SKRIPSI

PERANCANGAN DAN PEMBUATAN DISPLAY MATRIKS LED BERBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH

Disusun Oleh :

VICKY AGUS LESMANA

NIM 03.17.070

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK KOMPUTER DAN INFORMATIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2009

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN DISPLAY MATRIKS LED BEBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH

SKRIPSI

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

Disusun Oleh :

VICKY AGUS LESMANA
NIM : 03.17.070

Mengetahui

Ketua Jurusan Teknik Elektro S-1

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

Diperiksa dan Disetujui

Dosen Pembimbing

Joseph Dedy Irawan,ST. MT
NIP. 132 315 178

KONSENTRASI TEKNIK ELEKTRONIKA
JURUSAN TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2009

ABSTRAK

PERANCANGAN DAN PEMBUATAN DISPLAY MATRIKS LED BEBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH

Vicky Agus Lesmana
03.17.070

Konsentrasi Teknik Elektronika, Jurusan Teknik Elektro S-1
Fakultas Teknologi Industri, Institut Teknologi Nasional Malang
Jl. Raya Karanglo Km 2 Malang
e-mail : keelua_501@yahoo.com

Dosen Pembimbing : Joseph Dedy Irawan,ST. MT

kata kunci *: AT89S8252, Bluetooth ,Matriks led*

Abstrak

Dalam kehidupan sehari-hari, kita banyak memanfaatkan peralatan elektronik untuk mempermdah pekerjaan kita, baik didalam menyelesaikan pekerjaan di dalam rumah sampai di luar rumah, misalnya mesin cuci, penghisap debu, telepon, dan lain-lain. Selain itu juga berfungsi sebagai pemberi informasi yang ditampilkan pada matrik led, pada umumnya peralatan digital yang kita pergunakan tersebut dapat kita rubah sesuai informasi yang akan di sampaikan secara, karena terdapat peralatan pengontrol didalamnya, salah satu dari peralatan control tersebut adalah mikrokontroller

Supaya mikrokontroller dapat melakukan proses-proses tersebut, sebelumnya kita harus mengisikan program di dalamnya, dalam proses pengisian program di hubungkan terlebih dahulu ke komputer dengan menggunakan komunikasi pararel melalui pararel port(printer port)atau serial melalui serial port. Hal ini menjadi kendala bagi kita yang menggunakan laptop karena kita tidak menemukan printer port dan serial port, sehingga penulis berusaha membuat suatu display matriks led menggunakan komunikasi Bluetooth, mengingat komunikasi Bluetooth akan berkembang beberapa tahun ke depan .

KATA PENGANTAR

Dengan memanjatkan puji syukur Alhamdulillah kehadirat Allah SWT yang telah melimpahkan rahmat, taufik dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi ini. Skripsi ini merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Komputer dan Informatika ITN Malang dan untuk memperoleh gelar Sarjana Teknik.

Keberhasilan peyelesaian skripsi ini tidak lepas dari bimbingan dan dorongan berbagai pihak. Untuk itu penyusun menyampaikan ucapan terima kasih kepada :

1. Bapak Prof. Dr. Ir. Abraham Lomi MSEE., selaku rektor ITN Malang
2. Bapak Ir. H. Sidik Noertjahjono,MT selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
4. Bapak Joseph Dedy Irawan, ST, MT selaku Dosen Pembimbing
5. Keluarga yang telah memberikan bantuan materi, do'a, dan semangat.
6. Semua yang telah membantu dalam penyelesaian skripsi ini.

Penulis menyadari bahwa skripsi ini jauh dari sempurna. Oleh karena itu, mohon dengan sangat saran dan kritik yang membangun, agar karya tulis ini dapat menjadi lebih baik lagi.Harapan penulis semoga laporan skripsi ini memberikan manfaat bagi semua pembaca.

Malang, September 2009

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

mempunyai kemampuan untuk di tulis ulang hingga 1000 kali dan berisikan perintah standart MCS-51.

Dalam sistem mikrokontroller terdapat dua hal yang mendasar, yaitu :

Perangkat keras dan perangkat lunak yang keduanya saling berkaitan dan mendukung.

2.1.2. Arsitektur AT89S8252

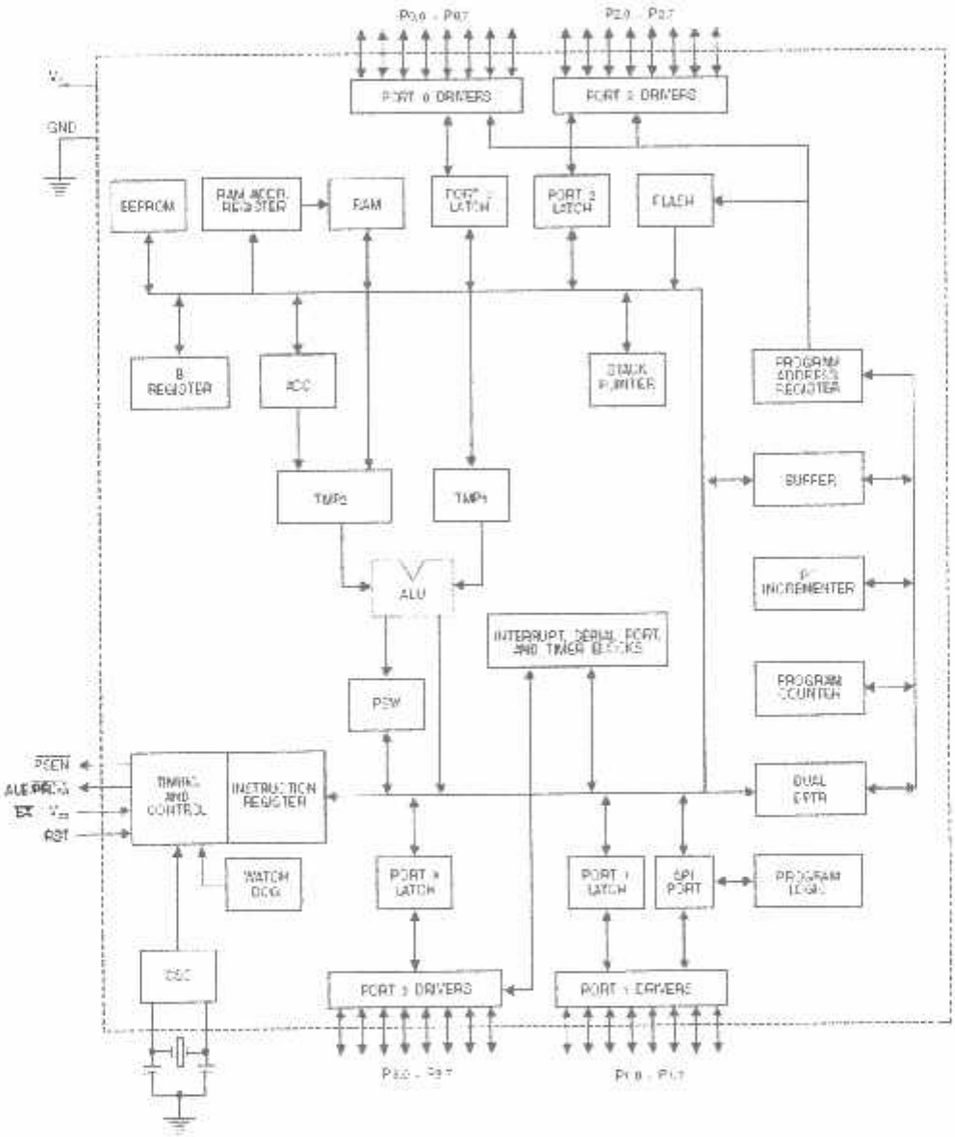
Sebagai *single chip* yaitu suatu sistem mikroprosesor yang terintegrasi mikrokontroller AT89S8252 mempunyai konfigurasi sebagai berikut :

1. *Central Processing Unit* (CPU) 8 *bit* dengan *register* A (*Accumultor*) dan *register* B
2. 16 *bit Program Counter* (PC) dan *Data Pointer* (DPTR)
3. 8 *bit Program Status Word* (PSW)
4. 8 *bit Stack Pointer* (SP)
5. 2 K*byte internal* EEPROM
6. 256 *byte internal* RAM
    - 4 *Bank register* masing-masing 8 *register*
    - 16 *byte* yang dapat dialamati pada *bit level*
    - 208 *byte general purpose memory data*
7. 32 pin input-output tersusun atas P0-P3, masing-masing 8 bit
8. 3 (TO&T1) dengan masing-masing 16 bit *timer / counter*
9. *Receiver register data serial full duplex: Serial Buffer*(SBUF)
10. *Central register* yaitu TCON, TMOP, SCON, PCON, IP dan IE

11. 9 buah sumber interupsi (2 buah *interrupt eksternal* dan 3 buah *interrupt internal*)

12. *Oscillator* dan *clock internal*.

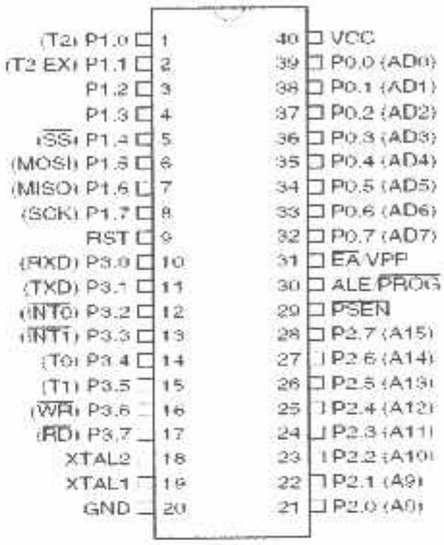
Arsitektur dasar dari mikrokontroller AT89S8252 seperti blok diagram berikut:



Gambar 2.1 Diagram Blok Arsitektur Mikrokontroler AT89S8252 [1]

2.1.3. Konfigurasi pin pada mikrokontroller AT89S8252

Susunan pin-pin mikrokontroller terdiri dari 40 pin telihat pada gambar



Gambar 2.2 Konfigurasi Pin AT89S8252 [1]

Fungsi-fungsi tiap pinnya adalah sebagai berikut :

- VCC (Supply tegangan), pin 40
- GND (*Ground*), pin 20
- Port 0, pin 32-39

  Merupakan port input-output dua arah, tanpa internal pull-up dan konfigurasi sebagai multipleks bus alamat rendah (A_0-A_7) dan data selain pengaksesan program memori dan data memori eksternal.

- Port 1, pin 1-8

  Merupakan port input-output dua arah dengan internal *pull-up*.

- Port 2, pin 21-28

Merupakan port input-output dengan internal *pull-up*. Mengeluarkan alamat tinggi selama pengambilan program memori eksternal.

- Port 3, pin 10-17

  Merupakan port input-output dengan internal pull-up, dimana port 3 juga memiliki fungsi khusus dan dapat dilihat pada table berikut :

Tabel 2.1 Fungsi Khusus Port 3 [11]

Identitas Port	Fungsi Khusus
Port 3.0	RxD (Port masukan khusus)
Port 3.1	TxD (Port keluaran khusus)
Port 3.2	/INT0 (Masukan interupsi eksternal 0)
Port 3.3	/INT1 (Masukan interupsi eksternal 1)
Port 3.4	T0 (Masukan pewaktu eksternal 0)
Port 3.5	T1 (Masukan pewaktu eksternal 1)
Port 3.6	/WR (Sinyal tulis memori data eksternal)
Port 3.7	/RD (Sinyal baca memori data eksternal)

- RST (*Reset*), pin 9

  Input reset merupakan reset master untuk AT89S8252

- ALE / Prog (*Address Lacth Enable*), pin 30

  Digunakan untuk menahan alamat memori eksternal selama pelaksanaan instruksi

- PSEN (*Program Store Enable*), pin 29

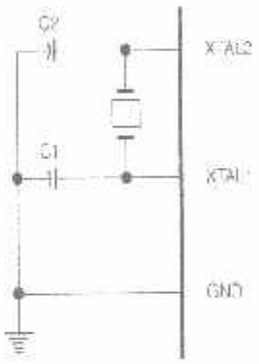
  Merupakan sinyal pengontrol yang memperbolehkan program memori eksternal masuk ke dalam bus

- EA / VPP (*External Access*), pin 31

Dapat diberikan logika rendah (*Ground*) atau logika tinggi (+5 Volt). Jika di berikan logika tinggi maka mikrokontroller akan mengakses program dari ROM internal (EEPROM/*Flash Memory*), dan jika diberikan logika rendah maka mikrokontroller akan mengakses program dari memori eksternal.

- X-TAL 1 dan X-TAL 2, pin 18,19

  Pin ini dihubungkan dengan kristal bila menggunakan oscillator internal. X-TAL 1 merupakan masukan ke rangkaian oscillator internal sedangkan X-TAL 2 keluaran dari rangkaian oscillator internal. Untuk keperluan ini diperlukan kapasitor penstabil sebesar 30pF. Dan nilai dari X-TAL tersebut antara 4-24 MHz. Untuk lebih jelasnya dapat dilihat gambar pemasangan X-TAL serta kapasitor yang digunakannya.



Gambar 2.3 Osilator Eksternal AT89S8252 [11]

2.1.4. Organisasi Memori

Organisasi memori pada mikrokontroller AT89S8252 dapat dibagi menjadi dua bagian besar yaitu memori program dan memori data. Pembagian tersebut didasarkan atas fungsi dari penyimpanan data maupun program. Memori program digunakan untuk

menyimpan instruksi-instruksi yang akan dijalankan oleh mikrokontroller, sedangkan memori data digunakan sebagai tempat yang sedang diolah mikrokontroller.

Program mikrokontroller disimpan dalam memori program berupa ROM. Mikrokontroller AT89S8252 dilengkapi dengan ROM internal, sehingga untuk menyimpan program tidak digunakan ROM eksternal yang terpisah dari mikrokontroller. Agar tidak menggunakan memori program eksternal, pin EA dihubungkan dengan Vcc (logika 1).

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

Selain program mikrokontroller AT89S8252 juga memiliki data internal sebesar 128 Byte dan mampu mengakses memori data eksternal sebesar 64 Kbyte. Semua memori data internal dapat dialamati data langsung atau tidak langsung. Ciri dari pengalamatan langsung adalah *operand* , *operand* adalah alamat register yang berisi alamat data yang akan diolah. Sebagian memori tersebut dapat dialamati dengan pengalamatan register, dan sebagian lagi dapat dialamati dengan memori satu bit. Untuk membaca data digunakan sinyal /RD sedangkan untuk menulis digunakan sinyal /WR.

2.1.5. 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 lacth port, timer, *program status word* (PSW), control peripheral dan sebagainya.

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

- *Accumulator* (ACC) merupakan rgister untuk penambahan dan pengurangan perintah Mneumonic untuk mengakses akumulator disederhanakan menjadi A.
- *Register B* merupakan register khusus yang berfungsi melayani operasi perkalian dan pembagian
- *Stack Pointer* (SP) merupakan register 8 bit yang dapat diletakkkan di alamat manapun pada RAM internal.
- 2 *Data pointer* (DPTR) terdiri atas dua register yaitu untuk byte tinggi (*Data Pointer High,* DPH) dan byte rendah (*Data Pointer Low*, DPL) yang berfungsi untuk mengunci alamat 16 bit.
- *Port 0 sampai port 3* merupakan register yang berfungsi untuk membaca dan mengeluarkan data pada port 0, 1, 2, 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 pelayanan timer/counter terdapat register khusus yaitu register TCON (*Timer/Counter Control*) serta pelayanan port serial menggunakan register SCON (*Serial Port Control*).

Tabel 2.2 Special Function Register (SFR) AT89S8252 [11]

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DF H
0D0H	PSW 00000000					SPCR 00000XX			0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			0CF H
0C0H									0C7H
0B8H	IP XX000000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 00000000		SPSR 00XXXXXX						0AFH
0A0H	P2 11111111								0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111						WMCON 00000010		97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR XXXXXXXX	PCON 0XXX0000	87H

2.1.6. Sistem Interupsi

Mikrokontroller AT89S8252 mempunyai total 6 buah sumber interupsi yang dapat membangkitkan permintaan yaitu INT0, INT1, T0, T1, T2 dan *port serial*.

Saat terjadinya interupsi mikrokontroller secara otomatis akan menuju kc subrutin pada alamat tersebut. Setelah interupsi selesai dikerjakan, mikrokontroller akan mengerjakan program semula. Tiap-tiap sumber interupsi dapat *enable* atau *disable* secara *software*.

Tingkat prioritas semua sumber interupsi dapat diprogram sendiri-sendiri dengan set atau clear bit pada (interup priority). Jika dua permintaan interupsi dengan tingkat prioritas yang berbeda diterima secara bersamaan akan dilakukan polling untuk menentukan mana yang akan dilayani.

Tabel 2.3 Alamat sumber interupsi [11]

Sumber Interupsi	Alamat Awal
Interupt Luar 0 (INT 0)	03_H
Pewaktu / Pencacah 0 (T0)	$0B_H$
Interupt Luar 1 (INT 1)	13_H
Pewaktu / Pencacah 1 (T1)	$1B_H$
Port Serial	23_H

2.2 EEPROM *(Electrically Erasable Programmable Read Only Memory)*

2.2.1 Deskripsi

EEPROM merupakan kependekan dari Electrically Erasable Programmable Read-Only Memory. EEPROM adalah tipe khusus dari PROM (Programmable Read-Only Memory) yang bisa dihapus dengan memakai perintah elektris. Seperti juga tipe PROM lainnya, EEPROM dapat menyimpan isi datanya, bahkan saat listrik sudah dimatikan.

EEPROM sangat mirip dengan flash memory yang disebut juga flash EEPROM. Perbedaan mendasar antara flash memory dan EEPROM adalah penulisan dan penghapusan EEPROM dilakukan pada data sebesar satu byte, sedangkan pada flash

memory penghapusan dan penulisan data ini dilakukan pada data sebesar satu block. Oleh karena itu flash memory lebih cepat.

Dengan ROM biasa, penggantian BIOS hanya dapat dilakukan dengan mengganti chip. Sedangkan pada EEPROM program akan memberikan instruksi kepada pengendali chip supaya memberikan perintah elektronis untuk kemudian mendownload kode BIOS baru untuk dikirimkan kepada chip. Hal ini berarti perusahaan dapat dengan mudah mendistribusikan BIOS baru atau update, misalnya dengan menggunakan disket. Hal ini disebut juga flash BIOS.

2.3 Teknologi Bluetooth

Bluetooth adalah sebuah teknologi komunikasi wireless (tanpa kabel) yang beroperasi dalam pita frekuensi 2,4 GHz *unlicensed ISM (industrial, Scientific and Medical)* dengan menggunakan sebuah *frequency hopping transceiver* yang mampu menyediakan layanan komunikasi data dan suara secara *real time* antara *host-host Bluetooth* dengan jarak jangkauan layanan yang terbatas. *Bluetooth* sendiari dapat berupa *card* yang bentuk dan fungsinya hampir sama dengan *card* yang digunakan untuk *wireless local area network (WLAN)* dimana menggunakan jangkauan jarak layanan yang lebih pendek dan kemampuan *transfer* data yang lebih rendah. Pada dasarnya *Bluetooth* diciptakan bukan hanya menggantikan atau menghilangkan penggunaan kabel di dalam melakukan pertukaran informasi, tetapi juga mampu menawarkan fitur yang baik untu teknologi *mobile wireless* dengan biaya yang relatif rendah, konsumsi daya yang rendah, *interoperability* yang menjanjikan, mudah dalam pengoperasian dan mampu

BAB I
PENDAHULUAN

1.1. LATAR BELAKANG

Sehubungan dengan perkembangan pola pikir manusia ilmu pengetahuan dan teknologi ternyata mengalami kemajuan terus menerus. Perbaikan terhadap teknologi yang sudah ada terus dilakukan agar menjadi lebih mudah. Salah satu bidang teknologi yang mengalami perkembangan lebih pesat adalah teknologi elektronika yang tidak terlepas dari tuntutan masyarakat yang terus – menerus berkembang sesuai dengan kondisi dan situasi yang dihadapi.

Seiring dengan perkembangan teknologi yang semakin berkembang, maka pemanfaatan teknologi dapat diterapkan dalam berbagai bidang, salah satunya pada bidang elektrikal. Aplikasi yang dapat dibuat bertujuan untuk memudahkan penggantian tampilan pada matriks led dengan menggunakan komunkasi bluetooth.

Karena saat ini masih banyak aplikasi ini yang komunikasinya menggunakan kabel serial port, belum banyak yang menggunakan media komunikasi bluetooth. Dengan menggunakan komunikasi bluetooth pada aplikasi ini maka beberapa keuntungannya antara lain dapat mengurangi pemakaian kabel pada saat instalasi aplikasi ini sehingga pemasangan instalasi terlihat lebih rapi.

Dalam proses penggantian tampilan pada matriks led dihubungkan terlebih dahulu ke komputer dengan menggunakan komunikasi paralel melalui paralel port (Printer Port) atau serial melalui serial port. Hal ini menjadi kendala bagi kita yang menggunakan laptop.

Karena kita tidak menemukan priter port dan serial port, sehingga penulis berusaha membuat suatu aplikasi menggunakan koneksi Bluetooth, mengingat komunikasi Bluetooth diperkirakan masih akan berkembang beberapa tahun kedepan.

1.2. RUMUSAN MASALAH

Dalam perancangan dan pembuatan display matriks led berbasis mikrokontroller AT89S8252 dengan menggunakan komunikasi bluetooth maka dapat dirumuskan beberapa masalah yang akan dibahas yaitu:

1. Bagaimana membentuk karakter dan angka pada matrik led
2. Bagaimana cara mengirim data dari laptop ke matrik led melalui bletooth
3. Bagaimana menampilkan karakter dan angka pada matriks led

1.3. BATASAN MASALAH

Agar pembahasan dari perancangan dan pembuatan display matriks led berbasis mikrokontroller AT89S8252 dengan menggunakan komunikasi bluetooth ini tidak terlalu meluas maka penyusun perlu membuat batasan-batasan masalah yang meliputi :

1. Bluetooth yang digunakan adalah easy Bluetooth

2. *Dot* Matriks LED Display 5 x 7 (16 character).
3. Karakter yang ditampilkan hanya sampai 32 karakter
4. Catu daya +5volt sampai +12volt

1.4. TUJUAN

Tujuan yang ingin dicapai oleh penulis dari pembuatan dan penyusunan tugas akhir ini adalah:

1. Mengerti cara kerja dan penggunaan *bluetooth*
2. Mengerti cara kerja dan penggunaan LED Dot matriks.
3. Merancang LED Dot matriks yang dikendalikan dengan Mikrokontroler dapat bekerja dengan baik.
4. Memahami sistem komunikasi antara Mikrokontroler dengan perangkat lain.
5. Membuat pemasangan display matriks led lebih terlihat rapi tanpa adanya kabel yang telah digantikan oleh bluetooth.

1.5. METODELOGI PEMBAHASAN

Adapun metode pembahasan yang digunakan dalam menyusun dan menganalisa tugas skripsi ini:

1. Study literature yang berhubungan dengan perancangan dan pembuatan sistem dan mempelajari serta memahami dasar-dasar Mikrokontroler entah itu *Software* maupun *Hardware*.

2. Perencanaan Sistem

   Merencanakan peralatan yang akan dirancang baik *Software* maupun *Hardware* sesuai dengan perencanaan.

3. Pembuatan Sistem

   Membuat sistem yang akan bekerja sesuai dengan apa yang direncanakan.

4. Pengujian Sistem

   Pengujian sistem yaitu untuk mengetahui sistem kerja dari alat dengan baik secara *Software* maupun *Hardware*.

1.6. SISTEMATIKA PENULISAN

Sistematika pembahasan dari skripsi ini terdiri dari pokok pembahasan yang saling berkaitan antara satu dengan lainnya, yaitu :

BAB I : PENDAHULUAN

Pada bab ini dibahas tentang latar belakang permasalahan, rumusan masalah, batasan masalah, sistematika pembahasan dari alat yang direncanakan.

BAB II : LANDASAN TEORI

Pada bab ini membahas tentang teori dasar alat-alat yang digunakan dalam perencanaan dan pembuatan sistem yaitu Mikrokontroler, *Hardware* dan teori dasar alat – alat pendukung lainnya.

BAB III : PERENCANAAN DAN PEMBUATAN ALAT

Pada bab ini dibahas tentang perencanaan dan pembuatan keseluruhan sistem perangkat keras (hardware) dan perangkat lunak (software).

BAB IV : PENGUJIAN ALAT

Pada bab ini dibahas tentang proses serta hasil dari pengujian alat.

BAB V : PENUTUP

Pada bab ini akan disampaikan kesimpulan dari perencanaan dan pembuatan sistem ini.

Bab II

LANDASAN TEORI

Landasan teori sangat membantu untuk dapat memahami suatu sistem. Selain daripada itu dapat juga di jadikan sebagai bahan acuan di dalam merencanakan suatu sistem. Dengan pertimbangan hal-hal tersebut, maka landasan teori merupakan bagian yang harus dipahami untuk pembahasan selanjutnya.

2.1. Mikrokontroller AT89S8252

2.1.1. Deskripsi

Perbedaan mendasar antara mikrokontroller dengan mikroprosesor adalah mikrokontroller selain memiliki *Central Processing Unit* (CPU) juga dilengkapi dengan memori, input-output yang merupakan kelengkapan sebagai minimum sistem mikrokomputer sehingga sebuah mikrokontroller dapat dikatakan sebagai mikrokomputer dalam keping tunggal (*Single Chip Microcomputer*) yang dapat berdiri sendiri.

Mikrokontroller AT89S8252 adalah mikrokontroller ATMEL yang kompatibel penuh dengan mikrokontroller keluarga MCS-51, membutuhkan daya yang rendah, memiliki *performance* yang tinggi dan merupakan mikrokomputer 8 bit yang dilengkapi 8Kbyte flash PEROM (*Programable and Eraseable Read Only Memory*) yaitu ROM yang dapat ditulis menggunakan programmer. Serta 2 Kbyte EEPROM (*Electrical Eraseale Programable Read Only Memory*) internal. Program memori dapat diprogram ulang dalam sistem atau dengan menggunakan *programmer Nonvolately* yang

menyediakan layanan yang bermacam-macam. *Bluetooth* menggunakan salah satu dari dua jenis frekuensi *Spread spectrum Radio* yang digunakan untuk kebutuhan *wireless*. Jenis frekuensi yang digunakan adalah *Frequecy Hopping Spread Spedtrum(FHSS)*, sedangkan yang satu lagi yaitu *Direct Sequence Spread Spectrum (DSSS)* digunakan oleh IEEE802.11xx. *Transceive* yang digunakan oleh *Bluetooth* bekerja pada frekuensi 2,4 GHz *unlicensed ISM (Industrial, Scientific, and Medical)*.

2.2.1. Perkembangan Sejarah Perangkat Bluetooth

Nama Bluetooth berawl dari proyek prestisius yang dipromotori oleh perusahaan-perusahaan raksasa internasional yang bergerak di bidang telekomunikasi dan computer, *di antaranya Ericsson, IBM, Intel, Nokia, dan Toshiba*. Proyek ini di awl tahun 1998 dengan kode nama Bluetooth, karena terinspirasi oleh raja Viking (Denmark) yang bernama Harald Blatand. Raja Harald Blatand ini berkuasa pada abad ke-10 denan menguasai sebagian besar daerah Denmark dan daerah Skandinavia pada masa itu. *Dikarenakan daerah kekuasaannya yang luas*, raja Harald Blatand ini membiayai para ilmuwan dan insinyur untuk membangun sebuah proyek berteknologi metamorphosis yang bertujuan untuk mengontrol pasukan dari suku-suku di daerah Skandinavia tersebut dari jarak jauh. Maka untuk menghormati raja Viking tersebut, yaitu Blatand yang berarti Bluetooth (dalam bahasa inggris) proyek ini diberi nama. Berikut adalah table perkembangan teknologi Bluetooth:

Tabel 2.4. Tabel Perkembangan Teknologi Bluetooth

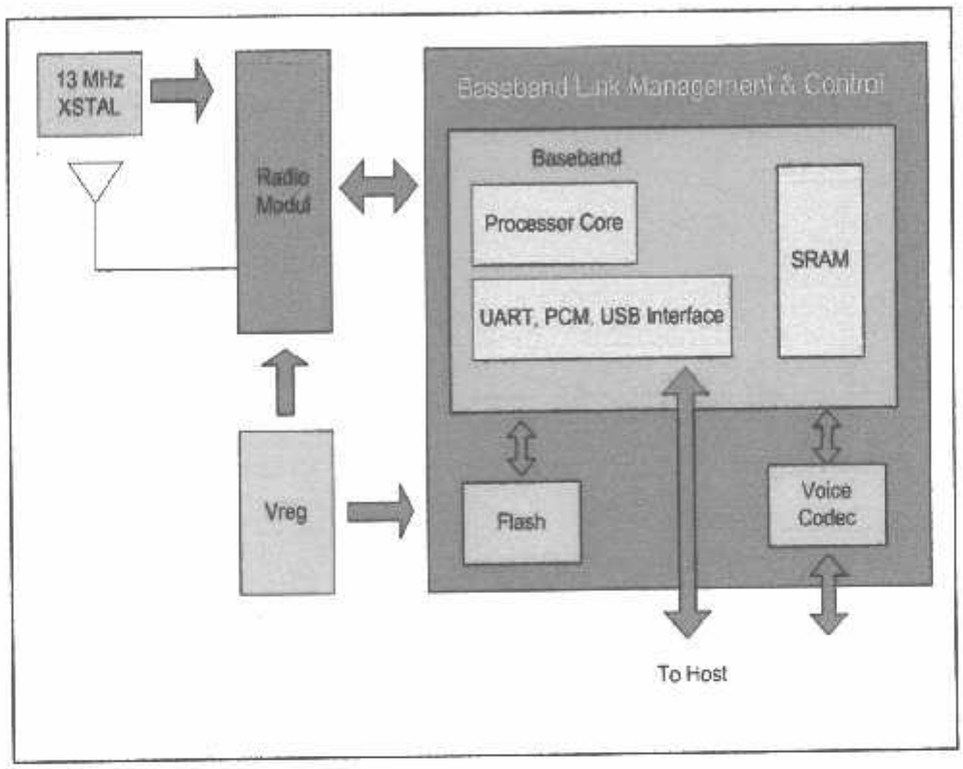
Tahun	Versi	Keterangan
Juli, 1999	1.0 dan 1.0 B	▪ Dibutuhkan perintah manual pada Hardware Device Addres (BD-ADDR) transmisi saat proses koneksi di antara dua device dalam satu jaringan (handshaking process). ▪ Keamanan pengguna tidak terjamin. ▪ Penggunaan protokol tanpa nama (anonymite mode) tidak dimungkinkan).
Oktober, 1999	1.1 dan 1.2	▪ Digunakannya masks pada perangkat Hardware Device Address (BD-ASSRS) untuk melindungi pengguna dari identity snooping (pengintai) maupun tracker. ▪ Penggunaan protokol tanpa nama (anonymite mode) sudah tersedia namun tidak di implementasikan, sehingga konsumen biasa tidak dapat menggunakannya. ▪ Adaptive Frequency Hopping (AFH), dengan memperbaiki daya tahan dari gangguan frekuensi radio yang digunakan oleh banyak orang di dalam hopping sequence.
	2.0	▪ Diperkenalkan Non –hopping narrowband channels. Pada channel ini bias digunakan untuk memperkenalkan layanan profile Bluetooth oleh berbagai device dengan volume yang sangat tinggi dari perangkat Bluetooth secara simultan. ▪ Tidak dienkripsinya informasi yang bersifat umum secara realtime, sehingga dasar kemacetan trafik informasi dan laju trafik ke

		tujuan dapat dihindari waktu ditransmisikan oleh perangkat dengan melewati setiap host dengan kecepatan tinggi ▪ Koneksi berkecepatan tinggi. ▪ Multiple speeds level.

2.2.2. Aplikasi dan Layanan Teknologi Bluetooth

Protokol Bluetooth menggunakan *sebuah kombinasi antara circuit switch*ing *dan packet switching*. Bluetooth dapat mendukung sebuah kanal data asinkron, tiga kanal suara sinkron simultan atau sebuah kanal dimana secara bersamaan mendukung layanan data asinkron dan suara sinkron. Setiap kanal suara mendukung sebuah kanal suara sinkron 64 kb/s. kanal asikron dapat mendukung kecepatan maksimal 723, 2 kb/s asimetris, dimana untuk arah sebaliknya dapat mendukung sampai dengan kecepatan 57,6 kb/s. sedangkan untuk mode simetris dapat mendukung sampai dengan kecepatan 433,9 kb/s. sebuah perangkat yang memiliki teknologi wireless Bluetooth akan mempunyai kemampuan untuk melakukan pertukaran informasi dengan jarak jangkauan sampai dengan 10 meter (~30 feet), bahkan untuk daya kelas 1 bisa sampai pada jarak 100 meter. Sistem Bluetooth terdiri dari sebuah radio transceiver, baseband link Management dan Control, Baseband (processor core, SRAM, UART, PCM USB Interface), flash dan voice code. sebuah link manager. Baseband link controller menghubungkan perangkat keras radio ke baseband processing dan layer protokol fisik. Link manager melakukan akitivitas-aktivitas protokol tingkat tinggi seperti melakukan link setup, autentikasi dan konfigurasi. Secara umum

blok fungsional pada system Bluetooth secara umum dapat dilihat pada Gambar dibawah ini.

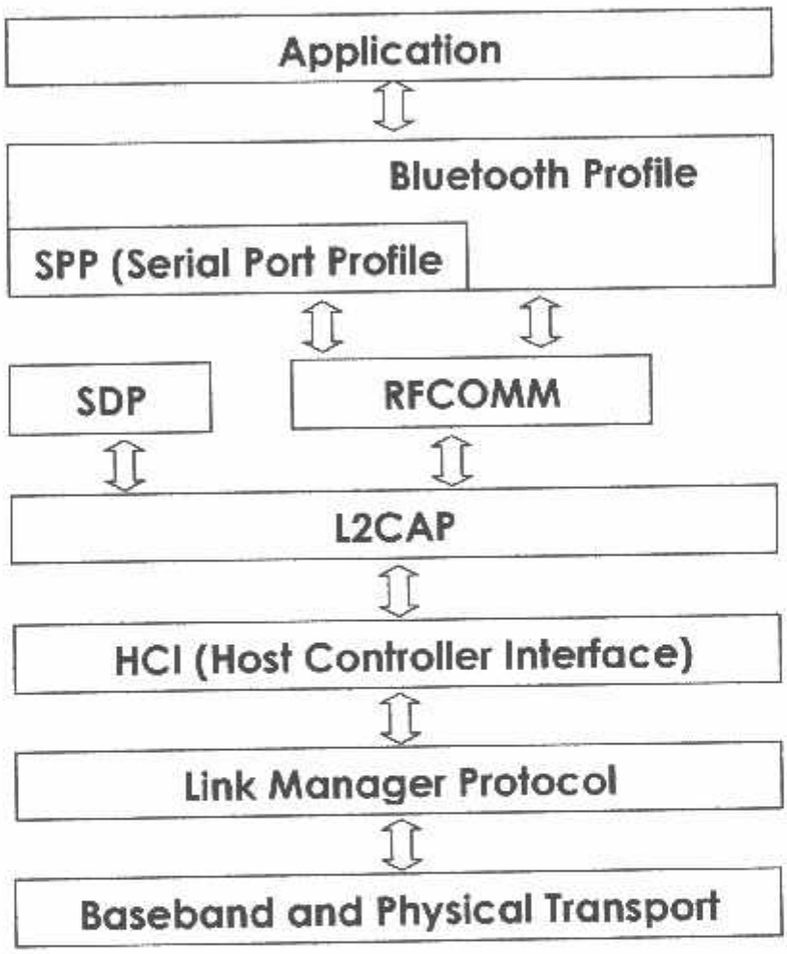


Gambar 2.3. Blok Fungsional Sistem Bluetooth

2.2.3. Profil Bluetooth

Peralatan Bluetooth dapat mensupport interoperabilitas dengan satu atau lebih peralatan Bluetooth. Agar dua perangkat Bluetooth dapat berkomunikasi satu sama lain, maka mereka harus melakukan sharing paling sedikit dengan sebuah common profil. Sebagai contoh, jika kita ingin agar PDA (Personal Digital Asistant) kita dapat berkomunikasi dengan sebuah modul Bluetooth (easy bluetooth), maka kita harus dapat memastikan bahwa kedua perangkat Bluetooth

tersebut telah mensupport profil yang sama. Perangkat modul bluetooth (easy bluetooth) telah mensupport *Serial Port Profil* (*SPP*) yang mana merupakan satu dari profil paling atas dan merupakan sebagian profil yang disupport Bluetooth.



Gambar 2.4. Elemen Utama Bluetooth

Sebagian sebuah typical diagram dari sebuah TCP/IP, terdapat bagian-bagian detail yang tersembunyi oleh bagian tumpukan (stack) yang terlihat sederhana, atau lebih spesifiknya bahwa terdapat bagian dari sebuah profil yang terletak diatas layer L2CAP

yang menyediakan power yang banyak dan lebih komplek pada sebuah protocol Bluetooth. Profil tersebut merupakan jalan masuk utama menuju stack untuk sebuah aplikasi. Terutama ketika aplikasi tersebut mendefinisikan set dari sebuah layanan yang diinginkan oleh sebuah aplikasi. Sekarang ini terdapat lebih dari 25 profil berbeda yang diterapkan oleh Bluetooth SIG *(Special Interest Group)*. Dengan banyaknya jenis profil tersebut, untuk memperolch beberapa seluk-beluk pengertian dari Bluetooth adalah bukan merupakan tugas yang sederhana. Akan tetapi, secara singkat pengertain bahwa sebuah sebuah single profile dapat disediakan untuk sebuah aplikasi yang menggunakan profil tersebut tanpa harus tahu secara detail.

2.3 Modul easy bluetooth

Modul easy bluetooth memiliki dua mode operasi yaitu data mode dan command mode. Ketika power di hidukan maka easy bluetooth akan masuk pada kondisi command mode dan siap menerima instruksi-instruksi serial.

2.3.1. Command Mode

Pada mode ini terdapat bagian perintah yang dapat dikirim untuk mengubah nilai baudrate, menandai perangkat lain yang berada dalam jangkauan, mengecek versi program, dan sebagainya. Semua perintah dikirimkan menggunakan kode-kode ASCII. Ketika perintah pengiriman suksess dijalankan, maka ACK akan dikembalikan, dan jika terjadi masalah pada syntax dalam sebuah transmisi maka NAK akan di-return, setelah salah satu dari NAK atau ACK di return maka karakter Carriage- return <CR> akan dikembalikan. Ketika prompting di return

itu berarti bahwa easy bluetooth berada pada kondisi idle dan menunggu perintah lainnya.

2.3.2. Data Mode

Sekali easy bluetooth terkoneksi pada perangkat Bluetooth lainnya, maka easy bluetooth akan di switch pada kondisi data mode secara otomatis. Semua data yang dikirim pada mode ini akan dikirim pada device dan kemudian tidak akan ada perintah yang dapat dikirim lagi sampai easy bluetooth di matikan atau switch pada keadaan command mode dengan menggunakan mode control I/O. Status koneksi easy bluetooth dapat dimonitor untuk mengetahui bahwa koneksi sedang aktif (terjadi) antara easy bluetooth dengan perangkat Bluetooth lainnya, indicator ini dapat diketahui pada led yang menyala pada modul easy bluetooth.

2.3.3. Spesifikasi easy bluetooth

Komunikasi *Bluetooth* yang digunakan easy bluetooth memiliki karakteristik sebagai berikut:

- Kekuatan pengiriman sinyal sebesar 4dBm (maksimum).
- Menggunakan tipe komunikasi serial dengan jangkauan komunikasi pada lapangan terbuka dapat lebih dari 30 meter (328 kaki).
- Easy bluetooth dapat bekerja dengan baik pada temperature 0° sampai 45°C
- *Supply* power sebesar 3,3 hingga 5,5 VDC

Pin yang dipakai adalah VSS, VIN, RX, TX, dan *Connection status*. Pin RX *Flow* dan TX *Flow* tidak digunakan karena format data serial yang dipakai tidak menggunakan sistem *flow control*. Sedangkan *pin Mode Control* tidak digunakan, karena proses perubahan mode pada easy bluetooth dapat dilakukan secara *software*. Komunikasi yang digunakan antara easy bluetooth dan mikrokontroller adalah serial TTL (Standar Pabrik = 9600 *Baud*, 8 *Data Bits*, 1 *Stop Bits*, *No Parity*, dan *No Flow Control*) dimana *baud rate* dan konfigurasi *flow control*-nya dapat di modifikasi sesuai keinginan pemrogram. Berikut ini adalah konfigurasi pin-pin pada modul easy bluetooth :

Tabel 2.5. Konfigurasi Pin Easy Bluetooth

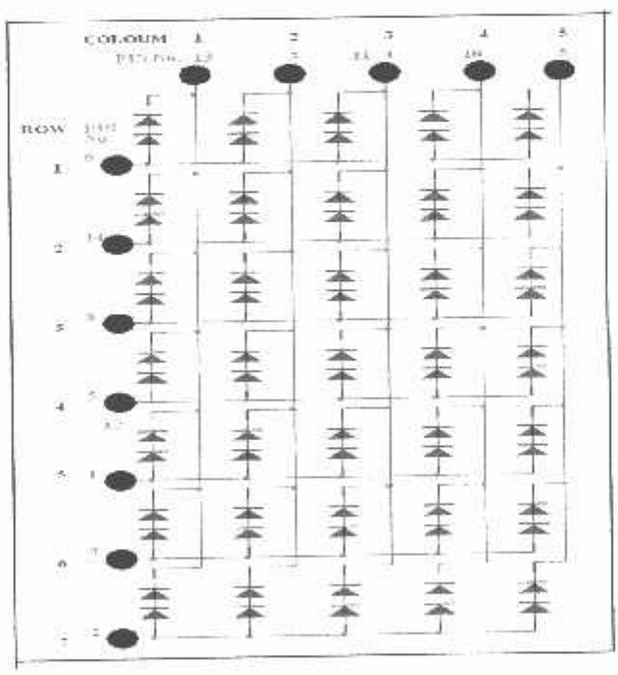
Pin	Name	Function
1	Vdd	+3.3 to +5 VDC (+5.5 max voltage)
2	RX	Receive Serial Communication (CMOS & TTL Level Compatible)
3	TX	Transmit Serial Communication (CMOS & TTL Level Compatible)
4	NC	Not Connected
5	NC	Not Connected
6	NC	Not Connected
7	NC	Not Connected
8	NC	Not Connected
9	NC	Not Connected
10	Vss	Ground

2.4 DOT MATRIKS LED

2.4.1. Dot Matriks LED

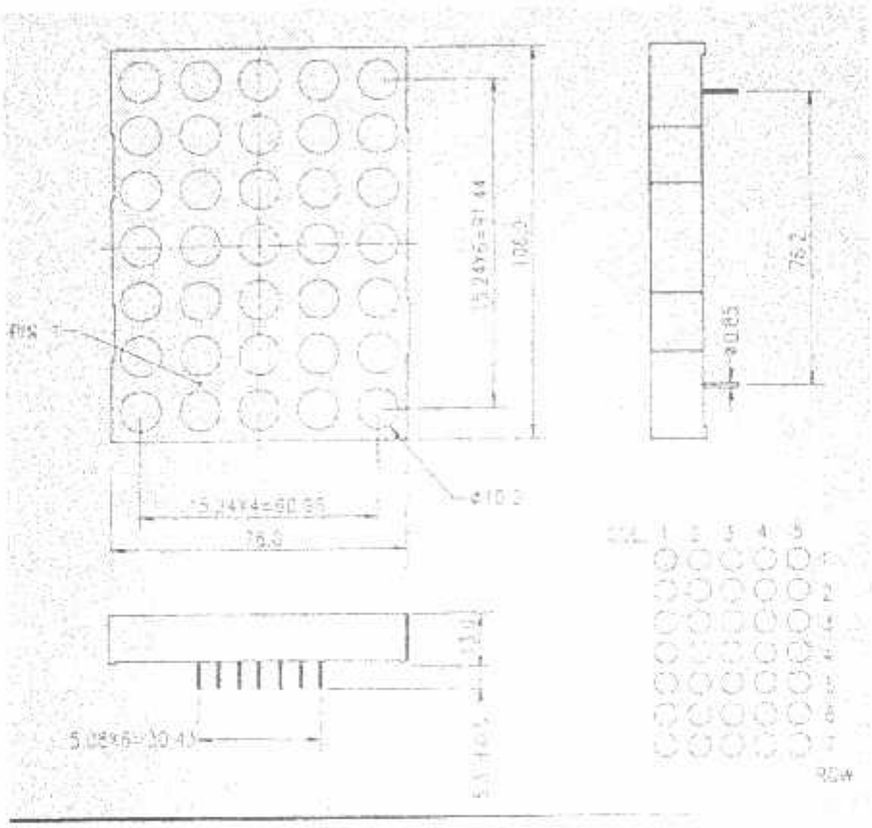
Pada umumnya *Dot* Matriks di difinisikan sebagai suatu susunan *data* dalam bentuk baris dan kolom, dan *data – data* yang ada pada *dot* matriks tersebut saling berkaitan satu dengan yang lainnya.

Piranti *dot* matriks mempunyai sejumlah besar sumber cahaya yang berbentuk seperti titik. Contoh umum adalah matriks 5 x 7 yang terlihat pada gambar 2.23 berikut. Untuk menghidupkan sebuah LED dalam matriks ini, harus menarapkan tegangan dalam anodanya dan mengroundkan lebih dari satu baris, maka dapat memperagakan setiap angka decimal, setiap huruf abjad, serta berbagai karakter lain.



Gambar 2.5. Internal Circuit Diagram Dot Matriks 5x7

Tampilan matriks titik (*Dot Matrix*) terdiri atas sejumlah LED yang disusun dalam baris dan kolom. Susunan yang paling sering digunakan adalah 5 x 7, yaitu lima kolom dan tujuh baris yang jumlah titik – titiknya adalah 35 titik, seperti pada gambar 2.9 berikut.



Gambar 2.6. Package Dimension Dot matriks 5x7 (2")

Matriks 5 x 7 dapat digunakan untuk menyajikan karakter alfanumerik yang lengkap. Proses pembangkitan karakter antara lain melibatkan proses scanning baris atau kolom, memilih LED yang tepat dalam baris atau kolom dan menyalakannya. Proses yang ini diulang untuk baris atau kolom berikutnya. Setelah semua baris atau kolom dipilih

dengan urutan tertentu, proses diatas diulang mulai dari baris paling atas atau kolom pertama.

Jika frekuensi scanningnya cukup cepat (sekitar 50Hz), akan diperoleh karakter bebas kedip. Jika matriks discan dari kiri ke kanan, kolom demi kolom disebut *Vertical scanning*, dan jika dilakukan baris demi baris disebut *Horisontal scanning*.

2.4.2. Dekode Matriks LED

Untuk menyalakan suatu huruf alfanumerik dengan pembacaan dot matriks, maka LED yang dibutuhkan tidak dinyalakan dengan serentak atau bersamaan, proses ini dilakukannya yaitu dengan proses scanning pada baris dan kolom sehingga dapat membentuk suatu huruf alfanumerik atau karakter.

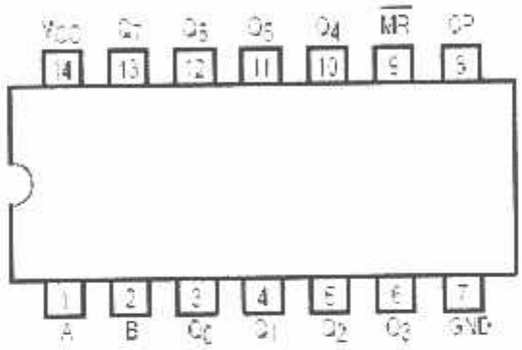
Rangkaian yang mendriver matriks LED ini cukup rumit, karena harus melakukan scan baris-baris horisontal dan pada saat yang sama memasang tegangan pada kolom yang sesuai. Secara umum pencacah lingkar (*Ring Counter*) melakukan scan dalam baris horisontal, sedangkan memori berisi data kolom LED yang menyala menberikan tegangan dalam kolom – kolom vertikal.

2.5. IC TTL (Transistor – Transistor Logic)

Serkuit terpadu atau IC *(Intergrated Circuit)* TTL adalah komponen dasar yang terdiri dari resistor, transistor dan lain-lain. IC adalah komponen yang dipakai sebagai otak peralatan elektronika.

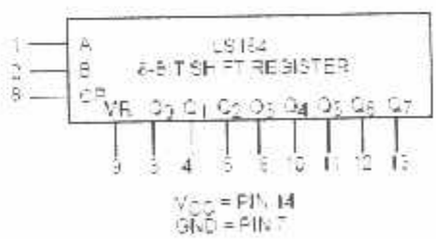
2.5.1. IC 74LS164

Pada perancangan sistem ini untuk *driver Dot* matrik menggunakan IC 74LS164 yaitu sebagai pengerak kolom yang mendapat masukan dari mikrokontroler. 74LS164 yaitu suatu IC yang memiliki kecepatan tinggi 8 *bit Serial Input Parallel Output* (*SIPO*) *Shift Register*. Data serial yang masukan melalui dua masukan gerbang AND disinkronisasi dengan tinggi rendahnya suatu peralihan waktu.



Gambar 2.7. Konfigurasi Pin SN74LS164

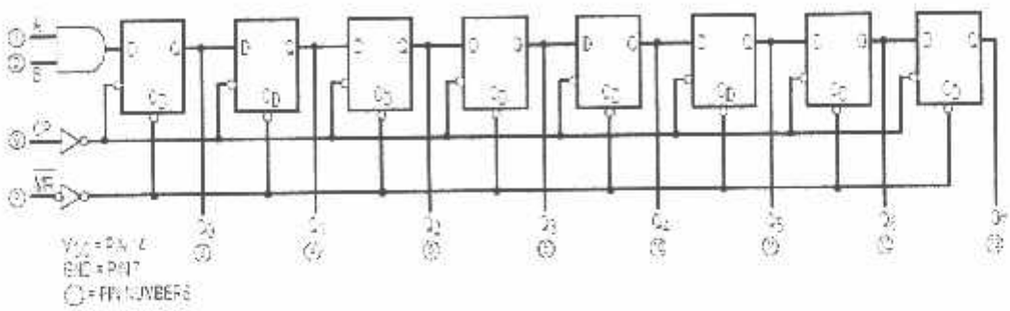
IC 74LS164 adalah IC register geser 8 *bit* dengan 2 masukan serial dan 8 keluaran paralel. Keluaran IC 74LS164 dapat di*reset* dengan memberikan logika *LOW* pada pin 9 (*Clear*). Jika hanya ingin memberikan satu masukan serial, maka masukan serial yang lain harus diberi logika *HIGH*.



Gambar 2.8. Logic Symbol

2.5.2. Konfigurasi Pin 74LS164

- A dan B = Pin A dan B adalah sebagai masukan data.
- CP = Clock (*Active High Going Edge*) Input
- $\overline{MR}$ = Master Reset (*Active Low*) Input
- Q0-Q7 adalah data Keluaran.



Gambar 2.9. Diagram Logic 74LS164

Pada digram logic diatas dapat ketahui bahwa masukan pada IC 74LS164 ini adalah pin nomor 1 dan 2 yaitu A dan B sebagai data masukan yang berupa kondisi *High* atau *Low* (1 atau 0). Sedangkan keluaran pada IC 74LS164 adalah Q0-Q7 sebagai data keluran. Untuk pin nomor 8 sebagai clock input (CP) dan MR sebagai *master reset* yaitu mengubah kondisi semua kembali ke kondisi semula atau normal.

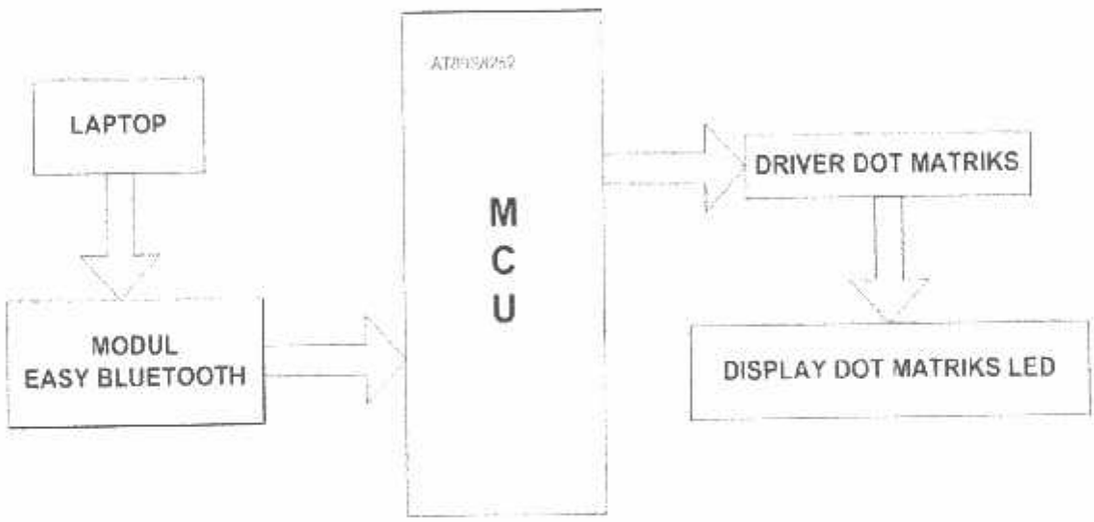
BAB III

PERANCANGAN DAN PEMBUATAN SISTEM

Pada bab ini akan di bahas tentang perancangan dan pembuatan sistem yang meliputi perancangan *software* (perangkat lunak) dan *hardware* (perangkat keras) untuk lebih jelas maka akan dibahas pada sub-sub bab berikut:

3.1. Diagram Blok Rrangkaian

Secara garis besar, prinsip kerja dari alat ini digambarkan diagram bloknya sebagai berikut:



Gambar 3.1 Diagram Blok Sistem

Prinsip kerja dari masing-masing blok:

- Mikrokontroler AT89S8252

  Mikorkontroler master ini adalah sebagai otak pengolah data dari masukan *laptop* yang akan ditampilkan pada matriks led melalui bluetooth.

- *EASY BLUETOOTH* Modul

  Easy Bluetooth adalah Suatu modul *bluetooth* buatan Parallax yang dapat di hubungkan langsung dengan mikrokontroller secara serial sehingga sinyal kontrol yang diterima oleh *EASY BLUETOOTH* dapat langsung diolah oleh mikrokontroller.

- *Laptop*

  Laptop bekerja sebagai data masukan yang akan dikirim ke mikrokontroller dan Display dot matriks, data tertsebut berupa karakter-karakter dan huruf abjad.

- Memori internal EEPROM

  Electrically Erasable Programmable Read-Only Memory (EEPROM) sebagai penyimpan data sementara yang akan dikirim ke display *Dot* matriks.

- Driver Dot matriks

  Driver *Dot* matriks yaitu rangkaian yang berfungsi untuk mengerakkan display *Dot* matriks yaitu kolom dan baris dan memberikan tegangan pada display *Dot* matriks.

- Display *Dot* matriks LED

  Hasil tampilan akhir dari masukan-masukan yang diberikan yaitu berupa karakter atau huruf abjad.

3.2. Perencanaan Hardware

Dalam perencanaan ini, rancangan *hardware* yang dibuat bertujuan guna mendukung dan memberikan kemudahan pada proses kerja perancangan *software* agar nantinya sesuai dengan kondisi yang diinginkan. Untuk perancangan *hardware* sendiri dibagi menjadi 3 bagian yaitu :

- Minimum Sistem AT89S8252
- Driver *Dot* matriks
- Display Dot matriks

3.2.1. Perancangan Minimum Sistem AT89S8252

Penggunaan mikrokontroler AT89S8252 harus didukung oleh beberapa rangkaian penunjang agar dapat melakukan fungsinya, antara lain rangkaian *clock* dan rangkaian *reset*. Selain itu juga harus ditentukan penggunaan port-portnya untuk rangkaian pendukung yang lain.

3.2.1.1. Rangkaian *Clock*

Kecepatan proses pengolahan data pada mikrokontroler ditentukan oleh *clock* (pewaktu) yang dikendalikan oleh mikrokontroller tersebut. Pada mikrokontroller AT89S8252 terdapat *internal clock generator* yang berfungsi sebagai sumber *clock*, tapi masih memerlukan rangkaian tambahan untuk membangkitkan *clock* yang diinginkan.

Rangkaian tambahan ini terdiri atas 2 buah kapasitor dan sebuah kristal yang terangkai sedemikian rupa dan kemudian dihubungkan dengan port yang khusus tersedia pada mikrokontroler.

Dalam perancangan rangkaian ini menggunakan :

- 2 Kapasitor 30 pF. Penentuan besarnya kapasitansi disesuaikan dengan spesifikasi pada data sheet.
- Kristal 12 MHz

Dengan demikian perhitungannya dapat dilihat sebagai berikut :

f = 12 MHz

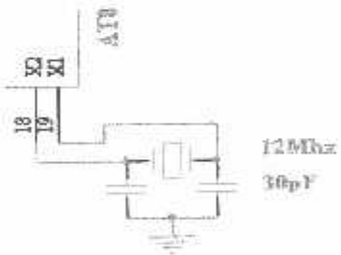
$$T = \frac{1}{f}$$

$$T = \frac{1}{12x10^6}$$

karena 1 siklus mesin = 12T maka

$$1 \text{ siklus mesin} = 12 \times \frac{1}{12x10^6} = 0{,}999\ \mu s$$

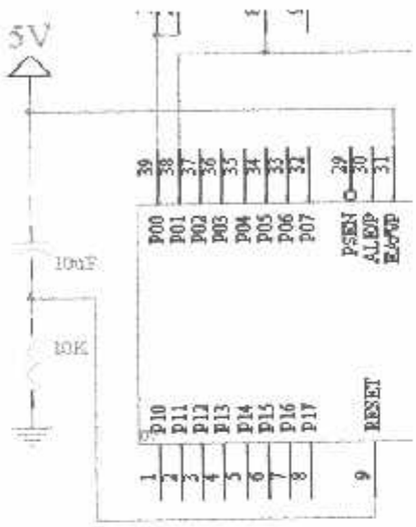
Adapun gambar rangkaian *clock* sebagai berikut :



Gambar 3.2 Rangkaian *Clock*

3.2.1.2. Rangkaian *Reset*

Reset pada mikrokontroler merupakan masukan aktif *high* '1' pulsa transisi dari rendah '0' ketinggi akan me-*reset* mikrokontroler menuju alamat 0000H. Pin *reset* dihubungkan dengan rangkaian *power-on reset* seperti pada gambar 3-3:



Gambar 3.3 Rangkaian *Reset*

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

Rangkaian *reset* terbentuk oleh komponen resistor dan kapasitor yang sudah baku (ditetapkan oleh perusahaan pembuat IC AT89S8252). Nilai resistor yang dipakai adalah 10KΩ dan kapasitor 10μF. Karena kristal yang digunakan mempunyai frekuensi sebesar 12 MHz, maka satu periode membutuhkan waktu sebesar :

$$T = \frac{1}{f_{XTAL}} = \frac{1}{12MHz}\ S - 8,333\ x\ 10^{-8}$$

Sehingga waktu minimal logika yang dibutuhkan untuk me-*reset* mikrokontroller adalah :

Reset (minimal) = T x periode yang dibutuhkan

= 8,333 x 10^{-8} x 12 = 1,999 µs

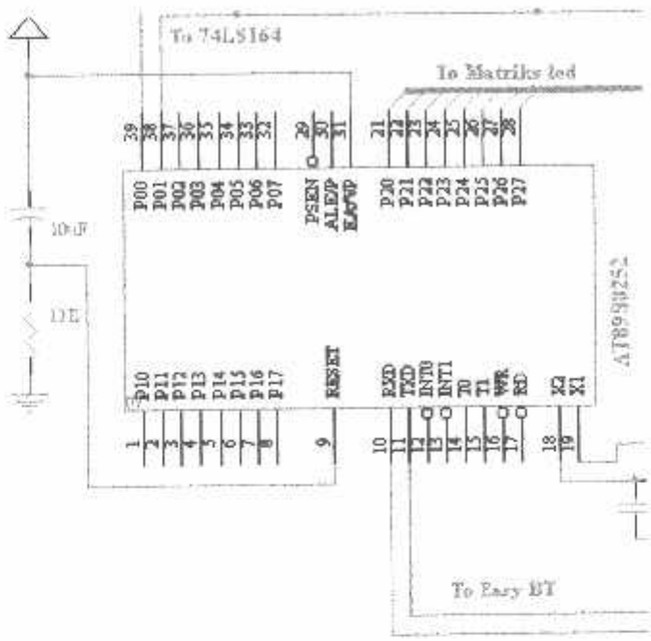
Jadi mikrokontroller membutuhkan waktu minimal 1,999 µs untuk me-*reset*. Waktu inilah yang dijadikan pedoman untuk menetukan nilai R dan C. Dengan menetukan nilai R = 10 kΩ dan C = 10 µF, maka :

T = 0.357 R.C =0,357 x 10000 Ω x 10.10^{-6} = 35,7 ms

Jadi dengan nilai komponen R = 10 kΩ dan C = 10 µF dapat memenuhi syarat minimal untuk waktu yang dibutukan mikrokontroler

3.2.1.3. Perancangan Penggunaan Port-Port Pada Mikrokontroller AT89S8252

Pada skripsi ini mikrokontroller AT89S8252 digunakan sebagai pusat pengendali kerja dari alat yang dibuat Gambar 3-4 menunjukkan rancangan port-port I/O pada mikrokontroller AT89S8252 yang dimanfaatkan pada skripsi.



Gambar 3.4 Perancangan mikrokontroler

1. Port 0

   Port 0.0 – Port 0.1 (pin 39 – 38) digunakan sebagai *outputan* data yang dikirim ke IC 74LS164

2. Port 2

   Pin 21 – 28 dihungkan ke matriks led

3. Port 3

   Pin 10 yaitu RXD dihubungkan ke pin 11 TXD *easy bluetooth*

   Pin 11 yaitu TXD dihubungkan ke pin 10 RXD easy bluetooth

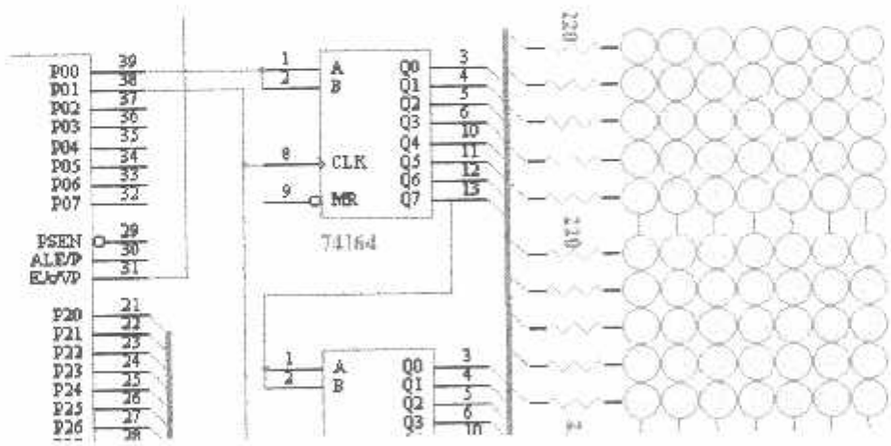
4. Pin 9 (*reset)* dihubungkan dengan rangkaian *reset*

5. Pin 18 dan pin 19 dihubungkan dengan rangkaian rangkaian *clock* atau *Oscillator external*

6. Pin 31 (EA) diberi logika tinggi (*high)* atau dihubungkan dengan Vcc maka mikrokontroller akan mengakses program dari ROM internal (EPROM atau *flash memory*)
7. Pin 29 dan pin 30 (ALE/PROG dan PSEN) tidak digunakan karena pada pembuatan alat ini tidak menggunakan atau mengakses *memory eksternal*
8. Pin 40 (Vcc) dihubungkan dengan tegangan *supply* −5V
9. Pin 20 (GND) dihubungkan dengan tegangan *supply ground*

3.2.2. Driver Dot matriks

Pada perancangan driver *Dot* matrik yaitu digunakan *IC* register 74LS164 sebagai pengeser data kolom pada dot matrik dan transistor TIP31 sebagai pembangkit tegangan atau saklar pada driver *Dot* matriks, berikut adalah gambar perancangan untuk *driver Dot* matrik.



Gamabar 3.6 Perancangan Rangkaian Driver Dot Matrik

Konfigurasi pin – pin dari perancangan driver dot matrik adalah:

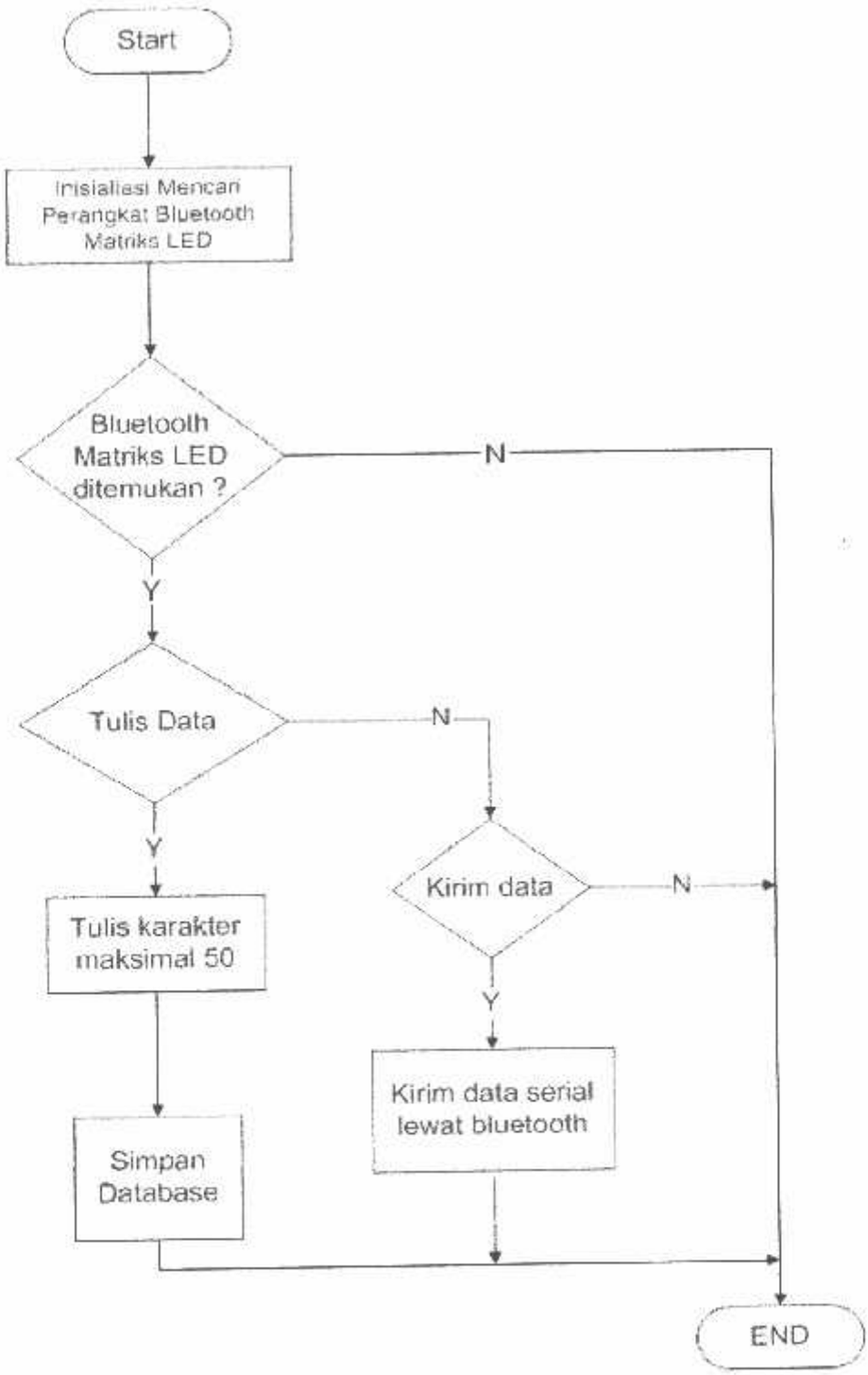
- Pin 1 dan pin 2 adalah inputan data yaitu dihubungkan dengan pin 39 dari mikrokontroller AT89S8252.
- Pin 8 sebagai clock yang dihubungkan dengan pin 38 dari mikrokontroler AT89S8252
- Pin 3, 4, 5, 6, 10, 11, 12 dan 13 sebagai keluaran data yang dihubungkan dengan dot matrik yang pada baris-barisnya. Disini pin 13 juga sebagai inputan ke IC 74LS164 yang lain yaitu dihubungkan ke pin 1 dan 2 IC berikutnya.

3.3. Perancangan Perangkat Lunak (*Software*)

Dalam menunjang kerja sistem secara kesceluruhan diperlukan suatu perangkat lunak (*software*). Perangkat lunak *(software)* yang digunakan untuk AT89S8252 disini menggunakan bahasa assembler keluarga MSC52. Program yang ditulis dengan bahasa *assembly* terdiri dari *label: kode mnemonic* dan lain sebagainya yang pada umumnya dinamakan sebagai program sumber (*source code*) yang belum bisa diterima oleh prosesor untuk dijalankan sebagai program, tetapi harus dijalankan dulu menjadi bahasa mesin dalam bentuk *kode biner*.

- Penulisan program dengan menggunakan teks editor dan disimpan dengan ekstensi *Asm*.
- Meng-compile program yang telah ditulis dengan menggunakan Compiler MCS52 sehingga didapatkan file dengan ekstensi *Hex*.

- Mengubah file berekstensi *Hex* menjafi file berekstensi *Bin*.
- Men-download file berekstensi Bin ke dalam EPROM Mikrokontroler AT89S8252


Start
Inisialiasi Mencari Perangkat Bluetooth Matriks LED
Bluetooth Matriks LED ditemukan ?
N
Y
Tulis Data
N
Y
Tulis karakter maksimal 50
Simpan Database
Kirim data
N
Y
Kirim data serial lewat bluetooth
END


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB IV

PENGUJIAN SISTEM

Pada bab ini akan membahas tentang pengujian alat yang telah dirancang, dirakit serta direalisasikan. Tujuan pengujian alat ini adalah mengetahui kerja dari masing-masing sistem yang dibuat secara per-blok. Dengan demikian dapat diketahui kepersisan kerja dari alat yang direncanakan dan dibuat, pada bab ini akan dibahas tentang pengujian sistem yang telah dirancang, yaitu sebagai :

1. Pengujian terhadap rangkaian mikrokontroler
2. Pengujian pengiriman data dari laptop ke matriks led
3. Pengujian Keseluhan Sistem

Dari pengujian sistim ini tujuannya adalah sebagai berikut:

1. Mengetahui proses kerja dari masing-masing rangkaian (blok).
2. Memudahkan pendataan spesifikasi alat.
3. Mengetahui hasil dari suatu perancangan yang telah dibuat.
4. Memudahkan perawatan dan perbaikan apabila sewaktu-waktu terjadi kerusakan pada sistim tersebut.

4.1. Pengujian Rangkaian Mikrokontroler

4.1.1. Tujuan

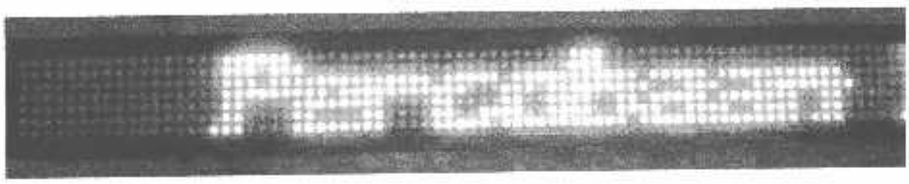
Dalam pengujian rangkaian mikrokontroler bertujuan untuk mengetahui sistem kerja dari display Dot matrik dan IC register yang bekerja sesuai dengan apa yang telah direncanakan dalam perancangan tersebut.

4.1.2. Langkah – langkah Pengujian

Langkah – langkah dalam pengujian ini dilakukan dengan menghubungkan mikrokontroler dengan LED dot matrik. Mikrokontroler ini diprogram dengan kalimat "Percobaan". Hasil dari pengujian ini dapat dilihat pada Gambar 4.1.

4.1.3. Hasil dan Analisa

Pada hasil pengujian ini hanya terlihat beberapa kalimat saja dari tampilan matriks led, ini disebabkan karena sistem kerja dari dot matrik tersebut adalah menampilkan kalimat dengan cara bergeser atau berjalan.



Gambar 4.1 Hasil Pengujian Pada Mikrokontroler

Dari hasil pengujian pada mikrokontroler , diketahui bahwa sistem ini bekerja sesuai dengan masukan yang diberikan-Nya berupa tulisan "Percobaan" yang terlihat pada gambar 4.1 diatas, sehingga dapat disimpulkan bahwa mikrokontroler tersebut telah berhasil diuji.

4.2. Pengujian Pengiriman Data Dari laptop ke matriks led melalui bluetooth

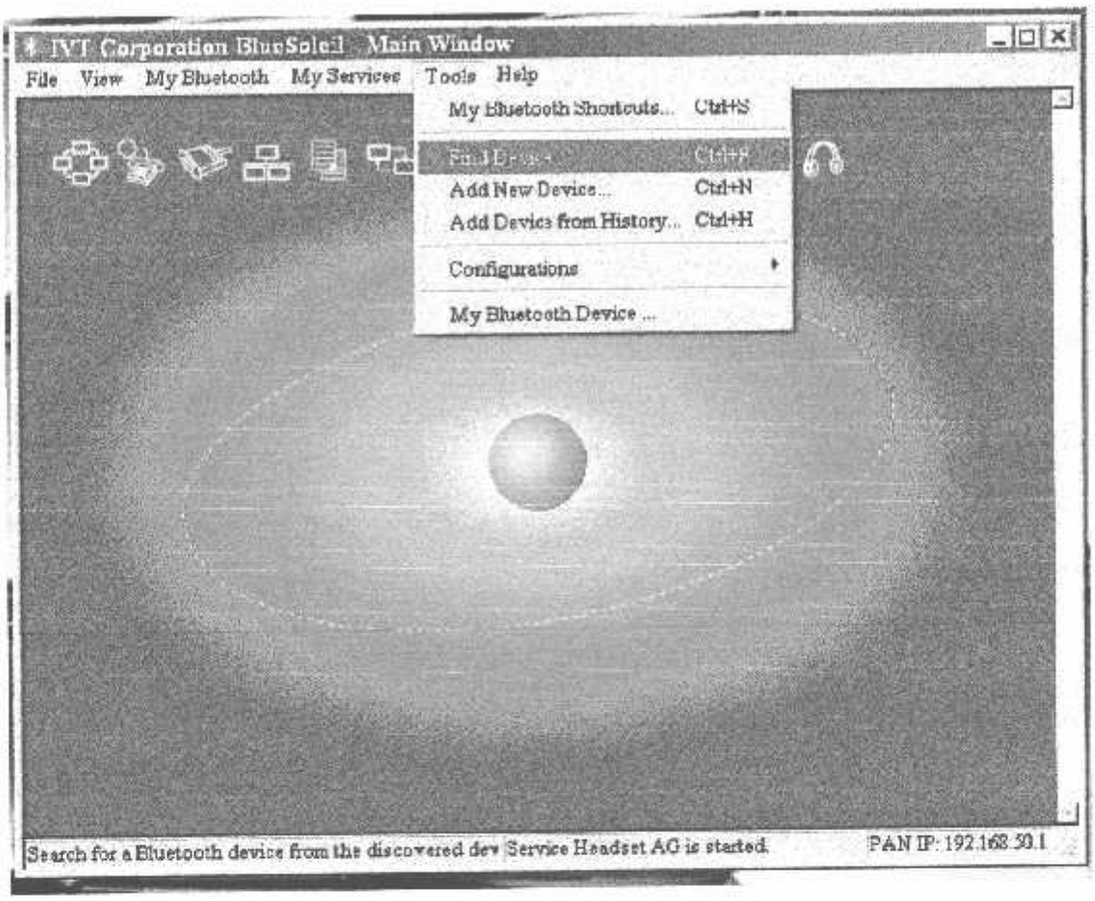
4.2.1. Tujuan

Pada pengujian pengiriman data dari laptop ke matriks led, bertujuan untuk mengetahui pengiriman data dari laptop ke matriks led melalui bluetooth.

4.2.2. Langkah – langkah

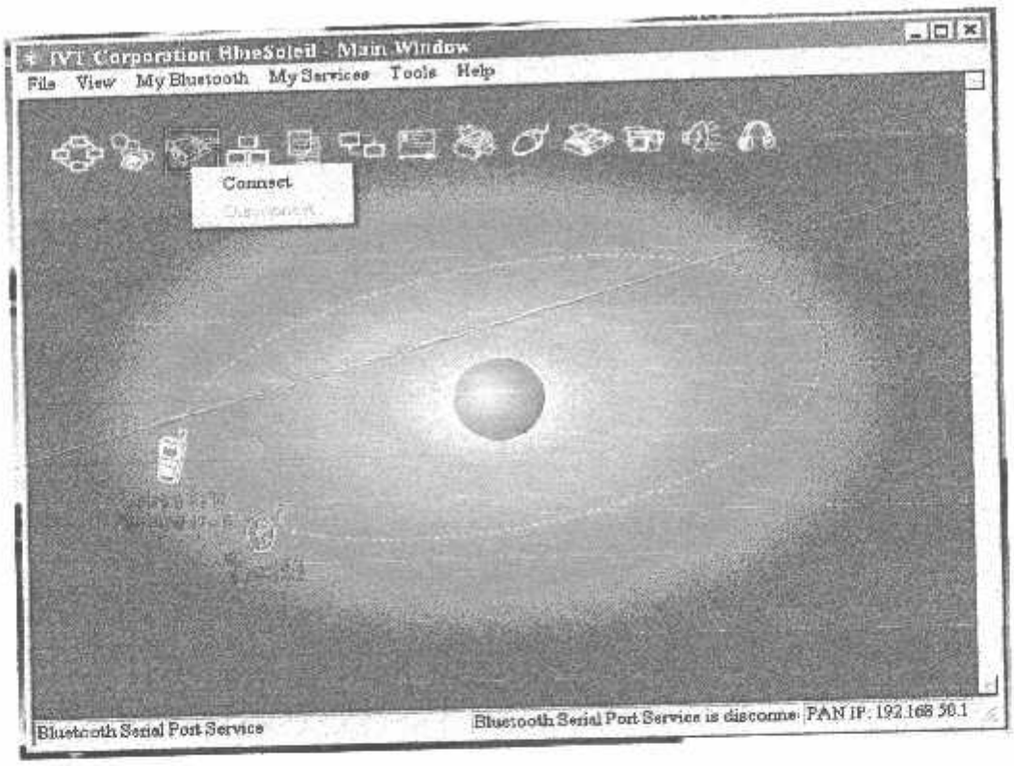
Untuk melakukan pengujian ini, maka diperlukan hubungan antara bluetooth pada laptop dan bluetooth pada mikorokontroler maka dapat dilakukan dengan cara

mencari perangkat bluetooth yang terdapat pada matriks led.

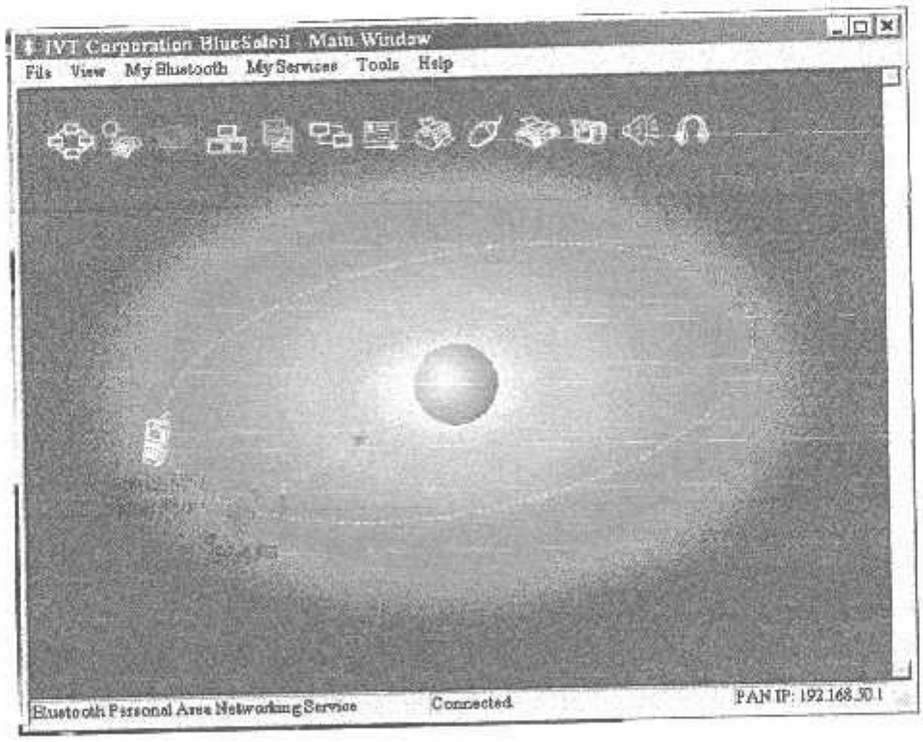


Gambar 4-2 .mencari koneksi bluetooth

Setelah itu muncul bluetooth device hal ini memberitahukan user bahwa easy bluetooth sedang dikoneksikan ke Laptop. Setelah easy bluetooth terhubung maka akan muncul gambar bahwa easy bluetooth telah terhubung ke laptop.

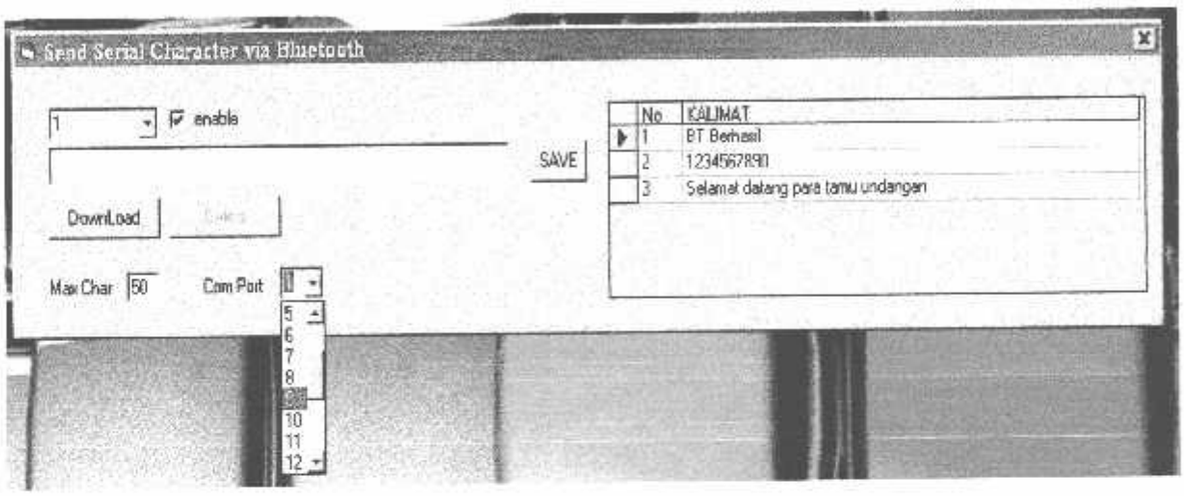


Gambar 4-3, Option connect

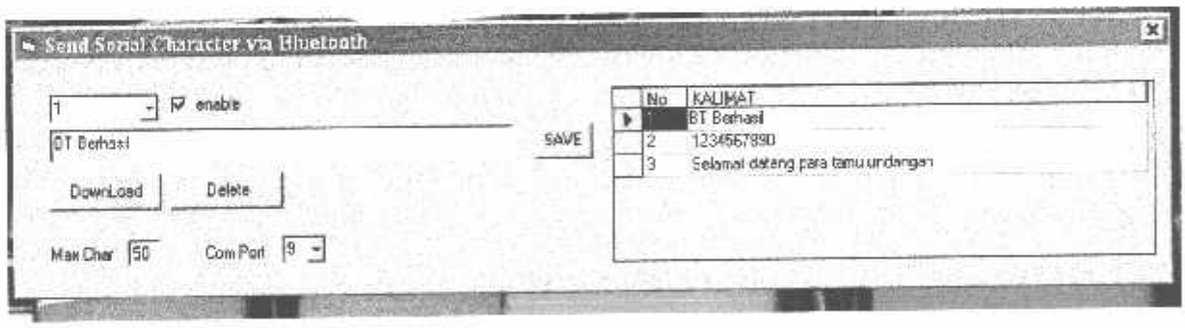


Gambar 4-4, easy bluetooth Connected

Setelah itu kita akan melakukan penyetingan pada program, dengan menggunakan Com9 (com virtual yang digunakan laptop untuk berkomunikasi dengan easy bluetooth). Setelah selesai setting maka kita dapat mengirimkan data karakter dengan menuliskan kalimat "BT Berhasil" pada program yang terdapat pada laptop dan data tersebut dikirimkan ke mikrokontroler melalui bluetooth yang selanjutnya akan ditampilkan ke *dot* matriks LED.



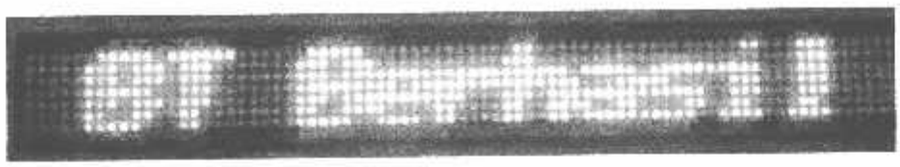
Gambar 4-5, Mengganti Com pada program



Gambar 4-6, Tampilan program saat akan mengirimkan data ke matriks led melalui bluetooth

4.2.3. Hasil dan Analisa

Dari pengujian ini dapat dilihat pada gambar 4-7 berikut:



Gambar 4-7, Tampilan Hasil Pengiriman Data dari laptop ke matriks led melalui bluetooth

Pada pengujian pengiriman data dari laptop ke mikrokontroler ini diketahui bahwa, data tulisan yang dikirimkan dari laptop ke mikrokonroller telah berhasil diuji karena hasil akhir yang ditampilkan pada dot matriks LED sama seperti data tulisan yang ada pada laptop .

Pada jarak tertentu laptop tidak dapat menerima atau mendeteksi adanya bluetooth, dapat di lihat dari tabel percobaan koneksi bluetooth.

Tabel 4-1 Percobaan koneksi bluetooth

JARAK	KONEKSI
10 meter	Terhubung dengan baik
20 meter	Terhubung dengan baik
30 meter	Dalam 10 kali tranfes 3 kali gagal
40 meter	Tidak dapat terdeteksi oleh laptop

4.3. Pengujian Rangkaian Keseluruhan Sistem

4.3.1. Tujuan

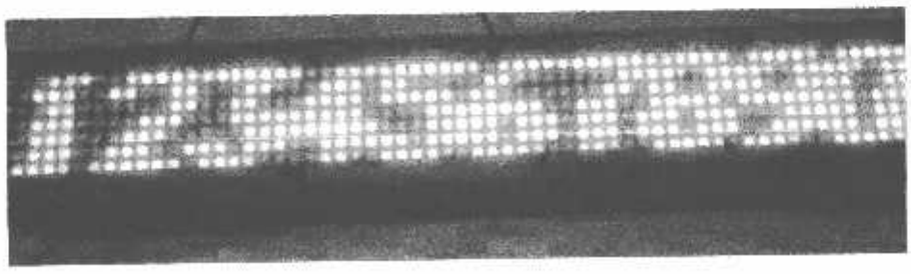
Dalam pengujian sistem keseluruhan ini, bertujuan agar mengetahui cara kerja dari sistem yang telah dirancang bekerja dengan baik, sesuai dengan apa yang telah direncanakan.

4.3.2. Langkah – langkah

Untuk melakukan pengujian pada sistem keseluruhan ini, maka menghubungan antara laptop, mikrokontroler terhubung dengan baik melalui bluetooth. Pada laptop diketikkan karakter "1234567890" yang akan dikirim ke *dot* matriks LED dan hasil tampilan pada *Dot* matriks hanya muncul "1234567890" karena hasil tampilan pada *Dot* matriks tersebut bersifat pergesaran karakter. Pergeseran ini dilakukan oleh IC 74LS164 yang sebagai IC register untuk mengeser baris pada *dot* matriks LED.

4.3.3. Hasil dan Analisa

Dari hasil pengujian sistem keseluruhan ini dapat dilihat pada gambar 4.5 berikut :



Gambar 4.5 Hasil Pengujian Sistem Keseluruhan

Dari hasil pengujian sistem keseluruhan ini, diketahui bahwa tulisan yang diketikkan pada *laptop* dan dikirim ke *dot* matriks LED telah berhasil diuji karena hasil tampilan pada *dot* matriks LED sama seperti tulisan yang diketikkan dari *laptop*.

BAB V

PENUTUP

Bab ini akan dibahas tentang kesimpulan yang diambil dari keseluruhan pada sub bah – bab sebelumnya, yaitu akan dibahas sebagai berikut:

5.1 KESIMPULAN

Dari hasil perancangan dan pengujian sistem ini maka dapat diambil kesimpulan-nya seabagai berikut:

1. Komunikasi bluetooth dapat digunakan sebagai pengganti serial port dan pararell port yang tidak terdapat pada laptop, dalam menampilkan karakter pada matriks lcd .
2. Proses penulisan informasi yang akan di tampilkan pada matriks led dapat berlangsung dengan baik.
3. Komunikasi antara laptop dan mikrokontroller dapat dilakukan dengan baik dan mudah dengan bantuan modul easy bluetooth
4. Pada jarak 10 meter hingga 20 meter koneksi bluetooth terkoneksi dengan lancar, sedangkan pada jarak 30 meter dalam sepuluh kali percobaan mengalami 3 kali kegagalan koneksi, dan pada jarak 40 meter ke atas bluetooth tidak dapat terdeteksi oleh laptop atau tidak dapat terjadi koneksi dengan laptop.
5. Pada kerja sistem ini, apabila hubungan dengan bluetooth terputus, maka tulisan yang terakhir kali tampil pada *dot* matriks LED masih tetap tersimpan dan tetap di tampilkan, sehingga pada saat bluetooth terhubung kembali maka tulisan tersebut masih ditampilkan dan dapat di rubah.

5.2 SARAN

Dalam perancangan sistem ini, untuk lebih lanjut-Nya dalam penggembangan sistem ini, maka penulis menyarankan bahwa :

1. Sistem pengiriman data atau komunikasi antara laptop dengan mikrokontroler menggunakan komunikasi bluetooth jarak pengirimanya bisa semakin jauh.
2. Data yang dikirimkan oleh laptop melalui bluetooth untuk di tampilkan pada matriks led bukan hanya berupa karakter saja tetapi juga dapat mengirimkan gambar.
3. Pada pengiriman data ke mikrokontroller yang akan di tampilkan pada matriks led, menggunakan komunikasi bluetooth. Tidak hanya dapat menerima data dari bluetooth pada laptop tetapi juga dapat menerima data yang dikirimkan oleh bluetooth yang terdapat pada handphone.

DAFTAR PUSTAKA

[1] www.Atmel.com

[2] www.Beyondlogic.org

[3] Data sheet 74LS164 www.alldatasheet.com

[4] www.dtechtronic.com

[5] Simulasi Elektronika Digital menggunakan Electronic workbench, **Purnomo Gatot Santosa,** ANDY Yogyakarta 2007.

[6] www.digi-ware.com

[7] www.parallax.com

Lampiran

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
Jl. Karanglo km 2, Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama : Vicky Agus Lesmana

NIM : 03.17.070

Jurusan : Teknik Elektro S-1

Konsentrasi : Teknik Elektronika S-1

Judul Skripsi : **PERENCANGAN DAN PEMBUATAN DISPLAY MATRIKS LED BERBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH.**

Dipertahankan di hadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari : Selasa

Tanggal : 06 Oktober 2009

Dengan Nilai : 78,25 (B+)

Ketua Majelis Penguji

Ir. H. Sidik Noertjahjono, MT
NIP.Y 102 8700 163

Sekretaris Majelis Penguji

Ir. F. Yudi Limpraptono, MT
NIP.Y 103 9500 274

Penguji I

Ir. F. Yudi Limpraptono, MT
NIP.Y 103 9500 274

Penguji II

I Komang Somawirata, ST, MT
NIP. 103 0100 361

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO S-1
Jl. Karanglo km 2, Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata Satu (S-1) Jurusan Teknik Elektro Konsentrasi Teknik Komputer dan Informatika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

Nama	: Vicky Agus Lesmana
NIM	: 03.17.070
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Elektronika S-1
Masa Bimbingan	: 23 Juni 2009 s/d 23 Desember 2009
Judul Skripsi	: **PERENCANGAN DAN PEMBUATAN DISPLAY MATRIKS LED BERBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH.**

Tanggal	Uraian	Paraf
Penguji I 06 Okt 2009	Kesimpulan di tambahkan	
Penguji II 06 Okt 2009	Perbaiki Flowchart	

Disetujui :

Penguji I

Ir. F. Yudi Limpraptono, MT
NIP.Y 103 9500 274

Penguji II

I Komang Somawirata, ST, MT
NIP. 103 0100 361

Mengetahui :

Dosen Pembimbing

Joseph Dedy Irawan
NIP. 132.315.178

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

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

NAMA : Vicky Agus K.
NIM :
Perbaikan meliputi :

1) Kompilasi file ditambahkan dan hasil pengujian.

Malang, 200

( )

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

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

NAMA : VICKY AGUS L.
NIM : 03.17.070
Perbaikan meliputi :

* Flow Chart. (tuh tu ?)

Malang, 06- 10- 2009

(_____)

Institu Teknologi Nasional Malang
Fakultas Teknologi Industri
Jurusan Teknik Elektro S-1

FORMULIR BIMBINGAN SKRIPSI

Nama : VICKY AGUS LESMANA
Nim : 03.17.070
Masa Bimbingan : 23 Juni 2009 s/d 23 Desember 2009
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN DISPLAY MATIRKS LED BERBASIS MIKROKONTROLLER AT89S8252 DENGAN MENGGUNAKAN KOMUNIKASI BLUETOOTH**

NO	Tanggal	Uraian	Paraf Pembimbing
1	28/07/2009	Acc bab I	
2	28/07/2009	Rev bab II Rev bab III	
3	4/08/2009	Acc bab II Rev bab III Rev bab IV	
4	18/08/09	Acc bab III Acc bab IV	
5	25/08/09	Acc bab V	
6	8/09/2009	Acc Seminar hasil	
7	28-9-2009	Acc Lompre	
8			
9			
10			

Malang,
Dosen Pembimbing

Joseph Dedy Irawan, ST. MT.
NIP.132 315 178

Form S-4b

Features

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

Description

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

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

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

8-bit Microcontroller with 8K Bytes Flash

AT89S8252

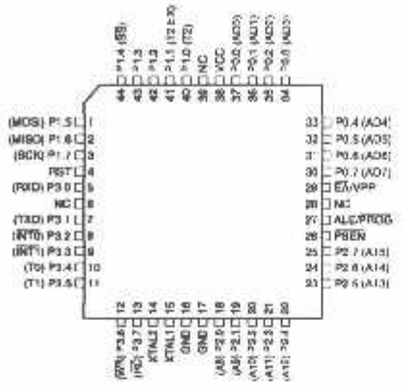
Not Recommended for New Designs. Use AT89S8253.

0401G–MICRO–3/06

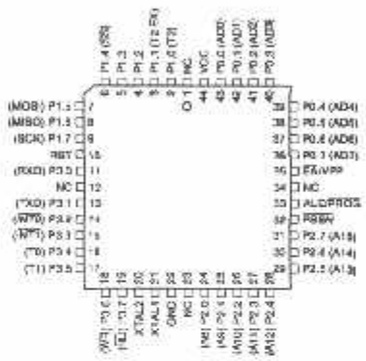
Pin Configurations

PDIP

TQFP



PLCC



Pin Description

VCC Supply voltage.

GND Ground.

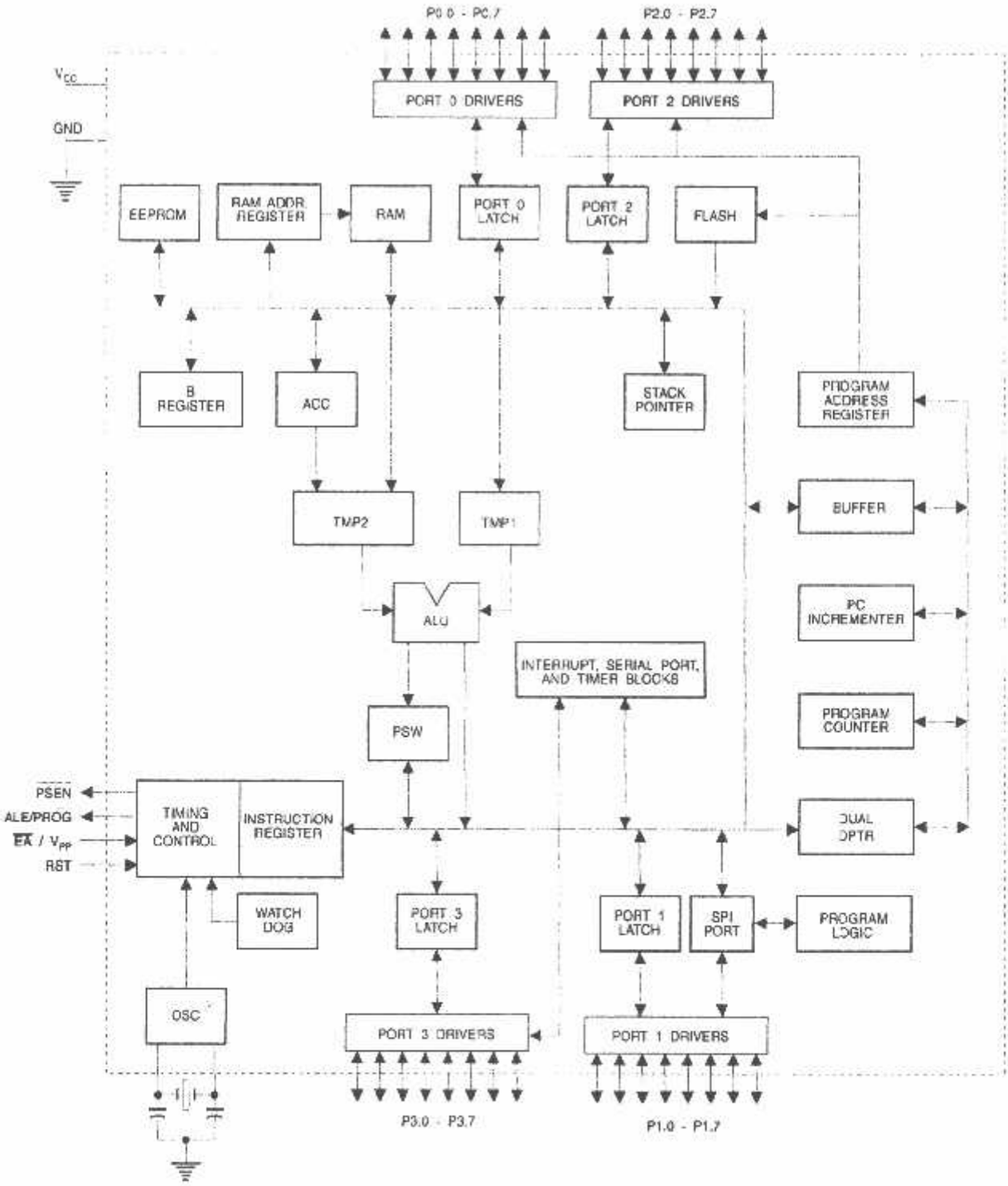
Port 0 Port 0 is an 8-bit open drain bi-didirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

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.

Block Diagram



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

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

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

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

Port 2

Port 2 is an 8-bit bi-directional I/O port with internal 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 AT89S8252, as shown in the following table.

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

RST

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

ALE/$\overline{PROG}$

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

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

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

$\overline{PSEN}$

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

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

$\overline{EA}$/VPP

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

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

XTAL1

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

XTAL2

Output from the inverting oscillator amplifier.

Special Function Registers

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

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

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

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

Table 1. AT89S8252 SFR Map and Reset Values

0F8H									0FFH
0F0H	B 00000000								0F7H
0E8H									0EFH
0E0H	ACC 00000000								0E7H
0D8H									0DFH
0D0H	PSW 00000000					SPCR 000001XX			0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			0CFH
0C0H									0C7H
0B8H	IP XX000000								0BFH
0B0H	P3 11111111								0B7H
0A8H	IE 0X000000		SPSR 00XXXXXX						0AFH
0A0H	P2 11111111								0A7H
98H	SCON 00000000	SBUF XXXXXXXX							9FH
90H	P1 11111111						WMCON 00000010		97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR XXXXXXXX	PCON 0XXX0000	87H

Table 2. T2CON – Timer/Counter 2 Control Register

T2CON Address = 0C8H — Reset Value = 0000 0000B

Bit Addressable

Bit	7	6	5	4	3	2	1	0
	TF2	EXF2	RCLK	TCLK	EXEN2	TR2	C/$\overline{T2}$	CP/$\overline{RL2}$

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

Watchdog and Memory Control Register The WMCON register contains control bits for the Watchdog Timer (shown in Table 3). The EEMEN and EEMWE bits are used to select the 2K bytes on-chip EEPROM, and to enable byte-write. The DPS bit selects one of two DPTR registers available.

Table 3. WMCON—Watchdog and Memory Control Register

WMCON Address = 96H — Reset Value = 0000 0010B

Bit	PS2	PS1	PS0	EEMWE	EEMEN	DPS	WDTRST	WDTEN
	7	6	5	4	3	2	1	0

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

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

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

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

Power Off Flag The Power Off Flag (POF) is located at bit_4 (PCON.4) in the PCON SFR. POF is set to "1" during power up. It can be set and reset under software control and is not affected by RESET.

Table 4. SPCR – SPI Control Register

SPCR Address = D5H — Reset Value = 0000 01XXB

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

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

Table 5. SPSR – SPI Status Register

SPSR Address = AAH — Reset Value = 00XX XXXXB

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

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

Table 6. SPDR – SPI Data Register

SPDR Address = 86H — Reset Value = unchanged

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

Data Memory – EEPROM and RAM

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

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

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

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

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

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

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

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

The EEMWE bit in the WMCON register needs to be set to "1" before any byte location in the EEPROM can be written. User software should reset EEMWE bit to "0" if no further EEPROM write is required. EEPROM write cycles in the serial programming mode are self-timed and typically take 2.5 ms. The progress of EEPROM write can be monitored by reading the RDY/$\overline{BSY}$ bit (read-only) in SFR WMCON. RDY/$\overline{BSY}$ = 0 means

programming is still in progress and RDY/$\overline{BSY}$ = 1 means EEPROM write cycle is completed and another write cycle can be initiated.

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

Programmable Watchdog Timer

The programmable Watchdog Timer (WDT) operates from an independent internal oscillator. The prescaler bits, PS0, PS1 and PS2 in SFR WMCON are used to set the period of the Watchdog Timer from 16 ms to 2048 ms. The available timer periods are shown in the following table and the actual timer periods (at V_{CC} = 5V) are within ±30% of the nominal.

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

Table 7. Watchdog Timer Period Selection

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

Timer 0 and 1

Timer 0 and Timer 1 in the AT89S8252 operate the same way as Timer 0 and Timer 1 in the AT89C51 and AT89C52. For further information on the timers' operation, refer to the Atmel web site (http://www.atmel.com). From the home page, select "Products", then "Microcontrollers, then "8051-Architecture". Click on "Documentation", then on "Other Documents". Open the document "AT89 Series Hardware Description".

Timer 2

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

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

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

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

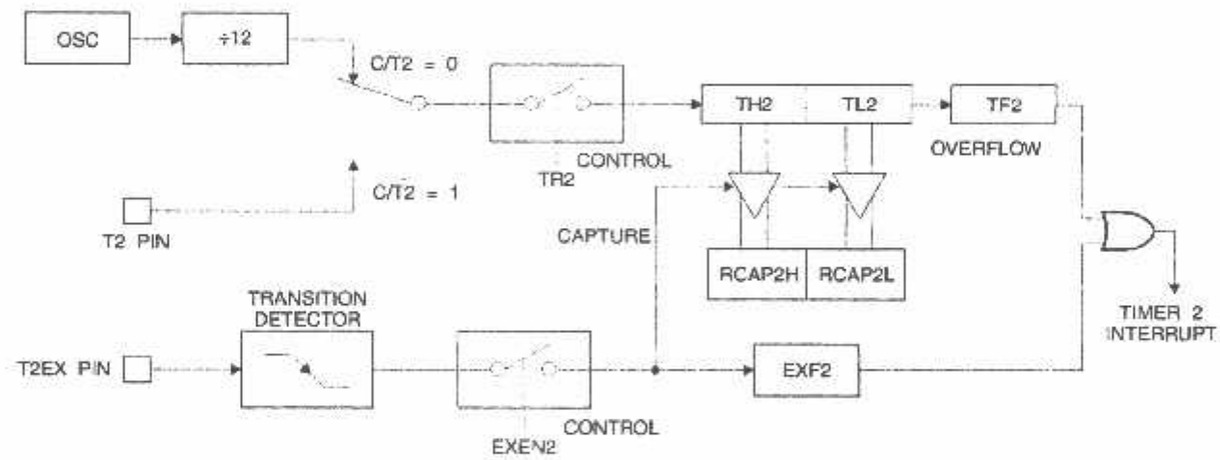
Table 8. Timer 2 Operating Modes

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)

Capture Mode

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

Figure 1. Timer 2 in Capture Mode



Auto-reload (Up or Down Counter)

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

Figure 2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H and RCAP2L. The values in RCAP2H and RCAP2L are preset by software. If EXEN2 = 1, a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at external input T2EX. This transition also sets the EXF2 bit. Both the TF2 and EXF2 bits can generate an interrupt if enabled.

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

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

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

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

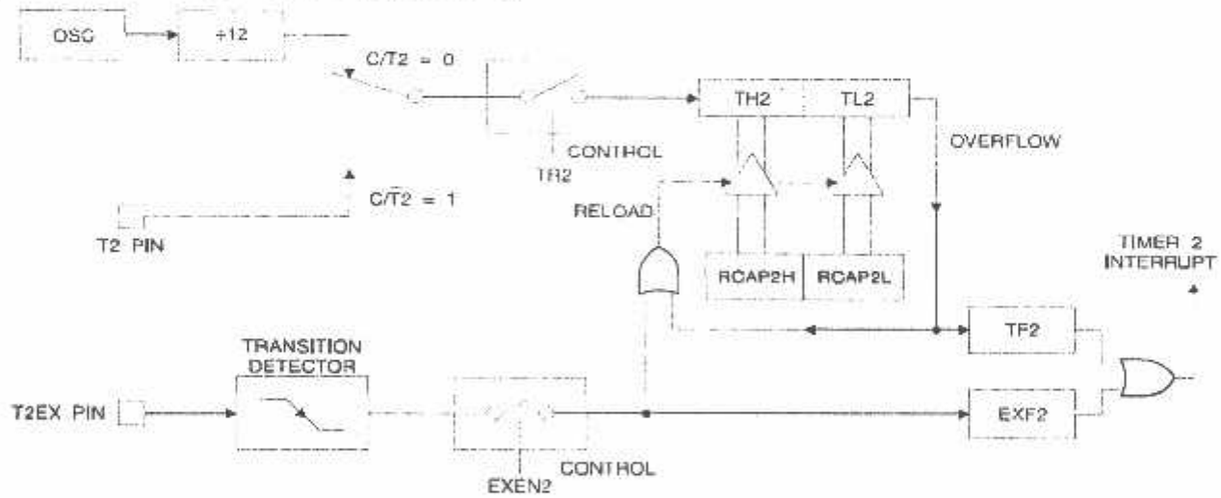


Table 9. T2MOD – Timer 2 Mode Control Register

T2MOD Address = 0C9H — Reset Value = XXXX XX00B

Not Bit Addressable

Bit	7	6	5	4	3	2	1	0
	–	–	–	–	–	–	T2OE	DCEN

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

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

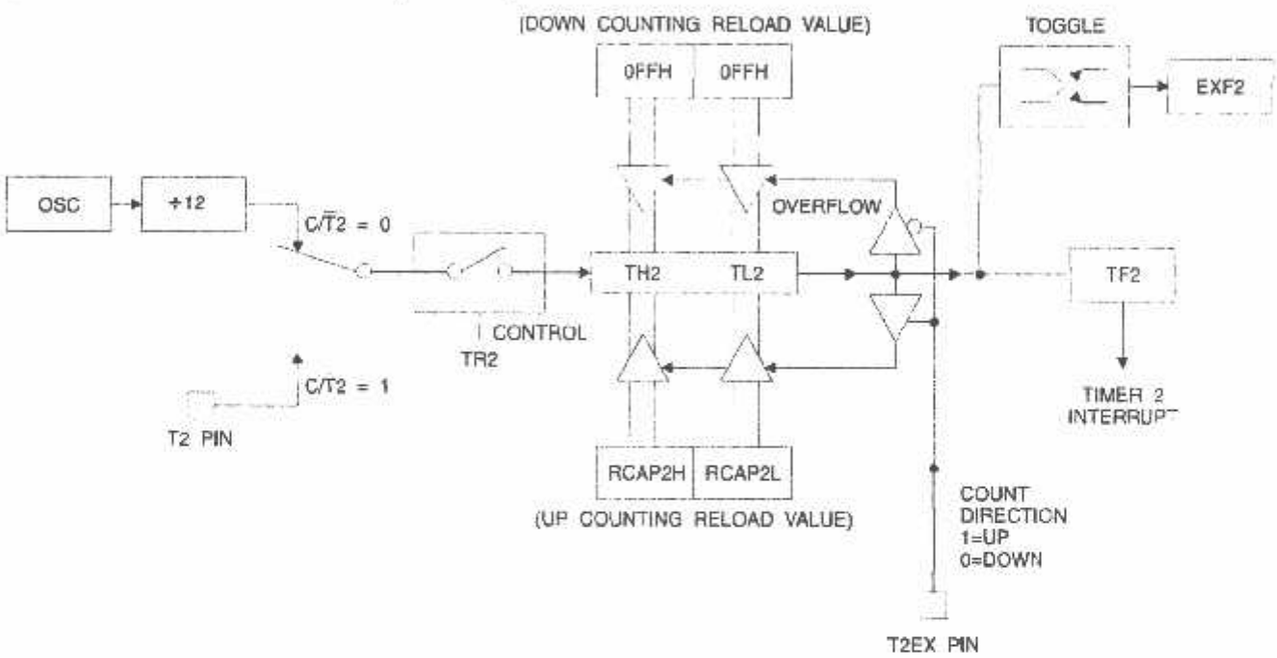
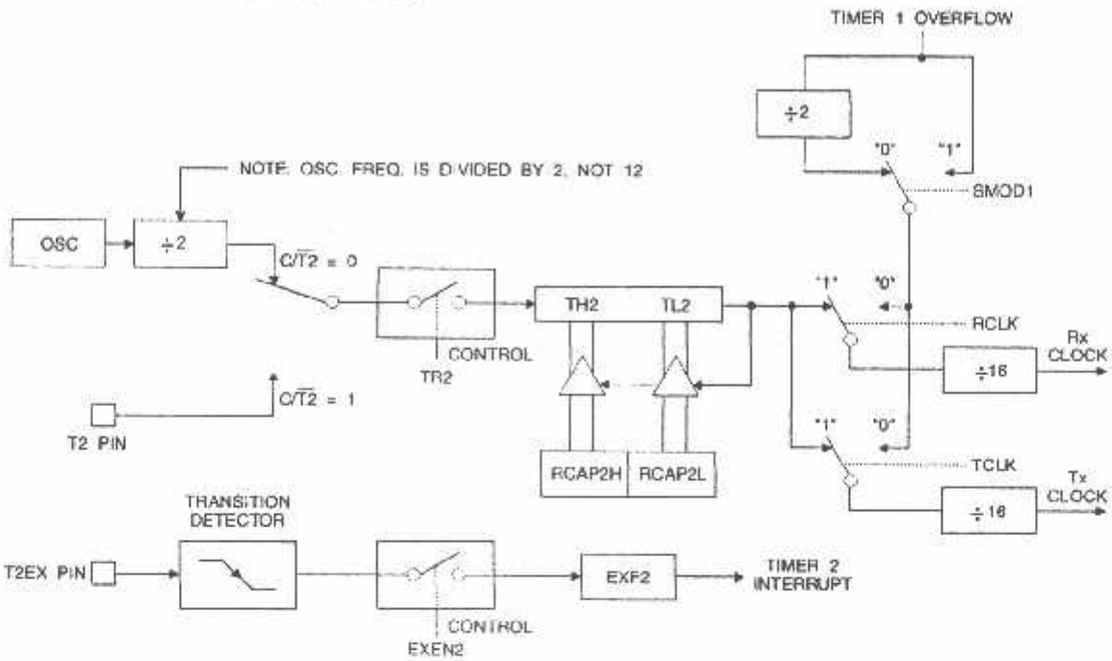


Figure 4. Timer 2 in Baud Rate Generator Mode



Baud Rate Generator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 2). Note that the baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 4.

The baud rate generator mode is similar to the auto-reload mode, in that a rollover in TH2 causes the Timer 2 registers to be reloaded with the 16-bit value in registers RCAP2H and RCAP2L, which are preset by software.

The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate according to the following equation.

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

The Timer can be configured for either timer or counter operation. In most applications, it is configured for timer operation ($CP/\overline{T2} = 0$). The timer operation is different for Timer 2 when it is used as a baud rate generator. Normally, as a timer, it increments every machine cycle (at 1/12 the oscillator frequency). As a baud rate generator, however, it increments every state time (at 1/2 the oscillator frequency). The baud rate formula is given below.

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

where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

Timer 2 as a baud rate generator is shown in Figure 4. This figure is valid only if RCLK or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2 and will not generate an interrupt. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt.

Note that when Timer 2 is running (TR2 = 1) as a timer in the baud rate generator mode, TH2 or TL2 should not be read from or written to. Under these conditions, the Timer is incremented every state time, and the results of a read or write may not be accurate. The RCAP2 registers may be read but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

Programmable Clock Out

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 5. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz (for a 16-MHz operating frequency).

To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

The clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L), as shown in the following equation.

$$\text{Clock Out Frequency} = \frac{\text{Oscillator Frequency}}{4 \times [65536 - (\text{RCAP2H,RCAP2L})]}$$

In the clock-out mode, Timer 2 rollovers will not generate an interrupt. This behavior is similar to when Timer 2 is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and clock-out frequencies cannot be determined independently from one another since they both use RCAP2H and RCAP2L.

Figure 5. Timer 2 in Clock-out Mode

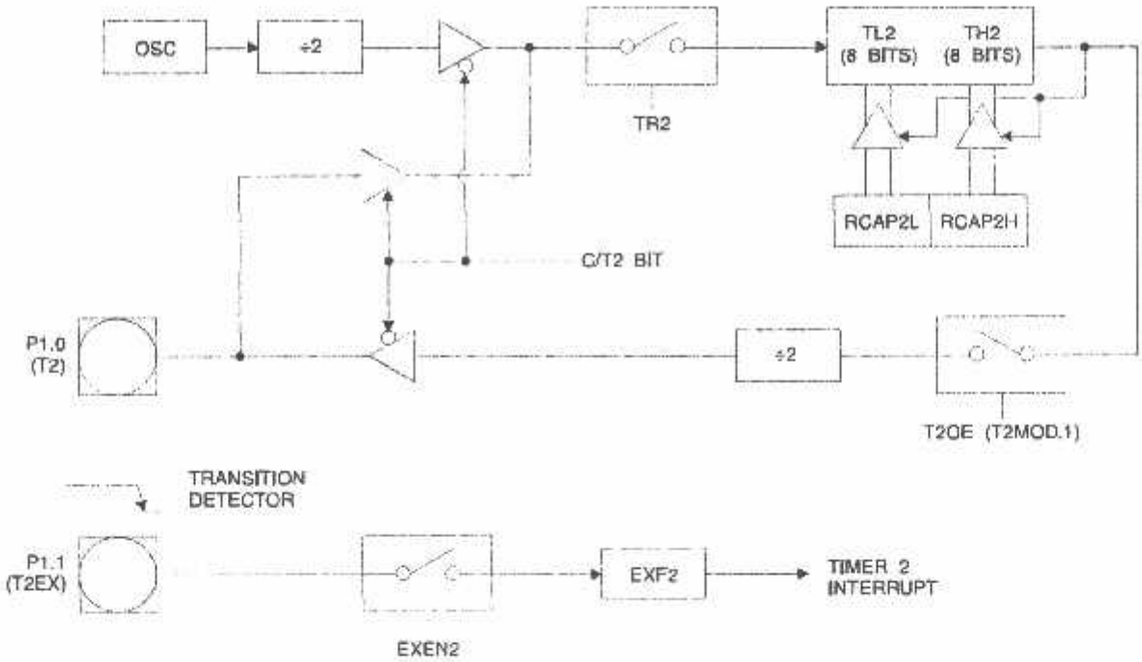
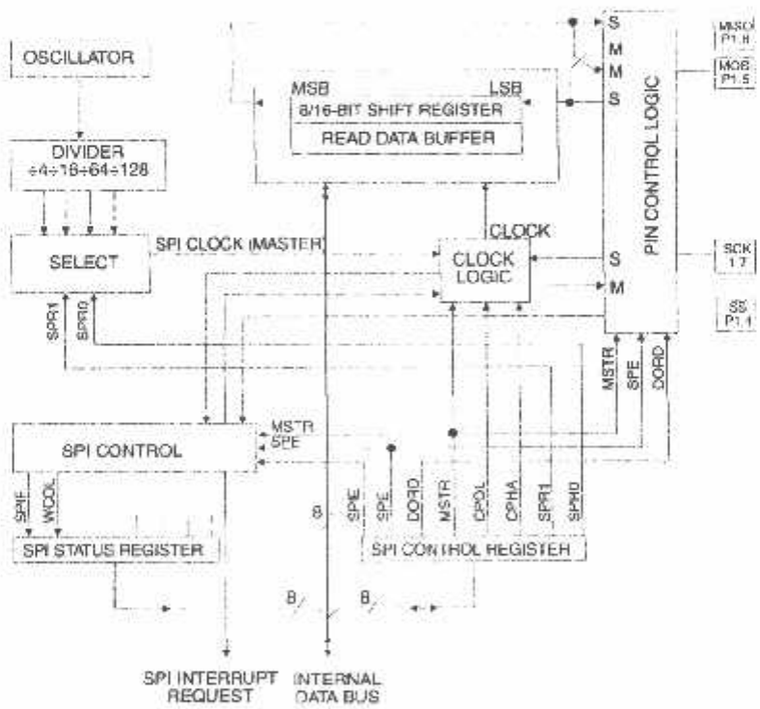


Figure 6. SPI Block Diagram



UART

The UART in the AT89S8252 operates the same way as the UART in the AT89C51 and AT89C52. For further information on the UART operation, refer to the Atmel web site (http://www.atmel.com). From the home page, select "Products", then "Microcontrollers, then "8051-Architecture". Click on "Documentation", then on "Other Documents". Open the document "AT89 Series Hardware Description".

Serial Peripheral Interface

The serial peripheral interface (SPI) allows high-speed synchronous data transfer between the AT89S8252 and peripheral devices or between several AT89S8252 devices. The AT89S8252 SPI features include the following:

- Full-Duplex, 3-Wire Synchronous Data Transfer
- Master or Slave Operation
- 1.5 MHz Bit Frequency (max.)
- LSB First or MSB First Data Transfer
- Four Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wakeup from Idle Mode (Slave Mode Only)

The interconnection between master and slave CPUs with SPI is shown in the following figure. The SCK pin is the clock output in the master mode but is the clock input in the slave mode. Writing to the SPI data register of the master CPU starts the SPI clock generator, and the data written shifts out of the MOSI pin and into the MOSI pin of the slave CPU. After shifting one byte, the SPI clock generator stops, setting the end of transmission flag (SPIF). If both the SPI interrupt enable bit (SPIE) and the serial port interrupt enable bit (ES) are set, an interrupt is requested.

The Slave Select input, $\overline{SS}$/P1.4, is set low to select an individual SPI device as a slave. When $\overline{SS}$/P1.4 is set high, the SPI port is deactivated and the MOSI/P1.5 pin can be used as an input.

There are four combinations of SCK phase and polarity with respect to serial data, which are determined by control bits CPHA and CPOL. The SPI data transfer formats are shown in Figure 8 and Figure 9.

Figure 7. SPI Master-slave Interconnection

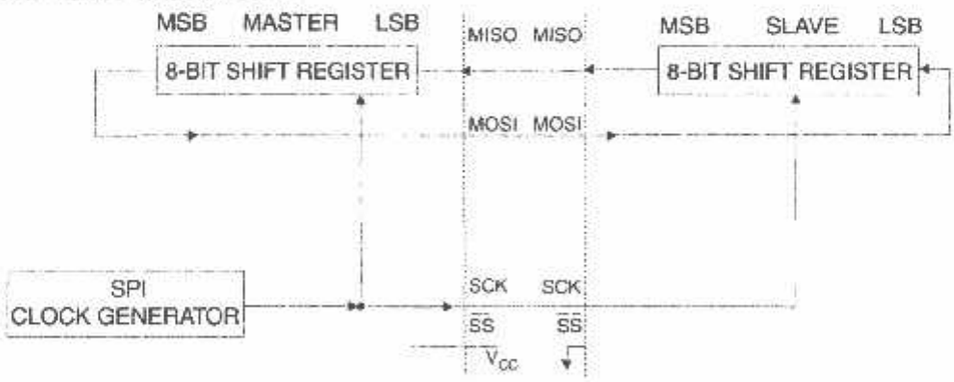
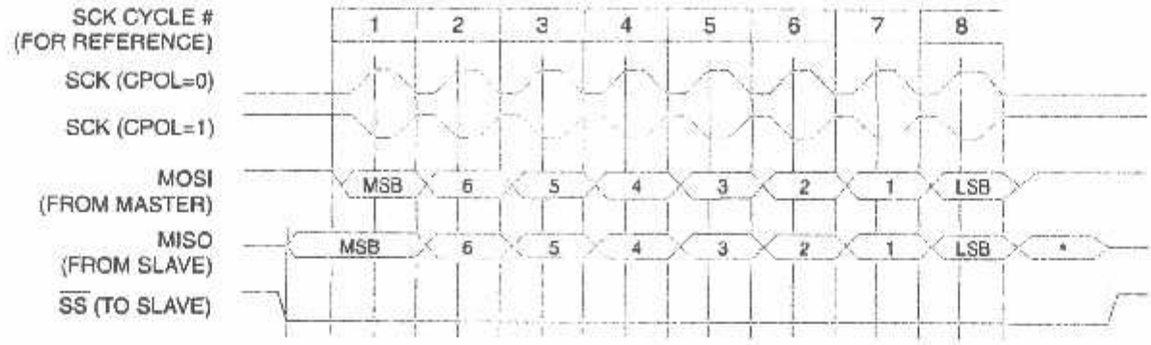
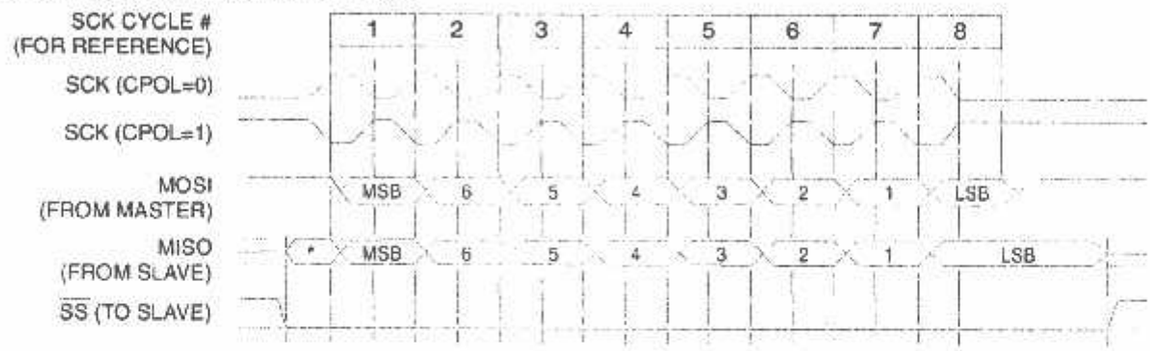


Figure 8. SPI transfer Format with CPHA = 0



Note: *Not defined but normally MSB of character just received

Figure 9. SPI Transfer Format with CPHA = 1



Note: *Not defined but normally LSB of previously transmitted character.

Interrupts

The AT89S8252 has a total of six interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 10.

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 shows that bit position IE.6 is unimplemented. In the AT89C51, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products.

Timer 2 interrupt is generated by the logical OR of bits TF2 and EXF2 in register T2CON. Neither of these flags is cleared by hardware when the service routine is vectored to. In fact, the service routine may have to determine whether it was TF2 or EXF2 that generated the interrupt, and that bit will have to be cleared in software.

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. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

Table 10. Interrupt Enable (IE) Register

(MSB)(LSB)

EA	–	ET2	ES	ET1	EX1	ET0	EX0

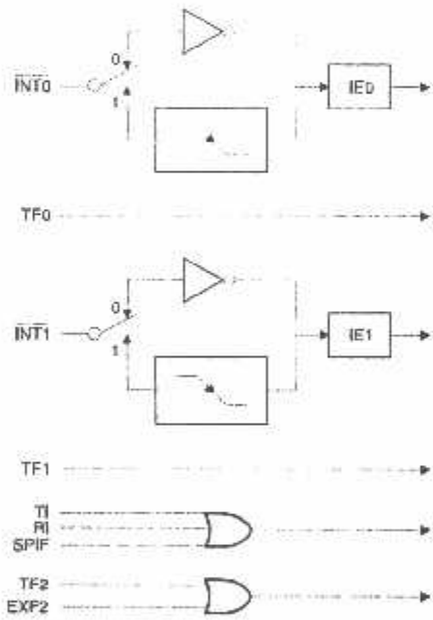
Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

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

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

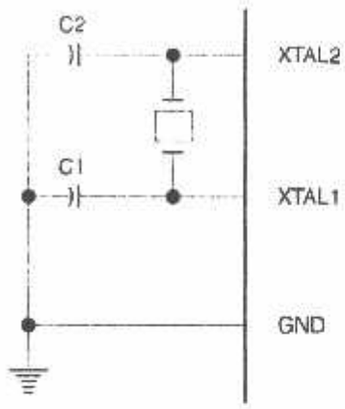
Figure 10. Interrupt Sources



Oscillator Characteristics

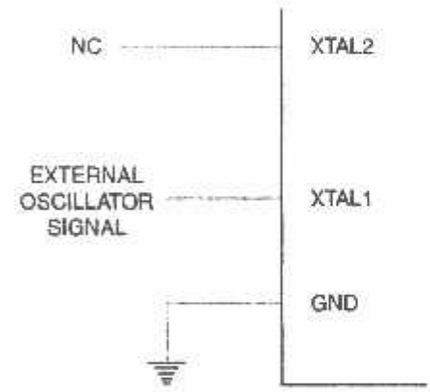
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 11. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 12. 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. Oscillator Connections



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

Figure 12. External Clock Drive Configuration



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 functions 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.

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

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 can be initiated either by a hardware reset or by an enabled external interrupt. 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.

To exit power-down via an interrupt, the external interrupt must be enabled as level sensitive before entering power-down. The interrupt service routine starts at 16 ms (nominal) after the enabled interrupt pin is activated.

Program Memory Lock Bits

The AT89S8252 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the following table.

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.

Once programmed, the lock bits can only be unprogrammed with the Chip Erase operations in either the parallel or serial modes.

Lock Bit Protection Modes[1][2]

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No internal memory lock feature.
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 (parallel or serial mode) is disabled.
3	P	P	U	Same as Mode 2, but parallel or serial verify are also disabled.
4	P	P	P	Same as Mode 3, but external execution is also disabled.

Notes: 1. U = Unprogrammed
2. P = Programmed

Programming the Flash and EEPROM

Atmel's AT89S8252 Flash Microcontroller offers 8K bytes of in-system reprogrammable Flash Code memory and 2K bytes of EEPROM Data memory.

The AT89S8252 is normally shipped with the on-chip Flash Code and EEPROM Data memory arrays in the erased state (i.e. contents = FFH) and ready to be programmed. This device supports a High-voltage (12-V V_{PP}) Parallel programming mode and a Low-voltage (5-V V_{CC}) Serial programming mode. The serial programming mode provides a convenient way to reprogram the AT89S8252 inside the user's system. The parallel programming mode is compatible with conventional third party Flash or EPROM programmers.

The Code and Data memory arrays are mapped via separate address spaces in the serial programming mode. In the parallel programming mode, the two arrays occupy one contiguous address space: 0000H to 1FFFH for the Code array and 2000H to 27FFH for the Data array.

The Code and Data memory arrays on the AT89S8252 are programmed byte-by-byte in either programming mode. An auto-erase cycle is provided with the self-timed programming operation in the serial programming mode. There is no need to perform the Chip Erase operation to reprogram any memory location in the serial programming mode unless any of the lock bits have been programmed.

In the parallel programming mode, there is no auto-erase cycle. To reprogram any non-blank byte, the user needs to use the Chip Erase operation first to erase both arrays.

Parallel Programming Algorithm: To program and verify the AT89S8252 in the parallel programming mode, the following sequence is recommended:

1. Power-up sequence:
   Apply power between V_{CC} and GND pins.
   Set RST pin to "H".
   Apply a 3 MHz to 24 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.
2. Set $\overline{PSEN}$ pin to "L"
   ALE pin to "H"
   $\overline{EA}$ pin to "H" and all other pins to "H".
3. Apply the appropriate combination of "H" or "L" logic levels to pins P2.6, P2.7, P3.6, P3.7 to select one of the programming operations shown in the Flash Programming Modes table.
4. Apply the desired byte address to pins P1.0 to P1.7 and P2.0 to P2.5.
   Apply data to pins P0.0 to P0.7 for Write Code operation.
5. Raise $\overline{EA}/V_{PP}$ to 12V to enable Flash programming, erase or verification.
6. Pulse ALE/$\overline{PROG}$ once to program a byte in the Code memory array, the Data memory array or the lock bits. The byte-write cycle is self-timed and typically takes 1.5 ms.
7. To verify the byte just programmed, bring pin P2.7 to "L" and read the programmed data at pins P0.0 to P0.7.
8. Repeat steps 3 through 7 changing the address and data for the entire 2K or 8K bytes array or until the end of the object file is reached.
9. Power-off sequence:
   Set XTAL1 to "L".
   Set RST and $\overline{EA}$ pins to "L".
   Turn V_{CC} power off.

In the parallel programming mode, there is no auto-erase cycle and to reprogram any non-blank byte, the user needs to use the Chip Erase operation first to erase both arrays.

Data Polling: The AT89S8252 features DATA Polling to indicate the end of a byte write cycle. During a byte write cycle in the parallel or serial programming mode, an attempted read of the last byte written will result in the complement of the written datum on P0.7 (parallel mode), and on the MSB of the serial output byte on MISO (serial mode). Once the write cycle has been completed, true data are valid on all outputs, and the next cycle may begin. DATA Polling may begin any time after a write cycle has been initiated.

Ready/Busy: The progress of byte programming in the parallel programming mode can also be monitored by the RDY/BSY output signal. Pin P3.4 is pulled Low after ALE goes High during programming to indicate BUSY. P3.4 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 or Data byte can be read back via the address and data lines for verification. The state of the lock bits can also be verified directly in the parallel programming mode. In the serial programming mode, the state of the lock bits can only be verified indirectly by observing that the lock bit features are enabled.

Chip Erase: Both Flash and EEPROM arrays are erased electrically at the same time. In the parallel programming mode, chip erase is initiated by using the proper combination of control signals and by holding ALE/PROG low for 10 ms. The Code and Data arrays are written with all "1"s in the Chip Erase operation.

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 16 ms.

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

Serial Programming Fuse: A programmable fuse is available to disable Serial Programming if the user needs maximum system security. The Serial Programming Fuse can only be programmed or erased in the Parallel Programming Mode.

The AT89S8252 is shipped with the Serial Programming Mode enabled.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 030H and 031H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows:

(030H) = 1EH indicates manufactured by Atmel
(031H) = 72H indicates 89S8252

Programming Interface

Every code byte in the Flash and EEPROM arrays can be written, and the entire array can be erased, 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 worldwide major programming vendors offer support for the Atmel AT89 microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Serial Downloading

Both the Code and Data memory arrays can be programmed using the serial SPI bus 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 program/erase operations can be executed.

An auto-erase cycle is built into the self-timed programming operation (in the serial mode ONLY) and there is no need to first execute the Chip Erase instruction unless any of the lock bits have been programmed. The Chip Erase operation turns the content of every memory location in both the Code and Data arrays into FFH.

The Code and Data memory arrays have separate address spaces:

0000H to 1FFFH for Code memory and 000H to 7FFH for Data memory.

Either an external system clock is 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 1/40 of the crystal frequency. With a 24 MHz oscillator clock, the maximum SCK frequency is 600 kHz.

Serial Programming Algorithm

To program and verify the AT89S8252 in the serial programming mode, the following sequence is recommended:

1. Power-up sequence:

   Apply power between VCC and GND pins.

   Set RST pin to "H".

   If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 24 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 40.
3. The Code or Data array is programmed one byte at a time by supplying the address and data together with the appropriate Write instruction. The selected memory location is first automatically erased before new data is written. The write cycle is self-timed and typically takes less than 2.5 ms at 5V.
4. Any memory location can be verified by using the Read instruction which 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 operation.
6. Power-off sequence (if needed):

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

   Set RST to "L".

   Turn V_{CC} power off.

Serial Programming Instruction

The Instruction Set for Serial Programming follows a 3-byte protocol and is shown in the following table:

Instruction Set

Instruction	Input Format			Operation
	Byte 1	Byte 2	Byte 3	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	Enable serial programming interface after RST goes high.
Chip Erase	1010 1100	xxxx x100	xxxx xxxx	Chip erase both 8K & 2K memory arrays.
Read Code Memory	aaaa a001	low addr	xxxx xxxx	Read data from Code memory array at the selected address. The 5 MSBs of the first byte are the high order address bits. The low order address bits are in the second byte. Data are available at pin MISO during the third byte.
Write Code Memory	aaaa a010	low addr	data in	Write data to Code memory location at selected address. The address bits are the 5 MSBs of the first byte together with the second byte.
Read Data Memory	00aa a101	low addr	xxxx xxxx	Read data from Data memory array at selected address. Data are available at pin MISO during the third byte.
Write Data Memory	00aa a110	low addr	data in	Write data to Data memory location at selected address.
Write Lock Bits	1010 1100	LB1 LB2 LB3 x x111	xxxx xxxx	Write lock bits. Set LB1, LB2 or LB3 = "0" to program lock bits.

Notes:
1. DATA polling is used to indicate the end of a byte write cycle which typically takes less than 2.5 ms at 5V.
2. "aaaaa" = high order address.
3. "x" = don't care.

Flash and EEPROM Parallel Programming Modes

Mode	RST	$\overline{PSEN}$	ALE/$\overline{PROG}$	$\overline{EA}$/V_{pp}	P2.6	P2.7	P3.6	P3.7	Data I/O P0.7:0	Address P2.5:0 P1.7:0
Serial Prog. Modes	H	h[1]	h[1]	x						
Chip Erase	H	L	(2)	12V	H	L	L	L	X	X
Write (10K bytes) Memory	H	L		12V	L	H	H	H	DIN	ADDR
Read (10K bytes) Memory	H	L	H	12V	L	L	H	H	DOUT	ADDR
Write Lock Bits:	H	L		12V	H	L	H	L	DIN	X
Bit - 1									P0.7 = 0	X
Bit - 2									P0.6 = 0	X
Bit - 3									P0.5 = 0	X
Read Lock Bits:	H	L	H	12V	H	H	L	L	DOUT	X
Bit - 1									@P0.2	X
Bit - 2									@P0.1	X
Bit - 3									@P0.0	X
Read Atmel Code	H	L	H	12V	L	L	L	L	DOUT	30H
Read Device Code	H	L	H	12V	L	L	L	L	DOUT	31H
Serial Prog. Enable	H	L	(2)	12V	L	H	L	H	P0.0 = 0	X
Serial Prog. Disable	H	L	(2)	12V	L	H	L	H	P0.0 = 1	X
Read Serial Prog. Fuse	H	L	H	12V	H	H	L	H	@P0.0	X

Notes:
1. "h" = weakly pulled "High" internally.
2. Chip Erase and Serial Programming Fuse require a 10 ms $\overline{PROG}$ pulse. Chip Erase needs to be performed first before reprogramming any byte with a content other than FFH.
3. P3.4 is pulled Low during programming to indicate RDY/$\overline{BSY}$.
4. "X" = don't care

Figure 13. Programming the Flash/EEPROM Memory

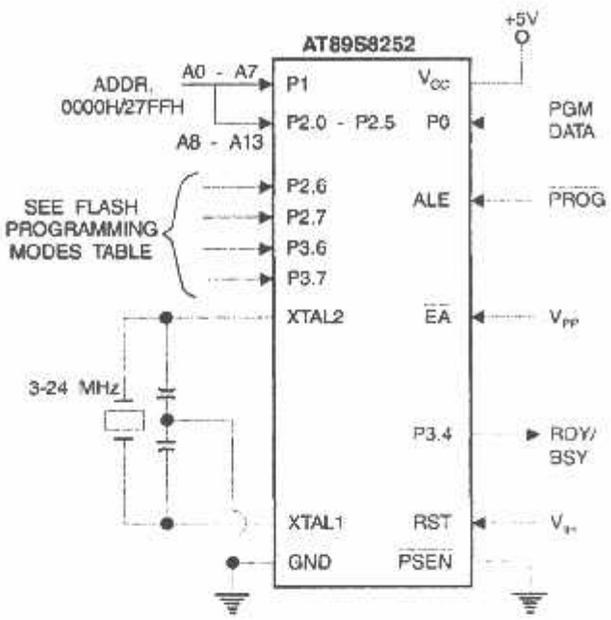


Figure 14. Verifying the Flash/EEPROM Memory

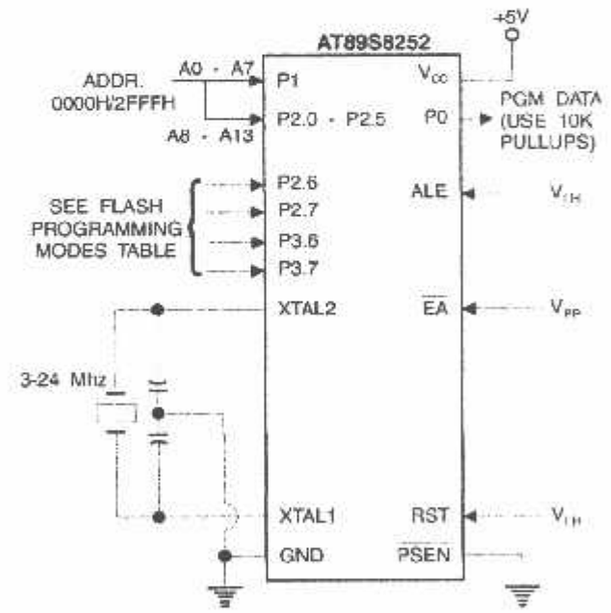
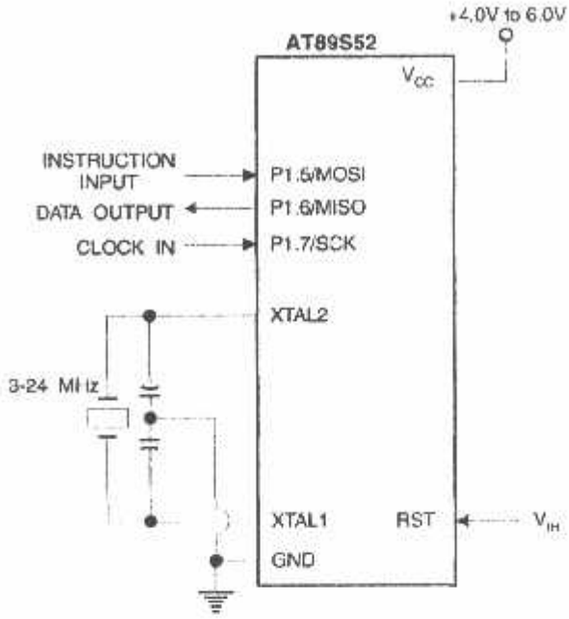


Figure 15. Flash/EEPROM Serial Downloading

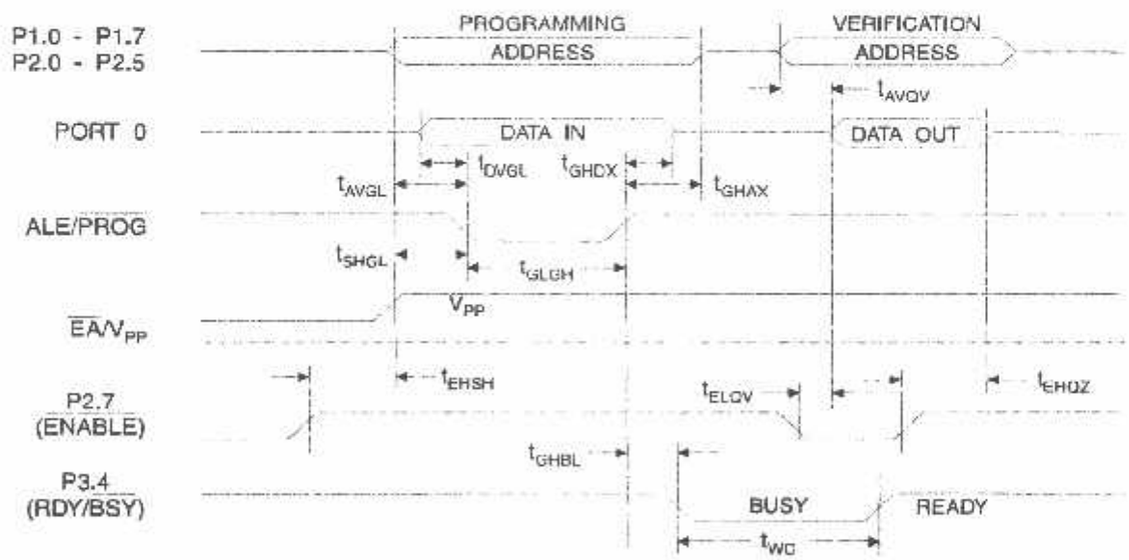


Flash Programming and Verification Characteristics – Parallel Mode

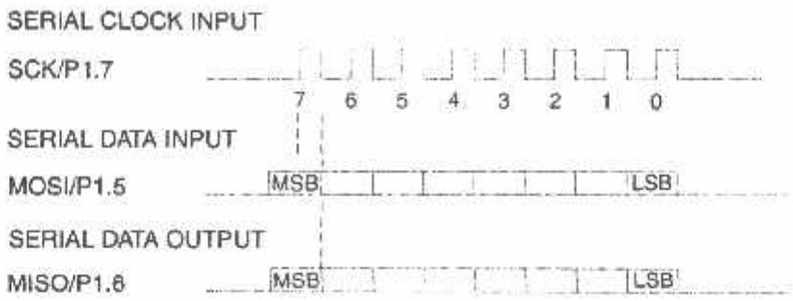
T_A = 0°C to 70°C, V_{CC} = 5.0V ± 10%

Symbol	Parameter	Min	Max	Units
V_{PP}	Programming Enable Voltage	11.5	12.5	V
I_{PP}	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	3	24	MHz
t_{AVGL}	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHAX}	Address Hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{DVGL}	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
t_{GHDX}	Data Hold after $\overline{PROG}$	$48t_{CLCL}$		
t_{EHSH}	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
t_{SHGL}	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
t_{GLGH}	$\overline{PROG}$ Width	1	110	µs
t_{AVQV}	Address to Data Valid		$48t_{CLCL}$	
t_{ELQV}	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
t_{EHQZ}	Data Float after $\overline{ENABLE}$	0	$48t_{CLCL}$	
t_{GHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
t_{WC}	Byte Write Cycle Time		2.0	ms

Flash/EEPROM Programming and Verification Waveforms – Parallel Mode



Serial Downloading Waveforms



Serial Programming Characteristics

Figure 16. Serial Programming Timing

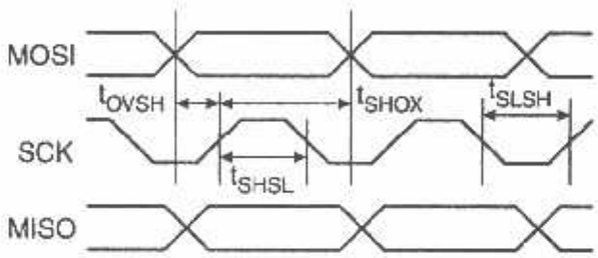


Table 11. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 4.0 - 6.0V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0		24	MHz
t_{CLCL}	Oscillator Period	41.6			ns
t_{SHSL}	SCK Pulse Width High	24 t_{CLCL}			ns
t_{SLSH}	SCK Pulse Width Low	24 t_{CLCL}			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2 t_{CLCL}			ns

Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC 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.

DC Characteristics

The values shown in this table are valid for T_A = -40°C to 85°C and V_{CC} = 5.0V ± 20%, 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		0.5	V
V_{OL1}	Output Low-voltage [1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.5	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%		-650	µA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
RRST	Reset Pull-down 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} = 6V		100	µA
		V_{CC} = 3V		40	µA

Notes:
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

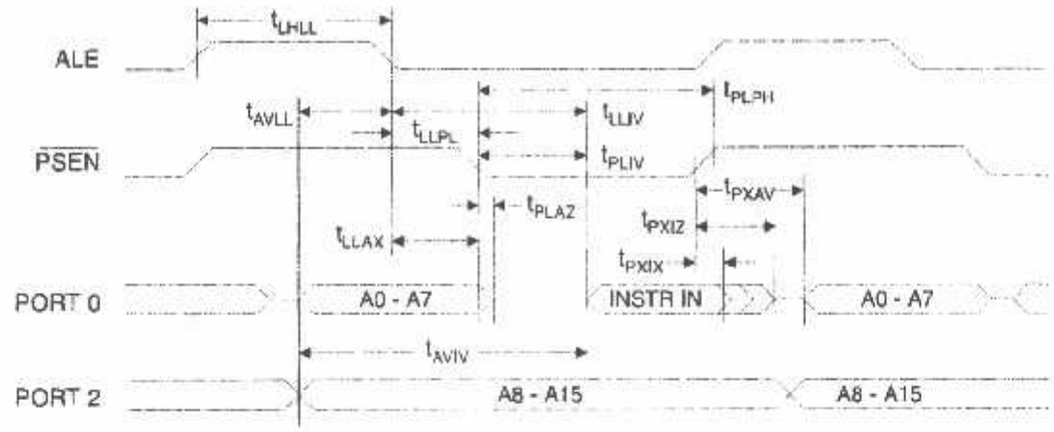
AC Characteristics

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

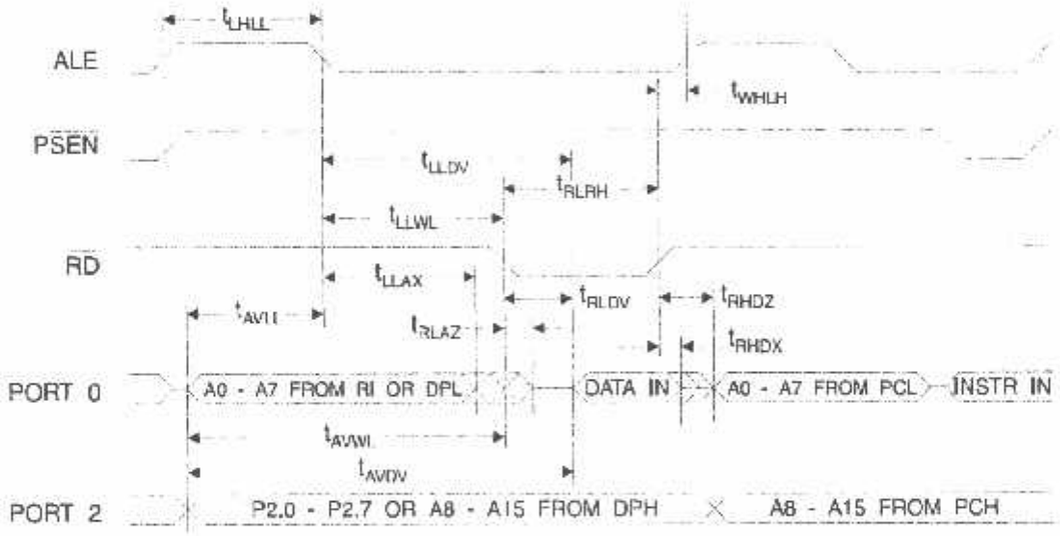
External Program and Data Memory Characteristics

Symbol	Parameter	Variable Oscillator		Units
		Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{LHLL}	ALE Pulse Width	$2t_{CLCL}$ - 40		ns
t_{AVLL}	Address Valid to ALE Low	t_{CLCL} - 13		ns
t_{LLAX}	Address Hold after ALE Low	t_{CLCL} - 20		ns
t_{LLIV}	ALE Low to Valid Instruction In		$4t_{CLCL}$ - 65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	t_{CLCL} - 13		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	$3t_{CLCL}$ - 20		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		$3t_{CLCL}$ - 45	ns
t_{PXIX}	Input Instruction Hold after $\overline{PSEN}$	0		ns
t_{PXIZ}	Input Instruction Float after $\overline{PSEN}$		t_{CLCL} - 10	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	t_{CLCL} - 8		ns
t_{AVIV}	Address to Valid Instruction In		$5t_{CLCL}$ - 55	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	$6t_{CLCL}$ - 100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	$6t_{CLCL}$ - 100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		$5t_{CLCL}$ - 90	ns
t_{RHDX}	Data Hold after $\overline{RD}$	0		ns
t_{RHDZ}	Data Float after $\overline{RD}$		$2t_{CLCL}$ - 28	ns
t_{LLDV}	ALE Low to Valid Data In		$8t_{CLCL}$ - 150	ns
t_{AVDV}	Address to Valid Data In		$9t_{CLCL}$ - 165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	$3t_{CLCL}$ - 50	$3t_{CLCL}$ + 50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	$4t_{CLCL}$ - 75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	t_{CLCL} - 20		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	$7t_{CLCL}$ - 120		ns
t_{WHQX}	Data Hold after $\overline{WR}$	t_{CLCL} - 20		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{CLCL} - 20	t_{CLCL} + 25	ns

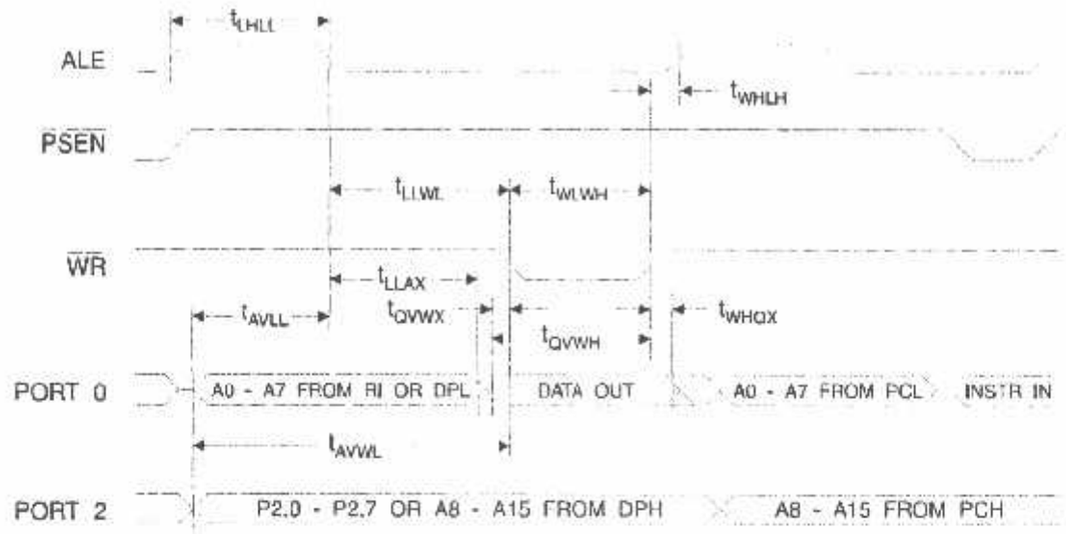
External Program Memory Read Cycle



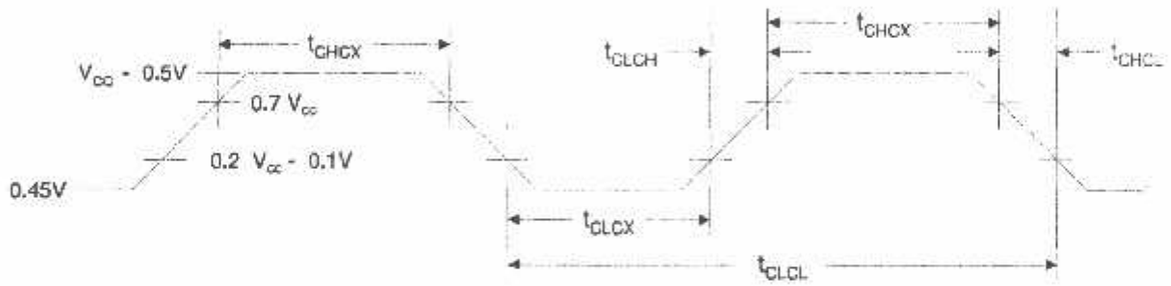
External Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

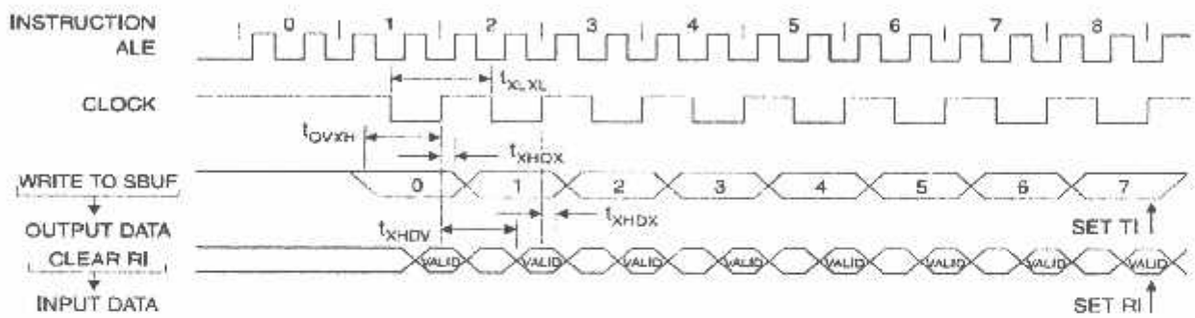
Symbol	Parameter	V_{CC} = 4.0V to 6.0V		Units
		Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	15		ns
t_{CLCX}	Low Time	15		ns
t_{CLCH}	Rise Time		20	ns
t_{CHCL}	Fall Time		20	ns

Serial Port Timing: Shift Register Mode Test Conditions

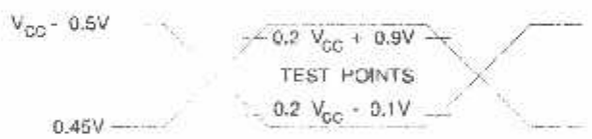
The values in this table are valid for V_{CC} = 4.0V to 6V and Load Capacitance = 80 pF.

Symbol	Parameter	Variable Oscillator		Units
		Min	Max	
t_{XLXL}	Serial Port Clock Cycle Time	$12t_{CLCL}$		µs
t_{QVXH}	Output Data Setup to Clock Rising Edge	$10t_{CLCL}$ - 133		ns
t_{XHQX}	Output Data Hold after Clock Rising Edge	$2t_{CLCL}$ - 117		ns
t_{XHDX}	Input Data Hold after Clock Rising Edge	0		ns
t_{XHDV}	Clock Rising Edge to Input Data Valid		$10t_{CLCL}$ - 133	ns

Shift Register Mode Timing Waveforms



AC Testing Input/Output Waveforms(1)



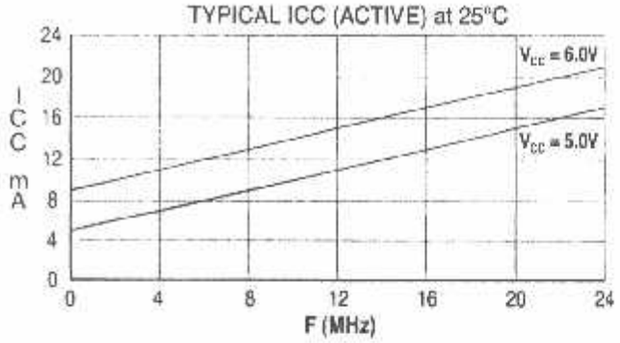
Note: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

Float Waveforms(1)

Note: 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.

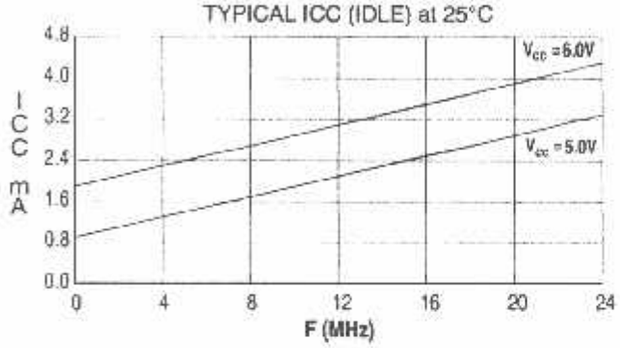
AT89S8252

TYPICAL ICC (ACTIVE) at 25°C



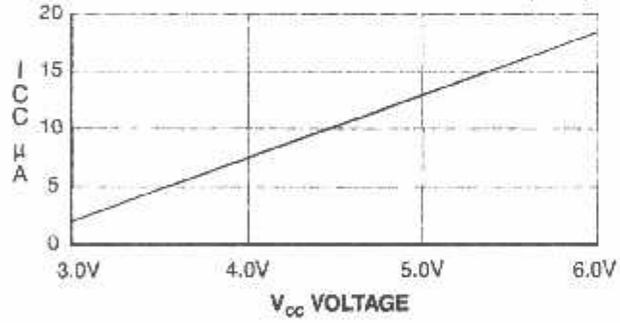
AT89S8252

TYPICAL ICC (IDLE) at 25°C



AT89S8252

TYPICAL ICC vs. VOLTAGE - POWER DOWN (85°C)



Notes: 1. XTAL1 tied to GND for Icc (power-down)
2. Lock bits programmed

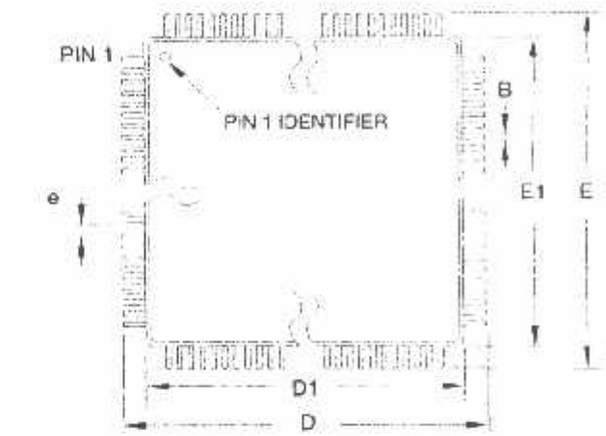
Ordering Information

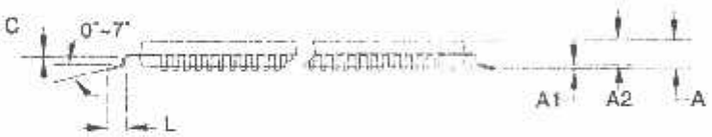
Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 6.0V	AT89S8252-24AC	44A	Commercial (0°C to 70°C)
		AT89S8252-24JC	44J	
		AT89S8252-24PC	40P6	
	4.0V to 6.0V	AT89S8252-24AI	44A	Industrial (-40°C to 85°C)
		AT89S8252-24JI	44J	
		AT89S8252-24PI	40P6	

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

Packaging Information

44A – TQFP





COMMON DIMENSIONS
(Unit of Measure = mm)

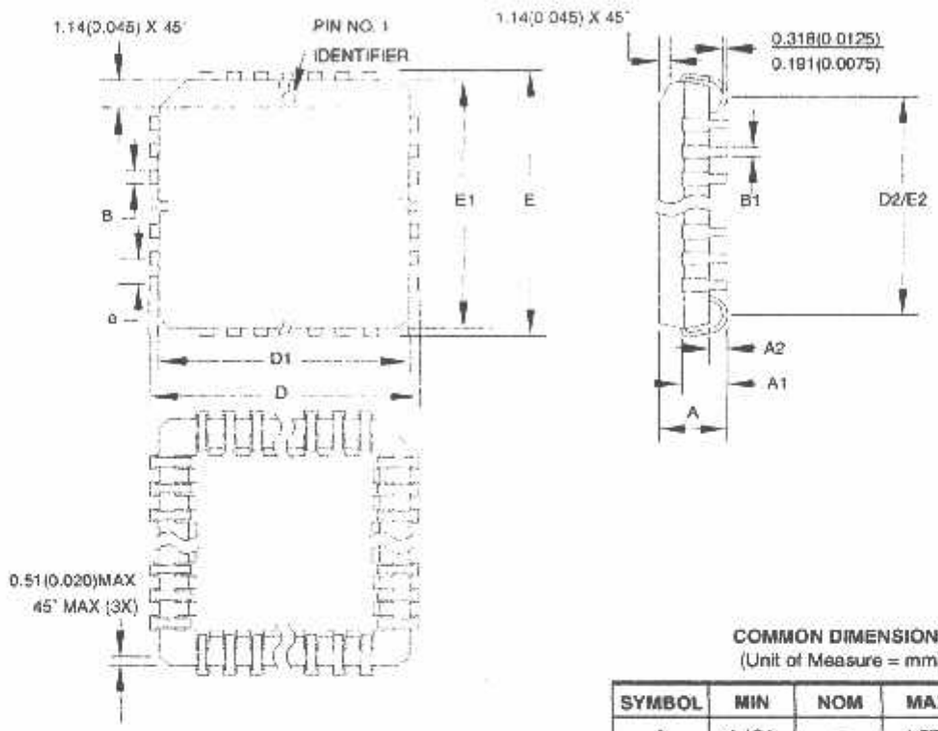
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

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

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**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)	44A	B

44J – PLCC



COMMON DIMENSIONS
(Unit of Measure = mm)

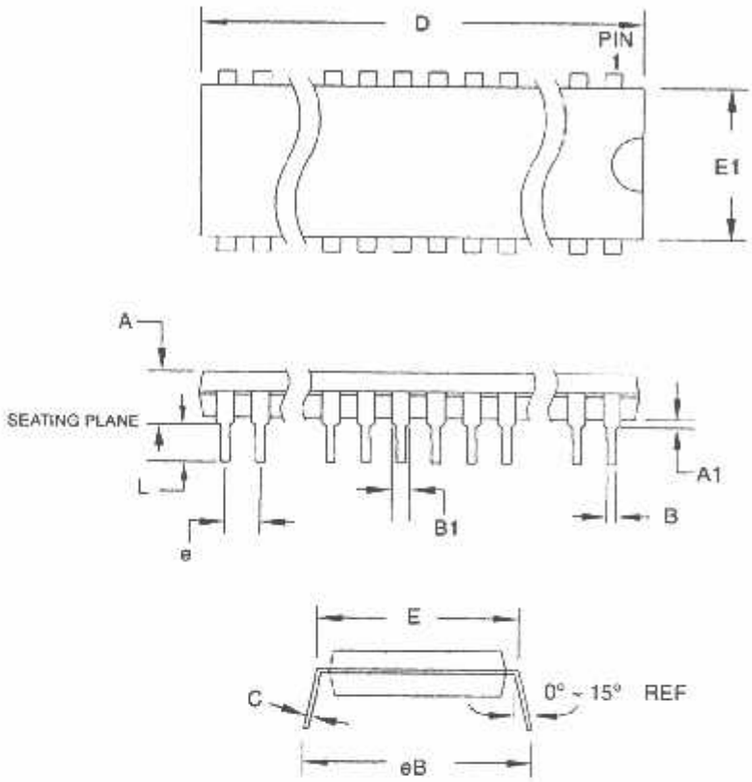
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

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.
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.
2325 Orchard Parkway San Jose, CA 95131	**40P6**, 40-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	40P6	B

Atmel Corporation

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

Regional Headquarters

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

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

Japan
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

Printed on recycled paper.

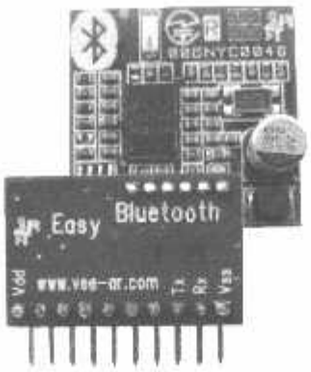
0401G–MICRO–3/06 xM

PARALLAX

Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

Easy Bluetooth (#30085)



The Easy Bluetooth is a RoboTech RBT-001 Bluetooth serial module with an adapter specifically designed to be used with the Parallax Board of Education® AppMod Header or breadboard. In addition, the Easy Bluetooth can plug into any 0.1 inch spacing development platform which allows it to be breadboard friendly, and also easy to implement into a soldering application. The module has two parts, the RBT-001 module and the SIP with voltage regulator PCB. With the on-board regulator, the module can be connected to voltages higher than 3.3 VDC, such as the Board of Education regulated supply (+5 VDC) without worry of damaging the unit; while the RX and TX can utilize serial communication at CMOS and TTL levels.

Features

- 1.x & 2.0 Bluetooth Compliant
- Class 2 Operation (nominal range up to 30 meters)
- 10-pin SIP package for breadboard, perfboard, or Board of Education AppMod Header
- On-board regulator for safe operations across various voltages
- CMOS & TTL Compatible

Key Specifications

- Power requirements: 3.3 to 5.5 VDC
- Communication: UART Command/Data Port supports for up to 921.6k baud
- Operating temperature: +32 to +113 °F (0 to +45 °C)
- Dimensions: 1.40 x 1.79 x .49 in (34.41 x 45.65 x 12.51 mm)

Application Ideas

- Control a Boe-Bot via Bluetooth from a PC, Cell Phone, or another Bluetooth module
- Communicate with a device or project wirelessly

Packing List

- RBT-001 Module
- 10-pin SIP with connector Module (pre-assembled with RTB-001)

Quick Start Circuit for the BASIC Stamp 2 and Board of Education

There are a few steps to take to install an Easy Bluetooth module, creating a Bluetooth connection on a PC operating with Windows XP, and finally writing a program for the BASIC Stamp® 2. Once you are done, you will have a working Easy Bluetooth module communicating with the PC via Bluetooth.

Installing the Easy Bluetooth Module

1. Carefully open the Easy Bluetooth package and check that the two small boards are properly plugged into each other as shown in Figure 1.

2. Now plug the Easy Bluetooth module into the AppMod header of the Board of Education: insert the module in the left row of the AppMod Header and notice there are labels on the module that indicate correct pin placement (Rx uses P0, Tx uses P2, Vss with Vss and Vdd with Vdd). Visually confirm the module is inserted correctly using Figure 2 before powering.

3. Connect a communication cable and power supply to the Board of Education and turn the power switch to location 1.

Figure 1

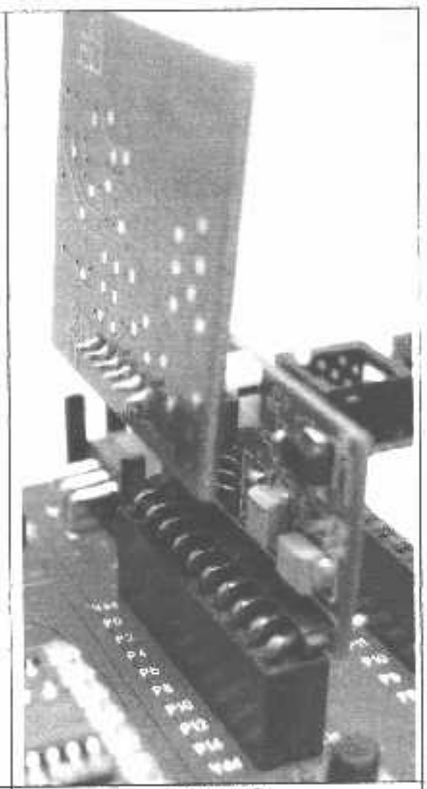


Figure 2

Creating a new Bluetooth Connection

1. Be sure that the Bluetooth dongle that you installed on your PC is working correctly and is turned on (see Bluetooth Dongle user manual for specific instructions to ensure proper operation).

    a. Open Control Panel, click "Printer and Other Hardware" and then click Bluetooth Devices. If in Classic View double click "Bluetooth Devices".

    b. Click "Add".

c. Check "My Device is set up and ready to be found" in the "Add Bluetooth Device Wizard" and click "Next".

d. Select "EasyBT" from the displayed Bluetooth devices and click "Next".

e. Select "Use the passkey found in the documentation" and enter the Passkey Code 0000 (zero four times) and click "Next".

f. Bluetooth Manager should display and designate an Outgoing and Incoming COM port for the Easy Bluetooth device: make a note of the COM ports because these are the ports you will use to communicate with the Easy Bluetooth.

g. Select "Finish" to complete the Bluetooth device configuration.

2. To close the "Bluetooth Devices" window click "OK".

A test for the BASIC Stamp 2 microcontroller

1. The Easy Bluetooth can communicate at numerous baud rates, and with many different microcontrollers. This example uses the AppMod Header on the Board of Education, the BASIC Stamp 2, at 9600 baud.

2. The easiest way to see in your code if a Bluetooth connection has been established is to wait for a byte to be received. You can copy the program below into the BASIC Stamp Editor to ensure a proper connection.

```
' {$STAMP BS2}
' {$PBASIC 2.5}

RX            CON     2       'Receive Pin
TX            CON     0       'Transmit Pin
Baud          CON     84      '9600 Baud

combyte          VAR Byte     'Communication Byte

DEBUG "Use This Screen for Display",CR

DO
  'Wait for a first byte indicating an active Bluetooth connection
  SERIN RX, Baud, [combyte]
  DEBUG combyte
LOOP
```

3. Download the program using Run => Run (ctrl + r) from the BASIC Stamp Editor

4. Keep Current DEBUG Screen open, and open another DEBUG Screen (ctrl +d) from the BASIC Stamp Editor, and select TX COM that the Bluetooth module (refer to COM ports during Bluetooth installation) is using; you should now have 2 DEBUG Terminal windows open. Click in the white area of the DEBUG window of DEBUG Terminal 2 and press any key; if all the steps were correctly done, the same key will be sent back to the DEBUG Terminal 1 window.

Resources and Downloads

You may download the RBT-001 user manual, example source codes, PC applications, and mobile phone application from www.parallax.com.

*Mobile phone must be compatible with the JSR-82 Java API; usually all Nokia and Sony Ericsson phone are compatible, but check users manual for compatibility. Configuration of mobile phone is not supported by Parallax Technical Support.

Device Information

Theory of Operation

Bluetooth is an open wireless protocol for exchanging data over short distances from fixed and mobile devices, creating Personal Area Networks (PANs). It was originally conceived as a wireless alternative to RS232 data cables. It can connect several devices, overcoming problems of synchronization.

There are various versions of Bluetooth communication and the Easy Bluetooth is compatible with versions 1.x and 2.0.

Precautions

The Easy Bluetooth uses the microwave radio frequency spectrum in the 2.4 GHz to 2.4835 GHz range. Maximum power output from this Bluetooth radio 2.5 mW; Class 2 devices are considered less of a potential hazard than mobile phones.

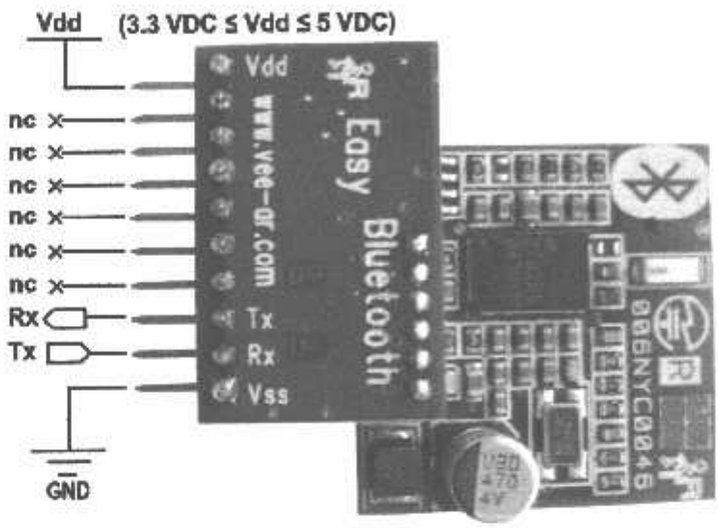
Pin Definitions and Ratings

Pin	Name	Function
1	Vdd	+3.3 to +5 VDC (+5.5 max voltage)
2	RX	Receive Serial Communication (CMOS & TTL Level Compatible)
3	TX	Transmit Serial Communication (CMOS & TTL Level Compatible)
4	NC	Not Connected
5	NC	Not Connected
6	NC	Not Connected
7	NC	Not Connected
8	NC	Not Connected
9	NC	Not Connected
10	Vss	Ground

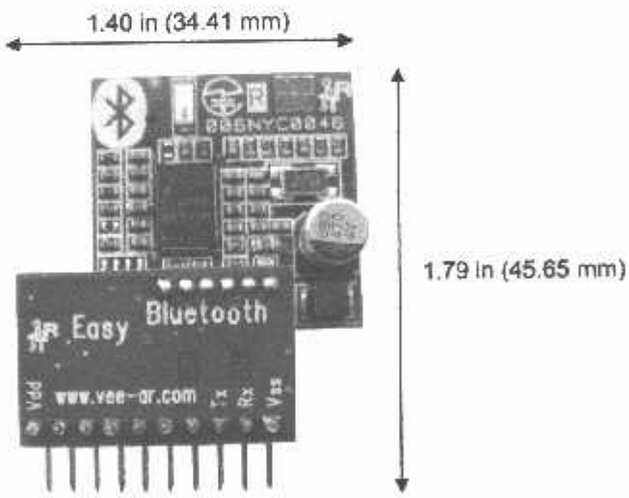
Communication Protocol

UART Command/Data Port supports up to 921.6k non-inverted baud.

Connection Diagram



Module Dimensions



Example Application

This example application uses the RoboTech SRL GUI interface to control a Boe-Bot® robot via Bluetooth.

*A note to mention is that this application uses a standard operational Boe-Bot; so please have an operational Boe-Bot ready while completing this example. For help building a fully operational Boe-Bot, please refer to Robotics with the Boe-Bot Text.

To install the GUI software, complete the following steps:

1. Install the Easy Bluetooth PC software from the "Easy Bluetooth Software.zip" file with all the default and install locations as prompted by the installer. At the end of the installations you will find a new program folder "Easy Bluetooth" under the "Parallax Inc" main folder named with a sub folder named "BoeBot Remote Control", in your Start Menu.

2. Select "BoeBot Remote Control" application from the sub-folder listed in the previous step.

3. Click "Options" and select the COM port that was previously established with the Bluetooth configuration.

4. Download the program below using the BASIC Stamp Editor (ctrl + r) to the BASIC Stamp 2 and then download the program below to the BASIC Stamp 2 using Run => Run or (crtl + r).

5. Disconnect the Boe-Bot from PC

6. Click "Connect" in the BoeBot Remote Control application to gain control of the Boe-Bot.

BASIC Stamp® 2 Program

```
'' =================================================================================
'
'   File..... Easy Bluetooth.bs2
'   Purpose... Control a Boe-Bot with a GUI interface
'   Author.... Technical Support & RoboTech SRL
'   E-mail.... support@parallax.com
'   Started... April 1st 2009
'   Updated... N/A
'
'   {$STAMP BS2}
'   {$PBASIC 2.5}
'
' =================================================================================


' -----[ Program Description ]---------------------------------------

' This program is what is loaded into the BASIC Stamp prior to using the GUI interface
' from RoboTech SLR. It allows a Boe-Bot to be controlled via a GUI interface.
'
' Controls are as followed:
'
'     Up Arrow    = Forward
'     Down Arrow  = Backwards
'     Left Arrow  = Left Turn
'     Right Arrow = Right Turn
```

```
'     Space Bar    = Stop
'
' You can press and hold and release an arrow key for 2 seconds to access a double
' speed of each action. For example, Press Up Arrow, release to make the Boe-Bot move
' faster forward.

' -----[ I/O Definitions ]-------------------------------------------------

BT_RX           PIN     2               ' RX of the Easy Bluetooth
BT_TX           PIN     0               ' TX of the Easy Bluetooth
LED             PIN     5               ' Indicator LED for Bluetooth Connection

' -----[ Constants ]-------------------------------------------------------

BT_SPEED        CON     84              ' baud 9600 true UART

' -----[ Variables ]-------------------------------------------------------

tLeft           VAR     Word            ' Left Servo control pulse durations
tRight          VAR     Word            ' Right Servo control pulse durations
temp            VAR     Word            ' Temp variable

' Buffer array not declared as buffer VAR Word(5) for SERIN functionality.
' It can still be accessed as buffer(0), buffer(1), etc.  However,
' buffer0, buffer1, etc. should be used in SERIN commands with variations
' of WAIT.

  buffer0       VAR     Byte            ' Buffer - Start char = $ff
  buffer1       VAR     Byte            ' Message Index value
  buffer2       VAR     Byte            ' Command
  buffer3       VAR     Byte            ' Argument 1 (return data 1)
  buffer4       VAR     Byte            ' Argument 2 (return data 2)
  buffer        VAR     buffer0         ' For standard array indexing
  msgIndex      VAR     Byte            ' message index
  rxc           VAR     Byte            ' Receive Clear

' -----[ Initialization ]--------------------------------------------------

Program_Start:
  LOW LED
  DEBUG CLS


  PAUSE 1000                            ' Wait for the RBT-001 radio to be ready.

  msgIndex = 0


  buffer0 = $FF                         ' Connection packet
  buffer1 = msgIndex
  buffer2 = $CC
  buffer3 = 100
  buffer4 = 100
  GOSUB Set_Servo_Speed


  DEBUG CR,"Waiting connection..."  ' wait for connection request
  SERIN BT_RX, BT_SPEED, [WAITSTR buffer \ 3, buffer3, buffer4]

' -----[ Program Code ]----------------------------------------------------

DO
```

```
  SELECT buffer2
  CASE $CC                          ' Connect
    HIGH LED
    msgIndex = 0
    DEBUG CR,"Connected"
    GOSUB Reply
  CASE $DD                          ' Disconnect
    LOW LED
    DEBUG CR,"Disconnected"
    GOSUB Reply
    GOTO Program_Start
  CASE $11                          ' Servo
    DEBUG CR,"Servo"
    GOSUB Set_Servo_Speed
    GOSUB Reply
  CASE ELSE                         ' Error
    buffer2 = $EE
    GOSUB Reply
  ENDSELECT

 Resume:                            ' If Message not rcvd, try again

 PULSOUT 13, tLeft                  ' Servo control pulses
 PULSOUT 12, tRight

 SERIN BT_RX, BT_SPEED, 10, Resume, ' Get next command
      [WAITSTR buffer \ 2, buffer2,
      buffer3, buffer4]

 PULSOUT 13, tLeft                  ' Servo control pulses again
 PULSOUT 12, tRight
LOOP

Reply:
  msgIndex = msgIndex + 1           ' Increment message index for reply.
  buffer1 = msgIndex                ' Next message from PC has to use reply's buf[1].
  SEROUT BT_TX, BT_SPEED, [STR buffer \5]


' -----[ Subroutines ]-----------------------------------------------

Set_Servo_Speed:                    ' Maps to 650 to 850 with 750 stopped.
  tLeft = buffer3 + 650             ' Decode servo speed.
  tRight = buffer4 + 650
RETURN

' -----[ Data ]------------------------------------------------------------

ResetOnOff          DATA 0          ' On/off toggle w/ Reset button
RequestConnect      DATA $FF, 0, 1, 0, 0
ConnectionGranted   DATA $FF, 0, 2, 0, 0
RequestCommand      DATA $FF, 0, 3, 0, 0
ServoSpeeds         DATA $FF, 0, 4, 0, 0
```

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised April 2000

DM74LS164
8-Bit Serial In/Parallel Out Shift Register

General Description

These 8-bit shift registers feature gated serial inputs and an asynchronous clear. A low logic level at either input inhibits entry of the new data, and resets the first flip-flop to the low level at the next clock pulse, thus providing complete control over incoming data. A high logic level on either input enables the other input, which will then determine the state of the first flip-flop. Data at the serial inputs may be changed while the clock is HIGH or LOW, but only information meeting the setup and hold time requirements will be entered. Clocking occurs on the LOW-to-HIGH level transition of the clock input. All inputs are diode-clamped to minimize transmission-line effects.

Features

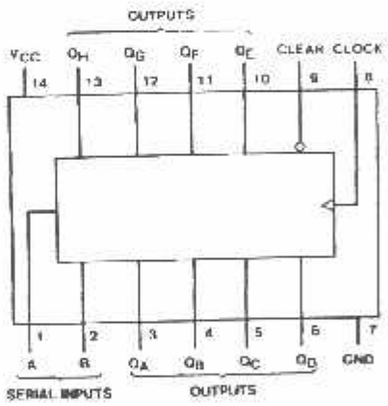
- Gated (enable/disable) serial inputs
- Fully buffered clock and serial inputs
- Asynchronous clear
- Typical clock frequency 36 MHz
- Typical power dissipation 80 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS164M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS164N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code

Connection Diagram

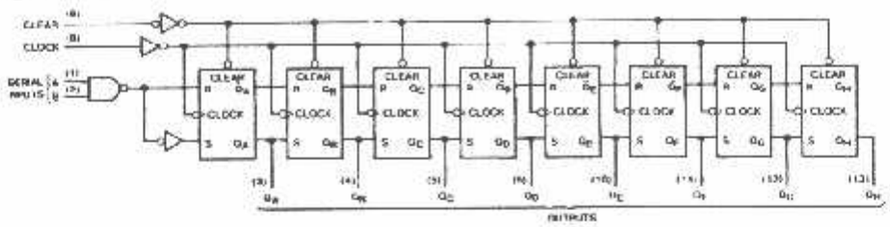


Function Table

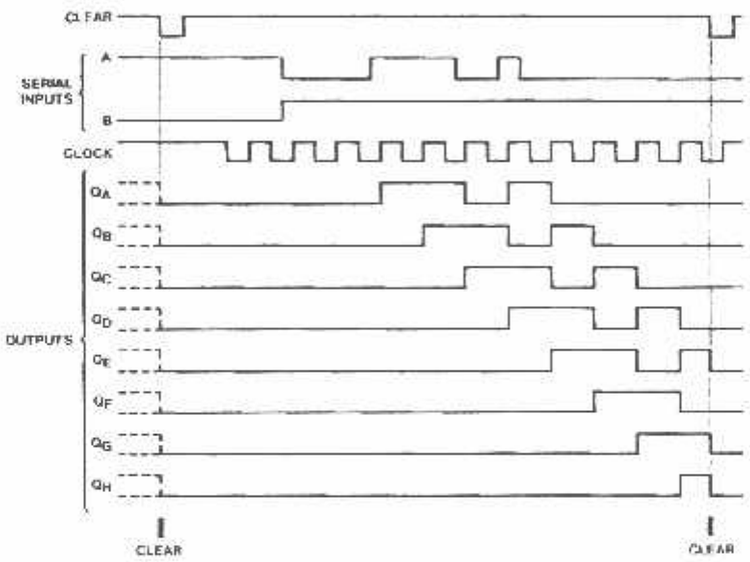
Inputs				Outputs			
Clear	Clock	A	B	Q_A	Q_B	...	Q_H
L	X	X	X	L	L	...	L
H	↑	X	X	Q_{A0}	Q_{B0}	...	Q_{H0}
H	↑	H	H	H	Q_{An}	...	Q_{Gn}
H	↑	L	X	L	Q_{An}	...	Q_{Gn}
H	↑	X	L	L	Q_{An}	...	Q_{Gn}

H = HIGH Level (steady state)
L = LOW Level (steady state)
X = Don't Care (any input, including transitions)
↑ = Transition from LOW-to-HIGH level
Q_{A0}, Q_{B0}, Q_{H0} = The level of Q_A, Q_B, or Q_H, respectively, before the indicated steady-state input conditions were established.
Q_{An}, Q_{Gn} = The level of Q_A or Q_G before the most recent ↑ transition of the clock, indicates a one-bit shift.

Logic Diagram



Timing Diagram



Absolute Maximum Ratings(Note 1)

Supply Voltage	7V
Input Voltage	7V
Operating Free Air Temperature Range	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" tables will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter		Min	Nom	Max	Units
V_{CC}	Supply Voltage		4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage		2			V
V_{IL}	LOW Level Input Voltage				0.8	V
I_{OH}	HIGH Level Output Current				−0.4	mA
I_{OL}	LOW Level Output Current				8	mA
f_{CLK}	Clock Frequency (Note 2)		0		25	MHz
t_W	Pulse Width (Note 2)	Clock	20			ns
		Clear	20			
t_{SU}	Data Setup Time (Note 2)		17			ns
t_H	Data Hold Time (Note 2)		5			ns
t_{REL}	Clear Release Time (Note 2)		30			ns
T_A	Free Air Operating Temperature		0		70	°C

Note 2: T_A = 25°C and V_{CC} = 5V.

Electrical Characteristics

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 3)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I = −18 mA			−1.5	V
V_{OH}	HIGH Level Output Voltage	V_{CC} = Min, I_{OH} = Max, V_{IL} = Max, V_{IH} = Min	2.7	3.4		V
V_{OL}	LOW Level Output Voltage	V_{CC} = Min, I_{OL} = Max, V_{IL} = Max, V_{IH} = Min		0.35	0.5	V
		I_{OL} = 4 mA, V_{CC} = Min		0.25	0.4	
I_I	Input Current @ Max Input Voltage	V_{CC} = Max, V_I = 7V			0.1	mA
I_{IH}	HIGH Level Input Current	V_{CC} = Max, V_I = 2.7V			20	µA
I_{IL}	LOW Level Input Current	V_{CC} = Max, V_I = 0.4V			−0.4	mA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 4)	−20		−100	mA
I_{CC}	Supply Current	V_{CC} = Max (Note 5)		16	27	mA

Note 3: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 4: Not more than one output should be shorted at a time, and the duration should not exceed one second.

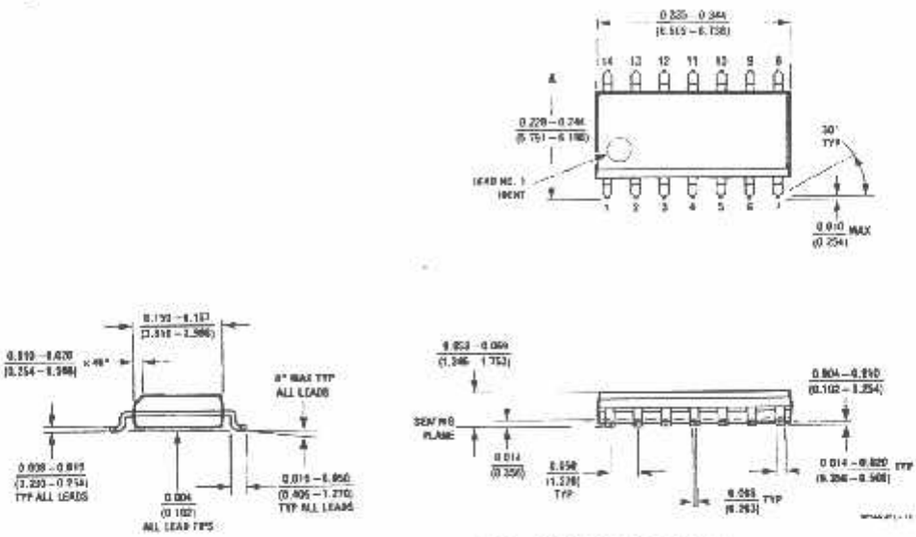
Note 5: I_{CC} is measured with all outputs OPEN, the SERIAL input grounded, the CLOCK input at 2.4V, and a momentary ground, then 4.5V, applied to the CLEAR input.

Switching Characteristics

at V_{CC} = 5V and T_A = 25°C

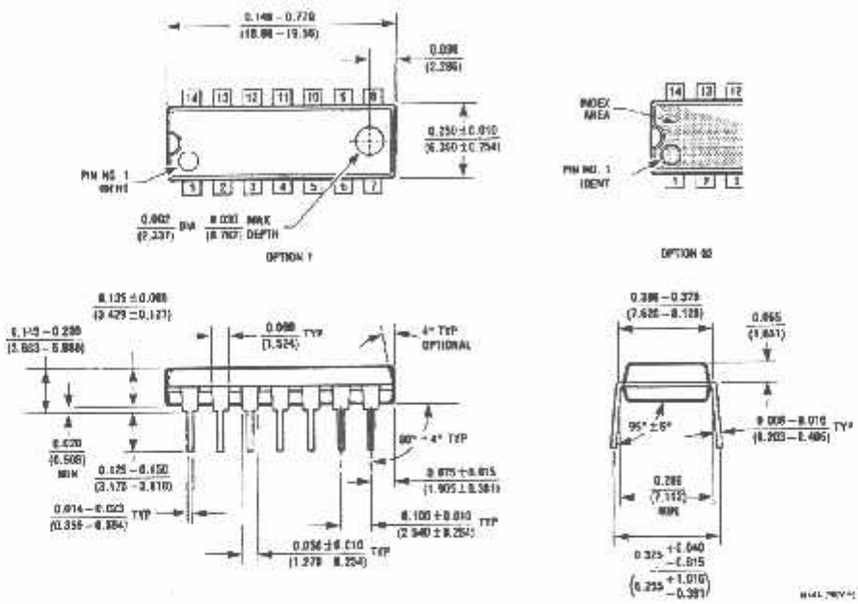
Symbol	Parameter	From (Input) To (Output)	R_L = 2 kΩ				Units
			C_L = 15 pF		C_L = 50 pF		
			Min	Max	Min	Max	
f_{MAX}	Maximum Clock Frequency		25				MHz
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Clock to Output		27		30	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clock to Output		32		40	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clear to Output		36		45	ns

Physical Dimensions inches (millimeters) unless otherwise noted



14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
Package Number M14A

Physical Dimensions inches (millimeters) unless otherwise noted (Continued)



14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide
Package Number N14A

FAIRCHILD
SEMICONDUCTOR TM

TIP32 Series(TIP32/32A/32B/32C)

Medium Power Linear Switching Applications

- Complement to TIP31/31A/31B/31C

TO-220

1.Base 2.Collector 3.Emitter

PNP Epitaxial Silicon Transistor

Absolute Maximum Ratings T_C=25°C unless otherwise noted

Symbol	Parameter		Value	Units
V_{CBO}	Collector-Base Voltage	: TIP32	- 40	V
		: TIP32A	- 60	V
		: TIP32B	- 80	V
		: TIP32C	- 100	V
V_{CEO}	Collector-Emitter Voltage	: TIP32	- 40	V
		: TIP32A	- 60	V
		: TIP32B	- 80	V
		: TIP32C	-100	V
V_{EBO}	Emitter-Base Voltage		- 5	V
I_C	Collector Current (DC)		- 3	A
I_{CP}	Collector Current (Pulse)		- 5	A
I_B	Base Current		- 3	A
P_C	Collector Dissipation (T_C=25°C)		40	W
P_C	Collector Dissipation (T_a=25°C)		2	W
T_J	Junction Temperature		150	°C
T_{STG}	Storage Temperature		- 65 ~ 150	°C

Electrical Characteristics T_C=25°C unless otherwise noted

Symbol	Parameter	Test Condition	Min.	Max.	Units
V_{CEO}(sus)	* Collector-Emitter Sustaining Voltage				
	: TIP32	I_C = - 30mA, I_B = 0	-40		V
	: TIP32A		-60		V
	: TIP32B		-80		V
	: TIP32C		-100		V
I_{CEO}	Collector Cut-off Current				
	: TIP32/32A	V_{CE} = - 30V, I_B = 0		- 0.3	mA
	: TIP32B/32C	V_{CE} = - 60V, I_B = 0		- 0.3	mA
I_{CES}	Collector Cut-off Current				
	: TIP32	V_{CE} = - 40V, V_{EB} = 0		- 200	µA
	: TIP32A	V_{CE} = - 60V, V_{EB} = 0		- 200	µA
	: TIP32B	V_{CE} = - 80V, V_{EB} = 0		- 200	µA
	: TIP32C	V_{CE} = - 100V, V_{CE} = 0		- 200	µA
I_{EBO}	Emitter Cut-off Current	V_{EB} = - 5V, I_C = 0		- 1	mA
h_{FE}	* DC Current Gain	V_{CE} = - 4V, I_C = - 1A	25		
		V_{CE} = - 4V, I_C = - 3A	10	50	
V_{CE}(sat)	* Collector-Emitter Saturation Voltage	I_C = - 3A, I_B = - 375mA		- 1.2	V
V_{BE}(sat)	* Base-Emitter Saturation Voltage	V_{CE} = - 4V, I_C = - 3A		- 1.8	V
f_T	Current Gain Bandwidth Product	V_{CE} = - 10V, I_C = - 500mA	3.0		MHz

* Pulse Test: PW≤300µs, Duty Cycle≤2%

Rev. A, February 2000

Typical Characteristics

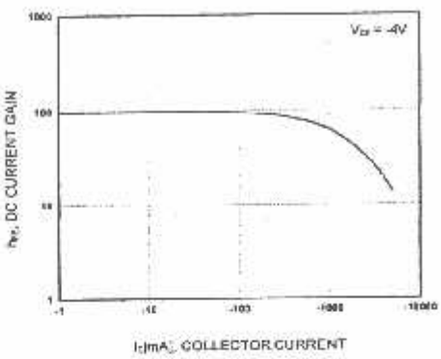


Figure 1. DC current Gain

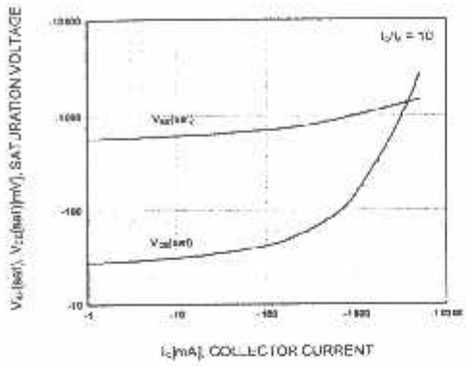


Figure 2. Base-Emitter Saturation Voltage
Collector-Emitter Saturation Voltage

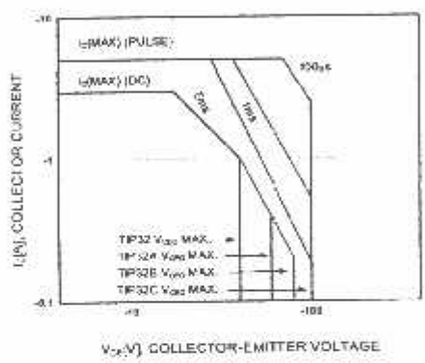


Figure 3. Safe Operating Area

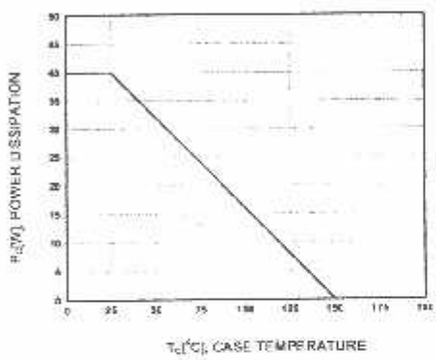
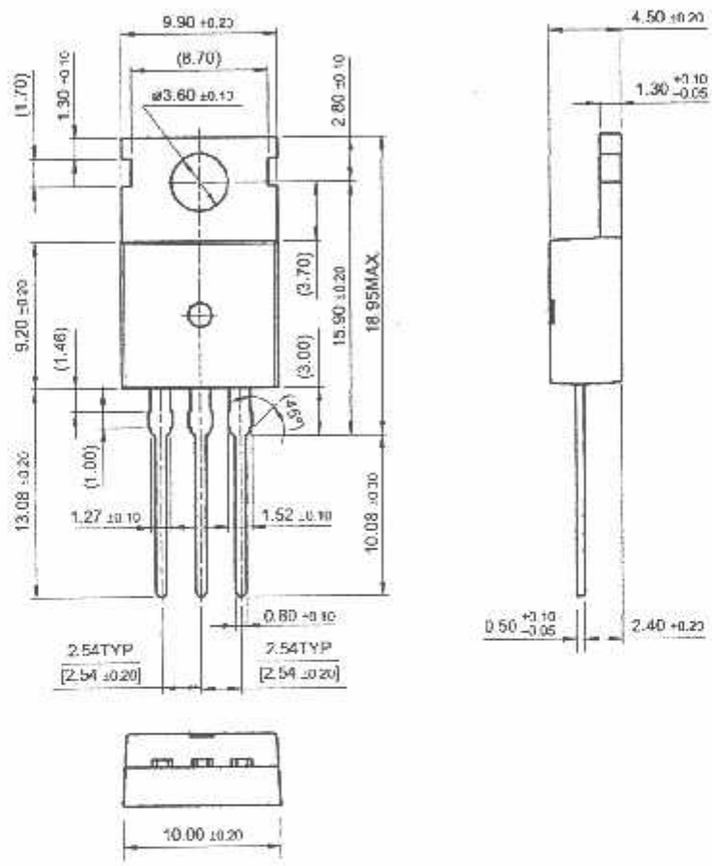


Figure 4. Power Derating

Package Demensions

TO-220



Dimensions in Millimeters

TRADEMARKS

The following are registered and unregistered trademarks Fairchild Semiconductor owns or is authorized to use and is not intended to be an exhaustive list of all such trademarks.

ACEx™
Bottomless™
CoolFET™
CROSSVOLT™
E^2CMOS™
FACT™
FACT Quiet Series™
FAST®
FASTr™
GTO™
HiSeC™
ISOPLANAR™
MICROWIRE™
POP™
PowerTrench®
QFET™
QS™
Quiet Series™
SuperSOT™-3
SuperSOT™-6
SuperSOT™-8
SyncFET™
TinyLogic™
UHC™
VCX™

DISCLAIMER

FAIRCHILD SEMICONDUCTOR RESERVES THE RIGHT TO MAKE CHANGES WITHOUT FURTHER NOTICE TO ANY PRODUCTS HEREIN TO IMPROVE RELIABILITY, FUNCTION OR DESIGN. FAIRCHILD DOES NOT ASSUME ANY LIABILITY ARISING OUT OF THE APPLICATION OR USE OF ANY PRODUCT OR CIRCUIT DESCRIBED HEREIN; NEITHER DOES IT CONVEY ANY LICENSE UNDER ITS PATENT RIGHTS, NOR THE RIGHTS OF OTHERS.

LIFE SUPPORT POLICY

FAIRCHILD'S PRODUCTS ARE NOT AUTHORIZED FOR USE AS CRITICAL COMPONENTS IN LIFE SUPPORT DEVICES OR SYSTEMS WITHOUT THE EXPRESS WRITTEN APPROVAL OF FAIRCHILD SEMICONDUCTOR INTERNATIONAL.

As used herein:

1. Life support devices or systems are devices or systems which, (a) are intended for surgical implant into the body, or (b) support or sustain life, or (c) whose failure to perform when properly used in accordance with instructions for use provided in the labeling, can be reasonably expected to result in significant injury to the user.

2. A critical component is any component of a life support device or system whose failure to perform can be reasonably expected to cause the failure of the life support device or system, or to affect its safety or effectiveness.

PRODUCT STATUS DEFINITIONS

Definition of Terms

Datasheet Identification	Product Status	Definition
Advance Information	Formative or In Design	This datasheet contains the design specifications for product development. Specifications may change in any manner without notice.
Preliminary	First Production	This datasheet contains preliminary data, and supplementary data will be published at a later date. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
No Identification Needed	Full Production	This datasheet contains final specifications. Fairchild Semiconductor reserves the right to make changes at any time without notice in order to improve design.
Obsolete	Not In Production	This datasheet contains specifications on a product that has been discontinued by Fairchild semiconductor. The datasheet is printed for reference information only.

Data Sheet

Central™ Semiconductor Corp.

145 Adams Avenue, Hauppauge, NY 11788 USA
Tel: (631) 435-1110 • Fax: (631) 435-1824

Manufacturers of World Class Discrete Semiconductors

TIP32
TIP32A
TIP32B
TIP32C

SILICON PNP POWER TRANSISTOR
3 AMPS, 40 WATTS

JEDEC TO-220 CASE

DESCRIPTION

The CENTRAL SEMICONDUCTOR TIP32 Series is a PNP Epitaxial-Base Silicon Power Transistor designed for power amplifier and high-speed switching applications.

MAXIMUM RATINGS (T_C=25°C unless otherwise noted)

	SYMBOL	TIP32	TIP32A	TIP32B	TIP32C	UNIT
Collector-Base Voltage	V_{CBO}	40	60	80	100	V
Emitter-Base Voltage	V_{EBO}	5.0	5.0	5.0	5.0	V
Collector-Emitter Voltage	V_{CEO}	40	60	80	100	V
Collector Current, Continuous	I_C	3.0	3.0	3.0	3.0	A
Collector Current, Peak	I	5.0	5.0	5.0	5.0	A
Base Current	I_B	1.0	1.0	1.0	1.0	A
Power Dissipation	P_D	40	40	40	40	W
Power Dissipation (T_A=25°C)	P_D	2.0	2.0	2.0	2.0	W
Operating and Storage Junction Temperature	T_J, T_{stg}	-65 TO +150		-65 TO +150		°C

ELECTRICAL CHARACTERISTICS (T_C=25°C unless otherwise noted)

SYMBOL	TEST CONDITIONS	MIN	MAX	UNIT
CEO	V_{CE}=30V (TIP32, TIP32A)		0.3	mA
CEO	V_{CE}=60V (TIP32B, TIP32C)		0.3	mA
CES	V_{CE}=Rated V_{CEO}		0.2	mA
EBO	V_{EB}=5.0V		1.0	mA
CEO	I_C=30mA, (TIP32)	40		V
CEO	I_C=30mA, (TIP32A)	60		V
CEO	I_C=30mA, (TIP32B)	80		V
CEO	I_C=30mA, (TIP32C)	100		V
E(SAT)	I_C=3.0A, I_B=375mA		1.2	V
E(on)	V_{CE}=4.0V, I_C=3.0A		1.8	V
E	V_{CE}=4.0V, I_C=1.0A	25		-
E	V_{CE}=4.0V, I_C=3.0A	10	50	-
e	V_{CE}=10V, I_C=0.5A, f=1 kHz	20		-
	V_{CE}=10V, I_C=0.5A, f=1 MH_z	3		MHz
n	I_C=1.0A, I_{B1}=I_{B2}=100mA, R_L=30 OHMS	0.3 TYP		uSEC
ff	I_C=1.0A, I_{B1}=I_{B2}=100mA, R_L=30 OHMS	1.0 TYP		uSEC

```
        org     00h
        ljmp    init
;
        org     23h
        clr     ES
        jnb     RI,$
        clr     RI
        mov     A,SBUF
        mov     R7,A
        setb    ES
        reti
;
        Sclk    Bit P2.0
        Sdta    Bit P2.1
        Eecn    Data 96h                ; data eeprom control
        Eeen    Equ 00001000b           ; bit eeprom enable (read)
        Eewr    Equ 00010000b           ; bit eeprom write
        Wtdg    Equ 00000010b           ; bit watchdog
        Jchr    Equ 30h                 ; jumlah character ascii
        Jcd0    Equ 31h                 ; jumlah colom dot low
        Jcd1    Equ 32h                 ; jumlah colom dot high
        Cnt0    Equ 33h
        Cnt1    Equ 34h
        Cnt2    Equ 35h
        Cnt3    Equ 36h
        DpcL    Equ 37h                 ; data pointer character low
        DpcH    Equ 38h                 ; data pointer character high
        DpdL    Equ 39h                 ; data pointer dot low
        DpdH    Equ 3Ah                 ; data pointer dot high
        Buf0    Equ 3Bh
        Buf1    Equ 3Ch
        Buf2    Equ 3Dh
        Buf3    Equ 3Eh
        Buf4    Equ 3Fh
        Buf5    Equ 40h
        Jrkn    Equ 41h
        Dly0    Equ 50h
        Dly1    Equ 51h
        Dly2    Equ 52h
;
init:   mov     P0,#000h                ; clr banner
        acall   delay2
        acall   srl_in
        acall   rstcmd
;
mulai:  mov     Jchr,#50                ; jumlah character ascii
        mov     DpcL,#00h               ; ambil dari data pointer
        mov     DpcH,#03h               ; address 2300h
        acall   cchchr                  ; cacah character
xxx:    mov     Jcd0,#124               ; jalankan terus menerus
        mov     Jcd1,#1                 ; dalam 380 colom dot matrix led
        acall   tljln1                  ; oke
        sjmp    xxx
;
loop0:  acall   tg_srl
        mov     Cnt0,#50
        mov     DPTR,#0400h
        acall   stwrmm
loop1:  movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        djnz    Cnt0,loop2
        acall   enwrmm
        sjmp    loop3
loop2:  acall   tg_srl
        sjmp    loop1
;
loop3:  mov     Jchr,#50                ; jumlah character ascii
        mov     DpcL,#00h               ; ambil dari data pointer
```

```
        mov     DpcH,#04h               ; address 2400h
        acall   cchchr                  ; cacah character
        acall   rstcmd
        mov     SP,#07h
        ljmp    xxx
;
;loop3:  mov     Jchr,#50               ; jumlah character ascii
;        mov     DpcL,#00h              ; ambil dari data pointer
;        mov     DpcH,#04h              ; address 2400h
;        acall   cchchr                 ; cacah character
;        mov     Jcd0,#124              ; jalankan terus menerus
;        mov     Jcd1,#1                        ; dalam 380 colom dot matrix led
;        acall   tljln1                 ; oke
;
        acall   clrdot                  ; sebanyak 480 colom
;
        mov     SP,#07h
        ljmp    loop0
;
clock:  setb    Sclk
        clr     Sclk
        ret
;
tljln0: mov     DpdL,#00                ; jalan & berhenti ditengah
        mov     DpdH,#00
        mov     Cnt1,#0
        mov     Cnt2,Jrkn               ; jarak dari kanan
        mov     Cnt3,#150               ; berhenti
tlj100: mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis0
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis1
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis2
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis3
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis4
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis5
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis6
```

```
        acall   enrdmm
brhnt0: mov     A,Cnt3
        jz      brhnt3
        mov     A,Cnt2
        jz      brhnt1
        dec     Cnt2
        sjmp    brhnt3
brhnt1: dec     Cnt3
        ljmp    tlj100
brhnt3: acall   gsrdot
        mov     A,Jcd1
        jz      tlj101
        djnz    Cnt1,tlj102
        djnz    Jcd1,tlj102
tlj101: mov     A,Jcd0
        jz      tlj103
        djnz    Jcd0,tlj102
        sjmp    tlj103
tlj102: ljmp    tlj100
tlj103: ret
;
tljln1: mov     DpdL,#00                ; jalan terus
        mov     DpdH,#00
        mov     Cnt1,#0
tlj110: mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis0
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis1
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis2
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis3
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis4
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis5
        acall   enrdmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     Cnt0,#80
        acall   strdmm
        lcall   tulis6
        acall   enrdmm
        acall   gsrdot
        mov     A,Jcd1
        jz      tlj111
```

```
        djnz    Cnt1,tljl12
        djnz    Jcd1,tljl12
tljl11: mov     A,Jcd0
        jz      tljl15
        djnz    Jcd0,tljl12
        sjmp    tljl15
tljl12: cjne    R7,#0FFh,tljl13
        sjmp    tljl14
tljl13: ljmp    loop0
tljl14: ljmp    tljl10
tljl15: ret
;
gsrdot: inc     DpdL
        mov     A,DpdL
        jnz     gsrdt
        inc     DpdH
gsrdt:  ret
;
tulis0: movx    A,@DPTR
        mov     C,Acc.1
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis0
        mov     P0,#00000001b
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis1: movx    A,@DPTR
        mov     C,Acc.2
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis1
        mov     P0,#00000010b
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis2: movx    A,@DPTR
        mov     C,Acc.3
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis2
        mov     P0,#00000100b
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis3: movx    A,@DPTR
        mov     C,Acc.4
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis3
        mov     P0,#00001000b
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis4: movx    A,@DPTR
        mov     C,Acc.5
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis4
        mov     P0,#00010000b
```

```
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis5: movx    A,@DPTR
        mov     C,Acc.6
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis5
        mov     P0,#00100000b
        acall   delay1
        mov     P0,#00000000b
        ret
;
tulis6: movx    A,@DPTR
        mov     C,Acc.7
        mov     Sdta,C
        lcall   clock
        inc     DPTR
        djnz    Cnt0,tulis6
        mov     P0,#01000000b
        acall   delay1
        mov     P0,#00000000b
        ret
;
cchchr: mov     DpdL,#50h
        mov     DpdH,#00h
        mov     DPL,DpcL
        mov     DPH,DpcH
cchch:  acall   strdmm
        movx    A,@DPTR
        acall   enrdmm
        acall   chrdot
        mov     DPL,DpcL
        mov     DPH,DpcH
        inc     DPTR
        mov     DpcL,DPL
        mov     DpcH,DPH
        djnz    Jchr,cchch
        ret
;
chrdot: mov     DPTR,#chrksg
        cjne    A,#'A',chdt00
        mov     DPTR,#charAb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt00: cjne    A,#'B',chdt01
        mov     DPTR,#charBb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt01: cjne    A,#'C',chdt02
        mov     DPTR,#charCb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt02: cjne    A,#'D',chdt03
        mov     DPTR,#charDb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt03: cjne    A,#'E',chdt04
        mov     DPTR,#charEb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt04: cjne    A,#'F',chdt05
```

```
        mov     DPTR,#charFb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt05: cjne    A,#'G',chdt06
        mov     DPTR,#charGb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt06: cjne    A,#'H',chdt07
        mov     DPTR,#charHb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt07: cjne    A,#'I',chdt08
        mov     DPTR,#charIb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt08: cjne    A,#'J',chdt09
        mov     DPTR,#charJb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt09: cjne    A,#'K',chdt10
        mov     DPTR,#charKb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt10: cjne    A,#'L',chdt11
        mov     DPTR,#charLb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt11: cjne    A,#'M',chdt12
        mov     DPTR,#charMb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt12: cjne    A,#'N',chdt13
        mov     DPTR,#charNb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt13: cjne    A,#'O',chdt14
        mov     DPTR,#charOb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt14: cjne    A,#'P',chdt15
        mov     DPTR,#charPb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt15: cjne    A,#'Q',chdt16
        mov     DPTR,#charQb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt16: cjne    A,#'R',chdt17
        mov     DPTR,#charRb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt17: cjne    A,#'S',chdt18
        mov     DPTR,#charSb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
```

```
chdt18: cjne    A,#'T',chdt19
        mov     DPTR,#charTb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt19: cjne    A,#'U',chdt20
        mov     DPTR,#charUb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt20: cjne    A,#'V',chdt21
        mov     DPTR,#charVb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt21: cjne    A,#'W',chdt22
        mov     DPTR,#charWb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt22: cjne    A,#'X',chdt23
        mov     DPTR,#charXb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt23: cjne    A,#'Y',chdt24
        mov     DPTR,#charYb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt24: cjne    A,#'Z',chdt25
        mov     DPTR,#charZb
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
;
chdt25: cjne    A,#'a',chdt26
        mov     DPTR,#charak
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt26: cjne    A,#'b',chdt27
        mov     DPTR,#charbk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt27: cjne    A,#'c',chdt28
        mov     DPTR,#charck
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt28: cjne    A,#'d',chdt29
        mov     DPTR,#chardk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt29: cjne    A,#'e',chdt30
        mov     DPTR,#charek
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt30: cjne    A,#'f',chdt31
        mov     DPTR,#charfk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt31: cjne    A,#'g',chdt32
        mov     DPTR,#chargk
        lcall   ambdta
```

```
        lcall   tlsmem
        ljmp    chdt99
chdt32: cjne    A,#'h',chdt33
        mov     DPTR,#charhk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt33: cjne    A,#'i',chdt34
        mov     DPTR,#charik
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt34: cjne    A,#'j',chdt35
        mov     DPTR,#charjk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt35: cjne    A,#'k',chdt36
        mov     DPTR,#charkk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt36: cjne    A,#'l',chdt37
        mov     DPTR,#charlk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt37: cjne    A,#'m',chdt38
        mov     DPTR,#charmk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt38: cjne    A,#'n',chdt39
        mov     DPTR,#charnk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt39: cjne    A,#'o',chdt40
        mov     DPTR,#charok
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt40: cjne    A,#'p',chdt41
        mov     DPTR,#charpk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt41: cjne    A,#'q',chdt42
        mov     DPTR,#charqk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt42: cjne    A,#'r',chdt43
        mov     DPTR,#charrk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt43: cjne    A,#'s',chdt44
        mov     DPTR,#charsk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt44: cjne    A,#'t',chdt45
        mov     DPTR,#chartk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt45: cjne    A,#'u',chdt46
        mov     DPTR,#charuk
```

```
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt46: cjne    A,#'v',chdt47
        mov     DPTR,#charvk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt47: cjne    A,#'w',chdt48
        mov     DPTR,#charwk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt48: cjne    A,#'x',chdt49
        mov     DPTR,#charxk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt49: cjne    A,#'y',chdt50
        mov     DPTR,#charyk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt50: cjne    A,#'z',chdt51
        mov     DPTR,#charzk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99

chdt51: cjne    A,#'0',chdt52
        mov     DPTR,#charn0
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt52: cjne    A,#'1',chdt53
        mov     DPTR,#charn1
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt53: cjne    A,#'2',chdt54
        mov     DPTR,#charn2
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt54: cjne    A,#'3',chdt55
        mov     DPTR,#charn3
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt55: cjne    A,#'4',chdt56
        mov     DPTR,#charn4
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt56: cjne    A,#'5',chdt57
        mov     DPTR,#charn5
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt57: cjne    A,#'6',chdt58
        mov     DPTR,#charn6
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt58: cjne    A,#'7',chdt59
        mov     DPTR,#charn7
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
```

```
chdt59: cjne    A,#'8',chdt60
        mov     DPTR,#charn8
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt60: cjne    A,#'9',chdt61
        mov     DPTR,#charn9
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99

chdt61: cjne    A,#':',chdt62
        mov     DPTR,#chrtt2
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt62: cjne    A,#'/',chdt63
        mov     DPTR,#chrgrm
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt63: cjne    A,#'.',chdt64
        mov     DPTR,#chrttk
        lcall   ambdta
        lcall   tlsmem
        ljmp    chdt99
chdt64: cjne    A,#' ',chdt65
        mov     DPTR,#chrsps
        lcall   ambdta
        lcall   tlsmem
chdt65: cjne    A,#',',chdt66
        mov     DPTR,#chrkom
        lcall   ambdta
        lcall   tlsmem
chdt66: cjne    A,#';',chdt67
        mov     DPTR,#chrtkm
        lcall   ambdta
        lcall   tlsmem
chdt67: cjne    A,#'[',chdt68
        mov     DPTR,#chrbkr
        lcall   ambdta
        lcall   tlsmem
chdt68: cjne    A,#']',chdt69
        mov     DPTR,#chrtpr
        lcall   ambdta
        lcall   tlsmem
chdt69: cjne    A,#'=',chdt70
        mov     DPTR,#chrsdg
        lcall   ambdta
        lcall   tlsmem
chdt70: cjne    A,#'-',chdt99
        mov     DPTR,#chrstr
        lcall   ambdta
        lcall   tlsmem
chdt99: ret
;
ambdta: clr     A
        movc    A,@A+DPTR
        mov     Buf0,A
        inc     DPTR
        clr     A
        movc    A,@A+DPTR
        mov     Buf1,A
        inc     DPTR
        clr     A
        movc    A,@A+DPTR
        mov     Buf2,A
        inc     DPTR
        clr     A
```

```
        movc    A,@A+DPTR
        mov     Buf3,A
        inc     DPTR
        clr     A
        movc    A,@A+DPTR
        mov     Buf4,A
        inc     DPTR
        clr     A
        movc    A,@A+DPTR
        mov     Buf5,A
        ret
;
tlsmem: acall   stwrmm
        mov     DPL,DpdL
        mov     DPH,DpdH
        mov     A,Buf0
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     A,Buf1
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     A,Buf2
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     A,Buf3
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     A,Buf4
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     A,Buf5
        movx    @DPTR,A
        acall   wt_wr
        inc     DPTR
        mov     DpdL,DPL
        mov     DpdH,DPH
        acall   enwrmm
        ret
;
clrdot: mov     DPTR,#0000              ; clear colom dot matrix
        mov     Cnt0,#0
        mov     Cnt1,#1
        mov     Cnt2,#124
        acall   stwrmm
cldot0: mov     A,#0
        movx    @DPTR,A
        lcall   wt_wr
        inc     DPTR
        djnz    Cnt0,cldot0
        djnz    Cnt1,cldot0
cldot1: mov     A,#0
        movx    @DPTR,A
        lcall   wt_wr
        inc     DPTR
        djnz    Cnt2,cldot1
        acall   enwrmm
        ret
;
stwrmm: orl     Eecn,#Eeen
        orl     Eecn,#Eewr
        ret
;
enwrmm: xrl     Eecn,#Eewr
        xrl     Eecn,#Eeen
```

```
        ret
;
strdmm: orl     Eecn,#Eeen
        ret
;
enrdmm: xrl     Eecn,#Eeen
        ret
;
wt_wr:  mov     A,Eecn
        anl     A,#Wtdg
        jz      wt_wr
        ret
;
srl_in: acall   delay1
        mov     TMOD,#20h
        mov     TH1,#0FDh
        mov     SCON,#50h
        setb    TR1
        setb    ES
        setb    EA
        ret
;
tg_srl: mov     R7,#0FFh
tgsrl0: cjne    R7,#0FFh,tgsrl1
        sjmp    tgsrl0
tgsrl1: ret
;
rstcmd: mov     R7,#0FFh
        ret
;
delay0: djnz    Dly0,delay0
        ret
;
delay1: mov     Dly1,#4
dely10: acall   delay0
        djnz    Dly1,dely10
        ret
;
delay2: mov     Dly2,#5
dely20: acall   delay0
        djnz    Dly1,dely20
        djnz    Dly2,dely20
        ret
;
charAb: DB      07Eh,090h,090h,090h,07Eh,000h
charBb: DB      0FEh,092h,092h,092h,06Ch,000h
charCb: DB      07Ch,082h,082h,082h,044h,000h
charDb: DB      0FEh,082h,082h,044h,038h,000h
charEb: DB      0FEh,092h,092h,092h,082h,000h
charFb: DB      0FEh,090h,090h,090h,080h,000h
charGb: DB      07Ch,082h,092h,092h,05Ch,000h
charHb: DB      0FEh,010h,010h,010h,0FEh,000h
charIb: DB      000h,082h,0FEh,082h,000h,000h
charJb: DB      004h,002h,082h,0FCh,080h,000h
charKb: DB      0FEh,010h,028h,044h,082h,000h
charLb: DB      0FEh,002h,002h,002h,002h,000h
charMb: DB      0FEh,040h,030h,040h,0FEh,000h
charNb: DB      0FEh,020h,010h,008h,0FEh,000h
charOb: DB      07Ch,082h,082h,082h,07Ch,000h
charPb: DB      0FEh,090h,090h,090h,060h,000h
charQb: DB      07Ch,082h,08Ah,084h,07Ah,000h
charRb: DB      0FEh,090h,098h,094h,062h,000h
charSb: DB      064h,092h,092h,092h,04Ch,000h
charTb: DB      080h,080h,0FEh,080h,080h,000h
charUb: DB      0FCh,002h,002h,002h,0FCh,000h
charVb: DB      0F8h,004h,002h,004h,0F8h,000h
charWb: DB      0FCh,002h,01Ch,002h,0FCh,000h
charXb: DB      0C6h,028h,010h,028h,0C6h,000h
charYb: DB      0E0h,010h,00Eh,010h,0E0h,000h
```

```
charzb: DB      086h,08Ah,092h,0A2h,0C2h,000h
charak: DB      004h,02Ah,02Ah,02Ah,01Eh,000h
charbk: DB      0FEh,012h,022h,022h,01Ch,000h
charck: DB      01Ch,022h,022h,022h,004h,000h
chardk: DB      01Ch,022h,022h,012h,0FEh,000h
charek: DB      01Ch,02Ah,02Ah,02Ah,018h,000h
charfk: DB      010h,07Eh,090h,080h,040h,000h
chargk: DB      010h,02Ah,02Ah,02Ah,03Ch,000h
charhk: DB      0FEh,010h,020h,020h,01Eh,000h
charik: DB      000h,022h,0BEh,002h,000h,000h
charjk: DB      004h,002h,022h,0BCh,000h,000h
charkk: DB      0FEh,008h,014h,022h,000h,000h
charlk: DB      000h,082h,0FEh,002h,000h,000h
charmk: DB      03Eh,020h,018h,020h,01Eh,000h
charnk: DB      03Eh,010h,020h,020h,01Eh,000h
charok: DB      01Ch,022h,022h,022h,01Ch,000h
charpk: DB      03Eh,028h,028h,028h,010h,000h
charqk: DB      010h,028h,028h,018h,03Eh,000h
charrk: DB      03Eh,010h,020h,020h,010h,000h
charsk: DB      012h,02Ah,02Ah,02Ah,024h,000h
chartk: DB      020h,0FCh,022h,002h,004h,000h
charuk: DB      03Ch,002h,002h,004h,03Eh,000h
charvk: DB      038h,004h,002h,004h,038h,000h
charwk: DB      03Ch,002h,004h,002h,03Ch,000h
charxk: DB      022h,014h,008h,014h,022h,000h
charyk: DB      030h,00Ah,00Ah,00Ah,03Ch,000h
charzk: DB      022h,026h,02Ah,032h,022h,000h

charn0: DB      07Ch,082h,082h,082h,07Ch,000h
charn1: DB      000h,042h,0FEh,002h,000h,000h
charn2: DB      042h,086h,08Ah,092h,062h,000h
charn3: DB      084h,082h,0A2h,0D2h,08Ch,000h
charn4: DB      018h,028h,048h,0FEh,008h,000h
charn5: DB      0E4h,0A2h,0A2h,0A2h,09Ch,000h
charn6: DB      03Ch,052h,092h,092h,00Ch,000h
charn7: DB      080h,080h,09Eh,0A0h,0C0h,000h
charn8: DB      06Ch,092h,092h,092h,06Ch,000h
charn9: DB      060h,092h,092h,094h,078h,000h

chrtt2: DB      000h,000h,024h,000h,000h,000h           ; titik dua
chrgrm: DB      004h,008h,010h,020h,040h,000h           ; garis miring
chrttk: DB      000h,000h,002h,000h,000h,000h           ; titik
chrkom: DB      000h,00Ah,00Ch,000h,000h,000h           ; koma
chrtkm: DB      000h,06Ah,06Ch,000h,000h,000h           ; titik koma
chrbkr: DB      000h,0FEh,082h,082h,000h,000h           ; buka kurung
chrtpr: DB      000h,082h,082h,0FEh,000h,000h           ; tutup kurung
chrsdg: DB      028h,028h,028h,028h,028h,000h           ; sama dengan
chrstr: DB      010h,010h,010h,010h,010h,000h           ; strip
chrsps: DB      000h,000h,000h,000h,000h,000h           ; spasi
chrksg: DB      000h,000h,000h,000h,000h,000h           ; kosong
;
        org     2000h
;
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
;
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
```

```
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
;
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
        DB      000h,000h,000h,000h,000h,000h,000h,000h,000h,000h
;
        org     2300h
;
        DB      'VICKY   NIM: 0317070   Institut Teknologi Nasional'
;
        end
```