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

PERANCANGAN DAN PEMBUATAN SISTEM PARKIR PRABAYAR MENGGUNAKAN KARTU MAGNETIK (KARTU MAHASISWA) YANG DIAPLIKASIKAN PADA TEMPAT PARKIR KAMPUS BERBASIS MIKROKONTROLLER AT89S51

SKRIPSI

Disusun Oleh :
AFIV NOER AZHARI
NIM. 01.17.068

MARET 2007

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN SISTEM PARKIR PRABAYAR MENGGUNAKAN KARTU MAGNETIK (KARTU MAHASISWA) YANG DIAPLIKASIKAN PADA TEMPAT PARKIR KAMPUS BERBASIS MIKROKONTROLLER AT89S51

SKRIPSI

Disusun Dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Elektronika Strata Satu (S1)

Oleh :

AFIV NOER AZHARI
NIM : 01.17.068

Diperiksa dan Disetujui

Ketua Jurusan Teknik Elektro S-1	Dosen Pembimbing
Ir. F. Yudi Limpraptono, MT	**Ir. F. Yudi Limpraptono, MT**
NIP. Y. 1039500274	**NIP. Y. 1039500274**

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ELEKTRONIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2007

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

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	: AFIV NOER AZHARI
N I M	: 01.17.068
Jurusan	: Teknik Elektro
Konsentrasi	: Teknik Elektronika S-1
Judul	: Perancangan dan Pembuatan Sistem Parkir Prabayar Menggunakan Kartu Magnetik (Kartu Mahasiswa) Yang Diaplikasikan Pada Tempat Parkir Kampus Berbasis Mikrokontroller AT89S51

Dipertahankan dihadapan tim penguji skripsi jenjang strata satu (S-1) pada :

Hari : Sabtu
Tanggal : 10 Maret 2007
Dengan Nilai : 83,25

Panitia Ujian Skripsi

Ketua	**Sekretaris**
Ir. Mochtar Asroni MS, ME	**Ir. F. Yudi Limpraptono, MT**
NIP.Y. 1018100036	**NIP. Y. 1039500274**

Anggota Penguji

Penguji I	**Penguji II**
Ir. Usman Djuanda, MM	**Ir. Teguh Herbasuki, MT**
NIP.Y. 1018700143	**NIP.Y. 1028700172**

ABSTRAKSI

Afiv Noer Azhari, 2006, "**Perancangan dan Pembuatan Sistem Parkir Prabayar Menggunakan Kartu Magnetik (Kartu Mahasiswa) Yang Diaplikasikan Pada Tempat Parkir Kampus Berbasis Mikrokontroller AT89S51**", Teknik Elektro S-1 / Teknik Elektronika, Fakultas Teknologi Industri, Institut Teknologi Nasional Malang. Dosen Pembimbing : Ir. F. Yudi Limpraptono, MT.

Kata kunci : Parkir, Magnetic Card, Mikrokontroller AT89S51.

Permasalahan yang terjadi pada sistem parkir kampus sangat beragam. Dari keberagaman masalah yang sangat kompleks tersebut bias berakibat fatal bagi petugas jika tidak mengetahui bagaimana cara mengatasinya. Berangkat dari permasalahan ini muncullah suatu gagasan untuk membuat suatu alat (sistem parkir prabayar menggunakan kartu magnetik) yang bertujuan untuk memberikan kemudahan, efisiensi waktu dan tenaga petugas parkir sehinga menghasilkan pelayanan yang cepat untuk para mahasiswa dalam memasuki area parkir kampus.

Metode pembuatan sistem parkir prabayar ini terdiri dari dua bagian yaitu PC sebagai penyimpan sekaligus pemroses data dan alat sebagai peraga yang diletakkan di tengah-tengah antara pintu masuk dan pintu keluar parkir. Interface alat dihubungkan ke PC melalui COM1 (port komunikasi serial), sedangkan megnetic card reader berfungsi sebagai pembaca kode kartu magnetik yang dihubungkan ke PS/2 keyboard. Daftarkan kartu tersebut dengan menggesekkan pada card reader dan isi data untuk disimpan di database. Gesekkan kartu tersebut pada saat masuk parkir, PC langsung mencocokkan kode kartu tersebut di database, jika tidak ada maka muncul tampilan "Belum Terdaftar" dan jika ada PC langsung mengurangi saldonya, jika tidak cukup maka akan muncul tampilan "Saldo tidak cukup", jika cukup maka palang pintu membuka dan infra red mendeteksi kendaraan yang lewat untuk menutup palang pintu.

Selama pengujian alat ini tidak pernah terjadi kesalahan dalam pengolahan data, baik data yang diproses maupun data yang ditampilkan.

KATA PENGANTAR

Puji dan syukur penulis panjatkan kehadirat Allah SWT, karena atas berkat dan rahmat-Nya penulis dapat menyelesaikan skripsi yang berjudul **"Perancangan dan Pembuatan Sistem Parkir Prabayar Menggunakan Kartu Magnetik (Kartu Mahasiswa) Yang Diaplikasikan Pada Tempat Parkir Kampus Berbasis Mikrokontroller AT89S51"**. Penyusunan skripsi merupakan syarat yang harus ditempuh mahasiswa jurusan Teknik Elektro S-1 Institut Teknologi Nasional Malang untuk memperoleh gelar Sarjana Teknik. Penulis mengucapkan terima kasih kepada :

1. Bapak Prof. Dr. Ir. Abraham Lomi, MSEE selaku Rektor Institut Teknologi Nasional Malang.
2. Bapak Ir. Mochtar Asroni, MSEE selaku Dekan Fakultas Teknologi Industri di Institut Teknologi Nasional Malang.
3. Bapak Ir. F. Yudi Limpraptono, MT selaku Kajur Teknik Elektro S1 di Institut Teknologi Nasional Malang dan juga selaku dosen pembimbing.
4. Teman-teman seperjuangan dan semua pihak yang telah membantu dalam proses penyusunan Skripsi ini.

Penulis menyadari bahwa skripsi ini mungkin kurang sempurna, sehingga penulis mengharapkan saran dan kritik kepada pembaca serta pengamat untuk pengembangan dan penyempurnaan, sehingga mendapatkan hasil yang lebih baik.

Malang, Maret 2007

Penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang Masalah

Dewasa ini perkembangan teknologi elektronika sangat pesat dan maju. Telah banyak ditemukan dan diciptakan peralatan elektronik yang pada dasarnya memberikan kemudahan, efektivitas, penghematan, keunggulan kinerja dan manfaat yang besar bagi kehidupan manusia. Salah satu perkembangan teknologi dalam bidang elektronika adalah teknologi komputer yang berkembang dalam segi hardware dan software.

Pada umumnya kartu mahasiswa dibuat hanya untuk identias mahasiswa itu sendiri. Namun seiring dengan perkembangan teknologi, kartu mahasiwa tersebut berubah bentuk dan fungsi menjadi kartu mahasiswa dan kartu atm (kartu magnetik) dari bank yang menjadi satu. Fasilitas kartu atm pada kartu ini tentunya hanya dipakai untuk pengambilan uang dari atm bank itu sendiri atau mungkin juga digunakan untuk mengurus registrasi perkuliahan. Sebagian besar mahasiswa pada kenyataannya jarang sekali menabung apalagi mengambil uang lewat atm, mungkin dikarenakan sudah mempunyai tabungan di bank yang lain sebelumnya atau dikarenakan kesulitan mengatur keuangannya sendiri, jadi atmnya hanya berfungsi sebagai identitas mahasiswa saja.

Di kampus ITN, sistem parkir kendaraan bermotor itu sendiri sudah diatur sedemikian rupa, namun sistem pembayaran parkir sering kali mangantri panajang yang dikarenakan pencatatan plat nomor untuk karcis, ditambah lagi terkadang mahasiswa lupa tidak membawa uang receh dan juga terkadang melakukan pembayaran uang kembali oleh pencatat plat nomor hal itu wajar kalau memakan waktu lebih. Melalui perkembangan teknologi tersebut, bermaksud membuat suatu sistem parkir prabayar dengan kartu magnetik (kartu mahasiswa) yang diimplementasikan menggunakan komputer. Penggunaan teknologi ini diharapkan dapat memfungsikan lebih kartu mahasiswa tersebut dan juga membantu petugas parkir kampus dalam menjalankan tugasnya, menguntungkan dalam segi finansial, efisiensi waktu dan tenaga dimana setiap kali memasuki tempat parkir kampus tidak terjadi antrian.

1.2. Rumusan Masalah

Dengan melihat latar belakang yang telah diuraikan sebelumnya, inti permasalahan dalam perancangan system tersebut adalah :

1. Bagaimana merencanakan dan membuat perangkat keras dan perangkat lunak yang dapat mengendalikan system.
2. Pemakaian card reader magnetik untuk mengetahui nomor registrasi kartu magnetik (kartu mahasiswa).
3. Pemakaian komponen bantu berupa RS 232 untuk melakuakan interfacing dalam pengolahan data.
4. Membuat perangakat lunak untuk PC dan perangkat lunak untuk sistem dalam hal ini adalah mikrokontroller.

1.3. Batasan Masalah

Hal-hal yang tidak dibahas dalam penulisan ini diantaranya :

1. Perangkat keras komputer.
2. Perangkat lunak dari Borland Delphi.
3. Pengisian data pada kartu magnetik.
4. Satu magnetic card reader untuk satu jalur masuk dan satu jalur keluar.
5. Pencocokan plat nomor kendaraan.
6. Hanya menggunakan satu PC.

1.4. Tujuan

Tujuan dari perancangan dan pembuatan alat ini adalah :

- Merancang dan membuat suatu model sistem parkir prabayar menggunakan kartu magnetik (kartu mahasiswa) sehingga dapat memberikan kemudahan yang lebih efektif dan efisien terhadap waktu dan tenaga manusia serta dapat menjamin keamanan kendaraan bermotor yang parkir didalam kampus ITN.

1.5. Metodologi Perencanaan

Langkah-langkah yang diambil untuk menyelesaikan perubahan teoritis pada perancangan dan pembuatan sistem parkir prabayar menggunakan kartu magnetik (kartu mahasiswa) berbasis mikrokontroller AT89S51 adalah :

1. Studi literatur tentang teori sistem kontrol menggunakan mikrokontroller secara umum dan teori tentang komponen yang akan digunakan.

2. Perencanaan perangkat keras yaitu perancangan blok yang berhubungan dengan kinerja alat yang digunakan dan perencanaan perangkat lunak.
3. Pembuatan alat antara lain meliputi pembuatan PCB, perakitan komponen serta penyoderan dan pembuatan perangkat lunak.
4. Pengujian alat yaitu melakukan pengujian setiap blok rangkaian dan kerja seluruh sistem.

1.6. Sistematika Penulisan

Sistematika penulisan yang digunakan dalam penyusunan laporan skripsi ini adalah sebagai berikut :

BAB I : Menjelaskan tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan sistematika penulisan.

BAB II : Menjelaskan tentang teori dasar yang berisi tentang prinsip dasar dari mikrokontroller dan rangkaian pendukung.

BAB III : Menjelaskan perencanaan dan pembuatan alat.

BAB IV : Menjelaskan tentang pengujian atat yang telah direalisasikan.

BAB V : Menjelaskan tentang kesimpulan dan saran.

BAB II

LANDASAN TEORI

2.1. Pendahuluan

Landasan teori ini sangat membantu untuk dapat memahami suatu sistem. Disamping itu dapat juga dijadikan sebagai bahan acuan didalam merencanakan suatu sistem. Dengan pertimbangan hal-hal tersebut maka landasan teori merupakan bagian yang harus dipahami untuk pembahasan selanjutnya. Bagian-bagian yang akan dibahas diantaranya :

- Mikrokontroller AT89S51.
- Pembaca Kartu Magnetik.
- Motor DC.
- Sensor Infra Merah.
- LCD (Liquid Crystal Display)
- Interface RS MAX 232
- Buzzer
- Relay
- ULN 2003

2.2. Mikrokontroller AT89S51

Perbedaan mendasar antara mikrokontroller dan mikroprosesor adalah mikrokontroller selain memiliki CPU juga dilengkapi dengan memori *I/O* yang merupakan kelengkapan sebagai suatu ***minimum system*** mikrokomputer sehingga mikrokontroller dapat dikatakan sebagai mikrokomputer dalam keping tunggal (*single chip microcomputer)* yang dapat berdiri sendiri.

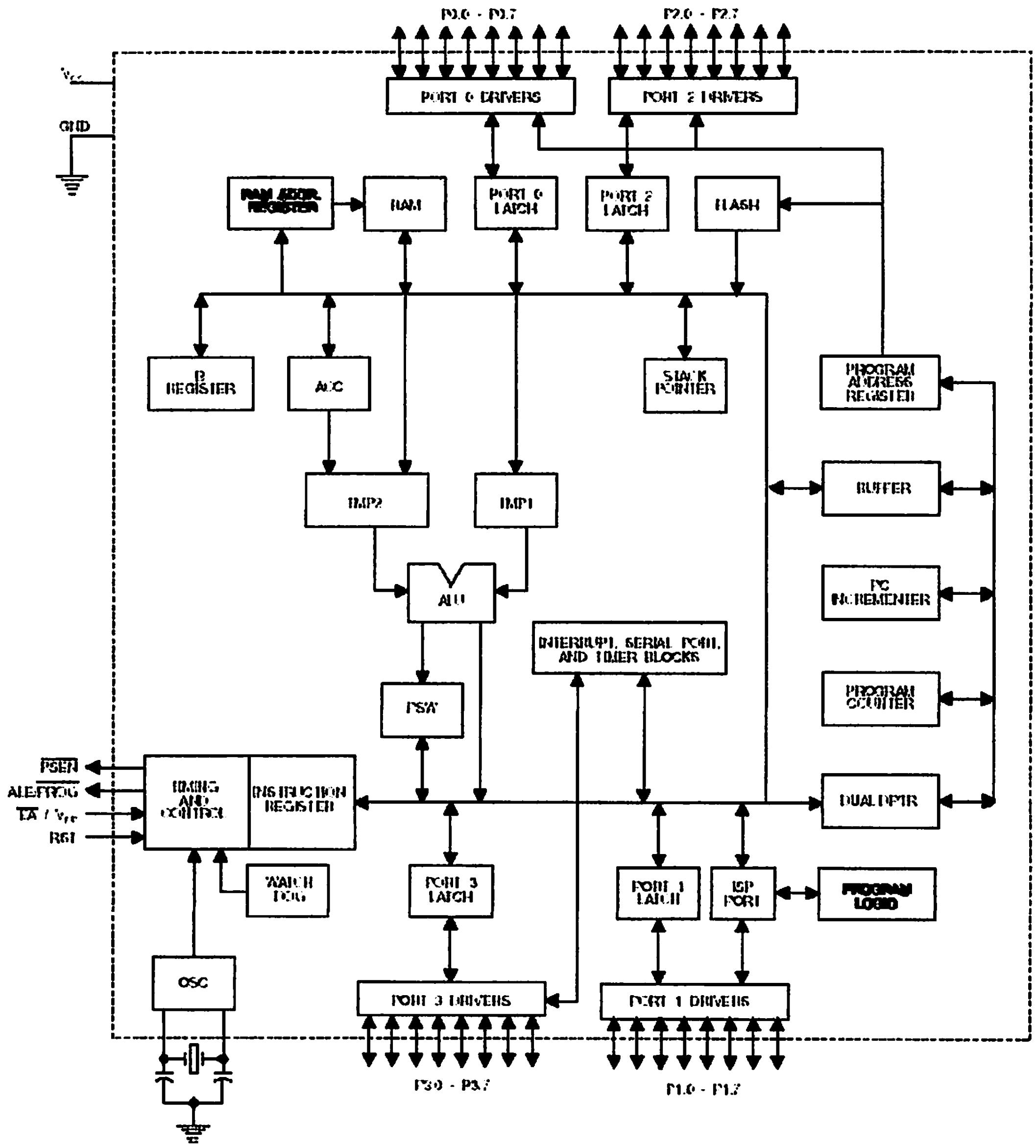
Mikrokontroller AT89S51 adalah mikrokontroller ATMEL *kompatibel* penuh dengan mikrokontroller keluarga MCS-51, dengan *supply* daya rendah, memiliki *performance* yang tinggi dan merupakan mikrokontroller 8 bit yang dilengkapi 4 Kbyte EPROM (*Enable and Programmable Read Only Memory*) dan 128 Byte RAM *internal.* Program memori dapat diprogram ulang dalam sistem atau dengan menggunakan *Programmer Nomolately Memory Konvensional.*

Dalam sistem mikrokontroller terdapat dua hal yang mendasar, yaitu: perangkat keras dan perangkat lunak yang keduanya saling terkait dan mendukung.

2.2.1. Perangkat keras Mikrokontroller AT89S51

Mikrokontroller AT89S51 secara umum memiliki:

- CPU 8 bit
- *Memory*
- *Port I/O*
- *Timer* dan *Counte*r
- Sumber *Interupt*
- *Program Serial* yang dapat diprogram
- Osilator dan Clock



Gambar 2.1. Blok diagram Mikrokontroller AT89S51 [1]

2.2.2. Arsitektur AT89S51

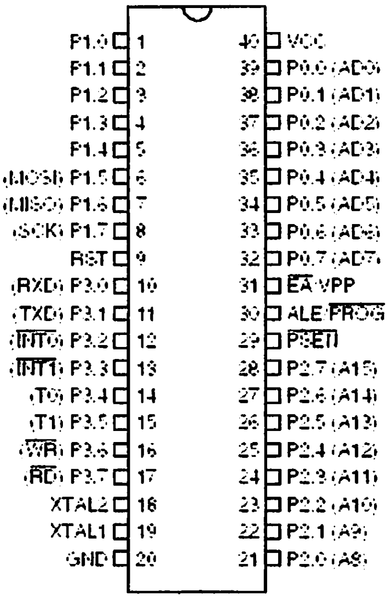
Arsitektur mikrokontroller AT89S51 adalah sebagai berikut :

1. CPU (*Central Processing Unit*) 8-bit dengan *register* A (*accumulator*) dan B
2. 16-Bit *Program Counter* (PC) dan *Data Pointer* (DPTR).
3. 8- Bit *Program Status Word* (PSW).

4. 4-Bit *Stack Pointer* (SP).
5. 4 Kbyte *internal* EPROM.
6. 128 byte *internal* RAM.
   - 4 *bank register*, masing-masing berisi 8 byte
   - 16 byte alamat serbaguna yang dapat diakses sebagai byte atau bit, tergantung *software* yang digunakan.
   - 80 byte *general purpose memory* data.
7. 32 pin *input-output* tersusun atas P0-P3, masing-masing 8-bit.
8. 2 buah 16-bit *Timer/Counter.*
9. 2 buah *port serial full duplex*
10. *Control Register*, antara lain : TCON, SCON, PCON, IP, dan IE
11. 5 buah sumber interupsi (2 buah sumber interup eksternal dan 3 buah sumber *internal*).
12. Osilator dan *Clock Internal.*
13. *Watch Dog Programmable Timer.*
14. ISP *Port* .

2.2.3. Penjelasan Fungsi Pin AT89S51

Mikrokontroller AT89S51 mempunyai 40 pin seperti yang ditunjukkan dalam gambar 2.2. Fungsi-fungsi pin dijelaskan sebagai berikut :



Gambar 2.2. Susunan Pin AT89S51 [1]

Fungsi dari tiap-tiap pin adalah sebagai berikut :

1. VCC, *Pin 40*

   Merupakan pin positif sumber tegangan 5 volt DC.

2. GND (ground), *Pin 20*

   Merupakan pin grounding sumber tegangan.

3. Port 0, *Pin 32 - 39*

   Merupakan port input dua arah dan dikonfigurasikan sebagai *multipleks* dua bus alamat rendah (A0-A7) dan data selama pengaksesan program memori dan data internal.

4. Port 1, *Pin 1 – 8*

   Merupakan port input dua arah dengan pull-up dan juga menerima *Low-order address byte* selama memprogram dan verifikasi dari flash. Pada mikrokontroller AT89S51 port 1 memikili 3 pin dengan fungsi khusus.

Tabel 2.1. Fungsi Khusus Port 1 [1]

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

5. Port 2, *Pin 21 - 28*

   Merupakan port I/O dengan *internal pull-up*. Mengeluarkan *address* tinggi selama pengambilan (*fetching*) program memori external. Selama pengaksesan ke external data memori, port 2 mengeluarkan isi SFR (*Special Function Register*). Menerima *address* dan beberapa sinyal control selama pemrograman.

6. Port 3, *Pin 10 - 17*

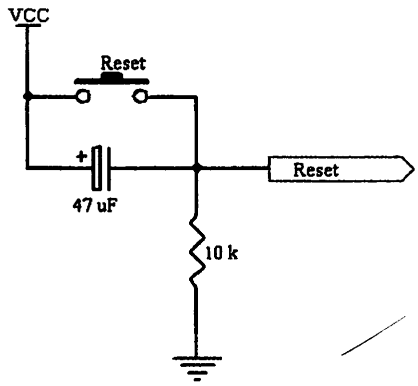
   Merupakan port I/O dengan *internal pull-up*. Port 3 juga memiliki fungsi khusus, yaitu:

Tabel 2.2. Fungsi Khusus 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*)

7. Reset, *Pin 9*

Perubahan taraf tegangan dari rendah ketinggi akan mereset AT 89S51.



Gambar 2.3. Rangkaian Reset [1]

8. ALE/PROG, *Pin 30*

   Pulsa output ALE digunakan untuk proses-proses '*latching*' *byte address* rendah (A0-A7) selama pengaksesan ke external memori. Pin ini juga digunakan untuk memasukkan pulsa program (prog) selama pemrograman.

9. PSEN, *Pin 29*

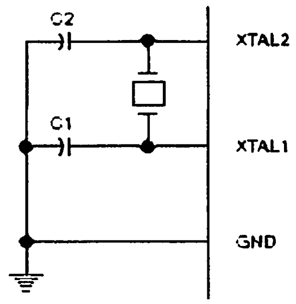
   Merupakan strobe baca ke program memori ekternal.

10. EA/VPP, *Pin 31*

    External Address Enable (EA) digroundkan jika mengakses memori ekternal, akan dihubungkan ke VCC jika digunakan untuk mengakses memori internal.

11. X-TALL 1 dan X-TALL 2, *Pin 19 dan Pin 18*

    Kaki ini dihubungkan dengan kristal bila menggunakan osilator internal. XTALL 1 merupakan input inverting osilator amplifier sedangkan X-TALL 2 merupakan output inverting osilator amplifier.



Gambar 2.4. Rangkaian Clock [1]

2.2.4. Organisasi Memori

Organisasi memori pada mikrokontroller AT89S51 dapat dibagi menjadi dua bagian besar yaitu memori program dan memori data. Pembagian tersebut didasarkan atas fungsi dari penyimpangan data maupun program. Memori program digunakan untuk menyimpan instruksi-instruksi yang akan diajarkan oleh mikrokontroller, sedangkan memori data digunakan sebagai tempat menyimpan instruksi-instruksi yang sedang diolah mikrokontroller.

Program mikrokontroller disimpan dalam memori program berupa ROM. Mikrokontrller AT89S51 dilengkapi dengan ROM internal, sehingga untuk menyimpan program tidak digunakan ROM ekstrernal yang terpisah dari mikrokontroller. Agar tidak menggunakan memori program eksternal, EA (*Eksternal Address enable*) dihubungkan dengan Vcc.

Memori program mikrokontroller menggunakan alamat 16 bit mulai 0000_H-$0FFF_H$, sehingga kapasitas penyimpanan program maksimal adalah 4Kbyte. Sinyal/PSEN (*Program Strobe Enable*) tidak digunakan jika menggunakan memori internal.

Selain program mikrokontroller AT 89S51 juga memiliki data internal 128 *byte* dan mampu mengakses memori data eksternal sebesar 64 Kb. Semua memori data internal dapat dialamati dengan data langsung atau tidak langsung. Ciri dari pengalamatan langsung adalah *operand* yaitu alamat *register* yang berisi alamat data yang akan diolah. Sebagian memori tersebut dapat dialamati dengan memori satu bit. Untuk membaca data digunakan sinyal/RD sedangkan untuk menulis data digunakan sinyal/WR.

2.2.5. SFR (*Special Function Register*)

Register Fungsi Khusus (*Special Function Register*) terletak pada 128 byte bagian atas memori data internal dan berisi *register-register* untuk pelayanan *latch port, timer, program status words, control peripheral* dan sebagainya. Alamat register fungsi khusus ditunjukkan pada tabel 2.3 dibawah ini:

Tabel 2.3. Special Function Register [1]

Simbol	Nama Register	Alamat
ACC	*Accumulator*	$E0_H$
B	*Register* B	$F0_H$
PSW	*Program Status Word*	$D0_H$
SP	*Stack Pointer*	81_H
DPTR	*Data Pointer* 2 Byte	
DPL	Bit Rendah	82_H
DPH	Bit Tinggi	83_H
P0	*Port* 0	80_H

P1	*Port* 1	90_H
P2	*Port* 2	$A0_H$
P3	*Port* 3	$B0_H$
IP	*Interupt Priority Control*	$D8_H$
IE	*Interupt Enable Control*	$A8_H$
TMOD	*Timer/Counter Mode Control*	89_H
TCON	*Timer/Counter Control*	88_H
TH0	*Timer/Counter 0 High byte*	$8C_H$
TL0	*Timer/Counter 0 Low byte*	$8A_H$
TH1	*Timer/Counter 1 High byte*	$8D_H$
TL1	*Timer/Counter 1 Low*	$8B_H$
SCON	*Serial Control*	98_H
SBUF	*Serial Data Buffer*	99_H
PCON	*Power Control*	87_H

Beberapa macam *register* fungsi khusus yang sering digunakan adalah sebagai berikut :

- *Accumulator* (ACC) merupakan *register* untuk penambahan dan pengurangan. Perintah *mnemonic* untuk mengakses akumulator disederhanakan sebagai A.
- *Register* B merupakan *register* khusus yang berfungsi melayani operasi perkalian dan pembagian.
- *Register R* merupakan delapan set *register* yang dinamakan R0, R1, R2, R3, R4, R5, R6 dan R7, fungsi dari *register-register* ini adalah sebagai *register*

yang membantu penyimpanan data yang menggunakan banyak operasi. *Register-register* ini yang membantu akumulator dalam melakukan operasi antara dua operan.

- *Stack Pointer* (SP) merupakan *register* 8 bit yang dapat diletakkan dialamat manapun pada RAM *internal.*
- *Data Pointer* (DPTR) terdiri dari dua *register*, yaitu *register* untuk byte tinggi *(Data Pointer High*, DPH) dan register untuk byte rendah (*Data Pointer Low*, DPL) yang berfungsi untuk mengunci alamat 16 bit. DPTR berfungsi untuk menunjuk suatu lokasi data, namun pada beberapa perintah DPTR digunakan untuk mengakses memori eksternal.
- *PC (Program Counter)* merupakan alamat 16 bit yang menginstruksikan AT89S51 alamat instruksi yang selanjutnya akan dilaksanakan. Saat inisialisasi AT89S51, PC terisi dengan 00000h dan akan bertambah satu setiap kali instruksi telah dilaksanakan. Harga PC tidak dapat langsung dirubah dengan menggunakan perintah MOV PC,2340h, namun dengan perintah LJMP 2340 yang akan mengisi PC dengan 2340h..
- *Program Status Word* (PSW) berisi bit-bit status yamg berkaitan dengan kondisi CPU saat itu. PSW terletak pada alamat D0H.

PSW
D0H

PSW.7	PSW.6	PSW.5	PSW.4	PSW.3	PSW.2	PSW.1	PSW.0
CY	AC	F0	RS1	RS0	OV	-	P

1. CY (*Flag Carry*)

   Flag carry, yang terletak pada alamat D7H, berfungsi sebagai pendeteksi terjadinya kelebihan pada operasi penjumlahan, atau terjadinya peminjaman *(borrow)* pada operasi pengurangan. Misalnya, jika data pada akumulator adalah FFH dan dijumlahkan dengan bilangan satu atau lebih, maka akan terjadi kelebihan sehingga akan membuat *carry* menjadi *set.* Demikian juga apabila data pada akumulator adalah 00H dan dikurangkan dengan bilangan satu atau lebih, akan terjadi peminjaman sehingga membuat *carry* juga menjadi *set.*

2. AC (*Flag Auxiliary Carry*)

   Flag auxiliary carry akan selalu dalam kondisi *set* apabila pada saat proses penjumlahan terjadi *carry* dari bit ketiga hingga bit keempat.

3. *Flag* 0

   Flag 0 dapat digunakan untuk tujuan umum tergantung pada kebutuhan pemakai.

4. RS *(Register Select)*

   Bit Pemilih *Bank Register (Register Bank Select Bits)* RS0 dan RS1 digunakan untuk menentukan lokasi dari *bank register* (R0-R7) pada memori. RS0 dan RS1 selalu bernilai 0 setiap kali sistem di reset sehingga lokasi dari *register* R0 hingga R7 akan berada pada alamat 00H hingga 07H.

5. OV (*Flag Overflow*)

   Flag overflow akan berada pada kondisi set jika pada operasi aritmatik menghasilkan bilangan yang lebih besar daripada 128 atau lebih kecil dari -128.

6. P (bit paritas)

   Bit paritas akan berada pada kondisi set jika jumlah bit 1 dalam akumulator adalah ganjil dan akan berada pada kondisi *clear* jika jumlah bit 1 dalam akumulator adalah genap. Misalnya, data yang tersimpan pada akumulator adalah 10101110b atau AEH maka *parity bit* akan berada pada kondisi *set*. Data AEH mempunyai lima bit yang berkondisi 1 atau dapat disebut mempunyai bit 1 dalam jumlah yang ganjil.

- *Port* 0 sampai *Port* 3 merupakan *register* yang berfungsi untuk membaca dan mengeluarkan data pada port 0,1,2 dan 3. Masing-masing *register* ini dapat dialamati per byte maupun per bit.
- *Control Register* terdiri dari register yang mempunyai fungsi kontrol. Untuk mengontrol sistem interupsi, terdapat dua *register* khusus yaitu *register* IP (*Interupt Priority*) dan *register* IE (*Interupt Enable*). Untuk mengontrol *timer/counter* terdapat dua *register* khusus yaitu register TCON (*Time Counter Control*) serta *port serial* menggunakan *register* SCON (*Serial Port Control*).

2.2.6. Metode Pengalamatan

1. Pengalamatan Langsung

   Pengalamatan langsung dilakukan dengan memberikan nilai ke suatu *register* secara langsung. Untuk melaksanakan pengalamatan langsung digunakan tanda #.

   Contoh : MOV A, #0A

2. Pengalamatan Tak Langsung

Operand pengalamatan tak langsung menunjuk ke sebuah *register* yang berisi lokasi alamat memori yang akan digunakan dalam operesi. Lokasi yang nyata tergantung pada isi *register* saat instruksi dijalankan. Untuk melaksanakan pengalamatan tak langsung digunakan simbol @.

Contoh : ADD A, @RO

2.2.7. Masukan dan Keluaran

Untuk saluran dan keluaran terdapat 4 buah *port* yang masing-masing 8 bit. Saluran ini bersifat dua arah (*bidirectional*) yang berarti dapat difungsikan sebagai masukan atau keluaran, serta dapat dialamati per bit. *Port* 3 selain digunakan sebagai *port* masukan dan keluaran juga dapat digunakan sebagai fungsi pengganti sebagaimana yang terdapat dalam tabel 2.4. berikut ini :

Tabel 2.4. Fungsi Pengganti Port 3 [1]

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

2.2.8.Timer dan Counter

Dalam mikrokontroler AT89S51 terdapat tiga buah pewaktu/pencacah (timer/counter 16) 16 bit yang dapat diatur melalui perangkat lunak, yaitu pewaktu/pencacah 0 dan pewaktu/pencacah 1. Timer/counter ini diatur oleh *special function* register yaitu Timer/Counter *Control* (TCON alamat 88H), dan Timer/Counter Mode *Control* (TMOD alamat 89H). Selain itu nilai byte bawah dan byte atas dari Timer/Counter disimpan dalam register TL dan TH.

Jika difungsikan sebagai Timer, maka akan menggunakan system clock sebagai sumber masukan pulsanya. Jika sebagai Counter (pencacah), maka akan menggunakan pulsa dari luar (eksternal) sebagai masukan pulsanya. Pada Port 3 terdapat fungsi khusus yaitu TO (masukan luar untuk Timer/Counter 0) dan T1 (masukan luar untuk Timer/Counter 1). Pemilihan mode Timer/Counter dikontrol oleh register TMOD. Dengan memberikan nilai tertentu pada register TMOD dapat dipilih mode operasi untuk Timer/Counter 0 dan Timer/Counter 1 seperti terlihat dalam Tabel 2.5. berikut ini :

Tabel 2.5. Mode Operasi Timer/Counter 0 dan 1 [1]

M2	M1	Mode	Keterangan
0	0	0	13 bit Timer
0	1	1	16 bit Timer
1	0	2	8 bit auto-reload
1	1	3	Split Mode

Timer 2 ini merupakan Timer/Counter 16 bit dan memiliki 3 mode operasi yaitu *capture, auto reload* (*up down counting*) dan baud rate generator. Untuk memilih mode ini dilakukan dengan mengatur bit pada SFR T2CON (Timer 2 Control

Register). Timer 2 ini terdiri dari 2 buah timer 8 bit register yaitu TH2 dan TL2. pada fungsi Timer, register TL2 dinaikkan (increment) tiap siklus mesin. Karena siklus mesin terdiri dari 12 periode osilasi, maka count rate menjadi 1/12 dari frekuensi osilator. Sedangkan pada fungsi Counter, register dinaikkan berdasarkan tanggapan adanya transisi tinggi ke rendah pada pena yang bersesuaian (dalam hal ini pin T2 atau P1.0). Tabel berikut menunjukkan mode operasi yang dapat dijalankan pada timer 2.

Tabel 2.6. Mode Operasi Timer 2 [1]

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

RCLK = Receive clock enable. Jika diset menyebabkan serial port menggunakan pulsa overflow Timer 2 sebagai detak penerimaan pada serial port. Jika RCLK = 0 Timer 1 yang digunakan.

TCLK = Transmit clok enable. Jika diset menyebabkan serial port menggunakan pulsa overflow Timer 2 sebagai detak pengiriman. Jika TCLK = 0 pulsa overflow timer 1 yang digunakan.

CP/RL2 = Pemilihan capture/reload. Jika diset maka proses capture yang terjadi sedangkan jika bit ini diclear maka proses reload.

TR2 = Bit untuk mengatur start/stop untuk timer 2 jika TR2 = 1 Timer akan aktif.

2.2.9.Idle Mode

Saat *Idle Mode* mikrokontroler tidak melakukan apa-apa. Tetapi peralatan lain yang terhubung tetap aktif. Kondisi ini dapat dihentikan dengan sebuah *interrupt* atau dengan me-*reset* system.

2.2.10. Sistem Interupt

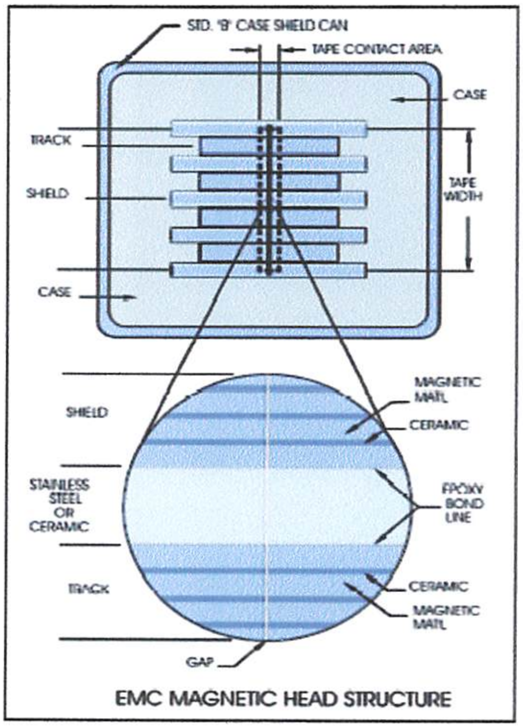
Mikrokontroler AT89S51 mempunyai 6 buah sumber interrupt yang dapat membangkitkan permintaan interrupt, yaitu INTO, INT1, T0, T1, T2 dan port serial.

Saat terjadinya interupt, mikrokontroler secara otomatis akan menuju ke *sub rutin* pada alamat tersebut. Setelah interrupt service selesai dikerjakan, mikrokontroler akan mengerjakan program semula. Dua sumber interrupt *externa*l adalah INTO dan INT1, dimana kedua interupsi eksternal akan aktif atau aktif transisi tergantung isi dari IT0 dan IT1 pada register TCON. Interupsi T0, T1, T2 aktif pada saat timer yang sesuai mengalami *roll over*, interupsi serial dibangkitkan dengan melakukan operasi OR pada R1 dan T1. Tiap-tiap sumber interupsi dapat *enable* atau *disable* secara otomatis.

Tingkat prioritas semua sumber interupsi dapat diprogram sendiri-sendiri dengan set atau *clear bit* pada SFRS IP (*interrupt Priority*).

2.3. Pembaca Kartu Magnetik

2.3.1. Struktur Head Magnetik



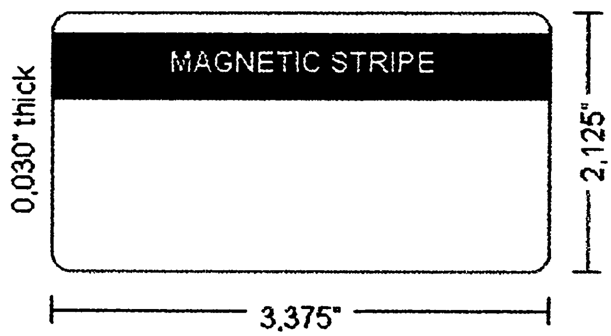
Gambar 2.5. Struktur Head Magnetik [2]

Head pembaca kartu magnetik terbuat dari bahan keramik sehingga mengasilkan permukaan yang tahan gesekan sehingga dapat digunakan dengan kecepatan 10 – 100 cm/s. Head inilah yang akan membaca atau merekam data fluks magnetik pada kartu magnetik ketika digesekkan.

Head magnetik ini kita fungsikan sebagai kartu magnetik, dimana kartu magnetik telah terisi data fluks magnetik digesekkan melalui head ini. Proses selanjutnya dilakukan oleh decoder head magnetik yang akan mengubah fluks tersebut menjadi sinyal digital, sehingga dapat diproses melalui mikrokontroller.

2.3.2. Kartu Magnetik

Kartu magnetik terbuat dari media magnetis yang berbahan keramik setebal 3 – 10 μ" dan beberapa terdapat beberapa jalur kecil yang disatukan (lamination). Ukuran kartu mangacu pada standart internasional (ISO) yaitu dengan panjang 3,375", lebar 2,125" dan tebal 0,03" seperti gambar 2.7.



Gambar 2.6. Kartu Magnetik Standar [2]

2.3.3. Proses Pengolahan Data

Pembaca kartu magnetik dapat diinterfacekan ke μCU melalui berbagai cara. Pada umumnya cara yang digunakan ada 2 yaitu pemrograman input 1 bit dan UART (Universal Asyncronous Reciver Transmitter).

2.3.3.1. Pemrograman Input 1 bit

Metode interface ini tidak memerlukan chip external untuk menerapkan data serial ke μCU. Fungsi ini digunakan melalui program (perangkat lunak) yang menyebabkan μCU dapat menerima atau mengirimkan data. Proses ini memerlukan pemrograman pewaktuan yang sangat kritis (ketepatan tinggi). Kerugiannya adalah ketika μCU menerima data serial, μCU harus difokuskan untuk fungsi ini saja, karena pewaktuan yang akurat hanya bisa diatur jika program tetap berada di dalam loop perwaktuan yang ketat tanpa dialihkan ke fungsi yang lain.

2.3.3.2. UART (Universal Asyncronous Reciver Transmitter)

UART adalah proses pengiriman dan penerimaan data secara asinkron, keuntungan yang diperoleh dengan menggunakan sistem ini adalah μCU tidak memerlukan pewaktuan pemrograman yang akurat sehingga tingkat akurasinya tinggi. Sinyal card present dapat dihubungkan pada pin input UART. Chip UART yang digunakan dalam pembacaan kartu magnetik adalah jenis F/2F yang dapat langsung dihubungkan ke head magnetik kemudian membaca flux magnetik dari kartu magnetik, selanjutnya mengkode dan merubahnya menjadi dinyal digital. Output dari IC F/2F ini kompatibel dengan TTL, sehingga dapat langsung digunakan bersama μCU.

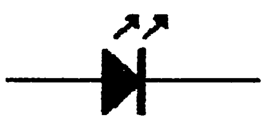
Dalam setiap track pada kartu magnetik terdapat perekaman 75-210 bpi (bit per inchi) dengan konfigurasi karakter 5-7 bit per karakter. Karakter yang digunakan berupa numerik maupun alfa numerik. Konversi karakter dapat dilihat pada daftar lampiran. Dalam laporan akhir ini penulis menggunalan pembaca kartu magnetik (Magnetic Card Reader) jenis MSR Series seperti gambar berikut :

Gambar 2.7. Magnetic Stripe Reader [2]

2.4. Sensor Infra Merah

2.4.1. LED Infra Merah

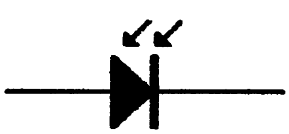
LED (Light Emitting Dioda) disini digunakan sebagai sumber cahaya yang mempunyai panjang gelombang sebesar 10-3 sampai 10-6, dengan frekuensi sebesar 1011 sampai 1014 Hz. Cahaya yang dipancarkan infra merah tergantung pada arus listrik yang masuk, semakin dipenuhi kebutuhan akan arus maksimalnya maka semakin terang cahaya yang dihasilkan.



Gambar 2.8. Simbol LED infra merah [3]

2.4.2. Photodioda

Photodioda adalah p-n junction semikonduktor yang memiliki daerah operasi terbatas pada daerah reverse bias. Dasar pengaturan arah arus (biasing) konstruksi dan simbol terdapat pada gambar dibawah ini.



Gambar 2.9. Simbol Photodioda [4]

Apliaksi cahaya ke junction dihasilkan dalam sebuah transfer energi dari peristiwa perjalanan gelombang cahaya (dalam bentuk foton) ke struktur atomik, menghasilkan peningkatan jumlah karier minoritas dan suatu peningkatan level arus balik (reverse current). Arus 'gelap' (dark current) adalah arus yang muncul saat tidak ada cahaya (penerangan), arus dapat kembali 0 dengan bias positif = V_T.

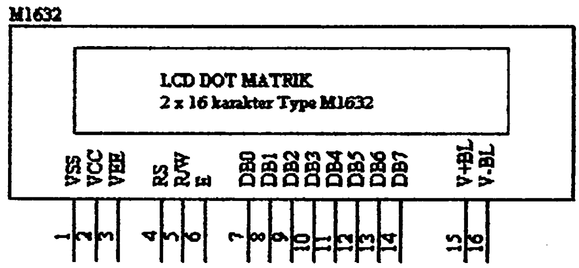
2.5. LCD (*Liquid Crystal Display*)

LCD merupakan komponen optoelektronik yaitu komponen yang bekerja atau dipengaruhi oleh sinar (optolistrik), komponen pembangkit cahaya (light emiting) dan komponen-komponen yang akan mengubah sinar. LCD terbuat dari bahan kristal cair yang merupakan suatu komponen organik dan mempunyai sifat optik seperti benda padat meskipun bahan tetap cair.

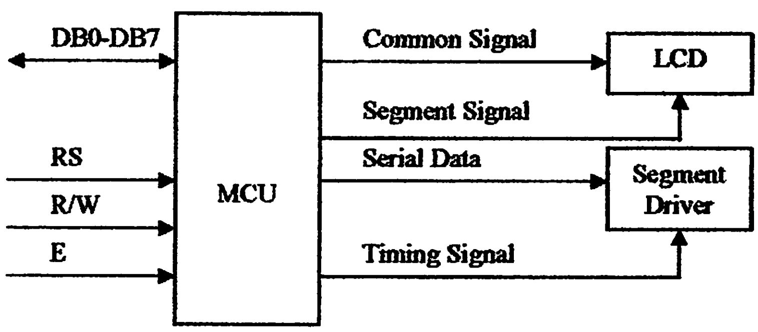
Sel kristal cair terdiri dari selapis bahan kristal cair yang diapit antara dua kaca tipis yang transparan. Antara dua lembar kaca tersebut diberi bahan kristal cair (*liquid crystal)* yang tembus cahaya. Permukaan luar dari masing-masing keping kaca mempunyai lapisan penghantar tembus cahaya seperti oxida timah (*tin oxide)* atau oxida indium (*indium oxide)*. Sel mempunyai ketebalan sekitar 1×10^{-5} meter dan diisi dengan kristal cair.

Karena sel-sel kristal cair merefleksikan cahaya dan bukan membangkitkan cahaya maka konsumsi daya yang dibutuhkan relatif rendah. Energi yang dipergunakan hanya untuk mengaktifkan kristal cair. Pada dasarnya LCD bekerja pada tegangan rendah (3 – 15 Vrms), frekuensi rendah (25 – 60 Hz) sinyal AC dan memakai arus listrik yang sangat kecil (25 - 300 μA). LCD seringkali ditata sebagai tampilan *seven segment* untuk menampilkan angka tetapi juga memiliki keistimewaan lain, yaitu kemampuan untuk menampilkan karakter dan berbagai macam simbol.

Salah satu jenis LCD diantaranya adalah LCD M1632, suatu jenis piranti dengan konsumsi daya yang rendah, disusun dari dot matrik dan dikontrol oleh ROM atau RAM generator karakter dan RAM data display. Pengontrolan utamanya adalah pada ROM generator dan display data RAM yang menghasilkan kode ASCII jika padanya diberikan input ASCII. Untuk dapat difungsikan dengan baik maka perlu diperhatikan proses analisasi yang telah ditentukan oleh pabrik pembuatnya. Timing penganalisasian sangat dipertimbangkan, karena jika meleset sampai ordo *milisecon* maka dapat dipastikan LCD tidak dapat berfungsi.



Gambar 2.10. Konfigurasi Kaki LCD [5]



Gambar 2.11. Blok Diagram LCD [5]

Adapun karakteristik dari LCD M1632 antara lain :

- Dengan 16 karakter 2 baris dalam bentuk dotmatrik 5x7 dan cursor.
- *Duty ratio* 1/16.
- Memiliki ROM pembangkitan karakter untuk 192 jenis karakter.
- RAM untuk data display sebanyak 80x8 bit.
- Dapat dirangkai dengan MPU 8 bit/4 bit.

- RAM data display dan RAM pembangkit karakter dapat dibaca oleh MPU.
- Memiliki fungsi instruksi antara lain *display on/off, Cursor on/off, display* karakter *blink, cursor shift* dan *display shif.t*
- Memiliki rangkaian osilator sendiri.
- Catu tegangan tunggal yaitu ± 5 V.
- Memiliki rangkaian reset otomatis pada catu daya yang dihidupkan.
- Temperatur operasi 0^0 - 50^0.

LCD memiliki 16 pin yang masing-masing mempunyai fungsi sebagai berikut :

Tabel 2.7. Fungsi Tiap-tiap Pin LCD [5]

<table>
<tr><th>No.Pin</th><th>Simbol</th><th>Level</th><th colspan="2">Fungsi</th></tr>
<tr><td>1</td><td>V_{SS}</td><td>-</td><td rowspan="3">Power Supply</td><td>0 V (GND)</td></tr>
<tr><td>2</td><td>V_{CC}</td><td>-</td><td>5 V ± 10%</td></tr>
<tr><td>3</td><td>V_{DD}</td><td>-</td><td>For LCD Drive</td></tr>
<tr><td>4</td><td>RS</td><td>H/L</td><td colspan="2">Sinyal seleksi register
H ; Data Input [register data (write/read)]
L ; Instruction Input [register instruksi (write), busy flag dan address counter (read)]</td></tr>
<tr><td>5</td><td>R/W</td><td>H/L</td><td colspan="2">H ; Read
L ; Write</td></tr>
<tr><td>6</td><td>E</td><td>H</td><td colspan="2">Enable Signal [sinyal penanda mulai operasi, aktif saat operasi write atau read]</td></tr>
<tr><td>7</td><td>DB0</td><td>H/L</td><td colspan="2" rowspan="4">4 bit bus data lower 2 arah, dapat dibaca atau ditulis terhadap mikrokontroler</td></tr>
<tr><td>8</td><td>DB1</td><td>H/L</td></tr>
<tr><td>9</td><td>DB2</td><td>H/L</td></tr>
<tr><td>10</td><td>DB3</td><td>H/L</td></tr>
</table>

11	DB4	H/L	4 bit bus data upper 2 arah, dapat dibaca atau ditulis terhadap mikrokontroler, DB7 juga sebagai busy flag	
12	DB5	H/L		
13	DB6	H/L		
14	DB7	H/L		
15	V+BL	-	Back Light Supply	4 – 4,2 V 50 – 200 mA
16	V-BL	-		0 Volt (GND)

2.5.1. Instruksi Operasi

Tabel 2.8. Instruksi Pada LCD [5]

Instruksi	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0
Display Clear	0	0	0	0	0	0	0	0	0	1
Cursor Home	0	0	0	0	0	0	0	0	1	*
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Cursor Display Shift	0	0	0	0	0	1	S/C	R/L	*	*
Function Set	0	0	0	0	1	DL	1	*	*	*
CG RAM Address Set	0	0	0	1	A_{CG}					
DD RAM Address Set	0	0	1	A_{DD}						
BF/Address Read	0	1	BF	AC						
Data Write to CG RAM	1	0	Write Data							
Data Read from CG RAM	1	1	Read Data							

* Invalid Bit

A_{CG} ; CG RAM Address

A_{DD} ; DD RAM Address

2.5.2. Operasi Dasar

- Register

Kontrol dari LCD memiliki 2 buah register 8 bit yaitu register instruksi (IR) dan register data (DR). IR memiliki instruksi seperti display, clear, cursor shift dan display data (DD RAM) serta karakter (CG RAM). DR menyimpan data untuk ditulis ke DD RAM ataupun membaca data dari DD RAM dan CG RAM. Ketika data ditulis ke DD RAM atau CG RAM maka DR secara otomatis menulis data ke DD RAM atau CG RAM. Ketika data pada CG RAM atau DD RAM akan dibaca maka alamat data ditulis pada IR. Sedangkan data akan dimasukkan melalui DR sehingga dapat dibaca oleh mikrokontroler.

Tabel 2.9. Pemilihan Register Pada LCD [5]

RS	RW	Operasi
0	0	Seleksi IR, IR Write Display Clear
0	1	Busy Flag (DB7), @ Counter (DB0-DB7), Read
1	0	Seleksi DR, DR Write
1	1	Seleksi DR, DR Read

- Busy Flag

Busy Flag menunjukkan bahwa modul siap untuk menerima instruksi selanjutnya sebagaimana terlihat pada tabel diatas. Register seleksi sinyal akan melalui DB7 jika RS=0 dan R/W=1. Jika bernilai 1 maka sedang melakukan kerja internal dan instruksi tidak akan dapat diterima, oleh karena itu status dari flag harus diperiksa sebelum melaksanakan instruksi selanjutnya.

- Address Counter (AC)

  AC menunjukkan lokasi memori dalam modul LCD. Pemilihan lokasi alamat lewat Ac diberikan lewat register instruksi (IR) ketika data pada A, maka AC secara otomatis menaikkan atau menurunkan alamat tergantung dari Entry Mode Set.

- Display Data RAM

  Pada LCD, masing-masing line memiliki range alamat tersendiri. Alamat itu diekspresikan dengan bilangan hexadesimal. Untuk line 1 range alamat berkisar antara 40_H - $4F_H$.

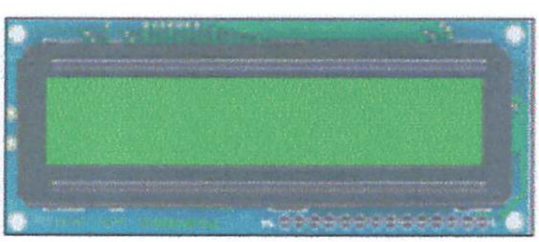
- Character Generator ROM (CG ROM)

  CG ROM memiliki tipe dot matrik 5x7, dimana pada LCD telah tersedian ROM sebagai pembangkit karakter dalam kode ASCII.

- Character Generator RAM (CG RAM)

  CG RAM dipakai untuk pembuatan karakter tersendiri melalui program.

Adapun bentuk fsik dari LCD M1632 adalah pada gambar berikut :



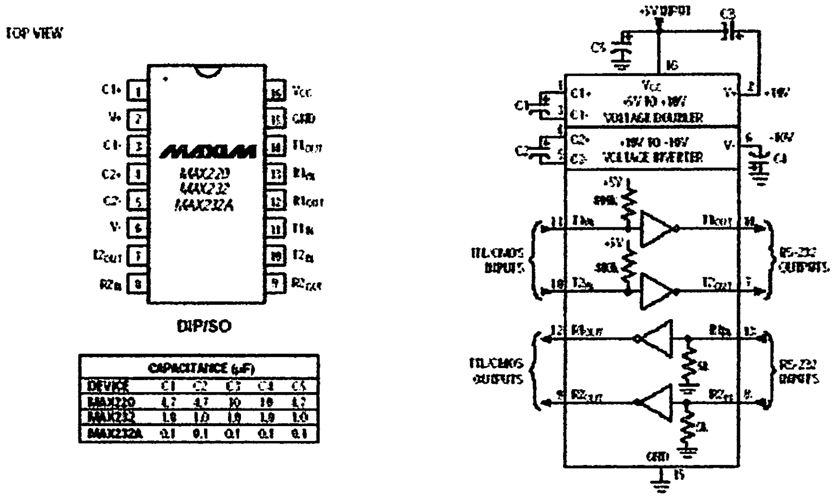
Gambar 2.12. Liquid Crystal Display [5]

2.6. RS-MAX 232

RS 232 merupakan salah satu jenis antar muka (interface) dalam proses transfer data antar komputer dalam bentuk serial transfer. RS 232 merupakan singkatan dari Recommbeded Standard number 232. Alat ini dibuat oleh Elektronik Industri Assosiation, untuk interface antara peralatan terminal data dan peralatan komunikasi data, dengan menggunakan data biner serial sebagai data yang ditransmisikan. IC RS MAX 232 ini mempunyai empat buah bagian konverter yaitu dua buah driver receiver dan dua buah driver transmitter.

Saluran data pada port seri PC menggunakan standard RS 232, dimana logic 0 (low) dinyatakan sebagai tegangan antara +3 volt sampai +10 volt, dan logic 1 (high) dinyatakan sebagai tegangan antara -3 volt sampai -10 volt. Level tegangan ini tidak sesuai dengan level tegangan yang dipakai pada port seri AT89S51 yang menggunakan standard TTL (Tranastor Transistor Logic), yaitu level tegangan baku dalam rangkaian-rangkaian digital.

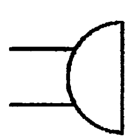
Dalam standard TTL, logic 0 (low) dinyatakan sbagai tegangan antara 0 volt sampai 0,8 volt, dan logic 1 (high) dinyatakan sebagai tegangan antara 3,5 volt sampai 5 volt. Untuk dua MCU yang dihubungkan secara serial pada jarak tertentu maka dibutuhkan IC MAX 232 karena level tegangan TTL terlalu kecil untuk ditrasfer.



Gambar 2.13. IC RS Max 232 [6]

2.7. Buzzer

Perangkat *Buzzer* digunakan untuk menghasilkan bunyi, merupakan komponen resonator *piezoelectric* yang digunakan untuk mengadakan isyarat terdengar sebagai indikator. Buzzer akan aktif dengan cara mengeluarkan sinyal suara (berbunyi) dengan lama waktu sesuai dengan perencanaan nanti.



Gambar 2.14. Simbol Buzzer

2.8. Limit Switch

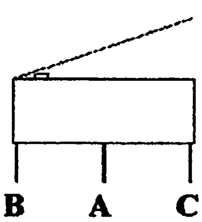
Limit swicth merupakan sebuah saklar yang bekerja karena ada suatu sentuhan. Limit swicth mempunyai beberapa bagian antara pengungkit dan roda penjulang yang merupakan bagian mekanik yang berfungsi sebagai pengunci letak palang pintu yang dikendalikan oleh motor stepper. Jika limit switch tersentuh oleh palang pintu maka diposisi palang pintu itulah dianggap 0^{O}.

Ada beberapa tipe limit switch yaitu limit swicth yang merupakan kontak NC (Normally Closed) dan NO (Normally Open). Limit yang merupakan kontak NO berfungsi sebagai penghubung sedangkan yang kontak NC berfungsi sebagai pemutus. Adapun simbol dari limit swicth atau push botton dari yang NO dan NC adalah sbb:

Pushbutton make NO (normally Open) **Pushbutton break NC (normally Closed)**

Gambar 2.15. Simbol Limit Switch

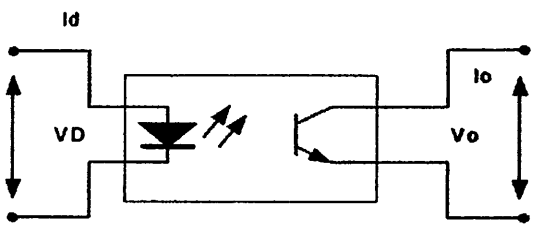
Sedangkan jenis limit switch yang digunakan dalam peralatan ini adalah yang terllihat seperti di bawah ini :



Gambar 2.16. Bentuk Fisik dari limit switch

2.9. Optocoupler

Optocoupler disebut juga optoisolator atau isolator yang tergandeng optic, menggabungkan LED dan fototransistor dalam satu kemasan. Gambar 2.27. menunjukkan salah satu contoh dari optocoupler. Komponen ini memiliki LED pada sisi masukan dan fototransistor pada sisi keluaran. Keuntungan utama optocoupler adalah pemisah secara listrik antara rangkaian masuk dengan rangkaian keluarnya. Dengan optocoupler, hubungan yang ada antara masukan dan keluaran hanya seberkas cahaya. Karena hal ini dapat memperoleh resistansi penyekatan diantara dua rangkaian tersebut. Optocoupler yang dipakai adalah yang terdiri dari satu LED dan satu transistor foto seperti terlihat dalam gambar berikut ini:



Gambar 2.17. Simbol Optocoupler [7]

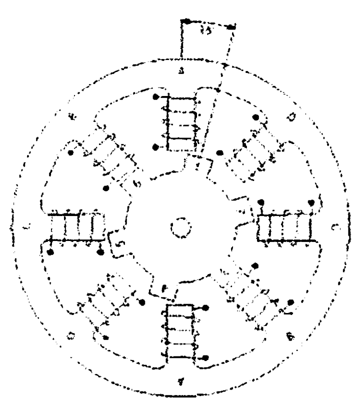
2.10. Motor Stepper

Motor *stepper* adalah perangkat elektromekanis yang bekerja dengan mengubah pulsa elektronis menjadi gerakan mekanis diskrit. Motor stepper bergerak berdasarkan urutan pulsa yang diberikan kepada motor. Karena itu, untuk menggerakkan motor stepper diperlukan pengendali motor stepper yang membangkitkan pulsa-pulsa periodik. Penggunaan motor stepper memiliki beberapa keunggulan dibandingkan dengan penggunaan motor DC biasa. Keunggulannya antara lain adalah :

- Sudut rotasi motor proporsional dengan pulsa masukan sehingga lebih mudah diatur.
- Motor dapat langsung memberikan torsi penuh pada saat mulai bergerak

- Posisi dan pergerakan dapat ditentukan secara presisi
- Memiliki respon yang sangat baik terhadap mulai, stop dan berbalik (perputaran).
- Frekuensi perputaran dapat ditentukan secara bebas dan mudah pada range yang luas.

Alat ini menggunakan motor stepper dengan type Variable Reluctance (VR). Motor stepper jenis ini telah lama ada dan merupakan jenis motor yang secara struktural paling mudah untuk dipahami. Motor ini terdiri atas sebuah rotor besi lunak dengan beberapa gerigi dan sebuah lilitan stator. Ketika lilitan stator diberi energi dengan arus DC, kutub-kutubnya menjadi termagnetasi. Perputaran terjadi ketika gigi-gigi rotor tertarik oleh kutub-kutub stator. Berikut ini adalah penampang melintang dari motor stepper tipe *variable reluctance* (VR):



Gambar 2.18. Penampang melintang motor stepper tipe VR [8]

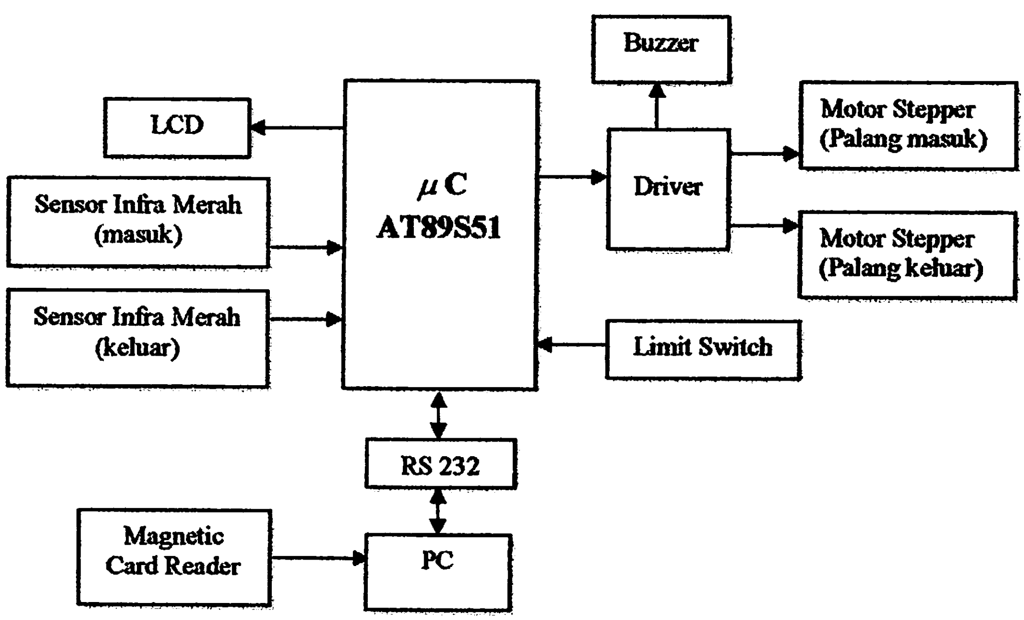
BAB III

PERANCANGAN DAN PEMBUATAN SISTEM PARKIR PRABAYAR MENGGUNAKAN KARTU MAGNETIK (KARTU MAHASISWA) BERBASIS MIKROKONTROLLER AT89S51

Perencanaan dan pembuatan dibedakan atas dua aspek yaitu aspek perangkat keras (*hardware*) dan perangkat lunak (*software*).

3.1. Perangkat Keras (*Hardware*)

Miniatur siatem dan blok diagram dari perancangan perangkat keras secara keseluruhan adalah sebagai berikut :



Gambar 3.1. Blok Diagram Alat
Sumber : Perancangan

Adapun penjelasan dari tiap-tiap blok akan dibahas berikut ini :

1. Mikrokontroller AT89S51

   Melakukan komunikasi data dengan PC melalui RS 232, pengatur driver motor stepper, penerima sinyal dari sensor infrared, mengaktifkan alarm, menampilkan karakter pada LCD.

2. Magnetic Card Reader

   Sebagai pembaca kode yang ada pada kartu magnetik.

3. Driver (Optocoupler)

   Sebagai penggerak mekanik yang dalam hal ini adalah palang pintu masuk dan keluar yang digerakkan oleh motor stepper dan juga sebagai pengendali buzzer.

4. Sensor Infra Merah

   Untuk mendeteksi kendaraan yang telah melewati palang pintu sehingga mikrokontroller aman untuk memerintahkan motor stepper untuk menutup palang pintu.

5. Buzzer

   Digunakan sebagai alarm, petunjuk bahwa ada hal yang tidak sesuai dengan yang seharusnya terjadi yaitu alarm akan berbunyi jika sensor kartu mendeteksi adanya suatu kartu yang masuk tetapi tidak menemukan kodenya didalam database, selain itu alarm akan berbunyi jika saldo lebih kecil dari bea parkir.

6. LCD

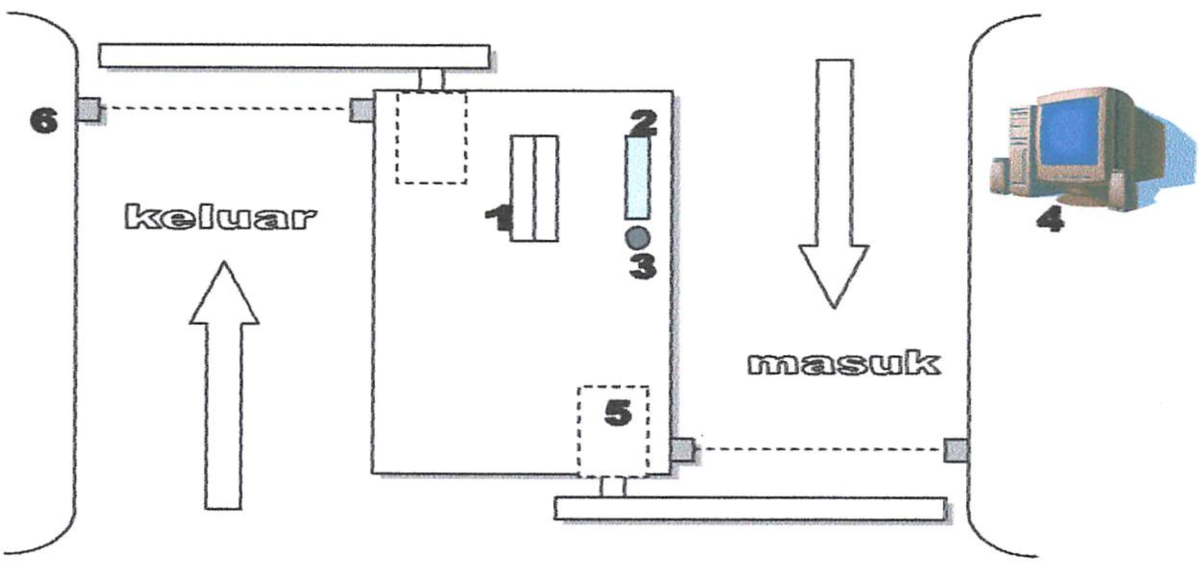
Berfungsi untuk menampilkan karakter yang berupa angka, huruf atau simbol sesuai dengan yang kita inginkan.

7. RS 232

Sebagai sarana komunikasi data antara masukan dari pembacaan rangkaian sensor dengan software yang di PC.

8. Limit Switch

Sebagai penentu posisi awal (0^{O}) palang pintu masuk dan palang pintu keluar.



Gambar 3.2. Perencanaan dan Pembuatan Sistem Parkir

Keterangan

1. Magnetic Racd Reader.
2. LCD
3. Buzzer
4. PC (Personal Computer)
5. Motor Stepper
6. Sensor Infra Merah

3.2. Prinsip Kerja Alat

Kartu magnetik yang dipakai pada dasarnya mempunyai data di dalam pitanya. Saat kartu digesekkan pada magnetic card reader, data biner pada kartu dibaca oleh head magnetik dan diubah kedalam bentuk digital berkarakter ASCII yang selanjutnya data tersebut dikirim ke mikrokontroller dalam hal ini AT89S51 kemudian dikirm ke PC melalui RS-232. Jika data sesuai maka LCD menampilkan sisa saldo yang tersisa, jika saldo mencukupi maka palang pintu membuka dan sensor mendeteksi kendaraan yang lewat yang dipergunakan untuk menutup palang pintu. Jika data pada kartu tidak sesuai maka buzzer aktif dan LCD menampilkan karakter yang sudah diprogram, contohnya "belum terdaftar".

3.3. Mikrokontroller AT89S51

Mikrokontroller AT89S51 adalah sebuah chip IC yang terdiri dari 40 pin. Dalam perencanaan sistem ini fungsi dari pin-pin yang digunakan adalah sebagai berikut:

1. Pin 32 – 39 (P0.0 – P0.7)

   merupakan port 0 yang digunakan sebagai outputan data dari mikrokontroller ke LCD M1632.

2. Pin 1 dan 2 (P1.0 dan P1.1)

   merupakan port yang digunakan sebagai inputan tegangan yang berasal dari sensor infra merah.

3. Pin 9 berfungsi sebagai Reset.

4. Pin 21 – 28 (P2.0 – 2.7)

   merupakan Port 2 dari mikrokontroller dihubungkan dengan optocoupler sebagai driver untuk mengendalikan motor stepper.

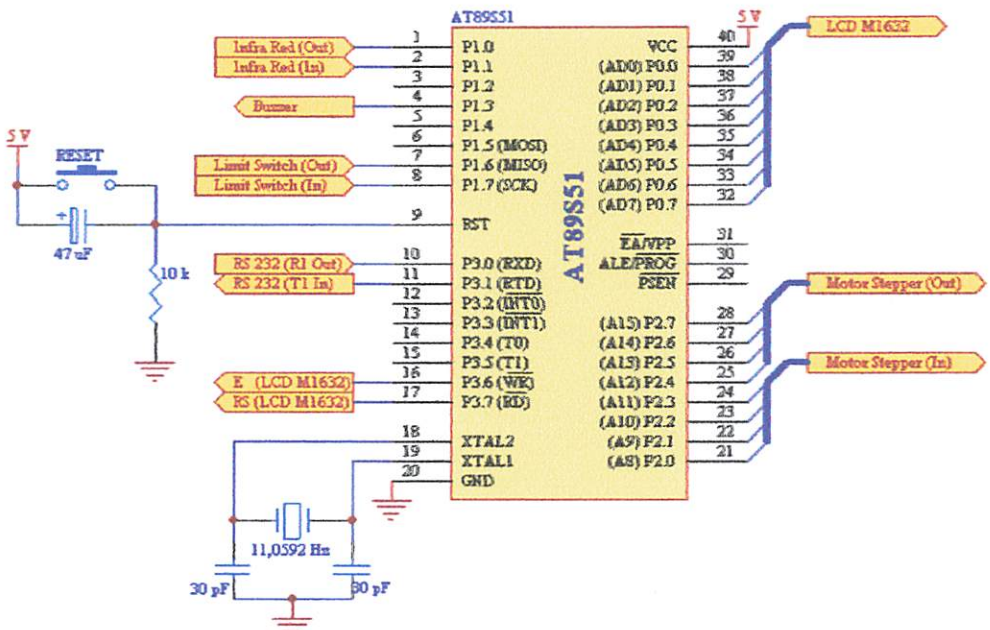
5. Pin 10 dan 11 (P3.0 – P3.1)

   merupakan Port 3 dari mikrokontroller yang digunakan sebagai kontrol yang terdiri dari:

   - Pin 10 (P3.0), RXD berfungsi sebagai receive data untuk serial port yang dihubungkan dengan data serial komputer.
   - Pin 11 (P3.1), TXD berfungsi sebagai transmit data untuk serial port yang berasal dari data serial komputer.

6. Pin 18 (XTAL 2) sebagai pembangkit ossilator (clock) XTAL 2.
7. Pin 19 (XTAL 1) sebagai pembangkit ossilator (clock) XTAL 1.
8. Pin 31 (EA/VPP) berfungsi sebagai VCC +5 Volt.
9. Pin 20 (GND) berfungsi sebagai ground.

Mikrokontroller pada alat ini tidak dapat bekerja sendiri sehingga masih membutuhkan komponen-komponen pendukung lain, komponen-komponen tersebut saling berhubungan secara hardware dan juga sofware. Rangkaian mikrokontroller dapat dilihat pada gambar dibawah ini:



Gambar 3.3. Rangkaian Mikrokontroller AT89S51

Sumber : Perancangan

3.3.1. Minimum Sistem AT89S51

Mikrokontroller AT89S51 dirancang untuk dapat berdiri sendiri, karena sudah terdapat 4 Kbytes PEROM (*Programmable and Erasable Read Only Memory*), 128 x 8 bit RAM internal, 32 *Programmable I/O*, dan terdapat dua 16 bit timer/counters. Dari fasilitas-fasilitas tersebut dapat memfungsikan mikrokontroller AT89S51 untuk bekerja dalam *single chip*, maksudnya dengan sebuah mikrokontroller saja sudah dapat mengontrol kerja dari keseluruhan sistem. Dalam perancangan ini digunakan 1 buah mikrokontroller AT89S51.

- ALE/PROG

Pulsa ALE (*Address Latch Enable*) hanya digunakan untuk akses ke eksternal memori. Sedangkan PROG digunakan untuk input program selama program *Flash* memori. Karena dalam perancangan ini menggunakan *single chip* maka untu pin ALE/PROG tidak digunakan.

- PSEN

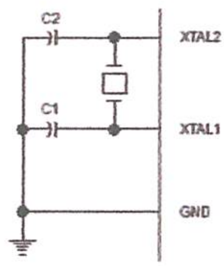
  Program Strobe Enable berfungsi membaca *strobe* ke program memori eksternal. Untuk itu PSEN tidak digunakan.

- EA/Vpp

  Logika yang diberikan pada pin ini menunjukkan kerja dari mikrokontroller AT89S51. Pin ini dihubungkan ke VCC karena pada perancangan ini tidak memanfaatkan memori eksternal.

- XTAL1 dan XTAL2

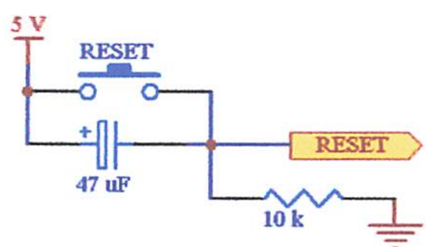
  Kecepatan proses yang dilakukan oleh mikrokontroller ditentukan oleh sumber *clock* (pewaktu) yang dikendalikan oleh mikrokontroller tersebut. Untuk mendapatkan *clock* pada mikrokontroller, maka digunakan pin XTAL1 dan XTAL2 yang dihubungkan dengan sebuah kristal yang sudah terancang dan tersedia di dalam chip AT89S51. Besar XTAL yang digunakan adalah 11,059 MHz dan kapasitansinya sebesar 30 pf yang sesuai dengan spesifikasi pada data sheet AT89S51 dengan frekuensi pewaktu berdasarkan kebutuhan dan kecepatan waktu pelaksanaan intruksi. Untuk menjaga kestabilan *clock*, maka ditambah 2 buah kapasitor seperti pada gambar dibawah ini:



Gambar 3.4. Rangkaian Clock [1]

- *Reset*

  Untuk melakukan reset sistem pada mikrokontroller yaitu untuk mengawali eksekusi program pada alamat paling rendah yang dapat dimanfaatkan pin reset yang ada pada mikrokontroller. Pin 9 dihubungkan dengan rangkaian *reset* rangkaian ini diharapkan agar dapat mempunyai kemampuan *power ON Reset*, yaitu Reset terjadi saat *power* diaktifkan. Dibawah ini adalah adalah rangkaian reset :



Gambar 3.5. Rangkaian Reset [1]

3.4. Pembaca Kartu Magnetik

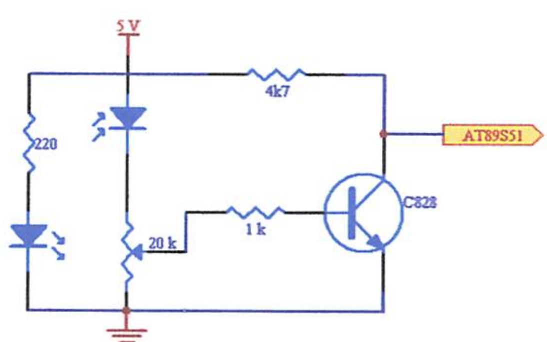
Keluaran dari pembaca kartu magnetik ini merupakan hasil olahan dari decoder F/2F yang terpasang pada pembaca kartu magnetik. Pada pembaca kartu magnetik ini sebenarnya terdapat 3 jenis keluaran yaitu track 1, 2 dan 3. tetapi dalam tugas akhir ini hanya digunakan track 3 sebagai acuan, sehingga hanya

hanya 3 jalur dari ke 9 jalur keluaran pembaca kartu magnetik yang digunakan. Sinyal keluaran dari magnetik card reader ini kompatibel dengan TTL sehingga dapat langsung dihubungkan ke port Mikrokontroller yaitu serial sinkron. Data yang dikeluarkan berupa kode ASCII.

Gambar 3.6. Pembaca kartu magnetik [2]

3.5. Rangkaian Sensor Infra Merah.

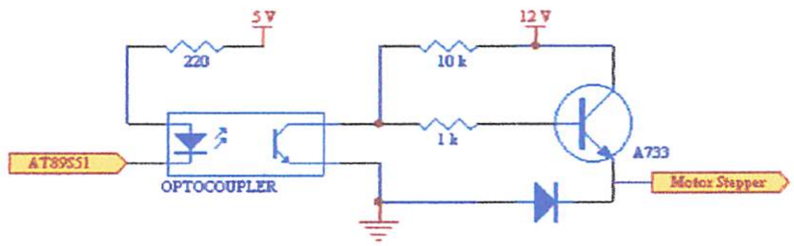
Dalam perancangan ini infra merah sebagai pemancar dan photodioda sebagai penerima cahaya dari infra merah yang diteruskan ke mikrokontroller.



Gambar 3.7. Rangkaian Sensor
Sumber : *Perancangan*

3.6. Perancangan Driver (Optocoupler).

Rangkaian ini berfungsi untuk menyambung dan memutuskan rangkaian yang ada diluar mikrokontroller, yaitu motor penggerak palang pintu masuk dan motor penggerak palang pintu keluar. Rangkaian driver ini digambarkan sebagai berikut :



Gambar 3.8. Driver Motor Stepper

Sumber : *Perancangan*

Motor stepper yang digunakan disini adalah motor stepper jenis Unipolar. Motor disini digunakan untuk proses membuka dan menutupnya palang pintu masuk dan palang pintu keluar. Pada proses pengaturan motor *stepper* dikontrol oleh software yang pengkalibrasiannya secara manual.

Mengacu pada gambar rangkaian diatas agar led dapat memancarkan sinar maka diperlukan arus (I_L) sebesar 20 mA dan tegangan (V_L) sebesar 1,5 volt sehingga dapat dihitung nilai R_L sebagai berikut :

$$R_L = \frac{Vcc - V_L}{I_L} \rightarrow R = \frac{5 - 1{,}5}{20.10^{-3}} = 175\ \Omega$$

Dalam hal ini resistor yang digunakan adalah 175 Ω. Karena resistor 175 Ω yang tidak ada dipasaran, maka digunakan resistor 330 Ω. Sedangkan untuk resistor pada phototransistor adalah sebagai berikut :

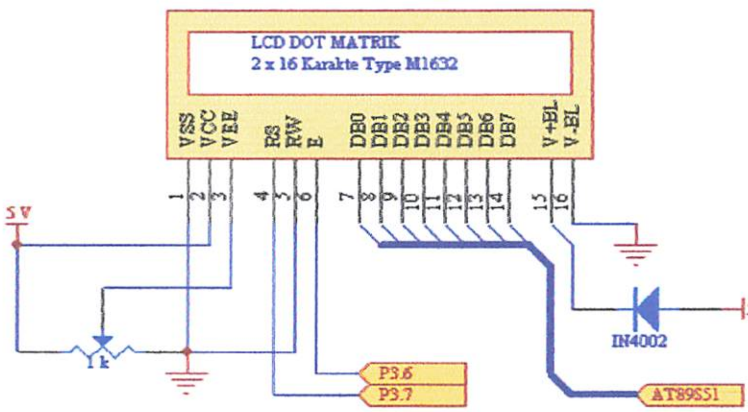
Diketahui arus yang diperlukan pada phototransistor (I_P) sebesar 0,5 mA, maka

$$R_L = \frac{Vcc}{I_P} \rightarrow R = \frac{5}{0{,}5.10^{-3}} = 10\ k\Omega$$

Pada kondisi dari sumber cahaya terhalang maka phototransistor akan cut off, sehingga tidak ada arus kolektor yang mengalir yang mengakibatkan tegangan output $V_0 = V_{CC}$, sebaliknya jika ada cahaya yang mengenai permukaan phototransistor maka phototransistor akan aktif sehingga ada arus yang mengalir dari tegangan $V_0 = V_{CE\ (Sat)} = 0{,}2$ Volt.

3.7. Perancangan Rangkaian LCD

Berdasarkan User Manual LCD M1632 buatan Seiko Instruments Inc, pin 1 (Vss) terhubung dengan 0V (Ground), pin 2 (Vcc) dengan +5V ±10%, pin 3 (Vee) dengan tegangan variabel antara 0V sampai 5V untuk mengatur kontras tulisan, pin 4 (RS) untuk memilih antara register data atau register instruksi (0=instruksi, 1=data), pin 5 (R/W) untuk proses tulis atau baca (0=tulis, 1=baca), pin 6 (E) untuk memberikan sinyal enable, pin 7 sampai dengan pin 14 (DB0 ..DB7) untuk bus data LCD, pin 15(V+BL) dan pin 16(V-BL) untuk power supply LED penghasil cahaya belakang (back light) yang besarnya antara +4V sampai dengan +4,2V. Terlihat bahwa hanya terdapat dua buah pin yang harus level logikanya diubah-ubah yaitu pin 4 (RS) dan pin 6 (E). Selain itu juga pin-pin bus data harus menerima data karakter atau instruksi. Karena itu, rangkaian modul LCD M1632 dapat dirancang seperti Gambar 3.11. Pada pin 15 dipasang dioda dalam umpan maju dengan tujuan untuk menurunkan tegangan catu +5V menjadi +4,2V (drop tegangan dioda sebesar +0,8V), sehingga mememuhi persyaratan yang ditentukan.



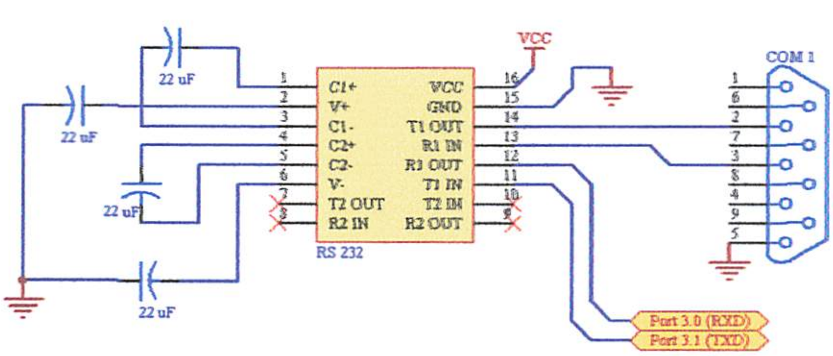
Gambar 3.9. Modul LCD M1632

Sumber: Perancangan

3.8. Perencanaan dan Pembuatan Perangkat Lunak

Untuk mendukung agar perangkat keras berfungsi sesuai dengan perencanaan, maka diperlukan perangkat lunak sebagai penunjangnya. Untuk mengatur dan mengendalikan keseluruhan sistem perangkat keras yang telah dibuat, harus dibantu dengan perangkat lunak.

Sistem aplikasi Mikrokontroller AT89S51 yang terintegrasi dengan PC ini dapat mengatur dan mengendalikan keseluruhan sistem apabila ada urutan instruksi yang mendefinisikan secara jelas urutan tugas yang harus dikerjakan.



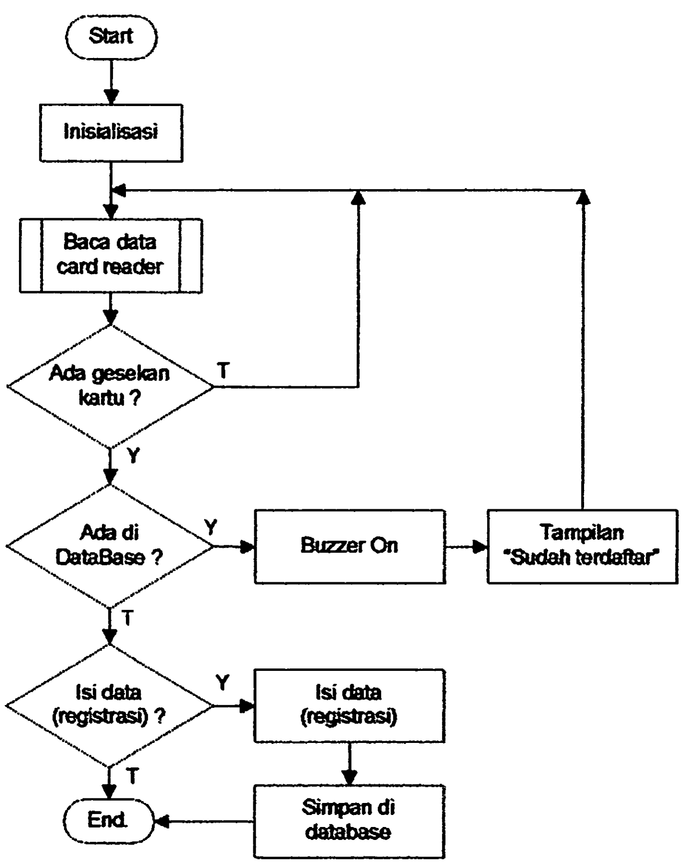
Gambar 3.10. Rangkaian RS 232

Sumber : *Perancangan*

Urutan instruksi ini sangat penting untuk didefinisikan, karena mikrokontroller bekerja secara pasti berdasarkan urutan instruksi ini, susunan logika perancangan yang salah tidak dapat diketahui oleh mikrokontroller. Selama instruksi yang diterima sesuai dengan aturannya, mikrokontroller tetap mengerjakan instruksi tersebut.

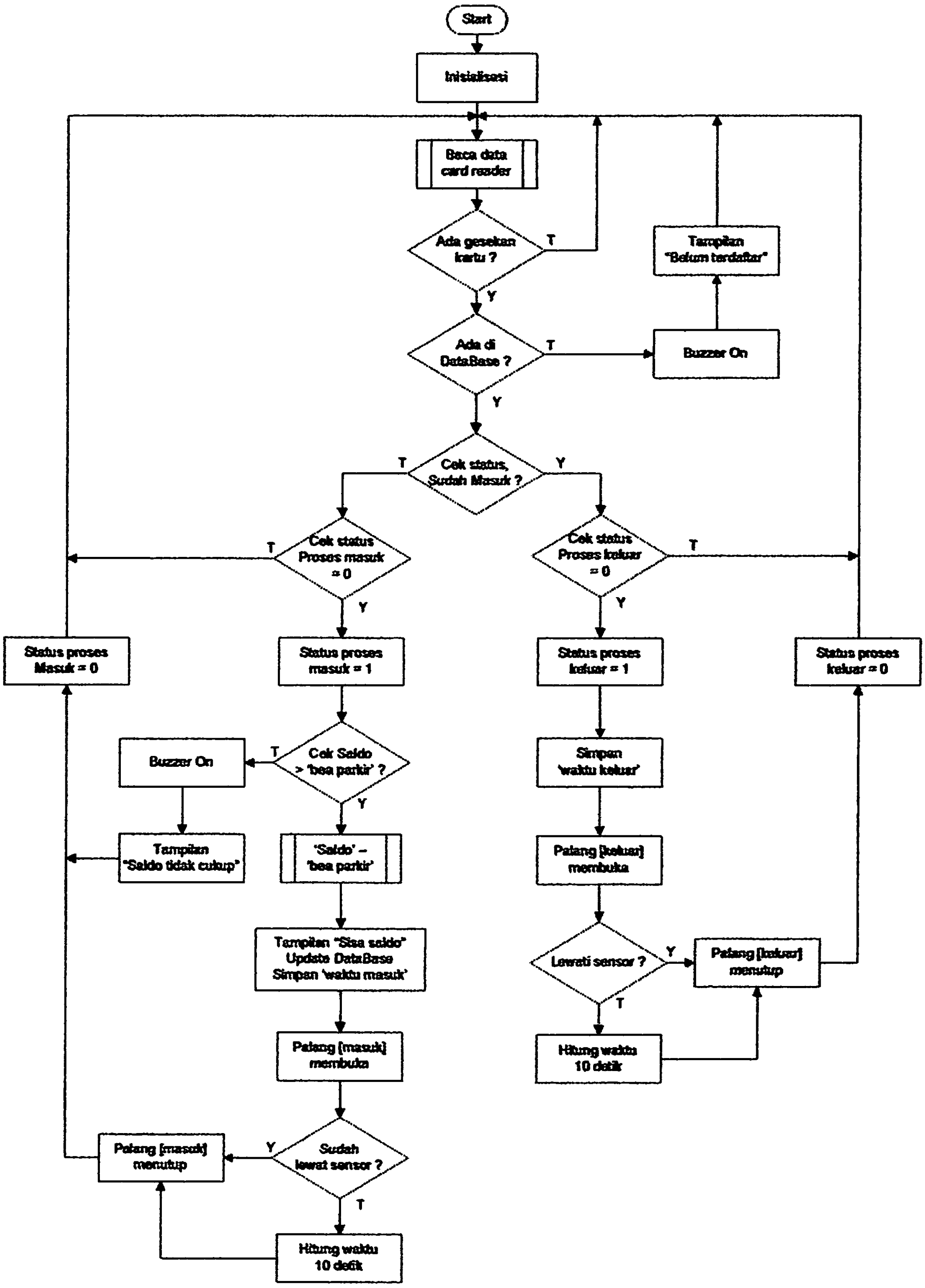
Kesalahan seperti ini baru diketahui ketika kerja sistem aplikasi tidak sesuai dengan spesifikasi awal. Sebuah mikrokontroller tidak akan bekerja bila tidak diberikan program kepadanya. Program tersebut memberitahukan apa yang harus dilakukan oleh mikrokontroller.

3.8.1. Flowchart Registrasi



Gambar 3.11. Flowchart registasi

3.8.2. Flowchart Sistem Parkir



Gambar 3.12. Flowchart sistem parkir

Prinsip Kerja Sistem

- Kartu digesekkan pada magnetic card reader.
- Jika belum ada dalam database maka buzzer akan aktif (on) dan LCD akan menampilkan karakter yang sudah diprogram (contohnya: "Belum Teradftar").
- Jika sudah ada dalam database maka software akan memeriksa status laporan.
    - Jika status kartu sama dengan 0 artinya kendaraan belum masuk (out), maka software akan memeriksa saldo, apakah lebih besar dari bea parkir.
        - Jika saldo lebih besar dari bea parkir, maka software akan mengurangi saldo dengan bea parkir, LCD menampilkan sisa saldo, secara otomatis software meng-update saldonya saja pada database dan mencatat dilaporan yaitu nama, NIM, nomor plat, sisa saldo dan waktu masuk. Palang pintu (masuk) membuka.
            - Sensor (masuk) mendeteksi kendaraan yang lewat, jika kendaraan sudah melewati sensor maka palang (masuk) menutup, jika belum lewat dalam waktu 10 detik maka palang (masuk) tetap akan menutup yang disertai dengan alarm.
        - Jika saldo < bea parkir, maka buzzer akan aktif (on) dan LCD akan menampilkan karakter pada LCD (contohnya: "Saldo tidak cukup"). Palang pintu tidak akan membuka.
    - Jika status kartu sama dengan 1 artinya kendaraan sudah masuk (in), maka software akan mencatat (waktu keluar) dilaporan. Palang pintu (keluar) membuka.

- Sensor mendeteksi kendaraan yang lewat, jika kendaraan sudah melewati sensor maka palang (keluar) menutup, jika belum lewat dalam waktu 10 detik maka palang (keluar) tetap akan menutup disertai dengan alarm. Sama dengan seperti pada saat melalui pintu masuk.

BAB IV

PENGUJIAN ALAT

Dalam bab ini membahas pengujian perangkat keras dan perangkat lunak sistem yang dirancang apakan telah berfungsi sebagaimana spesifikasi yang diinginkan.

4.1. Pengujian Modul Magnetic Card Reader

Dari penggesekan kartu magnetik yang dilakukan, dapat diketahui data yang berada didalamnya, berikut ini adalah data setelah kartu magnetik digesekkan.

60100425304296=0712101

Data yang diperoleh dari kartu magnetik ini adalah data biner dengan output dalam bentuk karakter ASCII sehingga dapat langsung dipakai sebagai data input oleh PC.

4.2. Pengujian Rangkaian Infra Red dan Photodioda

Tujuan dari pengujian ini adalah untuk mengetahui respon rangkaian apabila terlewati halangan atau tidak pada saat palang membuka. Peralatan yang digunakan adalah :

- Multimeter
- Rangkaian Infra Red dan Photodioda
- Logic probe
- Power suply

Langkah pengujian : dengan menggunakan logic probe, maka didapatkan output dari photodioda, apabila pada saat mikrokontroller membuka palang pintu, pancaran dari led infra red langsung mengenai photodioda dan apabila ada kendaraan yang lewat maka pancaran dari led infra red akan terhalangi, saat itu juga palang pintu akan menutup.

Tabel 4.1. Hasil Perbandingan Antara Pengukuran dan Perhitungan Rangkaian Sensor Infra Merah

Benda Penghalang	**Vout**		**Error (%)**
	Perhitungan	**Pengukuran**	
Tangan	5	4,84	3,20
Kertas	5	4,64	7,20
Besi	5	4,85	3,00
Kain	5	4,18	16,40
Plastik tipis	5	4,53	9,40
Plastik tebal	5	4,84	3,20
Plastik transparan	5	2,45	51,00
Kresek plastik	5	3,42	31,60
Karet	5	4,85	3,00
Kayu	5	4,85	3,00

Kondisi tegangan pada sensor photodioda ketika terhalang :

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

$$\text{Error rata-rata} = \frac{JumlahError}{BanyaknyaPercobaan} \times 100\,\%$$

$$= \frac{131}{10} \times 100\,\% = 13{,}1\,\%$$

4.3. Pengujian Motor Stepper

Penggerak palang pintu ini terdiri dari 2 buah motor stepper yang membutuhkan tegangan 12 volt yang dikendalikan oleh rangkaian Optocoupler tipe 4N25 sebanyak 4 buah untuk satu motor stepper dimana datanya diperoleh dari Mikrokontroller. Satu step putarannya membentuk sudut 15^{O}.

Tabel 4.3. Hasil pengujian motor stepper

INPUT								OUTPUT (Palang Motor Stepper)
Data (Logika)				Tegangan (Volt)				
A	**B**	**C**	**D**	**A**	**B**	**C**	**D**	
1	1	1	0	0	0	0	11,89	berputar ke kanan sebesar 15^{O}
1	1	0	1	0	0	11,89	0	
1	0	1	1	0	11,89	0	0	
0	1	1	1	11,89	0	0	0	
0	1	1	1	11,89	0	0	0	Berputar ke kiri sebesar $15^{O} \times 4 = 60^{O}$
1	0	1	1	0	11,89	0	0	
1	1	0	1	0	0	11,89	0	
1	1	1	0	0	0	0	11,89	

Kondisi tegangan pada output driver motor stepper :

$$\text{Error } V_{motor} = \frac{Vmotor(perhitungan) - Vmotor(pengukuran)}{Vmotor(perhitungan)} \times 100\ \%$$

$$\text{Error } V_{motor} = \frac{12 - 11,89}{12} \times 100\ \% \quad = 0,92\ \%$$

4.4. Pengujian Sistem Keseluruhan

Pada pengujian sistem ini, peralatan yang digunakan adalah sebagai berikut :

- Sistem lengkap beserta PC dengan sistem operasi Windows XP.
- Penghalang sensor infra merah.
- Kartu magnetik beserta magnetic card reader terkoneksi dengan PC.
- Miniatur sistem parkir terkoneksi dengan PC .

4.4.1. Tampilan Utama

- Operator menghidupkan power pada miniatur sistem parkir.
- Operator menghidupkan PC.
- Operator membuka program "Parkir Prabayar".

Saat pertama kali menjalankan program "Parkir Prabayar", akan muncul tampilan seperti berikut :

- Klik *Edit* ➔ *Tarip Parkir*, isi form tersebut dengan angka 400, klik OK.

Gambar 4.1. Form tampilan utama dan form tarip parkir.

4.4.1.1.Registrasi (Entry Data)

Langkah-langkah pengujian form registrasi :

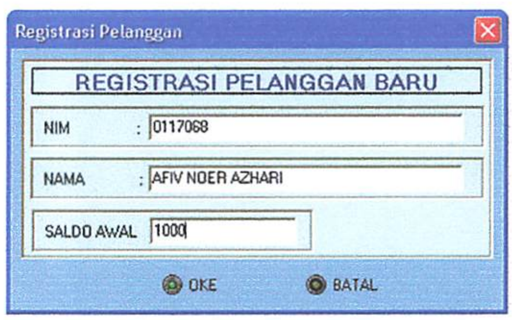
- Gesekkan kartu magnetik pada magnetic card reader dengan mengklik *Registrasi Baru*, sehingga muncul tampilan form dibawah ini :

Gambar 4.2. Form registrasi (data dari kartu magnetik).

- Jika telah terdaftar akan muncul tampilan berikut:

Gambar 4.3. Form tampilan 'telah terdaftar'

- Setelah digesekkan, secara otomatis akan tampil form berikut:



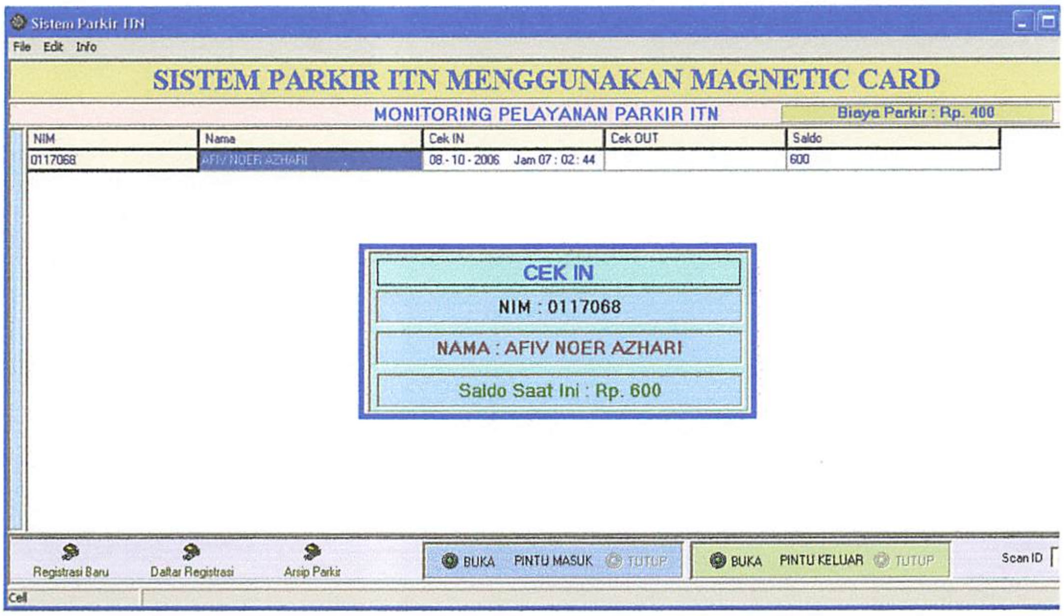
Gambar 4.4. Form registrasi pelanggan

- Isi data dan klik OK, maka data akan tersimpan di database.

4.4.1.2.Masuk Parkir

Langkah-langkah pengujian memasuki lahan parkir :

- Gesekkan kartu magnetik ke magnetic card reader.
- Pada saat kendaraan dengan kartu magnetik yang datanya tersimpan dalam database, statusnya belum masuk dan saldonya masih mencukupi, maka akan muncul tampilan sebagai berikut :

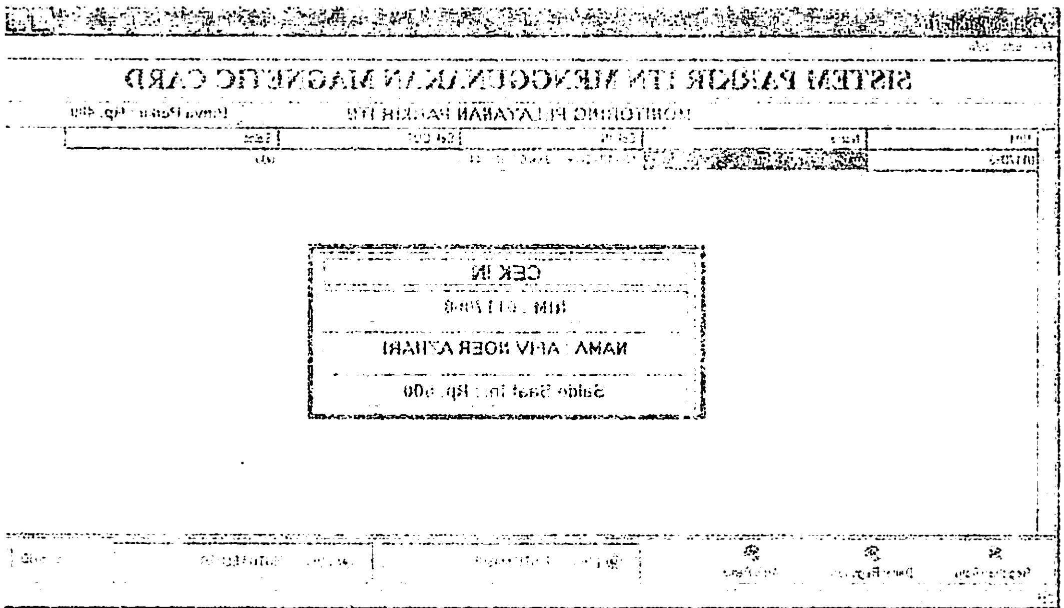


Gambar 4.5. Form sistem parkir saat pelanggan masuk parkir.

- Saldo dikurangi Rp. 400,- sisa saldo ditampilkan pada monitor dan LCD.
- Palang pintu masuk membuka sebesar 90^{O}.
- Gerakkan penghalang sebagai kendaraan yang melewati sensor palang pintu masuk sehingga palang pintuk masuk dapat menutup secara otomatis.

4.4.1.3. Keluar Parkir

- Gesekkan kartu magnetik ke magnetic card reader.
- Pada saat kendaraan dengan kartu magnetik yang datanya tersimpan dalam database dan statusnya “1” (in/sudah masuk), maka palang pintu keluar membuka.


CEK IN

Gambar 4.6. Form sistem parkir saat pelanggan keluar parkir.

- Gerakkan penghalang sebagai kendaraan yang melewati sensor palang keluar sehingga secara otomatis palang pintuk keluar menutup.

4.4.1.4.Pengisian Saldo

- Jika saldo tidak cukup untuk masuk parkir maka pada LCD akan tampil “Maaf ! Saldo anda tidak cukup”.
- Klik *Edit* ➔ *Refil Saldo*, maka akan muncul tampilan berikut :

Gambar 4.7. Form pengisian saldo

- Isi NIM-nya, secara otomatis akan mencari di database yaitu menampilkan nama dan sisa saldo. Jika NIM tersebut tidak ada dalam database maka form tersebut tidak tampil apa-apa.
- Ketik nominal yang akan ditambahkan, maka total saldo tertambahkan otomatis. Klik *update* maka akan database akan di-update.

4.4.1.5.Pemblokiran Kartu

- Klik *Edit* ➔ *Blokir Kartu*, maka akan muncul tampilan berikut:

Gambar 4.8. Form blokir kartu

- Isi NIM-nya, secara otomatis akan mencari di database yaitu menampilkan nama. Jika NIM tersebut tidak ada dalam database maka form tersebut tidak tampil apa-apa.
- Beri tanda pada blokir kartu, klik OK. Data tersebut tidak akan diproses pada saat kartu magnetik tersebut digesekkan dan akan tampil "Maaf! Kartu anda sudah di blokir" pada LCD.

BAB V

KESIMPULAN

5.1. Kesimpulan

Selama dalam perencanaan dan pengujian sistem parkir prabayar menggunakan kartu magnetik berbasis Mikrokotroller AT89S51 yang telah dibuat, maka dapat ditarik kesimpulan antara lain :

1. Data output yang ditampilkan PC dari kartu magnetik yang digesekkan dengan kecepatan sekitar 10 cm/s tidak ada perbedaannya jika digesekkan dengan kecepatan sekitar 300 cm/s. Hal ini dikarenakan percepatan gesekannnya konstan.
2. Pada pengujian rangkaian sensor infra merah, besi, karet dan kayu merupakan penghalang yang paling baik karena terdapat adanya *error* paling kecil yaitu pada $V_{Photodioda}$ sebesar 3 %.
3. Pada pengujian rangkaian driver motor stepper terdapat adanya *error* yaitu pada V_{motor} sebesar 0,92 %.
4. Tampilan dari software yang dijalankannya tidak pernah mengalami kesalahan selama pengujian, baik dalam menampilkan NIM, Nama, waktu masuk, waktu keluar maupun sisa saldo.

5.2. Saran-saran

1. Dalam penerapan alat ini dalam kehidupan sesungguhnya, sebaiknya dirancang suatu kondisi dimana magnetic card reader berada dalam keadaan terlindungi dari panas matahari dan hujan. Agar dapat menjamin alat bekerja dengan baik dan tahan lama.
2. Sensor infra red sebaiknya dirancang dengan sensitifitas yang tinggi karena cahaya dari luar berpengaruh buruk terhadap sensor ini.
3. Driver yang digunakan untuk menggerakkan palang pintu ini hanya simulasi, apabila diterapkan dalam kondisi yang sebenarnya maka driver harus disesuaikan.
4. Motor yang digunakan pada palang pintu disini hanya simulasi, jadi apabila ingin digunakan pada kondisi yan sebenarnya maka perlu adanya penyesuaian torsi, kecepatan, keamanan dan ketahanan.
5. Kestabilan alat ini akan lebih baik jika menggunakan komponen-komponen yang berpresisi tinggi dan berkualitas baik.
6. Pada penerapannya, sebaiknya disediakan pula tempat parkir untuk umum sehingga dapat mengunjungi kampus.

DAFTAR PUSTAKA

1. www.atmel.com Datasheet of MicroController AT89S51.
2. www.electromag.com/download/head.pdf Archives of Magnetic Stripe Reader.
3. www.p-tec.net Datasheet of InfraRed.
4. www.optoelectronics.perkinelmer.com Datasheet of Photodiodes.
5. www.delta-electronic.com Data Sheet M1632 Module LCD.
6. www.senet.com.au/~epeacock Data Sheet of RS 232.
7. Fairchild Semiconductor, General Puspose 6-pin Phototransistor Optocoupler.
8. www.abilitysystems.com Archives of Stepper Motor
9. Paulus Andi Naiwan, **Teknik Antarmuka dan Pemrograman Mikrokontroller AT89C51**, Elex Media Komputindo, 2003.
10. Hancook, Less & Kriger, Morris, **The C Primer**, McGraw-Hill, 1982.
11. Abdul Kadir, **Pemrograman Dasar Turbo C Untuk IBM PC**, Andi Offset, 1991.

LAMPIRAN

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

LEMBAR PERSETUJUAN PERBAIKAN SKRIPSI

Dalam pelaksanaan Ujian Skripsi Jenjang Strata 1 (S-1) Jurusan Teknik Elektro Konsentrasi Teknik Elektronika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama : AFIV NOER AZHARI
N I M : 01.17.068
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Elektronika
Judul Skripsi : Perancangan dan Pembuatan Sistem Parkir Prabayar Menggunakan Kartu Magnetik (Kartu Mahasiswa) Yang Diaplikasikan Pada Tempat Parkir Kampus Berbasis Mikrokontroller AT89S51

Perbaikan meliputi :

No.	Tanggal	Uraian	Paraf Penguji I	Paraf Penguji II
1.	10-3-2007	Pengujian : mohon ditambahkan item pengujian yang lain dan ditambah errornya.		
2.	10-3-2007	Kesimpulan : eror sistem dimasukkan.		

Disetujui

Penguji I

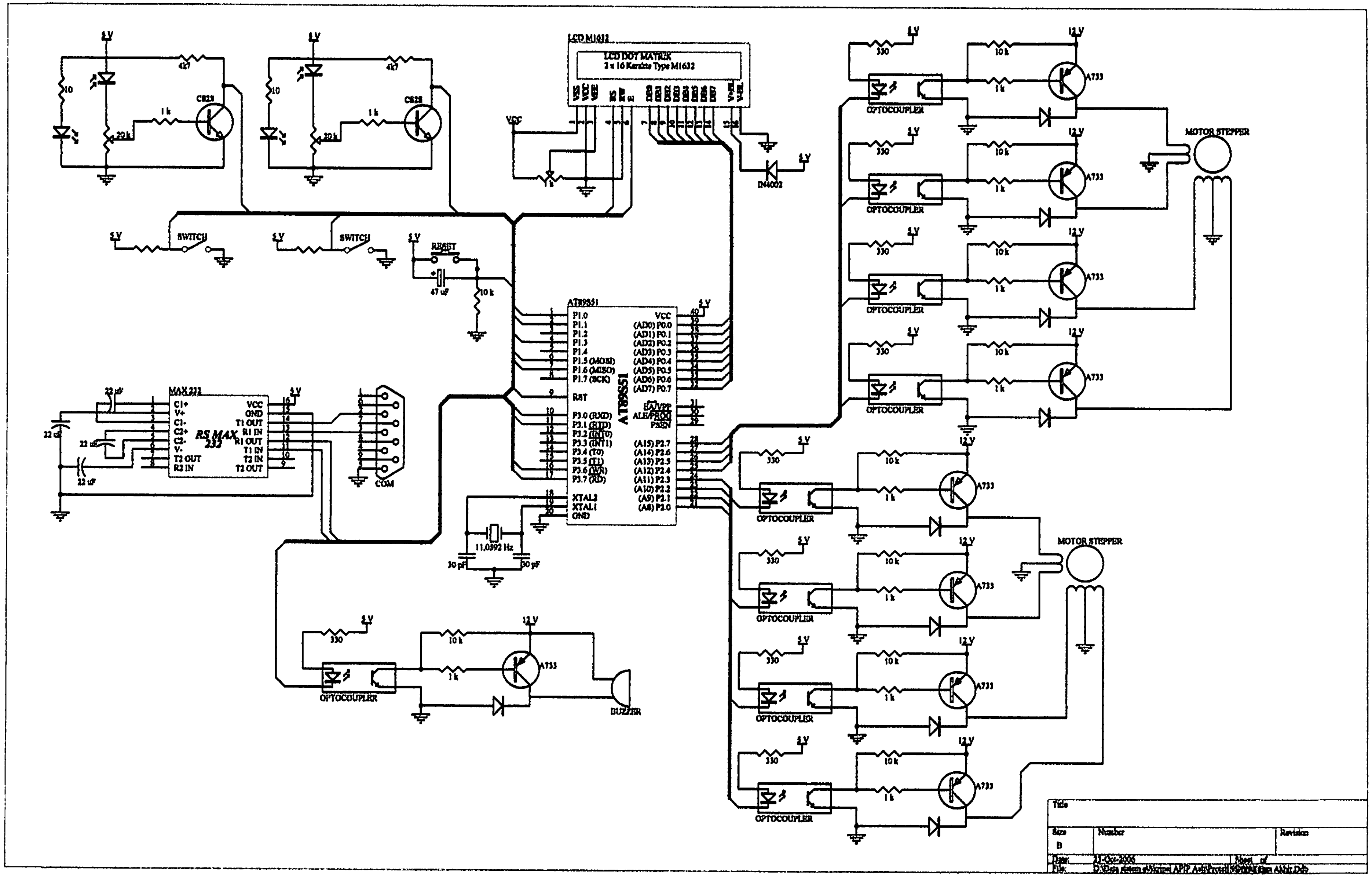
Ir. Usman Djuanda, MM
NIP.Y. 1018700143

Penguji II

Ir. Teguh Herbasuki, MT
NIP.Y. 1028700172

Mengetahui,
Dosen Pembimbing

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


LCD M1632
LCD DOT MATRIK
2 x 16 Karakter Type M1632
AT89S51
MAX 232
RS MAX 232
COM
SWITCH
RESET
47 uF
10 k
11,0592 Hz
30 pF
22 uF
C828
4k7
20 k
1 k
10
330
A733
OPTOCOUPLER
MOTOR STEPPER
BUZZER
IN4002
VCC
5 V
12 V
Title
Size
B
Number
Revision
Date:
Sheet of
File:

```
READY                EQU     0F0H
MOTOR1BUKA           EQU     0F1H
MOTOR1TUTUP          EQU     0F2H
MOTOR2BUKA           EQU     0F3H
MOTOR2TUTUP          EQU     0F4H
NOTREG               EQU     0F5H
HEADER               EQU     0F6H
ENDPOIN              EQU     0F7H
BLOKIR               EQU     0F8H
GACUKUP              EQU     0F9H
DATA_IN              EQU     0FAH
DATA_OUT             EQU     0FBH
CEK_CLOCK            EQU     0FCH

STATUS               BIT     20H.0

SWITCH_MASUK         BIT     P3.4
SWITCH_KELUAR        BIT     P3.5
SENSOR_IN            BIT     P3.2
SENSOR_OUT           BIT     P3.3

;***********INISIAL MEMORY LCD*************
LCD_RS               BIT     P3.7
LCD_EN               BIT     P3.6

TULIS                EQU     30H          ;JUMLAH KARAKTER LCD
POSISI               EQU     31H
KARAKTER             EQU     32H

                     ORG     00H
                     LJMP    START

                     ORG     23H
                     LJMP    CEK_DATA

;********* Inisialisasi LCD ***************
                     ORG     100H
START:               SETB    EA
                     SETB    STATUS
                     MOV     TMOD,#20H
                     MOV     TH1,#0FDH
                     MOV     SCON,#50H
                     SETB    TR1
                     SETB    ES
                     CALL    LDELAY
                     MOV     A,#03Fh              ;FUNCTION SET LCD
                     CALL    INSTRUKSI_LCD
                     CALL    INSTRUKSI_LCD
                     CALL    DELAY
                     MOV     A,#0Ch               ;ON/OFF CONTROL,CURSOR
                     CALL    INSTRUKSI_LCD
                     MOV     A,#06h
                     CALL    INSTRUKSI_LCD
                     MOV     A,#01h               ;CLEAR DISPLAY
                     CALL    INSTRUKSI_LCD
                     MOV     A,#0Ch
                     CALL    INSTRUKSI_LCD
                     CALL    DELAY
                     MOV     DPTR,#COBA
                     CALL    DISPLAY
                     CALL    LDELAY
                     CALL    LDELAY
                     CALL    DISPLAY
                     CALL    LDELAY
                     CALL    LDELAY
                     CALL    DISPLAY
                     CALL    LDELAY
                     CALL    LDELAY

NORMAL:              MOV     DPTR,#AWAL
                     CALL    BARISA
                     CALL    BARISB

                     CALL    BUKA_PINTU_MASUK
                     CALL    BUKA_PINTU_KELUAR
                     CALL    TUTUP_PINTU_MASUK
                     CALL    TUTUP_PINTU_KELUAR

CEKING:              JB      SENSOR_IN,CS2
                     MOV     A,#DATA_IN
                     CLR     ES
                     CALL    SEND
                     SETB    ES
                     CALL    DELAY
                     JNB     SENSOR_IN,$
```

```
CS2:                    JB          SENSOR_OUT,CEKING
                        MOV         A,#DATA_OUT
                        CLR         ES
                        CALL        SEND
                        SETB        ES
                        CALL        DELAY
                        JNB         SENSOR_OUT,$
                        SJMP        CEKING

TUTUP_PINTU_MASUK:      MOV         POSISI,#15
CLOSE1:                 CALL        PINTU_MASUK_TUTUP
                        DJNZ        POSISI,CLOSE1
                        RET

TUTUP_PINTU_KELUAR:     MOV         POSISI,#15
CLOSE2:                 CALL        PINTU_KELUAR_TUTUP
                        DJNZ        POSISI,CLOSE2
                        RET

BUKA_PINTU_KELUAR:      JNB         SWITCH_KELUAR,OKES
LOP2:                   CALL        PINTU_KELUAR_BUKA
                        JB          SWITCH_KELUAR,LOP2
OKES:                   RET

BUKA_PINTU_MASUK:       JNB         SWITCH_MASUK,CEKSW2
LOP1:                   CALL        PINTU_MASUK_BUKA
                        JB          SWITCH_MASUK,LOP1
CEKSW2:                 RET

PINTU_MASUK_TUTUP:      MOV         P2,#11110111B
                        CALL        TUNDA
                        MOV         P2,#11111011B
                        CALL        TUNDA
                        MOV         P2,#11111101B
                        CALL        TUNDA
                        MOV         P2,#11111110B
                        CALL        TUNDA
                        MOV         P2,#0FFH
                        RET

PINTU_MASUK_BUKA:       MOV         P2,#11111110B
                        CALL        TUNDA
                        MOV         P2,#11111101B
                        CALL        TUNDA
                        MOV         P2,#11111011B
                        CALL        TUNDA
                        MOV         P2,#11110111B
                        CALL        TUNDA
                        MOV         P2,#0FFH
                        RET

PINTU_KELUAR_BUKA:      MOV         P2,#01111111B
                        CALL        TUNDA
                        MOV         P2,#10111111B
                        CALL        TUNDA
                        MOV         P2,#11011111B
                        CALL        TUNDA
                        MOV         P2,#11101111B
                        CALL        TUNDA
                        MOV         P2,#0FFH
                        RET

PINTU_KELUAR_TUTUP:     MOV         P2,#11101111B
                        CALL        TUNDA
                        MOV         P2,#11011111B
                        CALL        TUNDA
                        MOV         P2,#10111111B
                        CALL        TUNDA
                        MOV         P2,#01111111B
                        CALL        TUNDA
                        MOV         P2,#0FFH
                        RET

TUNDA:                  MOV         R6,#80
TND:                    DJNZ        R7,$
                        DJNZ        R6,TND
                        RET

CEK_DATA:               CLR         ES
                        PUSH        ACC
                        JNB         RI,$
                        CLR         RI
                        MOV         A,SBUF
                        JB          STATUS,CK1
                        CJNE        A,#ENDPOIN,CEKIN
                        CLR         A
```

```
        MOV     @R0,A
        SETB    STATUS
        CALL    TAMPIL_SALDO
        JMP     OUT
CEKIN:  CJNE    A,#CEK_CLOCK,ISI
        CLR     A
        MOV     @R0,A
        SETB    STATUS
        CALL    TAMPIL_CEK_IN
        JMP     OUT
ISI:    MOV     @R0,A               ;ISI DATA SALDO
        INC     R0
        JMP     OUT
CK1:    CJNE    A,#READY,CK2
        MOV     DPTR,#AWAL
        CALL    BARISA
        CALL    BARISB
        CALL    SEND
        JMP     OUT
CK2:    CJNE    A,#MOTOR1BUKA,CK3
        CALL    BUKA_PINTU_MASUK
        JMP     OUT
CK3:    CJNE    A,#MOTOR1TUTUP,CK4
        CALL    TUTUP_PINTU_MASUK
        JMP     OUT
CK4:    CJNE    A,#MOTOR2BUKA,CK5
        CALL    BUKA_PINTU_KELUAR
        JMP     OUT
CK5:    CJNE    A,#MOTOR2TUTUP,CKNR
        CALL    TUTUP_PINTU_KELUAR
        JMP     OUT
CKNR:   CJNE    A,#NOTREG,CKAR
        MOV     DPTR,#NOT_REGISTER
        CALL    BARISA
        CALL    BARISB
        JMP     OUT
CKAR:   CJNE    A,#HEADER,BLK
        CLR     STATUS
        MOV     R0,#KARAKTER
        JMP     OUT
BLK:    CJNE    A,#BLOKIR,GCK
        MOV     DPTR,#BLOK
        CALL    BARISA
        CALL    BARISB
        JMP     OUT
GCK:    CJNE    A,#GACUKUP,OUT
        MOV     DPTR,#GAK_CUKUP
        CALL    BARISA
        CALL    BARISB
OUT:    POP     ACC
        SETB    ES
        RETI

TAMPIL_CEK_IN:  MOV     DPTR,#CEK_IN
        CALL    BARISA
        CALL    BARISB
        MOV     A,#88H              ;POSISI JAM
        CALL    INSTRUKSI_LCD
        MOV     R0,#KARAKTER
        MOV     A,@R0
        CALL    DATA_LCD
        INC     R0
        MOV     A,@R0
        CALL    DATA_LCD
        INC     R0
        MOV     A,#8DH              ;POSISI MENIT
        CALL    INSTRUKSI_LCD
        MOV     A,@R0
        CALL    DATA_LCD
        INC     R0
        MOV     A,@R0
        CALL    DATA_LCD
        INC     R0
        MOV     A,#0C0H             ;POSISI NIM
        CALL    INSTRUKSI_LCD
LOOP3:  MOV     A,@R0
        CJNE    A,#0,LCD
        RET
LCD:    CALL    DATA_LCD
        INC     R0
        JMP     LOOP3

TAMPIL_SALDO:   MOV     DPTR,#SALDO
        CALL    BARISA
        CALL    BARISB
        MOV     A,#0C3H
```

```
                    CALL       INSTRUKSI_LCD
                    MOV        R0,#KARAKTER
LOOP2:              MOV        A,@R0
                    CJNE       A,#0,TLCD
                    RET
TLCD:               CALL       DATA_LCD
                    INC        R0
                    JMP        LOOP2
SEND:               MOV        SBUF,A
                    JNB        TI,$
                    CLR        TI
                    RET
DELAY:              MOV        R7,#150
DELAY1: MOV         R5,#50h
                    DJNZ       R5,$
                    DJNZ       R7,DELAY1
                    RET
;
LDELAY: MOV         R2,#040h
LD1:                CALL       DELAY
                    DJNZ       R2,LD1
                    RET

;========================================
; Routine untuk menulis instruksi ke LCD
;========================================

INSTRUKSI_LCD:      CLR        LCD_RS
                    SETB       LCD_EN
                    MOV        P0,A           ;intruksi ke LCD
                    CLR        LCD_EN         ;module
                    SETB       LCD_EN
                    DJNZ       R7,$
                    DJNZ       R7,$
                    RET

;========================================
; Routine untuk menulis data ke LCD
;========================================

DATA_LCD:           SETB       LCD_RS
                    SETB       LCD_EN
                    MOV        P0,A           ;data ke LCD
                    CLR        LCD_EN         ;module
                    SETB       LCD_EN
                    DJNZ       R7,$
                    DJNZ       R7,$
                    RET
DISPLAY:            CALL       BARISA
                    CALL       BARISB
                    RET

;========================================

;menuliskan text judul di LCD atas
;========================================
barisa:             mov        TULIS,#16          ; tulis pada baris 1 sebanyak 16 char
                    mov        A,#80h             ; mulai dari kiri atas
                    acall      INSTRUKSI_LCD
tulis1:             clr        A
                    movc       A,@A+DPTR
                    Inc        DPTR
                    acall      DATA_LCD
                    djnz       TULIS,Tulis1
                    ret

;========================================
; menuliskan data dari ram terima
; ke LCD baris 2 sebanyak 16 alamat
;========================================
barisb:             mov        TULIS,#16          ; tulis pada baris 2 sebanyak 16 char
                    mov        A,#0C0h            ; mulai dari kiri bawah
                    acall      INSTRUKSI_LCD
tulis2:             clr        A
                    movc       A,@A+DPTR
                    INC        DPTR
                    acall      DATA_LCD
                    djnz       TULIS,Tulis2
                    ret

COBA:               DB         'TUGAS AKHIR 2006'
                    DB         '   ITN Malang   '
                    DB         ' Disusun Oleh : '
                    DB         'Afif            '
                    DB         '--- LOADING --- '
                    DB         '----------------'
```

```
AWAL:              DB        ' Silahkan Gesek '
                   DB        '  Kartu Anda    '
NOT_REGISTER:
                   DB        'Maaf! Nama Anda '
                   DB        'Tidak Terdaftar.'
SALDO:             DB        'Sisa SALDO Anda:'
                   DB        'Rp.             '
GAK_CUKUP:
                   DB        'Maaf! Saldo Anda'
                   DB        'Tidak Cukup.....'
BLOK:              DB        'Maaf! Kartu Anda'
                   DB        'Sudah di BLOKIR.'

CEK_IN:            DB        'CEK IN     :    '
                   DB        '                '

                   END
```

```
nit Unit1;

nterface

ses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, Menus, ImgList, ComCtrls, StdCtrls, Grids, Buttons, jpeg,
  ExtCtrls,REGISTRY, CPort;

ype
  TForm1 = class(TForm)
    Panel1: TPanel;
    StaticText2: TStaticText;
    Panel3: TPanel;
    Image1: TImage;
    StringGrid1: TStringGrid;
    Memo1: TMemo;
    Panel2: TPanel;
    Image3: TImage;
    SpeedButton1: TSpeedButton;
    StatusBar1: TStatusBar;
    Panel4: TPanel;
    ImageList1: TImageList;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    Open1: TMenuItem;
    Simpan1: TMenuItem;
    SetupPrinter1: TMenuItem;
    Exit1: TMenuItem;
    Edit1: TMenuItem;
    Info1: TMenuItem;
    Programmer1: TMenuItem;
    SpeedButton2: TSpeedButton;
    Edit2: TEdit;
    Label1: TLabel;
    Timer1: TTimer;
    arifParkir1: TMenuItem;
    ComPort1: TComPort;
    Timer2: TTimer;
    SpeedButton3: TSpeedButton;
    RefikkSaldo1: TMenuItem;
    Memo2: TMemo;
    proses: TMemo;
    Panel5: TPanel;
    Image2: TImage;
    Timer3: TTimer;
    Panel6: TPanel;
    SpeedButton4: TSpeedButton;
    SpeedButton5: TSpeedButton;
    Panel7: TPanel;
    SpeedButton6: TSpeedButton;
    SpeedButton7: TSpeedButton;
    Timer4: TTimer;
    BlokirKartu1: TMenuItem;
    Memo3: TMemo;
    Panel8: TPanel;
    procedure FormCreate(Sender: TObject);
    procedure SpeedButton1Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);
    procedure SpeedButton2Click(Sender: TObject);
    procedure arifParkir1Click(Sender: TObject);
    procedure Edit2KeyPress(Sender: TObject; var Key: Char);
    procedure Timer2Timer(Sender: TObject);
    procedure RefikkSaldo1Click(Sender: TObject);
    procedure Timer3Timer(Sender: TObject);
    procedure SpeedButton3Click(Sender: TObject);
    procedure FormShow(Sender: TObject);
    procedure Programmer1Click(Sender: TObject);
    procedure ComPort1RxChar(Sender: TObject; Count: Integer);
    procedure SpeedButton4Click(Sender: TObject);
    procedure SpeedButton5Click(Sender: TObject);
    procedure SpeedButton6Click(Sender: TObject);
    procedure SpeedButton7Click(Sender: TObject);
    procedure Timer4Timer(Sender: TObject);
    procedure BlokirKartu1Click(Sender: TObject);
```

```
 private
   { Private declarations }
 public
   { Public declarations }
 end;

:onst
·eady=$f0;
ılopen=$f1;
ılclose=$f2;
ı2open=$f3;
ı2close=$f4;
ıotreg=$f5;
ıeader=$f6;
ındpoin=$f7;
›lokir=$f8;
ıakcukup=$f9;
lata_in=$fa;
lata_out=$fb;
:ekclock=$fc;

'ar
 Form1: TForm1;
 namafile,namafile2,blockfile:string;
 money,counter,hitung,posisi:integer;
 MASUK:BOOLEAN;
mplementation

ses Unit2, Unit4, Unit5, Unit6, Unit7, Unit8, Unit9;

$R *.dfm}

rocedure TForm1.FormCreate(Sender: TObject);
AR REG:TREGISTRY;
egin
       counter:=1;
       memo1.Hide;
       memo2.Hide;
       memo3.Hide;
       proses.Hide;
       stringgrid1.ColWidths[0]:=160;          //no
       stringgrid1.ColWidths[1]:=210;         //nama
       stringgrid1.ColWidths[2]:=170;         //cek in
       stringgrid1.ColWidths[3]:=170;         //cek out
       stringgrid1.ColWidths[4]:=200;         //saldo

       stringgrid1.Cells[0,0]:=' NIM ';
       stringgrid1.Cells[1,0]:=' Nama ';
       stringgrid1.Cells[2,0]:=' Cek IN';
       stringgrid1.Cells[3,0]:=' Cek OUT';
       stringgrid1.Cells[4,0]:=' Saldo ';

       SPEEDBUTTON5.Enabled:=FALSE;
       SPEEDBUTTON7.Enabled:=FALSE;

       namafile:=ExtractFilePath(Application.ExeName)+'update.itn';
       namafile2:=ExtractFilePath(Application.ExeName)+'arsip.itn';
       blockfile:=ExtractFilePath(Application.ExeName)+'block.itn';
       try
               memo1.Lines.LoadFromFile(namafile);
       except
               memo1.Lines.SaveToFile(namafile);
       end;
       try
               proses.Lines.LoadFromFile(namafile2);
       except
               proses.Lines.SaveToFile(namafile2);
       end;

       try
               memo3.Lines.LoadFromFile(blockfile);
       except
               memo3.Lines.SaveToFile(blockfile);
       end;
```

```
        try
                memo2.Lines.LoadFromFile(extractfilepath(application.ExeName)+'proses.tmp');
        except
                memo2.Lines.LoadFromFile(extractfilepath(application.ExeName)+'proses.tmp');
        end;

        reg:=tregistry.Create;
        reg.RootKey:=HKEY_CLASSES_ROOT;
        if reg.OpenKey('Recnot\itnsoft\tts',true) then
        begin
                try
                        reg.ReadInteger('registrasi');
                except
                        reg.WriteInteger('BIAYA',0);
                        reg.WriteString('serie','SISTEM PARKIR');
                        reg.WriteString('Prog','AFIF ITN');
                        reg.CloseKey;
                        exit;
                end;
                money:=reg.ReadInteger('registrasi');
                reg.Free;
                panel8.Caption:='Biaya Parkir : Rp. '+inttostr(money);
        end;
nd;

rocedure TForm1.SpeedButton1Click(Sender: TObject);
egin
        timer1.Enabled:=false;
        form2.show;
        form1.Hide;
        form2.Edit1.Text:='';
        form2.label1.Caption:='Gesekkan Kartu Maghnetic Anda';
        form2.edit1.SelLength:=1;
        form2.edit1.SelStart:=0;
nd;

rocedure TForm1.Timer1Timer(Sender: TObject);
egin
        edit2.SetFocus;
nd;

rocedure TForm1.SpeedButton2Click(Sender: TObject);
egin
        form4.show;
        form1.Hide;
        timer1.Enabled:=false;
nd;

rocedure TForm1.arifParkir1Click(Sender: TObject);
egin
        TIMER1.Enabled:=FALSE;
        form5.show;
nd;

rocedure TForm1.Edit2KeyPress(Sender: TObject; var Key: Char);
ar      i,j,biaya,k,cekblock:integer;
        DATA,TEMP,waktu,nim,nama:STRING;
        present:tdatetime;
        tanggal,bulan,tahun,jam,menit,detik,mdetik:word;
        reg:tregistry;
egin
        reg:=tregistry.Create;
        reg.RootKey:=HKEY_CLASSES_ROOT;
        if reg.OpenKey('Recnot\itnsoft\tts',true) then
        begin
                biaya:=reg.ReadInteger('registrasi');
                reg.CloseKey;
                reg.Free;
        end;

        if key='=' then
        begin
                HITUNG:=0;
                edit2.MaxLength:=length(edit2.Text);
                present:=now;
                decodedate(present,tahun,bulan,tanggal);
```

```
                waktu:=' '+format('%0.2d',[tanggal])+' - '+format('%0.2d',[bulan])+' -
+format('%0.2d',[tahun])+'      Jam '+format('%0.2d',[jam])+' : '+format('%0.2d',[Menit])+' :
+format('%0.2d',[detik]);
                I:=MEMO1.Lines.IndexOfName(EDIT2.Text);
                IF I>-1 THEN
                BEGIN
                        cekblock:=MEMO3.Lines.IndexOfName(EDIT2.Text);
                        if cekblock>-1 then
                        begin
                                 comport1.WriteStr(char(blokir));
                                 sleep(4000);
                                 comport1.WriteStr(char(ready));
                                 edit2.Text:='';
                                 exit;
                        end;

                        data:=form1.Memo1.Lines.Strings[i];
                        delete(data,1,pos('=',data));
                        temp:=copy(data,1,pos(char($f1),data)-1);  //nim
                        form6.PANEL2.Caption:='NIM : '+temp;
                        nim:=temp;
                        delete(data,1,pos(char($f1),data));

                        temp:=copy(data,1,pos(char($f2),data)-1); //nama
                        form6.PANEL3.Caption:='NAMA : '+temp;
                        nama:=temp;
                        delete(data,1,pos(char($f2),data));


                        temp:=copy(data,1,pos(char($f3),data)-1);  //saldo
                        j:=strtoint(temp); //saldo akhir
                        delete(data,1,pos(char($f3),data));
                        if biaya>j then
                        begin
                                comport1.WriteStr(char(gakcukup));
                                sleep(3000);
                                comport1.WriteStr(char(ready));
                        end else
                        begin

                        k:=Memo2.Lines.IndexOfName(EDIT2.Text);
                        if k=-1 then
                        begin            //cek in
                                data:=form1.Memo1.Lines.Strings[i];
                                   //proses isi data cek in
                                insert(waktu,data,pos(char($f4),data)+1);
                                Memo2.Lines.Add(DATA);

emo2.Lines.SaveToFile(extractfilepath(application.ExeName)+'proses.tmp');
                               //
omport1.WriteStr(char(header)+format('%0.2d',[jam])+format('%0.2d',[Menit])+'NIM:'+nim+char(c
kclock));
                             //  sleep(4000);
                                comport1.WriteStr(char(header)+inttostr(j-
iaya)+char(endpoin));
                                Form6.StaticText1.Caption:='CEK IN';
                                form6.panel4.Caption:='Saldo Saat Ini : Rp. '+ inttostr(j-
iaya);
                                form6.Show;
                                memo2.Modified:=true;
                                MASUK:=TRUE;
                                TIMER4.Enabled:=TRUE;
                                form6.Timer1.Enabled:=true;
                        end else
                        begin     //cek out
                                temp:=memo2.Lines.Strings[k];
                                insert(waktu,temp,pos(char($f5),temp)+1);

                                memo2.Lines.Delete(k);

emo2.Lines.SaveToFile(extractfilepath(application.ExeName)+'proses.tmp');     //data prosesing

                                Form6.StaticText1.Caption:='CEK OUT';
```

```
                                form6.panel4.Caption:='Sisa SALDO = Rp. '+inttostr(j-biaya);
                                stringgrid1.Cells[3,k+1]:=waktu;
                                comport1.WriteStr(char(header)+inttostr(j-
biaya)+char(endpoin));
                                data:=form1.Memo1.Lines.Strings[i];
                                delete(data,pos(char($f2),data)+1,pos(char($f3),data)-
pos(char($f2),data)-1);
                                insert(inttostr(j-biaya),data,pos(char($f2),data)+1);
                                memo1.Lines.Delete(i);
                                memo1.Lines.Insert(i,data);     //database
                                insert(inttostr(biaya),temp,pos(char($f6),temp)+1);
//biaya parkir
                                proses.Lines.Add(temp);
                                timer1.Enabled:=false;
                                form6.Timer1.Enabled:=true;
                                memo1.Lines.SaveToFile(namafile);

                                proses.Lines.SaveToFile(namafile2);
                                edit2.Text:='';
                                memo2.Modified:=true;
                                MASUK:=FALSE;
                                TIMER4.Enabled:=TRUE;
                                form6.show;
                        end;
                        counter:=counter+1;
                        end;
                end else
                begin
                   comport1.WriteStr(char(notreg));
                   sleep(4000);
                   comport1.WriteStr(char(ready));
                end;
                timer2.Enabled:=true;
                key:=char(13);
        end;
nd;

rocedure TForm1.Timer2Timer(Sender: TObject);
egin
        edit2.Text:='';
        edit2.MaxLength:=0;
        timer2.Enabled:=false;
nd;

rocedure TForm1.RefikkSaldo1Click(Sender: TObject);
egin
        timer1.Enabled:=false;
        form7.show;
        form1.Hide;
nd;

rocedure TForm1.Timer3Timer(Sender: TObject);
ar      i,j:integer;
        data,temp:string;
egin
/         PANEL6.Caption:=INTTOSTR(HITUNG);
        if memo2.Modified=true then
        begin
        if memo2.Text<>'' then
        begin
                stringgrid1.RowCount:=Memo2.Lines.Count+1;
                for i:=0 to form1.Memo2.Lines.Count do
                begin
                        stringgrid1.Rows[1+1].Clear;

                end;
                for i:=0 to form1.proses.Lines.Count do
                begin
                        data:=memo2.Lines.Strings[i];
                        if data='' then exit;
                        delete(data,1,pos('=',data));
                        temp:=copy(data,1,pos(char($f1),data)-1);
                        stringgrid1.Cells[0,i+1]:=temp;

                        delete(data,1,pos(char($f1),data));
                        temp:=copy(data,1,pos(char($f2),data)-1);
```

```
                        delete(data,1,pos(char($f2),data));
                        temp:=copy(data,1,pos(char($f3),data)-1);
                        j:=strtoint(temp)-money;
                        //label1.Caption:=temp;
                        stringgrid1.Cells[4,i+1]:=inttostr(j);    //sisa saldo

                        delete(data,1,pos(char($f4),data));
                        temp:=copy(data,1,pos(char($f5),data)-1);
                        stringgrid1.Cells[2,i+1]:=temp;    //cek in

                        delete(data,1,pos(char($f5),data));
                        temp:=copy(data,1,pos(char($f6),data)-1);
                        stringgrid1.Cells[3,i+1]:=temp;    //cek out

                end;
                memo2.Modified:=false;
        end;
        end;
nd;

rocedure TForm1.SpeedButton3Click(Sender: TObject);
egin
        timer1.Enabled:=false;
        form1.Hide;
        form8.show;
nd;

rocedure TForm1.FormShow(Sender: TObject);
egin
        memo2.Modified:=true;
        memo1.Modified:=true;
        proses.Modified:=true;
nd;

rocedure TForm1.Programmer1Click(Sender: TObject);
egin
        SHOWMESSAGE('Di Bikin Oleh Apip Gitu Loh.');
nd;

rocedure TForm1.ComPort1RxChar(Sender: TObject; Count: Integer);
ar      data:byte;
egin
        comport1.Read(data,count);
        //label1.Caption:=inttostr(data)+':'+INTTOSTR(HITUNG)+':'+INTTOSTR(POSISI);
        if data=ready then TIMER4.Enabled:=FALSE;
        if (data=$fa)and (HITUNG=2) then
        BEGIN
                comport1.WriteStr(char(m1close));
                TIMER4.Enabled:=FALSE;
                HITUNG:=0;
        END;
        if (data=$fB)and (HITUNG=2) then
        BEGIN
                comport1.WriteStr(char(m2close));
                TIMER4.Enabled:=FALSE;
                HITUNG:=0;
        END;
        HITUNG:=HITUNG+1;
        SLEEP(400);
nd;

rocedure TForm1.SpeedButton4Click(Sender: TObject);
egin
        SPEEDBUTTON4.Enabled:=FALSE;
        SPEEDBUTTON5.Enabled:=TRUE;
        COMPORT1.WriteStr(CHAR(M1OPEN));
nd;

rocedure TForm1.SpeedButton5Click(Sender: TObject);
egin
        SPEEDBUTTON4.Enabled:=TRUE;
        SPEEDBUTTON5.Enabled:=FALSE;
        COMPORT1.WriteStr(CHAR(M1CLOSE));
nd;
```

```
rocedure TForm1.SpeedButton6Click(Sender: TObject);
egin
        SPEEDBUTTON7.Enabled:=TRUE;
        SPEEDBUTTON6.Enabled:=FALSE;
        COMPORT1.WriteStr(CHAR(M2OPEN));
nd;

rocedure TForm1.SpeedButton7Click(Sender: TObject);
egin
        SPEEDBUTTON6.Enabled:=TRUE;
        SPEEDBUTTON7.Enabled:=FALSE;
        COMPORT1.WriteStr(CHAR(M2CLOSE));
nd;

rocedure TForm1.Timer4Timer(Sender: TObject);
egin
        TIMER4.Enabled:=FALSE;
        IF (MASUK=TRUE) AND (HITUNG=1) THEN
        BEGIN
                COMPORT1.WriteStr(CHAR(M1CLOSE));
        END;
        IF (MASUK=FALSE) AND (HITUNG=1) THEN
        BEGIN
                COMPORT1.WriteStr(CHAR(M2CLOSE));
        END;
nd;

rocedure TForm1.BlokirKartu1Click(Sender: TObject);
egin
        timer1.Enabled:=false;
        form9.Panel3.Hide;
        form9.show;
        form1.Hide;
nd;

nd.
```

›atures

- ›ompatible with MCS®-51 Products
- lK Bytes of In-System Programmable (ISP) Flash Memory
  - Endurance: 1000 Write/Erase Cycles
- l.0V to 5.5V Operating Range
- ·ully Static Operation: 0 Hz to 33 MHz
- Three-level Program Memory Lock
- l28 x 8-bit Internal RAM
- l2 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- ·ull Duplex UART Serial Channel
- .ow-power Idle and Power-down Modes
- nterrupt Recovery from Power-down Mode
- Vatchdog Timer
-)ual Data Pointer
- ›ower-off Flag
- ·ast Programming Time
- ·lexible ISP Programming (Byte and Page Mode)
- ›reen (Pb/Halide-free) Packaging Option

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

AT89S51

Description

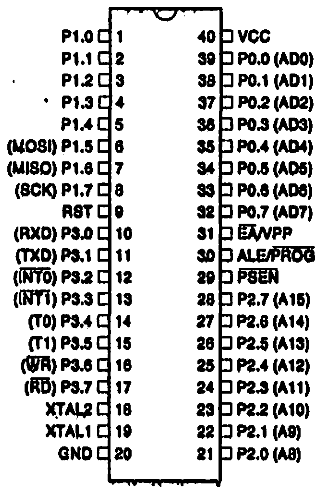
› AT89S51 is a low-power, high-performance CMOS 8-bit microcontroller with 4K ›es of In-System Programmable Flash memory. The device is manufactured using ›nel's high-density nonvolatile memory technology and is compatible with the indus- ›standard 80C51 instruction set and pinout. The on-chip Flash allows the program ›mory to be reprogrammed in-system or by a conventional nonvolatile memory pro- ›mmer. By combining a versatile 8-bit CPU with In-System Programmable Flash on ›onolithic chip, the Atmel AT89S51 is a powerful microcontroller which provides a ›hly-flexible and cost-effective solution to many embedded control applications.

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

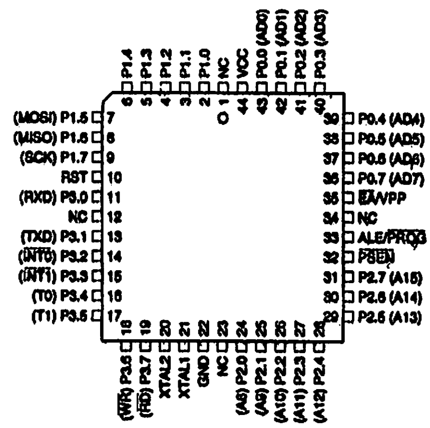
2487C–MICRO–03/05

Pin Configurations

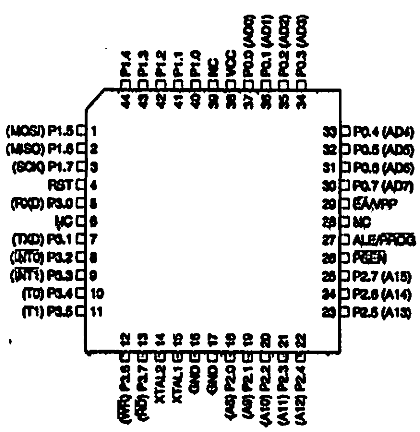
40-lead PDIP



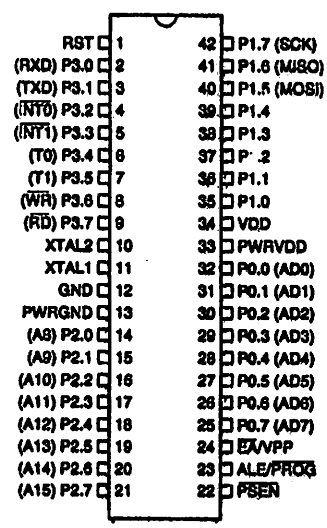
2.3 44-lead PLCC



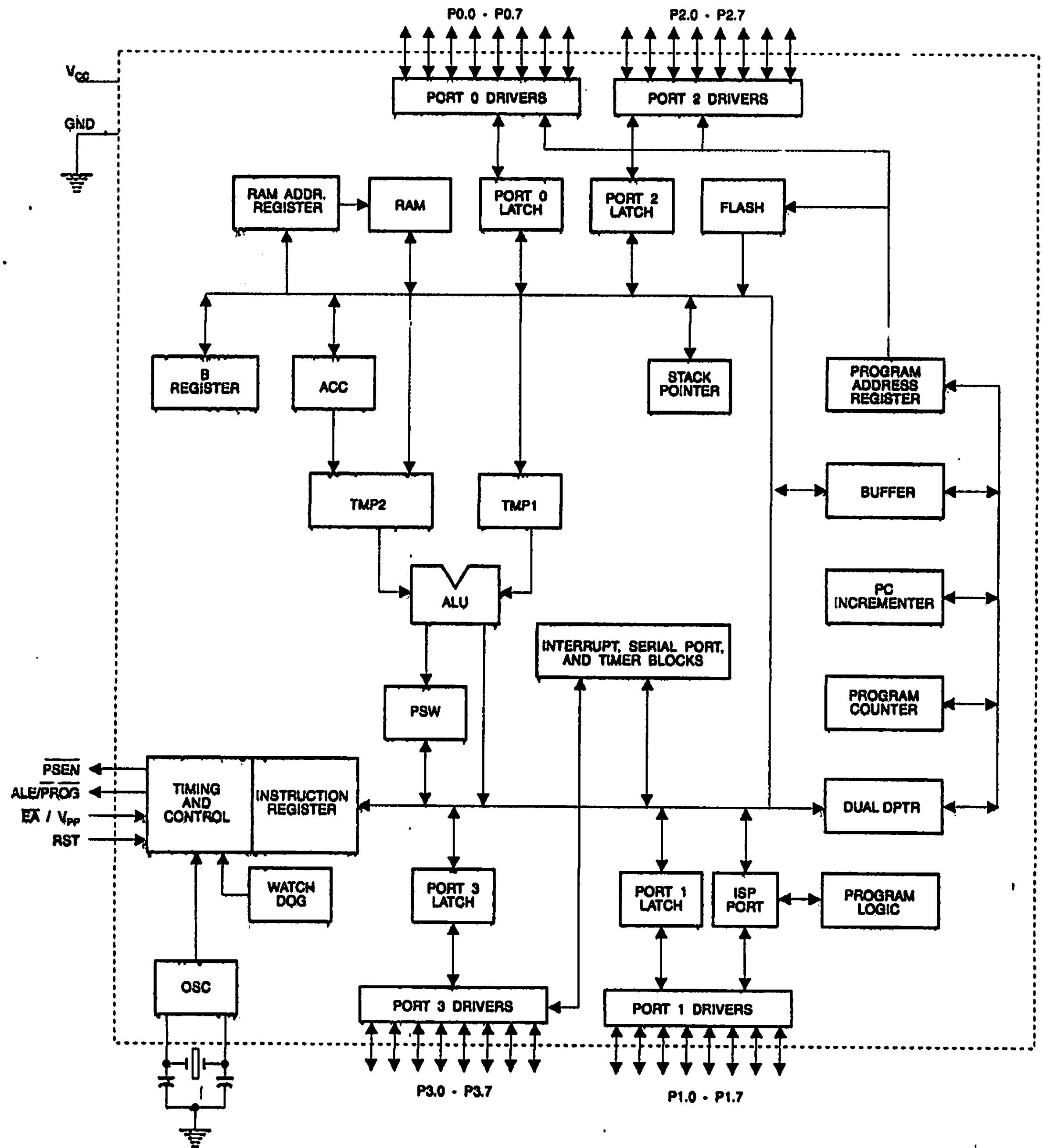
44-lead TQFP



2.4 42-lead PDIP



Block Diagram



Pin Description

VCC

Supply voltage (all packages except 42-PDIP).

GND

Ground (all packages except 42-PDIP; for 42-PDIP GND connects only the logic core and the embedded program memory).

VDD

Supply voltage for the 42-PDIP which connects only the logic core and the embedded program memory.

PWRVDD

Supply voltage for the 42-PDIP which connects only the I/O Pad Drivers. The application board **MUST** connect both VDD and PWRVDD to the board supply voltage.

PWRGND

Ground for the 42-PDIP which connects only the I/O Pad Drivers. PWRGND and GND are weakly connected through the common silicon substrate, but not through any metal link. The application board **MUST** connect both GND and PWRGND to the board ground.

Port 0

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

Port 0 can also be configured to be the multiplexed 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.**

Port 1

Port 1 is an 8-bit bi-directional 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)

Port 2

Port 2 is an 8-bit bi-directional 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.

Port 3

Port 3 is an 8-bit bi-directional 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	$\overline{\text{INT0}}$ (external interrupt 0)
P3.3	$\overline{\text{INT1}}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{\text{WR}}$ (external data memory write strobe)
P3.7	$\overline{\text{RD}}$ (external data memory read strobe)

10 RST

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.

11 ALE/$\overline{\text{PROG}}$

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 ($\overline{\text{PROG}}$) 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.

12 $\overline{\text{PSEN}}$

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

When the AT89S51 is executing code from external program memory, $\overline{\text{PSEN}}$ is activated twice each machine cycle, except that two $\overline{\text{PSEN}}$ activations are skipped during each access to external data memory.

13 $\overline{\text{EA}}$/VPP

External Access Enable. $\overline{\text{EA}}$ 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, $\overline{\text{EA}}$ will be internally latched on reset.

$\overline{\text{EA}}$ 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.

14 XTAL1

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

15 XTAL2

Output from the inverting oscillator amplifier

Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 5-1.

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.

ble 5-1. AT89S51 SFR Map and Reset Values

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	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

er software should not write 1s to these unlisted locations, since they may be used in future products to invoke new fea-
es. In that case, the reset or inactive values of the new bits will always be 0.

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

Table 5-2. 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 5-3. 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

Memory Organization

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.

1 Program 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.

2 Data 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.

Watchdog Timer (One-time Enabled with Reset-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.

1 Using 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.

2 WDT During Power-down and 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.

UART

The UART in the AT89S51 operates the same way as the UART in the AT89C51. For further information on the UART operation, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC4316.PDF

Timer 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, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC4316.PDF

0. Interrupts

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 10-1.

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 10-1 shows that bit positions IE.6 and IE.5 are 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 10-1. Interrupt Enable (IE) Register

(MSB) (LSB)

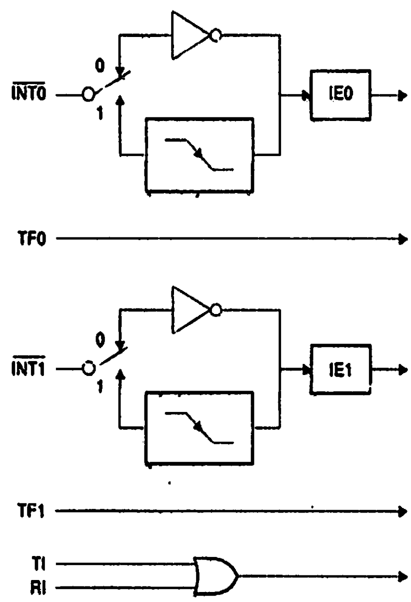
EA	–	–	ES	ET1	EX1	ET0	EX0

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.		

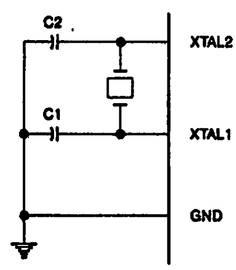
Figure 10-1. Interrupt Sources



Oscillator Characteristics

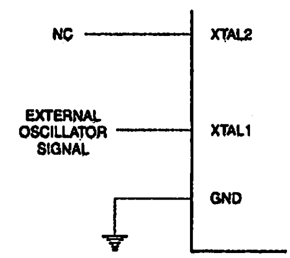
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 11-1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 11-2. 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 11-1. Oscillator Connections



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

Figure 11-2. External Clock Drive Configuration



2. Idle 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.

3. Power-down Mode

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 ($\overline{INT0}$ 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 13-1. 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

4. Program Memory Lock Bits

The AT89S51 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in Table 14-1.

Table 14-1. 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.

5. Programming the Flash – Parallel 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 Modes table (Table 17-1) and Figure 17-1 and Figure 17-2. 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.

$\overline{\text{Data}}$ Polling: The AT89S51 features $\overline{\text{Data}}$ 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. $\overline{\text{Data}}$ 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/$\overline{BSY}$ output signal. P3.0 is pulled low after ALE goes high during programming to indicate $\overline{BUSY}$. 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 AT89S51
(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.

6. Programming the Flash – Serial 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.

6.1 Serial Programming Algorithm

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

1. Power-up sequence:
   a. Apply power between VCC and GND pins.
   b. 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):

1. Set XTAL1 to "L" (if a crystal is not used).
2. Set RST to "L".
3. Turn V_{CC} power off.

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.

6.2 Serial Programming Instruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in the "Serial Programming Instruction Set" on page 20.

7. Programming Interface – Parallel 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.

Most major worldwide programming vendors offer worldwide support for the Atmel AT89 microcontroller series. Please contact your local programming vendor for the appropriate software revision.

able 17-1. Flash Programming Modes

Mode	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
Write Code Data	5V	H	L	⊔ (2)	12V	L	H	H	H	H	D_{IN}	A11-8	A7-0
Read Code Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A11-8	A7-0
Write Lock Bit 1	5V	H	L	⊔ (3)	12V	H	H	H	H	H	X	X	X
Write Lock Bit 2	5V	H	L	⊔ (3)	12V	H	H	H	L	L	X	X	X
Write Lock Bit 3	5V	H	L	⊔ (3)	12V	H	L	H	H	L	X	X	X
Read Lock Bits 1, 2, 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Chip Erase	5V	H	L	⊔ (1)	12V	H	L	H	L	L	X	X	X
Read Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	0000	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	51H	0001	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	06H	0010	00H

otes:
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 17-1. Programming the Flash Memory (Parallel Mode)

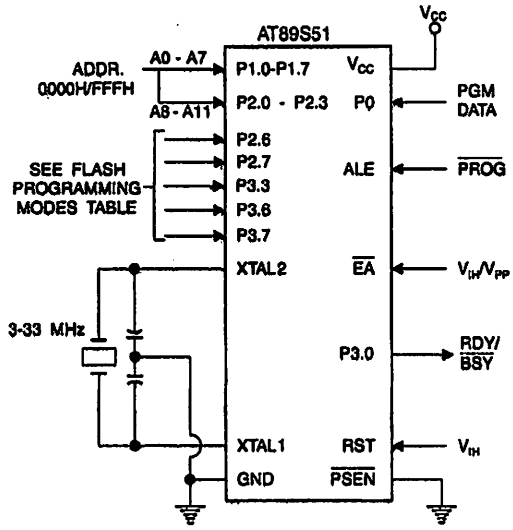
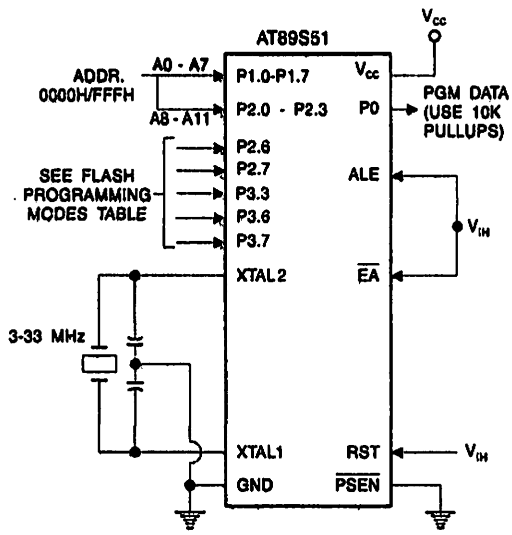


Figure 17-2. Verifying the Flash Memory (Parallel Mode)

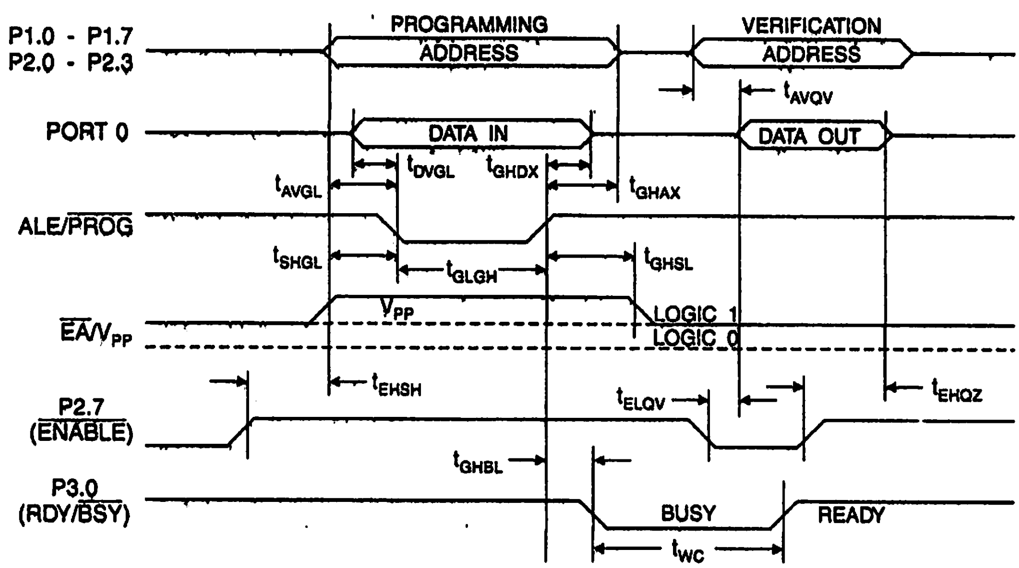


8. Flash Programming and Verification Characteristics (Parallel Mode)

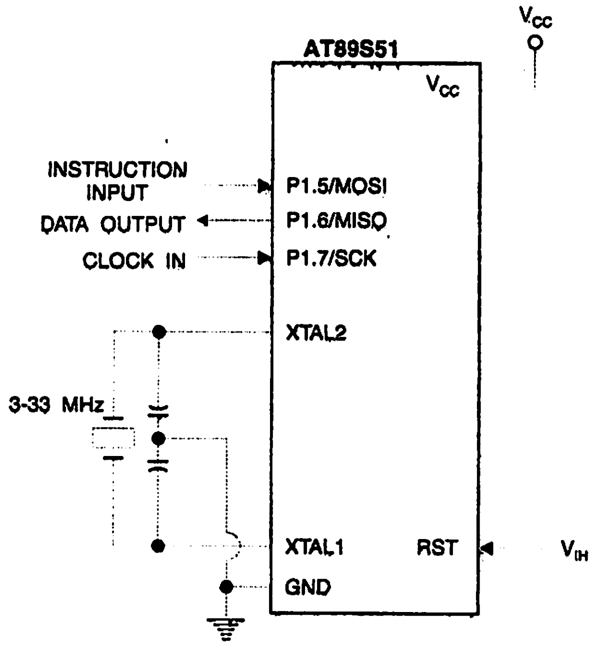
$_A$ = 20°C to 30°C, V_{CC} = 4.5 to 5.5V

Symbol	Parameter	Min	Max	Units
V_{PP}	Programming Supply Voltage	11.5	12.5	V
$_{PP}$	Programming Supply Current		10	mA
$_{CC}$	V_{CC} Supply Current		30	mA
$1/t_{CLCL}$	Oscillator Frequency	3	33	MHz
$_{AVGL}$	Address Setup to $\overline{PROG}$ Low	48 t_{CLCL}		
$_{GHAX}$	Address Hold After $\overline{PROG}$	48 t_{CLCL}		
$_{DVGL}$	Data Setup to $\overline{PROG}$ Low	48 t_{CLCL}		
$_{GHDX}$	Data Hold After $\overline{PROG}$	48 t_{CLCL}		
$_{EHSH}$	P2.7 ($\overline{ENABLE}$) High to V_{PP}	48 t_{CLCL}		
$_{SHGL}$	V_{PP} Setup to $\overline{PROG}$ Low	10		μs
$_{GHSL}$	V_{PP} Hold After $\overline{PROG}$	10		μs
$_{GLGH}$	$\overline{PROG}$ Width	0.2	1	μs
$_{AVQV}$	Address to Data Valid		$48t_{CLCL}$	
$_{ELQV}$	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
$_{EHQZ}$	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
$_{GHBL}$	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	μs
$_{WC}$	Byte Write Cycle Time		50	μs

gure 18-1. Flash Programming and Verification Waveforms – Parallel Mode

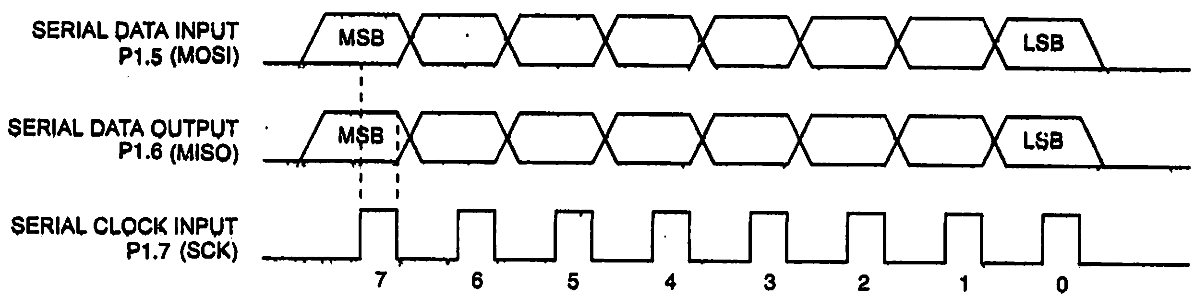


gure 18-2. Flash Memory Serial Downloading



9. Flash Programming and Verification Waveforms – Serial Mode

gure 19-1. Serial Programming Waveforms



20. Serial Programming Instruction Set

Instruction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte 4	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output on MISO)	Enable Serial Programming while RST is high
Chip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
Read Program Memory (Byte 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
Write Program Memory (Byte 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
Write Lock Bits[1]	1010 1100	1110 00B1B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (1).
Read Lock Bits	0010 0100	xxxx xxxx	xxxx xxxx	xxx LB3 LB2 LB1 xx	Read back current status of the lock bits (a programmed lock bit reads back as a "1")
Read Signature Bytes	0010 1000	xxxx A11 A10 A9 A8	A7 xxx xxx0	Signature Byte	Read Signature Byte
Read Program Memory (Page 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)
Write Program Memory (Page 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)

ote: 1. 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, B2 = 1 → Mode 4, lock bit 3 activated

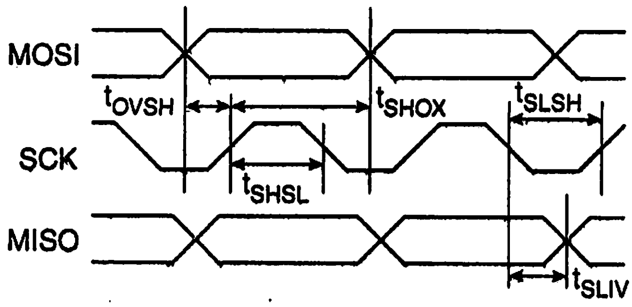
} Each of the lock bit modes need to be activated sequentially before Mode 4 can be executed.

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

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

1. Serial Programming Characteristics

igure 21-1. Serial Programming Timing



able 21-1. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
1/t_{CLCL}	Oscillator Frequency	3		33	MHz
CLCL	Oscillator Period	30			ns
SHSL	SCK Pulse Width High	8 t_{CLCL}			ns
SLSH	SCK Pulse Width Low	8 t_{CLCL}			ns
OVSH	MOSI Setup to SCK High	t_{CLCL}			ns
SHOX	MOSI Hold after SCK High	2 t_{CLCL}			ns
SLIV	SCK Low to MISO Valid	10	16	32	ns
ERASE	Chip Erase Instruction Cycle Time			500	ms
SWC	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	μs

2. Absolute Maximum Ratings*

)perating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
/oltage on Any Pin vith Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
)C 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.

3. DC Characteristics

he values 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.

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
V_{IL1}	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
V_{IH}	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
V_{IH1}	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
V_{OL}	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	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
V_{OH1}	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
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	μA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-300	μA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	μA
RRST	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	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

otes: 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.

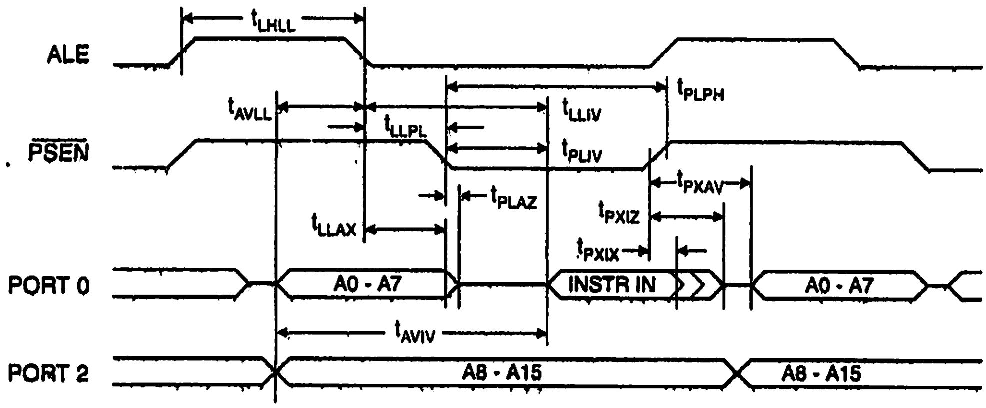
4. AC Characteristics

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

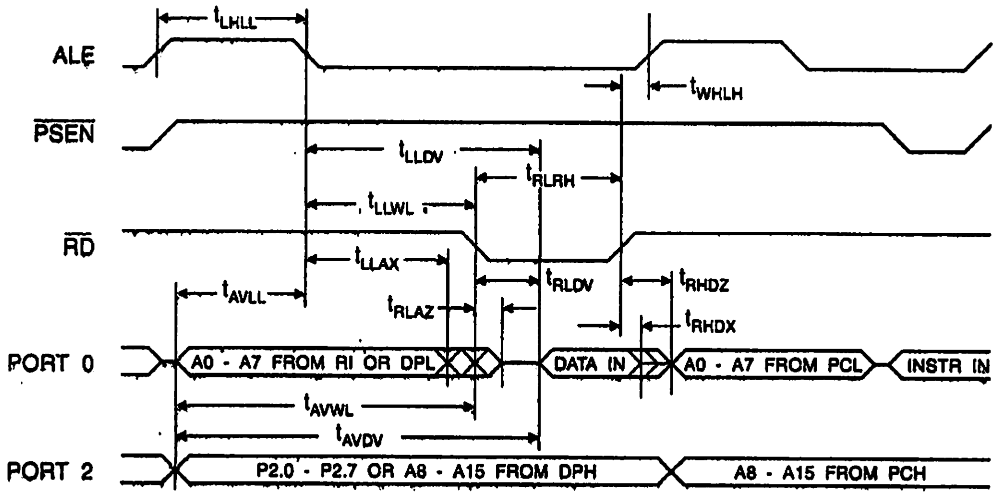
.1 External Program and Data Memory Characteristics

ymbol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
$/t_{CLCL}$	Oscillator Frequency			0	33	MHz
HLL	ALE Pulse Width	127		$2\ t_{CLCL}$-40		ns
VLL	Address Valid to ALE Low	43		t_{CLCL}-25		ns
LAX	Address Hold After ALE Low	48		t_{CLCL}-25		ns
LIV	ALE Low to Valid Instruction In		233		$4\ t_{CLCL}$-65	ns
LPL	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-25		ns
LPH	$\overline{PSEN}$ Pulse Width	205		$3\ t_{CLCL}$-45		ns
LIV	$\overline{PSEN}$ Low to Valid Instruction In		145		$3\ t_{CLCL}$-60	ns
XIX	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
XIZ	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-25	ns
XAV	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
VIV	Address to Valid Instruction In		312		$5\ t_{CLCL}$-80	ns
LAZ	$\overline{PSEN}$ Low to Address Float		10		10	ns
LRH	$\overline{RD}$ Pulse Width	400		$6\ t_{CLCL}$-100		ns
LWH	$\overline{WR}$ Pulse Width	400		$6\ t_{CLCL}$-100		ns
LDV	$\overline{RD}$ Low to Valid Data In		252		$5\ t_{CLCL}$-90	ns
HDX	Data Hold After $\overline{RD}$	0		0		ns
HDZ	Data Float After $\overline{RD}$		97		$2\ t_{CLCL}$-28	ns
LDV	ALE Low to Valid Data In		517		$8\ t_{CLCL}$-150	ns
VDV	Address to Valid Data In		585		$9\ t_{CLCL}$-165	ns
LWL	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3\ t_{CLCL}$-50	$3\ t_{CLCL}$+50	ns
VWL	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4\ t_{CLCL}$-75		ns
VWX	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-30		ns
VWH	Data Valid to $\overline{WR}$ High	433		$7\ t_{CLCL}$-130		ns
HQX	Data Hold After $\overline{WR}$	33		t_{CLCL}-25		ns
LAZ	$\overline{RD}$ Low to Address Float		0		0	ns
HLH	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-25	t_{CLCL}+25	ns

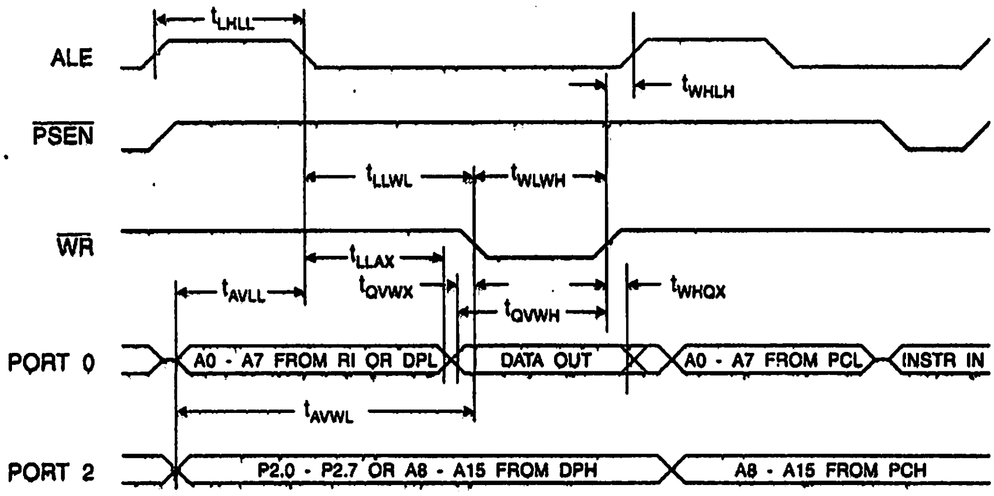
5. External Program Memory Read Cycle



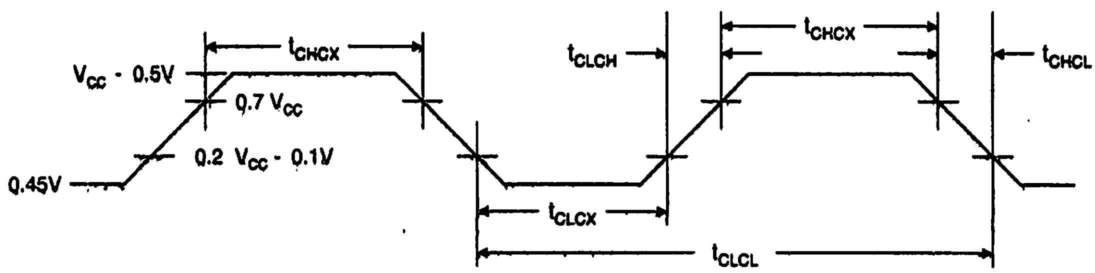
6. External Data Memory Read Cycle



7. External Data Memory Write Cycle



8. External Clock Drive Waveforms



9. External Clock Drive

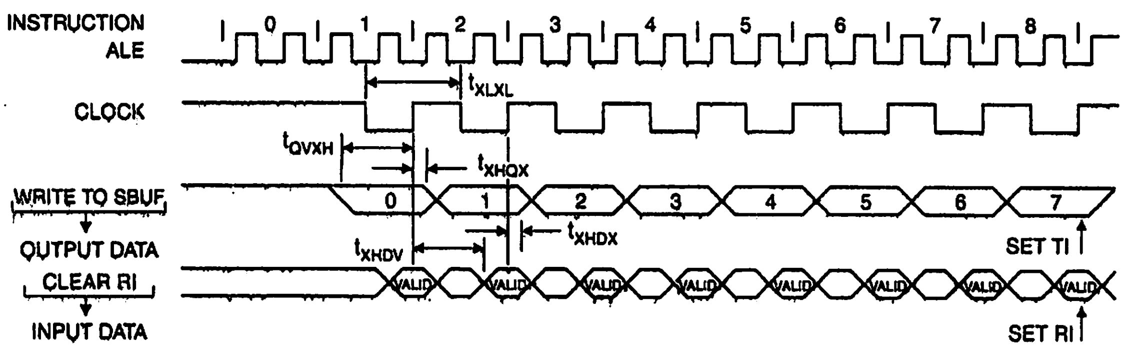
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	33	MHz
t_{CLCL}	Clock Period	30		ns
t_{CHCX}	High Time	12		ns
t_{CLCX}	Low Time	12		ns
t_{CLCH}	Rise Time		5	ns
t_{CHCL}	Fall Time		5	ns

). Serial Port Timing: Shift Register Mode Test Conditions

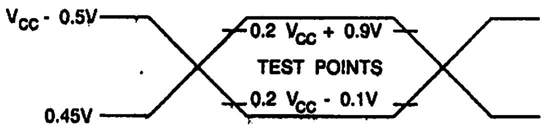
e values in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

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

I. Shift Register Mode Timing Waveforms

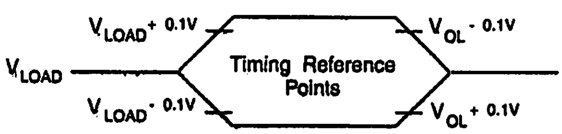


2. AC Testing Input/Output Waveforms(1)



te: 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.

3. Float Waveforms(1)



te: 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.

24. Ordering Information

24.1 Standard Package

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC AT89S51-24JC AT89S51-24PC AT89S51-24SC	44A 44J 40P6 42PS6	Commercial (0°C to 70°C)
		AT89S51-24AI AT89S51-24JI AT89S51-24PI AT89S51-24SI	44A 44J 40P6 42PS6	Industrial (-40°C to 85°C)
33	4.5V to 5.5V	AT89S51-33AC AT89S51-33JC AT89S51-33PC AT89S51-33SC	44A 44J 40P6 42PS6	Commercial (0°C to 70°C)

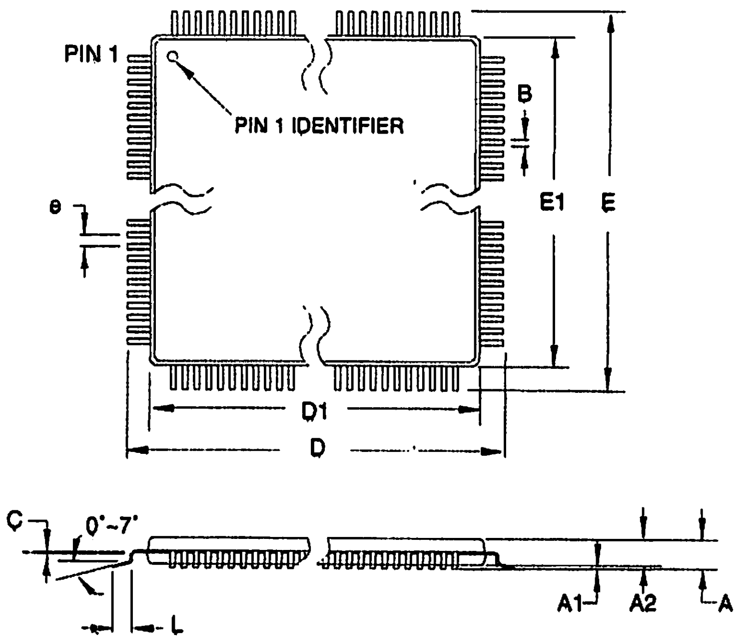
24.2 Green Package Option (Pb/Halide-free)

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AU AT89S51-24JU AT89S51-24PU	44A 44J 40P6	Industrial (-40°C to 85°C)

Package Type	
44A	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
44J	44-lead, Plastic J-leaded Chip Carrier (PLCC)
40P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
42PS6	42-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)

. Packaging Information

.1 44A – TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

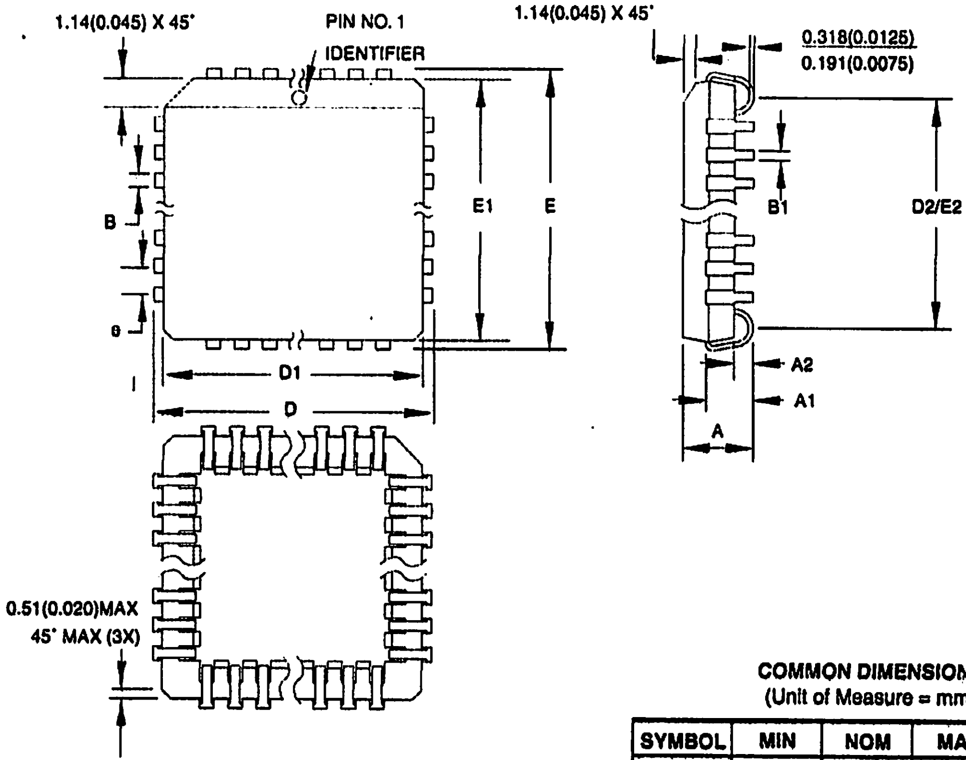
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation ACB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

10/5/2001

ATMEL 2325 Orchard Parkway San Jose, CA 95131	TITLE 44A, 44-lead, 10 x 10 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	DRAWING NO. 44A	REV. B

.2 44J – PLCC



COMMON DIMENSIONS
(Unit of Measure = mm)

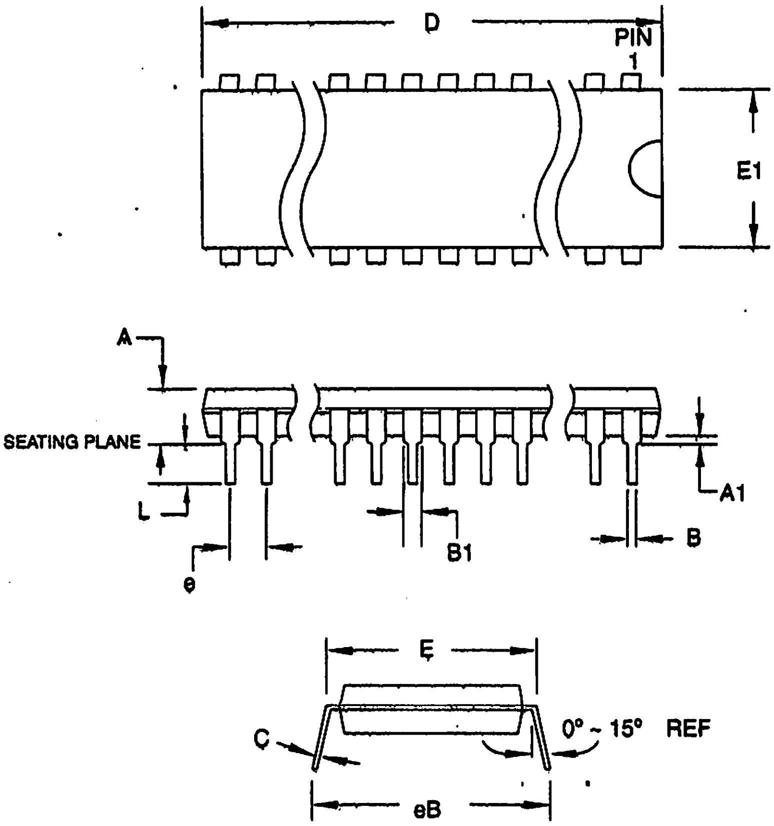
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

Notes:
1. This package conforms to JEDEC reference MS-018, Variation AC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is .010"(0.254 mm) per side. Dimension D1 and E1 include mold mismatch and are measured at the extreme material condition at the upper or lower parting line.
3. Lead coplanarity is 0.004" (0.102 mm) maximum.

10/04/01

	TITLE	DRAWING NO.	REV.
ATMEL 2325 Orchard Parkway San Jose, CA 95131	44J, 44-lead, Plastic J-leaded Chip Carrier (PLCC)	44J	B

40P6 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

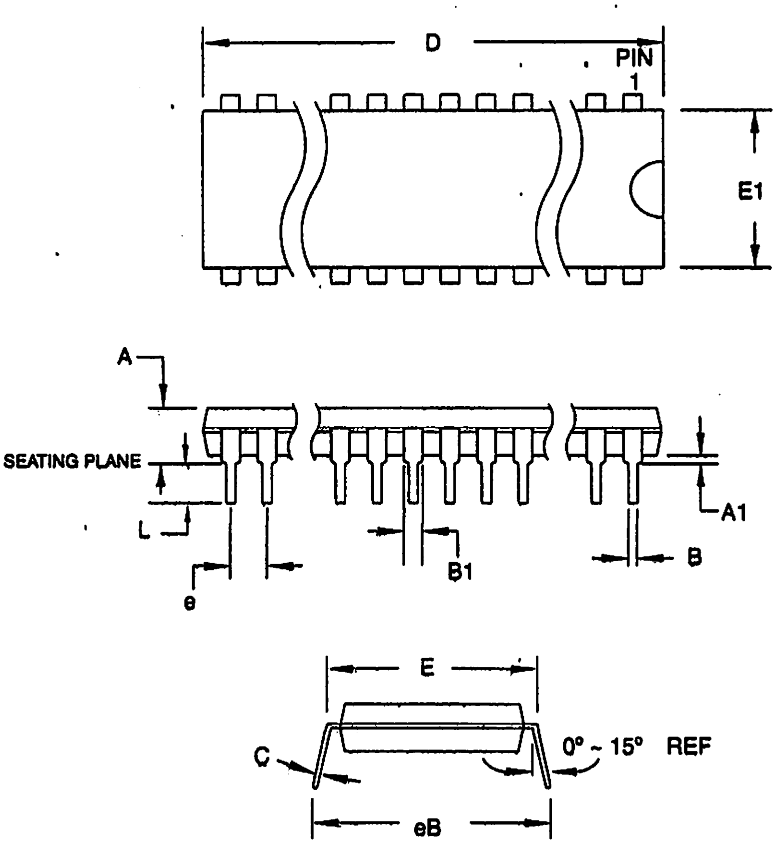
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

Notes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

09/28/01

	TITLE	DRAWING NO.	REV.
ATMEL 2325 Orchard Parkway San Jose, CA 95131	40P6, 40-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	40P6	B

4 42PS6 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.83	
A1	0.51	–	–	
D	36.70	–	36.96	Note 2
E	15.24	–	15.88	
E1	13.46	–	13.97	Note 2
B	0.38	–	0.56	
B1	0.76	–	1.27	
L	3.05	–	3.43	
C	0.20	–	0.30	
eB	–	–	18.55	
e	1.78 TYP			

Notes.
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

11/6/03

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	42PS6, 42-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	42PS6	A

tmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

egional Headquarters

rope

Atmel Sarl
Route ues Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

sia

Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

pan

9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel Operations

Memory

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards

Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive

Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

*Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom*

Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

2487C–MICRO–03/05 /xM

Interfacing the Serial / RS232 Port V5.0

sclaimer : While every effort has been made to make sure the information in this document is correct, the author can not be liable for y damages whatsoever for loss relating to this document. Use this information at your own risk.

Table of Contents

ntroduction

The Serial Port is harder to interface than the Parallel Port. In most cases, any device you connect to the serial port will need the serial transmission converted back to parallel so that it can be used. This can be done using a UART. On the software side of things, there are many more registers that you have to attend to than on a Standard Parallel Port. (SPP)

So what are the advantages of using serial data transfer rather than parallel?

1. Serial Cables can be longer than Parallel cables. The serial port transmits a '1' as -3 to -25 volts and a '0' as +3 to +25 volts where as a parallel port transmits a '0' as 0v and a '1' as 5v. Therefore the serial port can have a maximum swing of 50V compared to the parallel port which has a maximum swing of 5 Volts. Therefore cable loss is not going to be as much of a problem for serial cables than they are for parallel.

2. You don't need as many wires than parallel transmission. If your device needs to be mounted a far distance away from the computer then 3 core cable (Null Modem Configuration) is going to be a lot cheaper that running 19 or 25 core cable. However you must take into account the cost of the interfacing at each end.

3. Infra Red devices have proven quite popular recently. You may of seen many electronic diaries and palmtop computers which have infra red capabilities build in. However could you imagine transmitting 8 bits of data at the one time across the room and being able to (from the devices point of view) decipher which bits are which? Therefore serial transmission is used where one bit is sent at a time. IrDA-1 (The first infra red specifications) was capable of 115.2k baud and was interfaced into a UART. The pulse length however was cut down to 3/16th of a RS232 bit length to conserve power considering these devices are mainly used on diaries, laptops and palmtops.

4. Microcontroller's have also proven to be quite popular recently. Many of these have in built SCI (Serial Communications Interfaces) which can be used to talk to the outside world. Serial Communication reduces the pin count of these MPU's. Only two pins are commonly used, Transmit Data (TXD) and Receive Data (RXD) compared with at least 8 pins if you use a 8 bit Parallel method (You may also require a Strobe).

'art One : Hardware (PC's)

Hardware Properties

Devices which use serial cables for their communication are split into two categories. These are DCE (Data Communications Equipment) and DTE (Data Terminal Equipment.) Data Communications Equipment are devices such as your modem, TA adapter, plotter etc while Data Terminal Equipment is your Computer or Terminal.

The electrical specifications of the serial port is contained in the EIA (Electronics Industry Association) RS232C standard. It states many parameters such as -

1. A "Space" (logic 0) will be between +3 and +25 Volts.
2. A "Mark" (Logic 1) will be between -3 and -25 Volts.
3. The region between +3 and -3 volts is undefined.
4. An open circuit voltage should never exceed 25 volts. (In Reference to GND)
5. A short circuit current should not exceed 500mA. The driver should be able to handle this without damage. (Take note of this one!)

Above is no where near a complete list of the EIA standard. Line Capacitance, Maximum Baud Rates etc are also included. For more information please consult the EIA RS232-E standard. It is interesting to note however, that the RS232C standard specifies a maximum baud rate of 20,000 BPS!, which is rather slow by today's standards. Revised standards, EIA-232D & EIA-232E were released, in 1987 & 1991 respectively.

Serial Ports come in two "sizes", There are the D-Type 25 pin connector and the D-Type 9 pin connector both of which are male on the back of the PC, thus you will require a female connector on your device. Below is a table of pin connections for the 9 pin and 25 pin D-Type connectors.

Serial Pinouts (D25 and D9 Connectors)

D-Type-25 Pin No.	D-Type-9 Pin No.	Abbreviation	Full Name
Pin 2	Pin 3	TD	Transmit Data
Pin 3	Pin 2	RD	Receive Data
Pin 4	Pin 7	RTS	Request To Send
Pin 5	Pin 8	CTS	Clear To Send

Pin 6	Pin 6	DSR	Data Set Ready
Pin 7	Pin 5	SG	Signal Ground
Pin 8	Pin 1	CD	Carrier Detect
Pin 20	Pin 4	DTR	Data Terminal Ready
Pin 22	Pin 9	RI	Ring Indicator

Table 1 : D Type 9 Pin and D Type 25 Pin Connectors

Pin Functions

Abbreviation	Full Name	Function
TD	Transmit Data	Serial Data Output (TXD)
RD	Receive Data	Serial Data Input (RXD)
CTS	Clear to Send	This line indicates that the Modem is ready to exchange data.
DCD	Data Carrier Detect	When the modem detects a "Carrier" from the modem at the other end of the phone line, this Line becomes active.
DSR	Data Set Ready	This tells the UART that the modem is ready to establish a link.
DTR	Data Terminal Ready	This is the opposite to DSR. This tells the Modem that the UART is ready to link.
RTS	Request To Send	This line informs the Modem that the UART is ready to exchange data.
RI	Ring Indicator	Goes active when modem detects a ringing signal from the PSTN.

Null Modems

A Null Modem is used to connect two DTE's together. This is commonly used as a cheap way to network games or to transfer files between computers using Zmodem Protocol, Xmodem Protocol etc. This can also be used with many Microprocessor Development Systems.

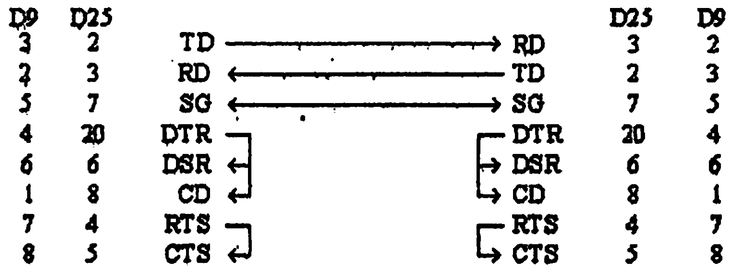


Figure 1 : Null Modem Wiring Diagram

Above is my preferred method of wiring a Null Modem. It only requires 3 wires (TD, RD & SG) to be wired straight through thus is more cost effective to use with long cable runs. The theory of operation is reasonably easy. The aim is to make to computer think it is talking to a modem rather than another computer. Any data transmitted from the first computer must be received by the second thus TD is connected to RD. The second computer must have the same set-up thus RD is connected to TD. Signal Ground (SG) must also be connected so both grounds are common to each computer.

The Data Terminal Ready is looped back to Data Set Ready and Carrier Detect on both computers. When the Data Terminal Ready is asserted active, then the Data Set Ready and Carrier Detect immediately become active. At this point the computer thinks the Virtual Modem to which it is connected is ready and has detected the carrier of the other modem.

All left to worry about now is the Request to Send and Clear To Send. As both computers communicate together at the same speed, flow control is not needed thus these two lines are also linked together on each computer. When the computer wishes to send data, it asserts the Request to Send high and as it's hooked together with the Clear to Send, It immediately gets a reply that it is ok to send and does so.

Notice that the ring indicator is not connected to anything of each end. This line is only used to tell the computer that there is a ringing signal on the phone line. As we don't have a modem connected to the phone line this is left disconnected.

LoopBack Plug

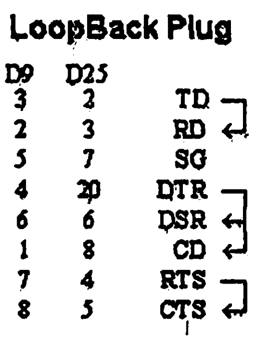


Figure 2 : Loopback Plug Wiring Diagram

This loopback plug can come in extremely handy when writing Serial / RS232 Communications Programs. It has the receive and transmit lines connected together, so that anything transmitted out of the Serial Port is immediately received by the same port. If you connect this to a Serial Port an load a Terminal Program, anything you type will be immediately displayed on the screen. This can be used with the examples later in this tutorial.

Please note that this is not intended for use with Diagnostic Programs and thus will probably not work. For these programs you require a differently wired Loop Back plug which may vary from program to program.

DTE / DCE Speeds

We have already talked briefly about DTE & DCE. A typical Data Terminal Device is a computer and a typical Data Communications Device is a Modem. Often people will talk about DTE to DCE or DCE to DCE speeds. DTE to DCE is the speed between your modem and computer, sometimes referred to as your terminal speed. This should run at faster speeds than the DCE to DCE speed. DCE to DCE is the link between modems, sometimes called the line speed.

Most people today will have 28.8K or 33.6K modems. Therefore we should expect the DCE to DCE speed to be either 28.8K or 33.6K. Considering the high speed of the modem we should expect the DTE to DCE speed to be about 115,200 BPS.(Maximum Speed of the 16550a UART) This is where some people often fall into a trap. The communications program which they use have settings for DCE to DTE speeds. However they see 9.6 KBPS, 14.4 KBPS etc and think it is your modem speed.

Today's Modems should have Data Compression build into them. This is very much like PK-ZIP but the software in your modem compresses and decompresses the data. When set up correctly you can expect compression ratios of 1:4 or even higher. 1 to 4 compression would be typical of a text file. If we were transferring that text file at 28.8K (DCE-DCE), then when the modem compresses it you are actually transferring 115.2 KBPS between computers and thus have a DCE-DTE speed of 115.2 KBPS. Thus this is why the DCE-DTE should be much higher than your modem's connection speed.

Some modem manufacturers quote a maximum compression ratio as 1:8. Lets say for example its on a new 33.6 KBPS modem then we may get a maximum 268,800 BPS transfer between modem and UART. If you only have a 16550a which can do 115,200 BPS tops, then you would be missing out on a extra bit of performance. Buying a 16C650 should fix your problem with a maximum transfer rate of 230,400 BPS.

However don't abuse your modem if you don't get these rates. These are MAXIMUM compression ratios. In some instances if you try to send a already compressed file, your modem can spend more time trying the compress it, thus you get a transmission speed less than your modem's connection speed. If this occurs try turning off your data compression. This should be fixed on newer modems. Some files compress easier than others thus any file which compresses easier is naturally going to have a higher compression ratio.

Flow Control

So if our DTE to DCE speed is several times faster than our DCE to DCE speed the PC can send data to your modem at 115,200 BPS. Sooner or later data is going to get lost as buffers overflow, thus flow control is used. Flow control has two basic varieties, Hardware or Software.

Software flow control, sometimes expressed as Xon/Xoff uses two characters Xon and Xoff. Xon is normally indicated by the ASCII 17 character where as the ASCII 19 character is used for Xoff. The modem will only have a small buffer so when the computer fills it up the modem sends a Xoff character to tell the computer to stop sending data. Once the modem has room for more data it then sends a Xon character and the computer sends more data. This type of flow control has the advantage that it doesn't require any more wires as the characters are sent via the TD/RD lines. However on slow links each character requires 10 bits which can slow communications down.

Hardware flow control is also known as RTS/CTS flow control. It uses two wires in your serial cable rather than extra characters transmitted in your data lines. Thus hardware flow control will not slow down transmission times like Xon-Xoff does. When the computer wishes to send data it takes active the Request to Send line. If the modem has room for this data, then the modem will reply by taking active the Clear to Send line and the computer starts sending data. If the modem does not have the room then it will not send a Clear to Send.

The UART (8250 and Compatibles)

UART stands for Universal Asynchronous Receiver / Transmitter. Its the little box of tricks found on your serial card which plays the little games with your modem or other connected devices. Most cards will have the UART's integrated into other chips which may also control your parallel port, games port, floppy or hard disk drives and are typically surface mount devices. The 8250 series, which includes the 16450, 16550, 16650, & 16750 UARTS are the most commonly found type in your PC. Later we will look at other types which can be used in your homemade devices and projects.

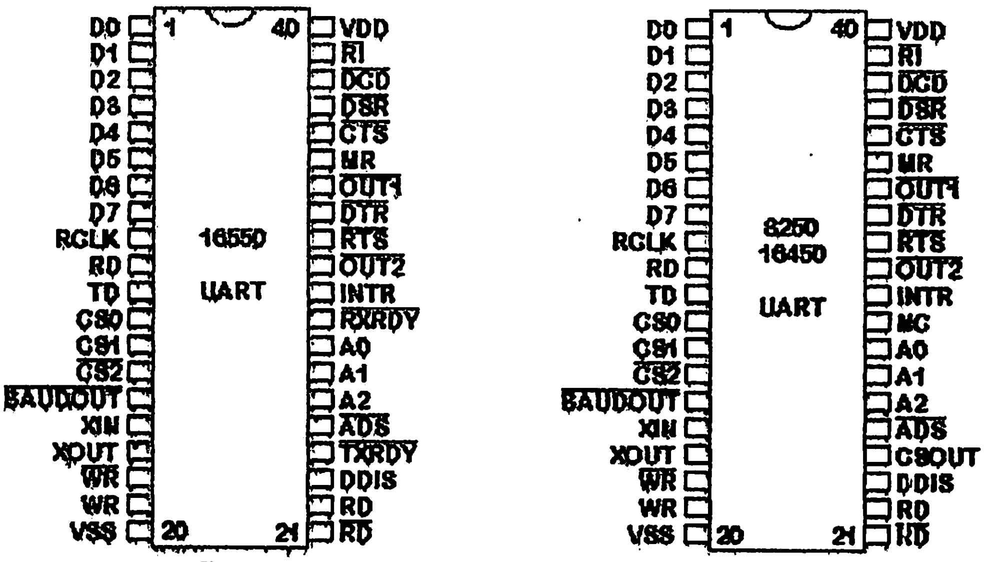


Figure 3 : Pin Diagrams for 16550, 16450 & 8250 UARTs

The 16550 is chip compatible with the 8250 & 16450. The only two differences are pins 24 & 29. On the 8250 Pin 24 was chip select out which functioned only as a indicator to if the chip was active or not. Pin 29 was not connected on the 8250/16450 UARTs. The 16550 introduced two new pins in their place. These are Transmit Ready and Receive Ready which can be implemented with DMA (Direct Memory Access). These Pins have two different modes of operation. Mode 0 supports single transfer DMA where as Mode 1 supports Multi-transfer DMA.

Mode 0 is also called the 16450 mode. This mode is selected when the FIFO buffers are disabled via Bit 0 of the FIFO Control Register or When the FIFO buffers are enabled but DMA Mode Select = 0. (Bit 3 of FCR) In this mode RXRDY is active low when at least one character (Byte) is present in the Receiver Buffer. RXRDY will go inactive high when no more characters are left in the Receiver Buffer. TXRDY will be active low when there are no characters in the Transmit Buffer. It will go inactive high after the first character / byte is loaded into the Transmit Buffer.

Mode 1 is when the FIFO buffers are active and the DMA Mode Select = 1. In Mode 1, RXRDY will go active low when the trigger level is reached or when 16550 Time Out occurs and will return to inactive state when no more characters are left in the FIFO. TXRDY will be active when no characters are present in the Transmit Buffer and will go inactive when the FIFO Transmit Buffer is completely Full.

All the UARTs pins are TTL compatible. That includes TD, RD, RI, DCD, DSR, CTS, DTR and RTS which all interface into your serial plug, typically a D-type connector. Therefore RS232 Level Converters (which we talk about in detail later) are used. These are commonly the DS1489 Receiver and the DS1488 as the PC has +12 and -12 volt rails which can be used by these devices. The RS232 Converters will convert the TTL signal into RS232 Logic Levels.

Pin No.	Name	Notes
Pin 1:8	D0:D7	Data Bus
Pin 9	RCLK	Receiver Clock Input. The frequency of this input should equal the receivers baud rate * 16
Pin 10	RD	Receive Data
Pin 11	TD	Transmit Data
Pin 12	CS0	Chip Select 0 - Active High
Pin 13	CS1	Chip Select 1 - Active High
Pin 14	nCS2	Chip Select 2 - Active Low
Pin 15	nBAUDOUT	Baud Output - Output from Programmable Baud Rate Generator. Frequency = (Baud Rate x 16)
Pin 16	XIN	External Crystal Input - Used for Baud Rate Generator Oscillator
Pin 17	XOUT	External Crystal Output
Pin 18	nWR	Write Line - Inverted
Pin 19	WR	Write Line - Not Inverted
Pin 20	VSS	Connected to Common Ground
Pin 21	RD	Read Line - Inverted
Pin 22	nRD	Read Line - Not Inverted
Pin 23	DDIS	Driver Disable. This pin goes low when CPU is reading from UART. Can be connected to Bus Transceiver in case of high capacity data bus.
Pin 24	nTXRDY	Transmit Ready
Pin 25	nADS	Address Strobe. Used if signals are not stable during read or write cycle
Pin 26	A2	Address Bit 2
Pin 27	A1	Address Bit 1
Pin 28	A0	Address Bit 0

Pin 29	nRXRDY	Receive Ready
Pin 30	INTR	Interrupt Output
Pin 31	nOUT2	User Output 2
Pin 32	nRTS	Request to Send
Pin 33	nDTR	Data Terminal Ready
Pin 34	nOUT1	User Output 1
Pin 35	MR	Master Reset
Pin 36	nCTS	Clear To Send
Pin 37	nDSR	Data Set Ready
Pin 38	nDCD	Data Carrier Detect
Pin 39	nRI	Ring Indicator
Pin 40	VDD	+ 5 Volts

Table 2 : Pin Assignments for 16550A UART

The UART requires a Clock to run. If you look at your serial card a common crystal found is either a 1.8432 MHZ or a 18.432 MHZ Crystal. The crystal in connected to the XIN-XOUT pins of the UART using a few extra components which help the crystal to start oscillating. This clock will be used for the Programmable Baud Rate Generator which directly interfaces into the transmit timing circuits but not directly into the receiver timing circuits. For this an external connection mast be made from pin 15 (BaudOut) to pin 9 (Receiver clock in.) Note that the clock signal will be at Baudrate * 16.

If you are serious about pursuing the 16550 UART used in your PC further, then would suggest downloading a copy of the PC16550D data sheet from National Semiconductor, (http://www.natsemi.com) Data sheets are available in .PDF format so you will need Adobe Acrobat Reader to read these. Texas Instruments (http://www.ti.com) has released the 16750 UART which has 64 Byte FIFO's. Data Sheets for the TL16C750 are available from the Texas Instruments Site.

Types of UARTS (For PC's)

8250	First UART in this series. It contains no scratch register. The 8250A was an improved version of the 8250 which operates faster on the bus side.
8250A	This UART is faster than the 8250 on the bus side. Looks exactly the same to software than 16450.
8250B	Very similar to that of the 8250 UART.
16450	Used in AT's (Improved bus speed over 8250's). Operates comfortably at 38.4KBPS. Still quite common today.

16550	This was the first generation of buffered UART. It has a 16 byte buffer, however it doesn't work and is replaced with the 16550A.
16550A	Is the most common UART use for high speed communications eg 14.4K & 28.8K Modems. They made sure the FIFO buffers worked on this UART.
16650	Very recent breed of UART. Contains a 32 byte FIFO, Programmable X-On / X-Off characters and supports power management.
16750	Produced by Texas Instruments. Contains a 64 byte FIFO.

ʾart Two : Serial Port's Registers (PC's)

Port Addresses & IRQ's

Name	Address	IRQ
COM 1	3F8	4
COM 2	2F8	3
COM 3	3E8	4
COM 4	2E8	3

Table 3 : Standard Port Addresses

Above is the standard port addresses. These should work for most P.C's. If you just happen to be lucky enough to own a IBM P/S2 which has a micro-channel bus, then expect a different set of addresses and IRQ's. Just like the LPT ports, the base addresses for the COM ports can be read from the BIOS Data Area.

Start Address	Function
0000:0400	COM1's Base Address
0000:0402	COM2's Base Address
0000:0404	COM3's Base Address
0000:0406	COM4's Base Address

Table 4 - COM Port Addresses in the BIOS Data Area;

The above table shows the address at which we can find the Communications (COM) ports addresses in the BIOS Data Area. Each address will take up 2 bytes. The following sample program in C, shows how you can read these locations to obtain the addresses of your communications ports.

```
#include <stdio.h>
#include <dos.h>

void main(void)
{
 unsigned int far *ptraddr;  /* Pointer to location of Port Addresses */
 unsigned int address;       /* Address of Port */
 int a;

 ptraddr=(unsigned int far *)0x00000400;

 for (a = 0; a <  4; a++)
   {
    address = *ptraddr;
    if (address == 0)
                printf("No port found for COM%d \n",a+1);
    else
                printf("Address assigned to COM%d is %Xh\n",a+1,address);
    *ptraddr++;
   }
}
```

Table of Registers

Base Address	DLAB	Read/Write	Abr.	Register Name
+ 0	=0	Write	-	Transmitter Holding Buffer
	=0	Read	-	Receiver Buffer
	=1	Read/Write	-	Divisor Latch Low Byte
+ 1	=0	Read/Write	IER	Interrupt Enable Register
	=1	Read/Write	-	Divisor Latch High Byte
+ 2	-	Read	IIR	Interrupt Identification Register
	-	Write	FCR	FIFO Control Register
+ 3	-	Read/Write	LCR	Line Control Register
+ 4	-	Read/Write	MCR	Modem Control Register
+ 5	-	Read	LSR	Line Status Register
+ 6	-	Read	MSR	Modem Status Register
+ 7	-	Read/Write	-	Scratch Register

Table 5 : Table of Registers

DLAB ?

You will have noticed in the table of registers that there is a DLAB column. When DLAB is set to '0' or '1' some of the registers change. This is how the UART is able to have 12 registers (including the scratch register) through only 8 port addresses. DLAB stands for Divisor Latch Access Bit. When DLAB is set to '1' via the line control register, two registers become available from which you can set your speed of communications measured in bits per second.

The UART will have a crystal which should oscillate around 1.8432 MHZ. The UART incorporates a divide by 16 counter which simply divides the incoming clock signal by 16. Assuming we had the 1.8432 MHZ clock signal, that would leave us with a maximum, 115,200 hertz signal making the UART capable of transmitting and receiving at 115,200 Bits Per Second (BPS). That would be fine for some of the faster modems and devices which can handle that speed, but others just wouldn't communicate at all. Therefore the UART is fitted with a Programmable Baud Rate Generator which is controlled by two registers.

Lets say for example we only wanted to communicate at 2400 BPS. We worked out that we would have to divide 115,200 by 48 to get a workable 2400 Hertz Clock. The "Divisor", in this case 48, is stored in the two registers controlled by the "Divisor Latch Access Bit". This divisor can be any number which can be stored in 16 bits (ie 0 to 65535). The UART only has a 8 bit data bus, thus this is where the two registers are used. The first register (Base + 0) when DLAB = 1 stores the "Divisor latch low byte" where as the second register (base + 1 when DLAB = 1) stores the "Divisor latch high byte."

Below is a table of some more common speeds and their divisor latch high bytes & low bytes. Note that all the divisors are shown in Hexadecimal.

Speed (BPS)	Divisor (Dec)	Divisor Latch High Byte	Divisor Latch Low Byte
50	2304	09h	00h
300	384	01h	80h
600	192	00h	C0h
2400	48	00h	30h
4800	24	00h	18h
9600	12	00h	0Ch
19200	6	00h	06h
38400	3	00h	03h
57600	2	00h	02h
115200	1	00h	01h

Table 6 : Table of Commonly Used Baudrate Divisors

Interrupt Enable Register (IER)

Bit	Notes
Bit 7	Reserved
Bit 6	Reserved
Bit 5	Enables Low Power Mode (16750)
Bit 4	Enables Sleep Mode (16750)
Bit 3	Enable Modem Status Interrupt
Bit 2	Enable Receiver Line Status Interrupt
Bit 1	Enable Transmitter Holding Register Empty Interrupt
Bit 0	Enable Received Data Available Interrupt

Table 7 : Interrupt Enable Register

The Interrupt Enable Register could possibly be one of the easiest registers on a UART to understand. Setting Bit 0 high enables the Received Data Available Interrupt which generates an interrupt when the receiving register/FIFO contains data to be read by the CPU.

Bit 1 enables Transmit Holding Register Empty Interrupt. This interrupts the CPU when the transmitter buffer is empty. Bit 2 enables the receiver line status interrupt. The UART will interrupt when the receiver line status changes. Likewise for bit 3 which enables the modem status interrupt. Bits 4 to 7 are the easy ones. They are simply reserved. (If only everything was that easy!)

Interrupt Identification Register (IIR)

Bit	Notes		
Bits 6 : 7	Bit 6	Bit 7	
	0	0	No FIFO
	0	1	FIFO Enabled but Unusable
	1	1	FIFO Enabled
Bit 5	64 Byte Fifo Enabled (16750 only)		
Bit 4	Reserved		

Bit 3	0	Reserved on 8250, 16450	
	1	16550 Time-out Interrupt Pending	
Bits 1 : 2	Bit 2	Bit 1	
	0	0	Modem Status Interrupt
	0	1	Transmitter Holding Register Empty Interrupt
	1	0	Received Data Available Interrupt
	1	1	Receiver Line Status Interrupt
Bit 0	0	Interrupt Pending	
	1	No Interrupt Pending	

Table 8 : Interrupt Identification Register

The interrupt identification register is a read only register. Bits 6 and 7 give status on the FIFO Buffer. When both bits are '0' no FIFO buffers are active. This should be the only result you will get from a 8250 or 16450. If bit 7 is active but bit 6 is not active then the UART has it's buffers enabled but are unusable. This occurs on the 16550 UART where a bug in the FIFO buffer made the FIFO's unusable. If both bits are '1' then the FIFO buffers are enabled and fully operational.

Bits 4 and 5 are reserved. Bit 3 shows the status of the time-out interrupt on a 16550 or higher.

Lets jump to Bit 0 which shows whether an interrupt has occurred. If an interrupt has occurred it's status will shown by bits 1 and 2. These interrupts work on a priority status. The Line Status Interrupt has the highest Priority, followed by the Data Available Interrupt, then the Transmit Register Empty Interrupt and then the Modem Status Interrupt which has the lowest priority.

First In / First Out Control Register (FCR)

Bit	Notes		
Bits 6 : 7	Bit 7	Bit 6	Interrupt Trigger Level
	0	0	1 Byte
	0	1	4 Bytes
	1	0	8 Bytes
	1	1	14 Bytes
Bit 5	Enable 64 Byte FIFO (16750 only)		
Bit 4	Reserved		

Bit 3	DMA Mode Select. Change status of RXRDY & TXRDY pins from mode 1 to mode 2.
Bit 2	Clear Transmit FIFO
Bit 1	Clear Receive FIFO
Bit 0	Enable FIFO's

Table 9 : FIFO Control Register

The FIFO register is a write only register. This register is used to control the FIFO (First In / First Out) buffers which are found on 16550's and higher.

Bit 0 enables the operation of the receive and transmit FIFO's. Writing a '0' to this bit will disable the operation of transmit and receive FIFO's, thus you will loose all data stored in these FIFO buffers.

Bit's 1 and 2 control the clearing of the transmit or receive FIFO's. Bit 1 is responsible for the receive buffer while bit 2 is responsible for the transmit buffer. Setting these bits to 1 will only clear the contents of the FIFO and will not affect the shift registers. These two bits are self resetting, thus you don't need to set the bits to '0' when finished.

Bit 3 enables the DMA mode select which is found on 16550 UARTs and higher. More on this later. Bits 4 and 5 are those easy type again, Reserved.

Bits 6 and 7 are used to set the triggering level on the Receive FIFO. For example if bit 7 was set to '1' and bit 6 was set to '0' then the trigger level is set to 8 bytes. When there is 8 bytes of data in the receive FIFO then the Received Data Available interrupt is set. See (IIR)

Line Control Register (LCR)

Bit	Notes			
Bit 7	1	Divisor Latch Access Bit		
	0	Access to Receiver buffer, Transmitter buffer & Interrupt Enable Register		
Bit 6	Set Break Enable			
Bits 3 : 5	Bit 5	Bit 4	Bit 3	Parity Select
	X	X	0	No Parity
	0	0	1	Odd Parity
	0	1	1	Even Parity
	1	0	1	High Parity (Sticky)
	1	1	1	Low Parity (Sticky)

Bit 2	Length of Stop Bit		
	0	One Stop Bit	
	1	2 Stop bits for words of length 6,7 or 8 bits or 1.5 Stop Bits for Word lengths of 5 bits.	
Bits 0 : 1	Bit 1	Bit 0	Word Length
	0	0	5 Bits
	0	1	6 Bits
	1	0	7 Bits
	1	1	8 Bits

Table 10 : Line Control Register

The Line Control register sets the basic parameters for communication. Bit 7 is the Divisor Latch Access Bit or DLAB for short. We have already talked about what it does. (See DLAB?) Bit 6 Sets break enable. When active, the TD line goes into "Spacing" state which causes a break in the receiving UART. Setting this bit to '0' Disables the Break.

Bits 3,4 and 5 select parity. If you study the 3 bits, you will find that bit 3 controls parity. That is, if it is set to '0' then no parity is used, but if it is set to '1' then parity is used. Jumping to bit 5, we can see that it controls sticky parity. Sticky parity is simply when the parity bit is always transmitted and checked as a '1' or '0'. This has very little success in checking for errors as if the first 4 bits contain errors but the sticky parity bit contains the appropriately set bit, then a parity error will not result. Sticky high parity is the use of a '1' for the parity bit, while the opposite, sticky low parity is the use of a '0' for the parity bit.

If bit 5 controls sticky parity, then turning this bit off must produce normal parity provided bit 3 is still set to '1'. Odd parity is when the parity bit is transmitted as a '1' or '0' so that there is a odd number of 1's. Even parity must then be the parity bit produces and even number of 1's. This provides better error checking but still is not perfect, thus CRC-32 is often used for software error correction. If one bit happens to be inverted with even or odd parity set, then a parity error will occur, however if two bits are flipped in such a way that it produces the correct parity bit then an parity error will no occur.

Bit 2 sets the length of the stop bits. Setting this bit to '0' will produce one stop bit, however setting it to '1' will produce either 1.5 or 2 stop bits depending upon the word length. Note that the receiver only checks the first stop bit.

Bits 0 and 1 set the word length. This should be pretty straight forward. A word length of 8 bits is most commonly used today.

Modem Control Register (MCR)

Bit	Notes
Bit 7	Reserved
Bit 6	Reserved
Bit 5	Autoflow Control Enabled (16750 only)
Bit 4	LoopBack Mode
Bit 3	Aux Output 2
Bit 2	Aux Output 1
Bit 1	Force Request to Send
Bit 0	Force Data Terminal Ready

Table 11 : Modem Control Register

The Modem Control Register is a Read/Write Register. Bits 5,6 and 7 are reserved. Bit 4 activates the loopback mode. In Loopback mode the transmitter serial output is placed into marking state. The receiver serial input is disconnected. The transmitter out is looped back to the receiver in. DSR, CTS, RI & DCD are disconnected. DTR, RTS, OUT1 & OUT2 are connected to the modem control inputs. The modem control output pins are then place in an inactive state. In this mode any data which is placed in the transmitter registers for output is received by the receiver circuitry on the same chip and is available at the receiver buffer. This can be used to test the UARTs operation.

Aux Output 2 maybe connected to external circuitry which controls the UART-CPU interrupt process. Aux Output 1 is normally disconnected, but on some cards is used to switch between a 1.8432MHZ crystal to a 4MHZ crystal which is used for MIDI. Bits 0 and 1 simply control their relevant data lines. For example setting bit 1 to '1' makes the request to send line active.

Bit	Notes
Bit 7	Error in Received FIFO
Bit 6	Empty Data Holding Registers
Bit 5	Empty Transmitter Holding Register
Bit 4	Break Interrupt
Bit 3	Framing Error
Bit 2	Parity Error
Bit 1	Overrun Error
Bit 0	Data Ready

Table 12 : Line Status Register

The line status register is a read only register. Bit 7 is the error in received FIFO bit. This bit is high when at least one break, parity or framing error has occurred on a byte which is contained in the FIFO.

When bit 6 is set, both the transmitter holding register and the shift register are empty. The UART's holding register holds the next byte of data to be sent in parallel fashion. The shift register is used to convert the byte to serial, so that it can be transmitted over one line. When bit 5 is set, only the transmitter holding register is empty. So what's the difference between the two? When bit 6, the transmitter holding and shift registers are empty, no serial conversions are taking place so there should be no activity on the transmit data line. When bit 5 is set, the transmitter holding register is empty, thus another byte can be sent to the data port, but a serial conversion using the shift register may be taking place.

The break interrupt (Bit 4) occurs when the received data line is held in a logic state '0' (Space) for more than the time it takes to send a full word. That includes the time for the start bit, data bits, parity bits and stop bits.

A framing error (Bit 3) occurs when the last bit is not a stop bit. This may occur due to a timing error. You will most commonly encounter a framing error when using a null modem linking two computers or a protocol analyzer when the speed at which the data is being sent is different to that of what you have the UART set to receive it at.

A overrun error normally occurs when your program can't read from the port fast enough. If you don't get an incoming byte out of the register fast enough, and another byte just happens to be received, then the last byte will be lost and a overrun error will result.

Bit 0 shows data ready, which means that a byte has been received by the UART and is at the receiver buffer ready to be read.

Modem Status Register (MSR)

Bit	Notes
Bit 7	Carrier Detect
Bit 6	Ring Indicator
Bit 5	Data Set Ready
Bit 4	Clear To Send
Bit 3	Delta Data Carrier Detect
Bit 2	Trailing Edge Ring Indicator
Bit 1	Delta Data Set Ready
Bit 0	Delta Clear to Send

Table 13 : Modem Status Register

Bit 0 of the modem status register shows delta clear to send, delta meaning a change in, thus delta clear to send means that there was a change in the clear to send line, since the last read of this register. This is the same for bits 1 and 3. Bit 1 shows a change in the Data Set Ready line where as Bit 3 shows a change in the Data Carrier Detect line. Bit 2 is the Trailing Edge Ring Indicator which indicates that there was a transformation from low to high state on the Ring Indicator line.

Bits 4 to 7 show the current state of the data lines when read. Bit 7 shows Carrier Detect, Bit 6 shows Ring Indicator, Bit 5 shows Data Set Ready & Bit 4 shows the status of the Clear To Send line.

Scratch Register

The scratch register is not used for communications but rather used as a place to leave a byte of data. The only real use it has is to determine whether the UART is a 8250/8250B or a 8250A/16450 and even that is not very practical today as the 8250/8250B was never designed for AT's and can't hack the bus speed.

Part 3 : Programming (PC's)

Polling or Interrupt Driven?

When writing a communications program you have two methods available to you. You can poll the UART, to see if any new data is available or you can set up an interrupt handler to remove the data from the UART when it generates a interrupt. Polling the UART is a lot slower method, which is very CPU intensive thus can only have a maximum speed of around 34.8 KBPS before you start losing data. Some newer Pentium Pro's may be able to achieve better rates that this. The other option is using a Interrupt handler, and that's what we have used here. It will very easily support 115.2K BPS, even on low end computers.

Termpoll.c - A sample Comms Program using Polling

```
/* Name       : Sample Comm's Program - Polled Version - termpoll.c     */
/* Written By : Craig Peacock <cpeacock@senet.com.au>                   */
/* Date       : Saturday 22nd February 1997                             */
/*        Copyright 1997 CRAIG PEACOCK <cpeacock@senet.com.au>          */
/*         See http://www.senet.com.au/~cpeacock/serial.htm             */
/*                        For More Information                          */

#include <dos.h>
#include <stdio.h>
#include <conio.h>

#define PORT1 0x3F8
/* Defines Serial Ports Base Address */
/* COM1 0x3F8                        */
/* COM2 0x2F8                          */
/* COM3 0x3E8                          */
/* COM4 0x2E8                          */

void main(void)
{
 int c;
 int ch;
 outportb(PORT1 + 1 , 0);    /* Turn off interrupts - Port1 */
```

```
/*          PORT 1 - Communication Settings         */

outportb(PORT1 + 3 , 0x80);  /* SET DLAB ON */
outportb(PORT1 + 0 , 0x03);  /* Set Baud rate - Divisor Latch Low Byte */
                             /* Default 0x03 =  38,400 BPS */
                             /*         0x01 = 115,200 BPS */
                             /*         0x02 =  56,700 BPS */
                             /*         0x06 =  19,200 BPS */
                             /*         0x0C =   9,600 BPS */
                             /*         0x18 =   4,800 BPS */
                             /*         0x30 =   2,400 BPS */
outportb(PORT1 + 1 , 0x00);  /* Set Baud rate - Divisor Latch High Byte */
outportb(PORT1 + 3 , 0x03);  /* 8 Bits, No Parity, 1 Stop Bit */
outportb(PORT1 + 2 , 0xC7);  /* FIFO Control Register */
outportb(PORT1 + 4 , 0x0B);  /* Turn on DTR, RTS, and OUT2 */

printf("\nSample Comm's Program. Press ESC to quit \n");

do { c = inportb(PORT1 + 5);          /* Check to see if char has been */
                                      /* received.                     */
     if (c & 1) {ch = inportb(PORT1); /* If so, then get Char          */
                 printf("%c",ch);}    /* Print Char to Screen          */

     if (kbhit()){ch = getch();         /* If key pressed, get Char */
                  outportb(PORT1, ch);} /* Send Char to Serial Port */

   } while (ch !=27); /* Quit when ESC (ASC 27) is pressed */
}
```

Polling the UART should not be dismissed totally. It's a good method for diagnostics. If you have no idea of what address your card is at or what IRQ you are using you can poll the UART at several different addresses to firstly find which port your card is at and which one your modem is attached to. Once you know this information, then you can set up the Interrupt routines for the common IRQs and by enabling one IRQ at a time using the Programmable Interrupt Controller you can find out your IRQ, You don't even need a screw driver!

Buff1024.c - An Interrupt Driven Sample Comms Program

```
/* Name       : Sample Comm's Program - 1024 Byte Buffer - buff1024.c   */
/* Written By : Craig Peacock <cpeacock@senet.com.au>                    */
/*         Copyright 1997 CRAIG PEACOCK <cpeacock@senet.com.au>          */
/*           See http://www.senet.com.au/~cpeacock/serial.htm            */
/*                        For More Information                           */

#include <dos.h>
#include <stdio.h>
#include <conio.h>

#define PORT1 0x2E8  /* Port Address Goes Here */
/* Defines Serial Ports Base Address */
/* COM1 0x3F8                        */
/* COM2 0x2F8                        */
/* COM3 0x3E8                        */
/* COM4 0x2E8                        */
#define INTVECT 0x0B /* Com Port's IRQ here */
                     /* (Must also change PIC setting) */

int bufferin = 0;
int bufferout = 0;
char ch;
char buffer[1025];

void interrupt (*oldport1isr)();

void interrupt PORT1INT()  /* Interrupt Service Routine (ISR) for PORT1 */
{
 int c;
 do { c = inportb(PORT1 + 5);
      if (c & 1) {buffer[bufferin] = inportb(PORT1);
```

```
                    bufferin++;
                    if (bufferin == 1024) bufferin = 0;}
    }while (c & 1);
 outportb(0x20,0x20);
}

void main(void)
{
 int c;
 outportb(PORT1 + 1 , 0);        /* Turn off interrupts - Port1 */

 oldport1isr = getvect(INTVECT); /* Save old Interrupt Vector for */
                                 /* later recovery */

 setvect(INTVECT, PORT1INT);     /* Set Interrupt Vector Entry */
                                 /* COM1 - 0x0C */
                                 /* COM2 - 0x0B */
                                 /* COM3 - 0x0C */
                                 /* COM4 - 0x0B */

/*          PORT 1 - Communication Settings         */

outportb(PORT1 + 3 , 0x80);  /* SET DLAB ON */
outportb(PORT1 + 0 , 0x03);  /* Set Baud rate - Divisor Latch Low Byte */
                             /* Default 0x03 =  38,400 BPS */
                             /*         0x01 = 115,200 BPS */
                             /*         0x02 =  56,700 BPS */
                             /*         0x06 =  19,200 BPS */
                             /*         0x0C =   9,600 BPS */
                             /*         0x18 =   4,800 BPS */
                             /*         0x30 =   2,400 BPS */
outportb(PORT1 + 1 , 0x00);  /* Set Baud rate - Divisor Latch High Byte */
outportb(PORT1 + 3 , 0x03);  /* 8 Bits, No Parity, 1 Stop Bit */
outportb(PORT1 + 2 , 0xC7);  /* FIFO Control Register */
```

```
        outportb(PORT1 + 4 , 0x0B); /* Turn on DTR, RTS, and OUT2 */

        outportb(0x21,(inportb(0x21) & 0xF7)); /* Set Programmable Interrupt */
                                               /* Controller */
                                               /* COM1 (IRQ4) - 0xEF  */
                                               /* COM2 (IRQ3) - 0xF7  */
                                               /* COM3 (IRQ4) - 0xEF  */
                                               /* COM4 (IRQ3) - 0xF7  */

        outportb(PORT1 + 1 , 0x01); /* Interrupt when data received */

        printf("\nSample Comm's Program. Press ESC to quit \n");

        do {

            if (bufferin != bufferout){ch = buffer[bufferout];
                                       bufferout++;
                                       if (bufferout == 1024) bufferout = 0;
                                       printf("%c",ch);}

            if (kbhit()){c = getch();
                         outportb(PORT1, c);}

           } while (c !=27);

        outportb(PORT1 + 1 , 0);       /* Turn off interrupts - Port1 */
        outportb(0x21,(inportb(0x21) | 0x08)); /* MASK IRQ using PIC */
                                               /* COM1 (IRQ4) - 0x10  */
                                               /* COM2 (IRQ3) - 0x08  */
                                               /* COM3 (IRQ4) - 0x10  */
                                               /* COM4 (IRQ3) - 0x08  */
        setvect(INTVECT, oldport1isr); /* Restore old interrupt vector */
       }
```

Note: *The source code on the earier pages is not a really good example on how to program but is rather cut down to size giving quick results, and making it easier to understand. Upon executing your communications program, it would be wise to store the status of the UART registers, so that they all can be restored before you quit the program. This is to cause the least upset to other programs which may also be trying to use the communications ports.*

The first step to using interrupts is to work out which interrupt services your serial card. Table 13 shows the base addresses and IRQ's of some standard ports. IRQ's 3 and 4 are the two most commonly used. IRQ 5 and 7 are sometimes used.

Interrupt Vectors

Once we know the IRQ the next step is to find it's interrupt vector or software interrupt as some people may call it. Basically any 8086 processor has a set of 256 interrupt vectors numbered 0 to 255. Each of these vectors contains a 4 byte code which is an address of the Interrupt Service Routine (ISR). Fortunately C being a high level language, takes care of the addresses for us. All we have to know is the actual interrupt vector.

INT (Hex)	IRQ	Common Uses
08	0	System Timer
09	1	Keyboard
0A	2	Redirected
0B	3	Serial Comms. COM2/COM4
0C	4	Serial Comms. COM1/COM3
0D	5	Reserved/Sound Card
0E	6	Floppy Disk Controller
0F	7	Parallel Comms.
70	8	Real Time Clock
71	9	Reserved
72	10	Reserved
73	11	Reserved
74	12	PS/2 Mouse

75	13	Maths Co-Processor
76	14	Hard Disk Drive
77	15	Reserved

Table 14 : Interrupt Vectors (Hardware Only)

The above table shows only the interrupts which are associated with IRQ's. The other 240 are of no interest to us when programming RS-232 type communications.

For example if we were using COM3 which has a IRQ of 4, then the interrupt vector would be 0C in hex. Using C we would set up the vector using the instruction `setvect(0x0C, PORT1INT);` where PORT1INT would lead us to a set of instructions which would service the interrupt.

However before we proceed with that I should say that it is wise to record the old vectors address and then restore that address once the program is finished. This is done using `oldport1isr = getvect(INTVECT);` where oldport1isr is defined using `void interrupt (*oldport1isr)();`

Not only should you store the old vector addresses, but also the configuration the UART was in. Why you Ask? Well it's simple, I wrote a communications program which was fully featured in the chat side of things. It had line buffering, so no body could see my spelling mistakes or how slowly I typed. It included anti-bombing routines and the list goes on. However I couldn't be bothered to program any file transfer protocols such as Zmodem etc into my communications program. Therefore I either had to run my communications program in the background of Telemate using my communications program for chat and everything else it was designed for and using Telemate to download files. Another method was to run, say Smodem as a external protocol to my communications program.

Doing this however would mean that my communications program would override the original speed, parity etc and then when I returned to the original communications program, everything stopped. Therefore by saving the old configuration, you can revert back to it before you hand the UART back over to the other program. Makes sense? However if you don't have any of these programs you can save yourself a few lines of code. This is what we have done here.

Interrupt Service Routine (ISR)

Now, could we be off track just a little? Yes that's right, PORT1INT is the label to our interrupt handler called a Interrupt Service Routine (ISR). You can put just about anything in here you want. However calling some DOS routines can be a problem.

```
void interrupt PORT1INT()
{
 int c;
 do { c = inportb(PORT1 + 5);
      if (c & 1) {
                  buffer[bufferin] = inportb(PORT1);
                  bufferin++;
                  if (bufferin == 1024) bufferin = 0;
                 }
    } while (c & 1);
outportb(0x20,0x20);
}
```

From the example above we check to see if there is a character to receive and if their is we remove it from the UART and place it in a buffer contained in memory. We keep on checking the UART, in case FIFO's are enabled, so we can get all data available at the time of interrupt.

The last line contains the instruction `outportb(0x20,0x20);` which tells the Programmable Interrupt Controller that the interrupt has finished. The Programmable Interrupt Controller (PIC) is what we must go into now. All of the routines above, we have assumed that everything is set up ready to go. That is all the UART's registers are set correctly and that the Programmable Interrupt Controller is set.

The Programmable Interrupt Controller handles hardware interrupts. Most PC's will have two of them located at different addresses. One handles IRQ's 0 to 7 and the other IRQ's 8 to 15. Mainly Serial communications interrupts reside on IRQ's under 7, thus PIC1 is used, which is located at 0020 Hex.

Bit	Disable IRQ	Function
7	IRQ7	Parallel Port
6	IRQ6	Floppy Disk Controller
5	IRQ5	Reserved/Sound Card
4	IRQ4	Serial Port
3	IRQ3	Serial Port
2	IRQ2	PIC2
1	IRQ1	Keyboard
0	IRQ0	System Timer

Table 15 : PIC1 Control Word (0x21)

Multi-Comm ports are getting quite common, thus table 16 includes data for PIC2 which is located at 0xA0. PIC2 is responsible for IRQ's 8 to 15. It operates in exactly the same way than PIC1 except that EOI's (End of Interrupt) goes to port 0xA0 while the disabling (Masking) of IRQ's are done using port 0xA1.

Bit	Disable IRQ	Function
7	IRQ15	Reserved
6	IRQ14	Hard Disk Drive
5	IRQ13	Maths Co-Processor
4	IRQ12	PS/2 Mouse
3	IRQ11	Reserved
2	IRQ10	Reserved

1	IRQ9	IRQ2
0	IRQ8	Real Time Clock

Table 16 : PIC2 Control Word (0xA1)

Most of the PIC's initiation is done by BIOS. All we have to worry about is two instructions. The first one is `outportb(0x21, (inportb(0x21) & 0xEF);` which selects which interrupts we want to Disable (Mask). So if we want to enable IRQ4 we would have to take 0x10 (16) from 0xFF (255) to come up with 0xEF (239). That means we want to disable IRQ's 7,6,5,3,2,1 and 0, thus enabling IRQ 4.

But what happens if one of these IRQs are already enabled and then we come along and disable it? Therefore we input the value of the register and using the & function output the byte back to the register with our changes using the instruction `outportb(0x21, (inportb(0x21) & 0xEF);`. For example if IRQ5 is already enabled before we come along, it will enable both IRQ4 and IRQ5 so we don't make any changes which may affect other programs or TSR's.

The other instruction is `outportb(0x20,0x20);` which signals an end of interrupt to the PIC. You use this command at the end of your interrupt service routine, so that interrupts of a lower priority will be accepted.

UART Configuration

Now we get to the UART settings (Finally)

It's a good idea to turn off the interrupt generation on the UART as the first instruction. Therefore your initialization can't get interrupted by the UART. I've then chosen to set up our interrupt vectors at this point. The next step is to set the speed at which you wish to communicate at. If you remember the process, we have to set bit 7 (The DLAB) of the LCR so we can access the Divisor Latch High and Low Bytes. We have decided to set the speed to 38,400 Bits per second which should be find for 16450's and 16550's. This requires a divisor of 3, thus our divisor latch high byte will be 0x00 and a divisor latch low byte, 0x03.

In today's standards the divisor low latch byte is rarely used but it still pays us to write 0x00 to the register just in case the program before us just happened to set the UART at a very very low speed. BIOS will normally set UARTs at 2400 BPS when the computer is first booted up which still doesn't require the Divisor Latch Low byte.

The next step would be to turn off the Divisor latch access bit so we can get to the Interrupt Enable Register and the receiver/transmitter buffers. What we could do is just write a 0x00 to the register clearing it all, but considering we have to set up our word length, parity as so forth in the line control register we can do this at the same time. We have decided to set up 8 bits, no parity and 1 stop bit which is normally used today. Therefore we write 0x03 to the line control register which will also turn off the DLAB for us saving one more I/O instruction.

The next line of code turns on the FIFO buffers. We have made the trigger level at 14 bytes, thus bits 6 and 7 are on. We have also enabled the FIFO's (bit 0). It's also good practice to clear out the FIFO buffers on initialization. This will remove any rubbish which the last program may of left in the FIFO buffers. Due to the fact that these two bits are self resetting, we don't have to go any further and turn off these bits. If my arithmetic is correct all these bits add up to 0xC7 or 199 for those people which still work in decimal.

Then DTR, RTS and OUT 2 is taken active by the instruction `outportb(PORT1 + 4,0x0B);`. Some cards (Both of Mine) require OUT2 active for interrupt requests thus I'm normally always take it high. All that is left now is to set up our interrupts which has be deliberately left to last as to not interrupt our initialization. Our interrupt handler is only interested in new data being available so we have only set the UART to interrupt when data is received.

Main Routine (Loop)

Now we are left with,

```
do {
    if (bufferin != bufferout){
                                  ch = buffer[bufferout];
                                  bufferout++;
                                  if (bufferout == 1024) bufferout = 0;
                                  printf("%c",ch);
                                 }
    if (kbhit()){
                  c = getch();
                  outportb(PORT1, c);
                 }
   } while (c !=27);
```

which keeps repeating until c = 27. This occurs when the ESC key is hit.

The next *if* statement checks to see if a key has been hit. (`kbhit()`) If so, it gets the character using the `getch()` statement and outputs it to the receiver buffer. The UART then transmits the character to the modem. What we have assumed here, is that the person using the Communications Program can't type as fast as the UART can send. However if the program wishes to send something, then a check should be made to see if BIT 5 of the Line Status Register is set before attempting to send a byte to the transmitter register.

*For more information on Interrupts, try **Using Interrupts**,*
http://www.geocities.com/SiliconValley/Bay/8302/interupt.pdf (62k)

Determining the type of UART via software

The type of UART you have installed in your system can be determined without even needing a screwdriver in most cases. As you can see from the Types of UARTs, each UART has minor differences, all we have to do it test these.

The first procedure we do is to set bit 0 to '1' in the FIFO control register. This tries to enable the FIFO buffers. Then we read bits 6 and 7 from the interrupt identification register. If both bits are '1' then the FIFO buffers are enabled. This would mean the UART is a 16550a. If the FIFO's were enabled but not usable then it would be a 16550. If there is no FIFO buffer enabled it is most likely to be a 16450 UART, but could be a 8250, 8250A or 8250B on very old systems.

AT's have a fast bus speed which the 8250 series of UART can't handle to well thus it is very unlikely to be found in any AT. However if you wish to test for them as well you can follow the same test as above to distinguish 16550's or 16550A's from the rest. If no FIFOs are enabled then a possible UART is the 16450, 8250, 8250A or 8250B. Once it is established the it could be one of these four chips, try writing a byte to the scratch register and then read it back and compare the results. If the results match then you must have a scratch register, if they don't you either don't have a scratch register, or it doesn't work to well.

From the descriptions of the UART above if you read back your byte from the scratch register then the UART must be a 16450 or 8250A. (Both have scratch registers) If you don't read back your byte then it's either a 8250 or 8250B.

The 16750 has 64 byte FIFO's, thus the easiest way to test for it's presence is to enable the 64 byte buffer using the FIFO Control Register and then read back the status of the Interrupt Identification Register. However I have never tested this.

Part 4 : Interfacing Devices to RS-232 Ports

RS-232 Waveforms

So far we have introduced RS-232 Communications in relation to the PC. RS-232 communication is asynchronous. That is a clock signal is not sent with the data. Each word is synchronized using it's start bit, and an internal clock on each side, keeps tabs on the timing.

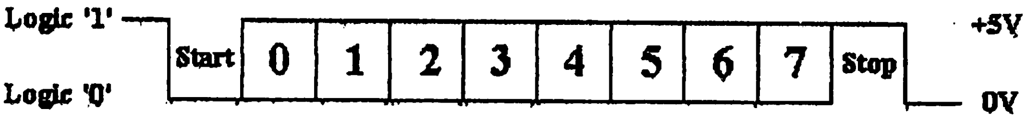


Figure 4 : TTL/CMOS Serial Logic Waveform

The diagram above, shows the expected waveform from the UART when using the common 8N1 format. 8N1 signifies 8 Data bits, No Parity and 1 Stop Bit. The RS-232 line, when idle is in the Mark State (Logic 1). A transmission starts with a start bit which is (Logic 0). Then each bit is sent down the line, one at a time. The LSB (Least Significant Bit) is sent first. A Stop Bit (Logic 1) is then appended to the signal to make up the transmission.

The diagram, shows the next bit after the Stop Bit to be Logic 0. This must mean another word is following, and this is it's Start Bit. If there is no more data coming then the receive line will stay in it's idle state(logic 1). We have encountered something called a "Break" Signal. This is when the data line is held in a Logic 0 state for a time long enough to send an entire word. Therefore if you don't put the line back into an idle state, then the receiving end will interpret this as a break signal.

The data sent using this method, is said to be *framed*. That is the data is *framed* between a Start and Stop Bit. Should the Stop Bit be received as a Logic 0, then a framing error will occur. This is common, when both sides are communicating at different speeds.

The above diagram is only relevant for the signal immediately at the UART. RS-232 logic levels uses +3 to +25 volts to signify a "Space" (Logic 0) and -3 to -25 volts for a "Mark" (logic 1). Any voltage in between these regions (ie between +3 and -3 Volts) is undefined. Therefore this signal is put through a "RS-232 Level Converter". This is the signal present on the RS-232 Port of your computer, shown below.

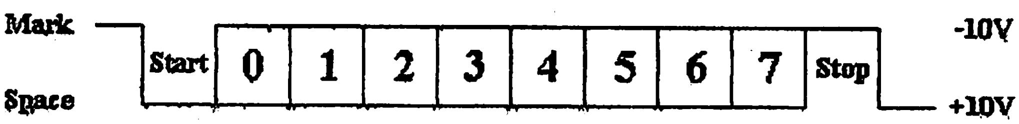


Figure 5 : RS-232 Logic Waveform

The above waveform applies to the Transmit and Receive lines on the RS-232 port. These lines carry serial data, hence the name Serial Port. There are other lines on the RS-232 port which, in essence are *Parallel* lines. These lines (RTS, CTS, DCD, DSR, DTR, RTS and RI) are also at RS-232 Logic Levels.

www.p-tec.net
sales@p-tec.net

P-tec

Tel: (719) 589 3122
Fax: (719) 589 3592

PLI90 INFRARED SERIES 3MM

Features

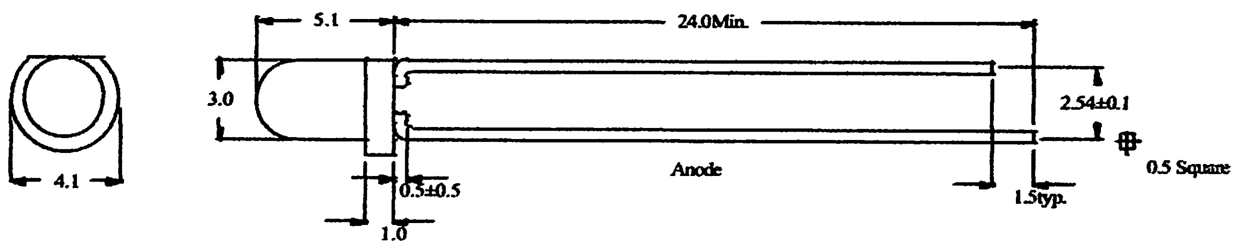
* High Radiant Power
* Lower Degradation
* IC Compatible

Absolute Maximum Ratings at $T_A = 25$ °C

Reverse Voltage	5.0V
Max Forward Current	100mA
Peak Forward Current (1/10 Duty Cycle, 0.1ms Pulse Width)	200mA
Operating Temperature Range	- 25°C to + 85°C
Storage Temperature Range	- 25°C to +100°C
Soldering Temperature (1.6 mm below body)	260° for 5 seconds

Electrical & Optical Characteristics at $T_A = 25$ °C

Lens Color Part Number		Infrared Chip Material	Peak Wave Length	View Angle 2θ ½	Forward Voltage (V)			Radiant Intensity (mcd)		
Water Clear	Blue Transparent		nm	Deg	Typ	Max	IF(mA)	Min	Typ	IF(mA)
PLI90-WCIR880	PLI90-BTIR880	GaAlAs	880	36	1.5	1.9	50	8.0	13	100
PLI90-WCIR940	PLI90-BTIR940	GaAs	940	36	1.5	1.9	50	8.0	13	100



PL190 INFRARED SERIES 3MM

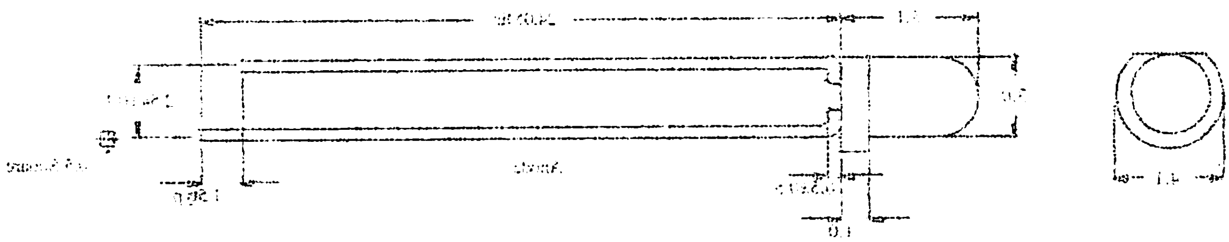
Features

* High Radiant Power
* Lower Degradation
* IC Compatible

Absolute Maximum Ratings at T_A = 25 °C	
Reverse Voltage	5.0V
Max Forward Current	100mA
Peak Forward Current (1/10 Duty Cycle, 0.1ms Pulse Width)	200mA
Operating Temperature Range	-25°C to + 85°C
Storage Temperature Range	-25°C to +100°C
Soldering Temperature (1.6 mm below body)	260° for 5 seconds

Electrical & Optical Characteristics at T_A = 25 °C

Lens Color Part Number		Infrared Chip Material	Peak Wave Length	View Angle 2θ½	Forward Voltage (V)			Radiant Intensity (mcd)		
Water Clear	Blue Transparent		nm	Deg	Typ	Max	IF(mA)	Min	Typ	IF(mA)
PL190-WCIR880	PL190-BTIR880	GaAlAs	880	36	1.5	1.9	50	8.0	13	100
PL190-WCIR940	PL190-BTIR940	GaAs	940	36	1.5	1.9	50	8.0	13	100

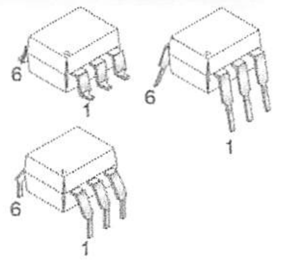


FAIRCHILD
SEMICONDUCTOR®

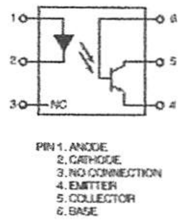
GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

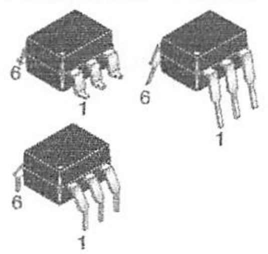
WHITE PACKAGE (-M SUFFIX)



SCHEMATIC



BLACK PACKAGE (NO -M SUFFIX)



DESCRIPTION

The general purpose optocouplers consist of a gallium arsenide infrared emitting diode driving a silicon phototransistor in a 6-pin dual in-line package.

FEATURES

- Also available in white package by specifying -M suffix, eg. 4N25-M
- UL recognized (File # E90700)
- VDE recognized (File # 94766)
  - Add option V for white package (e.g., 4N25V-M)
  - Add option 300 for black package (e.g., 4N25.300)

APPLICATIONS

- Power supply regulators
- Digital logic inputs
- Microprocessor inputs

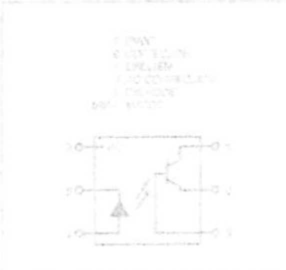
FAIRCHILD
SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

WHITE PACKAGE (-M SUFFIX)

BLACK PACKAGE (NO -M SUFFIX)



SCHEMATIC

DESCRIPTION

The general purpose optocouplers consist of a gallium arsenide infrared emitting diode driving a silicon phototransistor in a 6-pin dual in-line package.

FEATURES

- Also available in white package by specifying -M suffix, eg. 4N25-M
- UL recognized (File # E90700)
- VDE recognized (File # 94766)
  - Add option V for white package (e.g., 4N25V-M)
  - Add option 300 for black package (e.g., 4N25.300)

APPLICATIONS

- Power supply regulators
- Digital logic inputs
- Microprocessor inputs

ABSOLUTE MAXIMUM RATINGS (T_A = 25°C unless otherwise specified)

Parameter	Symbol	Value	Units
TOTAL DEVICE			
Storage Temperature	T_{STG}	-55 to +150	°C
Operating Temperature	T_{OPR}	-55 to +100	°C
Wave solder temperature (see page 14 for reflow solder profiles)	T_{SOL}	260 for 10 sec	°C
Total Device Power Dissipation @ T_A = 25°C	P_D	250	mW
Derate above 25°C		3.3 (non-M), 2.94 (-M)	
EMITTER			
DC/Average Forward Input Current	I_F	100 (non-M), 60 (-M)	mA
Reverse Input Voltage	V_R	6	V
Forward Current - Peak (300µs, 2% Duty Cycle)	I_F(pk)	3	A
LED Power Dissipation @ T_A = 25°C	P_D	150 (non-M), 120 (-M)	mW
Derate above 25°C		2.0 (non-M), 1.41 (-M)	mW/°C
DETECTOR			
Collector-Emitter Voltage	V_{CEO}	30	V
Collector-Base Voltage	V_{CBO}	70	V
Emitter-Collector Voltage	V_{ECO}	7	V
Detector Power Dissipation @ T_A = 25°C	P_D	150	mW
Derate above 25°C		2.0 (non-M), 1.76 (-M)	mW/°C

ELECTRICAL CHARACTERISTICS (T_A = 25°C unless otherwise specified)

INDIVIDUAL COMPONENT CHARACTERISTICS

Parameter	Test Conditions	Symbol	Min	Typ*	Max	Unit
EMITTER						
Input Forward Voltage	(I_F = 10 mA)	V_F		1.18	1.50	V
Reverse Leakage Current	(V_R = 6.0 V)	I_R		0.001	10	µA
DETECTOR						
Collector-Emitter Breakdown Voltage	(I_C = 1.0 mA, I_F = 0)	BV_{CEO}	30	100		V
Collector-Base Breakdown Voltage	(I_C = 100 µA, I_F = 0)	BV_{CBO}	70	120		V
Emitter-Collector Breakdown Voltage	(I_E = 100 µA, I_F = 0)	BV_{ECO}	7	10		V
Collector-Emitter Dark Current	(V_{CE} = 10 V, I_F = 0)	I_{CEO}		1	50	nA
Collector-Base Dark Current	(V_{CB} = 10 V)	I_{CBO}			20	nA
Capacitance	(V_{CE} = 0 V, f = 1 MHz)	C_{CE}		8		pF

ISOLATION CHARACTERISTICS

Characteristic	Test Conditions	Symbol	Min	Typ*	Max	Units
Input-Output Isolation Voltage	(Non '-M', Black Package) (f = 60 Hz, t = 1 min)	V_{ISO}	5300			Vac(rms)
	('-M', White Package) (f = 60 Hz, t = 1 sec)		7500			Vac(pk)
Isolation Resistance	(V_{I-O} = 500 VDC)	R_{ISO}	10^{11}			Ω
Isolation Capacitance	(V_{I-O} = &, f = 1 MHz)	C_{ISO}		0.5		pF
	('-M' White Package)			0.2	2	pF

Note

* Typical values at T_A = 25°C

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.)

DC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Current Transfer Ratio, Collector to Emitter	(I_F = 10 mA, V_{CE} = 10 V)	CTR	4N35 4N36 4N37	100			%
			H11A1	50			
			H11A5	30			
			4N25 4N26 H11A2 H11A3	20			
			4N27 4N28 H11A4	10			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = -55°C)		4N35 4N36 4N37	40			
	(I_F = 10 mA, V_{CE} = 10 V, T_A = +100°C)		4N35 4N36 4N37	40			
Collector-Emitter Saturation Voltage	(I_C = 2 mA, I_F = 50 mA)	$V_{CE\ (SAT)}$	4N25 4N26 4N27 4N28			0.5	V
	(I_C = 0.5 mA, I_F = 10 mA)		4N35 4N36 4N37			0.3	
			H11A1 H11A2 H11A3 H11A4 H11A5			0.4	
AC Characteristic Non-Saturated Turn-on Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		µs
Non Saturated Turn-on Time	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{ON}	4N35 4N36 4N37		2	10	µs

TRANSFER CHARACTERISTICS (T_A = 25°C Unless otherwise specified.) (Continued)

AC Characteristic	Test Conditions	Symbol	Device	Min	Typ*	Max	Unit
Turn-off Time	(I_F = 10 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)	T_{OFF}	4N25 4N26 4N27 4N28 H11A1 H11A2 H11A3 H11A4 H11A5		2		μs
	(I_C = 2 mA, V_{CC} = 10 V, R_L = 100Ω) (Fig.20)		4N35 4N36 4N37		2	10	

* Typical values at T_A = 25°C

TYPICAL PERFORMANCE CURVES

Fig. 1 LED Forward Voltage vs. Forward Current (Black Package)

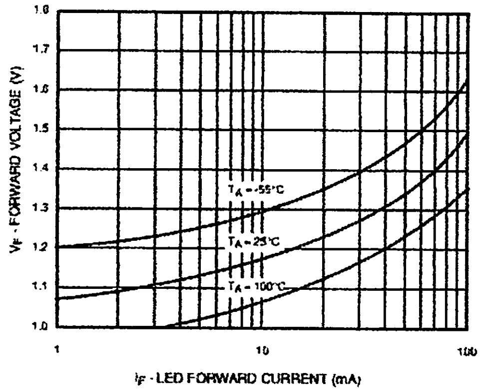


Fig. 2 LED Forward Voltage vs. Forward Current (White Package)

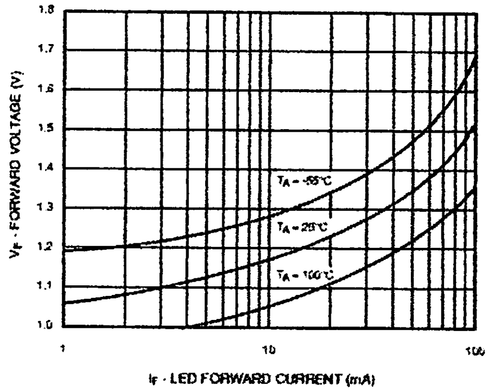


Fig.3 Normalized CTR vs. Forward Current (Black Package)

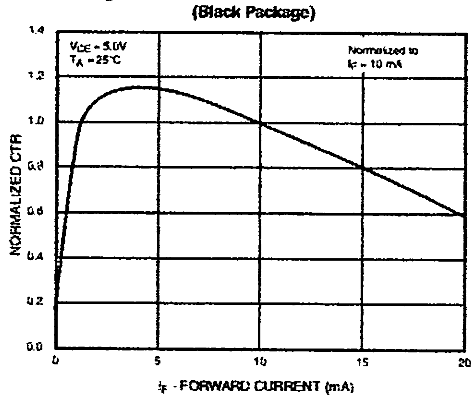


Fig.4 Normalized CTR vs. Forward Current (White Package)

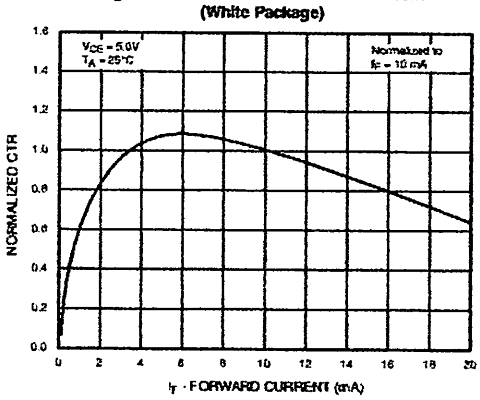


Fig. 5 Normalized CTR vs. Ambient Temperature (Black Package)

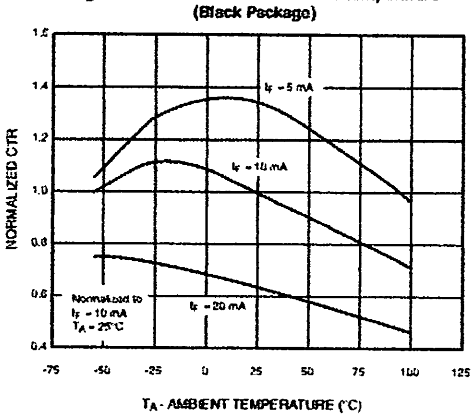


Fig. 6 Normalized CTR vs. Ambient Temperature (White Package)

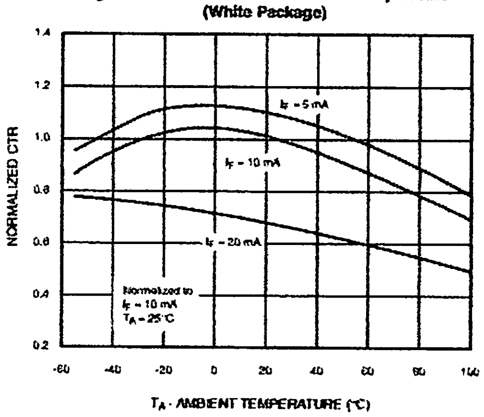


Fig. 7 CTR vs. RBE (Unsaturated) (Black Package)

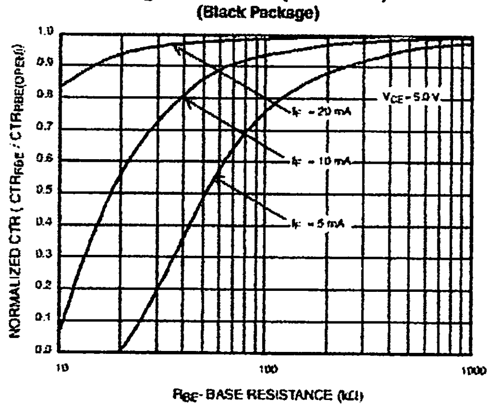


Fig. 8 CTR vs. RBE (Unsaturated) (White Package)

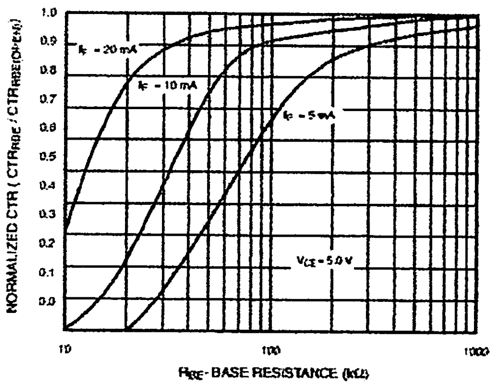


Fig. 9 CTR vs. RBE (Saturated) (Black Package)

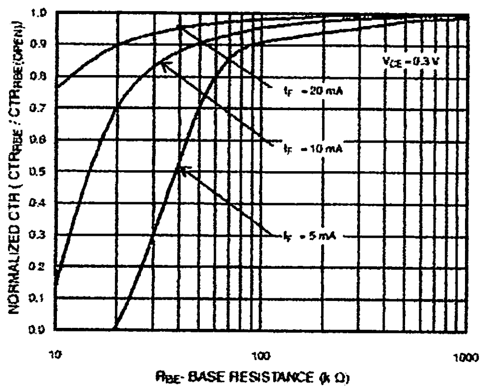


Fig. 10 CTR vs. RBE (Saturated) (White Package)

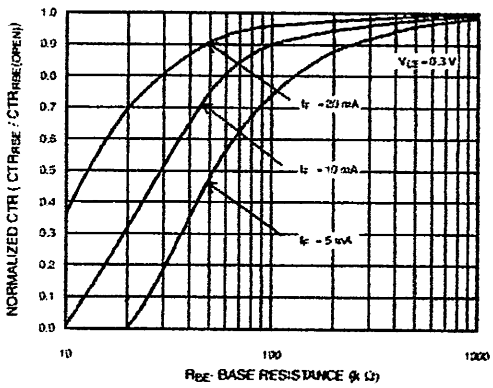


Fig. 11 Collector-Emitter Saturation Voltage vs Collector Current (Black Package)

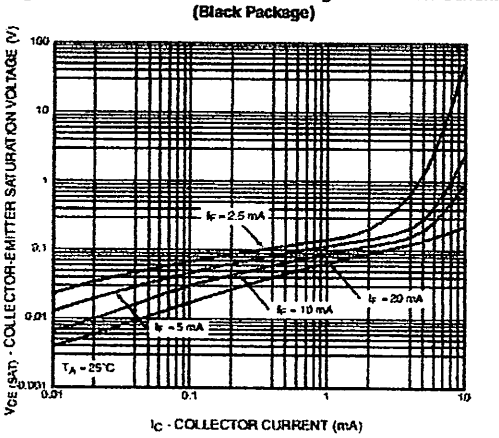


Fig. 12 Collector-Emitter Saturation Voltage vs Collector Current (White Package)

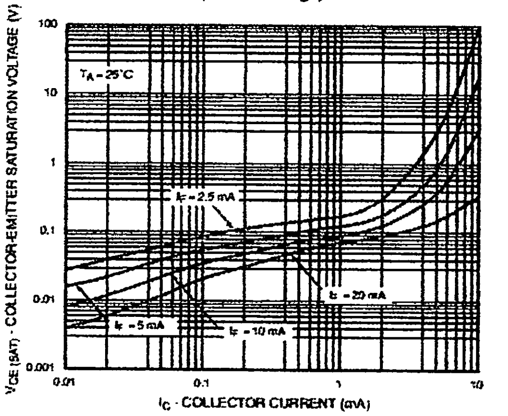


Fig. 13 Switching Speed vs. Load Resistor (Black Package)

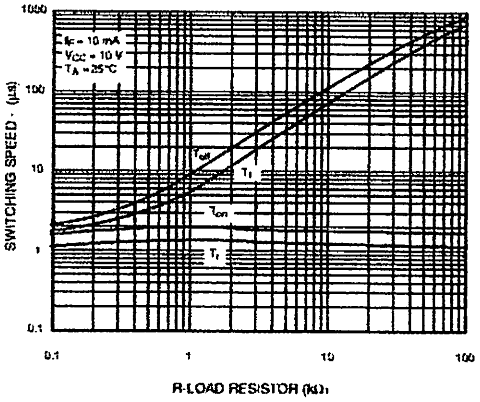


Fig. 14 Switching Speed vs. Load Resistor (White Package)

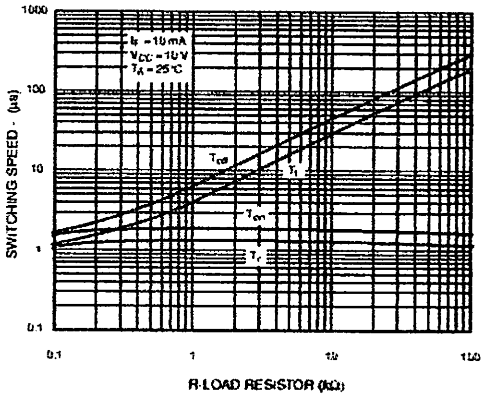


Fig. 15 Normalized t_{on} vs. R_{BE} (Black Package)

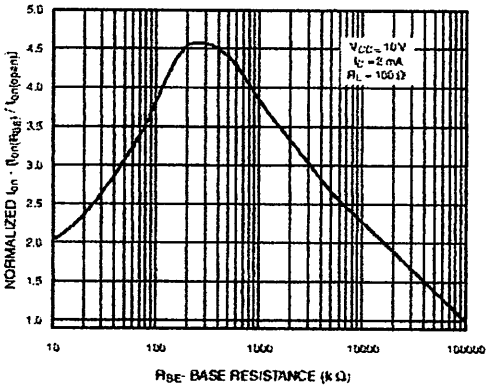


Fig. 16 Normalized t_{on} vs. R_{BE} (White Package)

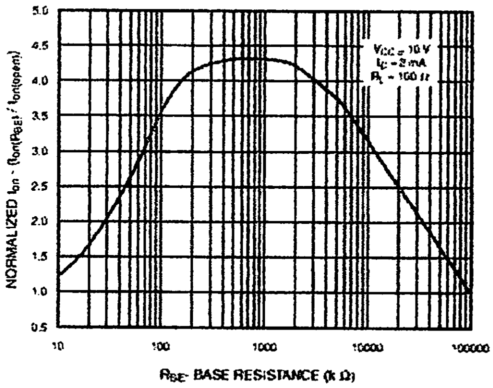


Fig. 17 Normalized t_{off} vs. R_{BE} (Black Package)

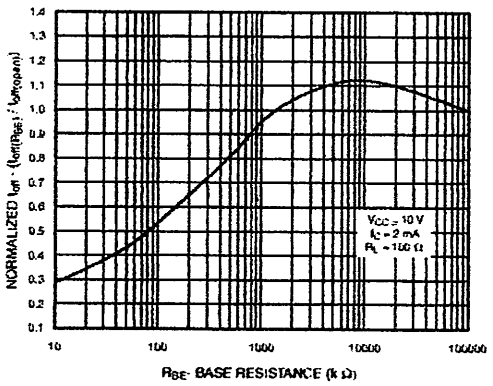


Fig. 18 Normalized t_{off} vs. R_{BE} (White Package)

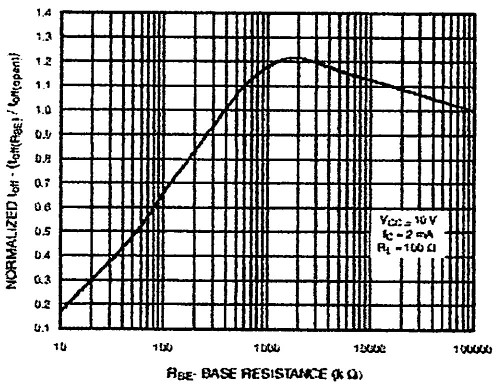


Fig. 13 Switching Speed vs. Load Resistor (Black Package)

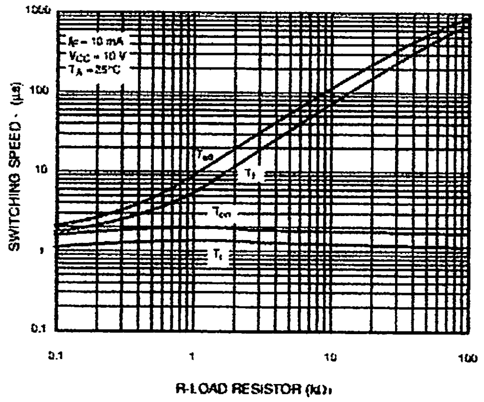


Fig. 14 Switching Speed vs. Load Resistor (White Package)

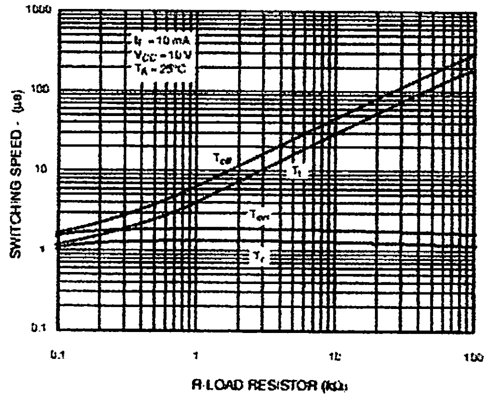


Fig. 15 Normalized t_{on} vs. R_{BE} (Black Package)

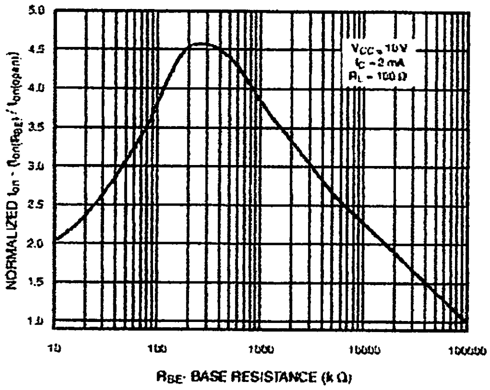


Fig. 16 Normalized t_{on} vs. R_{BE} (White Package)

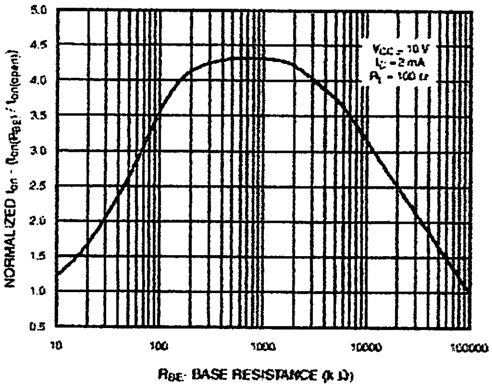


Fig. 17 Normalized t_{off} vs. R_{BE} (Black Package)

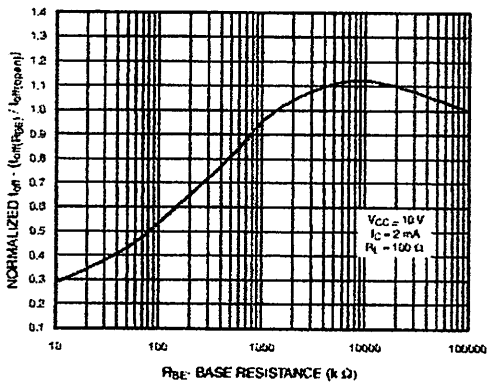


Fig. 18 Normalized t_{off} vs. R_{BE} (White Package)

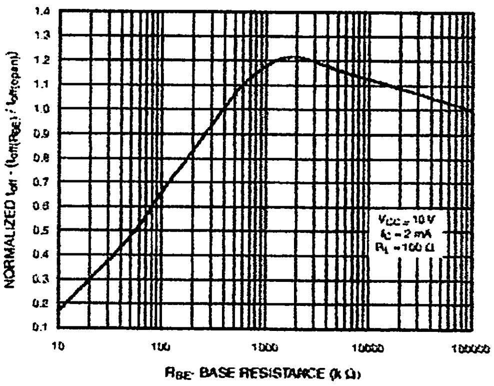
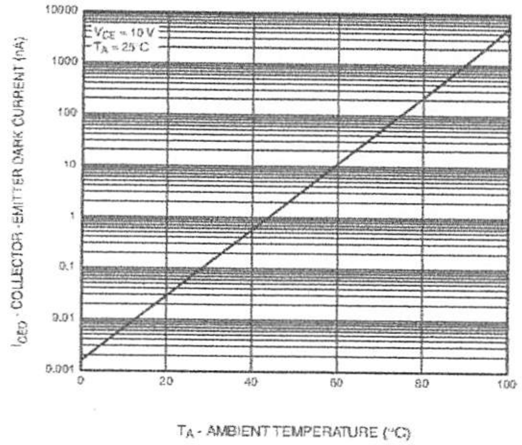


Fig. 19 Dark Current vs. Ambient Temperature



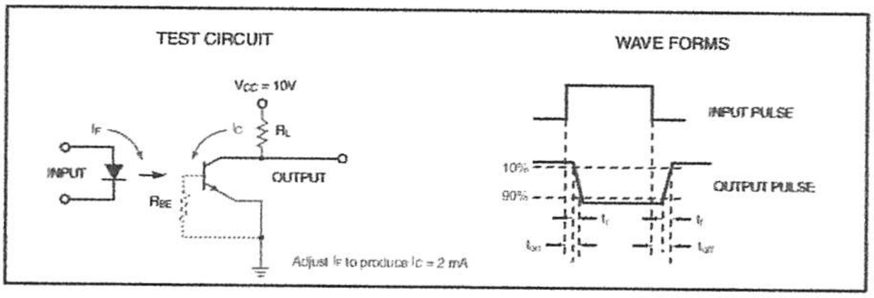
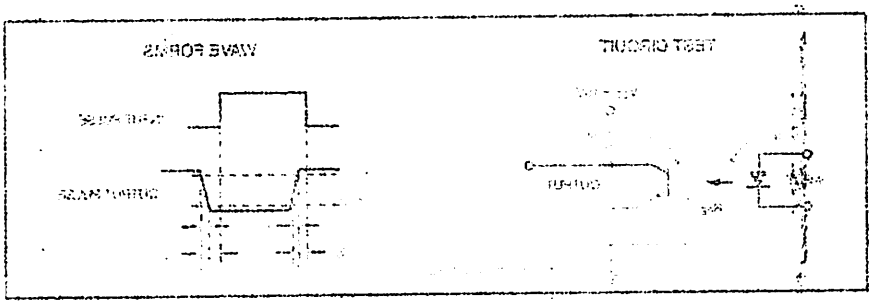


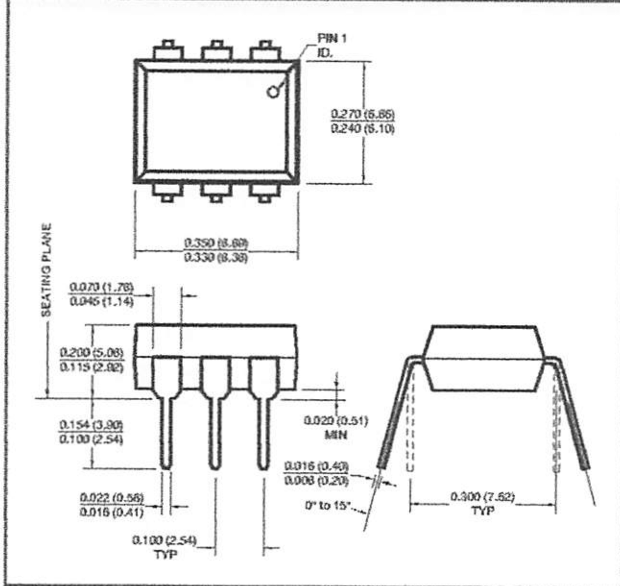
Figure 20. Switching Time Test Circuit and Waveforms


WAVE FORMS
TEST CIRCUIT
OUTPUT

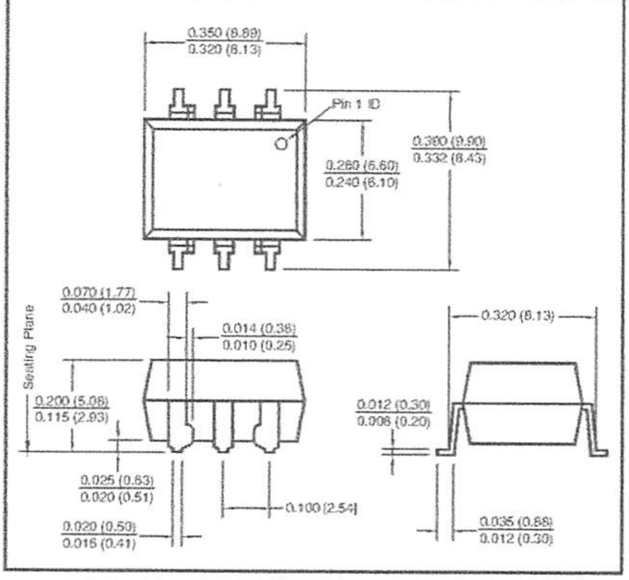
4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

Black Package (No -M Suffix)

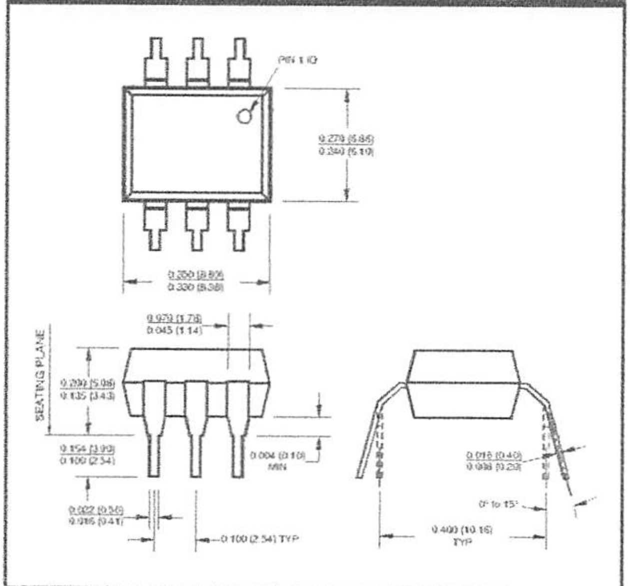
Package Dimensions (Through Hole)



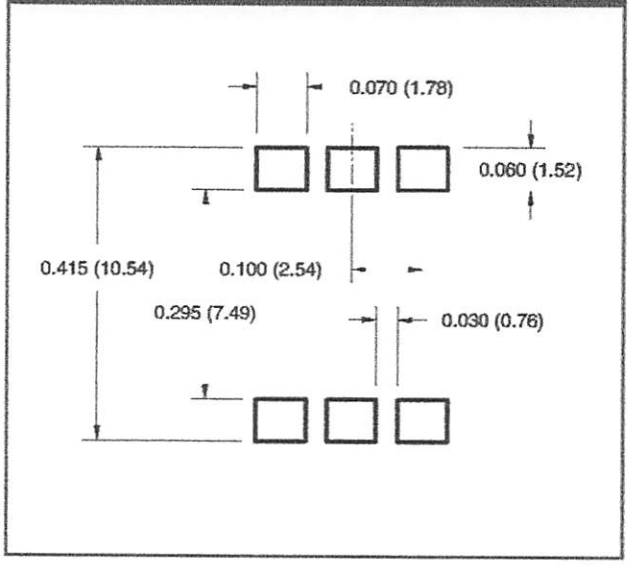
Package Dimensions (Surface Mount)



Package Dimensions (0.4" Lead Spacing)



Recommended Pad Layout for Surface Mount Leadform



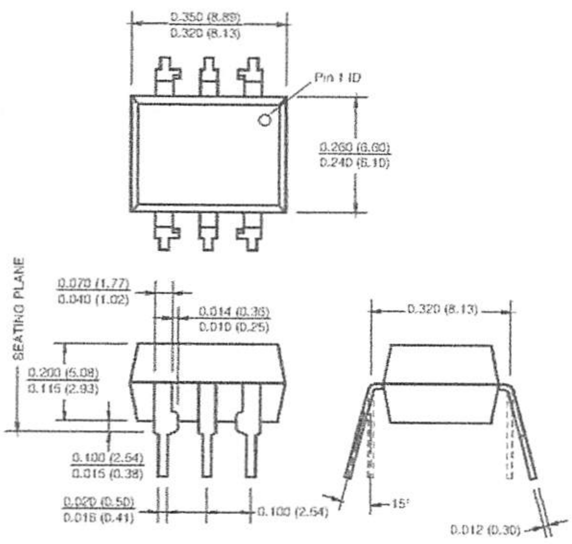
NOTE
All dimensions are in inches (millimeters)

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

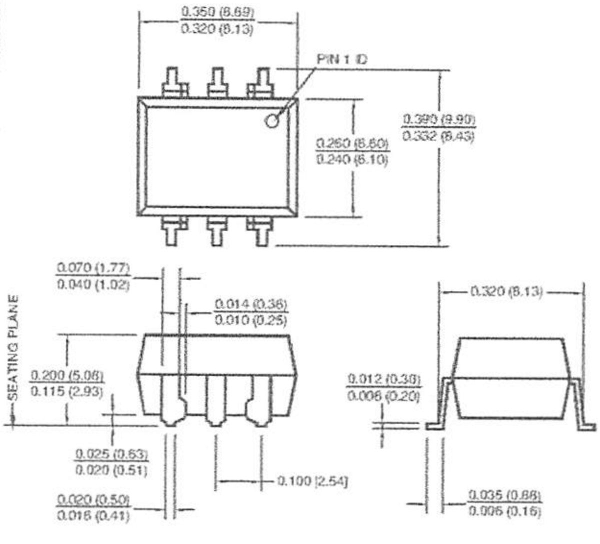
4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

White Package (-M Suffix)

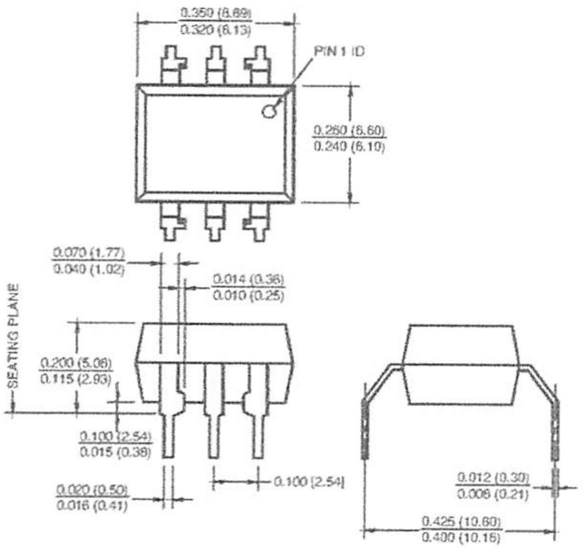
Package Dimensions (Through Hole)



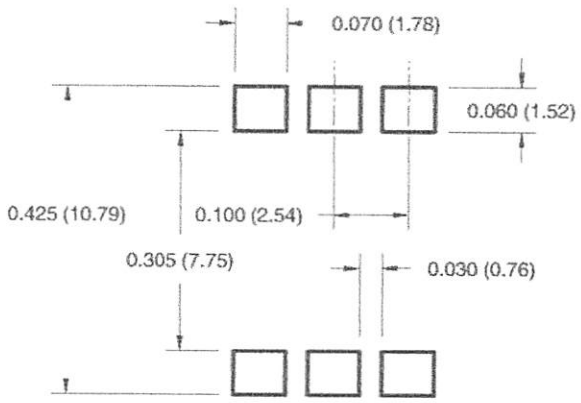
Package Dimensions (Surface Mount)



Package Dimensions (0.4" Lead Spacing)



Recommended Pad Layout for Surface Mount Leadform



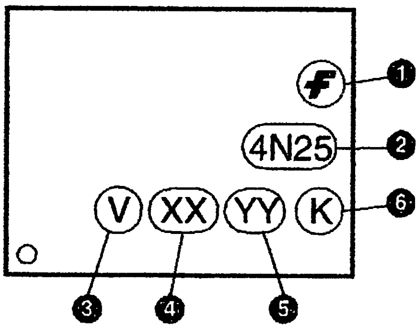
NOTE
All dimensions are in inches (millimeters)

4N25	4N26	4N27	4N28	4N35	4N36
4N37	H11A1	H11A2	H11A3	H11A4	H11A5

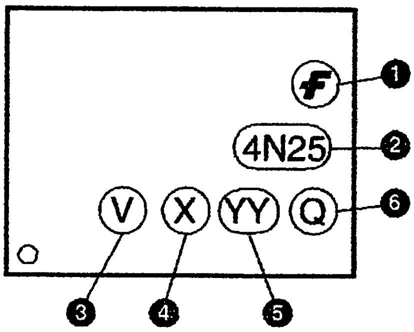
ORDERING INFORMATION

Order Entry Identifier		
Black Package (No Suffix)	**White Package (-M Suffix)**	**Option**
.S	S	Surface Mount Lead Bend
.SD	SR2	Surface Mount; Tape and reel
.W	T	0.4" Lead Spacing
.300	V	VDE 0884
.300W	TV	VDE 0884, 0.4" Lead Spacing
.3S	SV	VDE 0884, Surface Mount
.3SD	SR2V	VDE 0884, Surface Mount, Tape & Reel

MARKING INFORMATION



Black Package, No Suffix

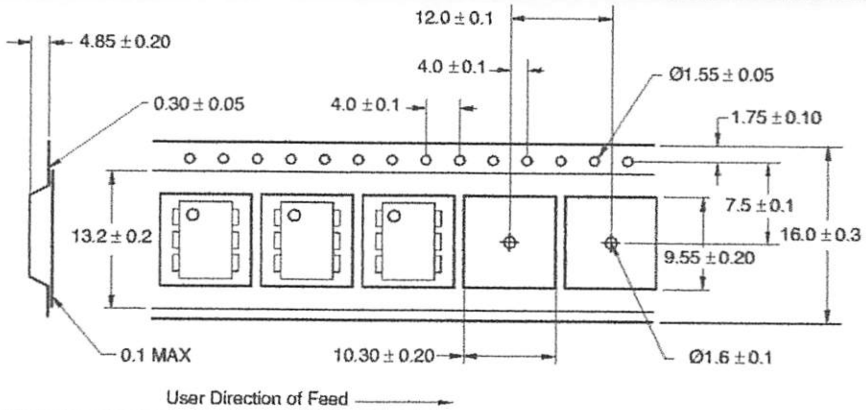


White Package, -M Suffix

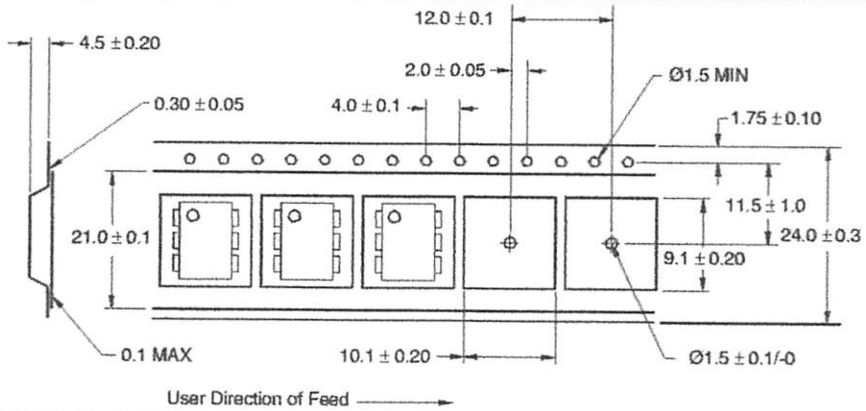
Definitions	
1	Fairchild logo
2	Device number
3	VDE mark (Note: Only appears on parts ordered with VDE option – See order entry table)
4	One or two digit year code • Two digits for black package parts, e.g., '03' • One digit for white package parts, e.g., '3'
5	Two digit work week ranging from '01' to '53'
6	Assembly package code

*Note – Parts built in the white package (M suffix) that do not have the 'V' option (see definition 3 above) that are marked with date code '325' or earlier are marked in the portrait format.

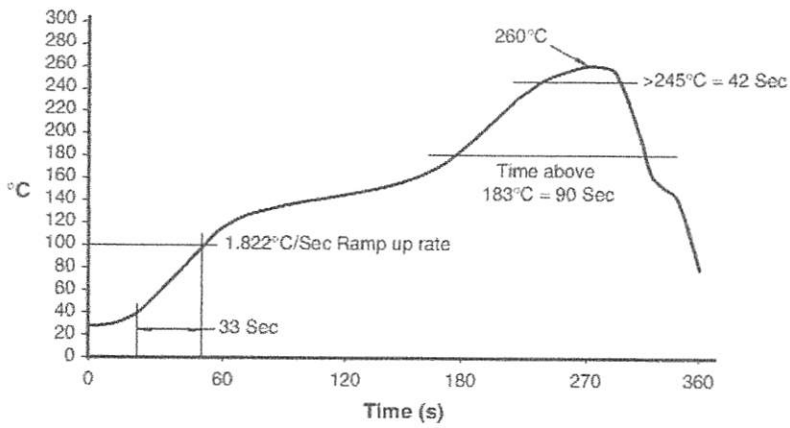
QT Carrier Tape Specifications (Black Package, No Suffix)



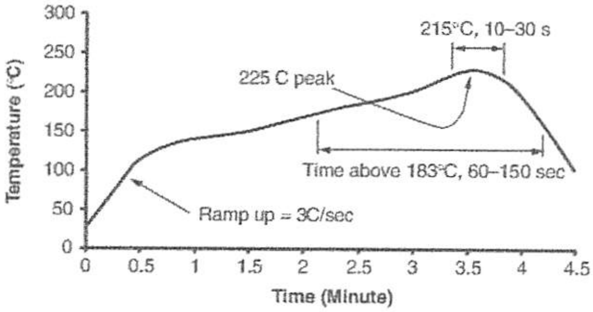
QT Carrier Tape Specifications (White Package, -M Suffix)



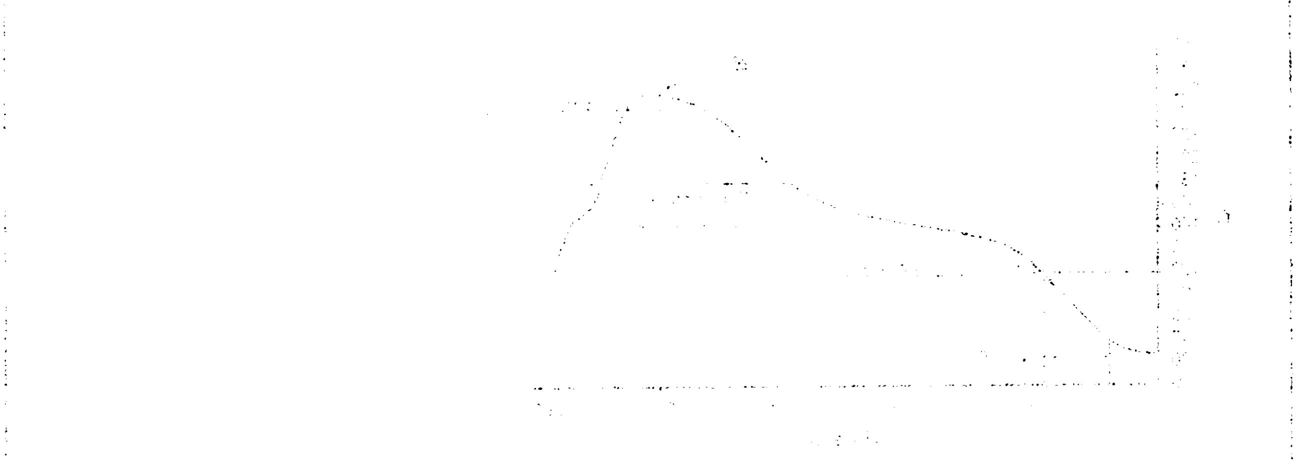
Reflow Profile (White Package, -M Suffix)



Reflow Profile (Black Package, No Suffix)



- Peak reflow temperature: 225°C (package surface temperature)
- Time of temperature higher than 183°C for 60–150 seconds
- One time soldering reflow is recommended



FAIRCHILD SEMICONDUCTOR®

GENERAL PURPOSE 6-PIN PHOTOTRANSISTOR OPTOCOUPLERS

4N25	**4N26**	**4N27**	**4N28**	**4N35**	**4N36**
4N37	**H11A1**	**H11A2**	**H11A3**	**H11A4**	**H11A5**

Computer Control #2

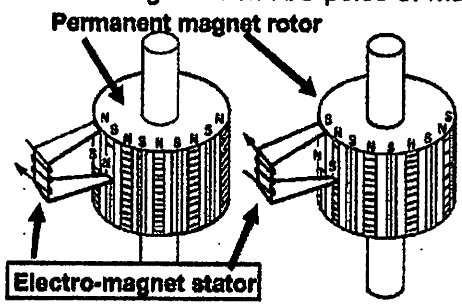
Friday, Nov. 9

Stepper Motors

- stepping motors move in response to a series of electrical pulses, one output "step" per input pulse
- open-loop control (without output position monitoring) is common
- three types of stepping motors are widely available
  - permanent magnet,
  - variable reluctance, and
  - hybrid

Permanent Magnet (or PM) Stepping Motors

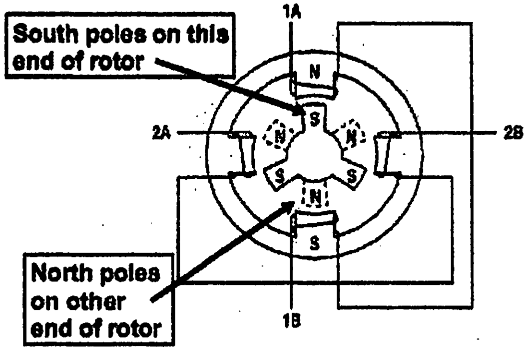
- rotor cross-section is gear shaped
- "teeth" of the gear form N/S poles of magnet



Permanent Magnet (or PM) Stepping Motors

- an electrical circuit alternately switches the polarity of the stator poles
- as the polarity of a stator pole changes, the rotor will move to approach an equilibrium position
- equilibrium positions where N/S rotor poles align with the S/N stator poles

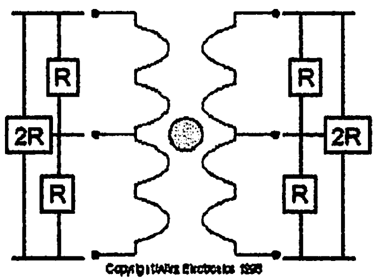
Figure 1.4 - Simple 12 step/rev hybrid motor



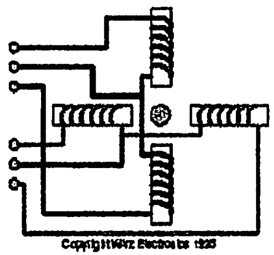
Selection of Stepper Motors

- steps per revolution (or degrees/step)
  - actual output position depends greatly on the static friction in the system
- maximum stepping torque
  - cannot be exceeded or the motor will slip
- Unipolar or Bipolar windings

Unipolar Windings

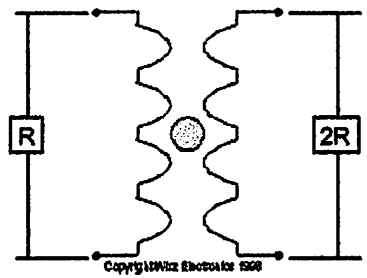


6 leads

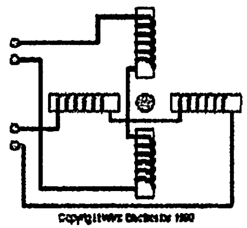


5 leads if two center "taps" are connected

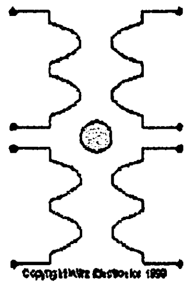
Bipolar Windings



4 leads

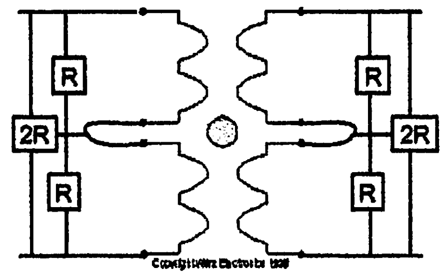


"Universal" Windings



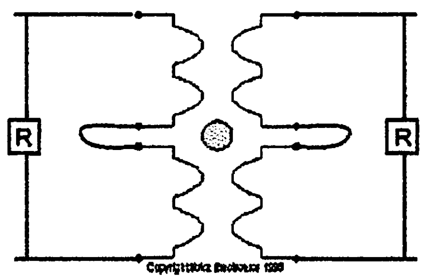
8 leads

"Universal" – Wired Unipolar



"6" leads

"Universal" – Wired Bipolar



"4" leads

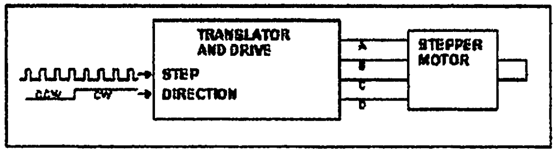
Indexer/Translator/Driver #1

- All step motor systems must consist of an indexer, translator-driver and motor.
- The indexer portion of the system controls timing and direction for each step of motion.
- Under control of the indexer, the translator-driver powers the motor windings so that each step of motion is accomplished

http://www.abilitysystems.com/indexlpt.htm

Indexer/Translator/Driver #2

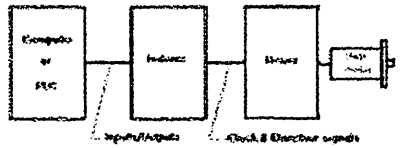
- Stepper motor translator and drive circuits typically require STEP and DIRECTION input signals to operate.
- These signal can be provided by PLC's, stepper indexers or stand-alone digital circuitry.



http://www.ontrak.net/step.htm

Indexer/Translator/Driver #3

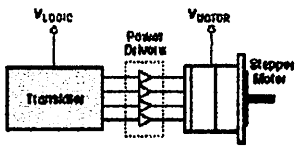
- The computer or PLC (programmable logic controller) sends commands to the indexer.
- The Indexer creates the clock pulses and direction signals.
- The Driver accepts clock pulses and direction signals and translates these signals into appropriate phase currents in the motor.



http://www.anaheimautomation.com/intro.htm

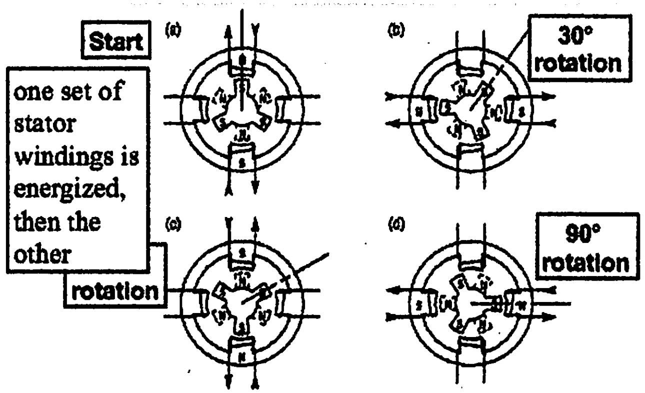
Indexer/Translator/Driver #4

- A circuit which is responsible for converting step and direction signals into winding energization patterns is called a translator.
- Most stepper motor control systems include a driver in addition to the translator, to handle the current drawn by the motor's windings.

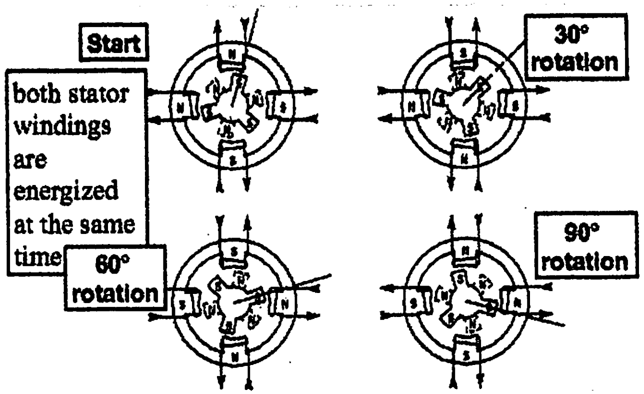


http://www.eio.com/jasstep.htm

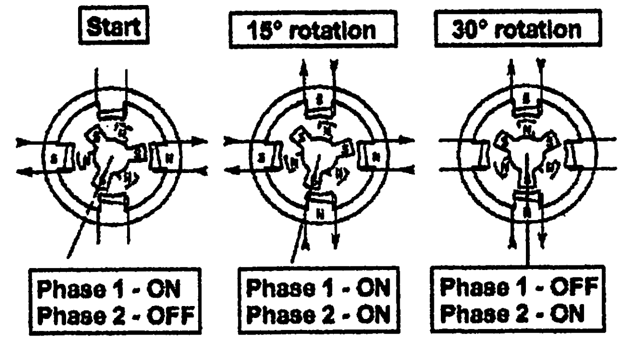
Wave Drive (full steps)



Two Phase Drive (full step)



Half Stepping



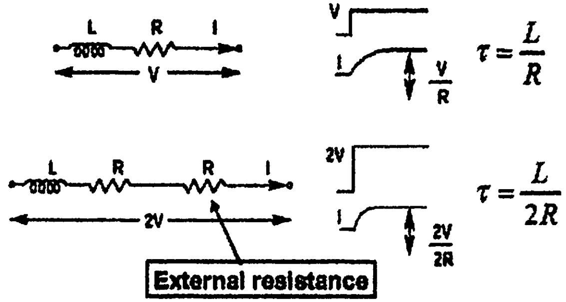
Half Step Mode

- Twice the resolution (steps/rev) from the same motor
- Much better smoothness at low speeds
- Less overshoot and ringing at end of each step
- Slight loss of torque
  - can be improved with the "profiled current" method of Figure 1.10

"Micro" Step Mode

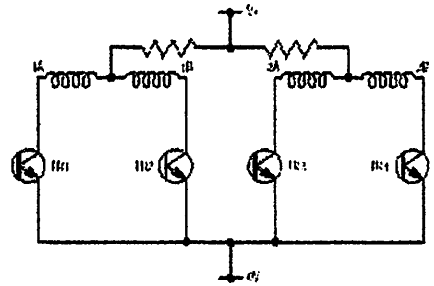
- Ratio the current in each of the two phases
  - rotor will be proportionally attracted to the stator pole with the most current
- 100 to 500 times the resolution (steps/rev) from the same motor
- Very smooth at low speeds
- Much more complicated electronics

Principle of R-L drive



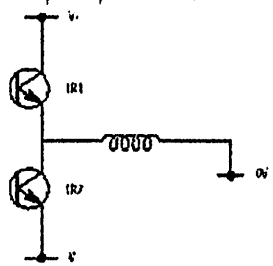
Stepper Motor Drives #1

Fig 2.5 Basic unipolar drive



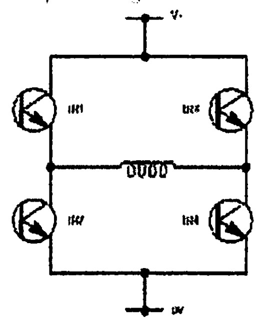
Stepper Motor Drives #2

Fig. 2.6 Simple bipolar drive



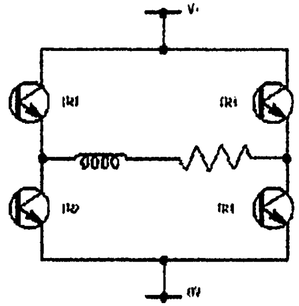
Stepper Motor Drives #3

Fig. 2.7 Bipolar bridge

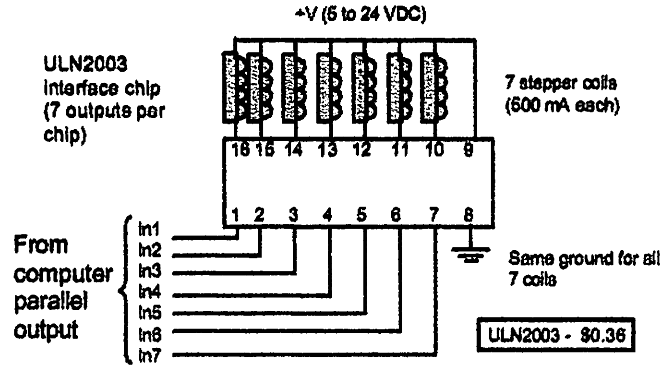


Stepper Motor Drives #4

Fig. 2.8 Bipolar R-L drive

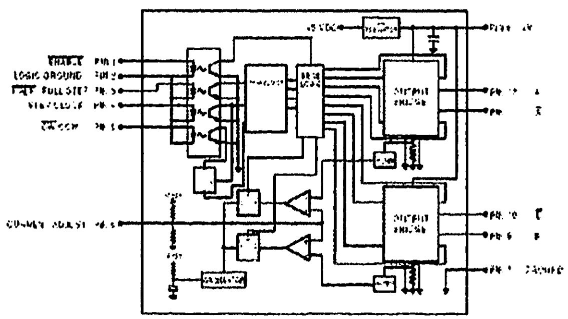


Output Interfacing

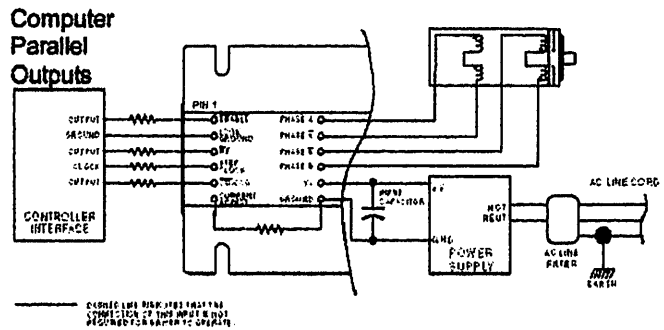


IMS #IB462

2 amp bipolar chopper driver



IMS #IB462, #IB104, #IB483



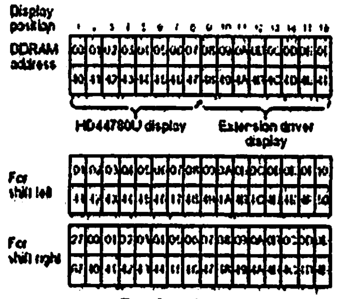
M1632 MODULE LCD 16 X 2 BARIS (M1632)

Deskripsi:

M1632 adalah merupakan modul LCD dengan tampilan 16 x 2 baris dengan konsumsi daya y rendah. Modul ini dilengkapi dengan mikrokontroler yang didisain khusus untuk mengendalikan L(Mikrokontroler HD44780 buatan Hitachi yang berfungsi sebagai pengendali LCD ini mempunyai CGR((Character Generator Read Only Memory), CGRAM (Character Generator Random Access Memory) DDRAM (Display Data Random Access Memory).

DDRAM

DDRAM adalah merupakan memori tempat karakter yang ditampilkan berada. Contoh, ur karakter 'A' atau 41H yang ditulis pada alamat 00, maka karakter tersebut akan tampil pada baris pert dan kolom pertama dari LCD. Apabila karakter tersebut ditulis di alamat 40, maka karakter tersebut a tampil pada baris kedua kolom pertama dari LCD.



Gambar 1
DDRAM M1632 (diambil dari data sheet HD44780)

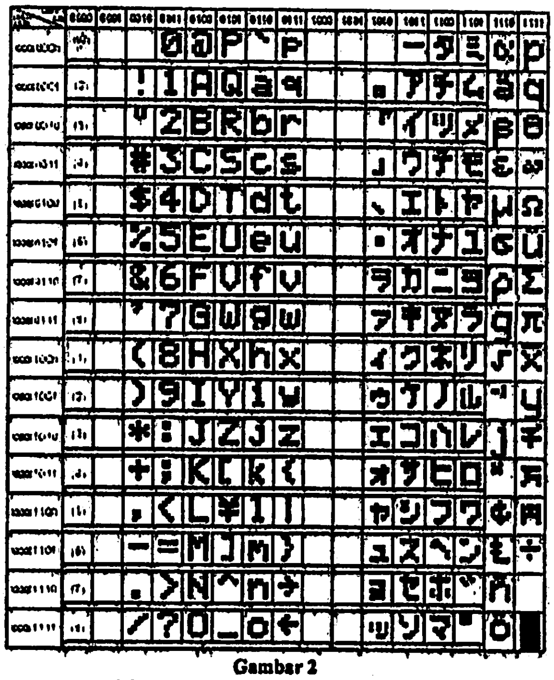
CGRAM

CGRAM adalah merupakan memori untuk menggambarkan pola sebuah karakter di m bentuk dari karakter dapat diubah-ubah sesuai keinginan. Namun memori ini akan hilang saat power sur tidak aktif, sehingga pola karakter akan hilang.

CGROM

CGROM adalah merupakan memori untuk menggambarkan pola sebuah karakter di mana r tersebut sudah ditentukan secara permanen dari HD44780 sehingga pengguna tidak dapat mengubah l Namun karena ROM bersifat permanen, maka pola karakter tersebut tidak akan hilang walaupun po supply tidak aktif

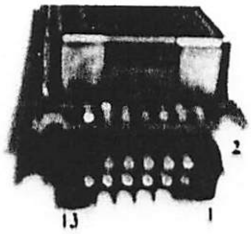
Pada gambar 2, tampak terlihat pola-pola karakter yang tersimpan dalam lokasi-lokasi terte dalam CGROM. Pada saat HD44780 akan menampilkan data 41H yang tersimpan pada DDRAM, m HD44780 akan mengambil data di alamat 41H (0100 0001) yang ada pada CGROM yaitu pola karakter A



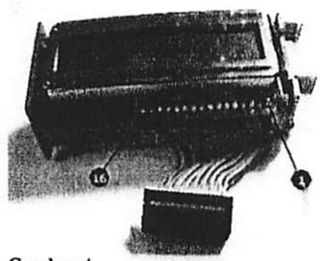
Gambar 2
Hubungan antara CGROM dan DDRAM (diambil dari data sheet HD44780)

Pin Out

No	Nama Pin	Deskripsi
1	VCC	+5V
2	GND	0V
3	VEE	Tegangan Kontras LCD
4	RS	Register Select, 0 = Register Perintah, 1 = Register I
5	R/W	1 = Read, 0 = Write
6	E	Enable Clock LCD, logika 1 setiap kali pengiriman pembacaan data
7	D0	Data Bus 0
8	D1	Data Bus 1
9	D2	Data Bus 2
10	D3	Data Bus 3
11	D4	Data Bus 4
12	D5	Data Bus 5
13	D6	Data Bus 6
14	D7	Data Bus 7
15	Anoda (Kabel coklat untuk LCD Hitachi)	Tegangan positif backlight
16	Katoda (Kabel merah untuk LCD Hitachi)	Tegangan negatif backlight



Gambar 3
Pin Out M1632 LCD Hitachi



Gambar 4
Pin Out LCD M1632 Standard

Register

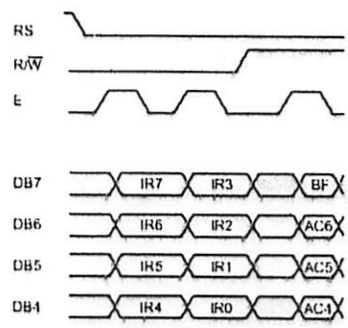
HD44780, mempunyai dua buah Register yang aksesnya diatur dengan menggunakan kaki Pada saat RS berlogika 0, maka register yang diakses adalah Register Perintah dan pada saat RS berlog 1, maka register yang diakses adalah Register Data

Register Perintah

Register ini adalah register di mana perintah-perintah dari mikrokontroler ke HD44780 pada : proses penulisan data atau tempat status dari HD44780 dapat dibaca pada saat pembacaan data.

Penulisan Data ke Register Perintah

Penulisan data ke Register Perintah dilakukan dengan tujuan mengatur tampilan LCD, inisiali dan mengatur Address Counter maupun Address Data. Gambar 5 menunjukkan proses penulisan data register perintah dengan menggunakan mode 4 bit interface. Kondisi RS berlogika 0 menunjukkan ak data ke Register Perintah. RW berlogika 0 yang menunjukkan proses penulisan data akan dilakukan. Nib tinggi (bit 7 sampai bit 4) terlebih dahulu dikirimkan dengan diawali pulsa logika 1 pada E Clo Kemudian Nibble rendah (bit 3 sampai bit 0) dikirimkan dengan diawali pulsa logika 1 pada E Clock l: Untuk mode 8 bit interface, proses penulisan dapat langsung dilakukan secara 8 bit (bit 7 ... bit 0) diawali sebuah pulsa logika 1 pada E Clock.



Gambar 5
Timing diagram Penulisan Data ke Register Perintah Mode 4 bit Interface

Tabel 1
Perintah-perintah M1632

Perintah	D7	D6	D5	D4	D3	D2	D1	D0	Deskripsi
Hapus Display	0	0	0	0	0	0	0	1	Hapus Display dan DDRAM
Posisi Awal	0	0	0	0	0	0	1	X	Set Alamat DDRAM di 0
Set Mode	0	0	0	0	0	1	I/D	S	Atur arah pergeseran cursor dan displ

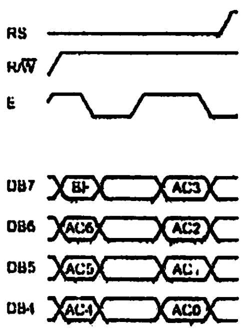
DELTA ELECTRONIC
http://www.delta-electronic.com

Display On/OFF	0	0	0	0	1	D	C	B	Atur display (D) On/OFF, cursor ON/OFF, Blinking (B)
Geser Cursor/Display	0	0	0	1	S/C	R/L	X	X	Geser Cursor atau display tanpa mun alamat DDRAM
Set Fungsi	0	0	1	DL	N	F	X	X	Atur panjang data, jumlah baris : tampil, dan font karakter
Set Alamat CGRAM	0	1	ACG	ACG	ACG	ACG	ACG	ACG	Data dapat dibaca atau ditulis set alamat diatur
Set Alamat DDRAM	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD	Data dapat dibaca atau ditulis set alamat diatur

X = diabaikan
I/D 1=Increment, 0=Decrement
S 0=Display tidak geser
S/C 1=Display Shift, 0=Geser Cursor
R/L 1=Geser Kiri, 0=Geser Kanan
DL 1=8 bit, 0=4bit
N 1=2 baris, 0=1 baris
F 1=5x10, 0=5x8
D 0=Display OFF, 1=Display ON
C 0=Cursor OFF, 1=Cursor ON
B 0=Blinking OFF, 1=Blinking ON

Pembacaan Data dari Register Perintah

Proses pembacaan data pada register perintah biasa digunakan untuk melihat status busy dari L atau membaca Address Counter. RS diatur pada logika 0 untuk akses ke Register Perintah, R/W diatur p logika 1 yang menunjukkan proses pembacaan data. 4 bit nibble tinggi dibaca dengan diawali pulsa lo 1 pada E Clock dan kemudian 4 bit nibble rendah dibaca dengan diawali pulsa logika 1 pada E Cl Untuk Mode 8 bit interface, pembacaan 8 bit (nibble tinggi dan rendah) dilakukan sekaligus dengan diav sebuah pulsa logika 1 pada E Clock.



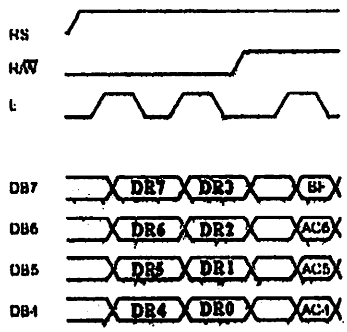
Gambar 6
Timing Diagram Pembacaan Register Perintah Mode 4 bit Interface

Register Data

Register ini adalah register di mana mikrokontroler dapat menuliskan atau membaca data ke dari DDRAM. Penulisan data pada register ini akan menempatkan data tersebut ke DDRAM sesuai den alamat yang telah diatur sebelumnya

Penulisan Data ke Register Data

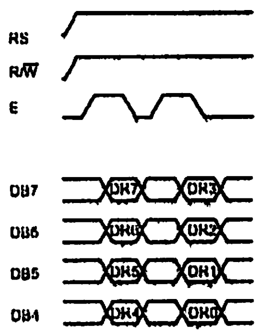
Penulisan data pada Register Data dilakukan untuk mengirimkan data yang akan ditampilkan pa LCD. Proses diawali dengan adanya logika 1 pada RS yang menunjukkan akses ke Register Data, kondisi R/W diatur pada logika 0 yang menunjukkan proses penulisan data. Data 4 bit nibble tinggi (bit 7 hingga bit 4) dikirim dengan diawali pulsa logika 1 pada sinyal E Clock dan kemudian diikuti 4 bit nibble rendah (bit 3 hingga bit 0) yang juga diawali pulsa logika 1 pada sinyal E Clock.



Gambar 7
Timing Diagram Penulisan Data ke Register Data Mode 4 bit Interface

Pembacaan Data dari Register Data

Pembacaan data dari Register Data dilakukan untuk membaca kembali data yang tampil p LCD. Proses dilakukan dengan mengatur RS pada logika 1 yang menunjukkan adanya akses ke Regi Data. Kondisi R/W diatur pada logika tinggi yang menunjukkan adanya proses pembacaan data. Data 4 nibble tinggi (bit 7 hingga bit 4) dibaca dengan diawali adanya pulsa logika 1 pada E Clock dan dilanjut dengan data 4 bit nibble rendah (bit 3 hingga bit 0) yang juga diawali dengan pulsa logika 1 pada E Clock



Gambar 8
Timing Diagram Pembacaan Data dari Register Data Mode 4 bit Interface

Antar muka LCD dengan mikrokontroler

DELTA ELECTRONIC
http://www.delta-electronic.com

photodiodes

Features

- Low-cost visible and near-IR photodetector
- Excellent linearity in output photocurrent over 7 to 9 decades of light intensity
- Fast response times
- Available in a wide range of packages including epoxy-coated, transfer-molded, cast, and hermetic packages, as well as in chip form
- Low noise
- Mechanically rugged, yet compact and lightweight
- Available as duals, quads or as linear arrays
- Usable with almost any visible or near-infrared light source such as solid state laser diodes, neon, fluorescent, incandescent bulbs, lasers, flame sources, sunlight, etc.
- Can be designed and tested to meet the requirements of your application

Typical Applications

- Fiber-Optic Communications
- Instrumentation
- High-Speed Switching
- Spot Position Tracking and Measurement
- Photometry
- Data Transmission
- UV Light Meters
- Fluorescent Light Detection
- Laser Range Finding
- Barcode Scanning
- Laser Safety Scanning
- Distance Measurement

Datasheets available upon request

Description

PerkinElmer Optoelectronics offers a broad array of Silicon and InGaAs PIN and APDs.

InGaAs Avalanche Photodiodes

The high-quality InGaAs avalanche photodiodes (APDs) are packaged in hermetically sealed TO cans and ceramic blocks designed for the 900 to 1700 nm wavelength region.

InGaAs PIN Photodiodes

High-quality Indium Gallium Arsenide photodiodes designed for the 900 to 1700 nm wavelength region, these photodiodes are available in standard sizes ranging from 50 microns to 5 mm in diameter. Packages include ceramic submount, TO packages, and chip form.

Silicon Avalanche Photodiodes

These are reliable, high-quality detectors in hermetically sealed TO packages designed for high-speed and high-gain applications. A "reach-through" structure is utilized which provides very low noise performance at high gains, and a full range of active areas is available.

Silicon PIN Photodiodes

Offered for low- to high-speed applications, these PINs are designed for the 250 nm to 1100 nm range. Standard sizes range from 100 microns to 10 mm in diameter.

Silicon PN Photodiodes

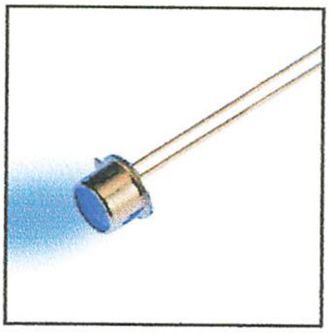
This format includes a variety of high-volume, low-cost silicon photodiodes that meet the demanding requirements of today's commercial and consumer markets.

Alternate Source/Second Source Photodiodes

PerkinElmer's nearest equivalent devices are selected on the basis of general similarity of electro-optical characteristics and mechanical configuration. Interchangeability in any particular application is not guaranteed, suitability should be determined by the customer's own evaluation.

Detector Modules

Preamplifier modules are hybrid devices with a photodiode and a matching amplifier in a compact hermetic TO package. An integral amplifier allows for better ease of use and noise bandwidth performance. 14-pin, DIL, and/or fibered packaged modules are available on a custom basis.



Indium Gallium Arsenide PIN Photodiodes, Large-Area, and Small-Area Indium Gallium Arsenide APDs

- High Responsivity
- Low Capacitance for High Bandwidths
- Available in Various Hermetic Packages

InGaAs APDs—900 nm to 1700 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. µm	Resp. A/W @1300 nm	Resp. A/W @1550 nm	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Bandwidth GHz into 50 W	NEP @ 1550 nm pW/√Hz	VOP for Gain=10 V
C30644E	TO window	50	8.4	9.4	6	0.15	1	2	0.03	40-90
C30644ECER	Ceramic	50	8.4	9.4	6	0.15	0.8	2	0.03	40-90
C30645E	TO window	80	8.4	9.4	10	0.25	1.2	1	0.13	40-90
C30645ECER	Ceramic	80	8.4	9.4	10	0.25	1	1	0.13	40-90
C30662E	TO window	200	8.4	9.4	200	1.4	2.5	0.2	0.15	40-90
C30662ECER	Ceramic	200	8.4	9.4	200	1.4	2.5	0.2	0.15	40-90
C30733ECER	Ceramic	30	8.4	9.4	5	<0.1	0.25	3	0.01	40-90

Test conditions: T = 22°C

InGaAs PIN Large-Area—900 nm to 1700 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. A/W @850 nm	Resp. A/W @1300 nm	Resp. A/W @1550 nm	Dark Curr. Id (nA)	NEP @ 1300 nm pW/√Hz	Cap. @100 kHz Cd (pF)	Bandwidth MHz into 50 W	Max. Power for .15 dB Linearity (dBm)	Bias Volt for these Specs V
C30619G	TO-18	0.5	0.2	0.86	0.95	5	<0.1	8	350	>+13	5
C30641G	TO-18	1	0.2	0.86	0.95	5	<0.1	40	75	>+13	2
C30642G	TO-5	2	0.2	0.86	0.95	10	0.1	350	20	+11	0
C30665G	TO-5	3	0.2	0.86	0.95	25	0.2	1000	3	+11	0
C30723G	TO-8	5	0.2	0.86	0.95	30	0.3	2500	2.5	+11	0

Test conditions: T = 22°C

InGaAs PIN Small-Area—900 nm to 1700 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. µm	Resp. A/W @1300 nm	Resp. A/W @1550 nm	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Bandwidth GHz into 50 W	NEP @ 1550 nm pW/√Hz	Bias Volt for these Specs V
C30616ECER	Ceramic	50	0.86	0.95	0.5	<0.02	0.35	>3.5	<0.02	5
C30637ECER	Ceramic	75	0.86	0.95	0.8	<0.02	0.4	3.5	<0.02	5
C30617ECER	Ceramic	100	0.86	0.95	1	<0.02	0.55	3.5	<0.02	5
C30617B	Ball lens	100	0.8	0.9	1	<0.02	0.8	3.5	<0.02	5
C30618ECER	Ceramic	350	0.86	0.95	2	0.02	4	0.8	0.02	5
C30618G	TO window	350	0.86	0.95	2	0.02	4	0.8	0.02	5

Test conditions: T = 22°C

photodiodes

Silicon Avalanche Photodiodes

- Hermetically Sealed Packages

Si APD—Standard Types–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. 900 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz: Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm fW/√Hz	VOP Range V
C30817E	TO-5	0.8	75	50	0.5	2	2	7	275-425
C30872E	TO-8	3	45	100	0.5	10	2	11	275-425
C30902E	TO-18	0.5	77 (@ 830 nm)	15	0.23	1.6	0.05	3 (@ 830 nm)	180-250
C30902S	TO-18	0.5	128 (@ 830 nm)	15	0.11	1.6	0.05	0.86 (@ 830 nm)	180-250
C30916E	TO-5	1.5	70	100	0.5	3	2	8	275-425

Test conditions: T = 22°C

Si APD—Arrays Quadrant and Linear–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @830 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 830 nm fW/√Hz	VOP Range V
C30927E-01	TO-8	1.5 total	62 (@900 nm)	25	0.25	1	3	16 (@900 nm)	275-425
C30927E-02	TO-8	1.5 total	62 (@900 nm)	25	0.25	1	3	16 (@900 nm)	275-425
C30927E-03	TO-8	1.5 total	62 (@900 nm)	25	0.25	1	3	16 (@900 nm)	275-425
C30985E	Custom	0.3 pitch	31	1	0.1	0.5	2	3	250-425

Test conditions: T = 22°C

Si APD—Low Cost, High Volume–400 nm to 1000 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @900 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm fW/√Hz	VOP Range V
C30724E	TO-18	0.5	9 (@ M=15)	25	0.1	1	5	11	120-200
C30724P	Plastic	0.5	9 (@ M=15)	25	0.1	1	5	11	120-200
C30737E	TO-18	0.5	47 (@ I-800 nm M=100)	20	0.3	2.5	0.3	6.4 (@ 800 nm M=100)	120-200

Test conditions: T = 22°C

Si APD—TE-Cooled

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @830 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 830 nm fW/√Hz	ADP VOP Range V
C30902S-TC	TO-66	0.5	128	2	0.04	1.6	0.5	0.3	160-250
C30902S-DTC	TO-66	0.5	128	1	0.02	1.6	0.5	0.16	160-250

Test conditions: T = 0°C for -TC and -20°C for -DTC

ADP VOP Range: temperature dependent

Silicon Avalanche Photodiodes
- Low Cost, High Volume

Si APD—NIR-Enhanced–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @1060 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm m=15 fW/√Hz	VOP Range V
C30954E	TO-5	0.8	36	50	0.5	2	2	14	275-425
C30955E	TO-5	1.5	34	100	0.5	3	2	15	275-425
C30956E	TO-8	3	25	100	0.5	10	2	20	275-425

Test conditions: T = 22°C

Si APD—Lightpipe

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @830 nm A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 830 nm fW/√Hz	VOP Range V
C30921E	TO-18	0.5	77	15	0.23	1.6	0.05	3	180-250
C30921S	TO-18	0.5	128	15	0.11	1.6	0.05	0.86	180-250

Test conditions: T = 22°C

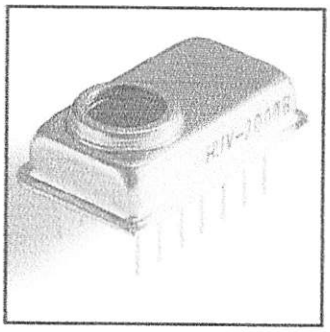
Si APD—Radiation Detection

Technical Specification

Part Number	Photo Sens. Diam. mm	Resp. A/W	Dark Curr. Id (nA)	Spect. Noise Curr. Dens. In (pA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ Peak fW/√Hz	VOP Range V
C30626	5x5	22 (@900 nm)	250	0.5	30	5	23 (@900 nm)	275-425
C30703	10x10	16 (@530 nm)	10	0.7	120	5	40 (@530 nm)	275-425

Test conditions: T = 22°C

photodiodes



Silicon PIN Photodiodes and Modules

- Broad Range of Photosensitive Areas
- Low Operating Voltage
- Hermetically Sealed Packages

Si PINs—Window and Lightpipe Packages, Fast Response–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @830 nm A/W	Dark Curr. Id nA	Spect. Noise Curr. Dens. In (fA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 830 nm fW/√Hz	Bias Volt for These Specs V
C30971E	TO-18	0.5	0.5	10	57	1.6	0.5	113	100
C30971EL	TO-18 Lightpipe	0.25	0.5	10	57	1.6	0.5	113	100

Test conditions: T = 22°C

Si PINs—Large Area, Fast Response–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @900 nm A/W	Dark Curr. Id nA	Spect. Noise Curr. Dens. (fA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm fW/√Hz	Bias Volt for These Specs V
FFD-100	TO-5	2.5	0.58	2	25	8.5	3.5	44	15
FFD-200	TO-8	5.1	0.58	4	36	30	5	62	15

Test conditions: T = 22°C

Si PINs—Quadrant–220 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. total mm	Resp. @900 nm A/W	Dark Curr. Id nA	Spect. Noise Curr. Dens. In (fA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm fW/√Hz	Bias Volt for These Specs V
C30845E	TO-5	8	0.6	7	47	8	6	79	45
UV-140BQ-4	TO-5	1.3x1.3 (x4)	0.58	—	4	34	<1 µsec	7	0
YAG-444-4A	Custom	11.4	0.4 @1.06 µm	40	118	9	25	295	180

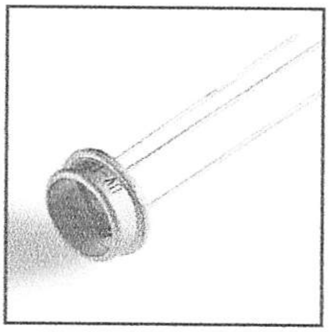
Test conditions: T = 22°C

Si PINs—Standard N-Type–400 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. @900 nm A/W	Dark Curr. Id nA	Spect. Noise Curr. Dens. In (fA/√Hz)	Cap. @100 kHz Cd (pF)	Resp. Time tr (ns)	NEP @ 900 nm fW/√Hz	Bias Volt for These Specs V
C30807E	TO-18	1	0.6	1	18	2.5	3	30	45
C30808E	TO-5	2.5	0.6	3	31	6	5	52	45
C30822E	TO-8	5	0.6	5	40	17	7	67	45
C30809E	TO-8	8	0.6	7	47	35	10	79	45
C30810E	Custom	11.4	0.6	30	98	70	12	163	45

Test conditions: T = 22°C



Silicon PINs—UV Enhanced

Si PINs—UV Enhanced, Low Noise–220 nm to 1100 nm

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. A/W @250 nm	Resp. A/W @900 nm	Shunt Resis. Rd MW	Spect. Noise Curr. Dens.: In (fW/√Hz)	Cap. @100 kHz: Cd (pF)	NEP @ 900 nm fA/√Hz
UV-040BQ	TO-8	1	0.12	0.58	2000	3	25	5
UV-100BQ	TO-8	2.5	0.12	0.58	1000	4	120	7
UV-215BQ	TO-8	5.4	0.12	0.58	250	8	450	25
UV-245BQ	TO-8	4.4x4.7	0.12	0.58	375	7	375	20
UV-140BQ-2	TO-5	2.5x1.3 (x2)	0.12	0.58	1000	4	68	7
UV-140BQ-4	TO-5	1.3x1.3 (x4)	0.12	0.58	1000	4	34	7

Test conditions: T = 22°C

Si PIN Modules—Low Bandwidth–1 kHz to 50 kHz

Technical Specification

Part Number	Standard Package	Photo Sens. Diam. mm	Resp. MV/W @250 nm	Resp. MV/W @900 nm	Spect. Noise Volt. Dens. Vn (µV/√Hz)	NEP @ 900 nm pW/√Hz	Bandwidth kHz into 50 W	Bias Volt for These Specs V
HUV-2000B	Custom	5.4	24	116	2.5	0.02	2	0
HUV-1100BG	TO-5	2.5	24	116	20	0.17	20	0

Test conditions: T = 22°C

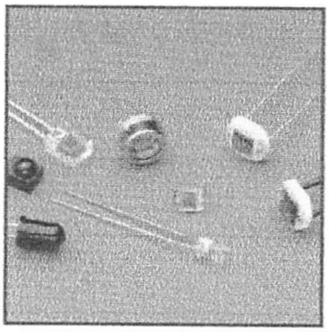
Si PIN Modules—High Bandwidth–40 MHz to 100 MHz

Technical Specification

Part Number	PIN or APD Used	Standard Package	Photo Sens. Diam. mm	Resp. kV/W @900 nm	Lin. Volt. Out Swing (V)	Spect. Noise Volt. Dens. Vn (nV/√Hz)	NEP @900 nm pW/√Hz	Bandwidth MHz (3 dB, into 50 W)	Photo. Diod. Bias Volt V
C30608E	C30971	TO-5	0.5	32 (@ 830 nm)	0.7	60	1.8 (@ 830 nm)	50	12
C30659-1550-R2A	C30662	TO-8	0.2	340 (@ 1550 nm)	2	35	0.103 (@ 1550 nm)	50	40-90
C30950E	C30817	TO-8	0.8	560	0.7	20	.036	50	275-425
C30919E	C30817	Custom	0.8	1000	0.7	25	.025	40	275-425

Test conditions: T = 22°C

photodiodes



Silicon PN Photodiodes

Table Key

I_{SC}	Short-Circuit Current H=100 fc, 2850 K
TC I_{SC}	I_{SC} Temperature Coefficient, 2850 K
V_{OC}	Open-Circuit Voltage H=100 fc, 2850 K
TC V_{OC}	V_{OC} Temperature Coefficient, 2850 K
I_D	Dark Current H=0, VR=10, 50, 100 V
R_{SH}	Shunt Resistance H=0, V=10 mV
C_j	Junction Capacitance H=0, V=0, 3, 15 V
R_E	Responsivity 880-940 nm
S_R	Sensitivity @ Peak
λ_{range}	Spectral Application Range
λ_p	Spectral Response @ Peak
V_{BR}	Breakdown Voltage

Table Key

I_{SC}	Short-Circuit Current H=1000 lux, 2850 K
TC I_{SC}	I_{SC} Temperature Coefficient H=1000 lux, 2850 K
I_D	Dark Current H=0, VR=100 mV
TC I_D	ID Temperature Coefficient H=0, VR=100 mV
R_{SH}	Shunt Resistance H=0, VR=10 mV
C_j	Junction Capacitance H=0, V=0 V, 1 MHz
S_R	Sensitivity @ 400 nm
R_E	Responsivity 400 nm, 0.18 A/W
t_R/t_R	Rise/Fall Time @ 1 KΩ load VR=1 V, 830 nm
V_{OC}	Open-Circuit Voltage H=1000 lux, 2850 K
TC V_{OC}	V_{OC} Temperature Coefficient H=1000 lux, 2850 K

Silicon PN—VTP Series

Technical Specification

Part Number	I_{sc} µA	TC I_{sc} %/°C	V_{oc} mV	TC V_{oc} mV/°C	I_D nA max.	R_{SH} GΩ	C_J pF	R_E A/(W/cm²)	S_R A/W	λ_{range} nm	λ_p nm	V_{BR} V
VTP100	55	0.24	300	-2	30	0.25	50 max.	0.047	0.5	725-1150	925	140
VTP100C	70	0.2	350	-2	30	0.25	50 max.	0.05	0.55	400-1150	925	140
VTP1012	17	0.2	350	-2	7	0.5	6 max.	0.011	0.55	400-1150	925	140
VTP1112	90	0.2	350	-2	7	0.5	6 max.	0.033	0.55	400-1150	925	140
VTP1188S	200	0.2	330	-2	30	67	180	—	0.55	400-1100	925	—
VTP1232	100 min.	0.2	420 min.	-2	25	—	180 max.	0.076	0.6	400-1100	920	—
VTP3310LA	36	0.2	350	-2	35	10	25 max.	0.015	0.55	400-1150	925	140
VTP3410LA	22	0.26	350	-2	35	10	25 max.	0.013	0.55	700-1150	925	140
VTP4085	200	0.2	330	-2	100	2	350	—	0.55	400-1100	925	—
VTP4085S	200	0.2	330	-2	50	4	350	—	0.55	400-1100	925	—
VTP5050	70	0.2	350	-2	18	0.25	24 max.	0.05	0.55	400-1150	925	140
VTP6060	200	0.2	350	-2	35	100	60 max.	0.14	0.55	400-1150	925	140
VTP7110	9	0.2	350	-2	35	7	25 max.	0.015	0.55	400-1150	925	140
VTP7210	7	0.26	350	-2	35	7	25 max.	0.015	0.55	700-1150	925	140
VTP7840	70	0.2	325	-2	20	0.25	40 max.	—	0.55	725-1150	925	1@10 mA
VTP8350	80	0.2	350	-2	30	100	50 max.	0.06	0.55	400-1150	925	140
VTP8440	55	0.2	350	-2	15	0.5	15 max.	0.025	0.55	400-1150	925	140
VTP8551	70	0.2	350	-2	30	0.15	50 max.	0.05	0.55	400-1150	925	140
VTP8651	55	0.24	300	-2	30	0.15	50 max.	0.045	0.5	725-1150	925	140
VTP9412	17	0.2	350	-2	7	0.4	6 max.	0.011	0.55	400-1150	925	140

Electro-optical characteristics @ 25°C

Silicon PN—VTS Series

Technical Specification

Part Number	I_{sc} mA	TC I_{sc} %/°C	I_D nA	TC I_D %/°C	R_{SH} MΩ	C_J nF	S_R A/W	R_E A/(W/cm²)	t_R/t_F µsec	V_{oc} V	TC V_{oc} mV/°C
VTS_80	3	0.2	200	+11	0.3	7.5	0.2	0.7	13	0.45	-2.6
VTS_81	1.5	0.2	100	+11	0.6	3.5	0.2	0.34	6.4	0.45	-2.6
VTS_82	0.69	0.2	50	+11	1.2	1.75	0.2	0.16	3.4	0.45	-2.6
VTS_83	0.64	0.2	50	+11	1.2	1.75	0.2	0.15	3.4	0.45	-2.6
VTS_84	0.33	0.2	40	+11	1.5	1	0.2	0.07	1.8	0.45	-2.6
VTS_85	0.16	0.2	20	+11	3	0.5	0.2	0.04	1.2	0.45	-2.6
VTS_86	0.080	0.2	10	+11	6	0.25	0.2	0.02	0.75	0.45	-2.6

Electro-optical characteristics @ 25°C

Table Key

I_{SC} Short-Circuit Current
940 nm, H=0.5 mW/cm^2 (VTD205, VTD206)
H=5 mW/cm^2, 2850 K (VTD31AA, VTB Series)
100 Lux, 2850 K (VTD34, VTD205K)
100 Lux, 2856 K (VTD206K)

TC I_{SC} I_{SC} Temperature Coefficient
2850 K (VTD31AA, VTD34, VTD34F, VTB Series)
2856 K (VTD205, VTD205K, VTD206, VTD206K)

V_{OC} Open-Circuit Voltage
940 nm, H=0.5 mW/cm^2 (VTD 205, VTD205K, VTD206, VTD206K)
2850 K (VTD31AA, VTD34, VTD34F)

TC V_{OC} V_{OC} Temperature Coefficient
2850 K (VTD31AA, VTD34, VTD34F, VTB Series)
2856 K (VTD205, VTD205K, VTD206, VTD206K)

I_D Dark Current
H=0, V_R=2 V (VTB Series)
H=0, V_R=10 V (VTD34, VTD34F, VTD205, VTD205K, VTD206, VTD206K, VTB100)
H=0, V_R=15 V (VTD31AA)

R_{SH} Shunt Resistance
H=0, V=10 mV (VTB Series)

TC R_{SH} R_{SH} Temperature Coefficient
H=0, V=10 mV (VTB Series)

C_J Junction Capacitance
H=0, V_R=0 V, 1 MHz (VTD205, VTD205K, VTD206, VTD206K)
@ 1 MHz, V_R=0 V (VTD34, VTD34F)
H=0, V=0 V (VTD31AA, VTB Series)

t_R/t_R Rise/Fall Time
@ RL=50 Ω, V_R=5 V, 850 nm (VTD205, VTD205K, VTD206, VTD206K)
@ RL=1 kΩ Lead, V_R=10 V, 833 nm (VTD34, VTD34F)

S_R Sensitivity @ Peak
365 nm (VTB Series)

λ_{range} Spectral Application Range

λ_p Spectral Response @ Peak

V_{BR} Breakdown Voltage

Silicon PN—VTD Series

Technical Specification

Part Number	I_{SC} µA	TC I_{SC} %/°C	V_{OC} mV	TC V_{OC} mV/°C	I_D nA max.	C_J pF	t_R/t_F nsec	S_R A/W	λ_{range} nm	λ_p nm	V_{BR} V
VTD31AA	150-225	0.2	350	-2	50	500 max.	—	0.55	400-1150	860	5 min.
VTD34	70	0.2	365	-2	30	60	50	0.6	400-1100	900	40 min.
VTD34F	—	—	350	-2	30	60	50	0.6	725-1150	940	40 min.
VTD205	25	0.2	350	-2.6	30.	72	20	0.6	800-1100	925	50
VTD205K	80	0.2	365	-2.6	30	72	20	0.6	400-1100	925	50
VTD206	25	0.2	350	-2.6	30	72	20	0.6	750-1100	925	50
VTD206K	80	0.2	365	-2.6	30	72	20	0.6	400-1100	925	50

Electro-optical characteristics @ 25°C

Silicon PN—VTB Series

Technical Specification

Part Number	I_{SC} µA	TC I_{SC} %/°C	V_{OC} mV	TC V_{OC} mV/°C	I_D pA max.	R_{SH} GΩ	TC R_{SH} %/°C	C_J nF	S_R A/W	λ_{range} nm	λ_p nm	V_{BR} V
VTB100	65	0.12	490	-2	500	1.4	-8	2 max.	0.1	320-1100	920	40
VTB1012	13	0.12	490	-2	100	0.25	-8	0.31	0.09	320-1100	920	40
VTB1012B	1.3	0.02	420	-2	100	0.25	-8	0.31	—	330-720	580	40
VTB1013	13	0.12	490	-2	20	7	-8	0.31	0.09	320-1100	920	40
VTB1013B	1.3	0.02	420	-2	20	7	-8	0.31	—	330-720	580	40
VTB1112	60	0.12	490	-2	100	0.25	-8	0.31	0.19	320-1100	920	40
VTB1112B	6	0.02	420	-2	100	0.25	-8	0.31	—	330-720	580	40
VTB1113	60	0.12	490	-2	20	7	-8	0.31	0.19	320-1100	920	40
VTB1113B	6	0.02	420	-2	20	7	-8	0.31	—	330-720	580	40
VTB4051	200	0.12	490	-2	250	0.56	-8	3	0.1	320-1100	920	40
VTB5051	130	0.12	490	-2	250	0.56	-8	3	0.1	320-1100	920	40
VTB5051B	13	0.02	420	-2	250	0.56	-8	3	—	330-720	580	40
VTB5051J	130	0.12	490	-2	250	0.56	-8	3	0.1	320-1100	920	40
VTB5051UV	130	0.12	490	-2	250	0.56	-8	3	0.1	200-1100	920	40
VTB5051UVJ	130	0.12	490	-2	250	0.56	-8	3	0.1	200-1100	920	40
VTB6061	350	0.12	490	-2	2000	0.1	-8	8	0.1	320-1100	920	40
VTB6061B	35	0.02	420	-2	2000	0.1	-8	8	—	330-720	580	40
VTB6061CIE	12	—	—	—	2000	0.1	-8	8	—	475-650	555	—
VTB6061J	350	0.12	490	-2	2000	0.1	-8	8	0.1	320-1100	920	40
VTB6061UV	350	0.12	490	-2	2000	0.1	-8	8	0.1	200-1100	920	40
VTB6061UVJ	350	0.12	490	-2	2000	0.1	-8	8	0.1	200-1100	920	40
VTB8341	60	0.12	490	-2	100	1.4	-8	1	0.1	320-1100	920	40
VTB8440	45	0.12	490	-2	2000	0.07	-8	1	0.1	320-1100	920	40
VTB8440B	5	0.02	420	-2	2000	0.07	-8	1	—	330-720	580	40
VTB8441	45	0.12	490	-2	100	1.4	-8	1	0.1	320-1100	920	40
VTB8441B	5	0.02	420	-2	100	1.4	-8	1	—	330-720	580	40
VTB9412	13	0.12	490	-2	100	0.25	-8	0.31	0.09	320-1100	920	40
VTB9412B	1.3	0.02	420	-2	100	0.25	-8	0.31	—	330-720	580	40
VTB9413	13	0.12	490	-2	20	7	-8	0.31	0.09	320-1100	920	40
VTB9413B	1.3	0.02	420	-2	20	7	-8	0.31	—	330-720	580	40