-55°C to +125°C
ige Temperature	-65°C to +150°C
ge on Any Pin Respect to Ground	-1.0V to +7.0V
mum Operating Voltage	6.6V
Output Current	15.0 mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

Characteristics

alues shown in this table are valid for T_A = -40°C to 85°C and V_{CC} = 4.0V to 5.5V, unless otherwise noted.

bol	Parameter	Condition	Min	Max	Units
	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-650	µA
	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
T	Reset Pulldown Resistor		50	300	KΩ
	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
	Power Supply Current	Active Mode, 12 MHz		25	mA
		Idle Mode, 12 MHz		6.5	mA
	Power-down Mode[2]	V_{CC} = 5.5V		50	µA

1. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 10 mA
   Maximum I_{OL} per 8-bit port:
   Port 0: 26 mA Ports 1, 2, 3: 15 mA
   Maximum total I_{OL} for all output pins: 71 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum V_{CC} for Power-down is 2V.

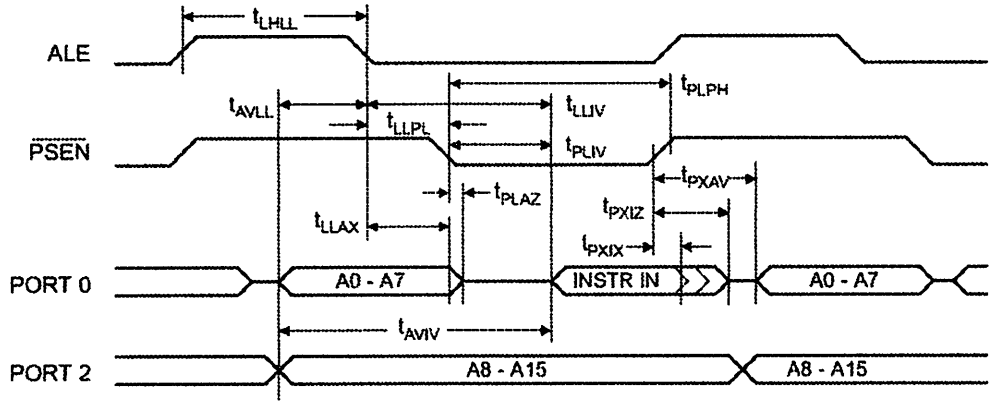
Characteristics

operating conditions, load capacitance for Port 0, ALE/$\overline{PROG}$, and $\overline{PSEN}$ = 100 pF; load capacitance for all other s = 80 pF.

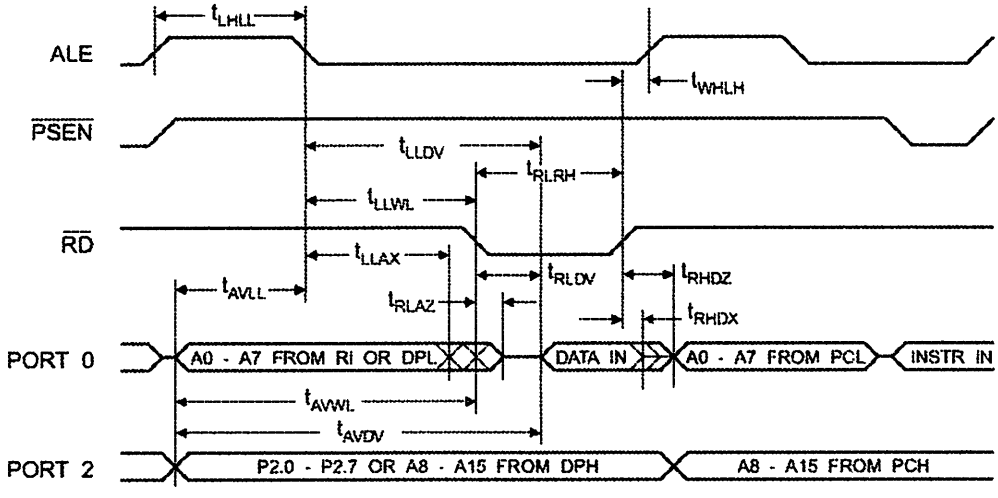
ernal Program and Data Memory Characteristics

bol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
CL	Oscillator Frequency			0	33	MHz
	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
	Address Valid to ALE Low	43		t_{CLCL}-25		ns
	Address Hold After ALE Low	48		t_{CLCL}-25		ns
	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-25		ns
	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-45		ns
	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-60	ns
	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-25	ns
	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
	Address to Valid Instruction In		312		$5t_{CLCL}$-80	ns
	$\overline{PSEN}$ Low to Address Float		10		10	ns
	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
H	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
X	Data Hold After $\overline{RD}$	0		0		ns
Z	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
L	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
X	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-30		ns
H	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-130		ns
X	Data Hold After $\overline{WR}$	33		t_{CLCL}-25		ns
	$\overline{RD}$ Low to Address Float		0		0	ns
H	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-25	t_{CLCL}+25	ns

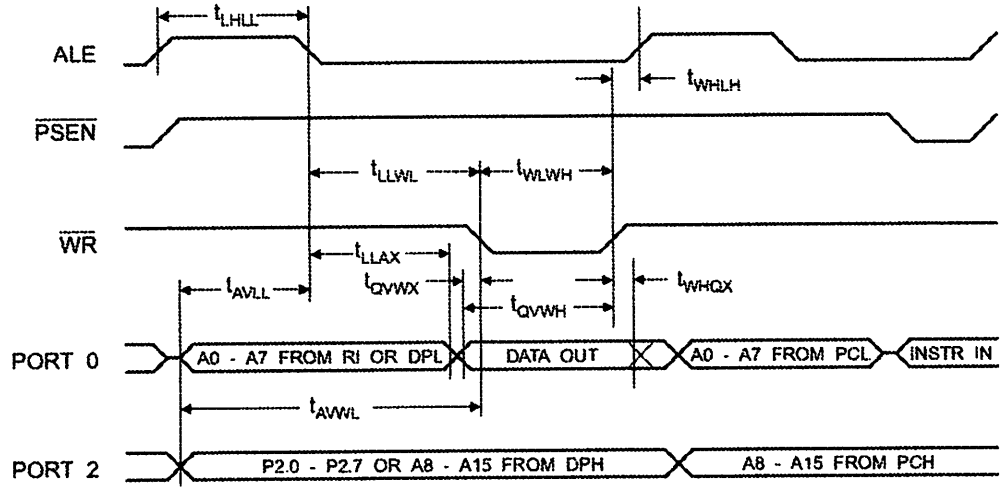
ernal Program Memory Read Cycle



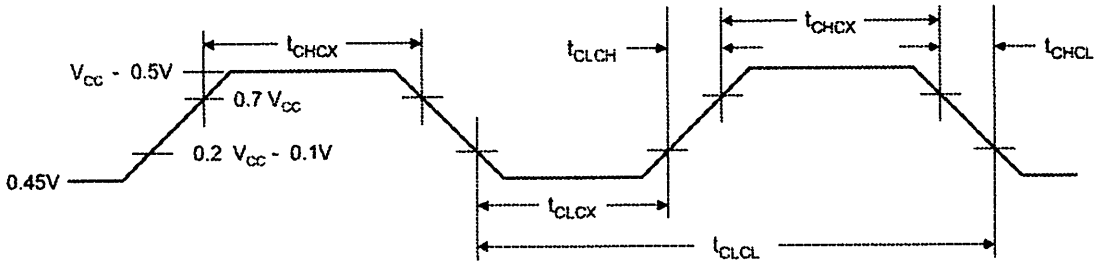
ernal Data Memory Read Cycle



ernal Data Memory Write Cycle



ernal Clock Drive Waveforms



ernal Clock Drive

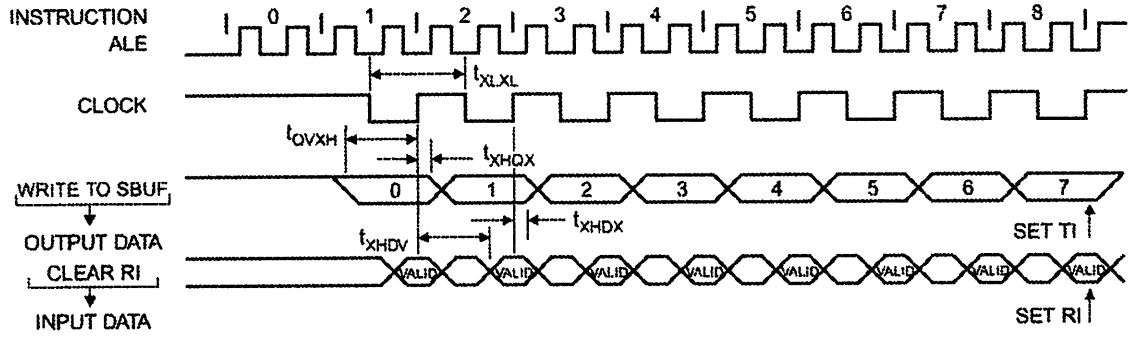
bol	Parameter	Min	Max	Units
CL	Oscillator Frequency	0	33	MHz
	Clock Period	30		ns
K	High Time	12		ns
C	Low Time	12		ns
H	Rise Time		5	ns
L	Fall Time		5	ns

al Port Timing: Shift Register Mode Test Conditions

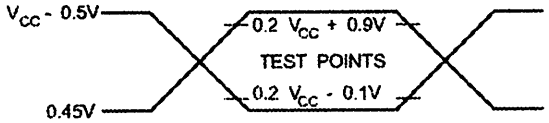
alues in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

bol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		µs
	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-80		ns
	Input Data Hold After Clock Rising Edge	0		0		ns
	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}$-133	ns

ft Register Mode Timing Waveforms

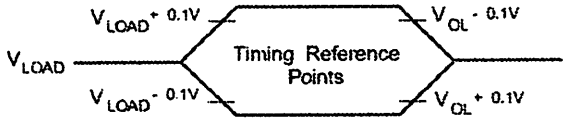


Testing Input/Output Waveforms(1)



1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

at Waveforms(1)



1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

ering Information

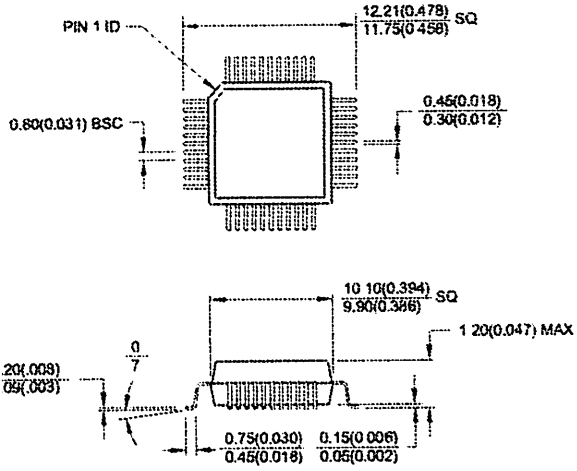
eed Hz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC AT89S51-24JC AT89S51-24PC	44A 44J 40P6	Commercial (0° C to 70° C)
		AT89S51-24AI AT89S51-24JI AT89S51-24PI	44A 44J 40P6	Industrial (-40° C to 85° C)
33	4.5V to 5.5V	AT89S51-33AC AT89S51-33JC AT89S51-33PC	44A 44J 40P6	Commercial (0° C to 70° C)

= Preliminary Availability

	Package Type
	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
	44-lead, Plastic J-leaded Chip Carrier (PLCC)
6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)

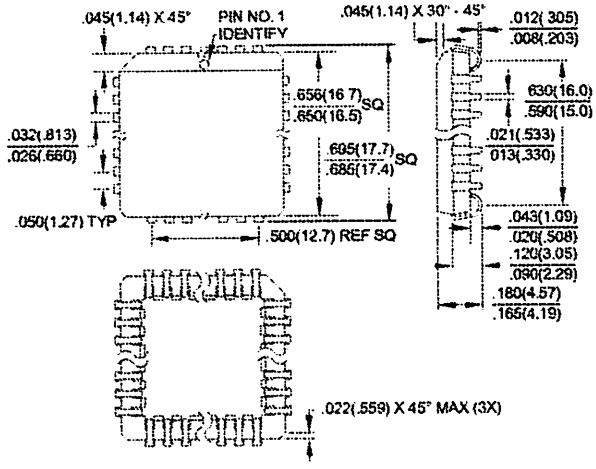
kaging Information

A, 44-lead, Thin (1.0 mm) Plastic Gull Wing Quad at Package (TQFP)
imensions in Millimeters and (Inches)*

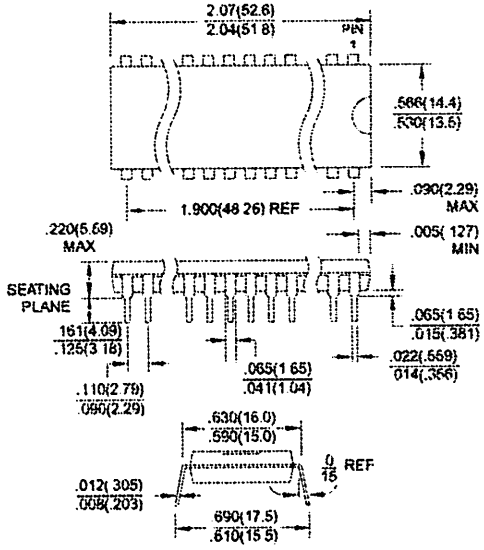


Controlling dimension: millimeters

44J, 44-lead, Plastic J-leaded Chip Carrier (PLCC)
Dimensions in Inches and (Millimeters)



0P6, 40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
Dimensions in Inches and (Millimeters)
EDEC STANDARD MS-011 AC



ıel Headquarters

ıorate Headquarters
25 Orchard Parkway
ın Jose, CA 95131
:L (408) 441-0311
\X (408) 487-2600

ıpe
mel SarL
ıute des Arsenaux 41
ısa Postale 80
ł-1705 Fribourg
vitzerland
:L (41) 26-426-5555
\X (41) 26-426-5500

ı
mel Asia, Ltd.
ıom 1219
ıinachem Golden Plaza
Mody Road Tsimhatsui
ıst Kowloon
ɔng Kong
:L (852) 2721-9778
\X (852) 2722-1369

ın
mel Japan K.K.
', Tonetsu Shinkawa Bldg.
24-8 Shinkawa
ıuo-ku, Tokyo 104-0033
ıpan
:L (81) 3-3523-3551
\X (81) 3-3523-7581

Atmel Product Operations

Atmel Colorado Springs
1150 E. Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906
TEL (719) 576-3300
FAX (719) 540-1759

Atmel Grenoble
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
TEL (33) 4-7658-3000
FAX (33) 4-7658-3480

Atmel Heilbronn
Theresienstrasse 2
POB 3535
D-74025 Heilbronn, Germany
TEL (49) 71 31 67 25 94
FAX (49) 71 31 67 24 23

Atmel Nantes
La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
TEL (33) 0 2 40 18 18 18
FAX (33) 0 2 40 18 19 60

Atmel Rousset
Zone Industrielle
13106 Rousset Cedex, France
TEL (33) 4-4253-6000
FAX (33) 4-4253-6001

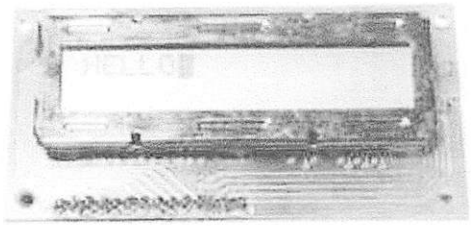
Atmel Smart Card ICs
Scottish Enterprise Technology Park
East Kilbride, Scotland G75 0QR
TEL (44) 1355-357-000
FAX (44) 1355-242-743

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

2487A–10/01/xM

How to use Intelligent L.C.D.s



By Julyan Ilett

An utterly "practical" guide to interfacing and programming intelligent liquid crystal display modules.

Part One

Recently, a number of projects using intelligent liquid crystal display (l.c.d.) modules have been featured in *EPE*. Their ability to display not just numbers, but also letters, words and all manner of symbols, makes them a good deal more versatile than the familiar 7-segment light emitting diode (l.e.d.) displays.

Although still quite expensive when purchased new, the large number of surplus modules finding their way into the hands of the "bargain" electronics suppliers, offers the hobbyist a low cost opportunity to carry out some fascinating experiments and realise some very sophisticated electronic display projects.

Basic Reading

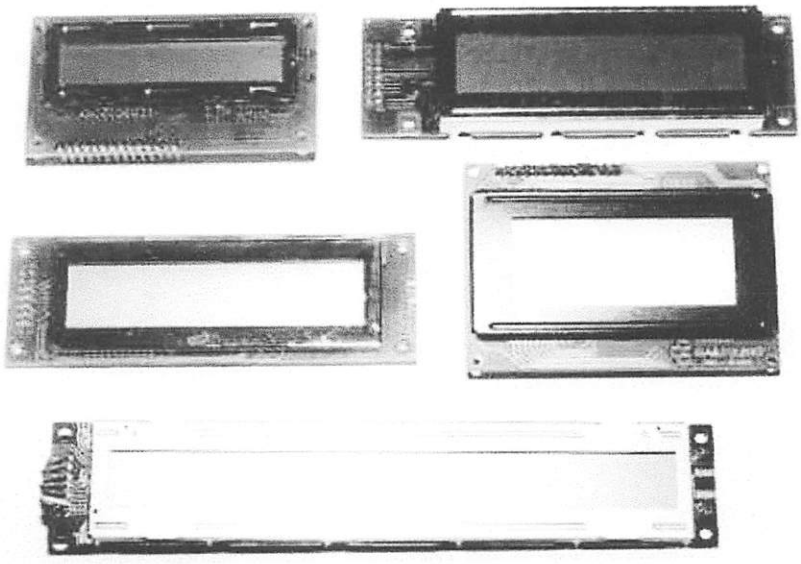
This article deals with the character-based l.c.d. modules which use the Hitachi HD44780 (or compatible) controller chip, as do most modules available to the hobbyist. Of course, these modules are not quite as advanced as the latest generation, full size, full colour, back-lit types used in today's laptop computers, but far from being "phased out," character-based l.c.d.s are still used extensively in commercial and industrial equipment, particularly where display requirements are reasonably simple.

The modules have a fairly basic interface, which mates well with traditional micro-processors such as the Z80 or the 6502. It is also ideally suited to the PIC microcontroller, which is probably the most popular microcontroller used by the electronics hobbyist.

However, even if, as yet, you know nothing of microcontrollers, and possess none of the PIC paraphernalia, don't despair, you can still enjoy all the fun of experimenting with l.c.d.s, using little more than a handful of switches!

Shapes and Sizes

Even limited to character-based modules, there is still a wide variety of shapes and sizes available. Line lengths of 8, 16, 20, 24, 32 and 40 characters are all standard, in one, two and four-line versions.



Several different liquid crystal technologies exist. "Supertwist" types, for example, offer improved contrast and viewing angle over the older "twisted nematic" types. Some modules are available with back-lighting, so that they can be viewed in dimly-lit conditions. The back-lighting may be either "electro-luminescent," requiring a high voltage inverter circuit, or simpler l.e.d. illumination.

Few of these features are important, however, for experimentation purposes. All types are capable of displaying the same basic information, so the cheaper types are probably the best bet initially.

Connections

Most l.c.d. modules conform to a standard interface specification. A 14-pin access is provided (14 holes for solder pin insertion or for an IDC connector) having eight data lines, three control lines and three power lines. The connections are laid out in one of two common configurations, either two rows of seven pins, or a single row of 14 pins. The two layout alternatives are displayed in Figure 1.

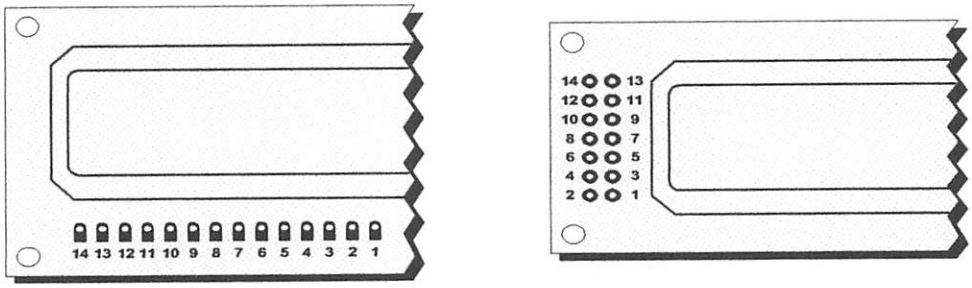


Figure 1: Pinouts of the two basic l.c.d formats.

On most displays, the pins are numbered on the l.c.d.'s printed circuit board, but if not, it is quite easy to locate pin 1. Since this pin is connected to ground, it often has a thicker p.c.b. track connected to it, and it is generally connected to the metalwork at some point.

The function of each of the connections is shown in Table 1. Pins 1 and 2 are the power supply lines, Vss and Vdd. The Vdd pin should be connected to the positive supply, and Vss to the 0V supply or ground.

Although the l.c.d. module data sheets specify a 5V d.c. supply (at only a few milliamps), supplies of 6V and 4.5V both work well, and even 3V is sufficient for some modules. Consequently, these modules can be effectively, and economically, powered by batteries.

Pin 3 is a control pin, Vee, which is used to alter the contrast of the display. Ideally, this pin should be connected to a variable voltage supply. A preset potentiometer connected between the power supply lines, with its wiper connected to the contrast pin is suitable in many cases, but be aware that some modules may require a negative potential; as low as 7V in some cases. For absolute simplicity, connecting this pin to 0V will often suffice.

Pin No.	Name	Function
1	Vss	Ground
2	Vdd	+ve supply
3	Vee	Contrast
4	RS	Register Select
5	R/W	Read/Write
6	E	Enable
7	D0	Data bit 0
8	D1	Data bit 1
9	D2	Data bit 2
10	D3	Data bit 3
11	D4	Data bit 4
12	D5	Data bit 5
13	D6	Data bit 6
14	D7	Data bit 7

Table 1. Pinout functions for all the l.c.d. types.

Pin 4 is the Register Select (RS) line, the first of the three command control inputs. When this line is low, data bytes transferred to the display are treated as commands, and data bytes read from the display indicate its status. By setting the RS line high, character data can be transferred to and from the module.

Pin 5 is the Read/Write (R/W) line. This line is pulled low in order to write commands or character data to the module, or pulled high to read character data or status information from its registers.

Pin 6 is the Enable (E) line. This input is used to initiate the actual transfer of commands or character data between the module and the data lines. When writing to the display, data is transferred only on the high to low transition of this signal. However, when reading from the display, data will become available shortly after the low to high transition and remain available until the signal falls low again.

Pins 7 to 14 are the eight data bus lines (D0 to D7). Data can be transferred to and from the display, either as a single 8-bit byte or as two 4-bit "nibbles." In the latter case, only the upper four data lines (D4 to D7) are used. This 4-bit mode is beneficial when using a microcontroller, as fewer input/output lines are required.

Prototype Circuit

For an l.c.d. module to be used effectively in any piece of equipment, a microprocessor or microcontroller is usually required to drive it. However, before attempting to wire the two together, some initial (and very useful) experiments can be performed, by connecting up a series of switches to the pins of the module. This can be quite a beneficial step, even if you are thoroughly conversant with the workings of microprocessors.

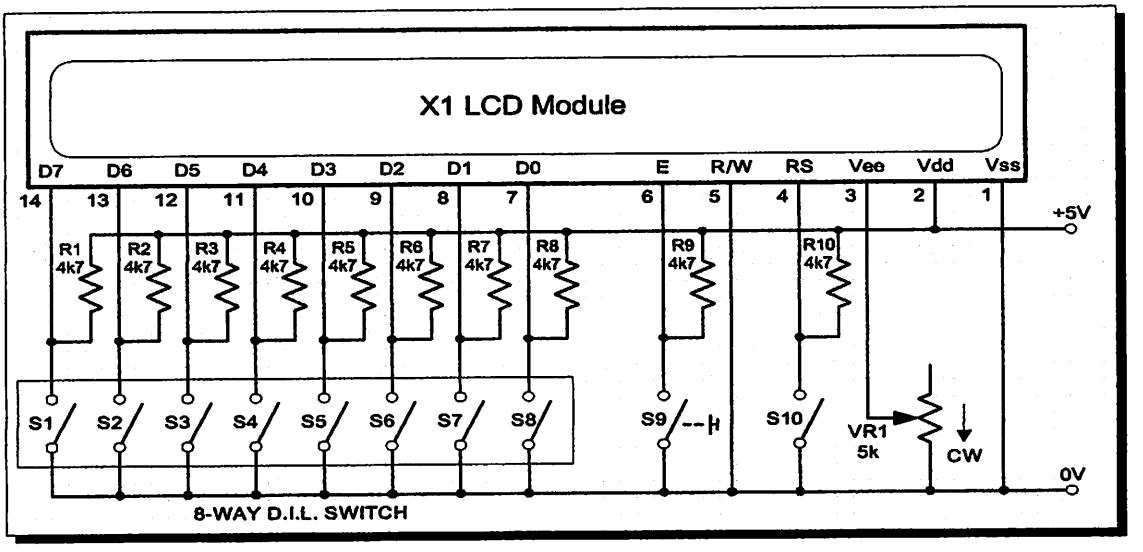


Figure 2: Circuit diagram for an l.c.d. experimental rig.

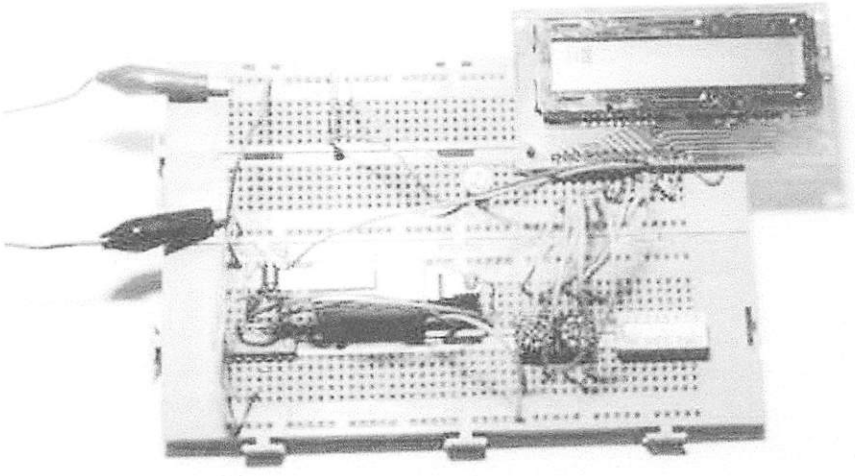
In Figure 2 is shown the circuit diagram of an l.c.d. experimentation rig. The circuit can be wired-up on a "plug-in" style prototyping board, using d.i.l. (dual-in-line) switches for the data lines (S1 to S8), a toggle switch for the RS input (S10), and a momentary action switch (or microswitch) for the E input (S9). The R/W line is connected to ground (0V), as the display is only going to be written to for the time being.

All of the resistors (R1 through R10) are 4K7 ohms. It is probably most convenient to use a s.i.l. (single-in-line) resistor pack for the eight pull-up resistors (R1 to R8) on the data lines. The other two resistors, R9 and R10, can be discrete types. Preset potentiometer VR1 (5K ohms) is used for the contrast control and is shown with one end left disconnected. If desired, this end can be connected to the positive line via a resistor of about 47K ohms (it should be connected to a negative supply, via a similar resistor, for those modules which require negative biasing).

All the switches should be connected so that they are "on" when in the "down" position, so that "down" generates a logic 0 (low) and "up" provides a logic 1 (high). The switches should also be arranged so that data bit D7 is on the left, and data bit D0 is on the right. In this way, binary numbers can be entered the right way round.

Initially, the contrast control should be adjusted fully clockwise, so that the contrast control input (Vee) is connected to ground. The initial settings of the switches are

unimportant, but it is suggested that the RS switch (S10) is "up" (set to logic 1), and the E switch (S9) is left unpressed. The data switches, S1 to S8, can be set to any value at this stage. All is now prepared to start sending commands and data to the l.c.d. module.



The experimental circuit can be built on plug-in prototyping boards.

Experiment 1: Basic Commands

When powered up, the display should show a series of dark squares, possibly only on part of the display. These character cells are actually in their off state, so the contrast control should be adjusted anti-clockwise (away from ground) until the squares are only just visible.

The display module resets itself to an initial state when power is applied, which curiously has the display blanked off, so that even if characters are entered, they cannot be seen. It is therefore necessary to issue a command at this point, to switch the display on.

A full list of the commands that can be sent is given in Table 2, together with their binary and hexadecimal values. The initial conditions of the l.c.d. after power-on are marked with an asterisk.

Throughout this article, emphasis will be placed on the binary value being sent since this illustrates which data bits are being set for each command. After each binary value, the equivalent hexadecimal value is quoted in brackets, the $ prefix indicating that it is hexadecimal.

The Display On/Off and Cursor command turns on the display, but also determines the cursor style at the same time. Initially, it is probably best to select a Blinking Cursor with Underline, so that its position can be seen clearly, i.e. code 00001111 ($0F).

Command	Binary D7	D6	D5	D4	D3	D2	D1	D0	Hex
Clear Display	0	0	0	0	0	0	0	1	01
Display & Cursor Home	0	0	0	0	0	0	1	x	02 or 03
Character Entry Mode	0	0	0	0	0	1	1/D	S	04 to 07
Display On/Off & Cursor	0	0	0	0	1	D	U	B	08 to 0F
Display/Cursor Shift	0	0	0	1	D/C	R/L	x	x	10 to 1F
Function Set	0	0	1	8/4	2/1	10/7	x	x	20 to 3F
Set CGRAM Address	0	1	A	A	A	A	A	A	40 to 7F
Set Display Address	1	A	A	A	A	A	A	A	80 to FF

1/D:	1=Increment*, 0=Decrement	R/L:	1=Right shift, 0=Left shift
S:	1=Display shift on, 0=Off*	8/4:	1=8-bit interface*, 0=4-bit interface
D:	1=Display on, 0=Off*	2/1:	1=2 line mode, 0=1 line mode*
U:	1=Cursor underline on, 0=Off*	10/7:	1=5x10 dot format, 0=5x7 dot format*
B:	1=Cursor blink on, 0=Off*		
D/C:	1=Display shift, 0=Cursor move	x = Don't care	* = Initialization settings

Table 2. The command control codes.

Set the data switches (S1 to S8) to 00001111 ($0F) and ensure that the RS switch (S10) is "down" (logic 0), so that the device is in Command mode. Now press the E switch (S9) momentarily, which "enables" the chip to accept the data, and Hey Presto, a flashing cursor with underline appears in the top left hand position!

If a two-line module is being used, the second line can be switched on by issuing the Function Set command. This command also determines whether an 8-bit or a 4-bit data transfer mode is selected, and whether a 5 x 10 or 5 x 7 pixel format will be used. So, for 8-bit data, two lines and a 5 x 7 format, set the data switches to binary value 00111000 ($38), leave RS (S10) set low and press the E switch, S9.

It will now be necessary to increase the contrast a little, as the two-line mode has a different drive requirement. Now set the RS switch to its "up" position (logic 1), switching the chip from Command mode to Character mode, and enter binary value 01000001 ($41) on the data switches. This is the ASCII code for a capital *A*.

Press the E switch, and marvel as the display fills up with capital *A*'s. Clearly, something is not quite right, and seeing your name in pixels is going to have to wait a while.

Bounce

The problem here is contact bounce. Practically every time the E switch is closed, its contacts will bounce, so that although occasionally only one character appears, most attempts will result in 10 or 20 characters coming up on the display. What is needed is a "debounce" circuit.

But what about the commands entered earlier, why didn't contact bounce interfere with them? In fact it did, but it doesn't matter whether a command is entered ("enabled") just once, or several times, it gets executed anyway. A solution to the bounce problem is shown in Figure 3.

Here, a couple of NAND gates are cross-coupled to form a set-reset latch (or flip-flop) which flips over and latches, so that the contact bounce is eliminated. Either a TTL 74LS00 or a CMOS 74HC00 can be used in this circuit. The switch must be an s.p.d.t. (single-pole, double-throw) type, a microswitch is ideal.

After modifying the circuit, the screen full of *A*'s can be cleared using the Clear Display command. Put binary value 00000001 ($01) on the data switches, set the RS switch to the "down" position and press the new modified E switch. The display is cleared.

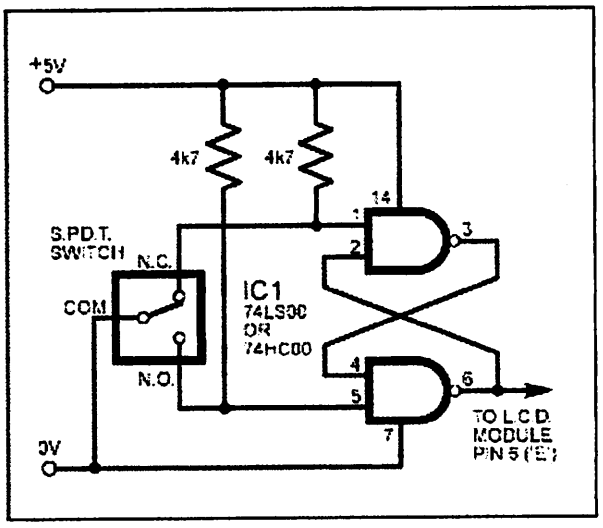


Figure 3. Switch debounce circuit.

Note that the output of the "de-bounce" circuit is high when the switch is pressed and low when the switch is released. Since it is the high to low transition that actually latches data into the l.c.d. module, it will be observed that characters appear on the display, not when the button is pressed, but when it is released.

Experiment 2: Entering Text

First, a little tip: it is manually a lot easier to enter characters and commands in hexadecimal rather than binary (although, of course, you will need to translate commands from binary into hex so that you know which bits you are setting). Replacing the d.i.l. switch pack with a couple of sub-miniature hexadecimal rotary switches is a simple matter, although a little bit of re-wiring is necessary.

The switches must be the type where On = 0, so that when they are turned to the zero position, all four outputs are shorted to the common pin, and in position "F", all four outputs are open circuit.

All the available characters that are built into the module are shown in Table 3. Studying the table, you will see that codes associated with the characters are quoted in binary and hexadecimal, most significant bits ("left-hand" four bits) across the top, and least significant bits ("right-hand" four bits) down the left.

Most of the characters conform to the ASCII standard, although the Japanese and Greek characters (and a few other things) are obvious exceptions. Since these intelligent modules were designed in the "Land of the Rising Sun," it seems only fair that their Katakana phonetic symbols should also be incorporated. The more extensive Kanji character set, which the Japanese share with the Chinese, consisting of several thousand different characters, is not included!

Table 3. Standard l.c.d character table.

Using the switches, of whatever type, and referring to Table 3, enter a few characters onto the display, both letters and numbers. The RS switch (S10) must be "up" (logic 1) when sending the characters, and switch E (S9) must be pressed for each of them. Thus

the operational order is: set RS high, enter character, trigger E, leave RS high, enter another character, trigger E, and so on.

The first 16 codes in Table 3, 00000000 to 00001111, ($00 to $0F) refer to the CGRAM. This is the Character Generator RAM (random access memory), which can be used to hold user-defined graphics characters. This is where these modules really start to show their potential, offering such capabilities as bargraphs, flashing symbols, even animated characters. Before the user-defined characters are set up, these codes will just bring up strange looking symbols.

Codes 00010000 to 00011111 ($10 to $1F) are not used and just display blank characters. ASCII codes "proper" start at 00100000 ($20) and end with 01111111 ($7F). Codes 10000000 to 10011111 ($80 to $9F) are not used, and 10100000 to 11011111 ($A0 to $DF) are the Japanese characters.

Codes 11100000 to 11111111 ($E0 to $FF) are interesting. Although this last block contains mainly Greek characters, it also includes the lower-case characters which have "descenders." These are the letters *g*, *j*, *p*, *q* and *y*, where the tail drops down below the base line of normal upper-case characters. They require the 5 x 10 dot matrix format, rather than the 5 x 7, as you will see if you try to display a lower-case *j*, for example, on a 5 x 7 module.

Some one-line displays have the 5 x 10 format facility, which allows these characters to be shown unbroken. With 5 x 7 two-line displays, the facility can be simulated by borrowing the top three pixel rows from the second line, so creating a 5 x 10 matrix.

For this simulation, set line RS low to put the chip into Command mode. On the data switches, enter the Function Set command using binary value 00110100 ($34). Press and release switch E. Return RS to high, and then send the character data for the last 32 codes in the normal way (remembering to trigger line E!).

Experiment 3: Addressing

When the module is powered up, the cursor is positioned at the beginning of the first line. This is address $00. Each time a character is entered, the cursor moves on to the next address, $01, $02 and so on. This auto-incrementing of the cursor address makes entering strings of characters very easy, as it is not necessary to specify a separate address for each character.

It may be necessary, however, to position a string of characters somewhere other than at the beginning of the first line. In this instance, a new starting address must be entered as a command. Any address between $00 and $7F can be entered, giving a total of 128 different addresses, although not all these addresses have their own display location. There are in fact only 80 display locations, laid out as 40 on each line in two-line mode, or all 80 on a single line in one-line mode. This situation is further complicated because not all display locations are necessarily visible at one time. Only a 40-character, two-line module can display all 80 locations simultaneously.

To experiment with addressing, first set the l.c.d. to two-line mode (if two lines are available), 8-bit data and 5 [P3] 7 format using the Function Set command, i.e. code 00111000 ($38). Note that the last two bits of this command are unimportant, as indicated by the X in the columns of Table 2, and either of them may be set to 0 or 1.

(From now on, we won't constantly remind you that RS must be set appropriately before Command or Character data is entered, or that E must be triggered after data has been entered you should know by now!)

Using the Display On/Off and Cursor command, set the display to On, with Underline and Blinking Cursor, code 00001111 ($0F). Now set the cursor to address 00001000 ($08). This is done by sending a Set Display Address command, binary value 10001000 ($88).

The cursor will jump to the ninth position on the display, at which point text can now be entered. The Set Display Address command is always 10000000 ($80) greater than the display address itself.

Experiment with different display addresses and note their display locations. Be aware that display addresses 00101000 to 00111111 ($28 to $3F) and 01101000 to 01111111 ($68 to $7F) cannot be used on any of the display types.

The relationship between addresses and display locations varies, depending on the type of module being used, but some typical examples are shown in Figure 4.

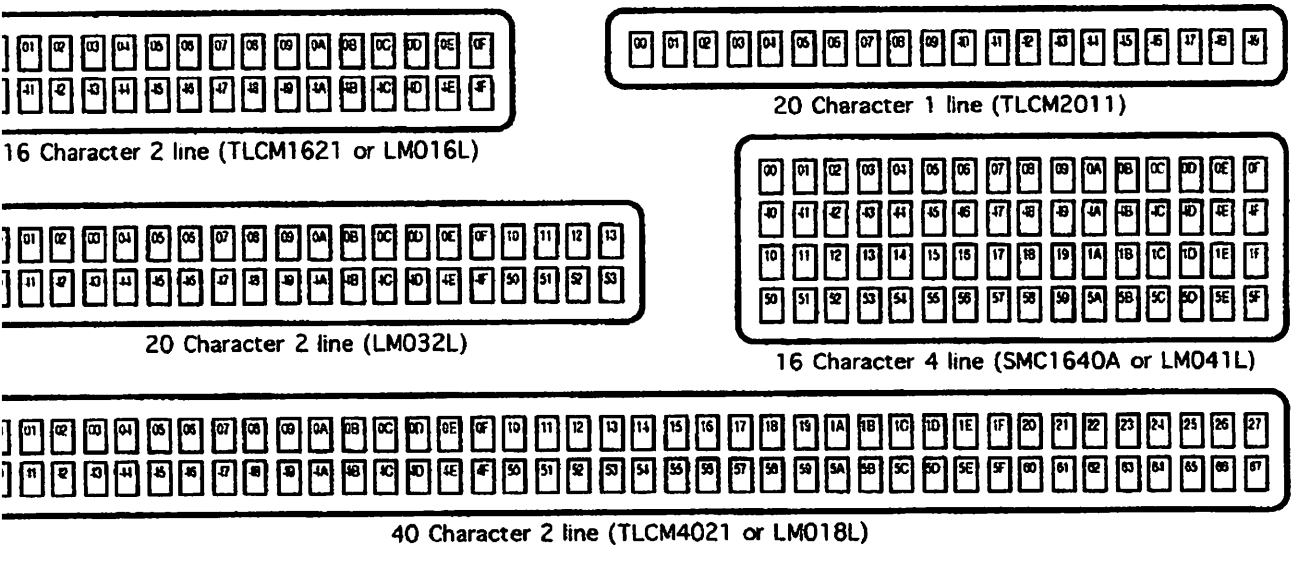


Figure 4: Examples of the relationship between addresses and display locations for typical module formats

Most are laid out conventionally, with two lines of characters, the first line starting at address 00000000 ($00) and the second line at address 01000000 ($40).
Two interesting exceptions were discovered during this article's research. The single-line module shown in Figure 4 is actually a two-line type, with the second line placed to the right of the first. In one-line mode, only the first 10 characters were visible.

The rather magnificent 4-line module is, actually, also a two-line type, with the two lines split and interlaced. This complicates the addressing a little, but can be sorted out with a bit of software.

Experiment 4: Shifting the Display

Regardless of which size l.c.d. module is being used, there are always 80 display locations that can be written to. On the smaller devices, not all 80 fit within the visible window of the module, but can be brought into view by shifting them all, either left or right, "beneath" the window area. This process must be carried out carefully, however, as it alters the relationship between addresses and their positions on the screen.

To experiment with shifting, first issue suitable Function Set, Display On/Off and Cursor commands, and, if necessary, the Clear Display command (you've met their codes above). Then enter all 26 letters of the alphabet as character data, e.g. 01000001 ($41) to 01011010 ($5A).

On a 16-character display, only *A* to *P* will be visible (the first 16 letters of the alphabet), and the cursor will have disappeared off the right-hand side of the display screen.

The Cursor/Display Shift command can now be used to scroll all the display locations to the left, "beneath" the l.c.d. window, so that letters *Q* to *Z* can be seen. The command is binary 00011000 ($18). Each time the command is entered (and using the E switch), the characters shift one place to the left. The cursor will re-appear from the right-hand side, immediately after the *Z* character.

Carry on shifting (*wasn't that a film title? Ed!*), and eventually the letters *A*, *B*, *C*, and so on, will also come back in from the right-hand side. Shifting eventually causes complete rotation of the display locations.

The binary command 00011100 ($1C) shifts the character locations to the right. It is important to note that this scrolling does not actually move characters into new addresses, it moves the whole address block left or right "underneath" the display window.

If the display locations are not shifted back to their original positions, then address $00 will no longer be at the left-hand side of the display. Try entering an Address Set command of value 10000000 ($80), after a bit of shifting, to see where it has moved to.

The Cursor Home command, binary 00000010 ($02), will both set the cursor back to address $00, and shift the address $00 itself back to the left-hand side of the display. This command can be used to get back to a known good starting position, if shifting and address setting gets a bit out of control.

The Clear Display command does the same as Cursor Home, but also clears all the display locations.

One final word about the Cursor/Display Shift command; it is also used to shift the cursor. Doing this simply increments or decrements the cursor address and actually has very little in common with shifting the display, even though both are achieved using the same command.

Experiment 5: Character Entry Mode

Another command listed in Table 2 is Character Entry Mode. So far, characters have been entered using auto-incrementing of the cursor address, but it is also possible to use auto-decrementing. Furthermore, it is possible to combine shifting of the display with both auto-incrementing and auto-decrementing.

Consider an electronic calculator. Initially, a single zero is located on the right-hand side of the display. As numbers are entered, they move to the left, leaving the cursor in a fixed position at the far right. This mode of character entry can be emulated on the l.c.d. module. Time for another experiment:

Send suitable Function Set, Display On/Off and Cursor commands as before. Next, and assuming a 16-character display, set the cursor address to 00010000 ($10). Then send the Character Entry Mode command, binary 00000111 ($07). This sets the entry mode to auto-increment/display shift left.

Finally, enter a few numbers from 0 to 9 decimal, i.e. from 00110000 to 00111001 ($30 to $39). Characters appear on the right-hand side and scroll left as more characters are entered, just like a normal calculator.

As seen in Table 2, there are four different Character Entry modes, 00000100 to 00000111 ($04 to $07), all of which have their different uses in real life situations.

Experiment 6: User-Defined Graphics

Commands 01000000 to 01111111 ($40 to $7F) are used to program the user-defined graphics. The best way to experiment with these is to program them "on screen." This is carried out as follows:

First, send suitable Function Set, Display On/Off and Cursor commands, then issue a Clear Display command. Next, send a Set Display Address command to position the cursor at address 00000000 ($00). Lastly, display the contents of the eight user character

locations by entering binary data 00000000 to 00000111 ($00 to $07) in turn. These characters will initially show up as garbage, or a series of stripes.

Now, send a Set CGRAM Address command, to start defining the user characters. Any value between 01000000 and 01111111 ($40 and $7F) is valid, but for now, use 01000000 ($40). The cursor will jump to the beginning of the second line, but ignore this, as it is not important.

Data entered from now on will build up the user-defined graphics, row by row. Try the following sequence of data: 00001110, 00010001, 00001110, 00000100, 00011111, 00000100, 00001010, 00010001 ($0E, $11, $0E, $04, $1F, $04, $0A, $11). A little "stick man" will appear on the display, with his feet in the gutter (the cursor line)!

By entering another set of eight bytes, the second user character can be defined, and so on.

How the CGRAM addresses correspond to the individual pixels of the user-defined graphics characters is illustrated in Figure 5. Up to eight graphics can be programmed, which then become part of the character set and can be called up using codes 00000000 to 00000111 ($00 to $07), or codes 00001000 to 00001111 ($08 to $0F), both of which produce the same result, i.e. 64 command codes available for user programming.

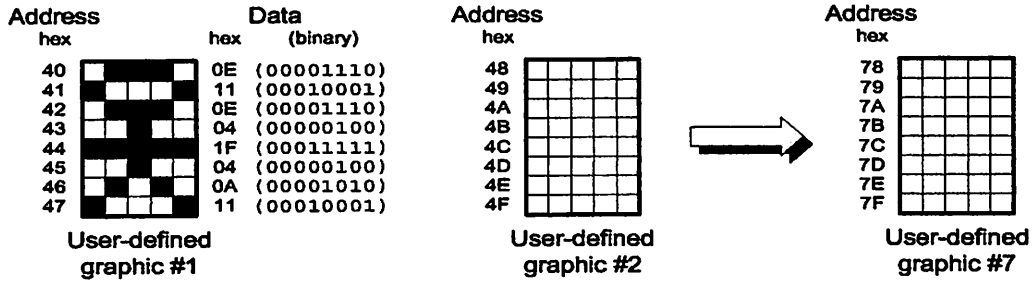


Figure 5: Showing how the CGRAM addresses correspond to individual pixels.

It can be seen that the basic character cell is actually eight pixels high by five pixels wide, but most characters just use the upper seven rows. The bottom row is generally used for the underline cursor. Since each character is only five pixels wide, only data bits 0 to 4 are used, bits 5 to 7 (the three "left-hand" bits) are ignored.

The CGRAM is volatile memory, which means that when the power supply is removed from the l.c.d. module, the user-defined characters will be lost. It is necessary for the microprocessor to load up the user-defined characters, by copying data from its own EPROM, early on in the program, certainly before it intends to display them.

Experiment 7: 4-Bit Data Transfer

The HD44780 l.c.d. control chip, found in most l.c.d. modules, was designed to be compatible with 4-bit microprocessors. The 4-bit mode is still very useful when interfacing to microcontrollers, including the PIC types.

Microcontroller input/output (I/O) pins are often at a premium and have to be rationed carefully between the various switches, displays and other input and output devices in a typical circuit. Bigger microcontrollers are available, which have more I/O pins, but miniaturisation is a key factor these days, along with cost, of course.

Once the display is put into 4-bit mode, using the Function Set command, it is a simple matter of sending two "nibbles" instead of one byte, for each subsequent command or character.

Nibble is a name devised by early computer enthusiasts in America, for half a byte, and is one of the more frivolous terms that has survived. By the time the 16-bit processors arrived, computing was getting serious, and the consumption analogies "gobble" and "munch" were never adopted!

When using 4-bit mode, only data lines D4 to D7 are used. On the prototype test rig, set the switches on the other lines, D0 to D3, to logic 0, and leave them there. Another experiment is now imminent.

In normal use, the unused data I/O lines D0 to D3 should either be left floating, or tied to one of the two power rails via a resistor of somewhere between 4k7[C24] and 47k[C24]. It is undesirable to tie them directly to ground unless the R/W line is also tied to ground, preventing them from being set into output mode. Otherwise the device could be programmed erroneously for 8-bit output, which could be unkind to lines D0 to D3, even though current limiting exists.

After power on, the l.c.d. module will be in 8-bit mode. The Function Set command must first be sent to put the display into 4-bit mode, but there is a difficulty. With no access to the lower four data lines, D0 to D3, only half the command can be applied.

Fortunately, or rather, by clever design, the 8-bit/4-bit selection is on data bit D4, which, even on the modified test rig, remains accessible. By sending a command with binary value 00100000 ($20), the 4-bit mode is invoked.

Now, another Function Set command can be sent, to set the display to two-line mode. Binary value 00101000 ($28) will do the trick. The value 00111000 ($38) may be a more familiar number, but it cannot be used now, or the display would be put straight back into 8-bit mode! Also, from now on, all commands and data must be sent in two halves, the upper four bits first, then the lower four bits.

Start by setting data lines D7, D6, D5 and D4 to 0010 ($2), the left-hand four bits of the 8-bit code, and press the E switch. Then we set the same four data lines to 1000 ($8), the right-hand four bits of the 8-bit code, and press the E switch again. It's a lot more laborious for a human being, but to a microcontroller, it's no problem!

Finish off by experimenting with other commands in 4-bit mode, and then try putting a few characters on the display.

A Final Note

The data sheets warn that under certain conditions, the l.c.d. module may fail to initialise properly when power is first applied. This is particularly likely if the Vdd supply does not rise to its correct operating voltage quickly enough.

It is recommended that after power is applied, a command sequence of three bytes of value 0011XXXX ($3X) is sent to the module. The value $30 is probably most convenient. This will guarantee that the module is in 8-bit mode, and properly initialised. Following this, switching to 4-bit mode (and indeed all other commands) will work reliably.

That's it – For Now!

Well, that's about it, really. You've made it this far, so now you know everything there is to know about l.c.d. modules. Well, almost everything!

The next step, of course, is to connect the display up to a controller of some sort, such as a PIC microcontroller, as will be seen next month. Then we shall also consider such things as signal timing and instruction delays.

Acknowledgement

The author expresses his gratitude to Bull Electrical in Hove and Greenweld Electronics in Southampton for their help in connection with this article.

-4323; Rev 10; 8/01

MAXIM

+5V-Powered, Multichannel RS-232 Drivers/Receivers

General Description

ıe MAX220–MAX249 family of line drivers/receivers is lended for all EIA/TIA-232E and V.28/V.24 communica-ıns interfaces, particularly applications where ±12V is ıt available.

ıese parts are especially useful in battery-powered sys-ms, since their low-power shutdown mode reduces ɔwer dissipation to less than 5µW. The MAX225, AX233, MAX235, and MAX245/MAX246/MAX247 use) external components and are recommended for appli-ıtions where printed circuit board space is critical.

Applications

Portable Computers

Low-Power Modems

Interface Translation

Battery-Powered RS-232 Systems

Multidrop RS-232 Networks

Features

Superior to Bipolar

- **Operate from Single +5V Power Supply (+5V and +12V—MAX231/MAX239)**
- **Low-Power Receive Mode in Shutdown (MAX223/MAX242)**
- **Meet All EIA/TIA-232E and V.28 Specifications**
- **Multiple Drivers and Receivers**
- **3-State Driver and Receiver Outputs**
- **Open-Line Detection (MAX243)**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX220CPE	0°C to +70°C	16 Plastic DIP
MAX220CSE	0°C to +70°C	16 Narrow SO
MAX220CWE	0°C to +70°C	16 Wide SO
MAX220C/D	0°C to +70°C	Dice*
MAX220EPE	-40°C to +85°C	16 Plastic DIP
MAX220ESE	-40°C to +85°C	16 Narrow SO
MAX220EWE	-40°C to +85°C	16 Wide SO
MAX220EJE	-40°C to +85°C	16 CERDIP
MAX220MJE	-55°C to +125°C	16 CERDIP

Ordering Information continued at end of data sheet.

**Contact factory for dice specifications.*

Selection Table

art umber	Power Supply (V)	No. of RS-232 Drivers/Rx	No. of Ext. Caps	Nominal Cap. Value (µF)	SHDN & Three-State	Rx Active in SHDN	Data Rate (kbps)	Features
AX220	+5	2/2	4	0.1	No	----	120	Ultra-low-power, industry-standard pinout
AX222	+5	2/2	4	0.1	Yes	----	200	Low-power shutdown
AX223 (MAX213)	+5	4/5	4	1.0 (0.1)	Yes	✔	120	MAX241 and receivers active in shutdown
AX225	+5	5/5	0	----	Yes	✔	120	Available in SO
AX230 (MAX200)	+5	5/0	4	1.0 (0.1)	Yes	----	120	5 drivers with shutdown
AX231 (MAX201)	+5 and +7.5 to +13.2	2/2	2	1.0 (0.1)	No	----	120	Standard +5/+12V or battery supplies, same functions as MAX232
AX232 (MAX202)	+5	2/2	4	1.0 (0.1)	No	----	120 (64)	Industry standard
AX232A	+5	2/2	4	0.1	No	----	200	Higher slew rate, small caps
AX233 (MAX203)	+5	2/2	0	----	No	----	120	No external caps
AX233A	+5	2/2	0	----	No	----	200	No external caps, high slew rate
AX234 (MAX204)	+5	4/0	4	1.0 (0.1)	No	----	120	Replaces 1488
AX235 (MAX205)	+5	5/5	0	----	Yes	----	120	No external caps
AX236 (MAX206)	+5	4/3	4	1.0 (0.1)	Yes	----	120	Shutdown, three state
AX237 (MAX207)	+5	5/3	4	1.0 (0.1)	No	----	120	Complements IBM PC serial port
AX238 (MAX208)	+5	4/4	4	1.0 (0.1)	No	----	120	Replaces 1488 and 1489
AX239 (MAX209)	+5 and +7.5 to +13.2	3/5	2	1.0 (0.1)	No	----	120	Standard +5/+12V or battery supplies; single-package solution for IBM PC serial port
AX240	+5	5/5	4	1.0	Yes	----	120	DIP or flatpack package
AX241 (MAX211)	+5	4/5	4	1.0 (0.1)	Yes	----	120	Complete IBM PC serial port
AX242	+5	2/2	4	0.1	Yes	✔	200	Separate shutdown and enable
AX243	+5	2/2	4	0.1	No	----	200	Open-line detection simplifies cabling
AX244	+5	8/10	4	1.0	No	----	120	High slew rate
AX245	+5	8/10	0	----	Yes	✔	120	High slew rate, int. caps, two shutdown modes
AX246	+5	8/10	0	----	Yes	✔	120	High slew rate, int. caps, three shutdown modes
AX247	+5	8/9	0	----	Yes	✔	120	High slew rate, int. caps, nine operating modes
AX248	+5	8/8	4	1.0	Yes	✔	120	High slew rate, selective half-chip enables
AX249	+5	6/10	4	1.0	Yes	✔	120	Available in quad flatpack package

5V-Powered, Multichannel RS-232 rivers/Receivers

SOLUTE MAXIMUM RATINGS—MAX220/222/232A/233A/242/243

ply Voltage (V_{CC}) .....-0.3V to +6V
t Voltages
.....-0.3V to (V_{CC} - 0.3V)
(Except MAX220) .....±30V
(MAX220).....±25V
UT (Except MAX220) (Note 1) .....±15V
UT (MAX220).....±13.2V
put Voltages
UT.....±15V
UT.....-0.3V to (V_{CC} + 0.3V)
er/Receiver Output Short Circuited to GND.....Continuous
tinuous Power Dissipation (T_A = +70°C)
-Pin Plastic DIP (derate 10.53mW/°C above +70°C)....842mW
-Pin Plastic DIP (derate 11.11mW/°C above +70°C)....889mW
20-Pin Plastic DIP (derate 8.00mW/°C above +70°C) ..440mW
16-Pin Narrow SO (derate 8.70mW/°C above +70°C) ...696mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C).....762mW
18-Pin Wide SO (derate 9.52mW/°C above +70°C).....762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C)....800mW
20-Pin SSOP (derate 8.00mW/°C above +70°C) 640mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C).....800mW
18-Pin CERDIP (derate 10.53mW/°C above +70°C).....842mW
Operating Temperature Ranges
MAX2_ _AC_ _, MAX2_ _C_ _.....0°C to +70°C
MAX2_ _AE_ _, MAX2_ _E_ _ .....-40°C to +85°C
MAX2_ _AM_ _, MAX2_ _M_ _.....-55°C to +125°C
Storage Temperature Range .....-65°C to +160°C
Lead Temperature (soldering, 10sec) .....+300°C

e 1: Input voltage measured with T_{OUT} in high-impedance state, $\overline{SHDN}$ or V_{CC} = 0V.
e 2: For the MAX220, V+ and V- can have a maximum magnitude of 7V, but their absolute difference cannot exceed 13V.

sses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional ration of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to olute maximum rating conditions for extended periods may affect device reliability.

ECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243

C = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, T_A = T_{MIN} to T_{MAX}, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
S-232 TRANSMITTERS						
utput Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±8		V
put Logic Threshold Low				1.4	0.8	V
put Logic Threshold High	All devices except MAX220		2	1.4		V
	MAX220: V_{CC} = 5.0V		2.4			
gic Pull-Up/Input Current	All except MAX220, normal operation			5	40	µA
	$\overline{SHDN}$ = 0V, MAX222/242, shutdown, MAX220			±0.01	±1	
utput Leakage Current	V_{CC} = 5.5V, $\overline{SHDN}$ = 0V, V_{OUT} = ±15V, MAX222/242			±0.01	±10	µA
	V_{CC} = $\overline{SHDN}$ = 0V, V_{OUT} = ±15V			±0.01	±10	
ata Rate				200	116	kb/s
ansmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V		300	10M		Ω
utput Short-Circuit Current	V_{OUT} = 0V		±7	±22		mA
S-232 RECEIVERS						
S-232 Input Voltage Operating Range					±30	V
S-232 Input Threshold Low	V_{CC} = 5V	All except MAX243 $R2_{IN}$	0.8	1.3		V
		MAX243 $R2_{IN}$ (Note 2)	-3			
S-232 Input Threshold High	V_{CC} = 5V	All except MAX243 $R2_{IN}$		1.8	2.4	V
		MAX243 $R2_{IN}$ (Note 2)		-0.5	-0.1	
S-232 Input Hysteresis	All except MAX243, V_{CC} = 5V, no hysteresis in shdn.		0.2	0.5	1	V
	MAX243			1		
S-232 Input Resistance			3	5	7	kΩ
TL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} - 0.2		V
TL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		

ELECTRICAL CHARACTERISTICS—MAX220/222/232A/233A/242/243 (continued)

(VCC = +5V ±10%, C1–C4 = 0.1µF, MAX220, C1 = 0.047µF, C2–C4 = 0.33µF, TA = TMIN to TMAX, unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
TTL/CMOS Output Leakage Current	$\overline{SHDN}$ = VCC or $\overline{EN}$ = VCC ($\overline{SHDN}$ = 0V for MAX222), 0V ≤ VOUT ≤ VCC			±0.05	±10	µA
$\overline{EN}$ Input Threshold Low	MAX242			1.4	0.8	V
$\overline{EN}$ Input Threshold High	MAX242		2.0	1.4		V
Operating Supply Voltage			4.5		5.5	V
ICC Supply Current ($\overline{SHDN}$ = VCC), Figures 5, 6, 11, 19	No load	MAX220		0.5	2	mA
		MAX222/232A/233A/242/243		4	10	
	3kΩ load both inputs	MAX220		12		
		MAX222/232A/233A/242/243		15		
Shutdown Supply Current	MAX222/242	TA = +25°C		0.1	10	µA
		TA = 0°C to +70°C		2	50	
		TA = -40°C to +85°C		2	50	
		TA = -55°C to +125°C		35	100	
$\overline{SHDN}$ Input Leakage Current	MAX222/242				±1	µA
$\overline{SHDN}$ Threshold Low	MAX222/242			1.4	0.8	V
$\overline{SHDN}$ Threshold High	MAX222/242		2.0	1.4		V
Transition Slew Rate	CL = 50pF to 2500pF, RL = 3kΩ to 7kΩ, VCC = 5V, TA = +25°C, measured from +3V to -3V or -3V to +3V	MAX222/232A/233A/242/243	6	12	30	V/µs
		MAX220	1.5	3	30	
Transmitter Propagation Delay TTL to RS-232 (normal operation), Figure 1	tPHLT	MAX222/232A/233A/242/243		1.3	3.5	µs
		MAX220		4	10	
	tPLHT	MAX222/232A/233A/242/243		1.5	3.5	
		MAX220		5	10	
Receiver Propagation Delay RS-232 to TTL (normal operation), Figure 2	tPHLR	MAX222/232A/233A/242/243		0.5	1	µs
		MAX220		0.6	3	
	tPLHR	MAX222/232A/233A/242/243		0.6	1	
		MAX220		0.8	3	
Receiver Propagation Delay RS-232 to TTL (shutdown), Figure 2	tPHLS	MAX242		0.5	10	µs
	tPLHS	MAX242		2.5	10	
Receiver-Output Enable Time, Figure 3	tER	MAX242		125	500	ns
Receiver-Output Disable Time, Figure 3	tDR	MAX242		160	500	ns
Transmitter-Output Enable Time ($\overline{SHDN}$ goes high), Figure 4	tET	MAX222/242, 0.1µF caps (includes charge-pump start-up)		250		µs
Transmitter-Output Disable Time ($\overline{SHDN}$ goes low), Figure 4	tDT	MAX222/242, 0.1µF caps		600		ns
Transmitter + to - Propagation Delay Difference (normal operation)	tPHLT - tPLHT	MAX222/232A/233A/242/243		300		ns
		MAX220		2000		
Receiver + to - Propagation Delay Difference (normal operation)	tPHLR - tPLHR	MAX222/232A/233A/242/243		100		ns
		MAX220		225		

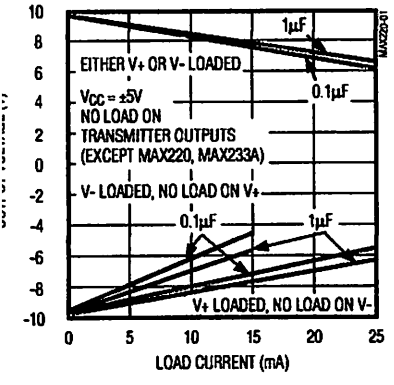
Note 3: MAX243 R2OUT is guaranteed to be low when R2IN is ≥ 0V or is floating.

±5V-Powered, Multichannel RS-232 Drivers/Receivers

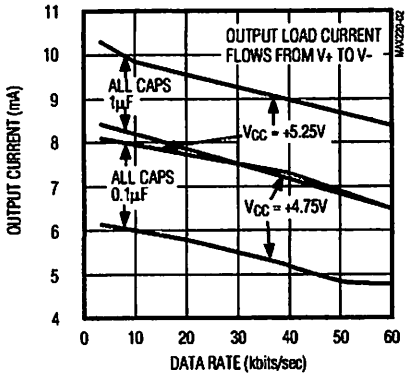
Typical Operating Characteristics

MAX220/MAX222/MAX232A/MAX233A/MAX242/MAX243

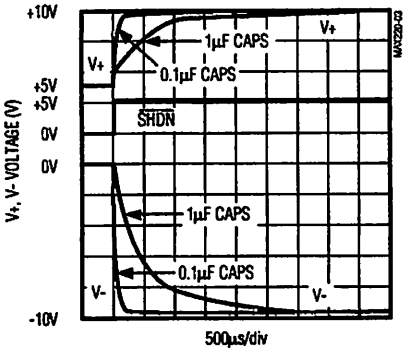
OUTPUT VOLTAGE vs. LOAD CURRENT



AVAILABLE OUTPUT CURRENT vs. DATA RATE



MAX222/MAX242 ON-TIME EXITING SHUTDOWN



ABSOLUTE MAXIMUM RATINGS—MAX223/MAX230–MAX241

VCC ..... -0.3V to +6V
V+ ..... (VCC - 0.3V) to +14V
V- ..... +0.3V to -14V
Input Voltages
TIN ..... -0.3V to (VCC + 0.3V)
RIN ..... ±30V
Output Voltages
TOUT ..... (V+ + 0.3V) to (V- - 0.3V)
ROUT ..... -0.3V to (VCC + 0.3V)
Short-Circuit Duration, TOUT ..... Continuous
Continuous Power Dissipation (TA = +70°C)
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C) 800mW
16-Pin Plastic DIP (derate 10.53mW/°C above +70°C) 842mW
20-Pin Plastic DIP (derate 11.11mW/°C above +70°C) 889mW
24-Pin Narrow Plastic DIP
(derate 13.33mW/°C above +70°C) 1.07W
24-Pin Plastic DIP (derate 9.09mW/°C above +70°C) 500mW
16-Pin Wide SO (derate 9.52mW/°C above +70°C) 762mW
20-Pin Wide SO (derate 10.00mW/°C above +70°C) 800mW
24-Pin Wide SO (derate 11.76mW/°C above +70°C) 941mW
28-Pin Wide SO (derate 12.50mW/°C above +70°C) 1W
44-Pin Plastic FP (derate 11.11mW/°C above +70°C) 889mW
14-Pin CERDIP (derate 9.09mW/°C above +70°C) 727mW
16-Pin CERDIP (derate 10.00mW/°C above +70°C) 800mW
20-Pin CERDIP (derate 11.11mW/°C above +70°C) 889mW
24-Pin Narrow CERDIP
(derate 12.50mW/°C above +70°C) 1W
24-Pin Sidebraze (derate 20.0mW/°C above +70°C) 1.6W
28-Pin SSOP (derate 9.52mW/°C above +70°C) 762mW
Operating Temperature Ranges
MAX2__C__ ..... 0°C to +70°C
MAX2__E__ ..... -40°C to +85°C
MAX2__M__ ..... -55°C to +125°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241

(MAX223/230/232/234/236/237/238/240/241, VCC = +5V ±10; MAX233/MAX235, VCC = 5V ±5%, C1–C4 = 1.0µF; MAX231/MAX239, VCC = 5V ±10%; V+ = 7.5V to 13.2V; TA = TMIN to TMAX; unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to ground		±5.0	±7.3		V
VCC Power-Supply Current	No load, TA = +25°C	MAX232/233		5	10	mA
		MAX223/230/234–238/240/241		7	15	
		MAX231/239		0.4	1	
V+ Power-Supply Current		MAX231		1.8	5	mA
		MAX239		5	15	
Shutdown Supply Current	TA = +25°C	MAX223		15	50	µA
		MAX230/235/236/240/241		1	10	
Input Logic Threshold Low	TIN; EN, $\overline{SHDN}$ (MAX233); $\overline{EN}$, SHDN (MAX230/235–241)				0.8	V
Input Logic Threshold High	TIN		2.0			V
	EN, $\overline{SHDN}$ (MAX223); $\overline{EN}$, SHDN (MAX230/235/236/240/241)		2.4			
Logic Pull-Up Current	TIN = 0V			1.5	200	µA
Receiver Input Voltage Operating Range			-30		30	V

LECTRICAL CHARACTERISTICS—MAX223/MAX230–MAX241 (continued)

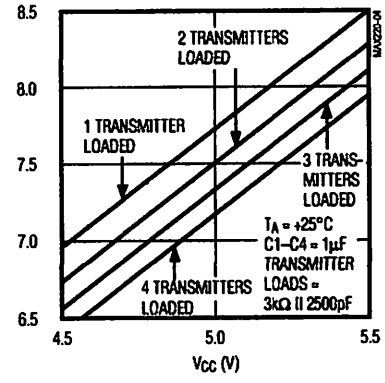
IAX223/230/232/234/236/237/238/240/241, V_{CC} = +5V ±10; MAX233/MAX235, V_{CC} = 5V ±5%, C1–C4 = 1.0µF; MAX231/MAX239, :C = 5V ±10%; V+ = 7.5V to 13.2V; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	CONDITIONS			MIN	TYP	MAX	UNITS
IS-232 Input Threshold Low	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)		0.8	1.2		V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)		0.6	1.5		
IS-232 Input Threshold High	T_A = +25°C, V_{CC} = 5V	Normal operation $\overline{SHDN}$ = 5V (MAX223) SHDN = 0V (MAX235/236/240/241)			1.7	2.4	V
		Shutdown (MAX223) $\overline{SHDN}$ = 0V, EN = 5V ($R4_{IN}$, $R5_{IN}$)			1.5	2.4	
IS-232 Input Hysteresis	V_{CC} = 5V, no hysteresis in shutdown			0.2	0.5	1.0	V
IS-232 Input Resistance	T_A = +25°C, V_{CC} = 5V			3	5	7	kΩ
TL/CMOS Output Voltage Low	I_{OUT} = 1.6mA (MAX231/232/233, I_{OUT} = 3.2mA)					0.4	V
TL/CMOS Output Voltage High	I_{OUT} = -1mA			3.5	V_{CC} - 0.4		V
TL/CMOS Output Leakage Current	0V ≤ R_{OUT} ≤ V_{CC}; EN = 0V (MAX223); $\overline{EN}$ = V_{CC} (MAX235–241)				0.05	±10	µA
Receiver Output Enable Time	Normal operation	MAX223			600		ns
		MAX235/236/239/240/241			400		
Receiver Output Disable Time	Normal operation	MAX223			900		ns
		MAX235/236/239/240/241			250		
Propagation Delay	RS-232 IN to TTL/CMOS OUT, C_L = 150pF	Normal operation			0.5	10	µs
		$\overline{SHDN}$ = 0V (MAX223)	t_{PHLS}		4	40	
			t_{PLHS}		6	40	
Transition Region Slew Rate	MAX223/MAX230/MAX234–241, T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V			3	5.1	30	V/µs
	MAX231/MAX232/MAX233, T_A = +25°C, V_{CC} = 5V, R_L = 3kΩ to 7kΩ, C_L = 50pF to 2500pF, measured from +3V to -3V or -3V to +3V				4	30	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V			300			Ω
Transmitter Output Short-Circuit Current					±10		mA

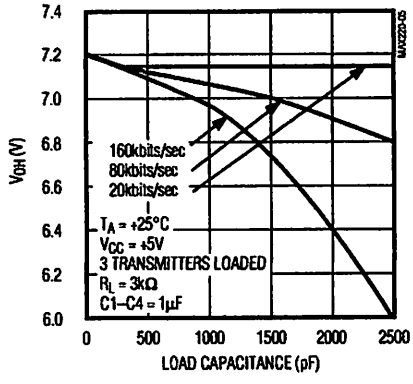
Typical Operating Characteristics

MAX223/MAX230–MAX241

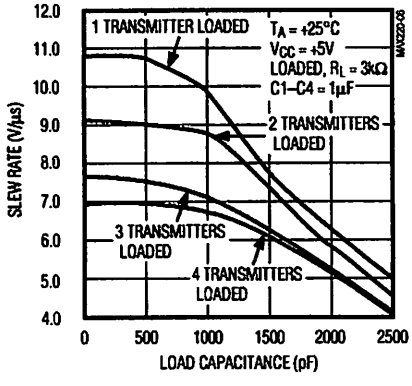
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. V_{CC}



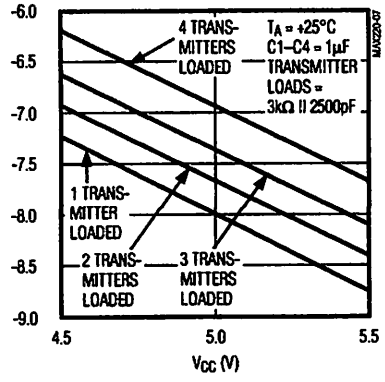
TRANSMITTER OUTPUT VOLTAGE (V_{OH}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



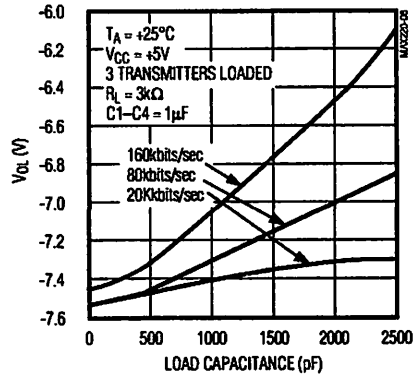
TRANSMITTER SLEW RATE vs. LOAD CAPACITANCE



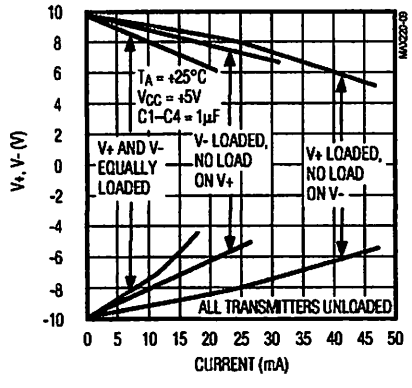
TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. V_{CC}



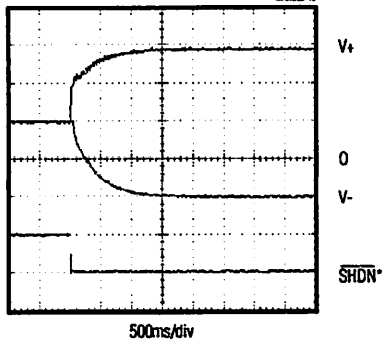
TRANSMITTER OUTPUT VOLTAGE (V_{OL}) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES



TRANSMITTER OUTPUT VOLTAGE (V+, V-) vs. LOAD CURRENT



V+, V- WHEN EXITING SHUTDOWN (1µF CAPACITORS)



*SHUTDOWN POLARITY IS REVERSED FOR NON MAX241 PARTS

+5V-Powered, Multichannel RS-232 Drivers/Receivers

ABSOLUTE MAXIMUM RATINGS—MAX225/MAX244–MAX249

Supply Voltage (V_{CC}) ..... -0.3V to +6V
Input Voltages
T_{IN}, $\overline{ENA}$, $\overline{ENB}$, $\overline{ENR}$, $\overline{ENT}$, $\overline{ENRA}$, $\overline{ENRB}$, $\overline{ENTA}$, $\overline{ENTB}$ ..... -0.3V to (V_{CC} + 0.3V)
R_{IN} ..... ±25V
T_{OUT} (Note 3) ..... ±15V
R_{OUT} ..... -0.3V to (V_{CC} + 0.3V)
Short Circuit (one output at a time)
T_{OUT} to GND ..... Continuous
R_{OUT} to GND ..... Continuous

Continuous Power Dissipation (T_A = +70°C)
28-Pin Wide SO (derate 12.50mW/°C above +70°C) 1W
40-Pin Plastic DIP (derate 11.11mW/°C above +70°C) ... 611mW
44-Pin PLCC (derate 13.33mW/°C above +70°C) 1.07W
Operating Temperature Ranges
MAX225C_ _, MAX24_C_ _ ..... 0°C to +70°C
MAX225E_ _, MAX24_E_ _ ..... -40°C to +85°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Note 4: Input voltage measured with transmitter output in a high-impedance state, shutdown, or V_{CC} = 0V.

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

ELECTRICAL CHARACTERISTICS—MAX225/MAX244–MAX249

(MAX225, V_{CC} = 5.0V ±5%; MAX244–MAX249, V_{CC} = +5.0V ±10%, external capacitors C1–C4 = 1µF; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
RS-232 TRANSMITTERS						
Input Logic Threshold Low				1.4	0.8	V
Input Logic Threshold High			2	1.4		V
Logic Pull-Up/Input Current	Tables 1a–1d	Normal operation		10	50	µA
		Shutdown		±0.01	±1	
Data Rate	Tables 1a–1d, normal operation			120	64	kbits/sec
Output Voltage Swing	All transmitter outputs loaded with 3kΩ to GND		±5	±7.5		V
Output Leakage Current (shutdown)	Tables 1a–1d	$\overline{ENA}$, $\overline{ENB}$, $\overline{ENT}$, $\overline{ENTA}$, $\overline{ENTB}$ = V_{CC}, V_{OUT} = ±15V		±0.01	±25	µA
		V_{CC} = 0V, V_{OUT} = ±15V		±0.01	±25	
Transmitter Output Resistance	V_{CC} = V+ = V- = 0V, V_{OUT} = ±2V (Note 4)		300	10M		Ω
Output Short-Circuit Current	V_{OUT} = 0V		±7	±30		mA
RS-232 RECEIVERS						
RS-232 Input Voltage Operating Range					±25	V
RS-232 Input Threshold Low	V_{CC} = 5V		0.8	1.3		V
RS-232 Input Threshold High	V_{CC} = 5V			1.8	2.4	V
RS-232 Input Hysteresis	V_{CC} = 5V		0.2	0.5	1.0	V
RS-232 Input Resistance			3	5	7	kΩ
TTL/CMOS Output Voltage Low	I_{OUT} = 3.2mA			0.2	0.4	V
TTL/CMOS Output Voltage High	I_{OUT} = -1.0mA		3.5	V_{CC} - 0.2		V
TTL/CMOS Output Short-Circuit Current	Sourcing V_{OUT} = GND		-2	-10		mA
	Shrinking V_{OUT} = V_{CC}		10	30		
TTL/CMOS Output Leakage Current	Normal operation, outputs disabled, Tables 1a–1d, 0V ≤ V_{OUT} ≤ V_{CC}, $\overline{ENR\_}$ = V_{CC}			±0.05	±0.10	µA

LECTRICAL CHARACTERISTICS—MAX225/MAX244–MAX249 (continued)

(MAX225, V_{CC} = 5.0V ±5%; MAX244–MAX249, V_{CC} = +5.0V ±10%, external capacitors C1–C4 = 1µF; T_A = T_{MIN} to T_{MAX}; unless otherwise noted.)

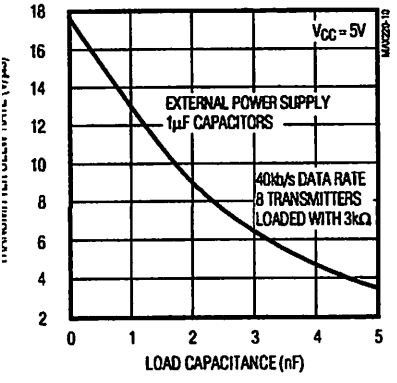
PARAMETER	CONDITIONS		MIN	TYP	MAX	UNITS
POWER SUPPLY AND CONTROL LOGIC						
Operating Supply Voltage		MAX225	4.75		5.25	V
		MAX244–MAX249	4.5		5.5	
V_{CC} Supply Current (normal operation)	No load	MAX225		10	20	mA
		MAX244–MAX249		11	30	
	3kΩ loads on all outputs	MAX225		40		
		MAX244–MAX249		57		
Shutdown Supply Current	T_A = +25°C			8	25	µA
	T_A = T_{MIN} to T_{MAX}				50	
Control Input	Leakage current				±1	µA
	Threshold low			1.4	0.8	V
	Threshold high		2.4	1.4		
AC CHARACTERISTICS						
Transition Slew Rate	C_L = 50pF to 2500pF, R_L = 3kΩ to 7kΩ, V_{CC} = 5V, T_A = +25°C, measured from +3V to -3V or -3V to +3V		5	10	30	V/µs
Transmitter Propagation Delay TLL to RS-232 (normal operation), Figure 1	t_{PHLT}			1.3	3.5	µs
	t_{PLHT}			1.5	3.5	
Receiver Propagation Delay TLL to RS-232 (normal operation), Figure 2	t_{PHLR}			0.6	1.5	µs
	t_{PLHR}			0.6	1.5	
Receiver Propagation Delay TLL to RS-232 (low-power mode), Figure 2	t_{PHLS}			0.6	10	µs
	t_{PLHS}			3.0	10	
Transmitter + to - Propagation Delay Difference (normal operation)	t_{PHLT} - t_{PLHT}			350		ns
Receiver + to - Propagation Delay Difference (normal operation)	t_{PHLR} - t_{PLHR}			350		ns
Receiver-Output Enable Time, Figure 3	t_{ER}			100	500	ns
Receiver-Output Disable Time, Figure 3	t_{DR}			100	500	ns
Transmitter Enable Time	t_{ET}	MAX246–MAX249 (excludes charge-pump start-up)		5		µs
		MAX225/MAX245–MAX249 (includes charge-pump start-up)		10		ms
Transmitter Disable Time, Figure 4	t_{DT}			100		ns

Note 5: The 300Ω minimum specification complies with EIA/TIA-232E, but the actual resistance when in shutdown mode or V_{CC} = 0V is 10MΩ as is implied by the leakage specification.

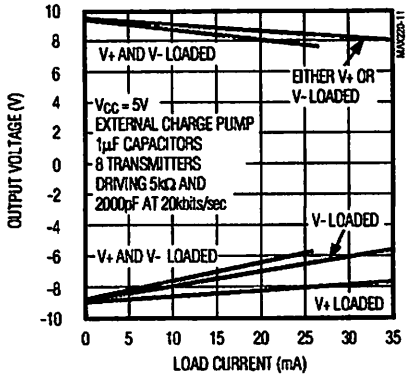
Typical Operating Characteristics

MAX225/MAX244–MAX249

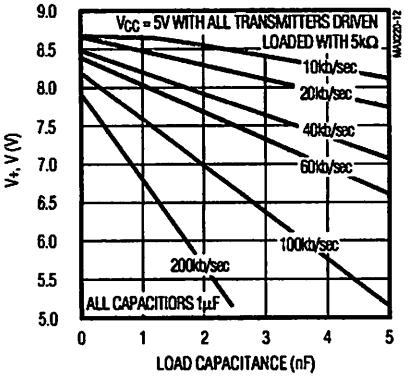
TRANSMITTER SLEW RATE vs. LOAD CAPACITANCE

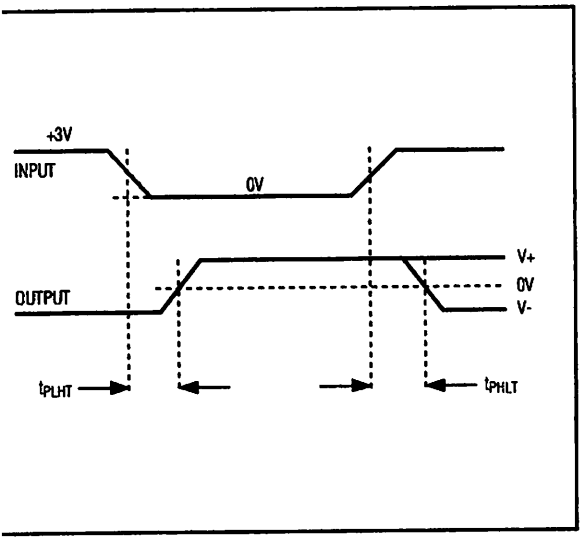


OUTPUT VOLTAGE vs. LOAD CURRENT FOR V+ AND V-



TRANSMITTER OUTPUT VOLTAGE (V+, V-) vs. LOAD CAPACITANCE AT DIFFERENT DATA RATES





gure 1. Transmitter Propagation-Delay Timing

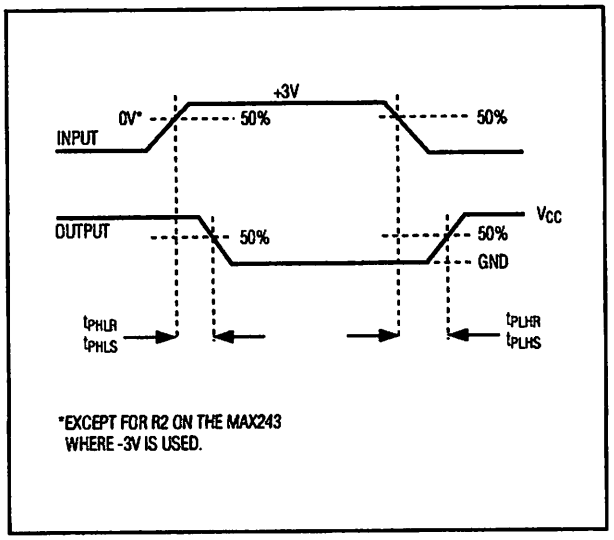


Figure 2. Receiver Propagation-Delay Timing

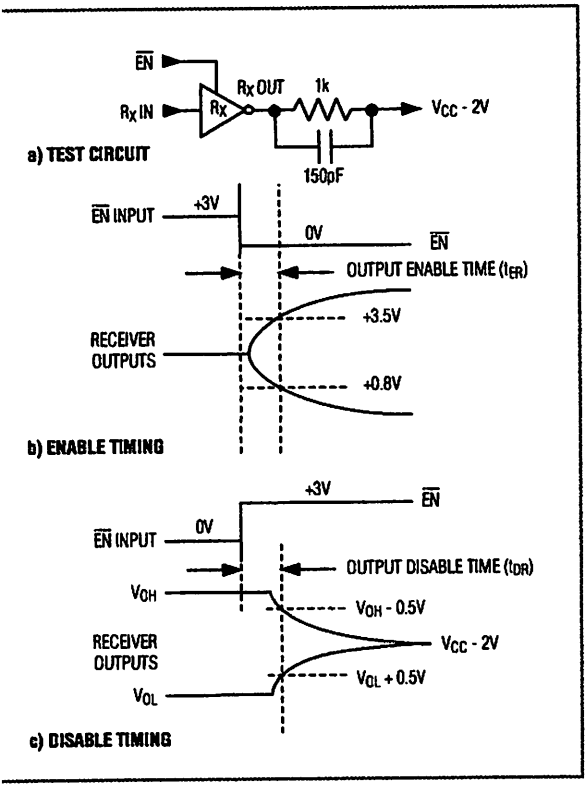


Figure 3. Receiver-Output Enable and Disable Timing

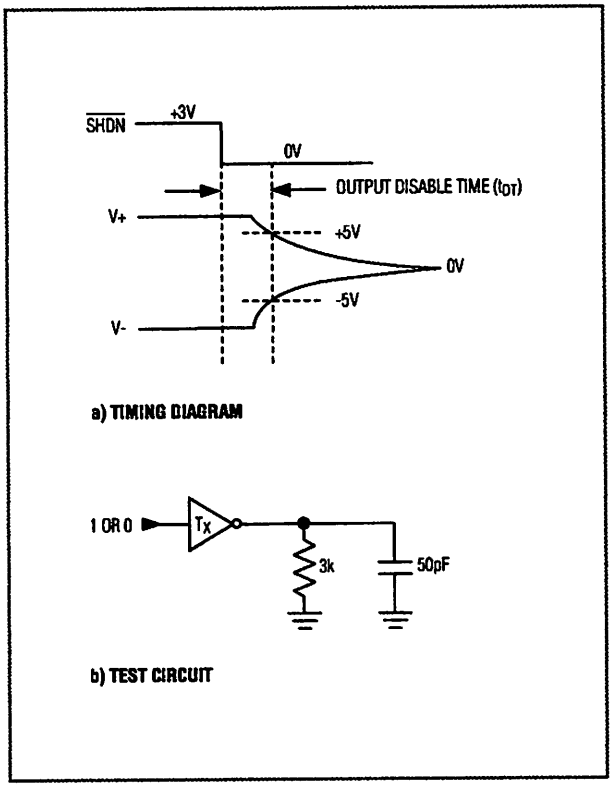


Figure 4. Transmitter-Output Disable Timing

·5V-Powered, Multichannel RS-232)rivers/Receivers

ıble 1a. MAX245 Control Pin Configurations

$\overline{ENT}$	$\overline{ENR}$	OPERATION STATUS	TRANSMITTERS	RECEIVERS
0	0	Normal Operation	All Active	All Active
0	1	Normal Operation	All Active	All 3-State
1	0	Shutdown	All 3-State	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State

ıble 1b. MAX245 Control Pin Configurations

$\overline{ENT}$	$\overline{ENR}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1–TA4	TB1–TB4	RA1–RA5	RB1–RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All Active	RA1–RA4 3-State, RA5 Active	RB1–RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All 3-State	All Low-Power Receive Mode	All Low-Power Receive Mode
1	1	Shutdown	All 3-State	All 3-State	RA1–RA4 3-State, RA5 Low-Power Receive Mode	RB1–RB4 3-State, RB5 Low-Power Receive Mode

ıble 1c. MAX246 Control Pin Configurations

$\overline{ENA}$	$\overline{ENB}$	OPERATION STATUS	TRANSMITTERS		RECEIVERS	
			TA1–TA4	TB1–TB4	RA1–RA5	RB1–RB5
0	0	Normal Operation	All Active	All Active	All Active	All Active
0	1	Normal Operation	All Active	All 3-State	All Active	RB1–RB4 3-State, RB5 Active
1	0	Shutdown	All 3-State	All Active	RA1–RA4 3-State, RA5 Active	All Active
1	1	Shutdown	All 3-State	All 3-State	RA1–RA4 3-State, RA5 Low-Power Receive Mode	RB1–RB4 3-State, RA5 Low-Power Receive Mode

Table 1d. MAX247/MAX248/MAX249 Control Pin Configurations

NTA	$\overline{ENTB}$	$\overline{ENRA}$	$\overline{ENRB}$	OPERATION STATUS		TRANSMITTERS		RECEIVERS	
					MAX247	TA1–TA4	TB1–TB4	RA1–RA4	RB1–RB5
					MAX248	TA1–TA4	TB1–TB4	RA1–RA4	RB1–RB4
					MAX249	TA1–TA3	TB1–TB3	RA1–RA5	RB1–RB5
0	0	0	0	Normal Operation		All Active	All Active	All Active	All Active
0	0	0	1	Normal Operation		All Active	All Active	All Active	All 3-State, except RB5 stays active on MAX247
0	0	1	0	Normal Operation		All Active	All Active	All 3-State	All Active
0	0	1	1	Normal Operation		All Active	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
0	1	0	0	Normal Operation		All Active	All 3-State	All Active	All Active
0	1	0	1	Normal Operation		All Active	All 3-State	All Active	All 3-State, except RB5 stays active on MAX247
0	1	1	0	Normal Operation		All Active	All 3-State	All 3-State	All Active
0	1	1	1	Normal Operation		All Active	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247
1	0	0	0	Normal Operation		All 3-State	All Active	All Active	All Active
1	0	0	1	Normal Operation		All 3-State	All Active	All Active	All 3-State, except RB5 stays active on MAX247
1	0	1	0	Normal Operation		All 3-State	All Active	All 3-State	All Active
1	0	1	1	Normal Operation		All 3-State	All Active	All 3-State	All 3-State, except RB5 stays active on MAX247
1	1	0	0	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	Low-Power Receive Mode
1	1	0	1	Shutdown		All 3-State	All 3-State	Low-Power Receive Mode	All 3-State, except RB5 stays active on MAX247
1	1	1	0	Shutdown		All 3-State	All 3-State	All 3-State	Low-Power Receive Mode
1	1	1	1	Shutdown		All 3-State	All 3-State	All 3-State	All 3-State, except RB5 stays active on MAX247

ɔminal 5kΩ values. The receivers implement Type 1 ıterpretation of the fault conditions of V.28 and IA/TIA-232E.

he receiver input hysteresis is typically 0.5V with a uaranteed minimum of 0.2V. This produces clear out-ut transitions with slow-moving input signals, even ith moderate amounts of noise and ringing. The ıceiver propagation delay is typically 600ns and is dependent of input swing direction.

Low-Power Receive Mode

he low-power receive-mode feature of the MAX223, IAX242, and MAX245–MAX249 puts the IC into shut-own mode but still allows it to receive information. This important for applications where systems are periodi-ally awakened to look for activity. Using low-power ıceive mode, the system can still receive a signal that ill activate it on command and prepare it for communi-ation at faster data rates. This operation conserves ystem power.

Negative Threshold—MAX243

he MAX243 is pin compatible with the MAX232A, differ-g only in that RS-232 cable fault protection is removed n one of the two receiver inputs. This means that control ıes such as CTS and RTS can either be driven or left ɔating without interrupting communication. Different ables are not needed to interface with different pieces of quipment.

he input threshold of the receiver without cable fault rotection is -0.8V rather than +1.4V. Its output goes ositive only if the input is connected to a control line ıat is actively driven negative. If not driven, it defaults ı the 0 or "OK to send" state. Normally, the MAX243's ther receiver (+1.4V threshold) is used for the data line D or RD), while the negative threshold receiver is con-ected to the control line (DTR, DTS, CTS, RTS, etc.).

ıther members of the RS-232 family implement the ptional cable fault protection as specified by EIA/TIA-32E specifications. This means a receiver output goes igh whenever its input is driven negative, left floating, r shorted to ground. The high output tells the serial ommunications IC to stop sending data. To avoid this, ıe control lines must either be driven or connected ith jumpers to an appropriate positive voltage level.

Shutdown—MAX222–MAX242

On the MAX222, MAX235, MAX236, MAX240, and MAX241, all receivers are disabled during shutdown. On the MAX223 and MAX242, two receivers continue to operate in a reduced power mode when the chip is in shutdown. Under these conditions, the propagation delay increases to about 2.5µs for a high-to-low input transition. When in shutdown, the receiver acts as a CMOS inverter with no hysteresis. The MAX223 and MAX242 also have a receiver output enable input ($\overline{EN}$ for the MAX242 and EN for the MAX223) that allows receiver output control independent of $\overline{SHDN}$ (SHDN for MAX241). With all other devices, $\overline{SHDN}$ (SHDN for MAX241) also disables the receiver outputs.

The MAX225 provides five transmitters and five receivers, while the MAX245 provides ten receivers and eight transmitters. Both devices have separate receiver and transmitter-enable controls. The charge pumps turn off and the devices shut down when a logic high is applied to the ENT input. In this state, the supply current drops to less than 25µA and the receivers continue to operate in a low-power receive mode. Driver outputs enter a high-impedance state (three-state mode). On the MAX225, all five receivers are controlled by the $\overline{ENR}$ input. On the MAX245, eight of the receiver outputs are controlled by the $\overline{ENR}$ input, while the remaining two receivers (RA5 and RB5) are always active. RA1–RA4 and RB1–RB4 are put in a three-state mode when $\overline{ENR}$ is a logic high.

Receiver and Transmitter Enable Control Inputs

The MAX225 and MAX245–MAX249 feature transmitter and receiver enable controls.

The receivers have three modes of operation: full-speed receive (normal active), three-state (disabled), and low-power receive (enabled receivers continue to function at lower data rates). The receiver enable inputs control the full-speed receive and three-state modes. The transmitters have two modes of operation: full-speed transmit (normal active) and three-state (disabled). The transmitter enable inputs also control the shutdown mode. The device enters shutdown mode when all transmitters are disabled. Enabled receivers function in the low-power receive mode when in shutdown.

ables 1a–1d define the control states. The MAX244 as no control pins and is not included in these tables.

he MAX246 has ten receivers and eight drivers with vo control pins, each controlling one side of the evice. A logic high at the A-side control input ($\overline{ENA}$) auses the four A-side receivers and drivers to go into three-state mode. Similarly, the B-side control input $\overline{ENB}$) causes the four B-side drivers and receivers to o into a three-state mode. As in the MAX245, one A-de and one B-side receiver (RA5 and RB5) remain ctive at all times. The entire device is put into shut-own mode when both the A and B sides are disabled $\overline{ENA}$ = $\overline{ENB}$ = +5V).

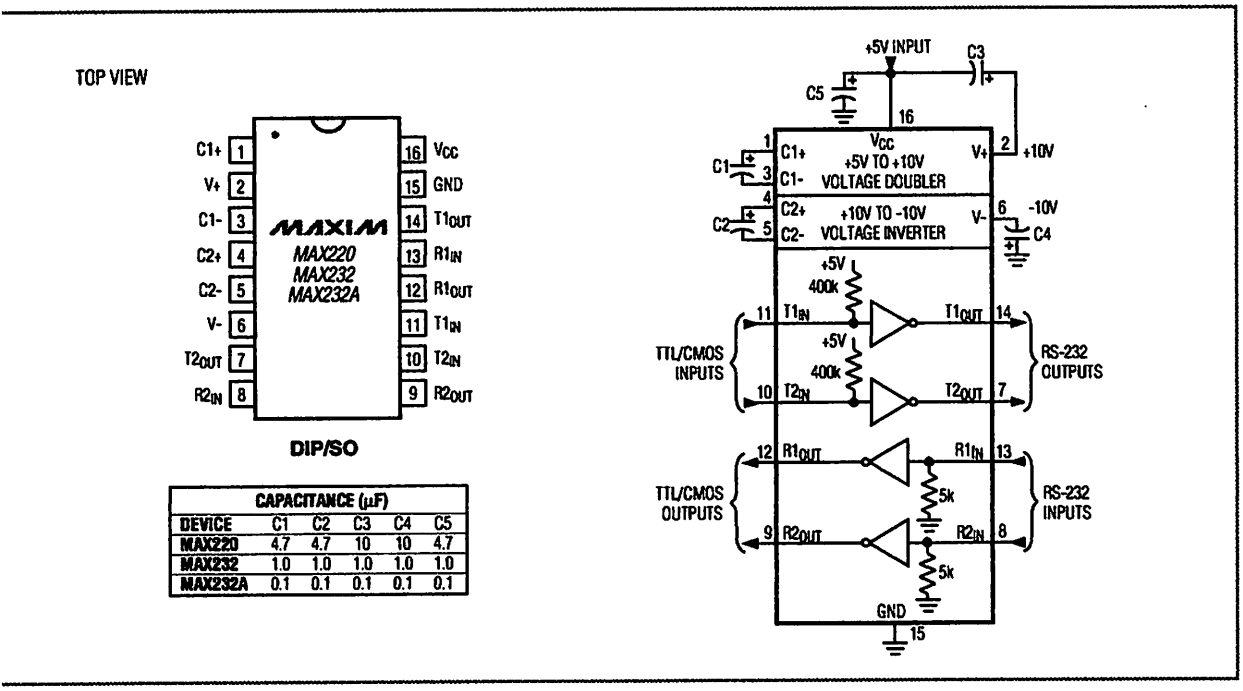
he MAX247 provides nine receivers and eight drivers ith four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver nable inputs each control four receiver outputs. The $\overline{NTA}$ and $\overline{ENTB}$ transmitter enable inputs each control ur drivers. The ninth receiver (RB5) is always active. he device enters shutdown mode with a logic high on oth $\overline{ENTA}$ and $\overline{ENTB}$.

he MAX248 provides eight receivers and eight drivers ith four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver nable inputs each control four receiver outputs. The $\overline{NTA}$ and $\overline{ENTB}$ transmitter enable inputs control four rivers each. This part does not have an always-active ceiver. The device enters shutdown mode and trans-itters go into a three-state mode with a logic high on oth $\overline{ENTA}$ and $\overline{ENTB}$.

The MAX249 provides ten receivers and six drivers with four control pins. The $\overline{ENRA}$ and $\overline{ENRB}$ receiver enable inputs each control five receiver outputs. The $\overline{ENTA}$ and $\overline{ENTB}$ transmitter enable inputs control three drivers each. There is no always-active receiver. The device enters shutdown mode and transmitters go into a three-state mode with a logic high on both $\overline{ENTA}$ and $\overline{ENTB}$. In shutdown mode, active receivers operate in a low-power receive mode at data rates up to 20kbits/sec.

Applications Information

Figures 5 through 25 show pin configurations and typical operating circuits. In applications that are sensitive to power-supply noise, V_{CC} should be decoupled to ground with a capacitor of the same value as C1 and C2 connected as close as possible to the device.



CAPACITANCE (µF)					
DEVICE	C1	C2	C3	C4	C5
MAX220	4.7	4.7	10	10	4.7
MAX232	1.0	1.0	1.0	1.0	1.0
MAX232A	0.1	0.1	0.1	0.1	0.1

Figure 5. MAX220/MAX232/MAX232A Pin Configuration and Typical Operating Circuit

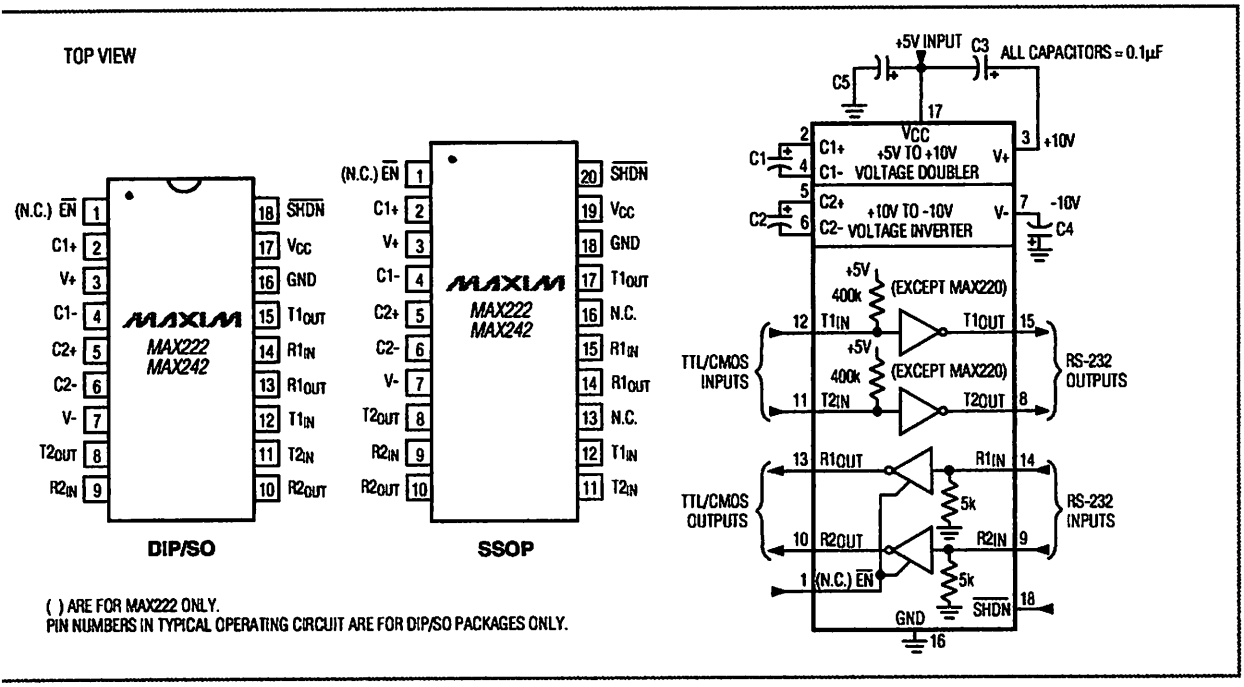
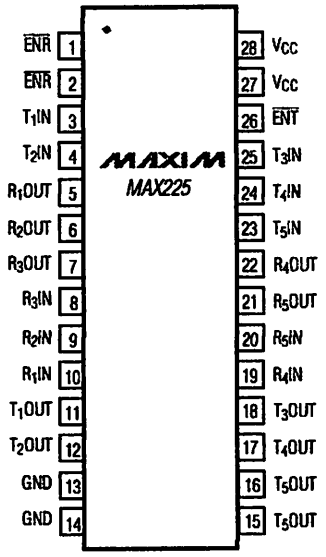


Figure 6. MAX222/MAX242 Pin Configurations and Typical Operating Circuit

TOP VIEW



SO

MAX225 FUNCTIONAL DESCRIPTION

5 RECEIVERS
5 TRANSMITTERS
2 CONTROL PINS
1 RECEIVER ENABLE ($\overline{ENR}$)
1 TRANSMITTER ENABLE ($\overline{ENT}$)

PINS ($\overline{ENR}$, GND, V_{CC}, T_5OUT) ARE INTERNALLY CONNECTED. CONNECT EITHER OR BOTH EXTERNALLY. T_5OUT IS A SINGLE DRIVER.

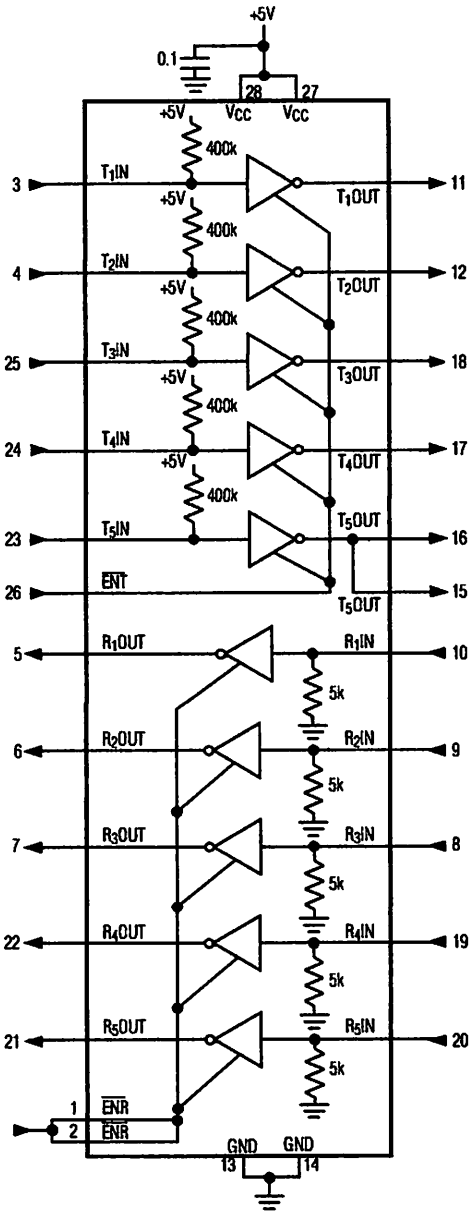
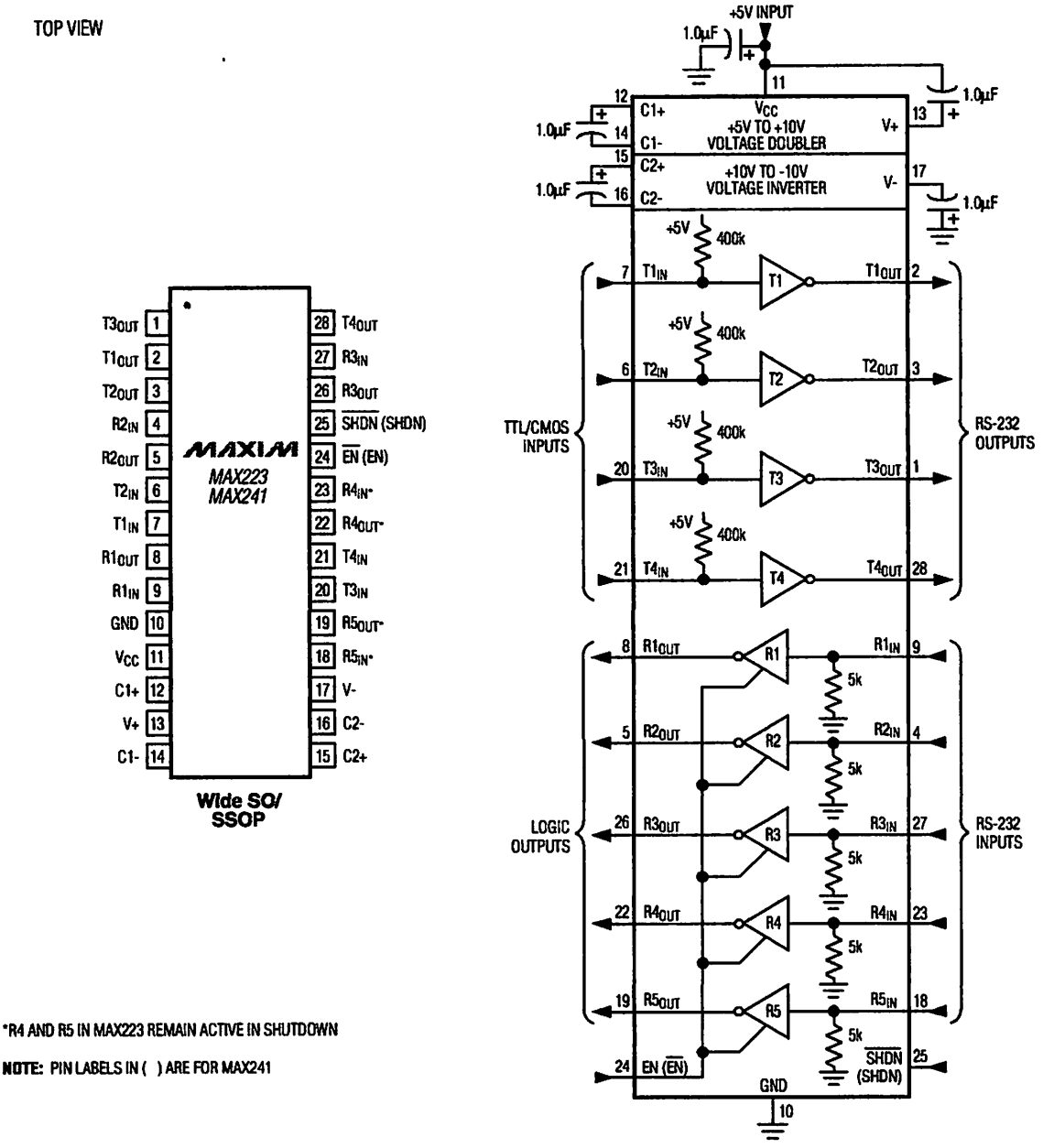


Figure 7. MAX225 Pin Configuration and Typical Operating Circuit



*R4 AND R5 IN MAX223 REMAIN ACTIVE IN SHUTDOWN

NOTE: PIN LABELS IN ( ) ARE FOR MAX241

Figure 8. MAX223/MAX241 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

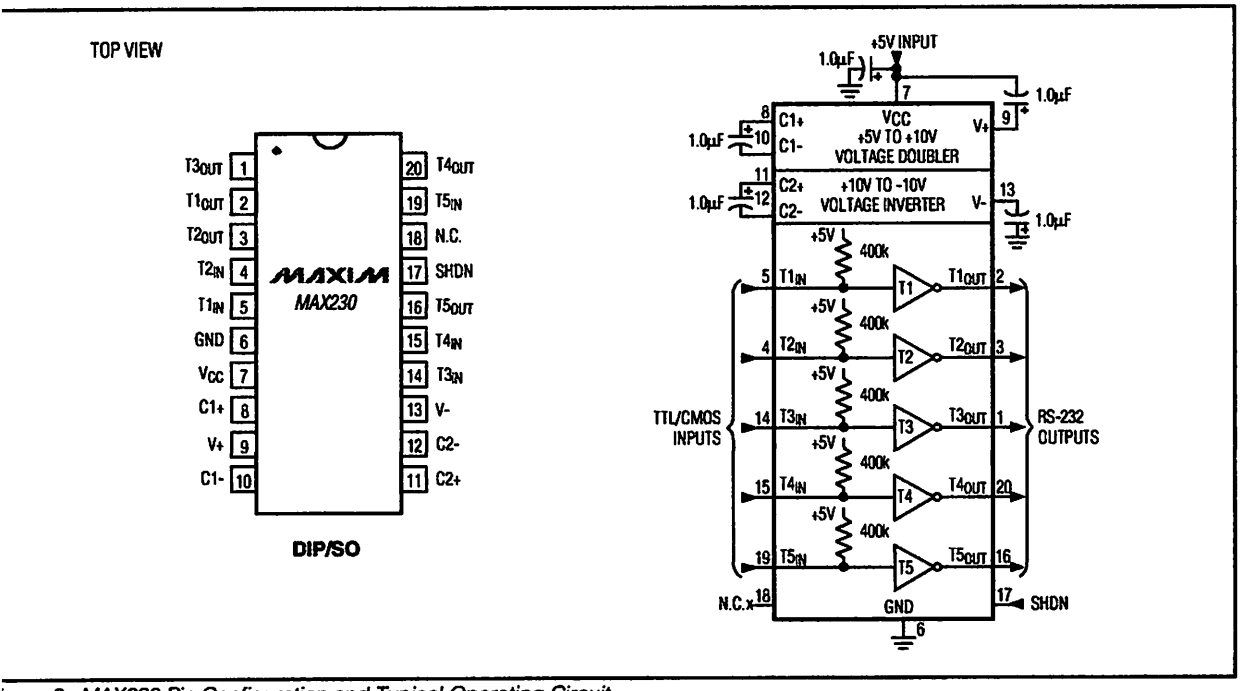


Figure 9. MAX230 Pin Configuration and Typical Operating Circuit

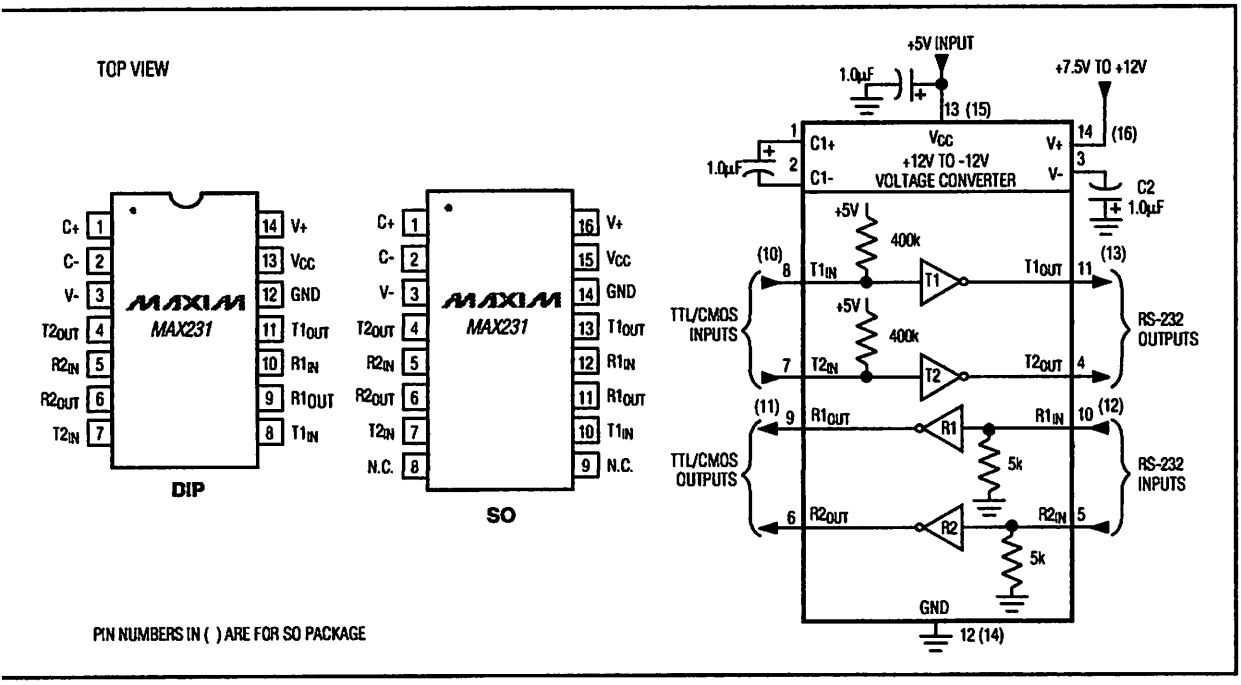


Figure 10. MAX231 Pin Configurations and Typical Operating Circuit

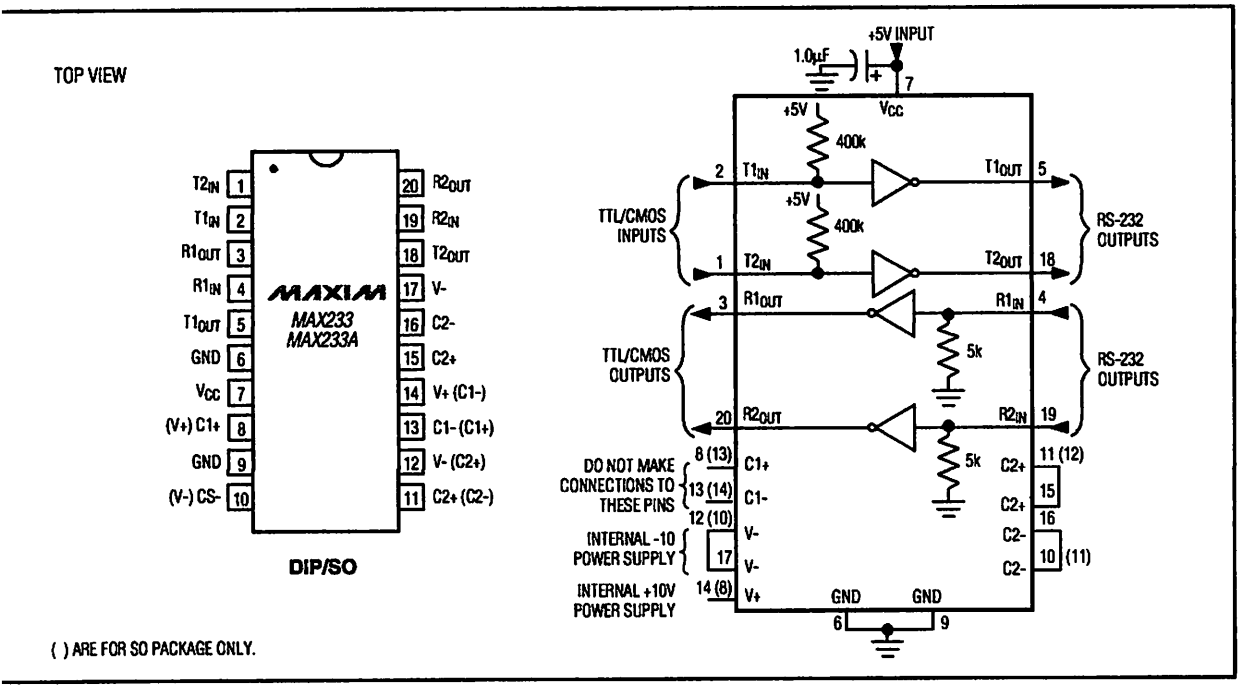


Figure 11. MAX233/MAX233A Pin Configuration and Typical Operating Circuit

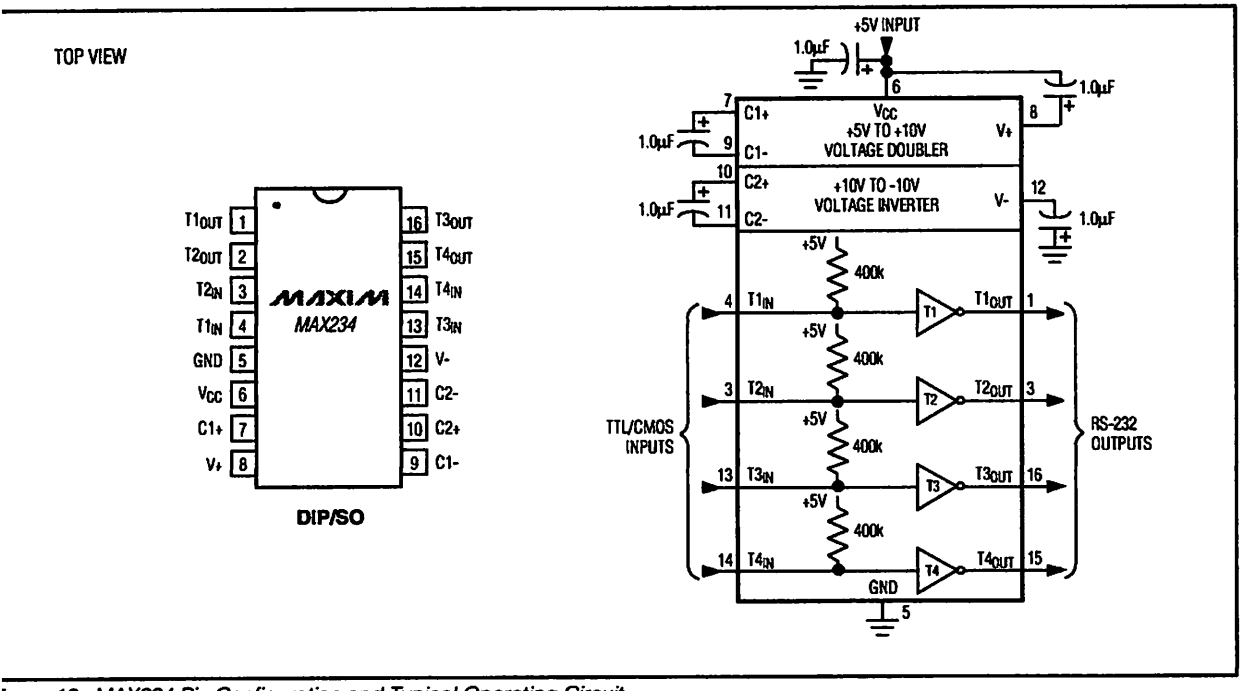
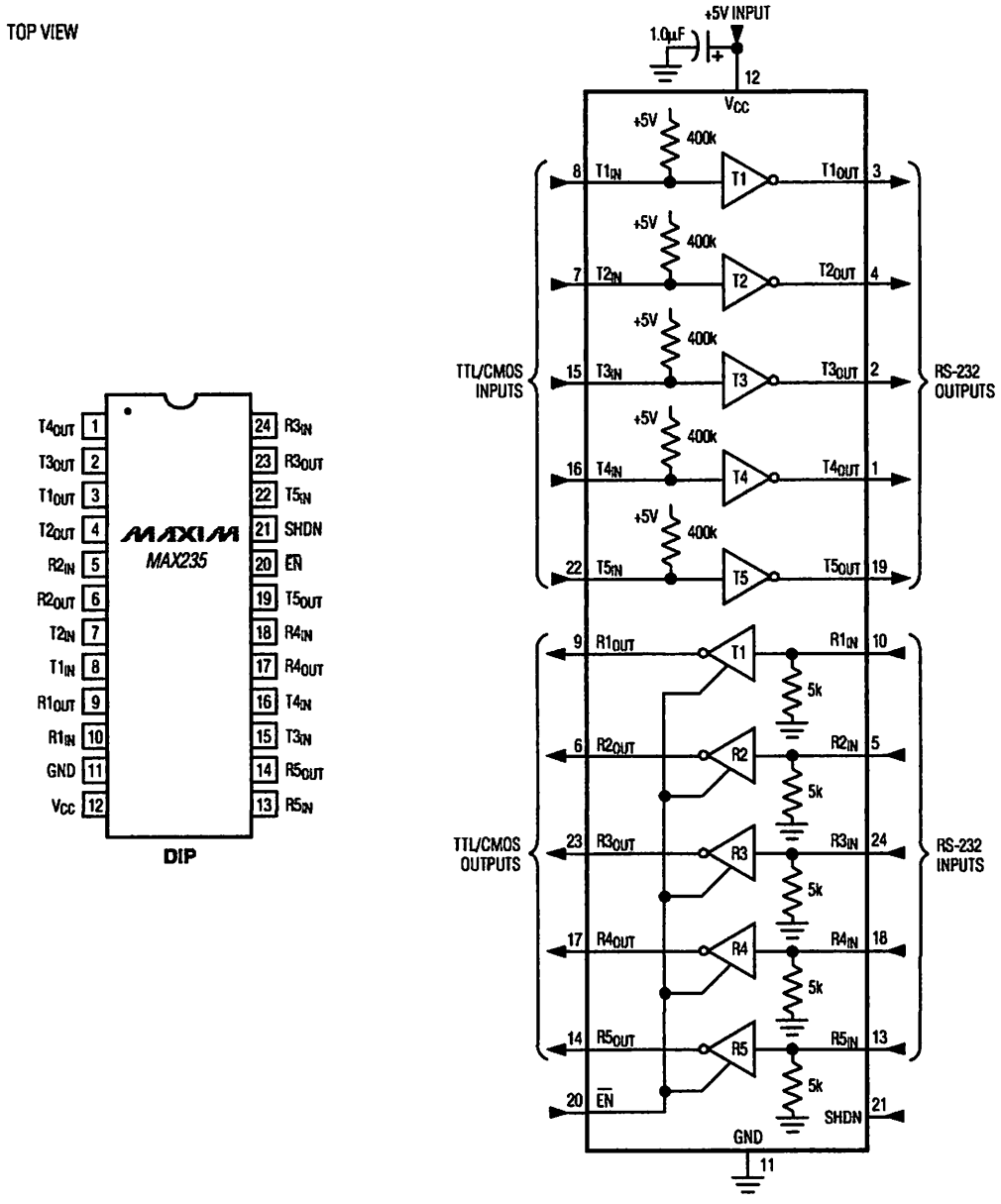


Figure 12. MAX234 Pin Configuration and Typical Operating Circuit



igure 13. MAX235 Pin Configuration and Typical Operating Circuit

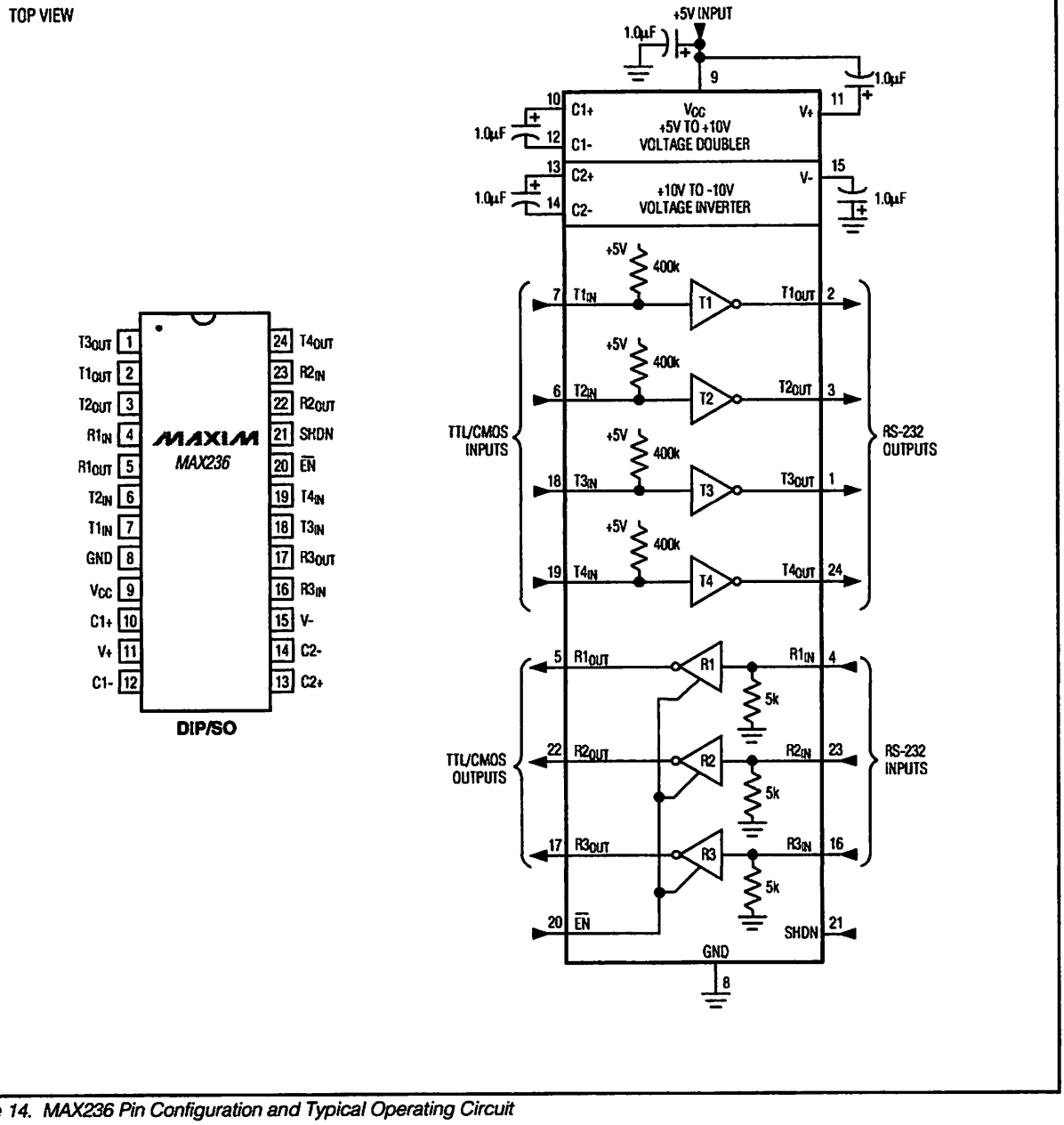


Figure 14. MAX236 Pin Configuration and Typical Operating Circuit

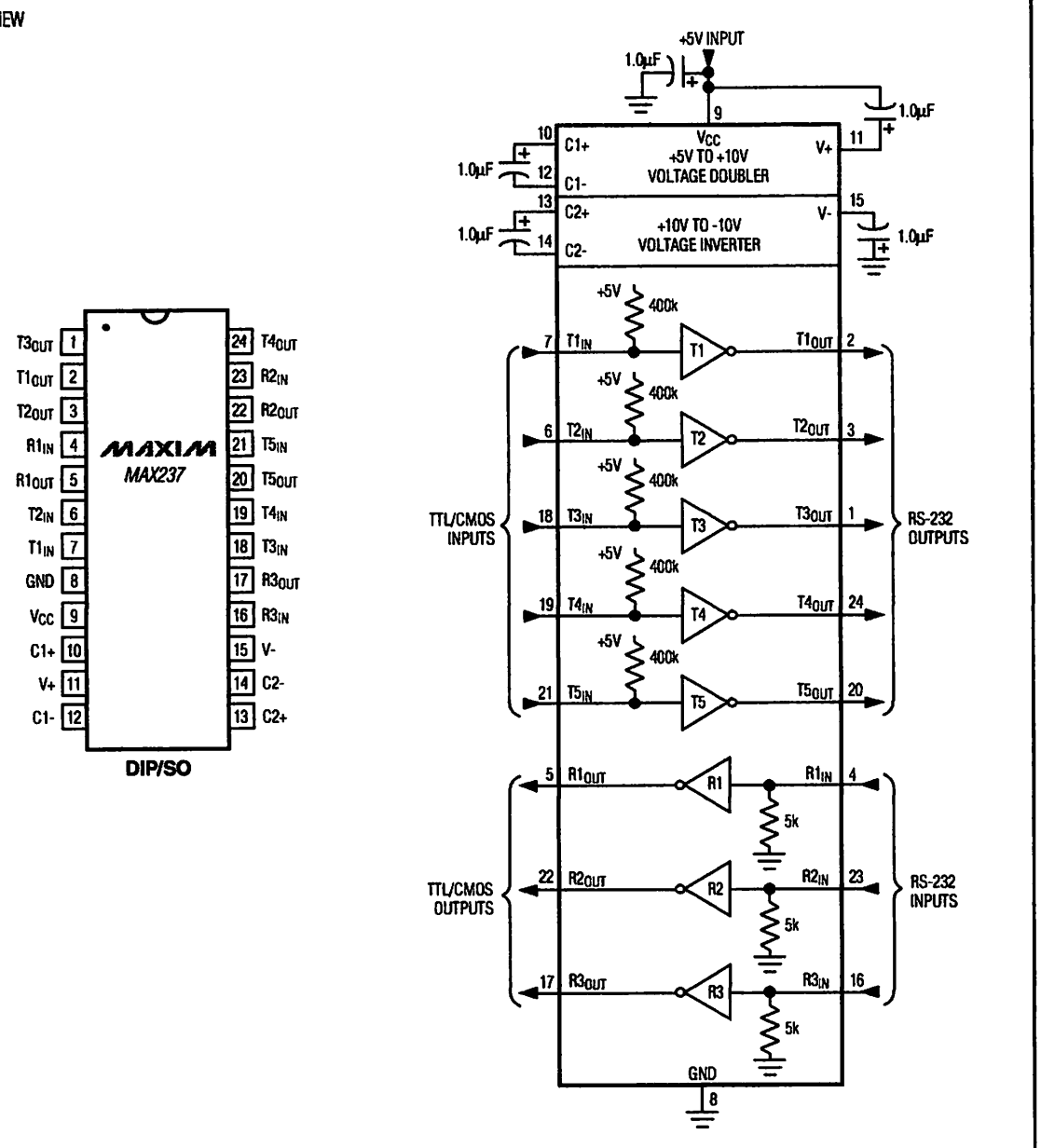


Figure 15. MAX237 Pin Configuration and Typical Operating Circuit

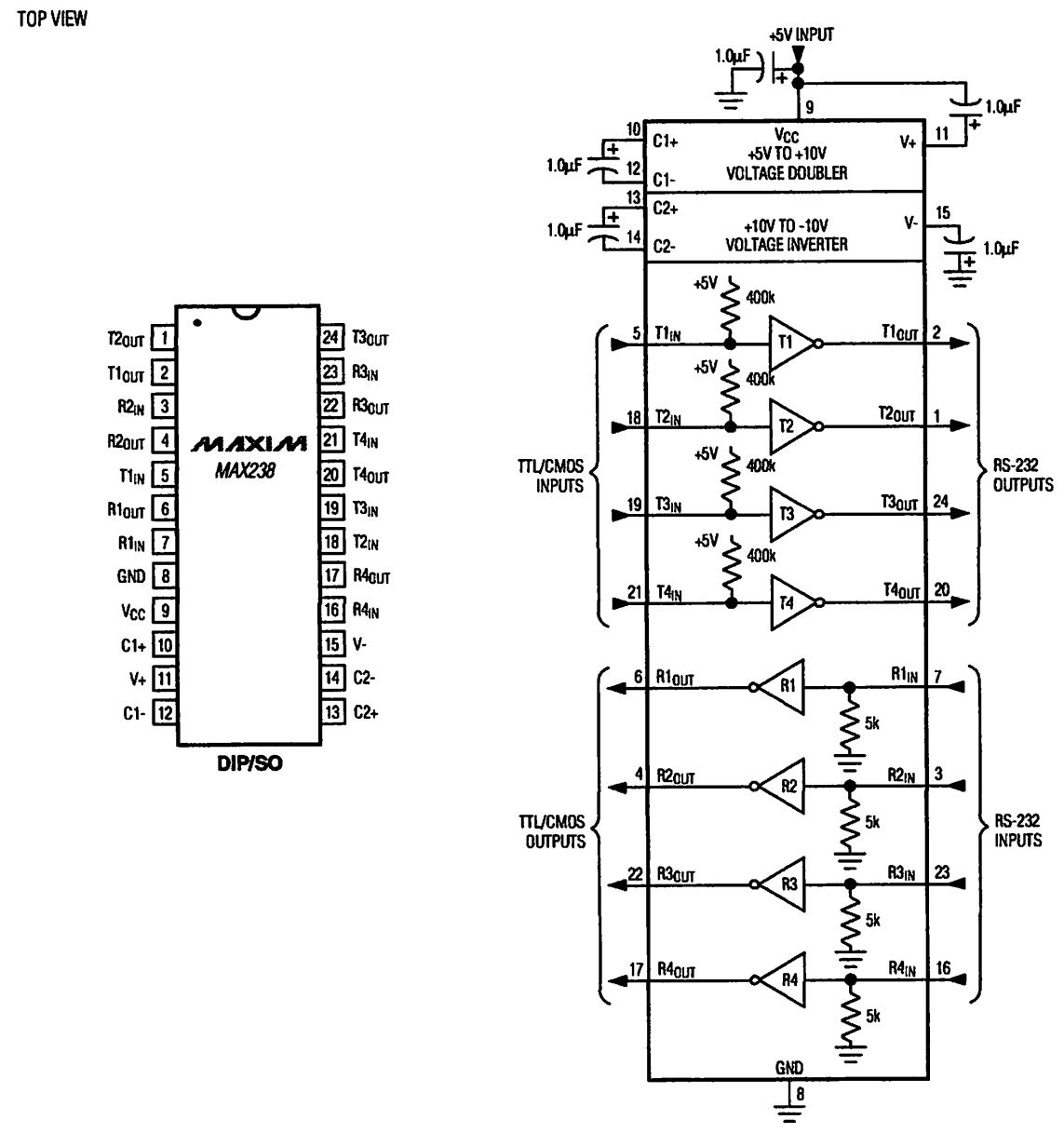


Figure 16. MAX238 Pin Configuration and Typical Operating Circuit

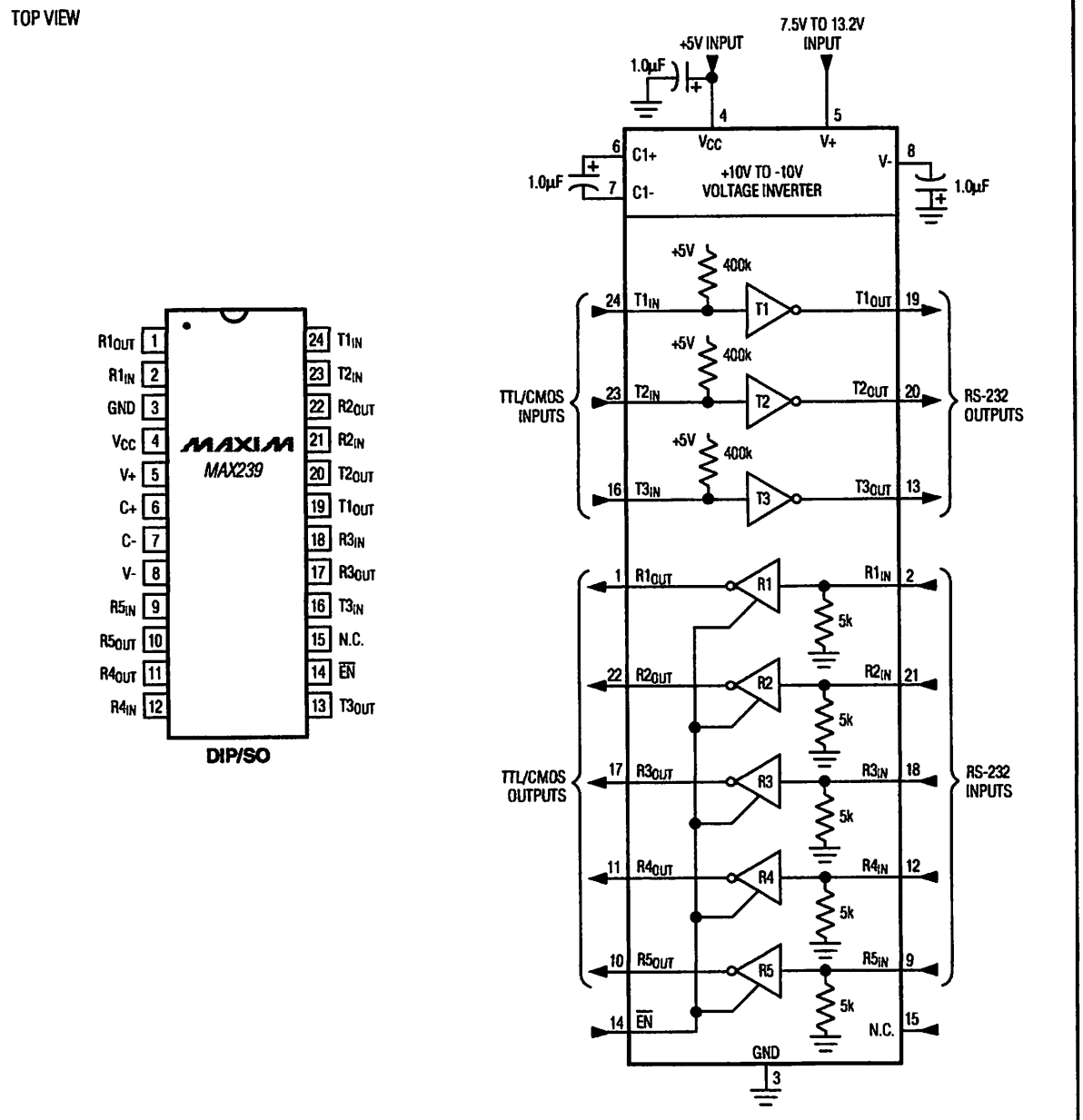


Figure 17. MAX239 Pin Configuration and Typical Operating Circuit

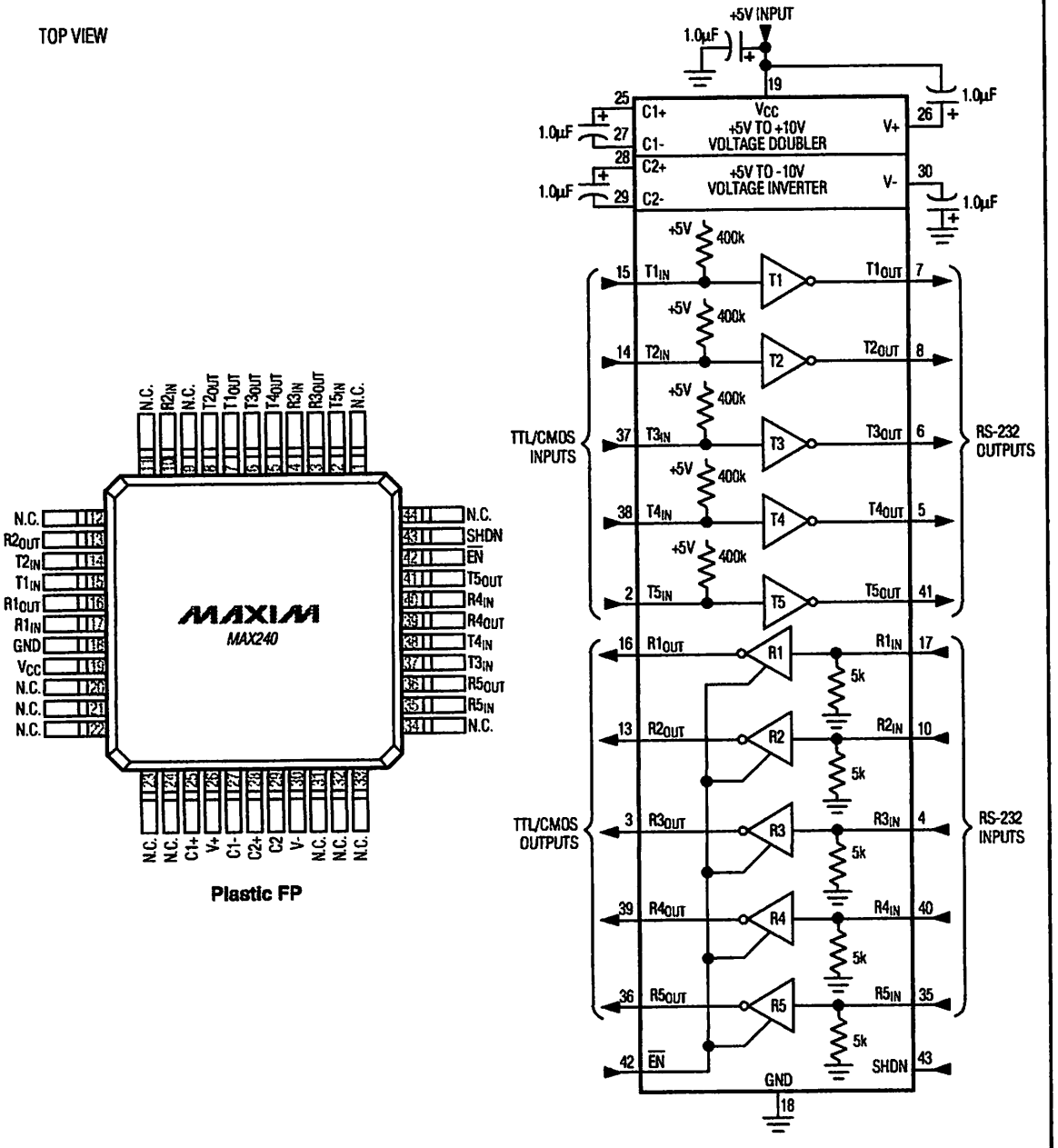
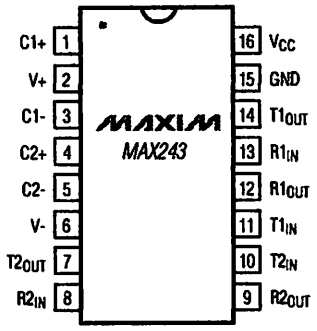
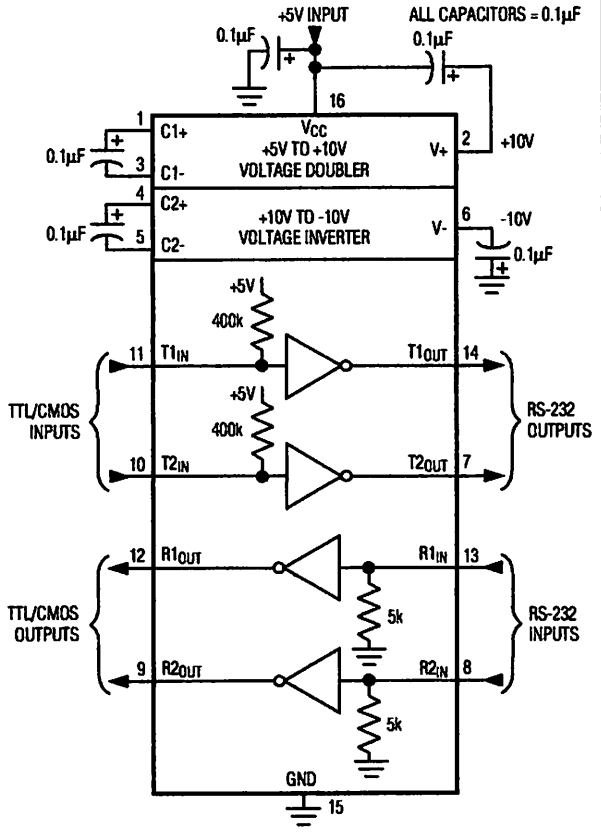


Figure 18. MAX240 Pin Configuration and Typical Operating Circuit

TOP VIEW

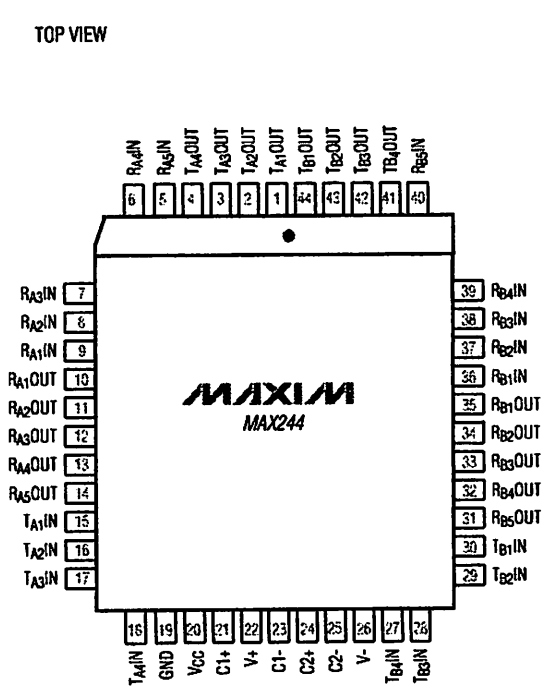


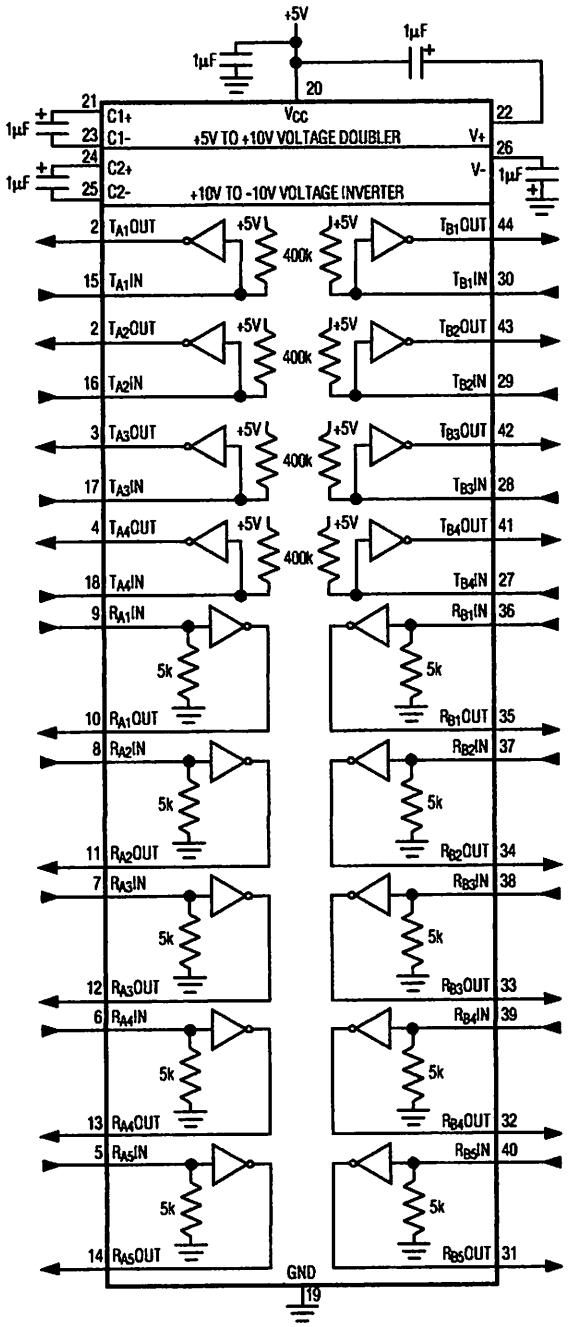
DIP/SO



RECEIVER INPUT	R1 OUTPUT	R2 OUTPUT
≤-3 V	HIGH	HIGH
OPEN	HIGH	LOW
≥+3V	LOW	LOW

Figure 19. MAX243 Pin Configuration and Typical Operating Circuit





MAX249 FUNCTIONAL DESCRIPTION

10 RECEIVERS
- 5 A-SIDE RECEIVER
- 5 B-SIDE RECEIVER

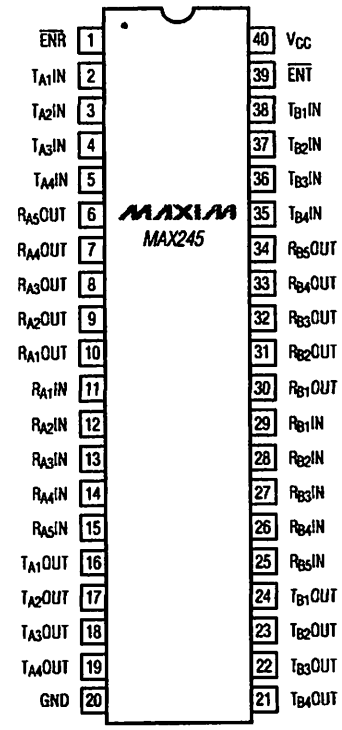
8 TRANSMITTERS
- 4 A-SIDE TRANSMITTERS
- 4 B-SIDE TRANSMITTERS

NO CONTROL PINS

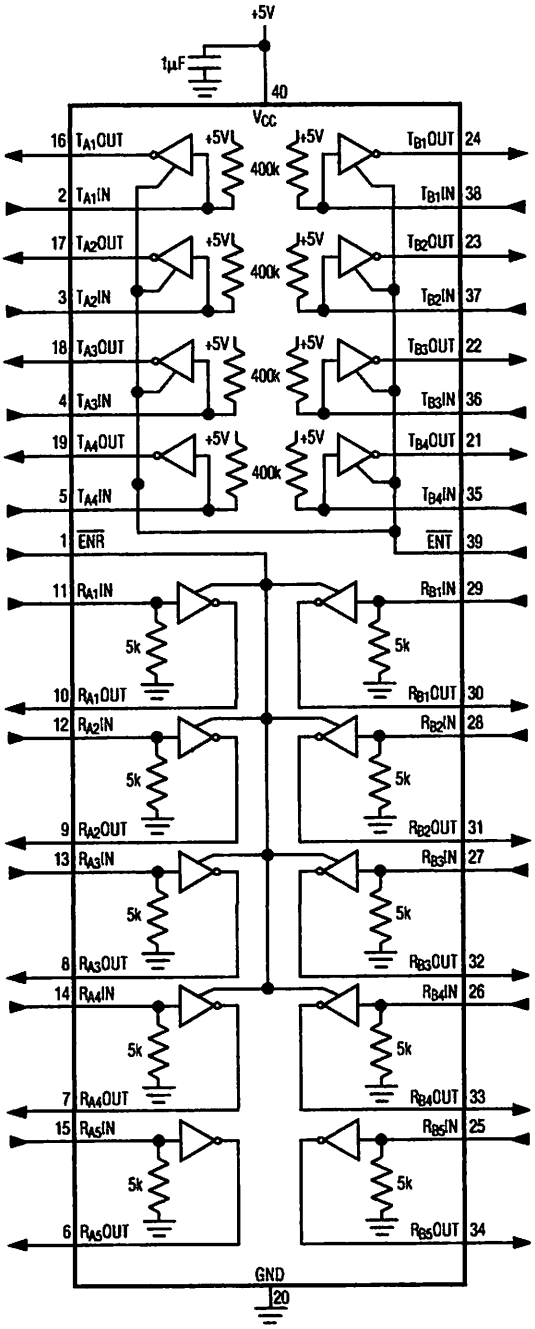
Figure 20. MAX244 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

TOP VIEW



DIP



MAX245 FUNCTIONAL DESCRIPTION

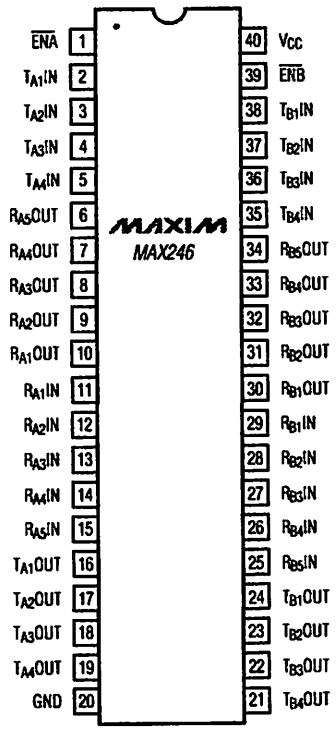
10 RECEIVERS
- 5 A-SIDE RECEIVERS (RA5 ALWAYS ACTIVE)
- 5 B-SIDE RECEIVERS (RB5 ALWAYS ACTIVE)

8 TRANSMITTTERS
- 4 A-SIDE TRANSMITTERS

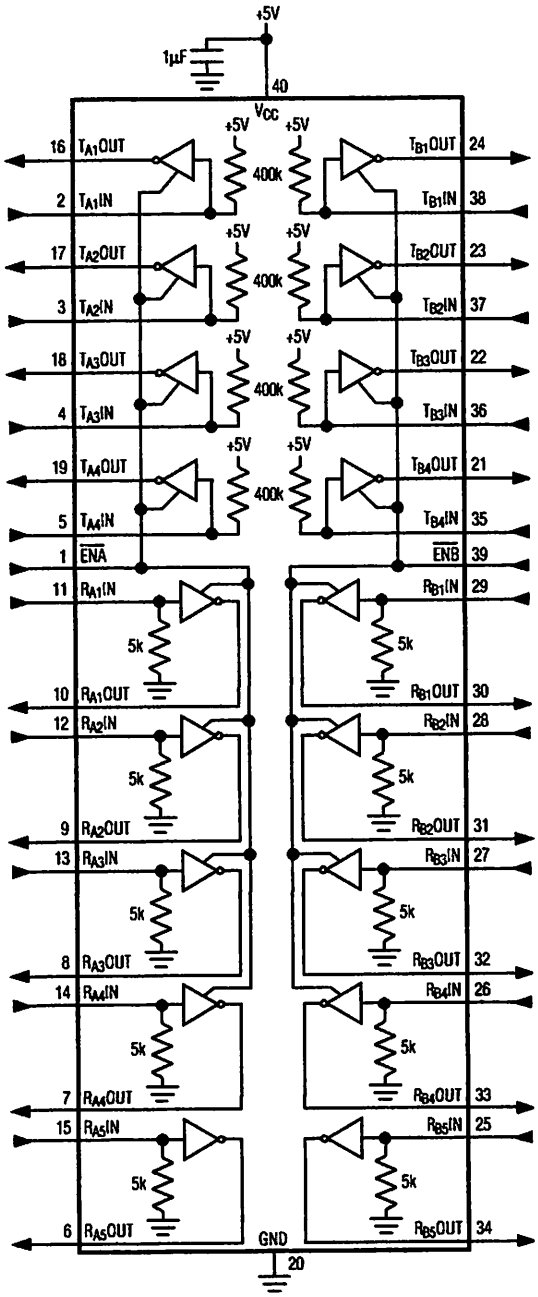
2 CONTROL PINS
- 1 RECEIVER ENABLE ($\overline{ENR}$)
- 1 TRANSMITTER ENABLE ($\overline{ENT}$)

Figure 21. MAX245 Pin Configuration and Typical Operating Circuit

TOP VIEW



DIP



MAX246 FUNCTIONAL DESCRIPTION

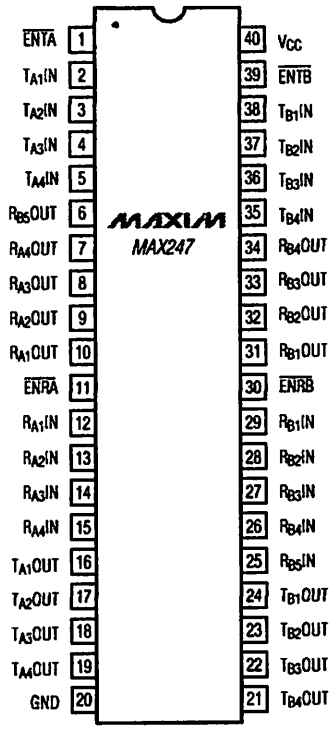
10 RECEIVERS
- 5 A-SIDE RECEIVERS (R_{A5} ALWAYS ACTIVE)
- 5 B-SIDE RECEIVERS (R_{B5} ALWAYS ACTIVE)

8 TRANSMITTERS
- 4 A-SIDE TRANSMITTERS
- 4 B-SIDE TRANSMITTERS

2 CONTROL PINS
- ENABLE A-SIDE ($\overline{ENA}$)
- ENABLE B-SIDE ($\overline{ENB}$)

Figure 22. MAX246 Pin Configuration and Typical Operating Circuit

TOP VIEW



DIP

MAX247 FUNCTIONAL DESCRIPTION

9 RECEIVERS
- 4 A-SIDE RECEIVERS
- 5 B-SIDE RECEIVERS (R_{B5} ALWAYS ACTIVE)

8 TRANSMITTERS
- 4 A-SIDE TRANSMITTERS
- 4 B-SIDE TRANSMITTERS

4 CONTROL PINS
- ENABLE RECEIVER A-SIDE ($\overline{ENRA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENRB}$)
- ENABLE RECEIVER A-SIDE ($\overline{ENTA}$)
- ENABLE RECEIVERr B-SIDE ($\overline{ENTB}$)

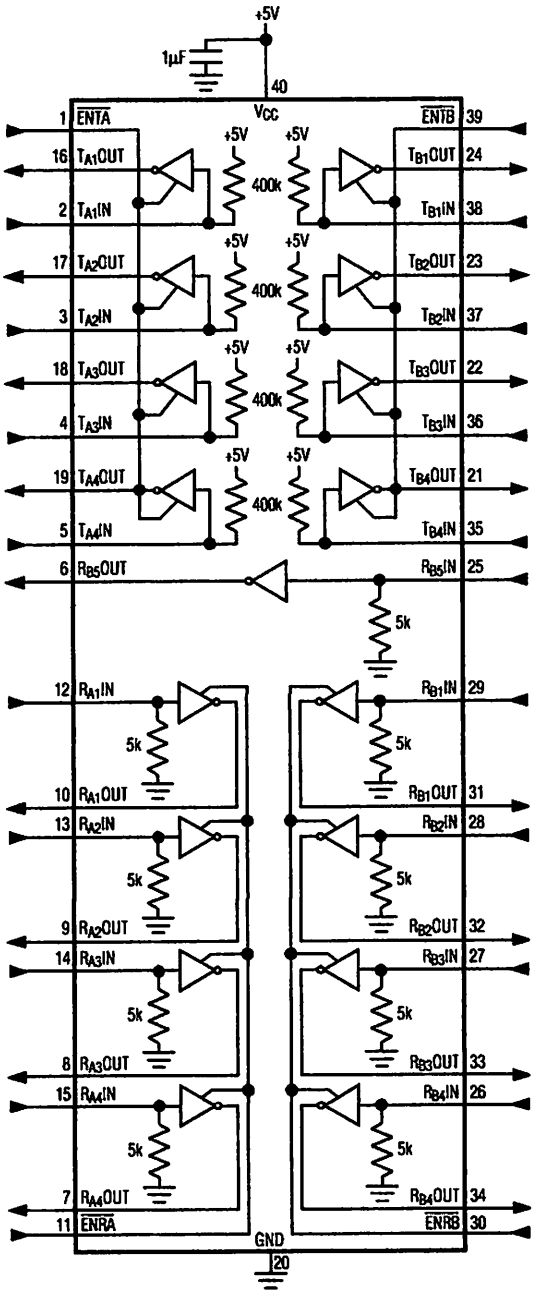
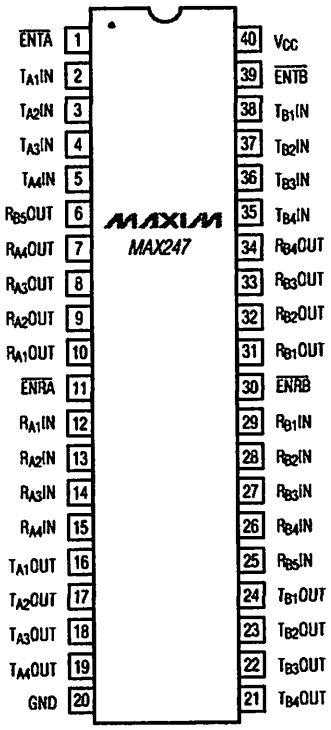


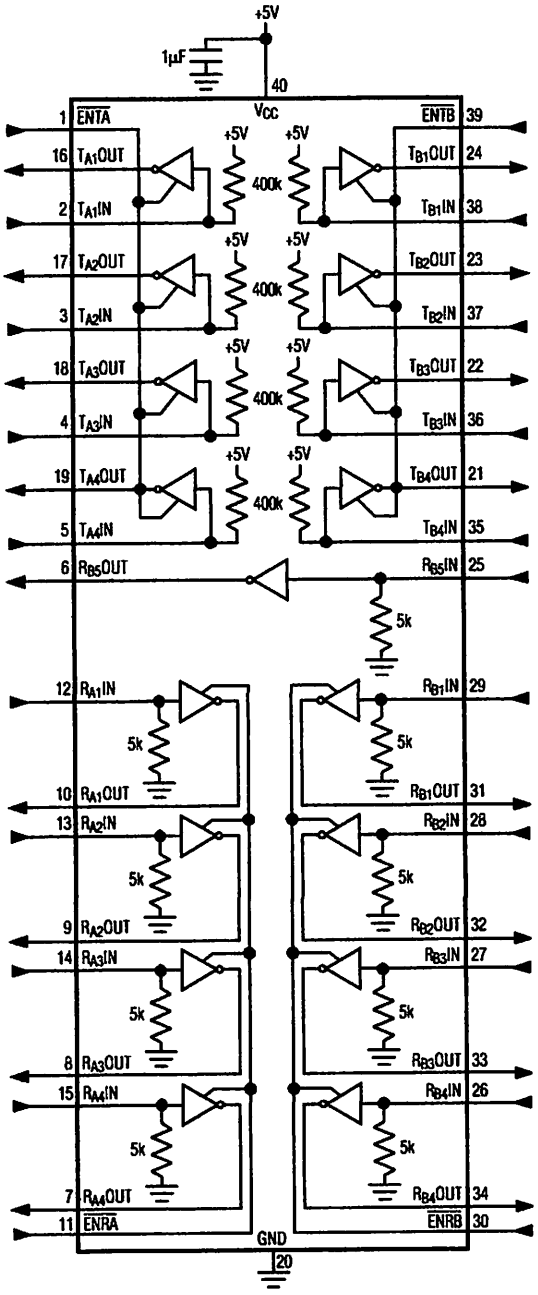
Figure 23. MAX247 Pin Configuration and Typical Operating Circuit

-5V-Powered, Multichannel RS-232 Drivers/Receivers

TOP VIEW



DIP



MAX247 FUNCTIONAL DESCRIPTION

9 RECEIVERS
- 4 A-SIDE RECEIVERS
- 5 B-SIDE RECEIVERS (R_{B5} ALWAYS ACTIVE)

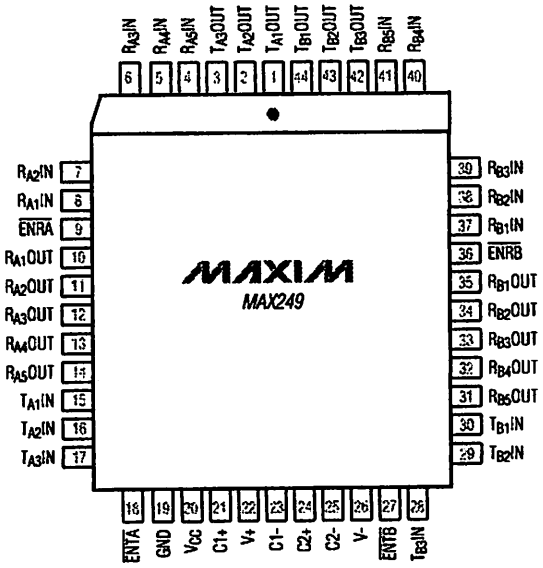
8 TRANSMITTERS
- 4 A-SIDE TRANSMITTERS
- 4 B-SIDE TRANSMITTERS

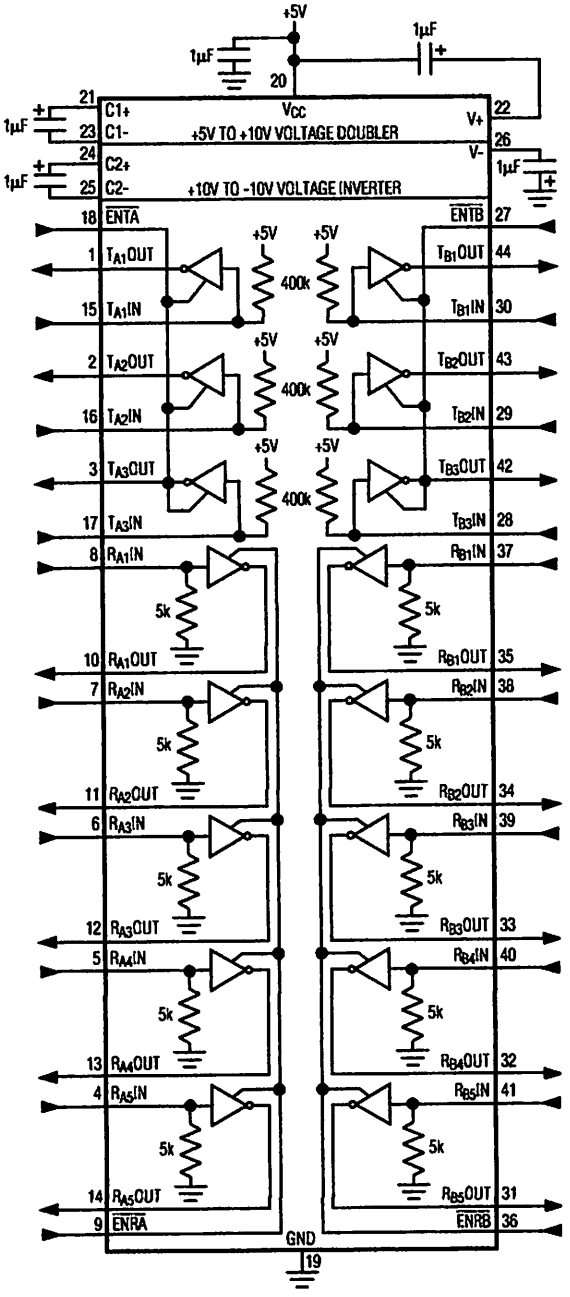
4 CONTROL PINS
- ENABLE RECEIVER A-SIDE ($\overline{ENRA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENRB}$)
- ENABLE RECEIVER A-SIDE ($\overline{ENTA}$)
- ENABLE RECEIVERr B-SIDE ($\overline{ENTB}$)

Figure 23. MAX247 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

TOP VIEW





MAX249 FUNCTIONAL DESCRIPTION

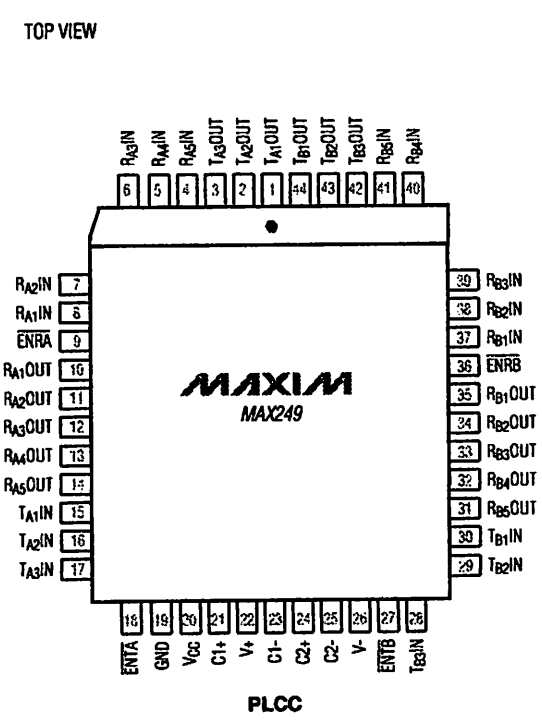
10 RECEIVERS
- 5 A-SIDE RECEIVERS
- 5 B-SIDE RECEIVERS

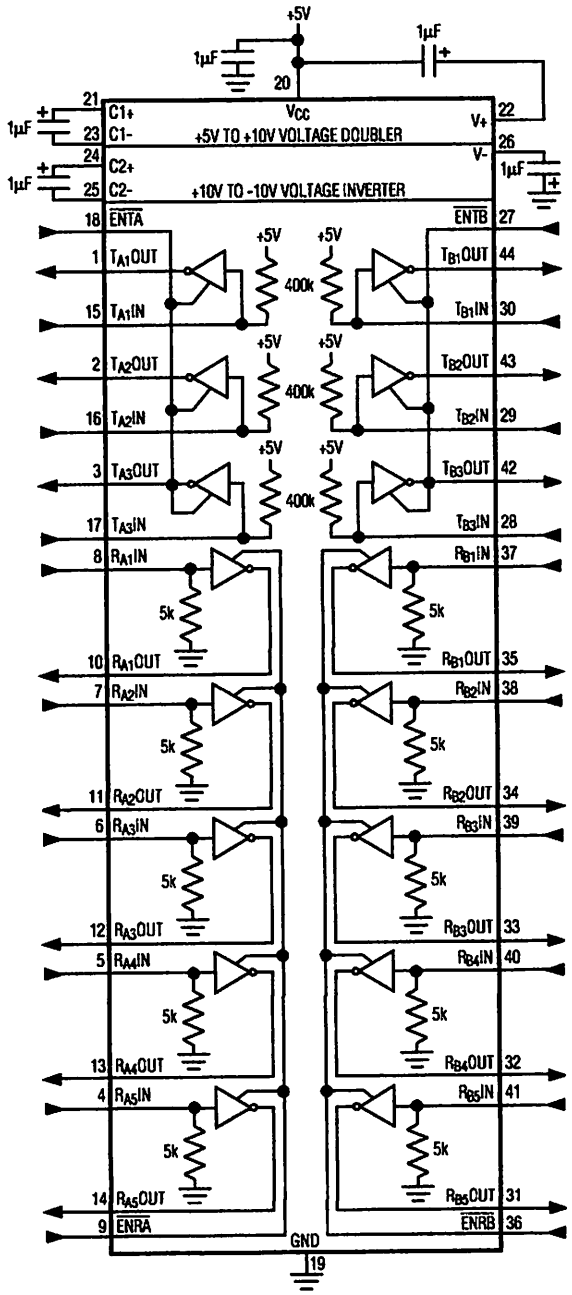
6 TRANSMITTERS
- 3 A-SIDE TRANSMITTERS
- 3 B-SIDE TRANSMITTERS

4 CONTROL PINS
- ENABLE RECEIVER A-SIDE ($\overline{ENRA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENRB}$)
- ENABLE RECEIVER A-SIDE ($\overline{ENTA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENTB}$)

Figure 25. MAX249 Pin Configuration and Typical Operating Circuit

-5V-Powered, Multichannel RS-232 Drivers/Receivers





MAX249 FUNCTIONAL DESCRIPTION

10 RECEIVERS
- 5 A-SIDE RECEIVERS
- 5 B-SIDE RECEIVERS

6 TRANSMITTERS
- 3 A-SIDE TRANSMITTERS
- 3 B-SIDE TRANSMITTERS

4 CONTROL PINS
- ENABLE RECEIVER A-SIDE ($\overline{ENRA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENRB}$)
- ENABLE RECEIVER A-SIDE ($\overline{ENTA}$)
- ENABLE RECEIVER B-SIDE ($\overline{ENTB}$)

Figure 25. MAX249 Pin Configuration and Typical Operating Circuit

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
MAX222CPN	0°C to +70°C	18 Plastic DIP
MAX222CWN	0°C to +70°C	18 Wide SO
MAX222C/D	0°C to +70°C	Dice*
MAX222EPN	-40°C to +85°C	18 Plastic DIP
MAX222EWN	-40°C to +85°C	18 Wide SO
MAX222EJN	-40°C to +85°C	18 CERDIP
MAX222MJN	-55°C to +125°C	18 CERDIP
MAX223CAI	0°C to +70°C	28 SSOP
MAX223CWI	0°C to +70°C	28 Wide SO
MAX223C/D	0°C to +70°C	Dice*
MAX223EAI	-40°C to +85°C	28 SSOP
MAX223EWI	-40°C to +85°C	28 Wide SO
MAX225CWI	0°C to +70°C	28 Wide SO
MAX225EWI	-40°C to +85°C	28 Wide SO
MAX230CPP	0°C to +70°C	20 Plastic DIP
MAX230CWP	0°C to +70°C	20 Wide SO
MAX230C/D	0°C to +70°C	Dice*
MAX230EPP	-40°C to +85°C	20 Plastic DIP
MAX230EWP	-40°C to +85°C	20 Wide SO
MAX230EJP	-40°C to +85°C	20 CERDIP
MAX230MJP	-55°C to +125°C	20 CERDIP
MAX231CPD	0°C to +70°C	14 Plastic DIP
MAX231CWE	0°C to +70°C	16 Wide SO
MAX231CJD	0°C to +70°C	14 CERDIP
MAX231C/D	0°C to +70°C	Dice*
MAX231EPD	-40°C to +85°C	14 Plastic DIP
MAX231EWE	-40°C to +85°C	16 Wide SO
MAX231EJD	-40°C to +85°C	14 CERDIP
MAX231MJD	-55°C to +125°C	14 CERDIP
MAX232CPE	0°C to +70°C	16 Plastic DIP
MAX232CSE	0°C to +70°C	16 Narrow SO
MAX232CWE	0°C to +70°C	16 Wide SO
MAX232C/D	0°C to +70°C	Dice*
MAX232EPE	-40°C to +85°C	16 Plastic DIP
MAX232ESE	-40°C to +85°C	16 Narrow SO
MAX232EWE	-40°C to +85°C	16 Wide SO
MAX232EJE	-40°C to +85°C	16 CERDIP
MAX232MJE	-55°C to +125°C	16 CERDIP
MAX232MLP	-55°C to +125°C	20 LCC
MAX232ACPE	0°C to +70°C	16 Plastic DIP
MAX232ACSE	0°C to +70°C	16 Narrow SO
MAX232ACWE	0°C to +70°C	16 Wide SO
MAX232AC/D	0°C to +70°C	Dice*
MAX232AEPE	-40°C to +85°C	16 Plastic DIP
MAX232AESE	-40°C to +85°C	16 Narrow SO
MAX232AEWE	-40°C to +85°C	16 Wide SO
MAX232AEJE	-40°C to +85°C	16 CERDIP
MAX232AMJE	-55°C to +125°C	16 CERDIP
MAX232AMLP	-55°C to +125°C	20 LCC
MAX233CPP	0°C to +70°C	20 Plastic DIP
MAX233EPP	-40°C to +85°C	20 Plastic DIP
MAX233ACPP	0°C to +70°C	20 Plastic DIP
MAX233ACWP	0°C to +70°C	20 Wide SO
MAX233AEPP	-40°C to +85°C	20 Plastic DIP
MAX233AEWP	-40°C to +85°C	20 Wide SO
MAX234CPE	0°C to +70°C	16 Plastic DIP
MAX234CWE	0°C to +70°C	16 Wide SO
MAX234C/D	0°C to +70°C	Dice*
MAX234EPE	-40°C to +85°C	16 Plastic DIP
MAX234EWE	-40°C to +85°C	16 Wide SO
MAX234EJE	-40°C to +85°C	16 CERDIP
MAX234MJE	-55°C to +125°C	16 CERDIP
MAX235CPG	0°C to +70°C	24 Wide Plastic DIP
MAX235EPG	-40°C to +85°C	24 Wide Plastic DIP
MAX235EDG	-40°C to +85°C	24 Ceramic SB
MAX235MDG	-55°C to +125°C	24 Ceramic SB
MAX236CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX236CWG	0°C to +70°C	24 Wide SO
MAX236C/D	0°C to +70°C	Dice*
MAX236ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX236EWG	-40°C to +85°C	24 Wide SO
MAX236ERG	-40°C to +85°C	24 Narrow CERDIP
MAX236MRG	-55°C to +125°C	24 Narrow CERDIP
MAX237CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX237CWG	0°C to +70°C	24 Wide SO
MAX237C/D	0°C to +70°C	Dice*
MAX237ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX237EWG	-40°C to +85°C	24 Wide SO
MAX237ERG	-40°C to +85°C	24 Narrow CERDIP
MAX237MRG	-55°C to +125°C	24 Narrow CERDIP
MAX238CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX238CWG	0°C to +70°C	24 Wide SO
MAX238C/D	0°C to +70°C	Dice*
MAX238ENG	-40°C to +85°C	24 Narrow Plastic DIP

** Contact factory for dice specifications.*

+5V-Powered, Multichannel RS-232 Drivers/Receivers

Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
MAX238EWG	-40°C to +85°C	24 Wide SO
MAX238ERG	-40°C to +85°C	24 Narrow CERDIP
MAX238MRG	-55°C to +125°C	24 Narrow CERDIP
MAX239CNG	0°C to +70°C	24 Narrow Plastic DIP
MAX239CWG	0°C to +70°C	24 Wide SO
MAX239C/D	0°C to +70°C	Dice*
MAX239ENG	-40°C to +85°C	24 Narrow Plastic DIP
MAX239EWG	-40°C to +85°C	24 Wide SO
MAX239ERG	-40°C to +85°C	24 Narrow CERDIP
MAX239MRG	-55°C to +125°C	24 Narrow CERDIP
MAX240CMH	0°C to +70°C	44 Plastic FP
MAX240C/D	0°C to +70°C	Dice*
MAX241CAI	0°C to +70°C	28 SSOP
MAX241CWI	0°C to +70°C	28 Wide SO
MAX241C/D	0°C to +70°C	Dice*
MAX241EAI	-40°C to +85°C	28 SSOP
MAX241EWI	-40°C to +85°C	28 Wide SO
MAX242CAP	0°C to +70°C	20 SSOP
MAX242CPN	0°C to +70°C	18 Plastic DIP
MAX242CWN	0°C to +70°C	18 Wide SO
MAX242C/D	0°C to +70°C	Dice*
MAX242EPN	-40°C to +85°C	18 Plastic DIP
MAX242EWN	-40°C to +85°C	18 Wide SO
MAX242EJN	-40°C to +85°C	18 CERDIP
MAX242MJN	-55°C to +125°C	18 CERDIP

MAX243CPE	0°C to +70°C	16 Plastic DIP
MAX243CSE	0°C to +70°C	16 Narrow SO
MAX243CWE	0°C to +70°C	16 Wide SO
MAX243C/D	0°C to +70°C	Dice*
MAX243EPE	-40°C to +85°C	16 Plastic DIP
MAX243ESE	-40°C to +85°C	16 Narrow SO
MAX243EWE	-40°C to +85°C	16 Wide SO
MAX243EJE	-40°C to +85°C	16 CERDIP
MAX243MJE	-55°C to +125°C	16 CERDIP
MAX244CQH	0°C to +70°C	44 PLCC
MAX244C/D	0°C to +70°C	Dice*
MAX244EQH	-40°C to +85°C	44 PLCC
MAX245CPL	0°C to +70°C	40 Plastic DIP
MAX245C/D	0°C to +70°C	Dice*
MAX245EPL	-40°C to +85°C	40 Plastic DIP
MAX246CPL	0°C to +70°C	40 Plastic DIP
MAX246C/D	0°C to +70°C	Dice*
MAX246EPL	-40°C to +85°C	40 Plastic DIP
MAX247CPL	0°C to +70°C	40 Plastic DIP
MAX247C/D	0°C to +70°C	Dice*
MAX247EPL	-40°C to +85°C	40 Plastic DIP
MAX248CQH	0°C to +70°C	44 PLCC
MAX248C/D	0°C to +70°C	Dice*
MAX248EQH	-40°C to +85°C	44 PLCC
MAX249CQH	0°C to +70°C	44 PLCC
MAX249EQH	-40°C to +85°C	44 PLCC

** Contact factory for dice specifications.*

MOTOROLA

SERIAL-IN PARALLEL-OUT SHIFT REGISTER

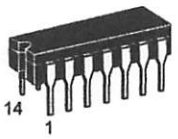
SN54/74LS164

SERIAL-IN PARALLEL-OUT SHIFT REGISTER

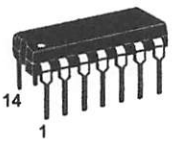
LOW POWER SCHOTTKY

The SN54/74LS164 is a high speed 8-Bit Serial-In Parallel-Out Shift Register. Serial data is entered through a 2-Input AND gate synchronous with the LOW to HIGH transition of the clock. The device features an asynchronous Master Reset which clears the register setting all outputs LOW independent of the clock. It utilizes the Schottky diode clamped process to achieve high speeds and is fully compatible with all Motorola TTL products.

- Typical Shift Frequency of 35 MHz
- Asynchronous Master Reset
- Gated Serial Data Input
- Fully Synchronous Data Transfers
- Input Clamp Diodes Limit High Speed Termination Effects
- ESD > 3500 Volts



J SUFFIX
CERAMIC
CASE 632-08



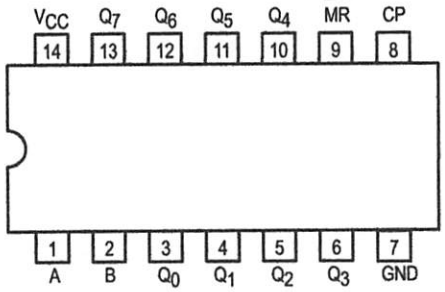
N SUFFIX
PLASTIC
CASE 646-06

D SUFFIX
SOIC
CASE 751A-02

ORDERING INFORMATION

SN54LSXXXJ	Ceramic
SN74LSXXXN	Plastic
SN74LSXXXD	SOIC

CONNECTION DIAGRAM DIP (TOP VIEW)



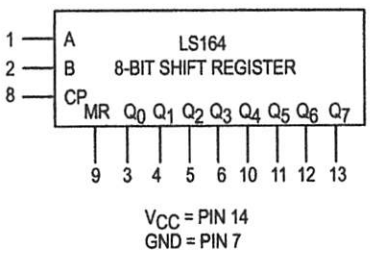
NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

PIN NAMES

		LOADING (Note a) HIGH	LOW
A, B	Data Inputs	0.5 U.L.	0.25 U.L.
CP	Clock (Active HIGH Going Edge) Input	0.5 U.L.	0.25 U.L.
MR	Master Reset (Active LOW) Input	0.5 U.L.	0.25 U.L.
Q_0–Q_7	Outputs (Note b)	10 U.L.	5 (2.5) U.L.

NOTES:
a) 1 TTL Unit Load (U.L.) = 40 µA HIGH/1.6 mA LOW.
b) The Output LOW drive factor is 2.5 U.L. for Military (54) and 5 U.L. for Commercial (74) Temperature Ranges.

LOGIC SYMBOL



V_{CC} = PIN 14
GND = PIN 7

MOTOROLA

SERIAL-IN PARALLEL-OUT SHIFT REGISTER

SN54/74LS164

SERIAL-IN PARALLEL-OUT
SHIFT REGISTER

LOW POWER SCHOTTKY

J SUFFIX
CERAMIC
CASE 632-07

N SUFFIX
PLASTIC
CASE 646-06

D SUFFIX
SOIC
CASE 751A-02

ORDERING INFORMATION

SN54LSXXXJ Ceramic
SN74LSXXXN Plastic
SN74LSXXXD SOIC

LOGIC SYMBOL

VCC = PIN 14
GND = PIN 7

- Typical Shift Frequency of 36 MHz
- Asynchronous Master Reset
- Gated Serial Data Input
- Fully Synchronous Data Transfers
- Input Clamp Diodes Limit High Speed Termination Effects
- ESD > 3500 Volts

CONNECTION DIAGRAM DIP (TOP VIEW)

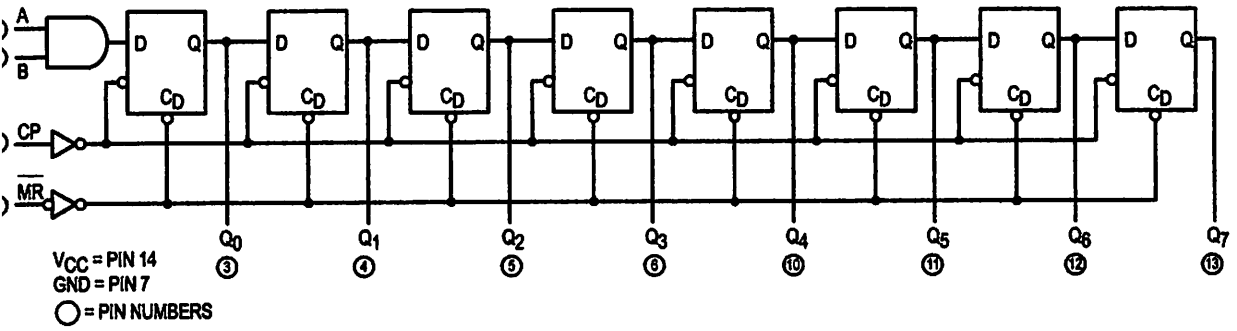
NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

PIN NAMES

		LOADING (Note a)	
		HIGH	LOW
A, B	Data Inputs	0.5 U.L.	0.25 U.L.
CP	Clock (Active HIGH Going Edge) Input	0.5 U.L.	0.25 U.L.
MR	Master Reset (Active LOW) Input	0.5 U.L.	0.25 U.L.
Q_0–Q_7	Outputs (Note b)	10 U.L.	5 (2.5) U.L.

NOTES:
a) 1 TTL Unit Load (U.L.) = 40 µA HIGH/1.6 mA LOW.
b) The Output LOW drive factor is 2.5 U.L. for Military (54) and 5 U.L. for Commercial (74) Temperature Ranges.

LOGIC DIAGRAM



UNCTIONAL DESCRIPTION

The LS164 is an edge-triggered 8-bit shift register with seri- data entry and an output from each of the eight stages. Data entered serially through one of two inputs (A or B); either of ese inputs can be used as an active HIGH Enable for data ntry through the other input. An unused input must be tied IGH, or both inputs connected together.

Each LOW-to-HIGH transition on the Clock (CP) input shifts data one place to the right and enters into Q_0 the logical AND of the two data inputs (A•B) that existed before the rising clock edge. A LOW level on the Master Reset (MR) input overrides all other inputs and clears the register asynchronously, forcing all Q outputs LOW.

MODE SELECT — TRUTH TABLE

OPERATING MODE	INPUTS			OUTPUTS	
	MR	A	B	Q_0	Q_1–Q_7
Reset (Clear)	L	X	X	L	L – L
Shift	H	l	l	L	q_0 – q_6
	H	l	h	L	q_0 – q_6
	H	h	l	L	q_0 – q_6
	H	h	h	H	q_0 – q_6

L (l) = LOW Voltage Levels
H (h) = HIGH Voltage Levels
X = Don't Care
q_n = Lower case letters indicate the state of the referenced input or output one set-up time prior to the LOW to HIGH clock transition.

UARANTEED OPERATING RANGES

Symbol	Parameter		Min	Typ	Max	Unit
V_{CC}	Supply Voltage	54 74	4.5 4.75	5.0 5.0	5.5 5.25	V
T_A	Operating Ambient Temperature Range	54 74	−55 0	25 25	125 70	°C
I_{OH}	Output Current — High	54, 74			−0.4	mA
I_{OL}	Output Current — Low	54 74			4.0 8.0	mA

DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

Symbol	Parameter		Limits Min	Typ	Max	Unit	Test Conditions	
V_{IH}	Input HIGH Voltage		2.0			V	Guaranteed Input HIGH Voltage for All Inputs	
V_{IL}	Input LOW Voltage	54			0.7	V	Guaranteed Input LOW Voltage for All Inputs	
		74			0.8			
V_{IK}	Input Clamp Diode Voltage			−0.65	−1.5	V	V_{CC} = MIN, I_{IN} = −18 mA	
V_{OH}	Output HIGH Voltage	54	2.5	3.5		V	V_{CC} = MIN, I_{OH} = MAX, V_{IN} = V_{IH} or V_{IL} per Truth Table	
		74	2.7	3.5				
V_{OL}	Output LOW Voltage	54, 74		0.25	0.4	V	I_{OL} = 4.0 mA	V_{CC} = V_{CC} MIN, V_{IN} = V_{IH} or V_{IL} per Truth Table
		74		0.35	0.5	V	I_{OL} = 8.0 mA	
I_{IH}	Input HIGH Current				20	µA	V_{CC} = MAX, V_{IN} = 2.7 V	
					0.1	mA	V_{CC} = MAX, V_{IN} = 7.0 V	
I_{IL}	Input LOW Current				−0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V	
I_{OS}	Short Circuit Current (Note 1)		−20		−100	mA	V_{CC} = MAX	
I_{CC}	Power Supply Current				27	mA	V_{CC} = MAX	

Note 1: Not more than one output should be shorted at a time, nor for more than 1 second.

AC CHARACTERISTICS (T_A = 25°C)

Symbol	Parameter	Limits Min	Typ	Max	Unit	Test Conditions
f_{MAX}	Maximum Clock Frequency	25	36		MHz	V_{CC} = 5.0 V C_L = 15 pF
t_{PHL}	Propagation Delay MR to Output Q		24	36	ns	
t_{PLH} t_{PHL}	Propagation Delay Clock to Output Q		17 21	27 32	ns	

AC SETUP REQUIREMENTS (T_A = 25°C)

Symbol	Parameter	Limits Min	Typ	Max	Unit	Test Conditions
t_W	CP, MR Pulse Width	20			ns	V_{CC} = 5.0 V
t_s	Data Setup Time	15			ns	
t_h	Data Hold Time	5.0			ns	
t_{rec}	MR to Clock Recovery Time	20			ns	

AC WAVEFORMS

*The shaded areas indicate when the input is permitted to change for predictable output performance.

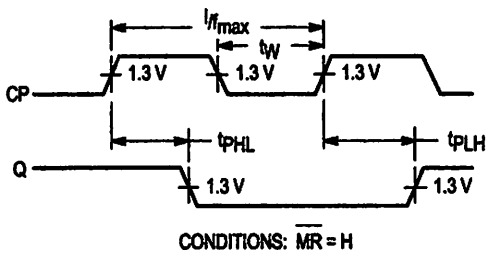


Figure 1. Clock to Output Delays and Clock Pulse Width

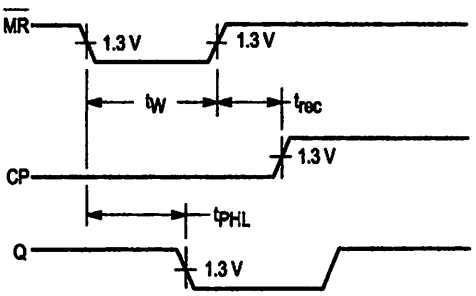


Figure 2. Master Reset Pulse Width, Master Reset to Output Delay and Master Reset to Clock Recovery Time

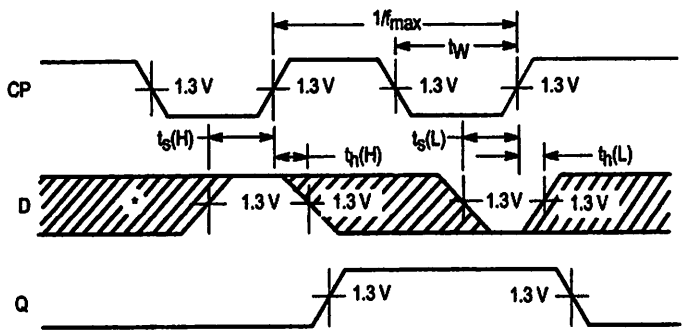


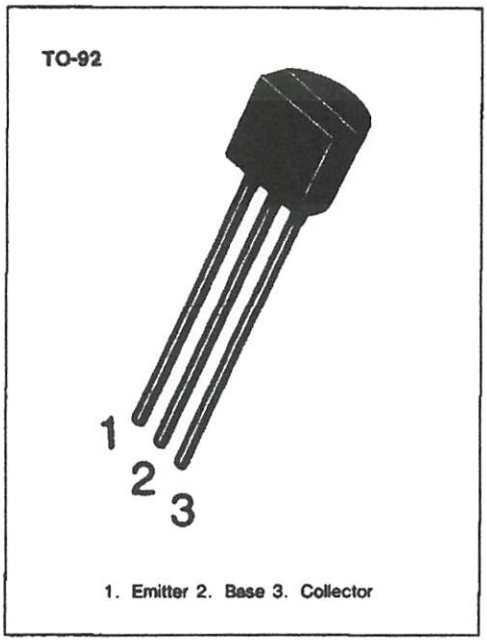
Figure 3. Data Setup and Hold Times

Transistors
2SC9014

USHA (INDIA) LTD

RE-AMPLIFIER, LOW LEVEL & LOW NOISE

- High total power dissipation. **(PT=450mW)**
- High h_{FE} and good linearity
- Complementary to SS9015



1. Emitter 2. Base 3. Collector

ABSOLUTE MAXIMUM RATINGS (T_a=25°C)

Characteristic	Symbol	Rating	Unit
Collector-Base Voltage	V_{CBO}	50	V
Collector-Emitter Voltage	V_{CEO}	45	V
Emitter-Base Voltage	V_{EBO}	5	V
Collector Current	I_C	100	mA
Collector Dissipation	P_C	450	mW
Junction Temperature	Tj	150	°C
Storage Temperature	Tstg	−55~150	°C

ELECTRICAL CHARACTERISTICS (T_a=25°C)

Characteristic	Symbol	Test Conditions	Min	Typ	Max	Unit
Collector-Base Breakdown Voltage	BV_{CBO}	I_C=100μA, I_E=0	50			V
Collector-Emitter Breakdown Voltage	BV_{CEO}	I_C=1mA, I_B=0	45			V
Emitter-Base Breakdown Voltage	BV_{EBO}	I_E=100μA, I_C=0	5			V
Collector Cutoff Current	I_{CBO}	V_{CB}=50V, I_E=0			50	nA
Emitter Cutoff Current	I_{EBO}	V_{EB}=5V, I_C=0			50	nA
DC Current Gain	h_{FE}	V_{CE}=5V, I_C=1mA	60	280	1000	
Collector-Base Saturation Voltage	V_{CE}(sat)	I_C=100mA, I_B=5mA		0.14	0.3	V
Base-Emitter Saturation Voltage	V_{BE}(sat)	I_C=100mA, I_B=5mA		0.84	1.0	V
Base-Emitter On Voltage	V_{BE}(on)	V_{CE}=5V, I_C=2mA	0.58	0.63	0.7	V
Output Capacitance	Cob	V_{CB}=10V, I_E=0 f=1MHz		2.2	3.5	pF
Current Gain-Bandwidth Product	f_T	V_{CE}=5V, I_C=10mA	150	270		MHz
Noise Figure	NF	V_{CE}=5V, I_C=0.2mA f=1KHz, Rs=2KΩ		0.9	10	dB

h_{FE} CLASSIFICATION

Classification	A	B	C	D
h_{FE}	60-150	100-300	200-600	400-1000

ISO 14001 CERTIFIED
AN ISO 9002 COMPANY

Transistors

2SC2874

USHA
(INDIA) LTD

RF-AMPLIFIER, LOW LEVEL & LOW NOISE

High fo, low noise dissipation. (PT=400mW)
High hFE and good linearity
Complementary to 2SA1162

ABSOLUTE MAXIMUM RATINGS (T_a=25°C)

Characteristic	Symbol	Rating	Unit
Collector-Base Voltage	V_{CBO}	60	V
Collector-Emitter Voltage	V_{CEO}	50	V
Emitter-Base Voltage	V_{EBO}	5	V
Collector Current	I_C	100	mA
Collector Dissipation	P_C	400	mW
Junction Temperature	T_j	150	°C
Storage Temperature	T_{stg}	−55~+150	°C

ELECTRICAL CHARACTERISTICS (T_a=25°C)

Characteristic	Symbol	Test Conditions	Min	Typ	Max	Unit
Collector-Base Breakdown Voltage	BV_{CBO}	I_C=100μA, I_E=0	60			V
Collector-Emitter Breakdown Voltage	BV_{CEO}	I_C=1mA, I_B=0	50			V
Emitter-Base Breakdown Voltage	BV_{EBO}	I_E=100μA, I_C=0	5			V
Collector Cutoff Current	I_{CBO}	V_{CB}=50V, I_E=0			50	nA
Emitter Cutoff Current	I_{EBO}	V_{EB}=5V, I_C=0			50	nA
DC Current Gain	h_{FE}	V_{CE}=6V, I_C=1mA	60	280	1000	
Collector-Base Saturation Voltage	$V_{CE(sat)}$	I_C=100mA, I_B=5mA		0.14	0.3	V
Base-Emitter Saturation Voltage	$V_{BE(sat)}$	I_C=100mA, I_B=5mA		0.84	1.0	V
Base-Emitter On Voltage	$V_{BE(on)}$	V_{CE}=6V, I_C=1mA	0.58	0.60	0.7	V
Output Capacitance	C_{ob}	V_{CB}=10V, I_E=0, f=1MHz		2.0	3.5	pF
Current Gain-Bandwidth Product	f_T	V_{CE}=6V, I_C=10mA	140	270		MHz
Noise Figure	NF	V_{CE}=6V, I_C=0.1mA, f=1kHz, R_G=2kΩ		0.9	10	dB

hFE CLASSIFICATION

Classification	A	B	C	D
h_{FE}	60~150	120~240	200~400	350~700

LISTING PROGRAM MIKROKONTROLLER

```
;    X1                  EQU   P2.0;
;    X2                  EQU   P2.1
;    X3                  EQU   P2.2
;    X4                  EQU   P2.3

;    Y1                  EQU   P2.7
;    Y2                  EQU   P2.6
;    Y3                  EQU   P2.5


;SCAN_KEYPAD:
;PORT LCD
        ORG   00H
RS                  EQU   P1.0
E                   EQU   P1.1
CON_LCD             EQU   P1.2
DATA_LCD            EQU   P1.3

;---------------------------------------
; INISIALISASI CONTROL LCD
DISPCLEAR           EQU   01H
FUNCSET             EQU   38H
ENTRMOD             EQU   06H
DISPON              EQU   0CH
DATA_SENSOR         EQU   30H
BUFFER              EQU   20H
BUFFER1             EQU   22H
AWAL                EQU   2FH
BUS1                EQU   P1.7
BUS2                EQU   P1.6
BUS3                EQU   P1.5
TANDA               EQU   21H
TANDA0              EQU   TANDA.0


        ORG 00H
        LJMP MULAI
        ORG 23H
        LJMP SERIAL

SERIAL:

; MENERIMA DATA DARI KOMPUTER
        PUSH   ACC
        JNB    RI,$
        MOV    A,SBUF
        CLR    RI
```

LISTING PROGRAM MIKROKONTROLLER

```
;   X1          EQU   P2.0
;   X2          EQU   P2.1
;   X3          EQU   P2.2
;   X4          EQU   P2.3

;   Y1          EQU   P2.7
;   Y2          EQU   P2.6
;   Y3          EQU   P2.5

;SCAN_KEYPAD:
;PORT LCD
            ORG   00H
RS          EQU   P1.0
E           EQU   P1.1
CON_LCD     EQU   P1.2
DATA_LCD    EQU   P1.3

;-------------------------------------------------
; INISIALISASI CONTROL LCD
DISPCLEAR   EQU   01H
FUNCSET     EQU   38H
ENTRMOD     EQU   06H
DISPON      EQU   0CH
DATA_SENSOR EQU   30H
BUFFER      EQU   20H
BUFFER1     EQU   22H
AWAL        EQU   2FH
BUS1        EQU   P1.7
BUS2        EQU   P1.6
BUS3        EQU   P1.5
TANDA       EQU   21H
TANDA0      EQU   TANDA.0


            ORG 00H
            LJMP MULAI
            ORG 23H
            LJMP SERIAL

SERIAL:

; MENERIMA DATA DARI KOMPUTER
            PUSH  ACC
            JNB   RI,S
            MOV   A,SBUF
            CLR   RI
```

```
TELEK1: CJNE A,#'A',TELEK2 ;===TAMPILKAN ACCOUNT
        LCALL KIRIMKOM
        LJMP AKHIR
TELEK2: CJNE A,#'B',TELEK3
        JB BUS1,TELEK21
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK21:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK3:
        CJNE A,#'C',TELEK4
        JB BUS2,TELEK31
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK31:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR

        JMP AKHIR
TELEK4:
        CJNE A,#'D',AKHIR
        JB BUS3,TELEK41
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK41:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR

AKHIR:
      POP ACC
      RETI

MULAI:
        LCALL INITLCD

        SETB  EA
        MOV   TMOD,#20H
        MOV   TH1,#0FDH     ; #0F3 2404 BPS  ; #0E6H UNTUK 1200 BPS
        SETB  TR1
        MOV   SCON,#50H
        SETB  ES

        SETB TANDA0
        LCALL CLRMEM
        MOV AWAL,#30H
```

```
TELEK1: CINE A,#'A'-TELEK2 ;===TAMPILKAN ACC'OUNT
        LCALL KIRIMKOM
        LJMP AKHIR
TELEK2: CINE A,#'B'-TELEK3
        JB BUS1,TELEK21
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK21:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK3:
        CINE A,#'C'-TELEK4
        JB BUS2,TELEK31
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK31:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR

        JMP AKHIR
TELEK4:
        CINE A,#'D'-AKHIR
        JB BUS3,TELEK41
        MOV A,#30H
        LCALL KIRIMSER
        LJMP AKHIR
TELEK41:
        MOV A,#31H
        LCALL KIRIMSER
        LJMP AKHIR

AKHIR:
        POP ACC
        RETI

MULAI:
        LCALL INIT_CD

        SETB  EA
        MOV   TMOD,#20H
        MOV   TH1,#0FDH     ; #0FD 2404 BPS ; #0E8H UNTUK 1200 BPS
        SETB  TR1
        MOV   SCON,#50H
        SETB  ES

        SETB TANDA0
        LCALL CLRMEM
        MOV AWAL,#30H
```

```
        MOV R5,#1
        MOV DPTR,#ANTRI1
        LCALL CETAK1
        MOV R5,#1
        MOV DPTR,#ANTRI2
        LCALL CETAK2
        LCALL DLY1000MS
        LCALL DLY1000MS
        LCALL DLY1000MS

PUTER:

        MOV   R5,#1
        MOV   DPTR,#AWAL1
        LCALL  CETAK1
        MOV   R5,#1
        MOV   DPTR,#AWAL2
        LCALL  CETAK2
        LCALL KEY
        MOV BUFFER,A
        LCALL BUSEK
        MOV R5,#1
        MOV DPTR,#KODE2
        LCALL CETAK2
        MOV R5,#1
        MOV DPTR,#KODE
        LCALL CETAK1
        MOV R4,#1
        MOV R1,#30H
        MOV A,BUFFER
        MOV @R1,A
        LCALL DATAOUT
PAD:
        LCALL KEY
        CJNE A,#23H,TERUS0
        LJMP TERUS1
TERUS0:
        CJNE A,#2AH,TERUS
        LCALL CLRMEM
        JMP PUTER
TERUS:
        INC  R1
        INC R4
        MOV @R1,A
        LCALL DATAOUT
        LJMP PAD
TERUS1:
        CLR TANDA0
        MOV AWAL,#31H
        LJMP PUTER
```

```
;===SUBPROGRAM

KURANG:
        MOV R5,#1
        MOV DPTR,#KODE2
        LCALL CETAK2
        MOV R5,#1
        MOV DPTR,#KODE
        LCALL CETAK1
        MOV R1,#30H
        MOV BUFFER1,R4
KURANG1:
        MOV A,@R1
        LCALL DATAOUT
        INC R1
        DJNZ BUFFER1,KURANG1
        RET

KIRIMKOM:
        MOV R0,#2FH
        MOV R2,#11
KIRIMKOM1:
            MOV A,@R0
            LCALL KIRIMSER
            INC R0
            DJNZ R2,KIRIMKOM1
            JB TANDA0,KIRIMKOM2
            LCALL CLRMEM
            MOV AWAL,#30H
            SETB TANDA0
KIRIMKOM2:
            RET

KIRIMSER:
            CLR   ES
            MOV    SBUF,A       ; KIRIM LEWAT SERIAL COMM.
            JNB   TI,$
            CLR   TI
            SETB   ES
            RET

CLRMEM:  MOV R0,#30H
            MOV R2,#10


CLRMEM1:
            MOV @R0,#20H
            INC R0
            DJNZ R2,CLRMEM1
            RET
```

```
BUSEK:  MOV R5,#1
        MOV DPTR,#HAPUS
        LCALL CETAK1
        MOV R5,#1
        MOV DPTR,#HAPUS
        LCALL CETAK2
        RET
KEY:
        MOV P2,#0FEH
        JB P2.4,KEY1
        MOV A,#31H
        JNB P2.4,$
        LJMP KEYAKHIR
KEY1:
        JB P2.6,KEY2
        MOV A,#32H
        JNB P2.6,$
        LJMP KEYAKHIR
KEY2:
        JB P2.5,KEY3
        MOV A,#33H
        JNB P2.5,$
        LJMP KEYAKHIR
KEY3:
        MOV P2,#11111101B
        JB P2.4,KEY4
        MOV A,#34H
        JNB P2.4,$
        LJMP KEYAKHIR
KEY4:
        JB P2.6,KEY5
        MOV A,#35H
        JNB P2.6,$
        LJMP KEYAKHIR
KEY5:
        JB P2.5,KEY6
        MOV A,#36H
        JNB P2.5,$
        LJMP KEYAKHIR
KEY6:
        MOV P2,#11111011B
        JB P2.4,KEY7
        MOV A,#37H
        JNB P2.4,$
        LJMP KEYAKHIR
KEY7:
        JB P2.6,KEY8
        MOV A,#38H
        JNB P2.6,$
        LJMP KEYAKHIR
```

```
BUSTEK:  MOV R5,#1
         MOV DPTR,#HAPPY
         LCALL CETAK1
         MOV R5,#1
         MOV DPTR,#HAPPY
         LCALL CETAK2
         RET
KEY:
         MOV P2,#0FFH
         JB P2.4,KEY1
         MOV A,#31H
         JNB P2.4,$
         LJMP KEYAKHIR
KEY1:
         JB P2.6,KEY2
         MOV A,#32H
         JNB P2.6,$
         LJMP KEYAKHIR
KEY2:
         JB P2.5,KEY3
         MOV A,#33H
         JNB P2.5,$
         LJMP KEYAKHIR
KEY3:
         MOV P2,#11110111B
         JB P2.4,KEY4
         MOV A,#34H
         JNB P2.4,$
         LJMP KEYAKHIR
KEY4:
         JB P2.6,KEY5
         MOV A,#35H
         JNB P2.6,$
         LJMP KEYAKHIR
KEY5:
         JB P2.5,KEY6
         MOV A,#36H
         JNB P2.5,$
         LJMP KEYAKHIR
KEY6:
         MOV P2,#11110111B
         JB P2.4,KEY7
         MOV A,#37H
         JNB P2.4,$
         LJMP KEYAKHIR
KEY7:
         JB P2.6,KEY8
         MOV A,#38H
         JNB P2.6,$
         LJMP KEYAKHIR
```

```
KEY8:
        JB P2.5,KEY9
        MOV A,#39H
        JNB P2.5,$
        LJMP KEYAKHIR
KEY9:
        JB P2.5,KEY10
        MOV A,#23H
        JNB P2.5,$
        LJMP KEYAKHIR

KEY10:
        JB P2.6,KEY11
        MOV A,#30H
        JNB P2.6,$
        LJMP KEYAKHIR
KEY11:
        JB P2.7,KEY12
        MOV A,#2AH
        JNB P2.7,$
        LJMP KEYAKHIR

KEY12:
        LJMP KEY
KEYAKHIR:

        RET

;INIT LCD

DELAY
        MOV   R6,#00H
DLYLCDLP
        MOV   R7,#00H
        DJNZ  R7,$
        DJNZ  R6,DLYLCDLP
        RET
BARIS2
        MOV A,R5
        ADD A,#0C0H ;11000000B
        LJMP POSISISUB

BARIS1
        MOV A,R5
        ADD A,#80H; 10000000B

POSISISUB
        DEC  A
        LCALL CONTROLOUT
        RET

CETAK2
```

```
        LCALL BARIS2
        LJMP ANSA

CETAK1
        LCALL BARIS1

ANSA
        LJMP OUTSTRING
LOOP
        LCALL DATAOUT
        INC DPTR

OUTSTRING
        CLR A
        MOVC A,@A+DPTR
        CJNE A,#'$',LOOP

        RET


CONTROLOUT
        PUSH DPH
        PUSH DPL

        CLR RS
        LJMP OUT

DATAOUT
        PUSH DPH
        PUSH DPL

        SETB RS

OUT
        SETB E
        LCALL GESER
        MOV R6,#250
        DJNZ R6,$
        POP DPL
        POP DPH
        CLR E
        RET

GESER
        CLR CON_LCD
        MOV   B,#8
A1
        RRC A
        MOV DATA_LCD,C
        NOP
        NOP
        NOP
```

```
            NOP
            SETB CON_LCD
            NOP
            NOP
            NOP
            NOP
            CLR CON_LCD
            DJNZ B,A1
            RET
DELAY2
            MOV    R6,#0FFH
DLYLCD1
            MOV    R7,#0FFH
            DJNZ   R7,$
            DJNZ   R6,DLYLCD1
            RET



DELAYPLUS    MOV    R5,#10
DELAYPLUS2   CALL   DELAY2
            DJNZ   R5,DELAYPLUS2
            RET

INITLCD
            MOV  A,#DISPCLEAR
            LCALL CONTROLOUT
            LCALL DELAY

            MOV A,#FUNCSET
            LCALL CONTROLOUT

            MOV A,#DISPON
            LCALL CONTROLOUT

            MOV A,#ENTRMOD
            LCALL CONTROLOUT

             RET
;=============================================
;RUTIN DELAY
;=============================================
DLY1000MS:
            MOV    R2,#10
ULDEL:
            LCALL  DLY100MS
            DJNZ   R2,ULDEL
            RET
DLY100MS:
            MOV R6,#200
            LJMP   WAKTU
```

```
DLY10MS:
        MOV  R6,#20
        LJMP    WAKTU

; TIMER 0,5 MS
; R6 = PERKALIAN
WAKTU:
        MOV    R7,#247
        DJNZ   R7,$
        DJNZ   R6,WAKTU
        RET

HAPUS
        DB '                    $'
AWAL1
        DB ' MASUKKAN KODE $'
AWAL2:
        DB '       BUS          $'
KODE:
        DB 'KODE=$'
KODE2:
        DB '*>HAPUS #>KIRIM  $'
ANTRI1:
        DB ' ANTRIAN BUS   $'
ANTRI2:
        DB ' ABDUL KADIR   $'
        END
```

LISTING PROGRAM DELPHI

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls,
Forms,
  DialogsCust, Menus, StdCtrls, Grids, DBGrids, ExtCtrls, ComCtrls,
Buttons,
  DB, DBTables, CPort, CPortCtl, StrUtils, DateUtils, MMSystem;

type
  TForm1 = class(TForm)
    MainMenu1: TMainMenu;
    EntryData1: TMenuItem;
    Konfigurasi1: TMenuItem;
    Bantuan1: TMenuItem;
    RekamKata1: TMenuItem;
    PeriodeAntrian1: TMenuItem;
    FormatPengumuman1: TMenuItem;
    KomunikasiSerial1: TMenuItem;
    PanduanOperasi1: TMenuItem;
    N1: TMenuItem;
    Penyusun1: TMenuItem;
    Kedatangan1: TMenuItem;
    Persiapan1: TMenuItem;
    Keberangkatan1: TMenuItem;
    DataBis1: TMenuItem;
    utup1: TMenuItem;
    DBGrid1: TDBGrid;
    Label1: TLabel;
    Label2: TLabel;
    DBGrid2: TDBGrid;
    Label3: TLabel;
    DBGrid3: TDBGrid;
    Bevel1: TBevel;
    Label4: TLabel;
    Jam: TLabel;
    TimerJam: TTimer;
    btLihat_data: TSpeedButton;
    Table1: TTable;
    Table2: TTable;
    Table3: TTable;
    DataSource1: TDataSource;
    DataSource2: TDataSource;
    DataSource3: TDataSource;
    Table4: TTable;
    DataSource4: TDataSource;
    ComComboBox1: TComComboBox;
    StatusBar1: TStatusBar;
    ComPort1: TComPort;
    JalurBus1: TMenuItem;
    Table5: TTable;
    Laporan1: TMenuItem;
```

```
    RekapPemberangkatan1: TMenuItem;
    Table6: TTable;
    BitBtn1: TBitBtn;
    Panel1: TPanel;
    Label5: TLabel;
    Label6: TLabel;
    Label7: TLabel;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Image1: TImage;
    procedure utup1Click(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure TimerJamTimer(Sender: TObject);
    procedure DataBis1Click(Sender: TObject);
    procedure Kedatangan1Click(Sender: TObject);
    procedure Persiapan1Click(Sender: TObject);
    procedure Keberangkatan1Click(Sender: TObject);
    procedure btLihat_dataClick(Sender: TObject);
    procedure PeriodeAntrian1Click(Sender: TObject);
    procedure KomunikasiSerial1Click(Sender: TObject);
    procedure ComPort1AfterClose(Sender: TObject);
    procedure ComPort1AfterOpen(Sender: TObject);
    procedure JalurBus1Click(Sender: TObject);
    procedure RekapPemberangkatan1Click(Sender: TObject);
    procedure BitBtn1Click(Sender: TObject);
  private
    { Private declarations }
    procedure Pengumuman(var stbus,stjurusan,stjalur,stnopol: string;
const stmode : string);
    procedure Masukkan_antrian(var stkode:string);
    function Cari_Jadwal(var stkode : string):boolean;
    procedure Pengumuman_persiapan(var jalur :byte);
    procedure Pengumuman_berangkat(var jalur:byte);
    procedure Periksa_jalur(var jalur : byte);
    procedure Laporan_berangkat(var stkode, stbus, stnopol: string);
    procedure PlayWav(var stfile : string);
  public
    { Public declarations }
  end;

var
  Form1: TForm1;
  fRun : Boolean;
  hitError : integer;

implementation

uses Unit2, Unit4;

{$R *.dfm}

const
  Title         = 'Antrian Bus';
  AppBGColor    = 11876877;
  szJam         = 8;
  szBusID       = 10;
```

```
  szJalur       = 2;
  szNama        = 20;
  szNopol       = 10;
  szPemilik     = 20;
  szAlamat      = 30;
  szJurusan     = 20;
  Karjlr1       = 'B';
  AdaBis        = '1';
  tmbersiap     = 10;              // waktu persiapan

var
  jamnya : TDateTime;
  recnum : integer;
procedure TForm1.utup1Click(Sender: TObject);
begin
  Comport1.Close;
  Application.Terminate;
end;
procedure TForm1.FormCreate(Sender: TObject);
var
  lebar : integer;
  pathnya,DBnya : string;
begin
  fRun := False;
  Application.Title := Title;
  Form1.Caption := '';
  Label4.Caption := title;
  lebar := Form1.Width;
  Bevel1.Width := lebar - 32;
  Bevel1.Left := 15;
  pathnya := GetCurrentDir;
  DBnya := Pathnya + '\TBDatang.db';
  if not FileExists(DBnya) then
  begin
    with Table1 do                          //    bikin_TBDatang;
    begin
      DatabaseName := Pathnya;
      TableType := ttPAradox;
      TableName := 'TBDatang';
      With FieldDefs do
      begin
        Clear;
        with AddFieldDef do begin
          Name := 'Jam';
          DataType := ftString;
          Required := True;
          Size   := szJam;
        end;
        with AddFieldDef do begin
          Name := 'BusID';
          DataType := ftString;
          Size := szBusID;
        end;
      end;
      with IndexDefs do begin
        Clear;
      end;
```

```
    CreateTable;
  end;
end;
DBnya := Pathnya + '\TBPersiapan.db';
if not FileExists(DBnya) then
begin
  with Table2 do                          //    bikin_TBPersiapan;
  begin
    DatabaseName := Pathnya;
    TableType := ttPAradox;
    TableName := 'TBPersiapan';
    With FieldDefs do
    begin
      Clear;
      with AddFieldDef do begin
        Name := 'Jam';
        DataType := ftDateTime;
        Required := True;
      end;
      with AddFieldDef do begin
        Name := 'BusID';
        DataType := ftString;
        Size := szBusID;
      end;
      with AddFieldDef do begin
        Name := 'Jalur';
        DataType := ftString;
        Size := szJalur;
      end;
      with AddFieldDef do begin
        Name := 'Mode';
        DataType := ftString;
        Size := 2;
      end;
    end;
    with IndexDefs do begin
      Clear;
    end;
    CreateTable;
  end;
end;
DBnya := Pathnya + '\TBJadwal.db';
if not FileExists(DBnya) then
begin
  with Table3 do                          //    bikin_TBJadwal;
  begin
    DatabaseName := Pathnya;
    TableType := ttPAradox;
    TableName := 'TBJadwal';
    With FieldDefs do
    begin
      Clear;
      with AddFieldDef do begin
        Name := 'Jam';
        DataType := ftString;
        Required := True;
        Size   := szJam;
```

```
      end;
      with AddFieldDef do begin
        Name := 'BusID';
        DataType := ftString;
        Size := szBusID;
      end;
      with AddFieldDef do begin
        Name := 'Nama';
        DataType := ftString;
        Size := szNama;
      end;
      with AddFieldDef do begin
        Name := 'Jalur';
        DataType := ftString;
        Size := szjalur;
      end;
      with AddFieldDef do begin
        Name := 'Jurusan';
        DataType := ftString;
        Size := szJurusan;
      end;
    end;
    CreateTable;
  end;
end;
DBnya := Pathnya + '\TBBus.db';
if not FileExists(DBnya) then
begin
  with Table4 do                          //    bikin_DBBus;
  begin
    DatabaseName := Pathnya;
    TableType := ttPAradox;
    TableName := 'TBBus';
    With FieldDefs do
    begin
      Clear;
      with AddFieldDef do begin
        Name := 'BusID';
        DataType := ftString;
        Required := True;
        Size   := szBusID;
      end;
      with AddFieldDef do begin
        Name := 'Nama';
        DataType := ftString;
        Size := szNama;
      end;
      with AddFieldDef do begin
        Name := 'NoPol';
        DataType := ftString;
        Size := szNopol;
      end;
      with AddFieldDef do begin
        Name := 'Sopir';
        DataType := ftString;
        Size := szPemilik;
      end;
```

```
      with AddFieldDef do begin
        Name := 'Alamat';
        DataType := ftString;
        Size := szAlamat;
      end;
      with AddFieldDef do begin
        Name := 'Jurusan';
        DataType := ftString;
        Size := szJurusan;
      end;
    end;
    with IndexDefs do begin
      Clear;
      with AddIndexDef do begin
        Name := 'Kode';
        Fields := 'BusID';
        Options := [ixPrimary];
      end;
    end;
    CreateTable;
  end;
end;
DBnya := Pathnya + '\TBJalur.db';
if not FileExists(DBnya) then
begin
  with Table5 do                          //    bikin_TJalur;
  begin
    DatabaseName := Pathnya;
    TableType := ttPAradox;
    TableName := 'TBJalur';
    With FieldDefs do
    begin
      Clear;
      with AddFieldDef do begin
        Name := 'Jalur';
        DataType := ftString;
        Size := szJalur;
      end;
      with AddFieldDef do begin
        Name := 'Jurusan';
        DataType := ftString;
        Required := True;
        Size   := szJurusan;
      end;
    end;
    CreateTable;
  end;
end;
DBnya := Pathnya + '\TBRekap.db';
if not FileExists(DBnya) then
begin
  with Table6 do                          //    bikin_TBRekap;
  begin
    DatabaseName := Pathnya;
    TableType := ttPAradox;
    TableName := 'TBRekap';
    With FieldDefs do
```

```
      begin
        Clear;
        with AddFieldDef do begin
          Name := 'Tgl_Jam';
          DataType := ftString;
          Size := szJam+szJam+4;
        end;
        with AddFieldDef do begin
          Name := 'Nopol';
          DataType := ftString;
          Size := szNopol;
        end;
        with AddFieldDef do begin
          Name := 'PO';
          DataType := ftString;
          Required := True;
          Size   := szNama;
        end;
      end;
      CreateTable;
    end;
  end;
  Table1.TableName := 'TBDatang';
  Table1.Open;
  Table2.TableName := 'TBPersiapan';
  Table2.Open;
  Table3.TableName := 'TBJadwal';
  Table3.Open;
  Table4.TableName := 'TBBus';
  Table4.Open;
  Table5.TableName := 'TBJalur';
  Table5.Open;
  Table6.TableName := 'TBRekap';
  Table6.Open;
  StatusBar1.Panels.Items[0].Text := Comport1.Port;
  StatusBar1.Panels.Items[1].Text := 'Stop';
  Comport1.Open;
end;
procedure TForm1.TimerJamTimer(Sender: TObject);
var
  dtout : char;
  hit, hot : byte;
  stkode,stjlr1,stjlr2,stjlr3 : string;
  hasil : boolean;
begin
  Jam.Caption := DateTimetoStr(Now);
  if fRun then
  begin
    hasil := true;                          // Query Kode
    dtout := 'A';                           // Kirim 'A' <- jawaban 11 byte
; byte 1 : 0=gak ada; 1=baca kodenya
    Comport1.WriteStr(dtout);
    stkode :='';
    Comport1.ReadStr(stkode,11);
    if length(stkode) < 11 then
      hasil := false;
    if (hasil) and (stkode[1] = '1') then
```

```
    begin
      if Cari_jadwal(stkode) then
        Masukkan_antrian(stkode);
      Statusbar1.Panels.Items[2].Text := '  >> ';
    end else if not hasil then
    begin
      Statusbar1.Panels.Items[2].Text := '  >> Gak ada jawaban !
Mungkin kabel putus atau coba port yang lain.';
      inc(hiterror);
    end;
    if (hiterror = 5) and (fRun = True) then
    begin
      Bitbtn1.Click;
    end;
    hot := 1;
    for hit := 1 to 3 do                    // mode antrian:
    begin                                   // 0 : antri
      Pengumuman_berangkat(hot);            // 1 : pengumuman persiapan
      Pengumuman_persiapan(hot);            // 2 : persiapan
      inc(hot);                             // 3 : pengumuman berangkat ->
hapus dr antrian -> catat di rekap
    end;
    hot := 1;
    for hit := 1 to 3 do
    begin
      Periksa_jalur(hot);                   // Query jalur
      inc(hot);
    end;
  end;
end;
procedure TForm1.DataBis1Click(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlDataBus.Color := Form1.Color;
  Form2.pnlDataBus.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Entry Data Bus';
  Form2.Table1.TableName := 'TBBus';
  Form2.Table1.Open;
  Form2.pnlDataBus.Visible := True;
  Form2.edKode.MaxLength := szBusID;
  Form2.edNopol.MaxLength := szNopol;
  Form2.edPemilik.MaxLength := szPemilik;
  Form2.edAlamat.MaxLength := szAlamat;
//  Form2.edJurusan.MaxLength := szJurusan;
  Table5.First;
  Form2.cbJurusan.Clear;
  if not Table5.IsEmpty then
  begin
    repeat
      Form2.cbJurusan.Items.Add(Table5.FieldValues['Jurusan']);
      Table5.Next;
    until
      Table5.Eof;
  end;
  Form2.edNama.MaxLength := szNama;
```

```
  Form2.ShowModal;
  Form2.Table1.Close;
  Form2.Free;
end;
procedure TForm1.Kedatangan1Click(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlDataBus.Color := Form1.Color;
  Form2.pnlDataBus.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Entry Data Jadwal Kedatangan';
  Form2.pnlDatang.Visible := True;
  Table4.TableName := 'TBBus';
  Table4.Open;
  Table4.First;
  if Table4.IsEmpty or Table5.isEmpty then
  begin
    MessageDlg('Tabel data bus dan dan tabel jalur masih kosong.'+#13
               +'Isi dulu tabel data bus dan tabel jalurnya !',
mtConfirmation,[mbOK],0);
  end else begin
    repeat
      Form2.ComboBox1.Items.Append(Table4.FieldValues['BusID']);
      Table4.Next;
    until
      Table4.Eof;
      Form2.ComboBox1.ItemIndex := 0;
      Form2.ComboBox1Change(Self);
      Form2.Table1.TableName := 'TBDatang';
      Form2.Table1.Open;
      Form2.ShowModal;
  end;
  Form2.Free;
  Table1.Refresh;
end;
procedure TForm1.Persiapan1Click(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlDataBus.Color := Form1.Color;
  Form2.pnlDataBus.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Data Jadwal Persiapan Keberangkatan';
  Form2.Table1.TableName := 'TBPersiapan';
  Form2.Table1.Open;
  Form2.DBGrid1.Visible := False;
  Form2.DBGrid2.Visible := True;
  Form2.pnlLihatdatabis.Visible := True;
  Form2.ShowModal;
  Form2.Free;
end;
procedure TForm1.Keberangkatan1Click(Sender: TObject);
var
  hit : integer;
  stnya : string;
begin
```

```
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlDataBus.Color := Form1.Color;
  Form2.pnlDataBus.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Entry Data Jadwal Keberangkatan';
  Form2.cbKode.Clear;
  Table4.TableName := 'TBBus';
  Table4.Open;
  Table4.First;
  if Table4.IsEmpty or Table5.IsEmpty then
  begin
    MessageDlg('Tabel data bus dan Tabel Jalur Bus masih kosong.'+#13
               +'Isi dulu tabel data bus dan data jalur
pemberangkatannya !', mtConfirmation,[mbOK],0);
  end else begin
    repeat
      Form2.cbKode.Items.Append(Table4.FieldValues['BusID']);
      Table4.Next;
    until
      Table4.Eof;
    Form2.cbKode.ItemIndex := 0;
    Form2.cbKodeChange(self);
    Form2.ComboBox1Change(Self);
    Form2.Table1.TableName := 'TBJadwal';
    Form2.Table1.Open;
    Form2.pnlBerangkat.Visible := True;
    for hit := 0 to 23 do
    begin
      stnya := InttoStr(hit);
      while length(stnya)<2 do
        stnya := '0'+stnya;
      Form2.ComboBox2.Items.Append(stnya);
    end;
    for hit := 0 to 59 do
    begin
      stnya := InttoStr(hit);
      while length(stnya)<2 do
        stnya := '0'+stnya;
      Form2.ComboBox3.Items.Append(stnya);
    end;
    Form2.ComboBox2.ItemIndex := 0;
    Form2.ComboBox3.ItemIndex := 0;
    Form2.ShowModal;
  end;
  Form2.Free;
  Table3.Refresh;
end;
procedure TForm1.btLihat_dataClick(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlLihatdatabis.Color := Form1.Color;
  Form2.pnlLihatdatabis.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Lihat Data Bus';
  Form2.Table1.TableName := 'TBBus';
```

```
  Form2.Table1.Open;
  Form2.DBGrid1.Visible := False;
  Form2.DBGrid2.Visible := True;
  Form2.pnlLihatdatabis.Visible := True;
  Form2.Caption := 'Lihat Data';
  Form2.ShowModal;
  Form2.Free;
end;
procedure TForm1.PeriodeAntrian1Click(Sender: TObject);
var
  stTunggu : string;
  Frcolor,FntColor, intTunggu : integer;
begin
  FrColor := Form1.Color;
  FntColor := Form1.Font.Color;
  stTunggu := InputBox('Konfigurasi','Waktu Tunggu Antrian
(menit):','15',FrColor,FntColor);
  try
    intTunggu := StrtoInt(stTunggu);
  except on EConvertError do
    begin
    MessageDlg('Masukan tidak valid !'+#13
              +'Menggunakan nilai default = 15 menit.', mtError,
[mbOK],0);
    IntTunggu := 15;
    end;
  end;
end;
procedure TForm1.KomunikasiSerial1Click(Sender: TObject);
begin
  Comport1.Close;
  TimerJam.Enabled := False;
  Application.CreateForm(TForm5,Form5);
  Form5.Color := Form1.Color;
  Form5.Font.Color := Form1.Font.Color;
  Form5.ShowModal;
  Comport1.Open;
  TimerJam.Enabled := True;
end;
procedure TForm1.ComPort1AfterClose(Sender: TObject);
begin
  StatusBar1.Panels.Items[0].Text := Comport1.Port;
  StatusBar1.Panels.Items[1].Text := 'Stop';
end;
procedure TForm1.ComPort1AfterOpen(Sender: TObject);
begin
  StatusBar1.Panels.Items[0].Text := Comport1.Port;
  StatusBar1.Panels.Items[1].Text := 'Aktif';
end;
procedure TForm1.JalurBus1Click(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlJalur.Color := Form1.Color;
  Form2.pnlJalur.Font.Color := Form1.Font.Color;
  Form2.pnlJalur.Visible := True;
```

```
  Form2.Label1.Caption := 'Entry Data Jalur Pemberangkatan';
  Form2.Table1.TableName := 'TBJalur';
  Form2.Table1.Open;
  Form2.pnlLihatdatabis.Visible := False;
  Form2.DBGrid1.Visible := True;
  Form2.Caption := 'Jalur Pemberangkatan';
  Form2.ShowModal;
  Form2.Free;
end;
procedure TForm1.Pengumuman(var stbus,stjurusan,stjalur,stnopol :
string ; const stmode : string);
var
  stpos,CurrDir,wavnya,kalimat : string;
  posisi : integer;
begin
  stpos :='';                              // DONE : play wav successively
  if stmode = 'bersiap' then
  begin
    stpos := 'di';
    kalimat := 'bis "'+stbus+'" '+stnopol+' jurusan '+stjurusan+' '+'
silahkan masuk jalur '+stjalur;
  end else begin
    stpos := 'dari';
    kalimat := 'bis "'+stbus+'" '+stnopol+' jurusan '+stjurusan+' '+'
silahkan diberangkatkan';
  end;
  kalimat := Trim(kalimat);
  CurrDir := GetCurrentDir+'\wav\';
  while length(kalimat) > 0 do
  begin
    posisi := AnsiPos('.',kalimat);
    if posisi = 2 then
    begin
      wavnya := lowercase(Trim(LeftStr(kalimat,1)));
      kalimat := Trim(RightStr(kalimat,length(kalimat)-posisi));
      wavnya := CurrDir+wavnya+'.wav';
      PlayWav(wavnya);
    end else begin
      posisi := AnsiPos('"',kalimat);
      if posisi = 1 then
      begin
        kalimat := Trim(RightStr(kalimat,length(kalimat)-1));
        posisi := AnsiPos('"',kalimat);
        wavnya :=  lowercase(Trim(LeftStr(kalimat,posisi-1)));
        kalimat := Trim(RightStr(kalimat,length(kalimat)-posisi));
        wavnya := CurrDir+wavnya+'.wav';
        PlayWav(wavnya);
      end else begin
        posisi := AnsiPos(' ',kalimat);
        if posisi <> 0 then
        begin
          wavnya := Lowercase(Trim(LeftStr(kalimat,posisi)));
          kalimat := Trim(RightStr(kalimat,length(kalimat)-posisi));
          wavnya := CurrDir+wavnya+'.wav';
          PlayWav(wavnya);
        end else begin
          wavnya := CurrDir+lowercase(Trim(kalimat))+'.wav';
```

```
          PlayWav(wavnya);
          kalimat := '';
        end;
      end;
    end;
  end;
end;
function TForm1.Cari_Jadwal(var stkode:string): boolean;
var
  jamskrg,jadwal : TDateTime;
  stjam,stjalur,stjurusan : string;
  hasil : boolean;
begin
  result := false;                        // DONE : Cari jadwal berangkat
yg terdekat
  recnum := 00;
  stkode := RightStr(stkode,10);
  jamnya := IncDay(Now);
  jamskrg := Now();
  Table3.First;
  repeat
    if Table3.FieldValues['BusID'] = stkode then
    begin
      jamskrg := TimeOf(jamskrg);
      stjam := Table3.FieldValues['Jam']+':00';
      jadwal := StrtoTime(stjam);
      if jadwal < jamskrg then
      begin
        jadwal := incday((Now-TimeOf(Now)) + jadwal);
      end;
      if jadwal < jamnya then          // bandingkan dg data sebelumnya
      begin
        jamnya := jadwal;
        recnum := Table3.RecNo;
      end;
      result := True;
    end;
    Table3.Next;
  until
    Table3.Eof;
end;
procedure TForm1.Masukkan_antrian(var stkode : string);
var
  stmode,stjurusan,stjalur,stjam : string;
  hasil : boolean;
  sekarang,jadwal1,jadwal : TDateTime;
begin
  Table3.RecNo := recnum;                 // DONE : Masukkan dalam
antrian dan urutkan
  sekarang := now;
  jadwal := (Sekarang-
TimeOf(Sekarang))+StrtoTime(Table3.FieldValues['Jam']+':00');
  if jadwal < sekarang then
    jadwal := IncDay(jadwal);
  stjalur := Table3.FieldValues['Jalur'];
  stjurusan := Table3.FieldValues['Jurusan'];
  hasil := false;
```

```
  Table2.First;
  if not Table2.Locate('BusID',stkode,[]) then
  begin
    Table2.First;
    repeat
      if Table2.Eof then
      begin
        stmode := '00';
        Table2.AppendRecord([jadwal,stkode,stjalur,stmode]);
        hasil := true;
      end else begin
        jadwal1 := Table2.FieldValues['Jam'];
        if jadwal < jadwal1 then
        begin
          stmode := '00';
          Table2.InsertRecord([jadwal,stkode,stjalur,stmode]);
          Table2.Edit;
          Table2.Post;
          hasil := true;
        end;
      end;
      Table2.Next;
    until
      hasil = true;
  end;
end;
procedure TForm1.Pengumuman_berangkat(var jalur: byte);
var
  sttamp,stjam,stbus,stmode,stkode,stjalur,stjurusan,stnopol : string;
  hasil : boolean;
  jamnya,jamskrg : TDateTime;
  hit : integer;
begin
  hasil := True;                          // DONE : baca tabel persiapan,
umumkan yg berangkat
  stjalur := InttoStr(jalur);
  if length(stjalur) < 2 then
    stjalur := '0'+stjalur;
  Table2.First;
  if Table2.Locate('Jalur',stjalur,[]) then
  begin
    jamnya := Table2.FieldValues['Jam'];
    jamskrg := Now;
    if jamnya<jamskrg then
    begin
      stmode := (Table2.FieldValues['Mode']);
      if stmode = '02' then
      begin
        stkode := Table2.FieldValues['BusID'];
        stjalur := Table2.FieldValues['Jalur'];
        Table4.First;
        Table4.Locate('BusID',stkode,[]);
        stjurusan := Table4.FieldValues['Jurusan'];
        stbus := Table4.FieldValues['Nama'];
        sttamp := Trim(Table4.FieldValues['Nopol']);
        laporan_berangkat(stkode,stbus,sttamp);
        stnopol := '';
```

```
        repeat
          stnopol := stnopol+LeftStr(sttamp,1)+'.';
          sttamp := RightStr(sttamp,length(sttamp)-1);
          sttamp := Trim(sttamp);
        until
          length(sttamp) < 1;
        stnopol := Trim(stnopol);
        stjalur := InttoStr(jalur);
        Pengumuman(stbus,stjurusan,stjalur,stnopol,'berangkat');
      end;
      Table2.Delete;
    end;
  end;
end;
procedure TForm1.Pengumuman_persiapan(var jalur: byte);
var
  sttamp,stjam,stbus,stkode,stmode,stjalur,stjurusan,stnopol :string;
  hasil : integer;
  jamnya,jamskrg : TDateTime;
begin
  stjalur := InttoStr(jalur);           // DONE : baca tabel persiapan,
umumkan yg bersiap
  while length(stjalur) < 2 do
    stjalur := '0'+stjalur;
  Table2.First;
  if Table2.Locate('Jalur',stjalur,[]) then
  begin
    jamnya := Table2.FieldValues['Jam'];
    jamskrg := Now;
    stmode := Table2.FieldValues['Mode'];
    if (jamnya<jamskrg) and (stmode = '01') then
    begin
      Table2.Delete;
    end else begin
      if (SecondSpan(jamskrg,jamnya) < (tmbersiap*60)) and (stmode =
'00') then
      begin
        Table2.Edit;
        Table2.FieldValues['Mode'] := '01';
        stkode := Table2.FieldValues['BusID'];
        stjalur := Table2.FieldValues['Jalur'];
        Table4.First;
        Table4.Locate('BusID',stkode,[]);
        stjurusan := Table4.FieldValues['Jurusan'];
        stbus := Table4.FieldValues['Nama'];
        sttamp := Trim(Table4.FieldValues['Nopol']);
        stnopol := '';
        repeat
          stnopol := stnopol+LeftStr(sttamp,1)+'.';
          sttamp := RightStr(sttamp,length(sttamp)-1);
          sttamp := Trim(sttamp);
        until
          length(sttamp) < 1;
        stnopol := Trim(stnopol);
        stjalur := InttoStr(jalur);
        Pengumuman(stbus,stjurusan,stjalur,stnopol,'bersiap');
        Table2.Post;
```

```
      end;
    end;
  end;
end;
procedure TForm1.Periksa_jalur(var jalur: byte);
var
  querykar : byte;
  dtin,stjalur,stmode : string;
begin
// DONE : periksa kondisi jalur
  querykar := byte(karjlr1) + (jalur-1);        // Kirim 'B' <- jawaban
jlr 0; 1 byte : 0=kosong ; 1=ada
  Comport1.Write(querykar,1);                   // Kirim 'C' <- jawaban
jlr 1; 1 byte : 0=kosong ; 1=ada
  Comport1.ReadStr(dtin,1);                     // Kirim 'D' <- jawaban
jlr 1; 1 byte : 0=kosong ; 1=ada
  stjalur := InttoStr(jalur);
  while length(stjalur) < 2 do
    stjalur := '0'+stjalur;
  Table2.First;
  if Table2.Locate('Jalur',stjalur,[]) then
  begin
    stmode := Table2.FieldValues['Mode'];
    if stmode = '01' then
    begin
      if not(dtin = adabis) then
      begin
         Table2.Edit;
         Table2.FieldValues['Mode'] := '02';
         Table2.Post;
      end;
    end;
  end;
end;
procedure TForm1.Laporan_berangkat(var stkode, stbus, stnopol: string);
var
  sttgljam : string;
begin
  stTgljam := DateTimetoStr(Now);               // DONE : Catat
keberangkatan
  Table6.AppendRecord([stTgljam,stnopol,stbus]);
end;
procedure TForm1.RekapPemberangkatan1Click(Sender: TObject);
begin
  Application.CreateForm(TForm2,Form2);
  Form2.Color := Form1.Color;
  Form2.Font.Color := Form1.Font.Color;
  Form2.pnlDataBus.Color := Form1.Color;
  Form2.pnlDataBus.Font.Color := Form1.Font.Color;
  Form2.Label1.Caption := 'Data Keberangkatan Bus';
  Form2.Table1.TableName := 'TBRekap';
  Form2.Table1.Open;
  Form2.DBGrid1.Visible := False;
  Form2.DBGrid2.Visible := True;
  Form2.pnlLihatdatabis.Visible := True;
  Form2.ShowModal;
  Form2.Free;
```

```
end;
procedure TForm1.PlayWav(var stfile: string);
var
  pFile : Pchar;
begin
  pFile := StrAlloc(length(stfile)+1);
  StrPCopy(pFile,stfile);
  PlaySound(pfile,0,SND_Sync);
end;
procedure TForm1.BitBtn1Click(Sender: TObject);
begin
  if BitBtn1.Caption = '&Mulai' then
  begin
    hitError := 0;
    fRun := True;
    BitBtn1.Caption := '&Stop';
  end else begin
    StatusBar1.Panels[2].Text := '>> Stop Komunikasi';
    fRun := False;
    BitBtn1.Caption := '&Mulai';
  end;
end;
end.
```