LAPORAN TUGAS AKHIR

PERENCANAAN DAN PEMBUATAN ALAT PENGENDALI LAMPU RUMAH MELALUI SMS BERBASIS MIKROKONTROLER AT89S8252

Disusun oleh :

Nama : DAVID ARIYANTO

NIM : 0552007

Konsentrasi : T.energi listrik

JURUSAN TEKNIK ELEKTRO DIII

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL

MALANG

2009

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

[illegible]

PERENCANAAN DAN PEMBUATAN ALAT PENGENDALI LAMPU RUMAH MELALUI SMS BERBASIS MIKROKONTROLER AT89S8252

TUGAS AKHIR

Disusun Oleh :

DAVID ARIYANTO

05.52.007

KONSENTRASI TEKNIK ENERGI LISTRIK D III

JURUSAN TEKNIK ELEKTRO D III

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2009

LEMBAR PERSETUJUAN

PERENCANAAN DAN PEMBUATAN ALAT PENGENDALI LAMPU RUMAH MELALUI SMS BERBASIS MIKROKONTROLER AT89S8252

TUGAS AKHIR

Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Diploma III Teknik Elektronika

Disusun Oleh :

DAVID ARIYANTO
NIM : 0552007

Diperiksa dan Disetujui

Ketua Jurusan Teknik Elektro DIII	Dosen Pembimbing
Ir. H. Taufik Hidayat, MT	**Bambang Prio H, ST. MT**
NIP. Y 1018700151	**NIP. Y 1028400082**

KONSENTRASI TEKNIK ENERGI LISTRIK DIII
JURUSAN TEKNIK ELEKTRO D III
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2009

LEMBAR PENGESAHAN

PERENCANAAN DAN PEMBUATAN ALAT PENGENDALI LAMPU RUMAH MELALUI SMS BERBASIS MIKROKONTROLER AT89S8252

TUGAS AKHIR

Disusun Oleh :

DAVID ARIYANTO
NIM : 0552007

Telah Dipertahankan Dihadapan Majelis Penguji Tugas Akhir
Jenjang Program Diploma Tiga (DIII)

Penguji I	Penguji II
Ir. H. Taufik Hidayat, MT	Ir. M. Abdul Hamid, MT
NIP. Y 101 8700 151	NIP. Y 101 8800 188

KONSENTRASI TEKNIK ELEKTRONIKA DIII
JURUSAN TEKNIK ELEKTRO DIII
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2009

BERITA ACARA UJIAN TUGAS AKHIR
FAKULTAS TEKNOLOGI INDUSTRI

NAMA	**: DAVID ARIYANTO**
NIM	**: 0552007**
JURUSAN	**: TEKNIK ELEKTRO D-III**
JUDUL SKRIPSI	**: PERENCANAAN DAN PEMBUATAN ALAT PENGENDALI LAMPU RUMAH MELALUI SMS BERBASIS MIKROKONTROLER AT89S8252**

Dipertahankan di hadapan Tim Penguji Tugas Akhir
Jenjang Program Diploma Tiga (DIII), pada:

Hari / Tanggal : Jum'at / 02 Oktober 2009
Dengan Nilai : 85 (A)

Panitia Ujian Tugas Akhir

Ketua

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

Sekertaris

Ir. H. Taufik Hidayat,MT
NIP.Y 101 8700 151

Anggota Penguji

Penguji I

Ir. H. Taufik Hidayat, MT
NIP. Y 101 8700 151

Penguji II

Ir. M. Abdul Hamid, MT
NIP. Y 101 8800 188

ABSTRAK

Perencanaan dan pembuatan alat pengendali lampu rumah melalui sms berbasis mikrokontroler AT89S8252

(David Ariyanto,0552007)
(Dosen Pembimbing :Bambang Prio Hartono,ST,MT)

Kita sering menjumpai orang yang pergi meninggalkan rumah dan ketika malam tiba penghuni rumah belum sampai. Seiring berkembangnya teknologi, lampu rumah bisa dinyalakan atau dimatikan dari jarak jauh melalui handphone, yaitu dengan menggunakan salah satu fasilitas yang terdapat pada handphone itu sendiri, dan tujuan dari pembuatan tugas akhir ini adalah merencanakan dan membuat alat pengendali lampu rumah melalui sms berbasis mikrokontroler AT89S8252.

Untuk menyelesaikan pembuatan laporan ini serta untuk mempermudah pembahasan maka penulis menjalankan beberapa metode diantaranya Studi *Literrature*, Perencanaan dan pembuatan alat dan pengujian alat, hal ini dilakukan untuk mengetahui apakah alat yang dibuat telah sesuai dengan yang direncanakan..

Pada driver lampu apabila kondisi lampu mati maka berlogika high("1")dengan keluaran tegangan 0V,dan apabila lampu dalam keadaan menyala maka logika("0")dengan keluaran 220 V. Input tegangan sebesar 4.93V ke mikrokontroler level tegangan telah memenuhi standar logika TTL dan dari hasil percobaan didapat rata-rata waktu pengiriman sampai lampu menyala adalah 09,38 detik

Keyword : Handpone, AT89S8252, Sensor, Driver Lampu

KATA PENGANTAR

Dengan mengucap rasa puji syukur kehadirat allah SWT berkat rahmat serta hidayah-NYA yang berlimpah sehingga penyusun dapat menyelesaikan laporan tugas akhir ini yang berjudul Perencanaan dan pembuatan alat pengendali lampu rumah melalui sms berbasis mikrokontroler AT89S8252. Pada kesempatan ini penyusun mengucapkan terimakasih pada:

1. Bapak Abraham lomi,ir.MSEE,DR selaku rector Institute Teknologi Nasional Malang
2. Bapak Ir.Taufik Hidayat,MT selaku ketua jurusan teknik elektro D-III Institut Teknologi Nasional Malang
3. Bapak Bambang Prio Hartono,ST,MT selaku dosen pembimbing Tugas Akhir
4. Kepada semua pihak yang telah membantu penyelesaian Tugas Akhir ini

Penyusun menyadari dalam penyelesaian laporan ini belum mencapai kesempurnaan yang sesungguhnya.Oleh karenanya saran dan kritik membangun diharapkan guna penyempurnaan laporanaan laporan ini.

Malang, Maret 2009

Penyususun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 Latar Belakang

Sesuai dengan perkembangan zaman dan teknologi, banyak ditemukan peralatan baru yang canggih dan serba otomatis. *Handphone* yang merupakan alat komunikasi juga dapat digunakan sebagai pengendali alat-alat elektronik maupun sebagai pemantau. Misalnya sebagai pengendali lampu sehingga dapat mengetahui kondisi lampu meskipun tidak berada ditempat apakah itu keadaan padam atau hidup. Pada waktu seseorang bepergian jauh dan lebih dari satu atau dua hari,mungkin orang tersebut akan khawatir dengan keadaan rumahnya karena lampu masih padam rumah terlihat gelap dan mungkin juga terlihat mistis(seram).

Selain itu alat ini juga dapat memberikan laporan melalui sms jika ada yang menghidupkan lampu berarti bisa diambil kesimpulan ada orang masuk padahal pintu terkunci.disamping itu jika lampu dikendalikan hidup atau padam akan memberikan laporan bahwa lampu hidup atau padam,jadi kita benar-benar yakin lampu sudah sesuai kendali.Menurut(yudi,2009)

Harapan dari terbuatnya alat ini agar suatu saat nanti dalam mengendalikan apapun tidak tidak hanya dengan sistem manual melainkan secara otomatis dan dapat dikendalikan dimanapun dan kapanpun.

Kenyataan kebanyakan orang mungkin beranggapan hal ini terlalu rumit dan ribet, dikarenakan alat ini tidak begitu sering difungsikan bahkan tidak difungsikan untuk setiap hari.

1.2 Rumusan masalah

Dari latar belakang diatas maka permasalahan dapat dirumuskan sebagai berikut:

1. Bagaimana merancang alat pengendali lampu yang memberikan laporan melalui SMS berbasis mikrokontroler AT89S8252.
2. Bagaimana menggunakan *handphone* sebagai pengendali.
3. Bagaimana membuat *driver handphone* lampu

1.3 Tujuan

Tujuan dari pembuatan laporan tugas akhir ini adalah merencanakan dan membuat alat pengendali lampu rumah melalui sms berbasis mikrokontroler AT89S8252.

1.4 Batasan masalah

Agar permasalahan tidak meluas,maka tugas akhir ini dibatasi hanya pada hal-hal berikut :

1. Jenis *handphone* yang digunakan adalah siemens M35 dan alat bekerja saat kondisi dimana telpon seluler dapat berfungsi.
2. Tidak membahas tentang sistem internal dari teipon seluler.
3. Tidak membahas tentang operator telepon seluler, jaringan telepon yang digunakan dan mekanisme pembayaran jasa operator telephone seluler.
4. Bentuk dari mekanik yang dibuat sebatas miniatur sederhana.
5. Tidak membahas sumber dari catu daya yang digunakan alat.

1.5 Metode Penulisan

Untuk menyelesaikan laporan ini serta untuk mempermudah pembahasan maka penulis menjalankan beberapa metode diantaranya :

a. Studi Literatur

Metode ini digunakan untuk mempelajari, mengkaji, dan mengumpulkan data-data yang berkaitan dengan perancangan dan pembuatan alat pada tugas akhir.

b. Perencanaan Dan Pembuatan Alat

Pada tahap ini, akan dibuat program dan alat pendukung berdasarkan rencana awal dengan berlandaskan pada pedoman yang ada.

c. Pengujian Alat

Hal ini dilakukan untuk mengetahui apakah alat yang dibuat telah sesuai dengan yang direncanakan.

d. Kesimpulan Dan Saran

1.6 Sistematika Penulisan

Adapun pembagian secara lengkap di uraikan sebagai berikut :

BAB I : PENDAHULUAN

Merupakan bab awal dari pembahasan yang terdiri dari latar belakang Rumusan Masalah, Tujuan, Batasan Masalah, Metodelogi penulisan dan sistematika penulisan.

BAB II : TEORI DASAR

Dalam bab ini membahas tentang teori penunjang dari pembahasan masalah antara lain membahas tentang komponen-komponen yang di gunakan dalam perencanaan alat ini.

BAB III : PERENCANAAN ALAT

Membahas tentang prinsip kerja alat, langkah-langkah perancangan, gambar diagram blok, hasil rancangan rangkaian, spesifikasi alat dan cara mengoprasikan alat.

BAB IV : ANALISA

Pada bab ini diisi oleh hasil pengujian alat dari setiap blok maupun secara keseluruhan dari alat tersebut.

BAB V : PENUTUP

Merupakan kesimpulan yang di ambil setelah menyelesaikan perencanaan dan memberikan saran-saran untuk penggunaan alat.

BAB II

DASAR TEORI

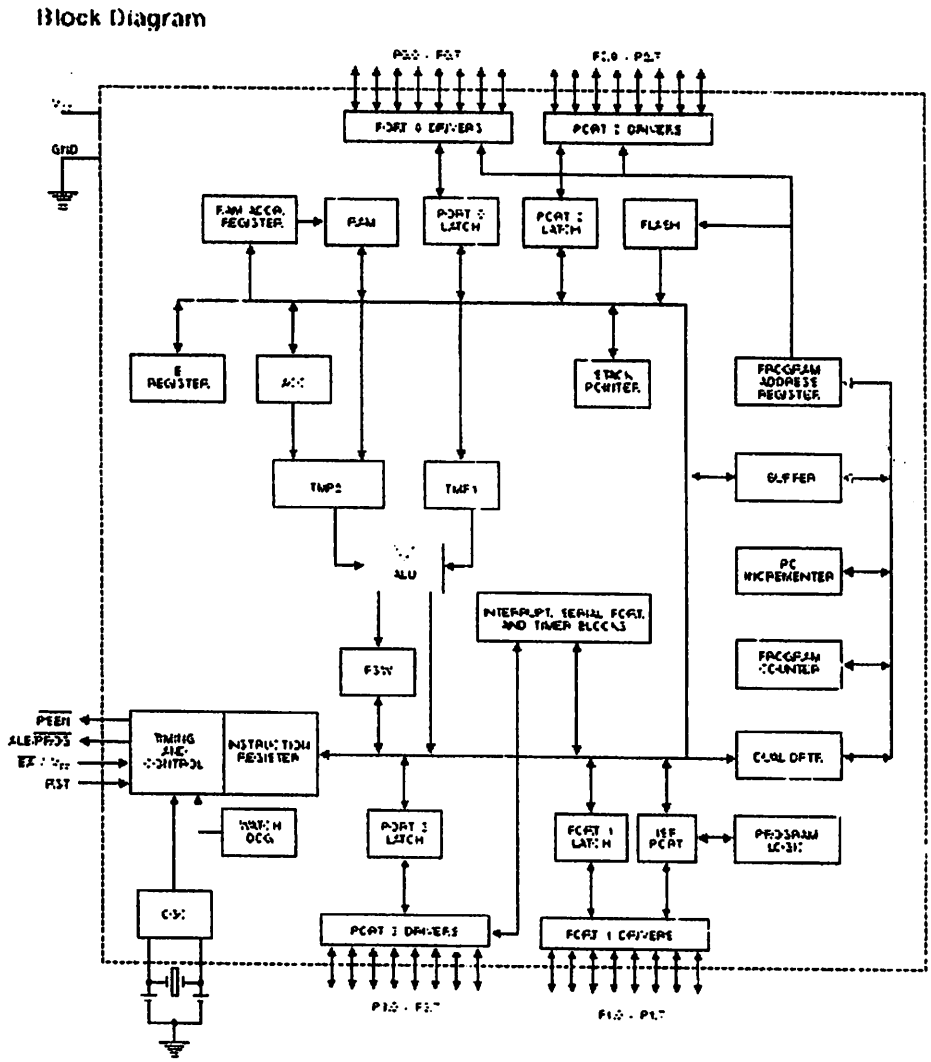
2.1. Mikrokontroler AT89S8252

Dari (.....) menerangkan Mikrokontroler AT89S8252 adalah anggota keluarga MCS-51. Port serial pada AT89S8252 bersifat dupleks-penuh atau *fullduplex*,artinya port serial bisa menerima dan mengirim secara bersama.Selain itu juga memiliki penyangga penerima, artinya port serial mulai bisa menerima *byte* yang kedua sebelum *byte* pertama dibaca oleh register penerima (jika sampai *byte* yang kedua selesai diterima sedangkan *byte* pertama belum belum juga dibaca, maka salah satu byte akan hilang).

Penerimaan dan pengiriman data port serial melalui register SBUF.Penulisan ke SBUF berarti mengisi register pengiraman SBUF sedangkan pembacaan dari SBUF berarti membaca register penerimaan port serial pada AT89S8252 bisa digunakan dalam 4 mode kerja yang berbeda. Dari 4 mode tersebut,1 mode diantaranya bekerja secara sinkron dan 3 lainya bekerja secara asinkron Keempat mode kerja tersebut adalah:

- Mode 0 : Mode ini bekerja secara sinkron,data serial dikirim dan diterima melalui kaki P3.0 (RxD),Sedangkan kaki P3.1 (TxD) dipakai untuk menyalurkan detak pendorong data serial yang dibangkitkan AT89S8252.Data dikirim/diterima 8 bit sekaligus,dimulai dari bit yang bobotnya paling kecil atau LSB (bit 0)dan diakhiri dengan bit yang bobotnya paling besar atau MSB(bit7).Kecepatan pengiriman data (*baudrate*) adalah 1/12 frekuensi kristal yang digunakan.

- *Special Function Register* (Register Fungsi Khusus),memori yang berisi register-register yang mempunyai fungsi-fungsi khusus yang disediakan oleh mikrokontroller tersebut,sebagai timer, serial dan lain-lain.
- Flash PEROM,memori yang digunakan untuk menyimpan intruksi-intruksi MCS-5!



Gambar2.1 Blok diagram AT89S8252

Secara umum keistimewaan yang dimiliki oleh mikrokontroler AT89S8252 adalah sebagai berikut:

- Sebuah CPU 8 bit yang termasuk dalam keluarga MCS-51.

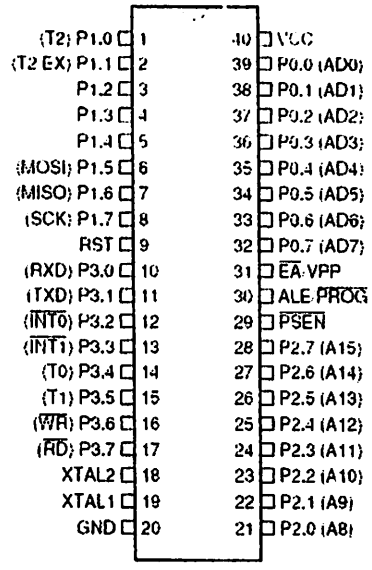
- Mode 1 : Pada mode ini tetap yaitu, data dikirim dan diterima melalui kaki P3.0(RxD), secara asinkron (juga mode 2 dan 3).pada Mode 1 data dikirim atau diterima 10 bit sekaligus,diawali dengan 1 bit start,disusul dengan 8 bit data yang dimulai dari bit yang bobotnya paling kecil (bit 0),diakhiri dengan 1 bit stop.pada AT89S8252 yang berfungsi sebagai penerima bit stop adalah RB8 dalam register SCON.Kecepatan pengiriman data (*baud rate*)bisa diatur sesuai dengan keperluan.Mode inilah (mode 2 dan 3) yang umum dikenal sebagai UAT (*Universal Asynchronous Transmiter*).
- Mode 2 : Data dikirim/diterima 11 bit sekaligus,diawali dengan 1 bit start, disusul 8 bit data yang dimulai dari bit yang bobotnynya paling kecil(bit 0), kemudian bit ke 9 yang bisa diatur lebih lanjut, diakhiri dengan 1 bit stop.Pada AT89S8252 yang berfungsi sebagai penerima, bit 9 ditampung pada bit RB* dalam register SCON, sedangkan bit stop diabaikan tidak ditampung.Kecepatan pengiriman data (*baud rate*) bisa dipilih antara 1/32 atau 1/64 frekuensi kristal yang digunakan.
- Mode 3 : mode ini sama dengan mode 2, hanya kecepatan pengiriman data (*baud rate*) bisa diatur sesuai dengan keperluan,seperti halnya mode 1. Pada mode asinkron (mode 1,mode 2 dan mode 3),port AT89S8252 bekerja secara *fullduplex.*

Struktur memori yang dimiliki oleh AT89S8252 adalah sebagai berikut:

- RAM internal,memori sebesar 256 byte yang biasanya digunakan untuk menyimpan variable atau data yang bersifat sementara.

- 8 Kbyte *Reprammable Flash Memory.*
- Kapasitas RAM internal sebesar 256 byte.
- Empat buah *programmable* port I/O,masing-masing terdiri atas 8 buah jalur I/O.
- Tiga buah pencacah (*counter*) atau pewaktu (timer) 16-bit.
- Kecepatan operasi hingga 33 MHz.
- Delapan buah jalur sLaan (*interrupt*).
- Ebuah port serial *full duplex.*

Berikut ini adalah bentuk fisik dan konfigurasi dari kaki-kaki pada MCU AT89S8252:



Gambar 2.2 Bentuk Fisik AT89S8252

Fungsi dari tiap-tiap kaki adalah sebagai berikut:

1. GND

   Dihubungkan dengan ground rangkaian.

2. VCC

   Dihubungkan dengan sumber tegangan +5 Volt.

3. Port 0(P0.0-P0.7)

Port merupakan I/O 8 bit dua arah.Port ini dihubungkan sebagai *multiplexks* bus alamat rendah dan dua bus data selama pengakseskan ke *memory* luar.

4. Port 1(p1.0-p1.7)

Port 1 dapat berfungsi sebagai input ataupun output dan bekerja untuk operasi bit maupun *byte*,tergantung dari pengaturan *Software*.

5. Port 2(P2.0-P2.7)

Port dapat digunakan sebagai bus byte tinggi selama adanya akses ke memory program luar atau memory data luar.

6. Port 3(P3.0-P3.7)

Port ni selain memiliki fungsi I/O juga mempunyai fungsi khusus sebagai berikut:

- RD (P3.7) Sinyal pembaca data luar.
- WR (P3.6) Sinyal penulisan data luar.
- T1 (P3.5) Masukan dari pewaktu atau pencacah 1.
- T0 (P3.4) Masukan dari pewaktu pencacah 0.
- INT I (P3.3) Masukan interrupt 1.
- INT 0 (P3.2) Masukan interrupt 0.
- TXD (P3.1) Keluaran pengiriman data untuk serial port.
- *(Asynchronous)* atau sebagai keluaran *clok (Synchronous)*
- RXD (P3.0) Masukan penerimaan data serial (synchronous)

7. RST /VPD

Merupan pin input aktif tinggi,jka pin ini aktif tinggi selama 2 siklus mesin ketika oscillator bekerja akan merest peralatan

8. ALE(*Adress Lacth Enable*)

Pin ALE (aktif tinggi)mengeluarkan pulsa output untuk melacth suatu byte alamat rendah selama mengakses kememory eksternal.ALE dapat mengendalikan TTL.Pin ini juga merupakan input pulsa program yang aktif rendah selama pemograman EPROM.Pada operasi normal,ALE dikeluarkan pada satu kecepatan yang konstan yaitu 1/6 dari frekwensi oscillator, dan digunakan suatu timing eksternal atau untuk tujuan pengeklokan.

9. PSEN (*Program Strobe Enable*)

Pin ini aktif rendah yang merupakan Strobe pembacaan ke program *memory Eksternal.*

10. XTAL 1 dan XTAL 2

Pin XTAL 1 merupakan pin input ke penguat oscillator pembalik dan pin XTAL 2 merupakan pin output dari penguat oscillator pembalik.

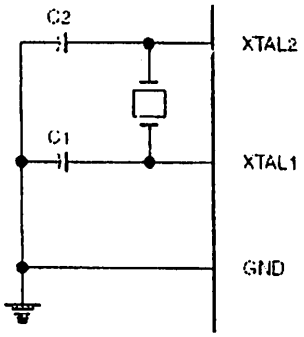
11. EA/VPP (*External Access/programming supply Voltage*)

Pin EA harus dihold rendah secara external atau dihubungkan ke VCC agar 8952 dapat mengakses kode mesin dari program memori eksternal dengan lokasi $0000 sampai $ OFFF.

Mikrokontroler AT89S8252 memiliki osilator internal (*on chip oscillator*) yang dapat digunakan sebagai sumber pewaktuan (*clock*) bagi COU.Untuk menggunakan oscilator internal diperlukan sebuah kristal atau resonator keramik

antara pin XTAL 1 dan pin XTAL 2 dan sebuah kapasitor ke *ground*.Untuk kristalnya dapat digunakan kristal dengan frekuensi sampai dengan 24 MHz.

Gambar berikut menunjukan rangkaian osilator yang digunakan.



Gambar 2.3 rangkaian Osilator

2.2 Telepon Seluler (*Handphone*)

Handphone merupakan suatu jenis perangkat telepon bergerak data komunikasi, sehingga memudahkan seseorang berkomunikasi dimanapun dan kapanpun selama dalam cangkupan suatu jaringan dari *SIM Card* yang digunakan. *Handphone* (HP) merupakan suatu perangkat yang dalam mengirim dan menerima data suara.Seiring dengan perkembangan teknologi dibidang *mobaile system,* HP tidak hanya dapat mengirim atau menerima data suara tetapi juga data karakter atau disebut dengan SMS (*Short Message Service*). Dengan kecanggihan HP pada zaman sekarang maka HP juga dilengkapi dengan fasilitas lainya, seperti EMS (*Enhaced Message Service*),MMS (*multimedia message service*) dan *polyphonic.*

2.3 Data Kabel HP Siemens M35

Siemens M35 dilengkapi dengan kabel data untuk melakukan komunikasi data serial dengan terminal lain.Kecepatan transfer data (baud rate) sebesar 19200. Untuk mengadakan komunikasi serial, pin-pin yang digunakan adalah:

Tabel 2.1 kabel data Siemens M35

Nomor Pin	**Nama**	**Fungsi**
1	GND	Data *Ground*
5	TEMS/DFMS – *Terminal Adaptor Equipment*	Serial Data Out (TX)
6	*From mobile station/data from mobile station* TIMS/DTMS – *Terminal Adaptor Equipment to Mobile Station/Data to mobile Station*	Serial Dta In (RX)

Tabel 2.2.*Pin Out* Handphone Siemens M35

Pin (1)	**Nama** (2)	**Fungsi** (3)	**I/Out** (4)
1	GND	*Ground*	
2	Self Service	*Recognition/control battery charger*	In/out
3	Load	*Charging Voltage*	In
4	Battrey	*Battery(Vcc)*	Out
5	TX	*Data Sent*	Out
6	RX	*Data Received*	In
7	Z_CLK	*Recognititon/control accessories*	
8	Z_DATA	*Recognititon/control accessories*	
(1)	(2)	(3)	(4)
9	MICG	*Ground For microphone*	In
10	MIC	*Microphone input*	
11	AUD	*Loudspeaker*	Out
12	AUDG	*Ground For external speaker*	

2.4 Layanan Pesan Singkat (SMS)

SMS adalah sebuah mekanisme pengiriman pesan singkat melalui jaringan bergerak (*mobile network*). Pesan yang dikirim oleh sebuah pesawat telepon seluler, ditampung di sentral SMS yang disebut dengan SMSC (*Short Message Service Centre*). oleh SMSC pesan ini kemudian di transfer ke telepon seluler yang dituju.

2.4.1 Format SMS

Menurut sepesifikasi ETSI (*European Telcomunications Standards Institute*) panjang maksimum sebuah pesan adalah 160 karakter.Fasilitas ini disediakan oleh jaringan telepon seluler.Teknologi yang mendukung layanan SMS antara lain adalah GSM,Time *Division Multiple Access* (TDMA) dan *Code Division Multiple access* (CDMA).

Sebenarnya, panjang pesan maksimum yang dapat dikirimkan melalui SMS adalah 140 karakter. Akan tetapi dengan teknik kompresi *septe-to-octet* yang mengkodekan data karakter 7 bit menjadi 8 bit,maka ditampung pesan sebanyak 160 karakter yang dikompres menjadi 140 karakter.Teknik ini bertumpu pada keadaan bahwa kode karakter SCII *alfanumerik* yang mempunyai lebar data 7 bit (bit ke7 selalu bernilai 0 sehingga bisa diabaikan).Teknik kompresi *sepet-to-octet* dilakukan dengan meyisipkan bit-bit LSB dari karakter selanjutnya kedalam bit-bit MSB dari data sebelumnya secara berkesinambungan. Pengiriman dan penerimaan pesan sms dapat dilakukan melalui mode PDU (*protocol Data Unit*).

2.4.1.1 Format Data Kirim SMS Dengan Mode PDU

Berikut ini adalah susunan format PDU *sent* yaitu untuk mengirimkan SMS dari *Mobile Equipment* ke SMSC,adapun PDU untuk mengirim SMS terdiri dari delapan *header* , yitu sebagai berikut:

a. Nomor SMS_*Centre Header* pertama ini terdiri atas tiga *subheader*, yaitu:

1. Jumlah pasangan heksa decimal SMS – centre (bilangan heksa).
2. National / international code.

   Untuk national,kode *subheadernya* adalah 81.

   Untuk international,kode *subheadernya* adalah 91.

3 No SMS – Centrenya sendiri,dalam pasangan heksa dibolak – balik.

Jika tertinggal satu angka heksa yang tidak memiliki pasangan,angka tersebut akan dipasangkan dengan huruf F didepanya.

Contoh untuk SMS _ *Centre Exelcom* dapat ditulis dengan dua cara, sebagai berikut:

Cara 1:

0818445009 diubah menjadi :

- 06
- 81 → 1 pasang
- 80-81-44-05-90 → 5 pasang

total = 6 pasang

Digabung menjadi :0681808144590

Cara 2 :

- 07 → ada 7 pasang

- 91 → 1 pasang ——— total = 7 pasang
- 26-18-48-54-00-F9 → 6 pasang ———

Digabung menjadi :07912618485400F9

Berikut ini beberapa nomor SMS _ Centre operator selluler di Indonesia:

Cara 1:

Tabel **2.3**. Nomor SMS –Coperator seluler di Indonesia

No	Operator Selluler	No SMS – C	Kode PDU
1	Telkomsel	081100000	0691801100000
2	Satelindo	0816124	058106121F5
3	Exelcom	0818445009	06818081440590
4	Indosat-M3	085500000	0681805500000

Cara 2 :

Tabel 2.4 Nomor SMSC Operator Selluler Di Indonesia yang telah *dikonversikan* ke dalam PDU kode [6].

No	Operator Selluler	No SMS – C	Kode PDU
1	Telkomsel	081100000	07912618010000F0
2	Satelindo	0816124	059126181652
3	Exelcom	0818445009	0791261848500F9
4	Indosat-M3	085500000	07912658050000F0

b. Tipe SMS

Untuk *send* tipe SMS = 1. jadi bilangna heksanyaadalah 01.

c. Nomor Referensi

Nomor referensi ini dibiarkan dulu 0 , jadi bilangan heksanya adalah 00. nantinya akan diberikan sebuah nomor referensi otomatis oleh ponsel / alat SMS.

d. Nomor Ponsel Penerima

Sama seperti cara menulis PDU header untuk SMSC, header ni juga terdiri atas tiga bagian, sebagai berikut:

- Jumlah bilangan decimal nomor ponsel yang dituju dalam bilangan heksa.
- National / International Kode
  - untuk national, kode subheadernya = 81
  - untuk international, kode subheadernya = 91
- Nomor ponsel yang dituju, dalam pasangan heksa dibolak-balik. Jika tertinggal satu angka heksa yang tidak dimiliki pasangan, angka tersebut dipasangkan dengan huruf F di depannya.

Contoh : Untuk nomor ponsel yag dituju = 628129573337, dapat ditulis dalam dua cara :

Cara 1 :

628129573337 diubah menjadi :

- 0B → ada 11 angka
- 81
- 80 – 21 – 59 – 37 – 33 – F7

Digbung menjadi : 0B818121593733F7

Cara 2 :

628129573337 diubah menjadi :

- 0C → ada 12 angka
- 91
- 62 – 18 – 92 – 75 – 00 – 73

Digabung menjadi 0C91621892750073

e. Bentuk SMS

Terdiri dari : 0 → 00 → dikirim sebagai SMS.

1 → 01 → dikirim sebagai telex.

2 → 02 → dikirim sebagai fax

f. Skema Encoding Data I/O

Terdapat dua skema, yaitu :

1. Skema 7 bit → ditandai dengan angka 0 → 00.
2. Skema 8 bit → ditandai dengan angka lebih besar dari 0 → diubah ke heksa.

Kebanyakan ponsel / SMS gateway yang ada dipasaran sekarang menggunakan skema 7 bit sehingga kita menggunakan kode 00.

g. Jangka Waktu Sebelum SMS Expired

Jika bagian ini di-skip, itu berarti kita tidak membatasi waktu berlakunya SMS. Sedangkan jika diisi dengan suatu bilangna *integer* yang kemudian diubah ke pasangan heksa tertentu, bilangan yang kita berikan tesebut akan mewakili validitas SMS tersebut.

h. Isi SMS

Header ini terdiri dari dua subheader, yaitu :

- Panjang isi (jumlah huruf dari isi)

  Misalnya : untuk kata "hello" → ada 5 huruf → 05

- Isi berupa pasangan bilangan heksa

  Untuk ponsel / SMSberskema encoding 7 bit, jika kita mengetikkan suatu huruf dari keyboard-nya berarti kita telah membuat 7 angka I/O berurutan.

Ada dua langkah yang harus kita lakukan untuk mengkonversikan isi SMS, yaitu

Langkah pertama : mengubahnya menjadi 7 bit.

Langkah kedua : mengubah kode 7 bit menjadi 8 bit, yang diwakili oleh pasangan heksa.

Contoh : untuk kata "hello"

Langkah pertama :

Bit	*7*	*I*
h	110	100
e	110	0101
L	110	1100
L	110	1100
o	110	1111

Table **2.5** Skema 7 Bit SMS pada Telepon Seluler

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	i	0	1	0	1
b4	**b3**	**b2**	**b1**		**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**
0	0	0	0	**0**	@	Δ	SP	0	-	P	-	p
0	0	0	1	**1**			!	1	A	Q	a	q
0	0	1	0	**2**	$	Φ	"	2	B	R	b	r
0	0	1	1	**3**		Γ	#	3	C	S	c	s
0	1	0	0	**4**		^		4	D	T	d	t
0	1	0	1	**5**		Ω	%	5	E	Ŭ	e	u
0	1	1	0	**6**		∏	-	6	F	V	f	v
0	1	1	1	**7**		ψ	.	7	G	W	g	w
1	0	0	0	**8**		Σ	(	8	H	X	h	x
1	0	0	1	**9**		Θ	)	9	I	Y	i	y
1	0	1	0	**10**	LF	≡	•	:	J	Z	j	z
1	0	1	1	**11**			+	;	K	Ă	k	ã
1	1	0	0	**12**			,	<	L	Ő	l	ŏ
1	1	0	1	**13**	CR		-	=	M		m	
1	1	1	0	**14**		β	.	>	N	Ű	n	ű
1	1	1	1	**15**			/	?	O		o	

Langkah kedua :

		E	8	
h	1	110	1000	

Dari data disamping uraian untuk memunculkan huruf (h – e – l –l – o) dari pasangan heksa dan akan membentuk kata 'hello'

		3	2	
e	00	11	0010	1

		9	B	
l	100	1	1011	00

		F	DD	
l	1111		1101	100

		0	6	
0	0000	0	110	1111

Dengan demikian kata "hello" hasil konversinya menjadi : E8329BF

Contoh format SMS kirim dalam PDU :

07912618485400F901000C9126189275337300000 5E8329BFD06

Penjelasan dari format diatas dijelaskan dalam table berikut :

Table 2.6. table penjelasan Format SMS kirim dalam PDU[7].

Heksa	**Penjelasan**
07	Panjang pesan nomor SMSC
91	Tipe kode national / international
2648485400F9	Nomor Service Center
01	Tipe SMS (01 = tipe untuk pengiriman)
00	Nomor referensi
0c	Panjang nomor HP penerima
91	Tipe kode national / international
261892753373	Nomor HP penerima
00	Tipe bentuk SMS (00 = dikirim sebagai SMS)
00	Tipe data Coding
05	Panjang pasangan dari SMS
E8239BFD06	Isi SMS

2.4.1.2 Format Data Terima SMS Dengan Mode PDU

Format PDU untuk penerimaan SMS terdiri dari 8 header, yaitu sebagai berikut :

1. No SMSC : +62855000000
2. Tipe SMS : *text*
3. No ponsel penerima : +6285755298897
4. Bentuk SMS :format yang sudah ditentukan
5. Skema Encoding
6. Tanggal dan waktu SMS di-stamp di SMSC

   Diwakili oleh 12 pasangan bilangan heksa (6 pasang) yang berarti:yy/mm/dd hh:mm:ss

   9020023 512380 → 23/02/2009 15 : 32 : 08
7. Batas waktu validitas → jika tidak dibatasi dilambangkan dengan 00
8. Isi SMS

   Isi SMS yang berupa tulisan nantinya akan dikonversi yaitu mengubahnya menjadi 8 bit dan kemudian menjadi 7 bit.

Contoh format SMS terima dalam PDU :

0791265805000f0040C912658161073980000207022512380000 5E8329BFD6

Penjelesan dariformat tersebut dapat dijelaskan sebagai berikut :

Tabel 2.7 Tabel penjelesan Format SMS Terima Dalam PDU

Heksa	**Penjelasan**
07	Panjang pesan nomor SMSC
91	Tipe kode national / international
2658050000F0	Nomor Service Center
04	Tipe SMS (01 = tipe untuk pengiriman)
0C	Panjang nomor HP penerima
91	Tipe kode national / international
265816107398	Nomor HP penerima
00	Tipe bentuk SMS (00 = dikirim sebagai SMS)
00	Tipe data Coding
207022512380	Waktu SMS sampai di SMSC yaitu tanggal 23/02/09,15-32-08
00	Jangka waktu SMS expired (00=tidak memeliki batas)
05	Panjang pesan dari SMS
E832bfd06	Isi pesan

2.4.2 Perintah AT-Command

Komunikasi data antar telpon seluler dengan peralatan lain (*computer atau mikrokontroler*) dilakukan secara serial menggunakan perintah-perintah *AT Command* seperti halnya sebuah modem. Dengan mengirimkan perintah-perintah AT yang sepesifik,dapat memerintahkan telpon seluler untuk melakukan apa yang diinginkan.

Sebagai contoh,jika mengirimkan "AT" ke telpon seluler, maka ia akan menjawab "OK"seperti perintah dibawah ini:

Pc/µC	**AT**
HP	**OK**

Pasangan AT-OK ini dapat digunakan untuk mendeteksi keberadaan telepon seluler.Perintah untuk mengirimkan pesan menggunakan AT adalah:

Pc/µC	**AT+CMSS=1**
HP	**+CMSS:96**
OK	

Berarti memerintahkan kepada telpon seluler untuk mengirimkan pesan nomor 1 yang ada di memori telponj.Satu hal yang menjadi cirri-ciri adalah bahwa pesawat telepon seluler selalu mengakhiri pesan dengan kata "OK".Sehingga mikrokontroler atau PC dapat mendeteksi akhir dari sebuah pesan yang dikirim oleh telepon seluler.

Selain perintah AT diatas juga masih ada perintah yang lain,missal:

Pc/µC	**AT+CMGR=1**
HP	**+CMGR=1**
OK	

Perintah AT+CMGR=1 berarti memerintahkan kepada telepon seluler untuk membaca pesan nomor 1 yang tersimpan dalam memori telepon seluler.

Perintah AT+CMGD digunakan untuk menghapus pesan yang tersimpan didalam memori telpon seluler.

Dalam laporan akhir ini tidak membahas seluruh perintah AT yang tersedia akan tetapi hanya membahas perintah-perintah AT yang berkaitan dengan operasi SMS seperti mengirim SMS dan menghapus SMS.

2.5 Komunikasi Serial

Komunikasi serial adalah Komunikasi dimana data yang dikirimkan dan data yang diterima secara berurutan tiap bitnya.*Tiap byte* data di transmisikan pada suatu waktu dengan interval waktu yang ditentukan untuk tip bit.Bit-bit dikirimkan secara berurutan dan tidak serempak, kecepatan pemindahan data lebih rendah dibandingkan pengiriman secara pararel.Pengiriman akan dimulai dari LSB (*Least Significant Bit*) dan diakhiri dengan MSB (*most Significant Bit*).setiap karakter yang dikirimkan disusun sesuai dengan suatu urutan bit tertentu.

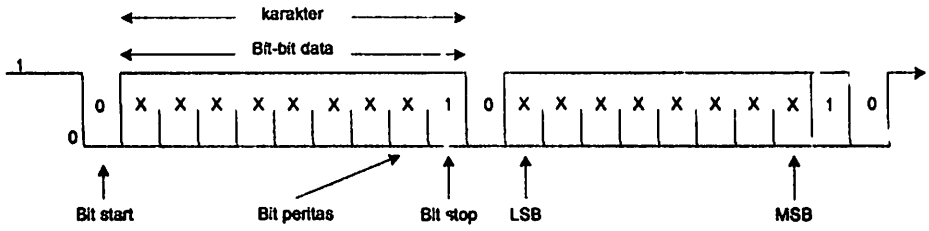
Berdasarkan formatnya ,pengiriman data komunikasi serial dibedakan menjadi dua bentuk,yaitu:

2.5.1 Komunikasi Serial Asinkron

Pengiriman data *asinkron* digunakan apabila setiap kali pengiriman data, dikirimkan hanya satu karakter yang didahului dengan *bit start* dan diakhiri oleh *bit stop*. Antara satu karakter dengan karakter yang lain tidak ada waktu yang tetap. Pengiriman dengan menggunakan *serial asinkron* lebih sederhana daripada pengiriman *serial sinkron*.

Transmisi asinkron biasanya disebut transmisi awal-akhir (start-stop transmission),karena tiap karakter mengalami sinkronisasi dengan jalan penggunaan bit awal dan bit akhir.Detak penerimaan dibangkitkan secara local didalam penerimaan dan tetap dijaga agar sesuai dengan detak pengiriman yang

menggunakan bit awal (start bit) dan bit akhir (stop bit)yang dikirimkan dengan setiap karakter.Penyesuaian detak pengiriman dan penerimaan terjadi karakter perkarakter.

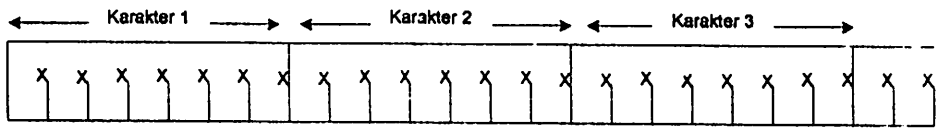


Gambar 2.4. Sinkronisasi awal-akhir

Gambar diatas menunjukkan sinkronisasi awal dan akhir setiap karakter.Dari gambar dapat dilihat bahwa bit kedelapan disebut bit paritas.Bit ini akan dipasang pada 1 atau 0 untuk menyakinkan cacah bit setiap karakter adalah genap untuk paritas genap atau ganjil untuk paritas ganjil.

2.5.2 Komunikasi Serial Sinkron

Pada komunikasi data sinkron pengiriman datanya akan dikirimkan secara bersambung tanpa bit awal dan bit akhir sehingga pada komunikasi sinkron pengiriman ataupun bit awal dan bit akhir sehingga pada komunikasi sinkron pengiriman ataupun penerimaan bekerja bersama-sama dan sinkronisasi dilakukan stiap sekian ribu bit data .Sinkronisasi dilakukan dan dijaga baik pada waktu tidak ada data yang dikirim maupun sesaat sebelum pengiriman terjadi.Sinkronisasi terjadi dengan jalan mengirimkan pola data tertentu antara pengirim dan penerima.Jika pada mode asinkron tiap huruf mempunyai bit awal-akhir sehingga apabila terjadi kesalahan karena sinkronisasi maka satu karakter yang hilang sedangkan mode sinkron, sinkronisasi dilakukan setiap ribu bit data sehingga apabila terjadi kesalahan sinkronisasi maka hanya satu bit data yang hilang.



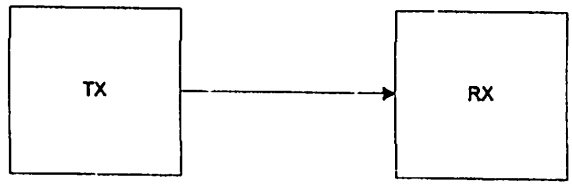
Gambar 2.5. Aliran Data Sinkron[3]

Pada transmisi sinkrondigunakan untuk menyalurkan data secara blok. Dalam transmisi tiap blok panjangnya sama. Komunikasi sinkron menggunakan kemampuan saluran data secara efisien karena hanya dilakukan bila telah mempunyai sejumlah blok data.

Berdasarkan arah komunikasinya pengiriman data serial debadakan menjadi dua yaitu :

2.5.3 Komunikasi Simplex

Komunikasi secara *simplex* merupakan sistem komunikasi data yang hanya dapat mengirimkan data pada satu arah saja (searah). Arah komunikasi secara *simplex* ini hanya sattu sisi saja yang melakukan penerimaan. Sisi pengirim (*transmitter*) tidak akan menerima tanggapan apapun dari sisi penerima (*receiver*).



Gambar 2.6. Komunikasi simplex[4]

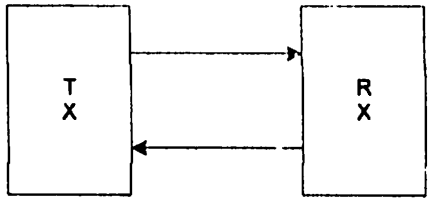
Gambar 2.6 menunjukkan komunikasi secara simplex dimana stasiun hanya membutuhkan satu *transmitter* atau *receiver*, sementara transmitter hanya menggunakan satu saluran transmisi.

2.5.4 Komunikasi Duplex

Komunikasi secara *duplex* adalah komunikasi yang dapat mengirimkan data dua arah. Pada sistem ini dibedakan menjadi dua yaitu *Half Duplex* dan *Full Duplex.*

2.5.4.1 Half Duplex

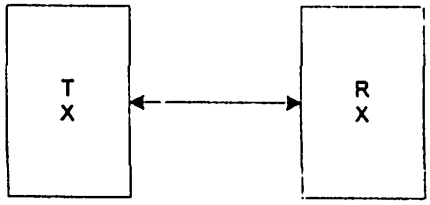
Sistem komunikasi yang dapat mengirimkan data secara bolak-balik (dua arah) secara bergantian antara dua sistem (Tx dan Rx) dan tidak dapat bersamaan. Gambar dibawah ini menunjukkan komunikasi half duplex.



Gambar 2.7. Komunikasi Half Duplex[4]

2.5.4.2 Full Duplex

Sistem komunikasi yang dapat mengirimkan data dalam dua arah pada waktu yang bersamaan, jadi tidak perlu saling menunggu untuk bergantian komunikasi. Pada gambar 2.8 menunjukkan komunikasi *Full Duplex*. Contoh untuk komunikasi full duplex adalah telepon kabel, pertukaran antara dua sistem computer yang dihubungkan ke suatu jaringan data.

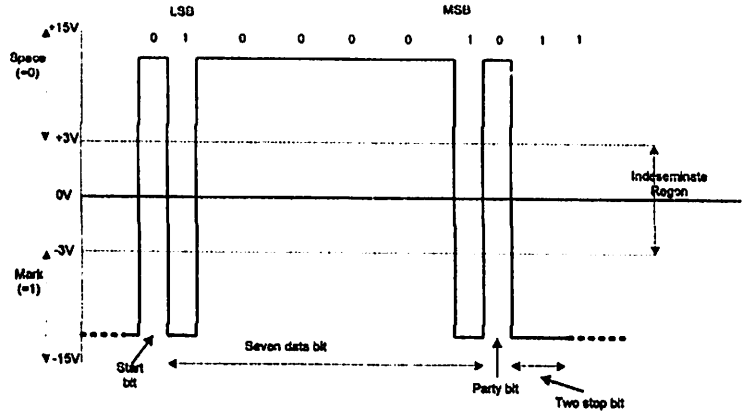


Gambar 2.8. Komunikasi Full Duplex[4]

2.6 RS-232

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

Karakteristik elektris dari sistem RS-232 menunjukkan bahwa level tegangan +3 Vdc sampai dengan +15 Vdc mewakili level logika rendah (logika 'NOL' / *spac*), dan tegangan -3 Vdc sampai dengan -15 Vdc mewakili level tinggi (logika 'SATU' / *mark*).



Gambar 2.9 Format Pengiriman Data Asinkron[5]

Beberapa protocol dalam *interface* RS-232 adalah :

- *Start Bit*

  Merupkan sebuah bit dengan logic "0" dimana bit ini yang menandakan bahwa akan ada karakter atau data yang mengikutinya. Bit ini langsung diberikan oleh sinyal device tanpa harus mensetnya terlebih dahulu.

- *Data Bit*

Merupakan bit yang mewakili dari karakter yang diikutinya. Data bit ini dapat diset sepanjang antara 5 sampai 8 bit.

- *Pariti Bit*

  Merupakan bit yang digunakan sebagai error checking pada receiver, apabila terjadi kesalahan maka *receiver* akan menset *error flag (parity error)* pada special register. Pariti bit ini menghitung jumlah data yang berlogic '1' pada data bit. Perhitungan jumlah data bit tersebut tergantung dari jenis pariti yang diset. Untuk pariti *even* maka jumlah data bit yang berlogic '1' ditambah dengan pariti bit dan akan menghasilkan jumlah yang ganjil. Sedangkan untuk pariti *mark* merupakan pariti bit selalu berlogic '1' begitu pula pada space, pariti bit selalu berlogic '0' dan pariti *none* disini parity bit yang diabaikan.

- *Stop Bit*

  Merupakan bit yang menandakan akhir dari suatu paket data (biasanya 1 byte data). Seperti pada *start* bit, bit ini langsung diberikan pada serial device. Stop bit ini dapat diset panjangnya menjadi satu nit, satu setengah dan dua bit.

- *Baut Rate*

  Sebenarnya baut rate berarti pergantian kondisi tiap detik (*State Change of the Line persecond*), tetapi karena hanya ada 2 kondisi pada serial (logic 0 dan 1) maka dapat juga digunakan untuk menunjukkan kecepatan dari transmisi (*bits persecond*).

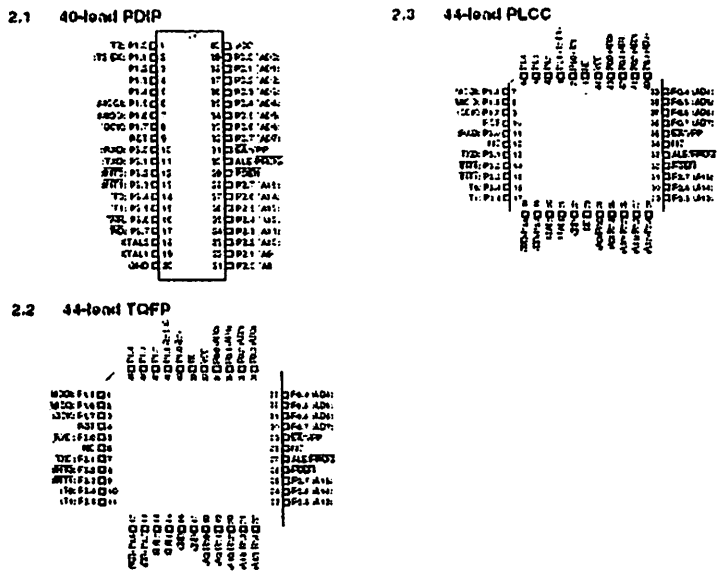
Table 2.8 Spesifikasi RS-232[5]

Keistimewaan	Karaktiristik
Jenis Operasi	Single ended (tak seimbang)
Jenis Penggerak dan Penerima per jalur	1 driver 1 receiver
Data rate maksimum	20 kbps
Panjang saluran maksimum	50 ft
Tegangan keluaran penggerak	±3 – ±15 volt
Sensitivitas penerima	±3 volt

2.6.1 MAX-232

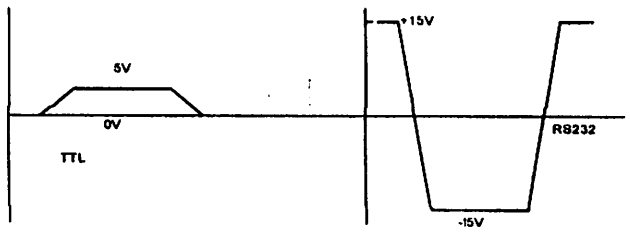
Saluran data pada port seri PC menggunakan standard RS-232, dimana logic 0 (*low*) dinyatakan sebagai tegangan antara +3 Volt sampai +15 Volt dan logic 1 (*high*) inyatakan tegangan antara -3 Volt sampai -15 Volt. Level tegangan ini tidak sesuai dengan level tegangan yang dipakai port seri mikrokontroler yang menggunakan standard TTL ((*Transistor Transistor Logic*), yaitu level tegangan baku dalam rangkaian – rangkaian digital.

Dalam standad TTl logic 0 (*low*) dinyatakan sebagai tegangan anara 0 Volt sampai 0.8 Volt, dan logic 1 (*high*) dinyatakan sebagai tegangan antara 3,5 Volt sampai 5 Volt. Karena perbedaan tegangan tersebut, maka agar port seri PC tidak merusak seri port Mikrokontroler antara keduanya dipasangkan IC MAX-232. Pada dasarnya IC ini hanya digunakan sebagai pengubah tegangan ke level TTL, tidak berfungsi sebagai pengkodean sinyal yang melewati RS-232, dan juga tidak mengkonversi data serial ke pararel.



Gambar 2.10. Konfigurasi Pin dan Blok diagram RS-232[5]

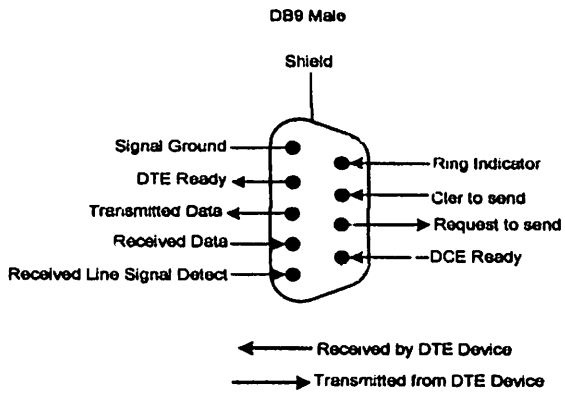
Konverter RS-232 level akan mengnggap tegnan antara +3 hingga +15 Volt sebagai teganga '0'. Sedangkan tegangan -3 hingga -15 Volt dianggap sebagai tegangan '1', level antara -3 hingga =3 tidak didefinisikn, sebab di daerah ini kemungkinan adalah *noise*. Level TTL diatas 2 Volt yang dianggap sebagi level '1' akan dikonversikan ke level RS-232 yaitu sebesar -15 Volt. Sedangkan level '0' TTL, yaitu tegangan dibawah 0.8 Volt akan dionversikan ke +15 Volt, demikin juga pada konversi sebliknya, level +3 hingga +15 Volt akan dikonversikan ke level TTL 5 Vol fan -3 hingga -15 vol akan dikonversikan ke 0 Volt.



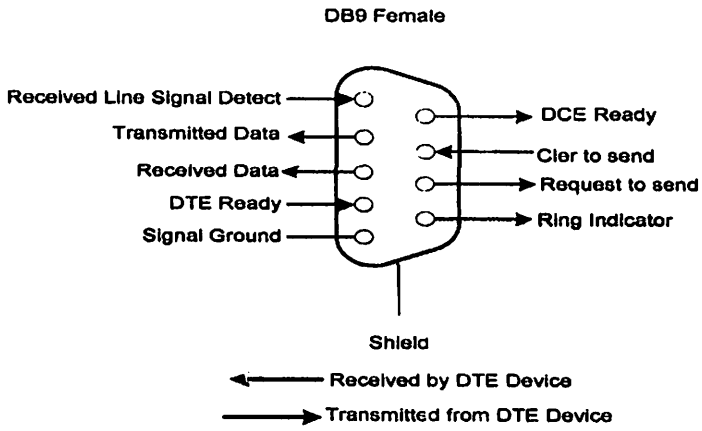
Gambar 2.11. Level Converter TTL ke RS-232[5]

2.7 DB-9

Proses transfer secara serial menggunakan RS-232 antara 2 buah terminal biasanya memerlukan sebuah DTE (*Data Terminal eqipment*) untuk masing-masing terminal dan sebuah DCE (*Data Commuication Eqipment*).



Gambar 2.12. DTE *Device Connector*[5]



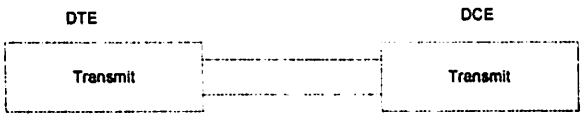
Gambar 2.13. DCE *Device Connector*[5]

Data yang ditransfer dari satu terminal akan diterima oleh terminal lainnya, dan emikian juga sebaliknya melalui perangkat peralatan di atas. Table berikut menjelaskan eskripsi pin out untuk DB-9 dan fungsi masing-masing pin.

Table 2.9 Deskripsi Pin DB-9 (male) dan Fungsinya[5]

Pin	Singkatan	Keterangan	Fungsi
1	CD	*Carrier Detect*	Saat modem mendeteksi suatu 'carrier' dari modem lain (di tempat lain) maka sinyal ini akan diaktifkan
2	RD	*Receive Data*	Untuk penerimaan data serial (RXD)
3	TD	*Transmit Data*	Untuk penerimaan data serial (TXD)
4	DR	*Data Terminal Ready*	Kebalikan dari DSR, memberiahukan bahwqa UART siap melakukan komunikasi
5	SG	*Signal Ground*	Ground
6	DSR	*Data Set Ready*	Memberitahu UART bahwa modem siap melakukan hubugan komunikasi
7	RTS	*Request To Send*	Sinyal untuk menginformasikan modem bahwa UART siap melakukan pertukaran data
8	CTS	*Clear To Send*	Digunakan untuk memberitahukan bahwa modem siap untuk melakukan perukaran data
9	RI	*Ring Indikator*	Akan aktif jika modem mendeteksi adanya saluran telepon

Jenis data yang akan ditransfer adlah dalam bentuk data biner (bit per bit transfer) dengan satuan baud untuk kecepaan proses transfernya (bi per detik).



Gambar 2.14. Transfer Data antara DTE dengan DCE[5]

Konsep *interface* antara DTE dan DCE dilakukan dengan cara sebagai berikut :

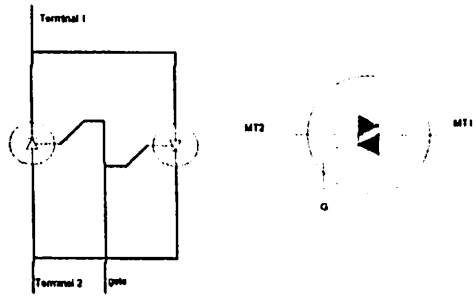
- Ketika DTE ingin mengirim data, sebuah protocol yaitu RTS dikirim untuk memberitahu DCE.
- Pada saat itu input RTS pada DCE menjadi aktif.
- Jika DCE mampu menerima data, maka ia akan membalasnya dengan mengirim CTS.
- Begitu DTE menerima balasan, input CTS-nya diaktifkan.
- Pengirim data dilakukan melalui TxD.
- Pengirim data dilakukan melalui RxD.
- Proses tersebut ddilakukan secara berulang-ulang sampai semua data selesai ditransfer.

2.8 Triac

Triac atau thyristor dua arah adalaah kependekan dan *Triode Alternating Current* yang merupkan switch atau saklar triode untuk arus AC. Triac dapat menghalangi arus dalam dua arah antara terminal utama (terminal 1 / anoda) dan (terminal 2 / katoda)dan juga dapat dipicu agar melakukan konduksi dalam dua arah dan tanggap terhaap sinyal positif maupun sinyal negative ynag dikenakan pada terminal gerbang.

Sekali triac mulai konduksi maka akan terus kondukai sampai ia dihentikan dari luar, sam seperti SCR. Triac berlaku seperti dua buah SCR yang masing-

masing bekeja pada setiap arah yang diunjukkan pada diagram rangkaian pengganti pada gambar 4a, sedangkan simbol standar dari triac ditunjukkan pada gambar 4b. simbol ini juga memperlihatkan sifa dua arah dari triac.



Gambar 2.14. (a) Hubungan Dua SCR yang Bekerja sebagai Triac;
(b) Simbol Triac

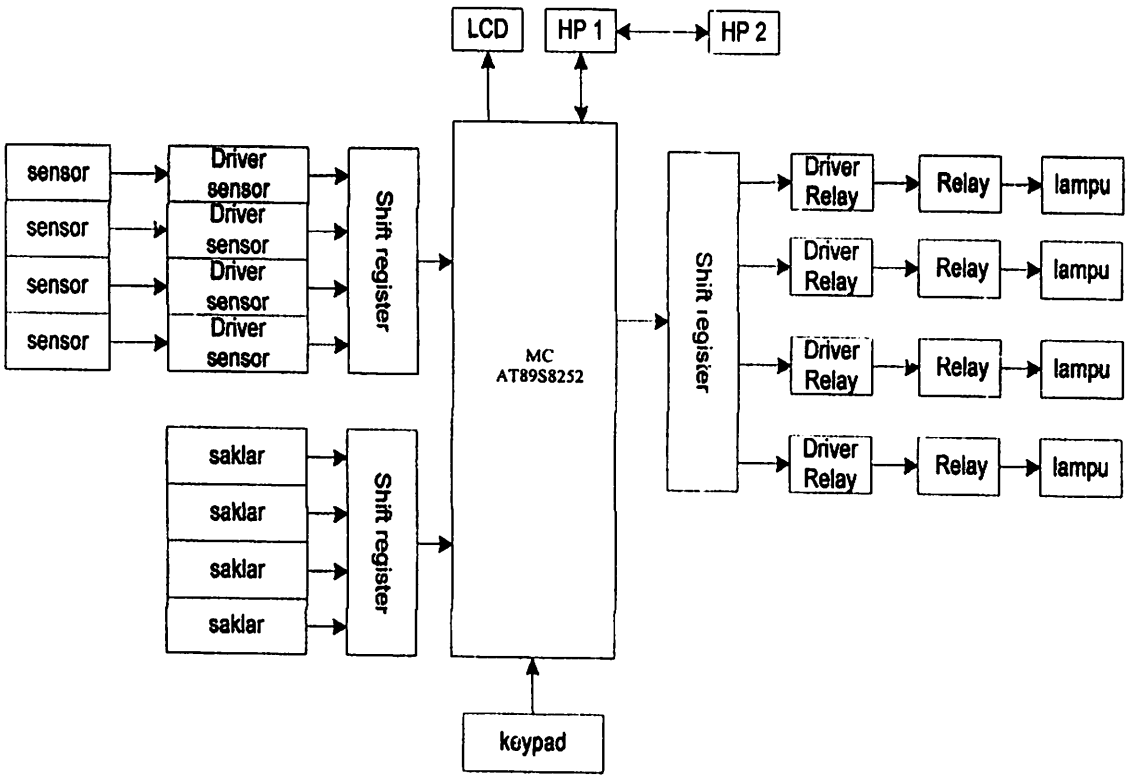
Dari gambar 4b terlihat bahwa elektroda triac terdiri dari iga kaki yaitu anoda (A), katoda (K), dan gate (G). Tegangan penyalaan biasanya tinggi, sehingga cara yang normal untuk menyalakan triac adalah dengan menerapkan pemicu berprategangan maju. Lembaran data untuk triac selalu mencatumkan tegangan pemicu dan arus pemicu ini. Apabila tegangan sumber V mempunyai polaritas seperti yng tampak padagambar 4a, maka harus menerapkan picu positif. Bila tegangan sumber V mempunyai tegangan berlawanan, dibutuhkan pemicu negatif.

BAB III

PERENCANAAN DAN PEMBUATAN ALAT

3.1. Diagram blok rangkaian

Sebelum membuat perangkat keras maka dibuat dahulu diagram blokнya untuk mempermudah pembuatan perangkat keras.Gambar dibawah ini merupakan diagram blok dari alat yang akan dibuat dalam tugas akhir.



Gambar 3.1. blok diagram

Keterangan :

a. HP 1 :Hp yang dipegang oleh pengguna untuk menyalakan atau mematikan lampu dan untuk mengaktifkan atau menonaktifkan sistem pada alat juga untuk menerima laporan keadaan lampu.

b. HP 2 :Hp yang berada pada rangkaian.

c. MC AT89S8252 :Digunakan sebagai pusat kontrol sistem.

d. Shift Register IN :Untuk menjalankan perintah data.dari perintah sensor ke MC

e. Driver Sensor :Sebagai pembaca sensor dan dirubah menjadi perintah data,

f. Sensor :Untuk menangkap sensor cahaya dari lampu

g. Saklar :Sebagai alat menghidupkan atau mematikan lampu secara manual

h. Shift Register OUT :Menjalankan perintah dari MC ke lampu.

i. Driver Relay :Sebagai perintah ke relay

j. Relay :untuk saklar lampu dan yang telah dijalankan dari rangkaian

k. Lampu :beban AC(220V)

l. Keypad :Untuk mengatur no Hp. Pengirim

3.2 Spesifikasi Alat

Alat pengendali lampu rumah melalui SMS ini mempunyai sepesifikasi berikut:

a. Mekanik

b. Menggunakan *handpone* M35

c. Sumber AC 220 Volt

d. Dapat memberikan report tentang keadaan lampu berupa SMS jika sistem di ON kan.

e. Dapat menyalakan dan memtikan lampu yang berjumlah 4 buah

f. Menggunakan MCU AT89S8252 sebagai control.

3.3 Langkah-langkah perencanaan

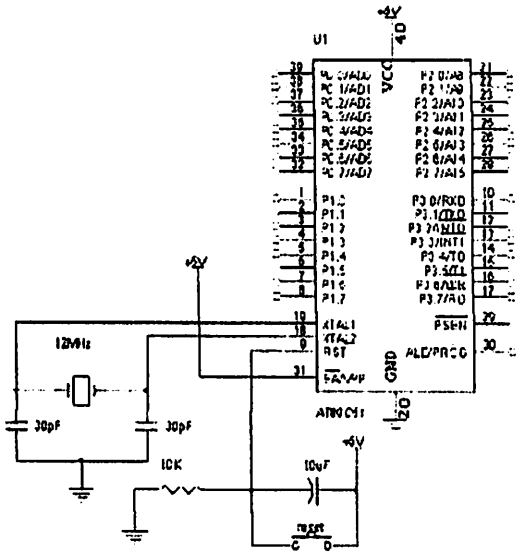
Perencanaan laporan akhir ini terdiri dari tiga bagian,yaitu perencanaan perangkat keras, perencanaan mekanik,dan perencanaan rangkaian mikrokontroler AT89S8252,perencanaan rangkaian interface RS-232 dari computer ke MCU, Perencanaan rangkaian driver lampu.Perencanaan mekanik meliputi perencanaan pembuatan book untuk tempat alat tersebut.Perencanaan Perangkat lunak meliputi pemilihan pin-pin yang digunakan pada mikrokontroler,diagram alir,dan urutan kerja program,

3.3.1 Perencanaan Perangkat Keras

Perencanaan perangkat keras meliputi perencanaan rangkaian mikrokontroler AT89S8252,perencanaan rangkaian interface RS-232 dari hp ke MCU,perencanaan rangkaian driver lampu.

3.3.1.1 Rangkaian control menggunakan AT89S8252

Rangkaian minimum dari mikrokontroler AT89S8252 dari ragkaian sederhana ini mikrokontroler dibuat sebagai system minimum menjadi pengontrol alat,disamping itu rangkaian ini dapat dibuat macam-macam alat dengan menambah sedikit komponen tambahan lainya.Dari rangkaian tersebut yang berpengaruh terhadap kecepatan proses menjalankan program adalah kristal.Adapun rangkainya ditunjukan seperti pada gambar 3.2



Gambar 3.2 *Minimum Sistem* AT89S8252

Agar sebuah mikrokontroler dapat bekerja sebagai pengontrol ,maka kaki-kaki/port mikrokontroler dihubungkan dalamrangkain-rankaian eksternal.

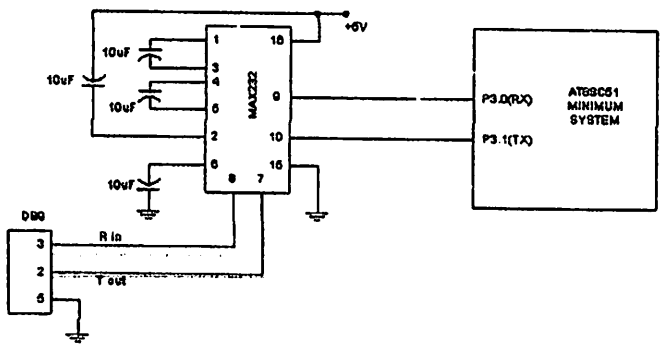
Dalam perancangan ini *,port* yang digunakan adalah sebagai berikut:

1. Port 2.7, 2.5, 2.3, 2.1 digunakan sebagai driver lampu.
2. Port 2.6, 2.4, 2.2, 2.0 digunakan sebagai umpan balik
3. port 3.0 dan port 3.1 digunakan sebagai transfer data dari hap ke MC melalui max 232
4. Port 3.5 digunakan untuk buzzer.
5. Port 3.6 dan 3.7 digunakan untuk buzzer.
6. XTAL1 dan XTAL2 digunakan sebagai input dari rangkaian osilator 11,059 MHz,kapasitor C1 dan C2 yang masing-masing bernilai 30pF,akan membangkitkan pulsa clock yang menjadi penggerak bagi seluruh operasi internal MCU.
7. VCC dihubungkan dengan tegangan sebesar +5V

8. GND dihubungkan ke *ground* catu daya.
9. Reset digunakan intuk mereset program contiol MCU,diman MCU memiliki masukan aktif *high.*

3.3.1.2 Rangkaian *interface* RS-232 dari hp ke MCU

Karena kabel data hp tidak bekerja pada level tegangan TTL,melainkan level tegangan RS-232.Untuk itu diperlukan komponen tambahan yang berfungsi untuk mengkonversi level tegangan TTL ke level teganga RS-232,begitu juga sebaliknya.Komponen ang digunakan adalah IC MAX232 yang memerlukan beberapa komponen tambahan berupa empat buah kapasitor yang nilainya adalah22 µF untuk seluruh kapasitor.MAX232 memiliki sepasang terminal masukan level tegangan TTL yang berkorespondensi dengan sepasang terminal keluaran level tegngan RS-232,juga sepasang terminal masukan level RS-232 yang berkorespondensi dengan sepasang terminal keluaran level tegangan TTL



Gambar3.3 Rangkaian MAX232

3.3.1.3. Perencanaan Rangkaian Driver lampu

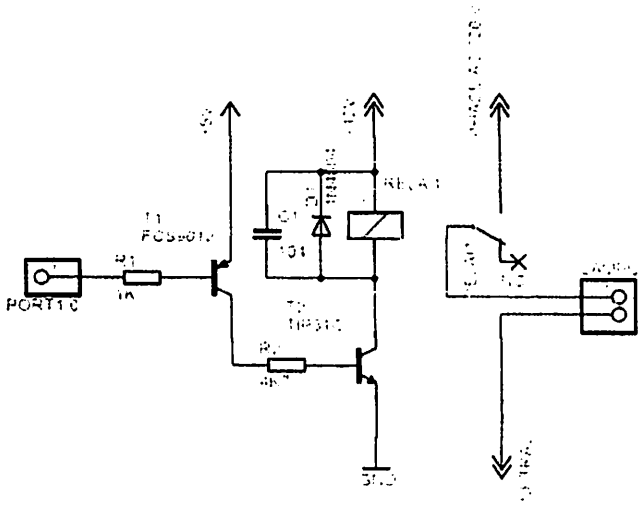
Rangkaian driver ini menggunakan triac sebagai switching yang digunakan untuk menyalakan dan mematikan lampu lampu.Dana sebagai pengaman ketika dihubungkan dengan MCU digunakan MOC 3021. Prinsip kerja rangkaian ini adalah sebagai berikut : keluaran mikrokontroler akan menjadi Vin dan kemudian melewati Rin yang menjadi pemicu 3021.Keluaran dari MOC 3021 ini digunakan untuk men-triger TRIAC sehungga bisa dilewati tegangan (Vac) sumber.

MOC 3021 ini merupakan opto diac,dimana untuk mengaktifkanya dibutuhkan I_F max = 15ma.MOC 3021 pada percobaan menggunakan I_F =6mA,mempunyai tegangan jatuh sebesar 1,5V Maka :

$$RI = \frac{Vcc - 1{,}5V}{I_F}$$

$$= \frac{5 - 1{,}5V}{6mA}$$

$$= 583\ \Omega$$



Gambar 3.4 Rangkaian driver lampu

Dengan menghubungkan rangkaian dioda secara seri maka setiap pasang dioda akan menghasilkan tegangan 0.7 V karena dioda yang digunakan adalah dioda silicon. Jadi ujung-ujung rangkaian dioda akan menghasilkan tegangan sebesar 0.7V x 2 atau sebesar 1,4 V.

Untuk menghasilkan riak pada basis kapasitor. Dengan aktifnya *fototransistor* arus kolektor akan mengalir melalui emitor menuju *ground.* Karena rangkaian ini bersifat low,jika ada arus AC pengontrol mikrokontroler akan menerima logika *high.* Sedangkan rangkaian Op Amp berfungsi sebagai buffer dan komparator yang berfungsi mengeluarkan logika 0 dan 1 ke mcu, karena keluaran dari rangkaian optocoupler ketika ada arus dan ketika idak ada arus sudah terjadi perbedaan , tetapi perbedaan itu belum mewakili perubahan logika 0 dan 1 pada level TTL maka diperlukan rangkaian dimana jika masukan tegangan yang mewakili logika satu sama atau lebih besar dari keluaran potensio di kaki input negative Op Amp maka logika keluaran akan bernilai 1, jika keluaran rangkaian optocoupler lebih kecil dari keluaran potensio dikaki input negative Op Amp maka logika keluaran akan bernilai 0.

3.3.1.4 Perencanaan Komunikasi Mikrokontroler AT89S8252 dengan Ponsel

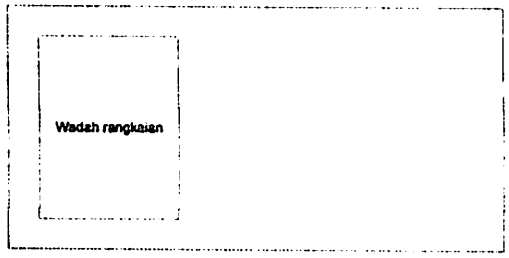
System ini dirancang untuk dapat berkomunikasi dengan ponsel Siemens M35. Mode komunikasi data yang digunakan adalah mode serial dengan *baud rate* sebesar 19.200 atau dengan kecepatan 19.200 bps (bit persecond). System komunikasi antara mikrokontroler dengan ponsel M35 adalah komunikasi seria *asinkron.*

3.3.2 Perencanaan Mekanik

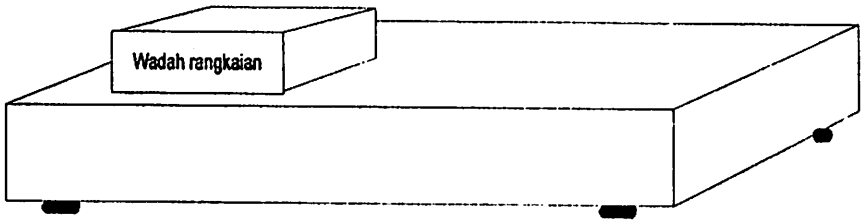
Perencanaan mekanik yang digunakan dalam tugas akhir ini adalah sebuah miniatur rumah yang digunakan sebagai simulasi sistem kerja sekaligus sebagai wadah penempatan.

a. Miniatur berupa papan penempatan alat dengan ukuran :
b. Panjang : 60cm
c. Lebar : 35cm
d. Tinggi : 50cm
e. Bahan papan dari kayu dan untuk wadah rangkaian dari mika.
f. Alat tersebut terdapat empat buah lampu.

Gambar dan bentuk papan yang digunakan untuk wadah rangkaian tampak dalum gambar dibawah ini.



Gambar 3.5 Miniatur dalam rumah tampak atas



Gambar 3.6 papan rankaian tampak dari samping

3.3.3 Perencanaan *Software*

Perancangan perangkat lunak (*software*) digunakan untuk memudahkan penyusunan program. Perangkat lunak ditulis dalam bahasa *Azembler* dan penulisannya sesuai dengan spesifaksi sistem yang telah ditentukan.

Untuk pemakaian mikrokontroler di dalam suatu sistem, perlu direncanakan perangkat lunak mikrokontroller yang dapat mengatur sistem tersebut. Perangkat lunak disini adalah pembuatan atau perintah-perintah (program) di dalam memori yang akan dieksekusikan oleh mikrontroller.

Perencanaan perangkat lunak atau *software* didasarkan pada perencanaan perangkat keras yang dibuat sebelumya, untuk mendapatkan sistem kerja yang diharapkan. Perencanaan *software* meliputi perencanaan komunikasi serial AT89S8252 dan perencanaan pembacaan sensor.

3.3.3.1 Perencanaan Komunikasi Serial AT89S8252

Langkah pertama untuk konfigurasi komunikasi serial pada MCS-51 adalah menentukan *Baud rate*.Yaitu kecepatan transfer data dalam suatu detik. *Baud rate* akan ditentukan sama dengan *baud rate* HP siemen M35 yaitu 19200 bps

Karena sistem ini akan memakai kecepatan *Baud rate* sebesar 19200 bps maka digunakan frekuensi generator pembangkit *Baud rate* dengan rumusan sebagai berikut :

$$Baud\ Rate = \frac{fose}{12x(256 - TH_1)x16}$$

$$19200 = \frac{11{,}0592Mhz}{12x(256 - TH_1)x16}$$

$$3686400 = \frac{11{,}0592Mhz}{256 - TH_1}$$

$$2567\text{-}TH_1 = \frac{11{,}0592x10^6}{3686400}$$

$$TH_1 = 256\text{-}3$$

$$= 253$$

3.3.3.2 Perencanaan Pembacaan dan Pengiriman SMS

Dalam perencanaan ini, untuk pengecekan sensor pada lampu, waktu dinyalakan atau dimatikan, dikirim sebuah karakter.

3.3.3.2.1 Algoritma Program pada MCU

1. Program utama

a. Star

b. Program memulai

c. Baca No

d. Apakah SMS yang masuk dari no yang sudah di *setting* di alat tersebut jika tidak batalkan untuk proses selanjutnya dan jika ya teruskan ke proses program selanjutnya SMS dibaca

e. Baca SMS

f. SMS dibaca apakah perintah dari SMS menyalakan atau memadamkan lampu dana lampu berapa yang parintah

g. Nyalakan atau Padamkan lampu

h. Menyalakan atau Padamkan lampu berupa sesuai perintah format SMS

i. Baca sensor

j. Sensor membaca bagaimana kondisi lampu menyala atau keaadaan padam lalu memberi perintah untuk memberi SMS

k. Mengirim SMS Laporan Keadaan Lampu

l. Mengirim SMS ke HP pemilik sesuai keadaan lampu

m. Tekan tombol

n. Untuk menyalakan atau memadamkan lampu secara manual

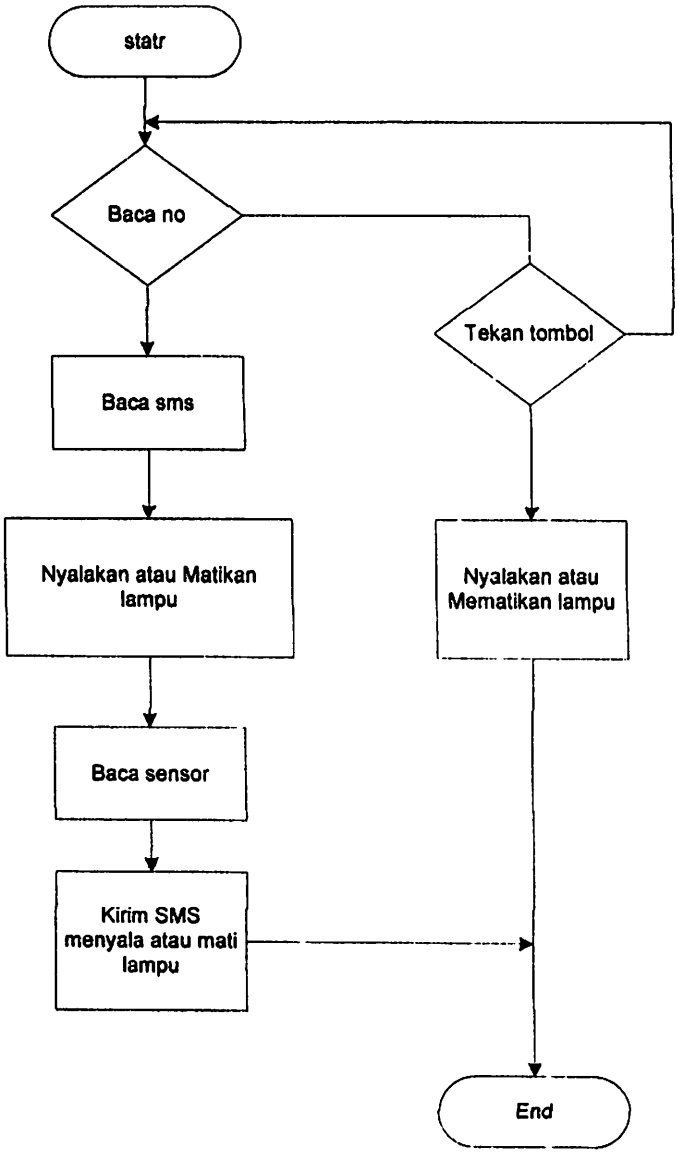
o. Nyalakan atau matikan lampu

p. Dari perintah tombol menyalakan atau matikan lampu

q. End

r. Berakhir

3.3.3.2.2 *Flowchart* program



Gambar 3.7.*Flowchart* Program Utama

BAB IV

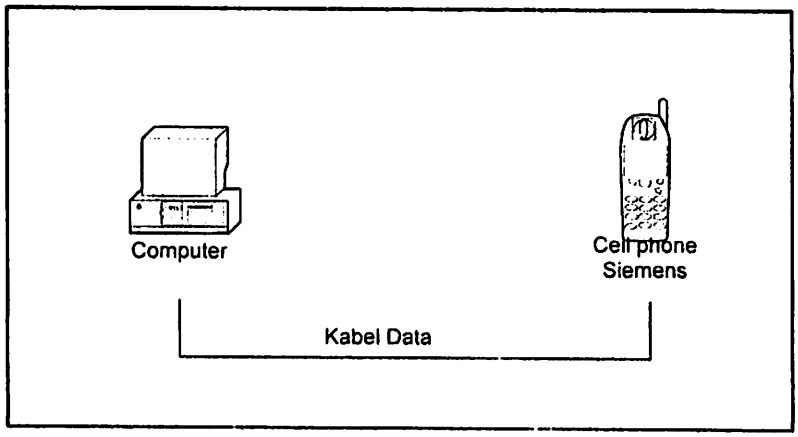
PENGUJIAN ALAT

Setelah perencanaan dan pembuatan sistem maka langkah selanjutnya yaitu melakukan pengujian terhadap sistem yang telah dibuat. Pengujian dilakukan untuk mengetahui apakah sistem yang dibuat dapat berjalan sesuai dengan yang diinginkan.

Pengujian yang dilakukan adalah pengujian terhadap *hardware* dan pengujian sistem secara keseluruhan.

4.1 Pengujian *AT Command* dan format data sms pada telepon seluler.

Untuk menguji fungsi *AT Command* pada telepon seluler dan mengetahui data PDU (Protocol Data Unit) yang dikirim dari telepon seluler pada alat pada telepon seluler pemilik.



Gambar 4.1 Blok diagram pengujian telepon seluler

4.1.1 Langkah pengujiannya adalah sebagai berikut :

a. Hubungkan telepon seluler dengan komputer menggunakan kabel data serial.

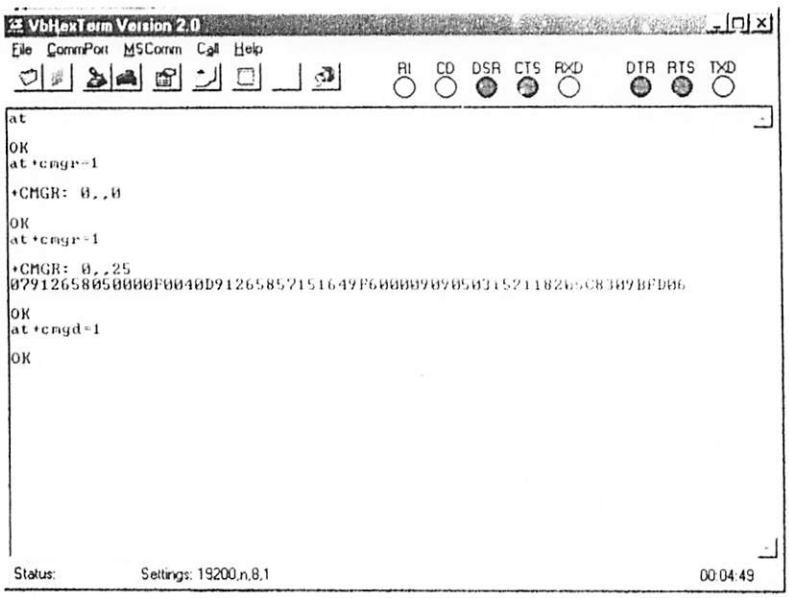
b. Menjalankan program *Hyper Terminal*.

c. Melakukan setting port serial pada program *Hyper Terminal.*

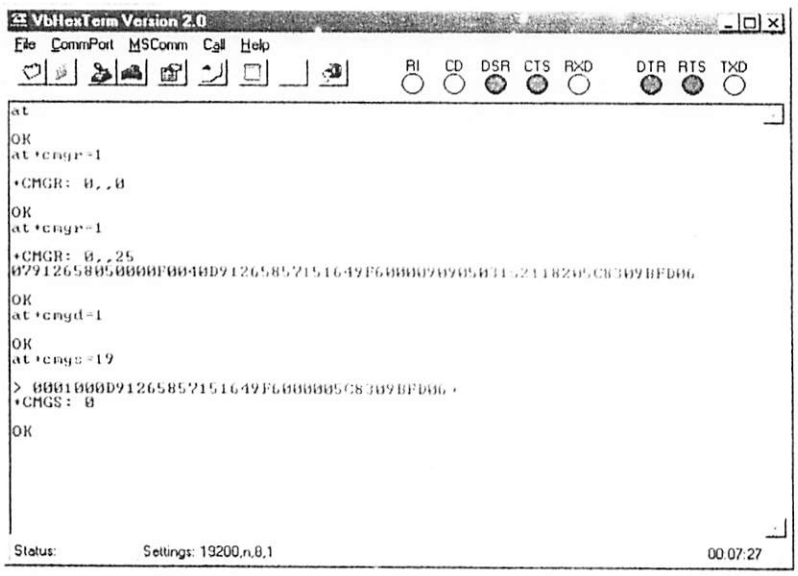
d. Mengetik instruksi AT+CMGC=1 untuk melihat pesan sms di kotak masuk.

e. Mencoba mengirim pesan dngan kata HALLO

4.1.2 Hasil pengujian



Gambar 4.2 Hasil pengujian dengan *AT command*



Gambar 4.3 Hasil pengujian mengirim peasan kata HALLO

4.1.3 Analisa hasil pengujian

Berdasarkan pengujian didapat gambar 4.2 dan 4.3 menunjukan bahwa komunikasi dengan telepon seluler dapat dilakuakan dengan menggunakan intruksi *AT Command* dan *AT Command* merupakan bahasa yang digunakan untuk menjalankan menu-menu pada telepon seluler.

Dari format data PDU yang diterima dapat dilihat ada delapan *header* didalamya yaitu:

1. Nomor *sms-centre*, terdapat tiga *subheader:*
   a) 07 = Jumlah pasangan heksa *sms-centre.*
   b) 91 = Kode Internasional.
   c) 2618010000 = Nomor sms-centre
2. Tipe SMS :

   Untuk SEND tipe sms = 1.Jadi bilangan heksanya adalah 01.
3. Nomer referensi :

   Nomor referensi ini dibiarkan dulu 0, Jadi bilangan heksanya adalah 00.
4. Nomor ponsel penerima, terhadap tiga *subheader :*
   a) OC = Jumlah bilangan desimal nomor pengirim(12 angka).
   b) 81 = Kode Nasional.
5. Bentuk SMS :

   00 = Menandakan data dikirim sebagai sms.
6. Sekema encoding :

   00 = Menandakan skema encoding mengunakan skema 7 bit.
7. Batas waktu validitas :

Agar SMS pasti terkirim sampai ke ponsel penerima sebaiknya tidak memberikan batasan waktu validnya.

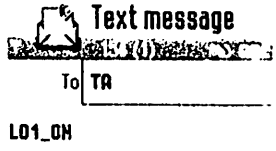
8. Isi SMS :
   a) 0D = jumlah karakter dari data yang dikirim.
   b) 0001000D91265857151649F6000005C8309BFD06 = Dengan kata HALLO

4.2 Pengujian Pengiriman SMS

Pengujian sensor dan pengiriman SMS ini bertujuan untuk mengetahui ketika sensor dalam keadaan aktif, apakah MCU dapat mengirim pesan sesuai dengan sensor yang aktif. Dan dapat menyalakan dan memetikan lampu melalui SMS.

4.2.1 Pengujian Pengiriman SMS Untuk menyalakan Lampu 01

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L01_ON".** Kirim ke no handphone alat.

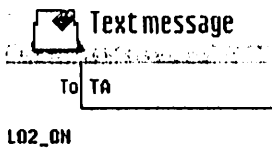


Gambar 4.4 SMS yang dikirim pemilik untuk menyalakan lampu 01

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 1 bekerja lalu lampu 1 menyala.

4.2.2 Pengujian Pengiriman SMS Untuk menyalakan Lampu 02

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L02_ON".** Kirim ke no handphone alat.

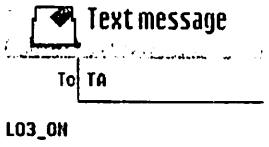


Gambar 4.5 SMS yang dikirim pemilik untuk menyalakan lampu 02

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 2 bekerja lalu lampu 2 menyala.

4.2.3 Pengujian Pengiriman SMS Untuk menyalakan Lampu 03

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L03_ON".** Kirim ke no handphone alat.

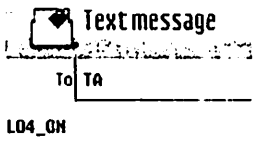


Gambar 4.6 SMS yang dikirim pemilik untuk menyalakan lampu 03

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 3 bekerja lalu lampu 3 menyala.

4.2.4 Pengujian Pengiriman SMS Untuk menyalakan Lampu 04

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L04_ON".** Kirim ke no handphone alat.

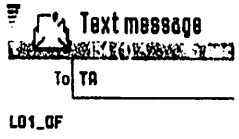


Gambar 4.7 SMS yang dikirim pemilik untuk menyalakan lampu 04

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 4 bekerja lalu lampu 4 menyala.

4.2.5 Pengujian Pengiriman SMS Untuk Memadamkan Lampu 01

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L01_OF".** Kirim ke no *handphone* alat.

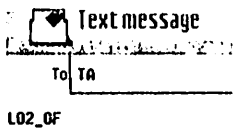


Gambar 4.8 SMS yang dikirim pemilik untuk memadamkan lampu 01

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 1 bekerja lalu lampu 1 Padam.

4.2.6 Pengujian Pengiriman SMS Untuk Memadamkan Lampu 02

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L02_OF".** Kirim ke no *handphone* alat

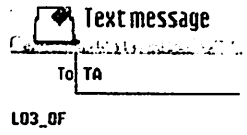


Gambar 4.9 SMS yang dikirim pemilik untuk memadamkan lampu 02

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 2 bekerja lalu lampu 2 Padam.

4.2.7 Pengujian Pengiriman SMS Untuk Memadamkan Lampu 03

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L03_OF".** Kirim ke no *handphone* alat

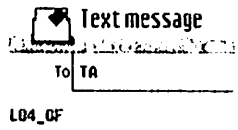


Gambar 4.10 SMS yang dikirim pemilik untuk memadamkan lampu 03

Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 3 bekerja lalu lampu 3 Padam.

4.2.8 Pengujian Pengiriman SMS Untuk Memadamkan Lampu 04

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"L04_OF".** Kirim ke no *handphone* alat

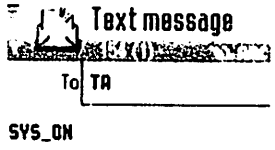


Gambar 4.11 SMS yang dikirim pemilik untuk memadamkan lampu 04

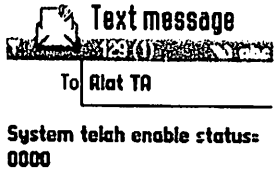
Setelah alat disms dengan format seperti diatas, HP alat akan menerima sms dari HP pemilik dan alat proses relay 4 bekerja lalu lampu 4 Padam.

4.2.9 Pengujian Pengiriman SMS Untuk Menghidupkan Sistem

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"SYS_ON".** Kirim ke no *handphone* alat.



Gambar 4.12 SMS yang dikirim pemilik untuk menghidupkan system

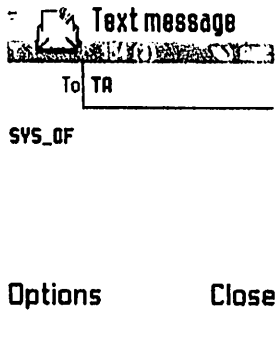


Gambar 4.13 SMS yang dikirim dari alat bahwa system ON

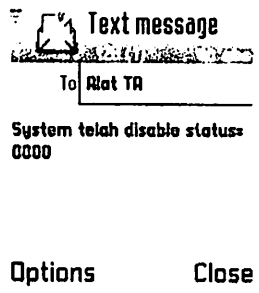
Sesuai dengan kondisi jika sistem hidup, maka akan mengirim SMS **"system telah *enable* status :0000"**. Hal ini menandakan bahwa kondisi system aktif dan hasil ini sesuaia dengan program yang telah dibuat

4.2.10 Pengujian Pengiriman SMS Untuk mematikan Sistem

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"SYS_OF".** Kirim ke no *handphone* alat.



Gambar 4.14 SMS yang dikirim pemilik untuk mematikan system

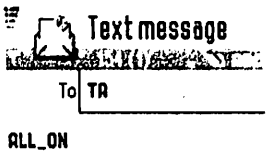


Gambar 4.15 SMS yang dikirim dari alat bahwa system OFF

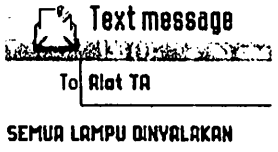
Sesuai dengan kondisi jika system mati, maka akan mengirim SMS **"system telah *disable* status :0000"**. Hal ini menandakan bahwa kondisi system non aktif dan hasil ini sesuaia dengan program yang telah dibuat

4.2.11 Pengujian Pengiriman SMS Untuk Menyalakan semua Lampu

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan "**ALL_ON**". Kirim ke no *handphone* alat.



Gambar 4.16 SMS yang dikirim pemilik untuk menghidupkan semua lampu

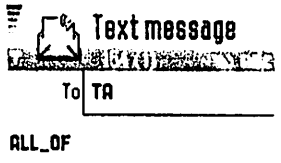


Gambar 4.17 SMS yang dikirim dari alat bahwa semua lampu menyala

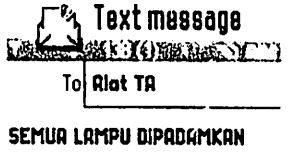
Sesuai dengan kondisi jika semua lampu menyala, maka akan mengirim SMS **"SEMUA LAMPU DINYALAKAN", status :1111"**. Hal ini menandakan bahwa kondisi semua lampu menyala dan hasil ini sesuaia dengan program yang telah dibuat.

4.2.12 Pengujian Pengiriman SMS Untuk Memadamkan semua Lampu

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan "**ALL_OF**". Kirim ke no *handphone* alat.



Gambar 4.18 SMS yang dikirim pemilik untuk mematikan semua lampu

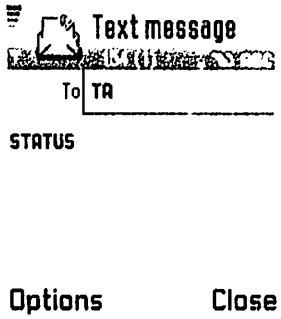


Gambar 4.19 SMS yang dikirim dari alat bahwa semua lampu padam

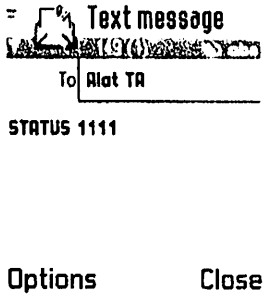
Sesuai dengan kondisi jika semua lampu padam, maka akan mengirim SMS **"SEMUA LAMPU DIPADAMKAN", status : 0000"**. Hal ini menandakan bahwa kondisi semua lampu padam dan hasil ini sesuai dengan program yang telah dibuat.

4.2.13 Pengujian Pengiriman SMS Untuk Mengetahui Keadaan Lampu

Pengetesan ini dilakukan dengan cara mengirimkan SMS dengan mengetikan **"STATUS".** Kirim ke no *handphone* alat.

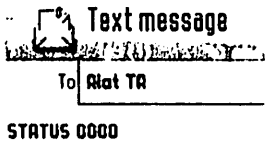


Gambar 4.20 SMS yang dikirim pemilik untuk mengetahui keadaan lampu



Gambar 4.21 SMS yang dikirim dari alat bahwa semua lampu menyala

Sesuai dengan kondisi jika lampu nyala, maka akan mengirim SMS **"STATUS 1111".** Hal ini menandakan bahwa kondisi lampu menyala semu dan hasil ini sesuaia dengan program yang telah dibuat.



Gambar 4.22 SMS yang dikirim dari alat bahwa semua lampu padam

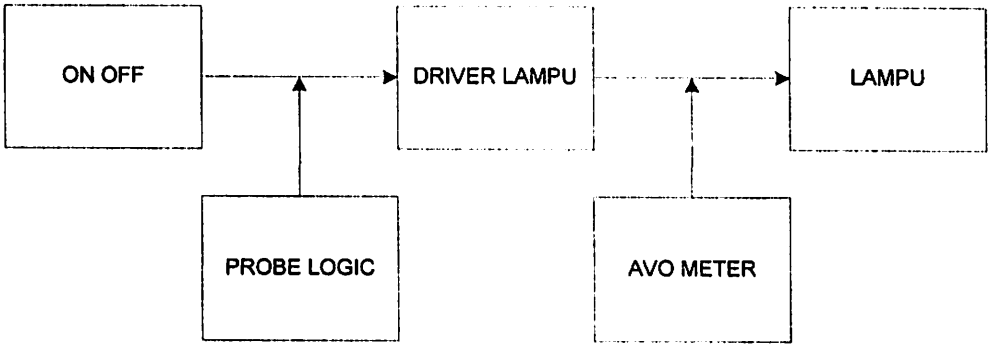
Sesuai dengan kondisi jika lampu padam, maka akan mengirim SMS **"STATUS 0000"**. Hal ini menandakan bahwa kondisi lampu padam semua dan hasil ini sesuaia dengan program yang telah dibuat.

Keterangan : (_) adalah tampilan untuk spasi

4.3 Pengujian Rangkaian Driver Lampu

pengujian ini bertujuan untuk mengetahui apakah *driver* lampu bekerja dengan baik dan sesuai dengan yang direncanakan.

Blok pengujian rangkaian *driver* lampu ditunjukkan dalam gambar 4.17



Gambar 4.23 Pengujian Rangkaian *Driver* Lampu

4.3.1 Langkah pengujianya adalah sebagai berikut :

a. Susun rangkaian seperti dalam blok diagram diatas.

b. Nyalakan saklar ON OFF.

c. Amati kondisi lampu.

d. Mengamati kondisi lampu.

e. Mengamati perubahan logika menggunakan *logic probe*.

f. Ukur menggunakan Avometer.

4.3.2 Hasil Pengujian

Tabel 4.1 Hasil Pengujian Rangkaian *Driver* lampu

Kondisi Lampu	Logika	Keluaran Driver (V)
Menyala	High (1)	3
Padam	Low (0)	2

4.3.2.1 Analisa hasil pengujian

Dari tabel 4.1 terlihat bahwa jika *driver* lampu berlogika tinggi ("1") maka kondisi lampu nyala , demikian sebaliknya jika *driver* lampu berlogika rendah ("0") maka kondisi lampu padam..

4.4 Prosdur Penggunaan Alat

1. Pertama yang dilakukan mengaktifkan sistem alat dengan cara SMS ke no alat 085755298897 dengan format SMS **"SYS ON"**. Alat akan membalas SMS **"System telah enable status 0000"**, status akan sesuai dengan kondisi nyala lampu.
2. Untuk menghidupkan lampu 1 sampai dengan 4

   Menghidupkan lampu 1, SMS ke no alat dengan format SMS **"L01 ON"**

   Menghidupkan lampu 2, SMS ke no alat dengan format SMS **"L02 ON"**

   Menghidupkan lampu 3, SMS ke no alat dengan format SMS **"L03 ON"**

Menghidupkan lampu 4, SMS ke no alat dengan format SMS **"L04 ON"**

3. Lampu di padamkan satu persatu secara manual melalui saklar

   Saklar 1 dipadamkan, alat merespon mengirim SMS ke HP 2 **"Lampu 1 mati"**

   Saklar 2 dipadamkan, alat merespon mengirim SMS ke HP 2 **"Lampu 2 mati"**

   Saklar 3 dipadamkan, alat merespon mengirim SMS ke HP 2 **"Lampu 3 mati"**

   Saklar 4 dipadamkan, alat merespon mengirim SMS ke HP 2 **"Lampu 4 mati"**

4. Untuk memadamkan lampu 1 sampai dengan 4

   Memadamkan lampu 1, SMS ke no alat dengan format SMS **"L01 OF"**

   Memadamkan lampu 2, SMS ke no alat dengan format SMS **"L02 OF"**

   Memadamkan lampu 3, SMS ke no alat dengan format SMS **"L03 OF"**

   Memadamkan lampu 4, SMS ke no alat dengan format SMS **"L04 OF"**

5. Lampu di nyalakan satu persatu secara manual melalui saklar

   Saklar 1 dinyalakan, alat merespon mengirim SMS ke HP 2 **"Lampu 1 hidup"**

   Saklar 2 dinyalakan, alat merespon mengirim SMS ke HP 2 **"Lampu 2 hidup"**

   Saklar 3 dinyalakan, alat merespon mengirim SMS ke HP 2 **"Lampu 3 hidup"**

   Saklar 4 dinyalakan, alat merespon mengirim SMS ke HP 2 **"Lampu 4 hidup"**

6. Semua lampu dinyalakan secara bersamaan melalui SMS

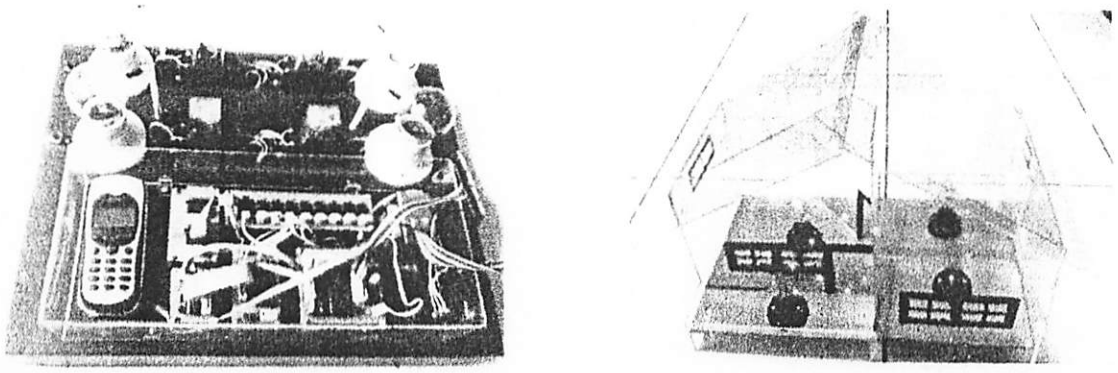
   SMS ke no HP alat dengan format SMS **"ALL ON"**, alat merespon mengirim SMS **"SEMUA LAMPU DINYALAKAN"**

7. Semua lampu dipadamkan secara bersamaan melalui SMS

   SMS ke no HP alat dengan format SMS **"ALL OF"**, alat merespon mengirim SMS **"SEMUA LAMPU DIMATIKAN"**

4.5 Pengujian Alat Keseluruhan

Alat yang digunakan adalah sebuah *handphone* M35, mikrokontroller AT89S8252. Mekanik terdapat sebuah papan kayu. Lampu berjumlah 4 buah. Alat yang terlihat pada gambar berikut:



Gambar 4.24 Gambar Alat Keseluruhan

Pengujian alat dilakukan dengan mengirim SMS ke HP yang berada di rumah. Pesan yang dikirim hanya digunakan untuk menyalakan dan mematikan lampu. Lampu yang dikontrol hanya berjumlah 4 buah .

Misal untuk menyalakan lampu no 1 maka dikirim perintah dengan menulis "L01_ON". Maka yang akan menyala lampu no 1. dan jika untuk memadamkan lampu no 01 maka dikirim SMS dengan format penulisan "L01_OF". Maka yang akan mati lampu 01. Setelah pesan untuk menyalakan dan memadamkan lampu maka berapa detik kemudian akan diterima laporan tentang keadaan lampu.

4.6 Pengujian Waktu Pengiriman

4.6.1 Tujuan

Untuk mengetahui waktu dari data mulai dikirim sampai data diterima oleh ponsel penerima

4.6.2 Langkah pengujian adalah sebagai berikut:

a) Ponsel pemilik akan mengirimkan sebuah SMS kepada ponsel yang ada pada alat hingga mengirimkan SMS tentang keadaan lampu.

b) Aktfkan *system* alat, apabila tidak diaktifkan terlebih dahulu alat tidak berfungsi

c) Amati *stopwatch* mulai pengiriman SMS sampai diterima oleh ponsel penerima.

d) Ulangi prosedur tersebut sampai beberapa kali

4.6.3 Hasil Pengujian

a) **pengujian 1 untuk menyalakan lampu 1 sampai dengan 4**

setelah dilakukan pengiriman sebanyak 3 kali didapat hasil sebagai berikut :

Tabel 4.2 pengiriman SMS Sampai laporan saat menyalakan lampu.

Pemngujian pertama	Format SMS	Waktu (detik)	Status
	L01_ON	09,9	Terkirim
	L02_ON	06,8	Terkirim
	L03_ON	07,5	Terkirim
	L04_ON	07,4	Terkirim
Pengujian Kedua	L01_ON	09,8	Terkirim
	L02_ON	08,9	Terkirim
	L03_ON	10,3	Terkirim
	L04_ON	09,5	Terkirim

Pengujian Ketiga	L01_ON	08,7	Terkirim
	L02_ON	11,6	Terkirim
	L03_ON	07,8	Terkirim
	L04_ON	08,5	Terkirim

b) **pengujian 1 untuk memadamkan lampu 1 sampai dengan 4**

setelah dilakukan pengiriman sebanyak 3 kali didapat hasil sebagai berikut :

Tabel 4.3 pengiriman SMS Sampai laporan saat memadamkan lampu

Pemngujian pertama	**Format SMS**	**Waktu (detik)**	**Status**
	L01_O	11,2	Terkirim
	L02_OF	09,8	Terkirim
	L03_OF	08,5	Terkirim
	L04_OF	09,9	Terkirim
Pengujian Kedua	L01_OF	12,4	Terkirim
	L02_OF	09,5	Terkirim
	L03_OF	07,3	Terkirim
	L04_OF	08,5	Terkirim
Pengujian Ketiga	L01_OF	09.6	Terkirim
	L02_OF	17,9	Terkirim
	L03_OF	08,6	Terkirim
	L04_OF	09,4	Terkirim

4.6.4 Analisis Hasil Pengujian

Dari hasil table 4.2 dan 4.3 pengujian 1 didapat waktu rata-rata sebesar 8,73 detik. Dari hasil pengujian waktu rata-rata sebesar 8,02 detik. Dari hasil pengujian, pada kendala pengujian ini yang tidak dapat dipastikan adalah gangguan terhadap jaringan operator jika ada SMS jadi tertunda dan alat tidak bisa langsung bekerja

4.5.5 Pembahasan

Dalam pengujian ini membahas waktu pengiriman pesan atau mengirimkan perintah terhadap alat karena untuk menghidupkan lampu bisa juga secara manual melalui saklar dan melalui pesan singkat

BAB V

PENUTUP

5.1 KESIMPULAN

Dari proses perencanaan sampai dengan proses pengujian pada sistem ini ,dapat diambil beberapa kesimpulan diantaranya :

1. Semua proses dikendalikan dengan mikrokontroler AT 89S8252 yang dilaporkan melalui SMS dengan media telepon genggam. Dengan format pengiriman SMS untuk menyalakan lampu dengan mengetik "L01_ON, L02_ON, L03_ON, dan L04_ON. Dan jika memadamkan lampu ketik "L01_OF, L02_OF, L03_OF, dan L04_OF. Didalam mengetikkan ini hanya menggunakan tulisan OF (F nya hanya satu) karena untuk menyamakan jumlah format perintah yang lain. Didalam alat ini menggunakkan type HP M55.
2. Pada tempat atau wadah alat tidak menggunakan miniatur yang berbentuk rumah tetapi balok kayu yang berbentuk persegi panjang. Bermaksud untuk mempermudah proses perakitan rangkaian dan lampu menggunakan 4 buah, sebenarnya didalam rangkaian bisa digunakan lampu sebanyak 8 buah tapi demi keefisiensi hanya menggunakan 4 buah lampu.
3. Input tegangan sebesar 4,93 V ke mikrokontroler disisi RS 232 level tegangan sebesar 12V telah syarat level tegangan RS 232 yaitu antara + 15V hingga +3V untuk logika "0" dan -15 hingga -5V untuk logika "1".dari hasil pengujian 1 didapat waktu rata-rata pengiriman 08,89 detik. Dari hasil pengujian 2 waktu rata-rata pengiriman sebesar 09,38 detik.

SARAN

Untuk mengembangkan sistem yang telah dibuat ini,maka penyusun mencoba memberikan saran sebagai berikut:

1. Sistem dapat dilengkapi dengan password agar lebih menarik
2. Ponsel sebaiknya dilengkapi dengan charger karena system digunakan terus menerus.
3. Alat ini bisa menggunakan saklar manual yang pada umumnya tidak menggunakan *push button.*
4. Alat ini agar lebih bagus lagi jika menggunakan sensor arus bukan sensor cahaya


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

DAFTAR PUSTAKA

1. Ibrahim, Kf, (1996). *Teknik Digital*, Andi Offset, Yogyakarta.
2. Malvino, (1992). *Prinsip-prinsip Elektronike*, Edisi Kedua, Erlangga, Jakarta.
3. Nalwan, Paulus Andi, (2003). *Teknik Antar Muka Dan Pemrograman Mikrokontroller AT89S51*, PT. Elex Media Komputindo, Jakarta.
4. Prasetyono, Dwi Sunar, (2003). *Belajar Sistem Cepat Elektronika.*
5. S, Wasito, (2001). *Vademekum Elektronika*, Edisi II, PT. Gramedia Pustaka Utama, Jakarta.
6. www.atmel.com


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

INSTITUT TEKNOLOGI NASIONAL MALANG
PROGRAM STUDI TEKNIK ELEKTRO DIPLOMA TIGA (D-III)
MALANG

UT - 2

KESEDIAAN PEMBIMBING
TUGAS AKHIR

Sesuai permohonan dari mahasiswa :

N a m a : David Ariyanto

No. Mahasiswa : 0452219

Program Studi : Teknik Elektro D-III

Judul Tugas Akhir : Pembuatan Alat Pengontrolan Lampu melalui SMS berbasis MK AT89C51

Bahwa kami bersedia membimbing Tugas Akhir dari mahasiswa tersebut.

Jangka waktu penyelesaian Tugas Akhir selama 4 (empat) bulan mulai tanggal 07/01/2009 s/d 07/05/2009 dan apabila dalam jangka waktu tersebut belum selesai maka tugas akhir tersebut dinyatakan **GUGUR**

Malang, 07 Januari 2009

Dosen Pembimbing;

Bambang Prio Hartono,ST,MT

NIP. 1028400082

Nb :
Setelah disetujui agar formulir ini diserahkan Mahasiswa yang bersangkutan kepada sekretaris Program Studi Teknik Elektro D-III

LEMBAR ASISTENSI BIBINGAN TUGAS AKHIR

Nama : DAVID ARIYANTO
NIM : 0552007
Judul : Perencanaan dan pembuatan alat pengendali lampu melalui sms berbasis mikrokontroler AT89S8252
Dosen Pembimbing : Bambang Prio Hartono,ST,MT

No	Tanggal	Asistensi	Paraf
1		Acc BAB I. Abstrak. Pendahuluan.	
2		Acc BAB II Aturan penulisan.	
3		Acc BAB III	
4		BAB IV. BAB V. Pengujian.	
5		Acc maju	

Malang,Agustus 2009
Dosen pembimbing

Bambang Prio Hartono,ST,MT
NIP . 1028400082

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
PROGRAM STUDI TEKNIK ENERGI LISTRIK DIII
MALANG

LEMBAR PERBAIKAN TUGAS AKHIR

NAMA : DAVID ARIYANTO

NIM : 0552007

JURUSAN : TEKNIK ELEKTRO D-III

PROGRAM STUDI : TEKNIK ENERGI LISTRIK

HARI / TANGGAL : JUM'AT 2–10-2009

No	Revisi	TTD
1.	Prosedur Penggunaan Alat	
2.	Spesifikasi Peralatan	
3.	Miniature Rumah	

Telah diperiksa / Disetujui

Penguji

Ir. H. Taufik Hidayat, MT
NIP. Y 101 8700 151

L

A

M

P

I

R

A

N

SIEMENS

SIEMENS M35

BACK TO : [illegible] | PONSEL CDMA | PDA / PDA PHONE | PONSEL BARU GSM | PONSEL BARU CDMA
ACCESS CASES PONSEL
[illegible]
PONSEL 3G | PDA PHONE CDMA | POWER BANK | [illegible]

[illegible]

DAFTAR TOKO HANDPHONE (COUNTER HP)

ACCESS CASES SIEMENS : [illegible]

Cek Ponsel & Accessories

NEW..... ***BUY FROM OUR***

[illegible]
- [illegible]
- BARU, [illegible]
- [illegible]

HARGA SECOND
[illegible]
- KUNING, [illegible]
- [illegible] LIHAT TOKO & ARSIP IKLAN

[illegible]
-

SPESIFIKASI

[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	• [illegible] • [illegible] • [illegible] • [illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	• [illegible] • [illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	[illegible]
[illegible]	• [illegible] • [illegible] • [illegible] • [illegible] • [illegible] • [illegible]

```
        org     00h
        ljmp    init
;
        org     23h
        clr     ES
        jnb     RI,$
        clr     RI
        mov     A,SBUF
        mov     R7,A
        setb    ES
        reti
;
        Odta    Bit P3.2
        Oclk    Bit P3.3
        Donf    Bit P3.4
        Ilod    Bit P3.5
        Iclk    Bit P3.6
        Idta    Bit P3.7
        Rest    Bit P2.6
        Enbl    Bit P2.7
        Eecn    Data 96h            ; data eeprom control
        Eeen    Equ 00001000b       ; bit eeprom enable (read)
        Eewr    Equ 00010000b       ; bit eeprom write
        Wtdg    Equ 00000010b       ; bit watchdog
        Sttc    Bit 20h.0           ; status command sms
        Chr0    Equ 30h
        Chr1    Equ 31h
        Chr2    Equ 32h
        Chr3    Equ 33h
        Chr4    Equ 34h
        Chr5    Equ 35h
        Chr6    Equ 36h
        Chr7    Equ 37h
        Chr8    Equ 38h
        Chr9    Equ 39h
        ChrA    Equ 3Ah
        ChrB    Equ 3Bh
        ChrC    Equ 3Ch
        ChrD    Equ 3Dh
        ChrE    Equ 3Eh
        ChrF    Equ 3Fh

        Nhp0    Equ 40h
        Nhp1    Equ 41h
        Nhp2    Equ 42h
        Nhp3    Equ 43h
        Nhp4    Equ 44h
        Nhp5    Equ 45h
        Nhp6    Equ 46h
        Nhp7    Equ 47h
        Nhp8    Equ 48h
        Nhp9    Equ 49h
        NhpA    Equ 4Ah
        NhpB    Equ 4Bh

        ChrH    Equ 50h             ; character HP
        Cnsm    Equ 51h
        Cndt    Equ 52h
        Cntr    Equ 53h
        Tmo0    Equ 54h
        Tmo1    Equ 55h
        Din1    Equ 56h             ; data input lampu
        Dint    Equ 57h             ; data input tombol
        Dlmp    Equ 58h             ; data lampu
```

```
        Ssys    Equ 59h             ; status system
        Char    Equ 5Ah
        Bufr    Equ 5Bh
        Dly0    Equ 5Ch
        Dly1    Equ 5Dh
        Dly2    Equ 5Eh
;
init:   lcall   lcd_in
;
        mov     DPTR,#teslmp
        lcall   line1
        mov     Char,#16
        lcall   tulis
        lcall   clrlmp
;
        lcall   led_on
        lcall   action
;
        lcall   led_of
        lcall   rd_mem
        mov     A,Dlmp
        lcall   drvlmp
;
        lcall   srl_in              ; inisialisasi serial
        lcall   noecho              ; komunikasi no-echo
        lcall   dsminc              ; display incoming sms
        lcall   rstcmd
;
mulai:  mov     DPTR,#nama
        lcall   line1
        mov     Char,#16
        lcall   tulis
        mov     DPTR,#nim
        lcall   line2
        mov     Char,#16
        lcall   tulis
        lcall   delay2
        mov     DPTR,#jur
        lcall   line1
        mov     Char,#16
        lcall   tulis
        mov     DPTR,#univ
        lcall   line2
        mov     Char,#16
        lcall   tulis
        lcall   delay2
;
oprsys: lcall   lcdclr
opsys0: mov     A,Ssys
        jnz     opsys1
        mov     DPTR,#tpsyof
        sjmp    opsys2
opsys1: mov     DPTR,#tpsyon
opsys2: lcall   line1
        mov     Char,#16
        lcall   tulis
;
cek00:  lcall   bc_inp
        mov     A,Dint
        jz      cek02
        lcall   scntbl
cek01:  lcall   delay1
        lcall   bc_inp
        mov     A,Dint
```

```
        jnz     cek01
cek02:  cjne    R7,#0FFh,cek03
        ljmp    cek04
cek03:  lcall   bc_sms
        lcall   rstcmd
cek04:  lcall   delay1
        ljmp    cek00
;
scntbl: mov     A,Dint
        cjne    A,#00000001b,scb102
        mov     A,Dlmp
scb100: jb      Acc.0,scb101
        orl     Dlmp,#00000001b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp001
        ljmp    scb123
scb101: jnb     Acc.0,scb100
        xrl     Dlmp,#00000001b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp001
        ljmp    scb123
;
scb102: mov     A,Dint
        cjne    A,#00000010b,scb105
        mov     A,Dlmp
scb103: jb      Acc.1,scb104
        orl     Dlmp,#00000010b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp002
        ljmp    scb123
scb104: jnb     Acc.1,scb103
        xrl     Dlmp,#00000010b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp002
        ljmp    scb123
;
scb105: mov     A,Dint
        cjne    A,#00000100b,scb108
        mov     A,Dlmp
scb106: jb      Acc.2,scb107
        orl     Dlmp,#00000100b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp003
        ljmp    scb123
scb107: jnb     Acc.2,scb106
        xrl     Dlmp,#00000100b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp003
        ljmp    scb123
;
scb108: mov     A,Dint
        cjne    A,#00001000b,scb111
```

```
        mov     A,Dlmp
scbl09: jb      Acc.3,scbl10
        orl     Dlmp,#00001000b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp004
        ljmp    scbl23
scbl10: jnb     Acc.3,scbl09
        xrl     Dlmp,#00001000b
        lcall   wr_mem
        mov     A,Dlmp
        lcall   drvlmp
        lcall   rsp004
        ljmp    scbl23
;
scbl11:
scbl23: ret
;
rsp001: mov     A,Ssys
        jz      rsp201
        lcall   delay1
        lcall   bc_inp
        mov     A,Dinl
        jb      Acc.0,rsp101
        lcall   klm1of                  ; kirim sms lampu mati
        ljmp    rsp201
rsp101: jnb     Acc.0,rsp201
        lcall   klm1on                  ; kirim sms lampu hidup
rsp201: ret
;
rsp002: mov     A,Ssys
        jz      rsp202
        lcall   delay1
        lcall   bc_inp
        mov     A,Dinl
        jb      Acc.1,rsp102
        lcall   klm2of                  ; kirim sms lampu mati
        ljmp    rsp202
rsp102: jnb     Acc.1,rsp202
        lcall   klm2on                  ; kirim sms lampu hidup
rsp202: ret
;
rsp003: mov     A,Ssys
        jz      rsp203
        lcall   delay1
        lcall   bc_inp
        mov     A,Dinl
        jb      Acc.2,rsp103
        lcall   klm3of                  ; kirim sms lampu mati
        ljmp    rsp203
rsp103: jnb     Acc.2,rsp203
        lcall   klm3on                  ; kirim sms lampu hidup
rsp203: ret
;
rsp004: mov     A,Ssys
        jz      rsp204
        lcall   delay1
        lcall   bc_inp
        mov     A,Dinl
        jb      Acc.3,rsp104
        lcall   klm4of                  ; kirim sms lampu mati
        ljmp    rsp204
rsp104: jnb     Acc.3,rsp204
```

```
        lcall   klm4on                  ; kirim sms lampu hidup
rsp204: ret
;
rbhsts: lcall   led_of
        lcall   rd_mem
        mov     DPTR,#tprbst
        lcall   line1
        mov     Char,#16
        lcall   tulis
rbsts0: mov     A,Ssys
        cjne    A,#0,rbsts1
        mov     DPTR,#tpsyof
rbsts1: cjne    A,#1,rbsts2
        mov     DPTR,#tpsyon
rbsts2: lcall   line2
        mov     Char,#16
        lcall   tulis
        lcall   tg_lps
rbsts3: lcall   scnkpd
        cjne    R0,#11,rbsts4
        lcall   rstcmd
        lcall   led_on
        mov     SP,#07h
        ljmp    oprsys
rbsts4: cjne    R0,#12,rbsts6
        mov     A,Ssys
        cjne    A,#0,rbsts5
        mov     Ssys,#1
        lcall   wr_mem
        ljmp    rbsts0
rbsts5: cjne    A,#1,rbsts6
        mov     Ssys,#0
        lcall   wr_mem
        ljmp    rbsts0
rbsts6: cjne    R0,#16,rbsts3
;
isinhp: lcall   lcdclr
        mov     DPTR,#tpnohp
        lcall   line1
        mov     Char,#16
        lcall   tulis
isnhp0: mov     DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        mov     A,Nhp0
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp1
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp2
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp3
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp4
        lcall   cknhpn
```

```
        lcall   wr_chr
        mov     A,Nhp5
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp6
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp7
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp8
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp9
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpA
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpB
        lcall   cknhpn
        lcall   wr_chr
        mov     P0,#0D0h
        lcall   w_ins
        lcall   tg_lps
isnhp1: lcall   scnkpd
        cjne    R0,#11,isnhp2
        lcall   rstcmd
        lcall   led_on
        mov     SP,#07h
        ljmp    oprsys
isnhp2: cjne    R0,#12,isnhp3
        ljmp    isnhp4
isnhp3: cjne    R0,#15,isnhp1
        ljmp    rbhsts
isnhp4: mov     DPTR,#kosong
        lcall   line2
        mov     Char,#16
        lcall   tulis
        lcall   rsnohp
        lcall   tg_lps
        mov     DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        lcall   tginhp
        mov     Nhp0,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp1,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp2,R0
        mov     A,R0
        lcall   wr_chr
```

```
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp3,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp4,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp5,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp6,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp7,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp8,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp9,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpA,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpB,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
isnhp5: lcall   wr_mem
        ljmp    isnhp0
;
tginhp: lcall   scnkpd
tinhp0: cjne    R0,#16,tinhp1
        ljmp    tginhp
tinhp1: cjne    R0,#15,tinhp2
        ljmp    tginhp
tinhp2: cjne    R0,#14,tinhp3
        ljmp    tginhp
tinhp3: cjne    R0,#13,tinhp4
        ljmp    tginhp
tinhp4: cjne    R0,#12,tinhp5
        ljmp    isnhp5
tinhp5: cjne    R0,#11,tinhp6
        lcall   rstcmd
        lcall   led_on
        mov     SP,#07h
```

```
        ljmp    oprsys
tinhp6: cjne    R0,#10,tinhp7
        ljmp    tginhp
tinhp7: ret
;
rsnohp: mov     Nhp0,#00Fh
        mov     Nhp1,#00Fh
        mov     Nhp2,#00Fh
        mov     Nhp3,#00Fh
        mov     Nhp4,#00Fh
        mov     Nhp5,#00Fh
        mov     Nhp6,#00Fh
        mov     Nhp7,#00Fh
        mov     Nhp8,#00Fh
        mov     Nhp9,#00Fh
        mov     NhpA,#00Fh
        mov     NhpB,#00Fh
        ret
;
cknhpn: cjne    A,#00Fh,cnhpn           ; cek angka no hp
        mov     A,#16                   ; if F then chr ' '
cnhpn:  ret
;
bc_sms: lcall   bcnohp                  ;\
        jnb     Sttc,bcsm00             ; | baca no HP
        lcall   bccmnd                  ; | cek status
        lcall   hpssms                  ; | 0=hapus sms, 1=baca
command
        ljmp    bcsm01                  ; | hapus sms
bcsm00: lcall   hpssms                  ; | cek command
        ret                     ;/
;
bcsm01: mov     DPTR,#cmnd00            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm02             ; | cek command
        orl     Dlmp,#00000001b           ; |
        lcall   wr_mem                  ; |
        mov     A,Dlmp                  ; | simpan memory
        lcall   drvlmp                  ; |
        lcall   rsp001                  ; | respon
        ret                     ;/
;
bcsm02: mov     DPTR,#cmnd01            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm03             ; | cek command
        mov     A,Dlmp                  ; |
        jnb     Acc.0,lmpt00            ; | lampu 01 on, matikan
        xrl     Dlmp,#00000001b           ; | lampu 01 of, biarkan
        lcall   wr_mem                  ; |
        mov     A,Dlmp                  ; | simpan memory
        lcall   drvlmp                  ; |
        lcall   rsp001                  ; | respon
lmpt00: ret                     ;/
;
bcsm03: mov     DPTR,#cmnd02            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm04             ; | cek command
        orl     Dlmp,#00000010b           ; |
        lcall   wr_mem                  ; |
        mov     A,Dlmp                  ; | simpan memory
        lcall   drvlmp                  ; |
        lcall   rsp002                  ; | respon
        ret                     ;/
;
```

```
bcsm04: mov     DPTR,#cmnd03        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm05         ; | cek command
        mov     A,Dlmp              ; |
        jnb     Acc.1,lmpt01        ; | lampu 02 on, matikan
        xrl     Dlmp,#00000010b           ; | lampu 02 of, biarkan
        lcall   wr_mem              ; |
        mov     A,Dlmp              ; | simpan memory
        lcall   drvlmp              ; |
        lcall   rsp002              ; | respon
lmpt01: ret                 ;/
;
bcsm05: mov     DPTR,#cmnd04        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm06         ; | cek command
        orl     Dlmp,#00000100b           ; |
        lcall   wr_mem              ; |
        mov     A,Dlmp              ; | simpan memory
        lcall   drvlmp              ; |
        lcall   rsp003              ; | respon
        ret                 ;/
;
bcsm06: mov     DPTR,#cmnd05        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm07         ; | cek command
        mov     A,Dlmp              ; |
        jnb     Acc.2,lmpt02        ; | lampu 03 on, matikan
        xrl     Dlmp,#00000100b           ; | lampu 03 of, biarkan
        lcall   wr_mem              ; |
        mov     A,Dlmp              ; | simpan memory
        lcall   drvlmp              ; |
        lcall   rsp003              ; | respon
lmpt02: ret                 ;/
;
bcsm07: mov     DPTR,#cmnd06        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm08         ; | cek command
        orl     Dlmp,#00001000b           ; |
        lcall   wr_mem              ; |
        mov     A,Dlmp              ; | simpan memory
        lcall   drvlmp              ; |
        lcall   rsp004              ; | respon
        ret                 ;/
;
bcsm08: mov     DPTR,#cmnd07        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm17         ; | cek command
        mov     A,Dlmp              ; |
        jnb     Acc.3,lmpt03        ; | lampu 04 on, matikan
        xrl     Dlmp,#00001000b           ; | lampu 04 of, biarkan
        lcall   wr_mem              ; |
        mov     A,Dlmp              ; | simpan memory
        lcall   drvlmp              ; |
        lcall   rsp004              ; | respon
lmpt03: ret                 ;/
;
bcsm17: mov     DPTR,#cmnd16        ;\
        lcall   ckcmnd              ; | ambil data command
        jnb     Sttc,bcsm18         ; | cek command
        mov     DPTR,#tpsyon        ; |
        lcall   line1               ; |
        mov     Char,#16            ; | LCD system on
        lcall   tulis               ; |
        mov     Ssys,#1             ; |
```

```
        lcall   wr_mem                  ; | system on
        lcall   delay1                  ; |
        lcall   ksenst                  ; | kirim system enable, stts:
        ret                     ;/
;
bcsm18: mov     DPTR,#cmnd17            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm19             ; | cek command
        mov     DPTR,#tpsyof            ; |
        lcall   line1                   ; |
        mov     Char,#16                ; | LCD system off
        lcall   tulis                   ; |
        mov     Ssys,#0                 ; |
        lcall   wr_mem                  ; | system off
        lcall   delay1                  ; |
        lcall   ksdsst                  ; | kirim system disable,
stts:
        ret                     ;/
;
bcsm19: mov     DPTR,#cmnd18            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm20             ; | cek command
        mov     Dlmp,#11111111b           ; |
        lcall   wr_mem                  ; |
        mov     A,Dlmp                  ; | simpan memory
        lcall   drvlmp                  ; |
        lcall   delay1                  ; |
        lcall   kslhst                  ; | kirim semua lampu
dihidupkan, stts
        ret                     ;/
;
bcsm20: mov     DPTR,#cmnd19            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm21             ; | cek command
        mov     Dlmp,#00000000b           ; |
        lcall   wr_mem                  ; |
        mov     A,Dlmp                  ; | simpan memory
        lcall   drvlmp                  ; |
        lcall   delay1                  ; |
        lcall   kslmst                  ; | kirim semua lampu
dimatikan, stts
        ret                     ;/
;
bcsm21: mov     DPTR,#cmnd20            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsm22             ; | cek command
        lcall   delay1                  ; |
        lcall   krstts                  ; | kirim status
        ret                     ;/
;
bcsm22: ret
;
bcnohp: mov     DPTR,#smsred            ;\
        mov     ChrH,#9                 ; |
        lcall   kr_ins                  ; | baca no hp
        mov     Cntr,#39                ; | header sms + service
center
        lcall   bc_fbk                  ; | dihilangkan
bcnolp: lcall   bc_srl                  ; |
        djnz    Cntr,bcnolp             ; |
        lcall   bc_dta                  ;/
;        lcall   tlsdta                 ; optional
;
cknohp: mov     DPTR,#angka             ;\
```

```
        mov     Cntr,#12                ; | jumlah no hp = 12
        mov     Cnsm,#0                 ; | reset counter sms
        mov     Cndt,#0                 ; | reset counter no hp
cknhp0: lcall   Dtasms                  ; | ambil data counter sms
        lcall   Dtanhp                  ; | ambil data counter no hp
        clr     C               ; |
        subb    A,B                     ; | samakan
        cjne    A,#0,cknhp1             ; | jika sama
        inc     Cnsm                    ; | ambil data counter sms
selanjutnya
        inc     Cndt                    ; | ambil data counter no hp
selajnutnya
        djnz    Cntr,cknhp0             ; | cek sampai no hp habis
        setb    Sttc                    ; | set status = 1
        ljmp    cknhp2                  ; | jika tidak
cknhp1: clr     Sttc                    ; | set status = 0
cknhp2: ret                     ;/
;
dtanhp: mov     A,Cndt
        cjne    A,#00,dnhp00
        mov     A,Nhp1
        ljmp    dnhp11
dnhp00: cjne    A,#01,dnhp01
        mov     A,Nhp0
        ljmp    dnhp11
dnhp01: cjne    A,#02,dnhp02
        mov     A,Nhp3
        ljmp    dnhp11
dnhp02: cjne    A,#03,dnhp03
        mov     A,Nhp2
        ljmp    dnhp11
dnhp03: cjne    A,#04,dnhp04
        mov     A,Nhp5
        ljmp    dnhp11
dnhp04: cjne    A,#05,dnhp05
        mov     A,Nhp4
        ljmp    dnhp11
dnhp05: cjne    A,#06,dnhp06
        mov     A,Nhp7
        ljmp    dnhp11
dnhp06: cjne    A,#07,dnhp07
        mov     A,Nhp6
        ljmp    dnhp11
dnhp07: cjne    A,#08,dnhp08
        mov     A,Nhp9
        ljmp    dnhp11
dnhp08: cjne    A,#09,dnhp09
        mov     A,Nhp8
        ljmp    dnhp11
dnhp09: cjne    A,#10,dnhp10
        mov     A,NhpB
        ljmp    dnhp11
dnhp10: cjne    A,#11,dnhp12
        mov     A,NhpA
dnhp11: cjne    A,#00Fh,dnhp12
        mov     A,#15
dnhp12: movc    A,@A+DPTR
        ret
;
bccmnd: mov     DPTR,#smsred            ;\
        mov     ChrH,#9                 ; |
        lcall   kr_ins                  ; | baca command
        mov     Cntr,#71                ; | header sms + service
center
```

```
        lcall   bc_fbk              ; | + no HP
bccmlp: lcall   bc_srl              ; | dihilangkan
        djnz    Cntr,bccmlp         ; |
        lcall   bc_dta              ; |
;        lcall   tlsdta             ; optional
        ret                     ;/
;
ckcmnd: mov     Cntr,#12            ;\
        mov     Cnsm,#0             ; | jumlah command
        mov     Cndt,#0             ; | reset counter sms
ckcnd0: lcall   Dtasms              ; | reset counter command
        lcall   Dtacmd              ; | ambil data counter sms
        clr     C               ; | ambil data counter command
        subb    A,B                 ; | samakan
        cjne    A,#0,ckcnd1         ; | jika sama
        inc     Cnsm                ; | ambil data counter sms
selanjutnya
        inc     Cndt                ; | ambil data counter command
selanjutnya
        djnz    Cntr,ckcnd0         ; | cek sampai command habis
        setb    Sttc                ; | set status = 1
        ljmp    ckcnd2              ; | jika tidak
ckcnd1: clr     Sttc                ; | set status = 0
ckcnd2: ret                     ;/
;
dtacmd: mov     A,Cndt              ;\
        movc    A,@A+DPTR           ; | ambil data pointer yang ke
cndt
        ret                     ;/
;
dtasms: clr     C               ;\
        mov     A,Cnsm              ; |
dtsm00: cjne    A,#00,dtsm01        ; |
        mov     B,Chr0              ; |
dtsm01: cjne    A,#01,dtsm02        ; |
        mov     B,Chr1              ; |
dtsm02: cjne    A,#02,dtsm03        ; |
        mov     B,Chr2              ; |
dtsm03: cjne    A,#03,dtsm04        ; |
        mov     B,Chr3              ; |
dtsm04: cjne    A,#04,dtsm05        ; |
        mov     B,Chr4              ; |
dtsm05: cjne    A,#05,dtsm06        ; |
        mov     B,Chr5              ; |
dtsm06: cjne    A,#06,dtsm07        ; |
        mov     B,Chr6              ; |
dtsm07: cjne    A,#07,dtsm08        ; |
        mov     B,Chr7              ; | samakan data counter
dtsm08: cjne    A,#08,dtsm09        ; | dengan character
        mov     B,Chr8              ; |
dtsm09: cjne    A,#09,dtsm0A        ; |
        mov     B,Chr9              ; |
dtsm0A: cjne    A,#10,dtsm0B        ; |
        mov     B,ChrA              ; |
dtsm0B: cjne    A,#11,dtsm0C        ; |
        mov     B,ChrB              ; |
dtsm0C: cjne    A,#12,dtsm0D        ; |
        mov     B,ChrC              ; |
dtsm0D: cjne    A,#13,dtsm0E        ; |
        mov     B,ChrD              ; |
dtsm0E: cjne    A,#14,dtsm0F        ; |
        mov     B,ChrE              ; |
dtsm0F: cjne    A,#15,dtsmsx        ; |
        mov     B,ChrF              ; |
```

```
dtsmsx: ret                     ;/
;
bc_dta: lcall   bc_srl              ;\
        mov     Chr0,A              ; |
        lcall   bc_srl              ; |
        mov     Chr1,A              ; |
        lcall   bc_srl              ; |
        mov     Chr2,A              ; |
        lcall   bc_srl              ; |
        mov     Chr3,A              ; |
        lcall   bc_srl              ; |
        mov     Chr4,A              ; |
        lcall   bc_srl              ; |
        mov     Chr5,A              ; |
        lcall   bc_srl              ; |
        mov     Chr6,A              ; |
        lcall   bc_srl              ; |
        mov     Chr7,A              ; |
        lcall   bc_srl              ; |
        mov     Chr8,A              ; | baca serial & simpan
        lcall   bc_srl              ; | ke memory character
        mov     Chr9,A              ; |
        lcall   bc_srl              ; |
        mov     ChrA,A              ; |
        lcall   bc_srl              ; |
        mov     ChrB,A              ; |
        lcall   bc_srl              ; |
        mov     ChrC,A              ; |
        lcall   bc_srl              ; |
        mov     ChrD,A              ; |
        lcall   bc_srl              ; |
        mov     ChrE,A              ; |
        lcall   bc_srl              ; |
        mov     ChrF,A              ; |
        lcall   delay1              ; |
        lcall   delay1              ; |
        lcall   delay1              ; |
        lcall   delay1              ; |
        lcall   delay1              ; |
        ret                     ;/
;
tlsdta: lcall   line2               ;\
        mov     P0,Chr0             ; |
        lcall   w_chr               ; |
        mov     P0,Chr1             ; |
        lcall   w_chr               ; |
        mov     P0,Chr2             ; |
        lcall   w_chr               ; |
        mov     P0,Chr3             ; |
        lcall   w_chr               ; |
        mov     P0,Chr4             ; |
        lcall   w_chr               ; |
        mov     P0,Chr5             ; |
        lcall   w_chr               ; |
        mov     P0,Chr6             ; |
        lcall   w_chr               ; |
        mov     P0,Chr7             ; |
        lcall   w_chr               ; |
        mov     P0,Chr8             ; | tulis character ke lcd
        lcall   w_chr               ; |
        mov     P0,Chr9             ; |
        lcall   w_chr               ; |
        mov     P0,ChrA             ; |
        lcall   w_chr               ; |
```

```
        mov     P0,ChrB             ; |
        lcall   w_chr               ; |
        mov     P0,ChrC             ; |
        lcall   w_chr               ; |
        mov     P0,ChrD             ; |
        lcall   w_chr               ; |
        mov     P0,ChrE             ; |
        lcall   w_chr               ; |
        mov     P0,ChrF             ; |
        lcall   w_chr               ; |
        ret                     ;/
;
kr_ins: clr A                   ;\
      movc  A,@A+DPTR           ; |
      lcall kr_srl                  ; |
      inc   DPTR                ; | kirim instruksi ke hp
      djnz  ChrH,kr_ins         ; | sebagai awal pengiriman data
        mov     A,#0Dh              ; |
      lcall kr_srl                  ; |
        ret                     ;/
;
kr_hlf: clr A                   ;\
      movc  A,@A+DPTR           ; |
      lcall kr_srl                  ; | kirim setengah data olahan
(PDU)
      inc   DPTR                ; | yang sudah pasti ke HP
      djnz  ChrH,kr_hlf         ; |
        ret                     ;/
;
kr_dta: clr A                   ;\
      movc  A,@A+DPTR           ; |
      lcall kr_srl                  ; |
      inc   DPTR                ; | kirim data olahan (PDU)
      djnz  ChrH,kr_dta         ; | ke HP
        mov     A,#26               ; |
      lcall kr_srl                  ; |
        ret                     ;/
;
kr_srl: clr     ES                  ;\
        mov     SBUF,A              ; |
        jnb     TI,$                ; |
        clr     TI                  ; | kirim serial ke hp
        setb    ES                  ; |
        lcall   delay0              ; |
        ret                     ;/
;
noecho: lcall   delay1              ;\
        mov     DPTR,#smseco        ; |
        mov     ChrH,#4             ; | kirim instruksi
        lcall   kr_ins              ; | no-echo ke hp
        lcall   bc_fbk              ; |
        ret                     ;/
;
dsminc: lcall   delay1              ;\
        mov     DPTR,#smsfrm        ; |
        mov     ChrH,#17            ; | kirim instruksi
        lcall   kr_ins              ; | display-incoming-sms ke hp
        lcall   bc_fbk              ; |
        ret                     ;/
;
hpssms: mov     DPTR,#smshps        ;\
        mov     ChrH,#9             ; |
        lcall   kr_ins              ; | hapus sms
        lcall   bc_fbk              ; | pada memory 1
```

```
        lcall   delay1                  ; |
        ret                     ;/
;
bc_fbk: mov     A,#0FFh                 ;\
bc_fb0: cjne    A,#0FFh,bc_fb1                  ; | tunggu feedback
        ljmp    bc_fb0                  ; | data dari hp
bc_fb1: ret                     ;/
;
bc_sr1: mov     A,#0FFh                 ;\
        mov     Tmo0,#10                ; |
tmlp:   mov     Tmo1,#0                 ; | baca serial
bc_sr0: cjne    A,#0FFh,bc_sr1                  ; | tunggu data bukan FF
        djnz    Tmo1,bc_sr0             ; | sebelum time out
        djnz    Tmo0,tmlp               ; |
bc_sr1: ret                     ;/
;
krnohp: mov     A,Nhp1
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp0
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp3
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp2
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp5
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp4
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp7
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp6
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp9
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp8
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,NhpB
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,NhpA
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
```

```
        ret
;
rstcmd: mov     R7,#0FFh
        ret
;
srl_in: lcall   delay1              ;\
        mov     TMOD,#20h           ; |
        mov     TH1,#0FDh           ; |
        mov     SCON,#50h           ; |
        mov     A,#80h              ; | set baudrate
        orl     87h,A               ; | 19200bps
        setb    TR1                 ; |
        setb    EA                  ; |
        setb    ES                  ; |
        ret                 ;/
;
inplod: clr     Iclk                ; input loading
        clr     Ilod
        setb    Iclk
        setb    Ilod
        ret
;
inpclk: clr     Iclk                ; input clock
        setb    Iclk
        ret
;
outclk: setb    Oclk                ; output clock high
        clr     Oclk                ; output clock low
        ret
;
bc_inp: clr     A           ; baca input
        mov     Cntr,#8
        lcall   inplod              ; loading
bcinp0: mov     C,Idta
        RLC     A
        lcall   inpclk              ; shift
        djnz    Cntr,bcinp0
        mov     Dinl,A              ; simpan data input lampu
        clr     A
        mov     Cntr,#8
bcinp1: mov     C,Idta
        RLC     A
        lcall   inpclk              ; shift
        djnz    Cntr,bcinp1
        cpl     A
        mov     Dint,A              ; simpan data input tombol
        ret
;
drvlmp: lcall   led_of
        cpl     A
        mov     Cntr,#8
drlmp:  RLC     A
        mov     Odta,C
        lcall   outclk
        djnz    Cntr,drlmp
        lcall   led_on
        ret
;
action: clr     Odta
        lcall   outclk
        lcall   delay1
        mov     Cntr,#8
        setb    Odta
actio:  lcall   outclk
```

```
        lcall   delay1
        djnz    Cntr,actio
        ret
;
clrlmp: mov     Cntr,#8
        setb    Odta
cllmp:  lcall   outclk
        djnz    Cntr,cllmp
        ret
;
led_on: clr     Donf
        ret
;
led_of: setb    Donf
        ret
;
nilai:  mov     B,#100
        div     AB
        lcall   wr_chr
        mov     A,B
        mov     B,#10
        div     AB
        lcall   wr_chr
        mov     A,B
        lcall   wr_chr
        ret
;
klmlon: mov     DPTR,#smskr0        ; kirim lampu 1 on
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmplon        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                   ; tunggu feedback dari HP
;
klmlof: mov     DPTR,#smskr0        ; kirim lampu 1 of
        mov     ChrH,#10
        lcall   kr_ins              ; amoil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmplof        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                   ; tunggu feedback dari HP
;
klm2on: mov     DPTR,#smskr0        ; kirim lampu 1 on
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
```

```
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp2on        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
klm2of: mov     DPTR,#smskr0        ; kirim lampu 1 of
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp2of        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
klm3on: mov     DPTR,#smskr0        ; kirim lampu 1 on
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp3on        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
klm3of: mov     DPTR,#smskr0        ; kirim lampu 1 of
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp3of        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
klm4on: mov     DPTR,#smskr0        ; kirim lampu 1 on
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
```

```
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp4on        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
klm4of: mov     DPTR,#smskr0        ; kirim lampu 1 of
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#lmp4of        ; tunggu feedback dari HP
        mov     ChrH,#32
        lcall   kr_dta              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
krstts: lcall   scnsts              ; kirim stts :
        lcall   jd_pdu              ; scan hardware
        mov     DPTR,#smskr1        ; rubah character -> PDU
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#smssts        ; tunggu feedback dari HP
        mov     ChrH,#20
        lcall   kr_hlf              ; kirim data setengah yg sudah
pasti
        lcall   kr_pdu              ; kirim data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                       ; tunggu feedback dari HP
;
ksenst: lcall   scnsts              ; kirim system enable, stts:
        lcall   jd_pdu              ; scan hardware
        mov     DPTR,#smskr2        ; rubah character -> PDU
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#syenst        ; tunggu feedback dari HP
        mov     ChrH,#48
        lcall   kr_hlf              ; kirim data setengah yg sudah
pasti
        lcall   kr_pdu              ; kirim data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
```

```
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                     ; tunggu feedback dari HP
;
ksdsst: lcall   scnsts              ; kirim system disable, stts:
        lcall   jd_pdu              ; scan hardware
        mov     DPTR,#smskr2        ; rubah character -> PDU
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#sydsst        ; tunggu feedback dari HP
        mov     ChrH,#48
        lcall   kr_hlf              ; kirim data setengah yg sudah
pasti
        lcall   kr_pdu              ; kirim data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                     ; tunggu feedback dari HP
;
kslhst: lcall   scnsts              ; kirim semua lampu
dihidupkan, stts:
        lcall   jd_pdu              ; scan hardware
        mov     DPTR,#smskr3        ; rubah character -> PDU
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#smlhst        ; tunggu feedback dari HP
        mov     ChrH,#62
        lcall   kr_hlf              ; kirim data setengah yg sudah
pasti
        lcall   kr_pdu              ; kirim data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                     ; tunggu feedback dari HP
;
kslmst: lcall   scnsts              ; kirim semua lampu dimatikan,
stts:
        lcall   jd_pdu              ; scan hardware
        mov     DPTR,#smskr3        ; rubah character -> PDU
        mov     ChrH,#10
        lcall   kr_ins              ; ambil data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri enter
        mov     DPTR,#header
        mov     ChrH,#12            ; kirim header
        lcall   kr_hlf
        lcall   krnohp
        mov     DPTR,#smlmst        ; tunggu feedback dari HP
        mov     ChrH,#62
        lcall   kr_hlf              ; kirim data setengah yg sudah
pasti
        lcall   kr_pdu              ; kirim data PDU
        lcall   bc_fbk              ; jumlah character ATCommand
        lcall   delay1              ; kirim data diakhiri CTRL+Z
        ret                     ; tunggu feedback dari HP
```

```
;
scnsts: lcall   bc_inp
        mov     A,Dinl
        jnb     Acc.0,scst00
        mov     Chr0,#'1'
scst00: jb      Acc.0,scst01
        mov     Chr0,#'0'
scst01: jnb     Acc.1,scst02
        mov     Chr1,#'1'
scst02: jb      Acc.1,scst03
        mov     Chr1,#'0'
scst03: jnb     Acc.2,scst04
        mov     Chr2,#'1'
scst04: jb      Acc.2,scst05
        mov     Chr2,#'0'
scst05: jnb     Acc.3,scst06
        mov     Chr3,#'1'
scst06: jb      Acc.3,scst07
        mov     Chr3,#'0'
scst07: mov     Chr4,#' '
scst08: jnb     Acc.4,scst09
        mov     Chr5,#' '
scst09: jb      Acc.4,scst10
        mov     Chr5,#' '
scst10: jnb     Acc.5,scst11
        mov     Chr6,#' '
scst11: jb      Acc.5,scst12
        mov     Chr6,#' '
scst12: jnb     Acc.6,scst13
        mov     Chr7,#' '
scst13: jb      Acc.6,scst14
        mov     Chr7,#' '
scst14: jnb     Acc.7,scst15
        mov     Chr8,#' '
scst15: jb      Acc.7,scst16
        mov     Chr8,#' '
scst16: mov     Chr9,#' '
        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
;
rd_mem: lcall   strdmm
        mov     DPTR,#00h
        movx    A,@DPTR
        lcall   cknull
        mov     Dlmp,A
        mov     DPTR,#01h
```

```
        movx    A,@DPTR
        lcall   cknull
        mov     Ssys,A
;
        mov     DPTR,#02h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp1,A
        mov     DPTR,#03h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp0,A
        mov     DPTR,#04h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp3,A
        mov     DPTR,#05h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp2,A
        mov     DPTR,#06h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp5,A
        mov     DPTR,#07h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp4,A
        mov     DPTR,#08h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp7,A
        mov     DPTR,#09h
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp6,A
        mov     DPTR,#0Ah
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp9,A
        mov     DPTR,#0Bh
        movx    A,@DPTR
        lcall   cknull
        mov     Nhp8,A
        mov     DPTR,#0Ch
        movx    A,@DPTR
        mov     NhpB,A
        mov     DPTR,#0Dh
        movx    A,@DPTR
        mov     NhpA,A
        lcall   enrdmm
        ret
;
wr_mem: lcall   stwrmm
        mov     DPTR,#00h
        mov     A,Dlmp
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#01h
        mov     A,Ssys
        movx    @DPTR,A
        lcall   wt_wr
;
        mov     DPTR,#02h
```

```
        mov     A,Nhp1
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#03h
        mov     A,Nhp0
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#04h
        mov     A,Nhp3
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#05h
        mov     A,Nhp2
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#06h
        mov     A,Nhp5
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#07h
        mov     A,Nhp4
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#08h
        mov     A,Nhp7
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#09h
        mov     A,Nhp6
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#0Ah
        mov     A,Nhp9
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#0Bh
        mov     A,Nhp8
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#0Ch
        mov     A,NhpB
        movx    @DPTR,A
        lcall   wt_wr
        mov     DPTR,#0Dh
        mov     A,NhpA
        movx    @DPTR,A
        lcall   wt_wr
        lcall   enwrmm
        ret
;
cknull: cjne    A,#0FFh,cknul
        mov     A,#00h
cknul:  ret
;
scnkpd: mov     R0,#10
        lcall   delay0
col1:   mov     P1,#11111110b
        mov     A,P1
clb1:   cjne    A,#11101110b,clb2
        mov     R0,#1
clb2:   cjne    A,#11011110b,clb3
        mov     R0,#2
clb3:   cjne    A,#10111110b,clb4
        mov     R0,#3
```

```
c1b4:   cjne    A,#01111110b,col2
        mov     R0,#13
;
col2:   mov     P1,#11111101b
        mov     A,P1
c2b1:   cjne    A,#11101101b,c2b2
        mov     R0,#4
c2b2:   cjne    A,#11011101b,c2b3
        mov     R0,#5
c2b3:   cjne    A,#10111101b,c2b4
        mov     R0,#6
c2b4:   cjne    A,#01111101b,col3
        mov     R0,#14
;
col3:   mov     P1,#11111011b
        mov     A,P1
c3b1:   cjne    A,#11101011b,c3b2
        mov     R0,#7
c3b2:   cjne    A,#11011011b,c3b3
        mov     R0,#8
c3b3:   cjne    A,#10111011b,c3b4
        mov     R0,#9
c3b4:   cjne    A,#01111011b,col4
        mov     R0,#15
;
col4:   mov     P1,#11110111b
        mov     A,P1
c4b1:   cjne    A,#11100111b,c4b2
        mov     R0,#11
c4b2:   cjne    A,#11010111b,c4b3
        mov     R0,#0
c4b3:   cjne    A,#10110111b,c4b4
        mov     R0,#12
c4b4:   cjne    A,#01110111b,back
        mov     R0,#16
back:   ret
;
tg_tkn: lcall   scnkpd
tg_tk0: cjne    R0,#16,tg_tk1
        ljmp    tg_tkn
tg_tk1: cjne    R0,#15,tg_tk2
        ljmp    tg_tkn
tg_tk2: cjne    R0,#14,tg_tk3
        ljmp    tg_tkn
tg_tk3: cjne    R0,#13,tg_tk4
        ljmp    tg_tkn
tg_tk4: cjne    R0,#12,tg_tk5
        ljmp    tg_tkn
tg_tk5: cjne    R0,#11,tg_tk6
        ljmp    tg_tkn
tg_tk6: cjne    R0,#10,tg_tk7
        ljmp    tg_tkn
tg_tk7: ret
;
tg_lps: lcall   scnkpd
        cjne    R0,#10,tg_lps
        ret
;
line1:  mov     P0,#080h
        lcall   w_ins
        ret
;
line2:  mov     P0,#0C0h
        lcall   w_ins
```

```
        ret
;
tulis:  clr     A
        lcall   wr_chr
        inc     DPTR
        djnz    Char,tulis
        ret
;
wr_chr: movc    A,@A+DPTR
        mov     P0,A
        lcall   w_chr
        ret
;
w_ins:  clr     Enbl
        clr     Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret
;
w_chr:  clr     Enbl
        setb    Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret
;
lcd_in: lcall   delay1
        mov     P0,#01h             ; Display Clear
        lcall   w_ins
        mov     P0,#38h             ; Function Set
        lcall   w_ins
        mov     P0,#0Dh             ; Display On, Cursor, Blink
        lcall   w_ins
        mov     P0,#06h             ; Entry Mode
        lcall   w_ins
        mov     P0,#02h             ; Cursor Home
        lcall   w_ins
        ret
;
lcdclr: mov     P0,#01h             ; Display Clear
        lcall   w_ins
        lcall   delay0
        lcall   delay0
        ret
;
delay0: djnz    Dly0,delay0
        ret
;
delay1: lcall   scnkpd
        cjne    R0,#13,dely10
        ljmp    rbhsts
dely10: cjne    R0,#14,dely11
;        ljmp    isinhp
dely11: djnz    Dly1,delay1
        ret
;
delay2: mov     Dly2,#20
dely2:  lcall   delay1
        djnz    Dly2,dely2
        ret
;
nama:   DB      ' David Ariyanto '
nim:    DB      ' NIM: 05.52.007 '
```

```
jur:     DB       ' Teknik Elektro '
univ:    DB       '   ITN Malang   '
teslmp:  DB       '   Test  Lamp   '
tpsyon:  DB       '   System  On   '
tpsyof:  DB       '   System Off   '
tprbst:  DB       '  Rubah Status  '
tpnohp:  DB       '   Nomor HP     '
tpintb:  DB       'Input Tombol 000'
tpinsn:  DB       'Input Sensor 000'
angka:   DB       '0123456789ABCDEF '
smseco:  DB       'ATE0'
smsfrm:  DB       'AT+CNMI=1,1,0,0,1'
smsred:  DB       'AT+CMGR=1'
smshps:  DB       'AT+CMGD=1'
smskr0:  DB       'AT+CMGS=27'
smskr1:  DB       'AT+CMGS=29'
smskr2:  DB       'AT+CMGS=35'
smskr3:  DB       'AT+CMGS=42'
cmnd00:  DB       '4C580CF47402'              ; L01 ON
cmnd01:  DB       '4C580CF43402'              ; L01 OF
cmnd02:  DB       '4C980CF47402'              ; L02 ON
cmnd03:  DB       '4C980CF43402'              ; L02 OF
cmnd04:  DB       '4CD80CF47402'              ; L03 ON
cmnd05:  DB       '4CD80CF43402'              ; L03 OF
cmnd06:  DB       '4C180DF47402'              ; L04 ON
cmnd07:  DB       '4C180DF43402'              ; L04 OF
cmnd16:  DB       'D3EC14F47402'              ; SYS ON
cmnd17:  DB       'D3EC14F43402'              ; SYS OF
cmnd18:  DB       '412613F47402'              ; ALL ON
cmnd19:  DB       '412613F43402'              ; ALL OF
cmnd20:  DB       '536A905A9D02'              ; STATUS
kosong:  DB       '                '
;
header:  DB       '0001000D9126'
lmp1on:  DB       '00000ECC701B5E07C162'      ; lampu 01 hidup
         DB       '20649A5C8703'
lmp1of:  DB       '00000ECC701B5E07C162'      ; lampu 01 mati
         DB       'A066989E0601'
lmp2on:  DB       '00000ECC701B5E07C164'      ; lampu 02 hidup
         DB       '20649A5C8703'
lmp2of:  DB       '00000ECC701B5E07C164'      ; lampu 02 mati
         DB       'A066989E0601'
lmp3on:  DB       '00000ECC701B5E07C166'      ; lampu 03 hidup
         DB       '20649A5C8703'
lmp3of:  DB       '00000ECC701B5E07C166'      ; lampu 03 mati
         DB       'A066989E0601'
lmp4on:  DB       '00000ECC701B5E07C168'      ; lampu 04 hidup
         DB       '20649A5C8703'
lmp4of:  DB       '00000ECC701B5E07C168'      ; lampu 04 hidup
         DB       'A066989E0601'
lmp5on:  DB       '00000ECC701B5E07C16A'      ; lampu 05 hidup
         DB       '20649A5C8703'
lmp5of:  DB       '00000ECC701B5E07C16A'      ; lampu 05 mati
         DB       'A066989E0601'
lmp6on:  DB       '00000ECC701B5E07C16C'      ; lampu 06 hidup
         DB       '20649A5C8703'
lmp6of:  DB       '00000ECC701B5E07C16C'      ; lampu 06 mati
         DB       'A066989E0601'
lmp7on:  DB       '00000ECC701B5E07C16E'      ; lampu 07 hidup
         DB       '20649A5C8703'
lmp7of:  DB       '00000ECC701B5E07C16E'      ; lampu 07 mati
         DB       'A066989E0601'
lmp8on:  DB       '00000ECC701B5E07C170'      ; lampu 08 hidup
         DB       '20649A5C8703'
```

```
lmp8of: DB      '00000ECC701B5E07C170'    ; lampu 08 mati
        DB      'A066989E0601'
;
smssts: DB      '000011533A7D0ED28140'          ; stts :
smple0: DB      '000011533A7D0ED28140'          ; contoh: stts:
0000 0000
        DB      '30180C0682C16030'
;
syenst: DB      '000021D3FC9C5E6E838A'          ; contoh: System
Enable , Stts :
        DB      'EEB0985D06B140533A7D'
        DB      '0ED28140'
sydsst: DB      '000021D3FC9C5E6E8388'          ; contoh: System
Disable , Stts :
        DB      'E97958CC2E8358A0299D'
        DB      '3E07E940'
smple1: DB      '0001000D912658571516'          ; contoh: kirim ke
HP MY
        DB      '49F6000021D3FC9C5E6E'          ; System Enable ,
Stts :  0000 0000
        DB      '838AEEB0985D06B14053'
        DB      '3A7D0ED28140330180C06'
        DB      '82C16030'
;
smlhst: DB      '000029D372BB1E0631C3'          ; contoh: Semua
Lampu Dihidupkan , Stts :
        DB      '6D781D444CA3D3E43A7C'
        DB      '1D768358A0299D3E07E9'
        DB      '40'
smlmst: DB      '000029D372BB1E0631C3'          ; contoh: Semua
Lampu Dimatikan , Stts :
        DB      '6D781D444CB7C3F4F43A'
        DB      'EC06B140533A7D0ED281'
        DB      '40'
smple2: DB      '0001000D912658571516'          ; contoh: kirim ke
HP MY
        DB      '49F6000029D372BB1E06'          ; Semua Lampu
Dihidupkan , Stts : 0000 0000
        DB      '31C36D781D444CA3D3E4'
        DB      '3A7C1D768358A0299D3E'
        DB      '07E94030180C0682C160'
        DB      '30'
;
kr_pdu: mov     DPTR,#angka          ;\
        mov     R0,Chr0              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr1              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr2              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr3              ; | cacah PDU
      lcall cchpdu                   ; | kirim PDU
        mov     R0,Chr4              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr5              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr6              ; |
      lcall cchpdu                   ; |
        mov     R0,Chr7              ; |
      lcall cchpdu                   ; |
        mov     A,#26                ; | character 26 = CTRL+Z
        lcall   kr_srl               ; | kirim
        ret                  ;/
;
```

```
cchpdu: mov     A,R0                    ;\
        anl     A,#0F0h                 ; |
        RR      A               ; |
        RR      A               ; |
        RR      A               ; |
        RR      A               ; | cacah character
        movc    A,@A+DPTR               ; | menjadi data hexa
        lcall   kr_srl                  ; |
        mov     A,R0                    ; |
        anl     A,#00Fh                 ; |
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        ret                     ;/
;
jd_pdu: mov     A,Chr1                  ;\
        RR      A               ; | datal geser 7x
        anl     A,#080h                 ; | and dg 1000 0000
        orl     A,Chr0                  ; | or dg data0
        mov     Chr0,A                  ;/
;
        mov     A,Chr1                  ;\
        anl     A,#11111110b            ; |
        RR      A               ; | datal hilangkan bit0
        mov     Bufr,A                  ; | datal geser kanan 1x
        mov     A,Chr2                  ; | simpan sementara
        RR      A               ; | and dg 1100 0000
        RR      A               ; | or dg bufr
        anl     A,#0C0h                 ; |
        orl     A,Bufr                  ; |
        mov     Chr1,A                  ;/
;
        mov     A,Chr2                  ;\
        anl     A,#11111100b            ; |
        RR      A               ; |
        RR      A               ; | datal hilangkan bit0 & bit1
        mov     Bufr,A                  ; | datal geser kanan 2x
        mov     A,Chr3                  ; | simpan sementara
        RR      A               ; | and dg 1110 0000
        RR      A               ; | or dg bufr
        RR      A               ; |
        anl     A,#0E0h                 ; |
        orl     A,Bufr                  ; |
        mov     Chr2,A                  ;/
;
        mov     A,Chr3
        anl     A,#11111C00b
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Chr4
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F0h
        orl     A,Bufr
        mov     Chr3,A
;
        mov     A,Chr4
        anl     A,#11110000b
        RR      A
        RR      A
        RR      A
```

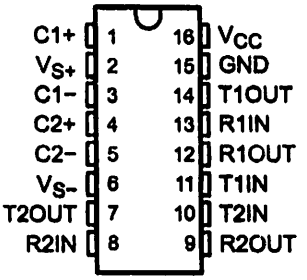
```
        RR      A
        mov     Bufr,A
        mov     A,Chr5
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F8h
        orl     A,Bufr
        mov     Chr4,A
;
        mov     A,Chr5
        anl     A,#11100000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Chr6
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FCh
        orl     A,Bufr
        mov     Chr5,A
;
        mov     A,Chr6
        anl     A,#11000000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Chr7
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FEh
        orl     A,Bufr
        mov     Chr6,A
;
        mov     A,Chr9
      RR      A
        anl     A,#080h
        orl     A,Chr8
        mov     Chr7,A
        ret
;
        org     2000h
        DB      0,0,5,8,5,7,1,5,1,6,4,9,0Fh,6
;               L S \_________no HP__________/
;
        end
```

MAX232, MAX232I
DUAL EIA-232 DRIVERS/RECEIVERS

SLLS047L – FEBRUARY 1989 – REVISED MARCH 2004

- Meets or Exceeds TIA/EIA-232-F and ITU Recommendation V.28
- Operates From a Single 5-V Power Supply With 1.0-μF Charge-Pump Capacitors
- Operates Up To 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current . . . 8 mA Typical
- ESD Protection Exceeds JESD 22
  - 2000-V Human-Body Model (A114-A)
- Upgrade With Improved ESD (15-kV HBM) and 0.1-μF Charge-Pump Capacitors is Available With the MAX202
- Applications
  - TIA/EIA-232-F, Battery-Powered Systems, Terminals, Modems, and Computers

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)



escription/ordering information

The MAX232 is a dual driver/receiver that includes a capacitive voltage generator to supply TIA/EIA-232-F voltage levels from a single 5-V supply. Each receiver converts TIA/EIA-232-F inputs to 5-V TTL/CMOS levels. These receivers have a typical threshold of 1.3 V, a typical hysteresis of 0.5 V, and can accept ±30-V inputs. Each driver converts TTL/CMOS input levels into TIA/EIA-232-F levels. The driver, receiver, and voltage-generator functions are available as cells in the Texas Instruments LinASIC™ library.

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube of 25	MAX232N	MAX232N
	SOIC (D)	Tube of 40	MAX232D	MAX232
		Reel of 2500	MAX232DR	
	SOIC (DW)	Tube of 40	MAX232DW	MAX232
		Reel of 2000	MAX232DWR	
	SOP (NS)	Reel of 2000	MAX232NSR	MAX232
−40°C to 85°C	PDIP (N)	Tube of 25	MAX232IN	MAX232IN
	SOIC (D)	Tube of 40	MAX232ID	MAX232I
		Reel of 2500	MAX232IDR	
	SOIC (DW)	Tube of 40	MAX232IDW	MAX232I
		Reel of 2000	MAX232IDWR	

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Function Tables

EACH DRIVER

INPUT TIN	OUTPUT TOUT
L	H
H	L

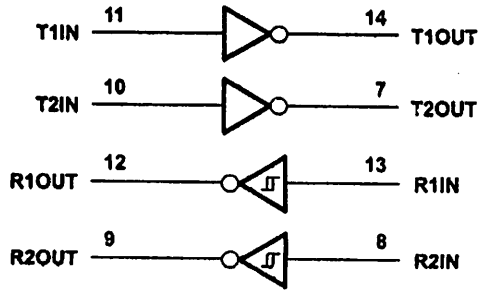
H = high level, L = low level

EACH RECEIVER

INPUT RIN	OUTPUT ROUT
L	H
H	L

H = high level, L = low level

c diagram (positive logic)



absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Input supply voltage range, V_{CC} (see Note 1) ..... −0.3 V to 6 V
Positive output supply voltage range, V_{S+} ..... V_{CC} − 0.3 V to 15 V
Negative output supply voltage range, V_{S-} ..... −0.3 V to −15 V
Input voltage range, V_I: Driver ..... −0.3 V to V_{CC} + 0.3 V
Receiver ..... ±30 V
Output voltage range, V_O: T1OUT, T2OUT ..... V_{S-} − 0.3 V to V_{S+} + 0.3 V
R1OUT, R2OUT ..... −0.3 V to V_{CC} + 0.3 V
Short-circuit duration: T1OUT, T2OUT ..... Unlimited
Package thermal impedance, θ_{JA} (see Notes 2 and 3): D package ..... 73°C/W
DW package ..... 57°C/W
N package ..... 67°C/W
NS package ..... 64°C/W
Operating virtual junction temperature, T_J ..... 150°C
Storage temperature range, T_{stg} ..... −65°C to 150°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTES: 1. All voltages are with respect to network GND.
2. Maximum power dissipation is a function of $T_J(max)$, θ_{JA}, and T_A. The maximum allowable power dissipation at any allowable ambient temperature is $P_D = (T_J(max) - T_A)/\theta_{JA}$. Operating at the absolute maximum T_J of 150°C can affect reliability.
3. The package thermal impedance is calculated in accordance with JESD 51-7.

recommended operating conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage (T1IN,T2IN)		2			V
V_{IL}	Low-level input voltage (T1IN, T2IN)				0.8	V
R1IN, R2IN	Receiver input voltage				±30	V
T_A	Operating free-air temperature	MAX232	0		70	°C
		MAX232I	−40		85	

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Note 4 and Figure 4)

PARAMETER		TEST CONDITIONS	MIN	TYP‡	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5 V, All outputs open, T_A = 25°C		8	10	mA

‡ All typical values are at V_{CC} = 5 V and T_A = 25°C.

NOTE 4: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

DRIVER SECTION

trical characteristics over recommended ranges of supply voltage and operating free-air perature range (see Note 4)

PARAMETER		TEST CONDITIONS	MIN	TYP†	MAX	UNIT
High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND	5	7		V
Low-level output voltage‡	T1OUT, T2OUT	R_L = 3 kΩ to GND		−7	−5	V
Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0, V_O = ±2 V	300			Ω
Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V, V_O = 0		±10		mA
Short-circuit input current	T1IN, T2IN	V_I = 0			200	µA

pical values are at V_{CC} = 5 V, T_A = 25°C.
algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage s only.
nore than one output should be shorted at a time.
4: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

ching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 4)

PARAMETER	TEST CONDITIONS	MIN	TYP	MAX	UNIT
Driver slew rate	R_L = 3 kΩ to 7 kΩ, See Figure 2			30	V/µs
Driver transition region slew rate	See Figure 3		3		V/µs
Data rate	One TOUT switching		120		kbit/s

4: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

RECEIVER SECTION

trical characteristics over recommended ranges of supply voltage and operating free-air perature range (see Note 4)

PARAMETER		TEST CONDITIONS	MIN	TYP†	MAX	UNIT
High-level output voltage	R1OUT, R2OUT	I_{OH} = −1 mA	3.5			V
Low-level output voltage‡	R1OUT, R2OUT	I_{OL} = 3.2 mA			0.4	V
Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C		1.7	2.4	V
Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V, T_A = 25°C	0.8	1.2		V
Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V	0.2	0.5	1	V
Receiver input resistance	R1IN, R2IN	V_{CC} = 5, T_A = 25°C	3	5	7	kΩ

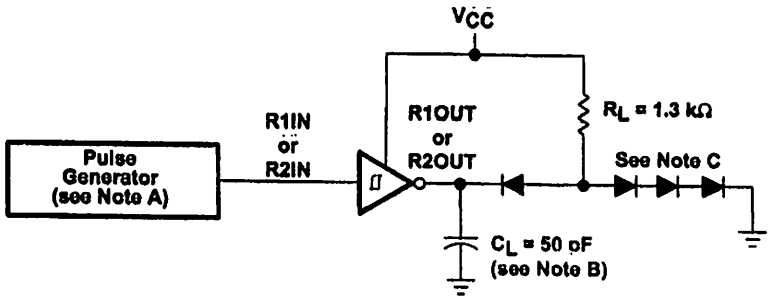
pical values are at V_{CC} = 5 V, T_A = 25°C.
algebraic convention, in which the least-positive (most negative) value is designated minimum, is used in this data sheet for logic voltage s only.
4: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

ching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 4 and Figure 1)

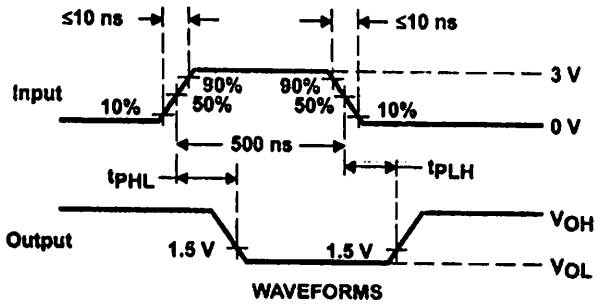
PARAMETER	TYP	UNIT
R) Receiver propagation delay time, low- to high-level output	500	ns
R) Receiver propagation delay time, high- to low-level output	500	ns

4: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

PARAMETER MEASUREMENT INFORMATION



TEST CIRCUIT

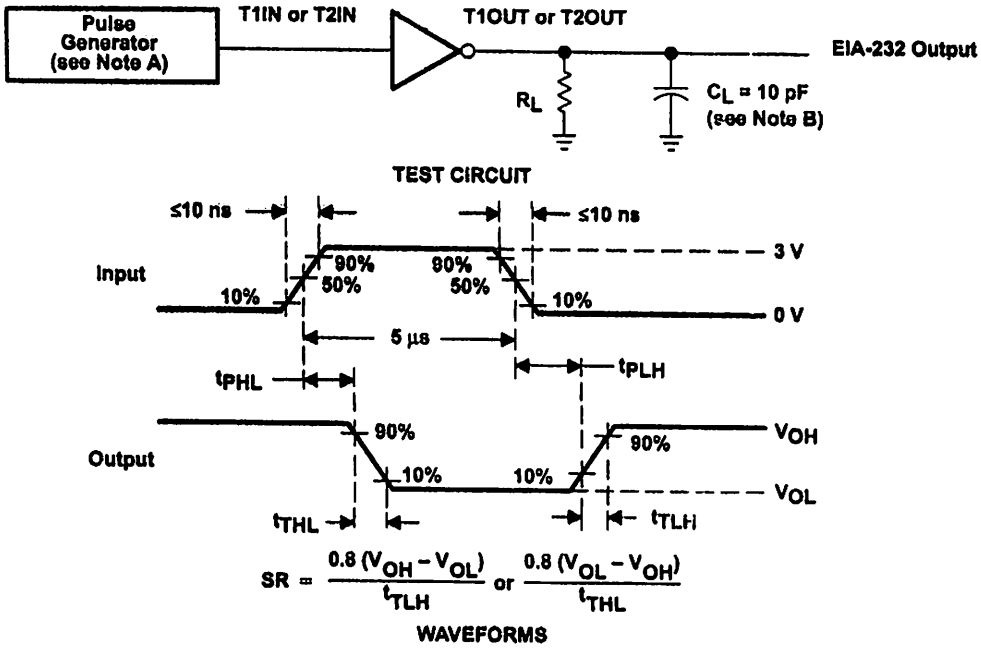


WAVEFORMS

NOTES: A. The pulse generator has the following characteristics: Z_O = 50 Ω, duty cycle ≤ 50%.
B. C_L includes probe and jig capacitance.
C. All diodes are 1N3064 or equivalent.

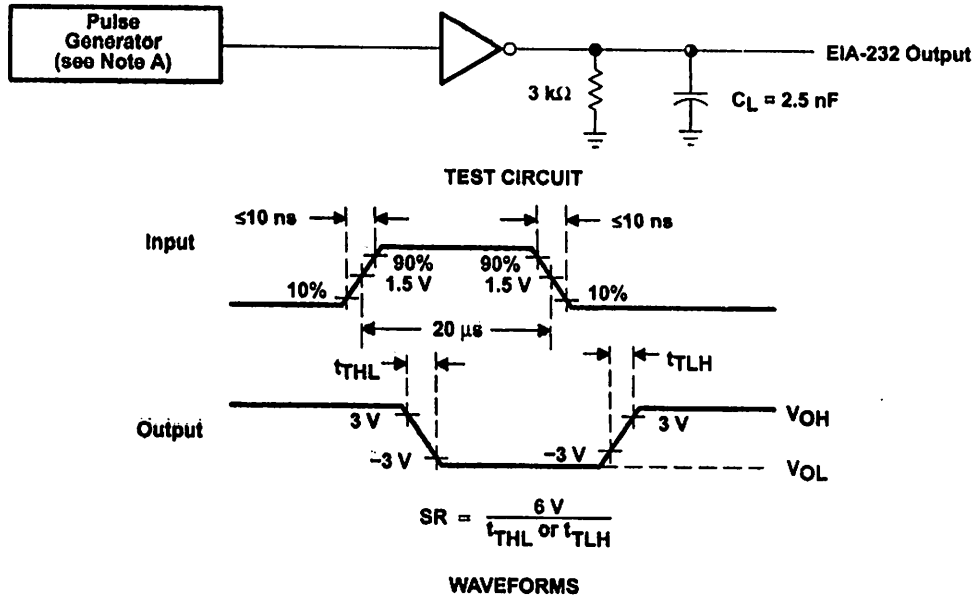
Figure 1. Receiver Test Circuit and Waveforms for t_{PHL} and t_{PLH} Measurements

PARAMETER MEASUREMENT INFORMATION



ES: A. The pulse generator has the following characteristics: Z_O = 50 Ω, duty cycle ≤ 50%.
B. C_L includes probe and jig capacitance.

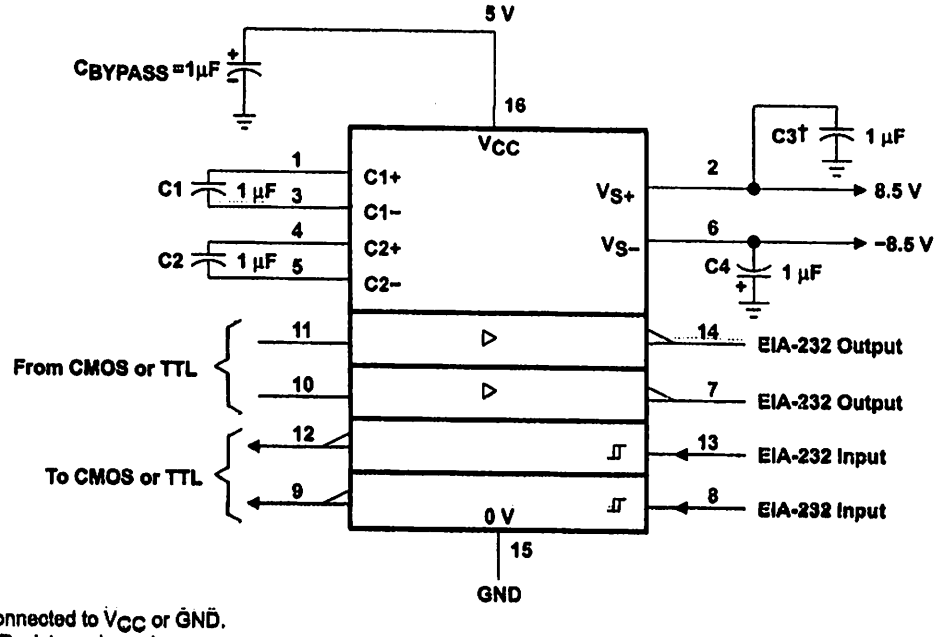
Figure 2. Driver Test Circuit and Waveforms for t_{PHL} and t_{PLH} Measurements (5-µs Input)



E A: The pulse generator has the following characteristics: Z_O = 50 Ω, duty cycle ≤ 50%.

Figure 3. Test Circuit and Waveforms for t_{THL} and t_{TLH} Measurements (20-µs Input)

APPLICATION INFORMATION



† C3 can be connected to V_{CC} or GND.

NOTES: A. Resistor values shown are nominal.

B. Nonpolarized ceramic capacitors are acceptable. If polarized tantalum or electrolytic capacitors are used, they should be connected as shown. In addition to the 1-µF capacitors shown, the MAX202 can operate with 0.1-µF capacitors.

Figure 4. Typical Operating Circuit

PACKAGING INFORMATION

Orderable Device	Status (1)	Package Type	Package Drawing	Pins	Package Qty	Eco Plan (2)	Lead/Ball Finish	MSL Peak Temp (3)
MAX232D	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DE4	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DG4	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DR	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DRE4	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DRG4	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DW	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DWE4	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DWG4	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DWR	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DWRE4	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232DWRG4	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232ID	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDE4	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDG4	ACTIVE	SOIC	D	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDR	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDRE4	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDRG4	ACTIVE	SOIC	D	16	2500	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDW	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDWE4	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDWG4	ACTIVE	SOIC	DW	16	40	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDWR	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDWRE4	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IDWRG4	ACTIVE	SOIC	DW	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232IN	ACTIVE	PDIP	N	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type

TEXAS INSTRUMENTS
www.ti.com

PACKAGE OPTION ADDENDUM

4-Jun-2007

Orderable Device	Status [1]	Package Type	Package Drawing	Pins	Package Qty	Eco Plan [2]	Lead/Ball Finish	MSL Peak Temp [3]
MAX232INE4	ACTIVE	PDIP	N	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type
MAX232N	ACTIVE	PDIP	N	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type
MAX232NE4	ACTIVE	PDIP	N	16	25	Pb-Free (RoHS)	CU NIPDAU	N / A for Pkg Type
MAX232NSR	ACTIVE	SO	NS	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232NSRE4	ACTIVE	SO	NS	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM
MAX232NSRG4	ACTIVE	SO	NS	16	2000	Green (RoHS & no Sb/Br)	CU NIPDAU	Level-1-260C-UNLIM

) The marketing status values are defined as follows:
CTIVE: Product device recommended for new designs.
IFEBUY: TI has announced that the device will be discontinued, and a lifetime-buy period is in effect.
IRND: Not recommended for new designs. Device is in production to support existing customers, but TI does not recommend using this part in new design.
REVIEW: Device has been announced but is not in production. Samples may or may not be available.
BSOLETE: TI has discontinued the production of the device.

) Eco Plan - The planned eco-friendly classification: Pb-Free (RoHS), Pb-Free (RoHS Exempt), or Green (RoHS & no Sb/Br) - please check ttp://www.ti.com/productcontent for the latest availability information and additional product content details.
BD: The Pb-Free/Green conversion plan has not been defined.
b-Free (RoHS): TI's terms "Lead-Free" or "Pb-Free" mean semiconductor products that are compatible with the current RoHS requirements or all 6 substances, including the requirement that lead not exceed 0.1% by weight in homogeneous materials. Where designed to be soldered t high temperatures, TI Pb-Free products are suitable for use in specified lead-free processes.
b-Free (RoHS Exempt): This component has a RoHS exemption for either 1) lead-based flip-chip solder bumps used between the die and ackage, or 2) lead-based die adhesive used between the die and leadframe. The component is otherwise considered Pb-Free (RoHS ompatible) as defined above.
reen (RoHS & no Sb/Br): TI defines "Green" to mean Pb-Free (RoHS compatible), and free of Bromine (Br) and Antimony (Sb) based flame tardants (Br or Sb do not exceed 0.1% by weight in homogeneous material)

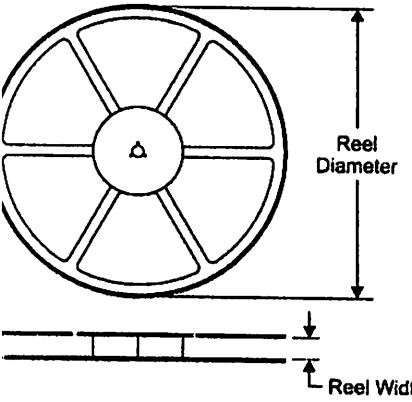
) MSL, Peak Temp. – The Moisture Sensitivity Level rating according to the JEDEC industry standard classifications, and peak solder mperature.

nportant Information and Disclaimer: The information provided on this page represents TI's knowledge and belief as of the date that it is rovided. TI bases its knowledge and belief on information provided by third parties, and makes no representation or warranty as to the ccuracy of such information. Efforts are underway to better integrate information from third parties. TI has taken and continues to take easonable steps to provide representative and accurate information but may not have conducted destructive testing or chemical analysis on coming materials and chemicals. TI and TI suppliers consider certain information to be proprietary, and thus CAS numbers and other limited formation may not be available for release.

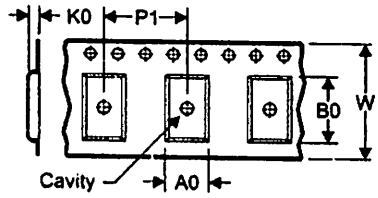
n no event shall TI's liability arising out of such information exceed the total purchase price of the TI part(s) at issue in this document sold by TI Customer on an annual basis.

TAPE AND REEL INFORMATION

REEL DIMENSIONS

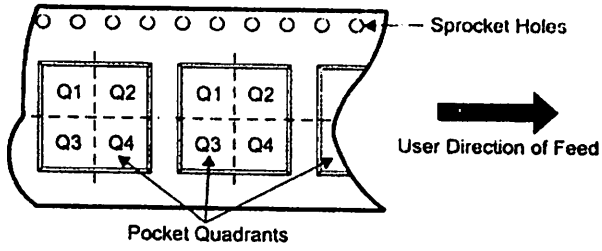


TAPE DIMENSIONS



A0	Dimension designed to accommodate the component width
B0	Dimension designed to accommodate the component length
K0	Dimension designed to accommodate the component thickness
W	Overall width of the carrier tape
P1	Pitch between successive cavity centers

QUADRANT ASSIGNMENTS FOR PIN 1 ORIENTATION IN TAPE



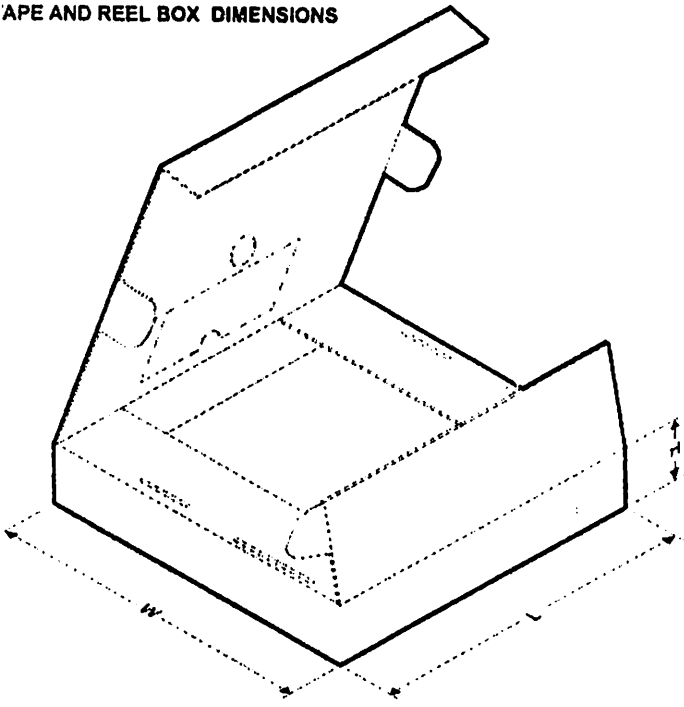
All dimensions are nominal

Device	Package Type	Package Drawing	Pins	SPQ	Reel Diameter (mm)	Reel Width W1 (mm)	A0 (mm)	B0 (mm)	K0 (mm)	P1 (mm)	W (mm)	Pin1 Quadrant
MAX232DR	SOIC	D	16	2500	330.0	16.4	6.5	10.3	2.1	8.0	16.0	Q1
MAX232DR	SOIC	D	16	2500	330.0	16.4	6.5	10.3	2.1	8.0	16.0	Q1
MAX232DWR	SOIC	DW	16	2000	330.0	16.4	10.75	10.7	2.7	12.0	16.0	Q1
MAX232IDR	SOIC	D	16	2500	330.0	16.4	6.5	10.3	2.1	8.0	16.0	Q1
MAX232IDWR	SOIC	DW	16	2000	330.0	16.4	10.75	10.7	2.7	12.0	16.0	Q1
MAX232NSR	SO	NS	16	2000	330.0	16.4	8.2	10.5	2.5	12.0	16.0	Q1

PACKAGE MATERIALS INFORMATION

19-Mar-2008

APE AND REEL BOX DIMENSIONS



All dimensions are nominal

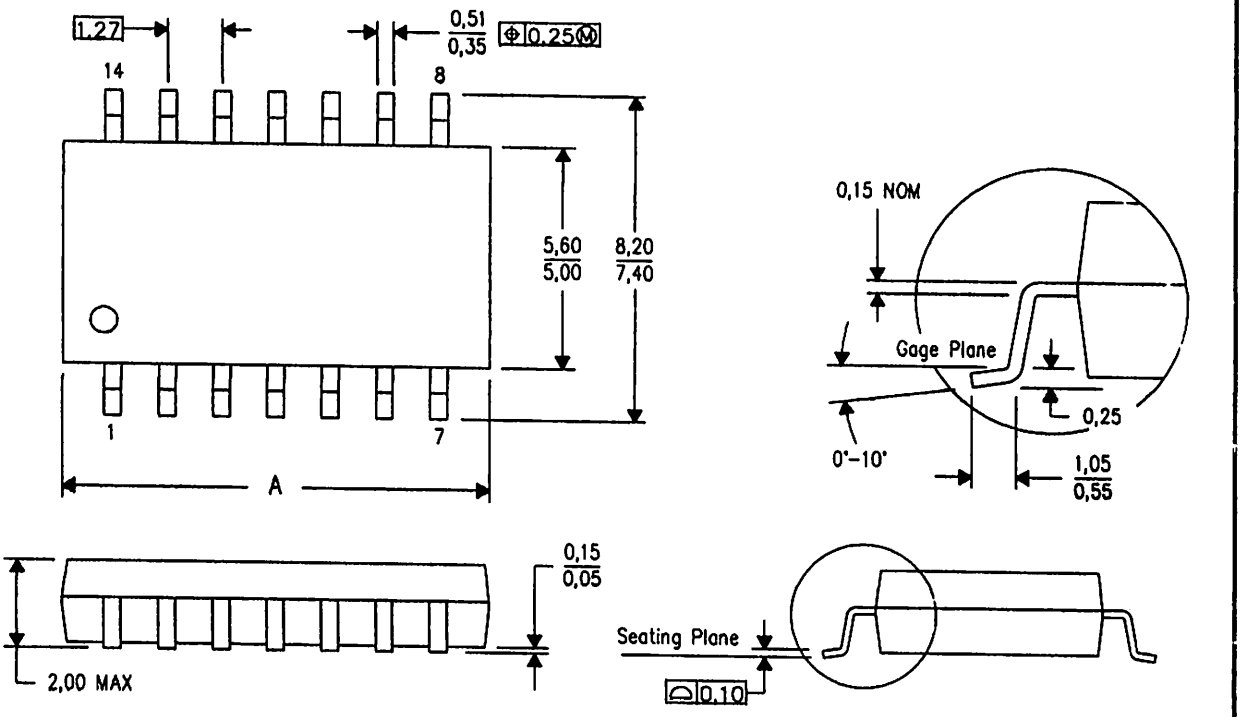
Device	Package Type	Package Drawing	Pins	SPQ	Length (mm)	Width (mm)	Height (mm)
MAX232DR	SOIC	D	16	2500	346.0	346.0	33.0
MAX232DR	SOIC	D	16	2500	333.2	345.9	28.6
MAX232DWR	SOIC	DW	16	2000	346.0	346.0	33.0
MAX232IDR	SOIC	D	16	2500	333.2	345.9	28.6
MAX232IDWR	SOIC	DW	16	2000	346.0	346.0	33.0
MAX232NSR	SO	NS	16	2000	346.0	346.0	33.0

MECHANICAL DATA

(R-PDSO-G**)

PINS SHOWN

PLASTIC SMALL-OUTLINE PACKAGE

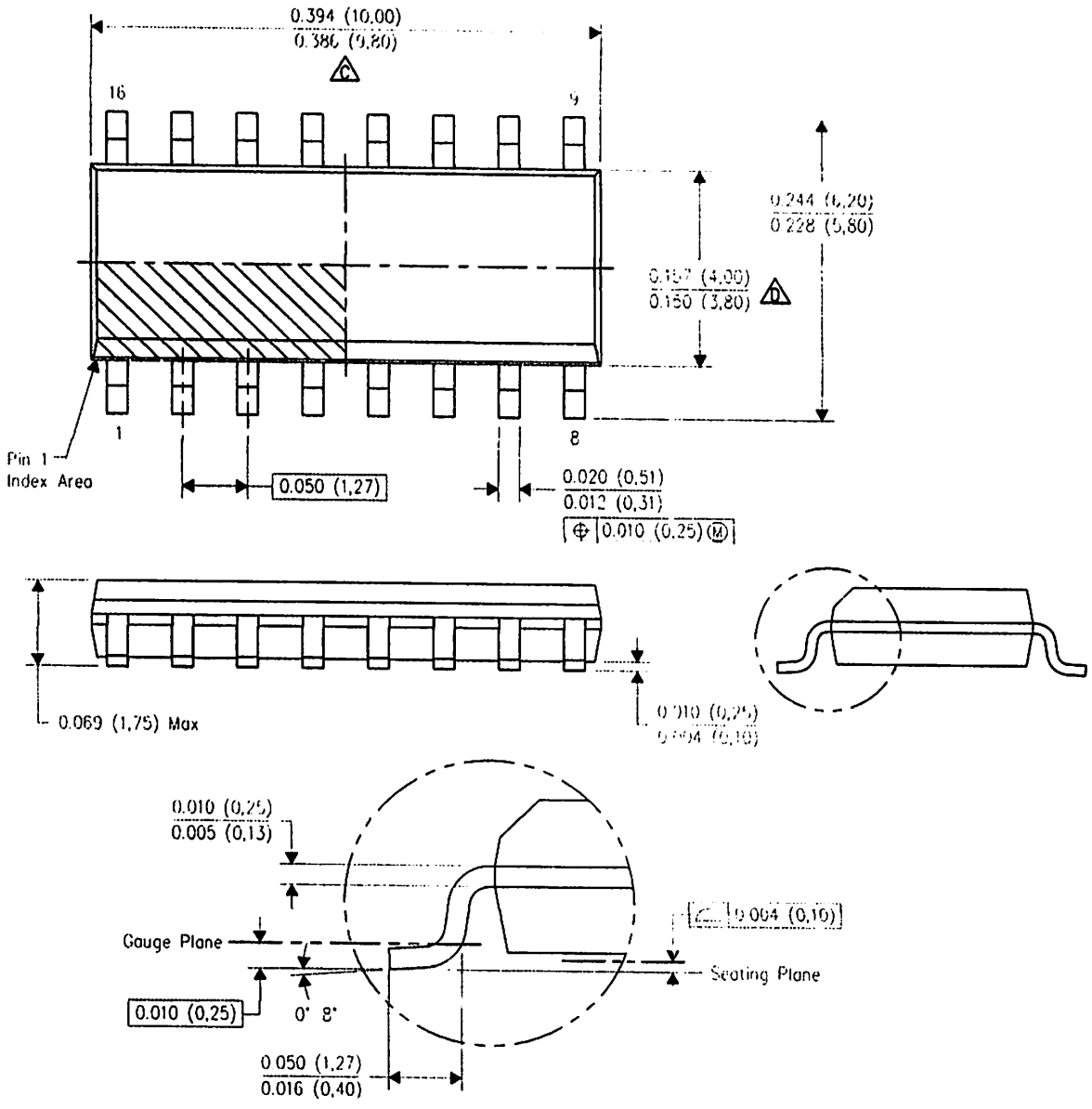


DIM \ PINS **	14	16	20	24
A MAX	10,50	10,50	12,90	15,30
A MIN	9,90	9,90	12,30	14,70

4040062/C 03/03

NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion, not to exceed 0,15.

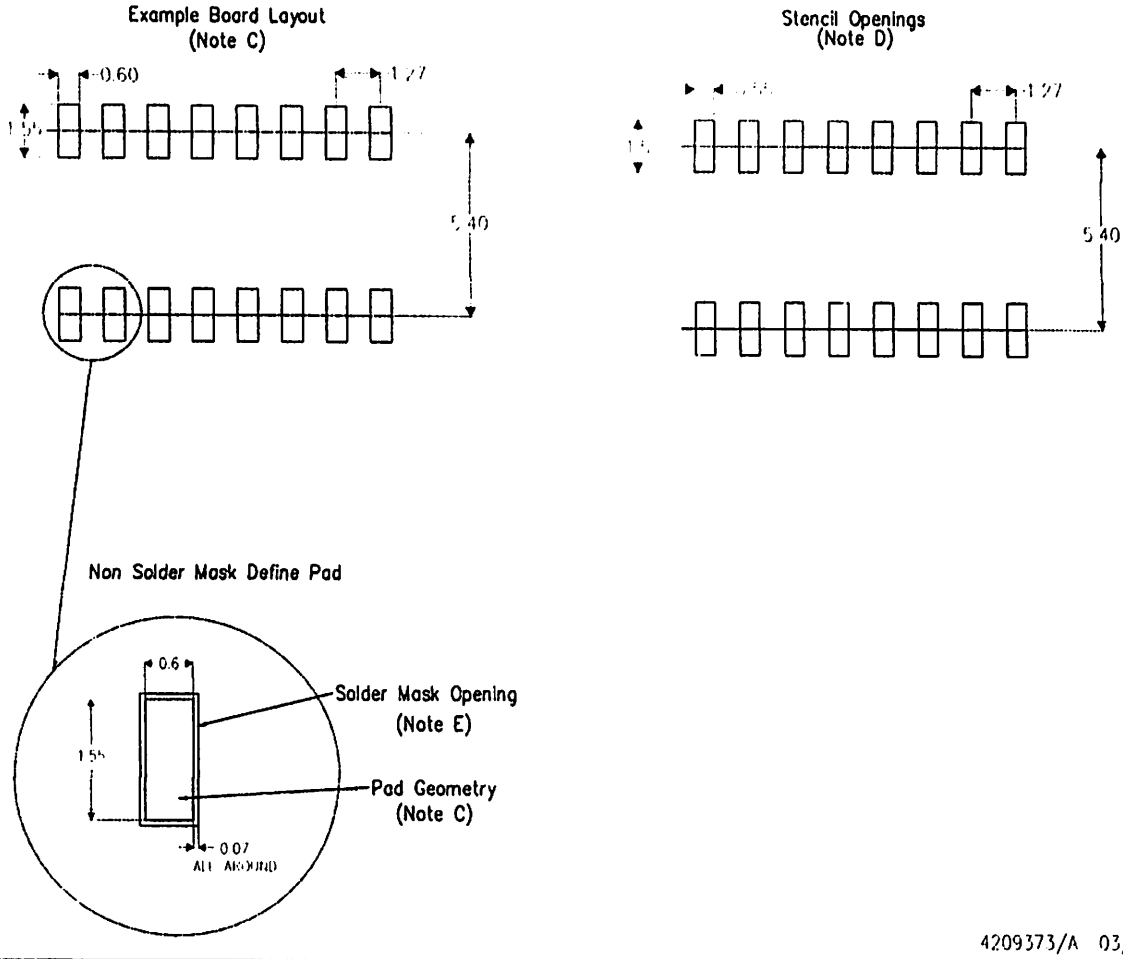
(R–PDSO–G16) PLASTIC SMALL–OUTLINE PACKAGE



4040047-5/J 09/09

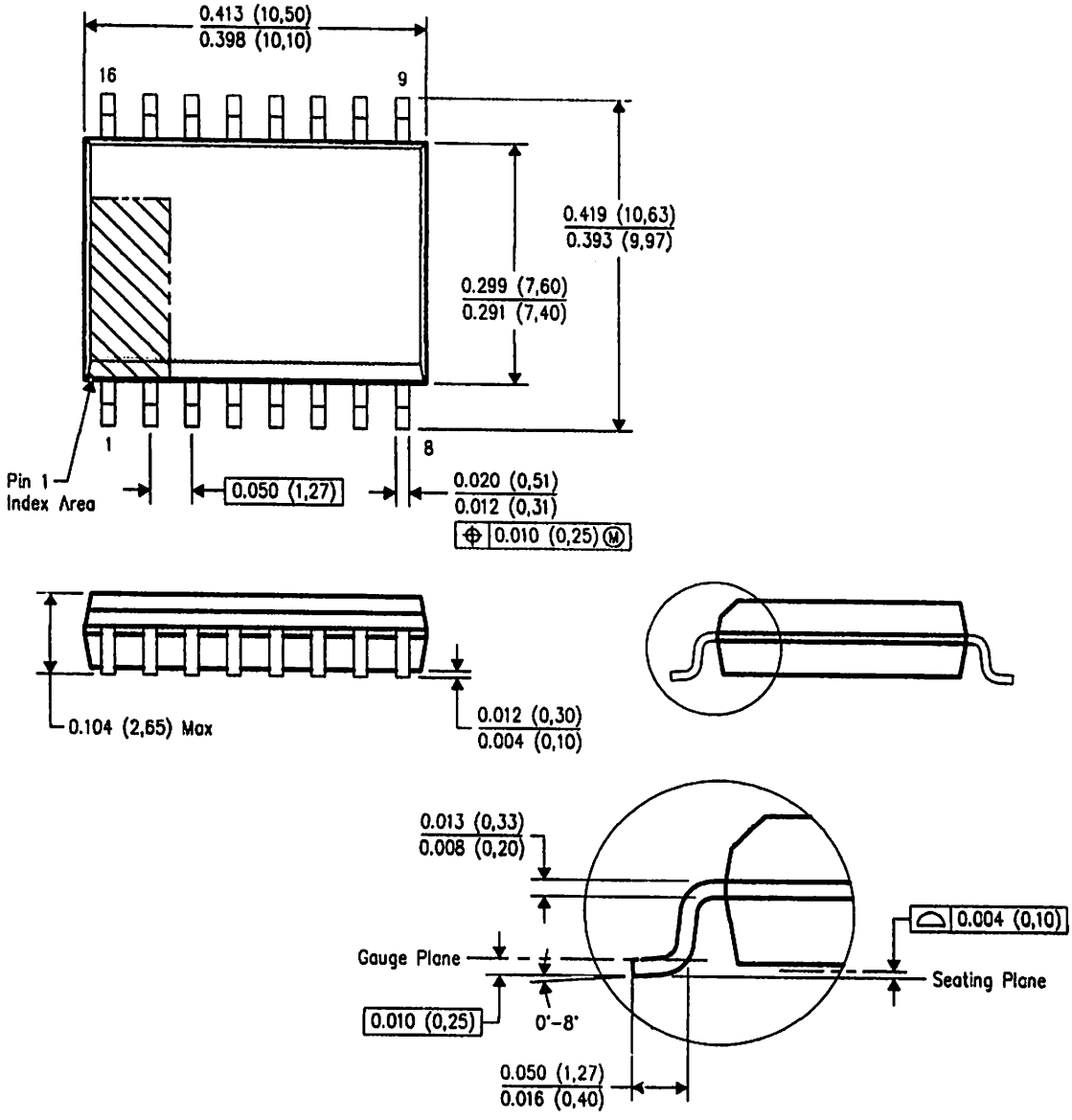
NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Body length does not include mold flash, protrusions, or gate burrs. Mold flash, protrusions, or gate burrs shall not exceed .006 (0,15) per end.
D. Body width does not include interlead flash. Interlead flash shall not exceed .017 (0,43) per side.
E. Reference JEDEC MS–012 variation AC.

)(R–PDSO–G16)



NOTES:
A. All linear dimensions are in millimeters.
B. This drawing is subject to change without notice.
C. Refer to IPC7351 for alternate board design.
D. Laser cutting apertures with trapezoidal walls and also rounding corners will offer better paste release. Customers should contact their board assembly site for stencil design recommendations. Refer to IPC–7525
E. Customers should contact their board fabrication site for solder mask tolerances between and around signal pads.

)W (R–PDSO–G16) PLASTIC SMALL–OUTLINE PACKAGE

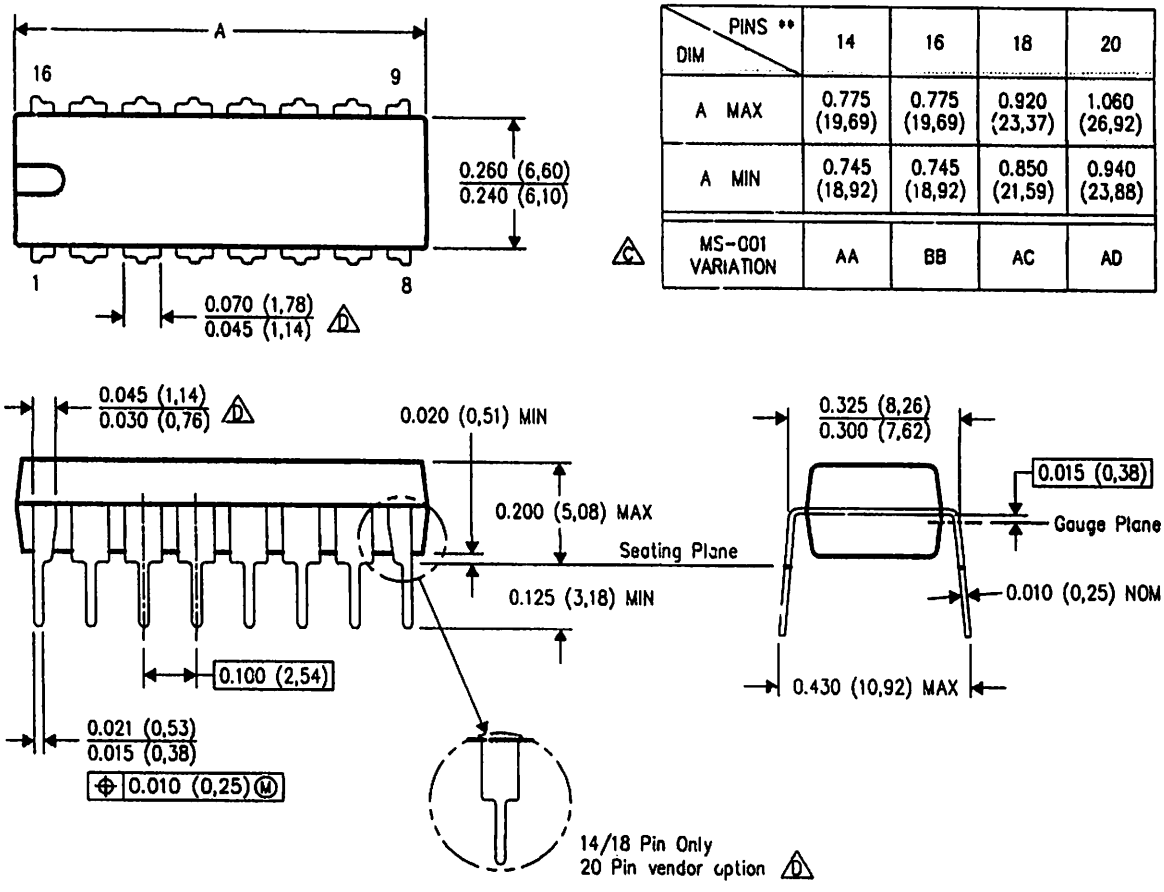


NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Body dimensions do not include mold flash or protrusion not to exceed 0.006 (0,15).
D. Falls within JEDEC MS–013 variation AA.

(R–PDIP–T**)

PINS SHOWN

PLASTIC DUAL–IN–LINE PACKAGE



DIM \ PINS **	14	16	18	20
A MAX	0.775 (19,69)	0.775 (19,69)	0.920 (23,37)	1.060 (26,92)
A MIN	0.745 (18,92)	0.745 (18,92)	0.850 (21,59)	0.940 (23,88)
MS–001 VARIATION	AA	BB	AC	AD

4040049/E 12/2002

NOTES:
A. All linear dimensions are in inches (millimeters).
B. This drawing is subject to change without notice.
C. Falls within JEDEC MS–001, except 18 and 20 pin minimum body length (Dim A).
D. The 20 pin end lead shoulder width is a vendor option, either half or full width.

IMPORTANT NOTICE

xas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, d other changes to its products and services at any time and to discontinue any product or service without notice. Customers should tain the latest relevant information before placing orders and should verify that such information is current and complete. All products are ld subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard rranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where ndated by government requirements, testing of all parameters of each product is not necessarily performed.

assumes no liability for applications assistance or customer product design. Customers are responsible for their products and plications using TI components. To minimize the risks associated with customer products and applications, customers should provide equate design and operating safeguards.

does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information blished by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a rranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual operty of the third party, or a license from TI under the patents or other intellectual property of TI.

production of TI information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive siness practice. TI is not responsible or liable for such altered documentation. Information of third parties may be subject to additional strictions.

sale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all press and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not sponsible or liable for any such statements.

products are not authorized for use in safety-critical applications (such as life support) where a failure of the TI product would reasonably expected to cause severe personal injury or death, unless officers of the parties have executed an agreement specifically governing ch use. Buyers represent that they have all necessary expertise in the safety and regulatory ramifications of their applications, and knowledge and agree that they are solely responsible for all legal, regulatory and safety-related requirements concerning their products d any use of TI products in such safety-critical applications, notwithstanding any applications-related information or support that may be ovided by TI. Further, Buyers must fully indemnify TI and its representatives against any damages arising out of the use of TI products in ch safety-critical applications.

products are neither designed nor intended for use in military/aerospace applications or environments unless the TI products are ecifically designated by TI as military-grade or "enhanced plastic." Only products designated by TI as military-grade meet military ecifications. Buyers acknowledge and agree that any such use of TI products which TI has not designated as military-grade is solely at e Buyer's risk, and that they are solely responsible for compliance with all legal and regulatory requirements in connection with such use.

products are neither designed nor intended for use in automotive applications or environments unless the specific TI products are signated by TI as compliant with ISO/TS 16949 requirements. Buyers acknowledge and agree that, if they use any non-designated oducts in automotive applications, TI will not be responsible for any failure to meet such requirements.

llowing are URLs where you can obtain information on other Texas Instruments products and application solutions:

oducts		**Applications**	
nplifiers	amplifier.ti.com	Audio	www.ti.com/audio
ta Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
.P® Products	www.dlp.com	Broadband	www.ti.com/broadband
SP	dsp.ti.com	Digital Control	www.ti.com/digitalcontrol
ocks and Timers	www.ti.com/clocks	Medical	www.ti.com/medical
erface	interface.ti.com	Military	www.ti.com/military
gic	logic.ti.com	Optical Networking	www.ti.com/opticalnetwork
wer Mgmt	power.ti.com	Security	www.ti.com/security
crocontrollers	microcontroller.ti.com	Telephony	www.ti.com/telephony
ID	www.ti-rfid.com	Video & Imaging	www.ti.com/video
F/IF and ZigBee® Solutions	www.ti.com/lprf	Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

LM311

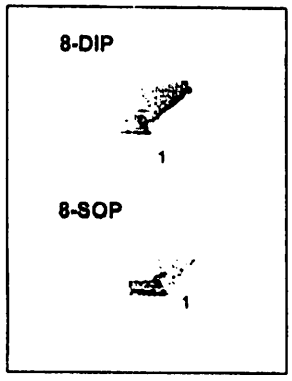
Single Comparator

Features

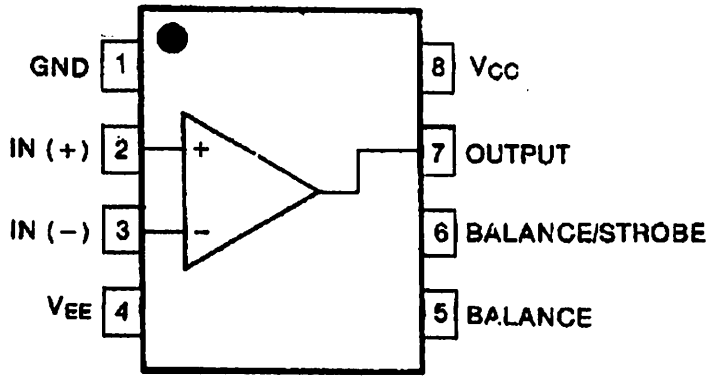
- Low input bias current : 250nA (Max)
- Low input offset current : 50nA (Max)
- Differential Input Voltage : ±30V
- Power supply voltage : single 5.0V supply to ±15V.
- Offset voltage null capability.
- Strobe capability.

Description

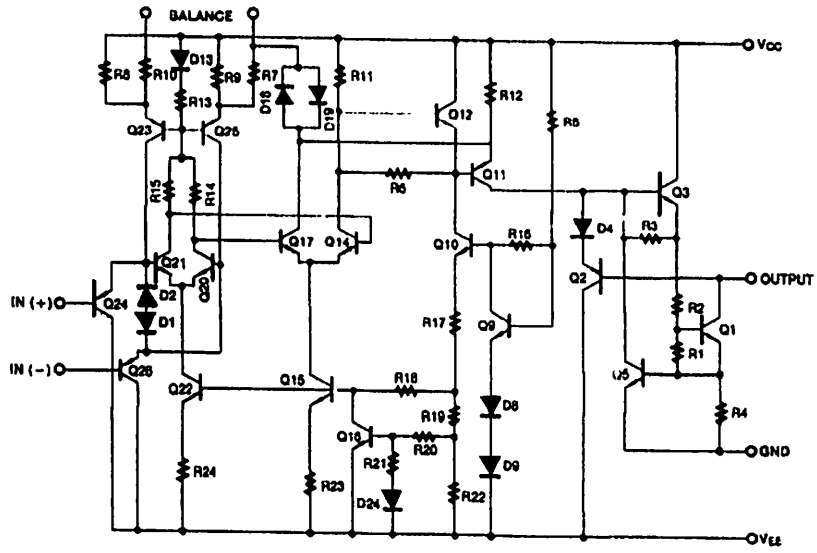
The LM311 series is a monolithic, low input current voltage comparator. The device is also designed to operate from dual or single supply voltage.



Internal Block Diagram



Schematic Diagram



Absolute Maximum Ratings

Parameter	Symbol	Value	Unit
Total Supply Voltage	V_{CC}	36	V
Output to Negative Supply Voltage LM311	V_O - V_{EE}	40	V
Ground to Negative voltage	V_{EE}	-30	V
Differential Input Voltage	$V_{I(DIFF)}$	30	V
Input Voltage	V_I	±15	V
Output Short Circuit Duration	-	10	sec
Power Dissipation	P_D	500	mW
Operating Temperature Range	T_{OPR}	0 ~ +70	°C
Storage Temperature Range	T_{STG}	- 65 ~ +150	°C

Electrical Characteristics

(V_{CC} = 15V, T_A = 25°C, unless otherwise specified)

Parameter	Symbol	Conditions		Min.	Typ.	Max.	Unit
Input Offset Voltage	V_{IO}	$R_S \leq 50K\Omega$		-	1.0	7.5	mV
			Note 1	-	-	10	
Input Offset Current	I_{IO}			-	6	50	nA
			Note 1	-	-	70	
Input Bias Current	I_{BIAS}			-	100	250	nA
			Note 1	-	-	300	
Voltage Gain	G_V	-		40	200	-	V/mV
Response Time	T_{RES}		Note 2	-	200	-	ns
Saturation Voltage	V_{SAT}	I_O =50mA, $V_I \leq$ -10mV		-	0.75	1.5	V
		$V_{CC} \geq$ 4.5V, V_{EE} =0V I_O =8mA, $V_I \leq$ -10mV,	Note 1	-	0.23	0.4	
Strobe "ON" Current	$I_{STR(ON)}$	-		-	3	-	mA
Output Leakage Current	I_{SINK}	I_{STR} =3mA, $V_I \geq$ 10mV V_O =15V, V_{CC}=±15V		-	0.2	50	nA
Input Voltage Range	$V_{I(R)}$	Note 1		-14.5 to 13.0	-14.7 to 13.8	-	V
Positive Supply Current	I_{CC}	-		-	3.0	7.5	mA
Negative Supply Current	I_{EE}	-		-	-2.2	-5.0	mA
Strobe Current	I_{STR}	-		-	3	-	mA

Notes :

1. $0 \leq T_A \leq +70°C$
2. The response time specified is for a 100mV input step with 5mV over drive.

Typical Performance Characteristics

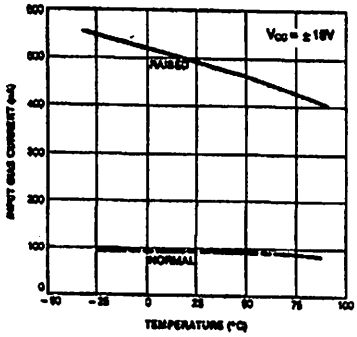


Figure 1. Input Bias Current vs Temperature

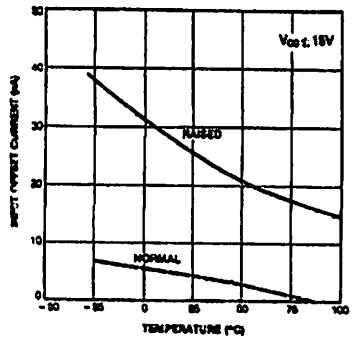


Figure 2. Input Offset Current vs Temperature

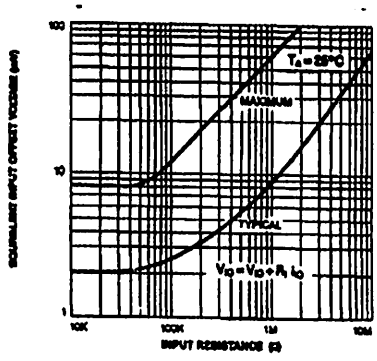


Figure 3. Offset Voltage vs Input Resistance

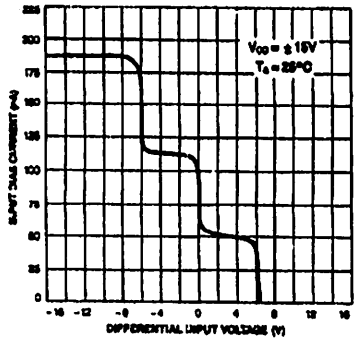


Figure 4. Input Bias Current vs Differential Input voltage

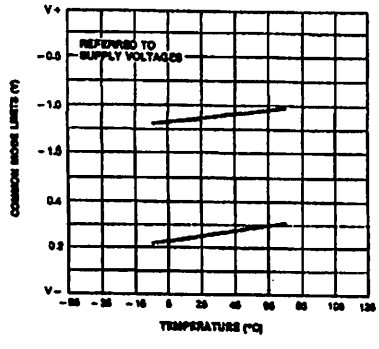


Figure 5. Common Mode Limits vs Temperature

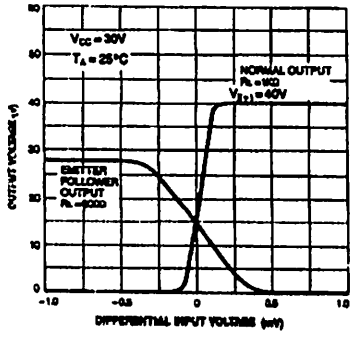


Figure 6. Output Voltage vs Differential Input voltage

Typical Performance Characteristics (continued)

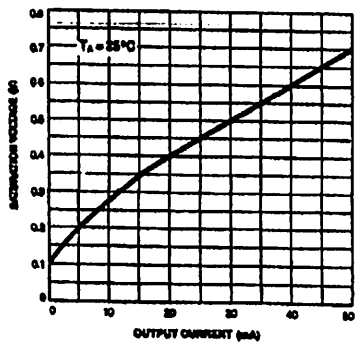


Figure 7. Saturation voltage vs Current

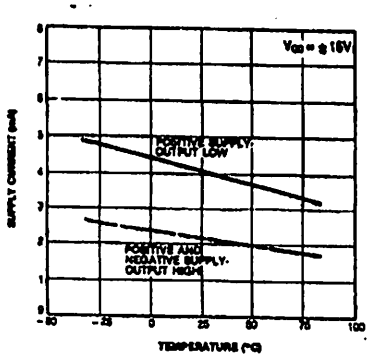


Figure 8. Supply Current vs Temperature

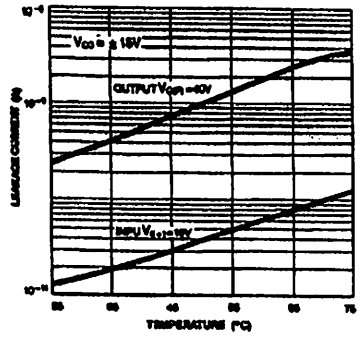


Figure 9. Leakage Current vs Temperature

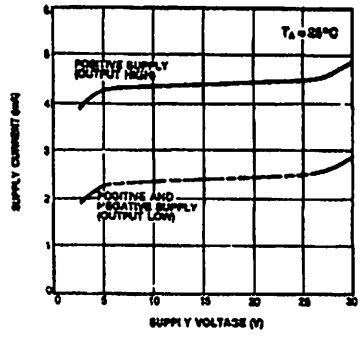


Figure 10. Supply Current vs Supply Voltage

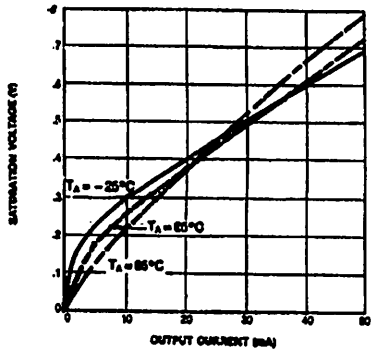


Figure 11. Current Saturation Voltage

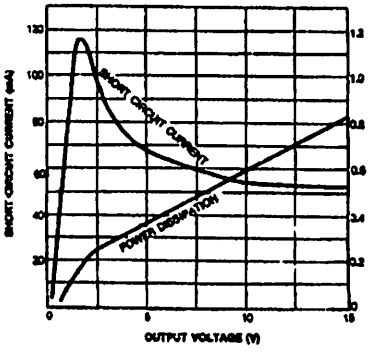
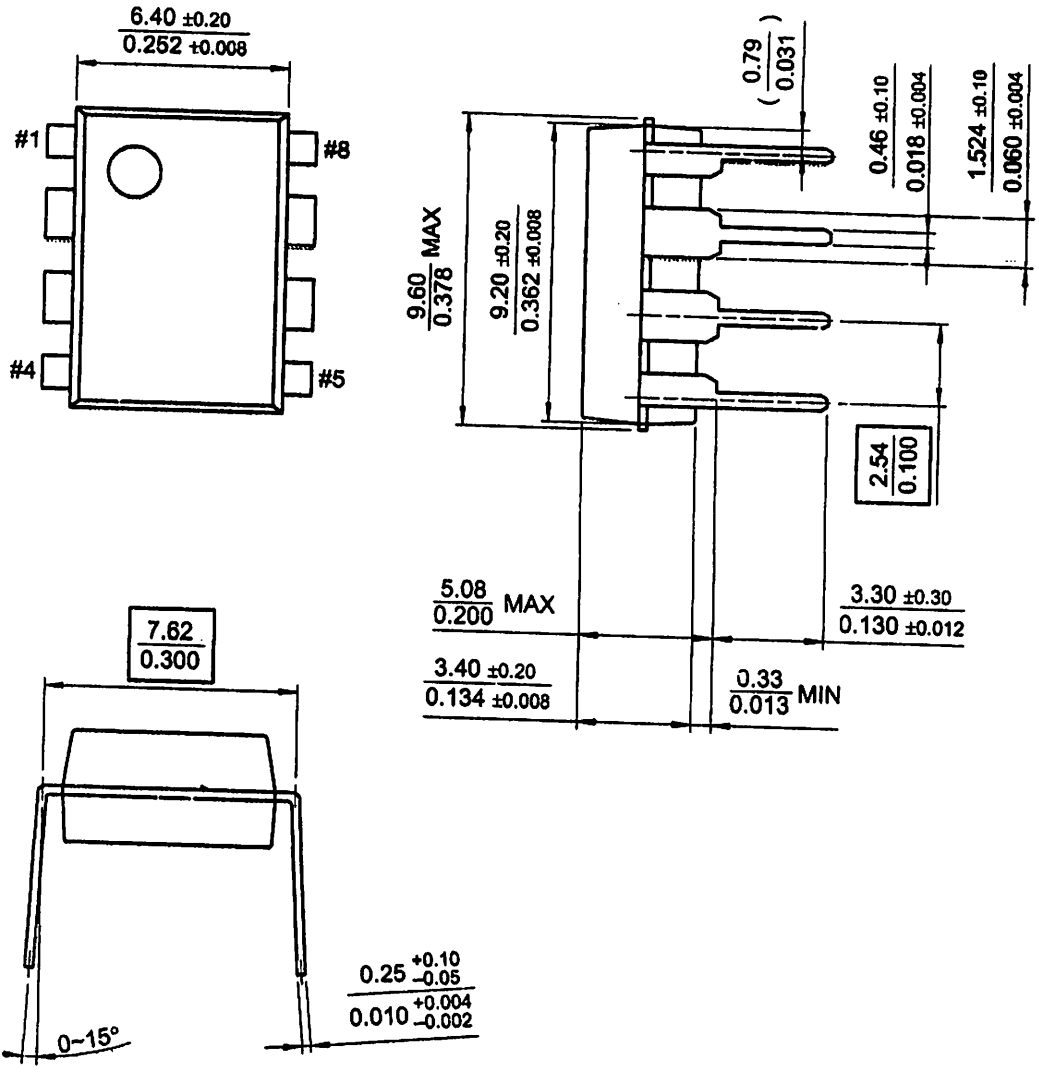


Figure 12. Output Limiting Characterstics

Mechanical Dimensions

Package

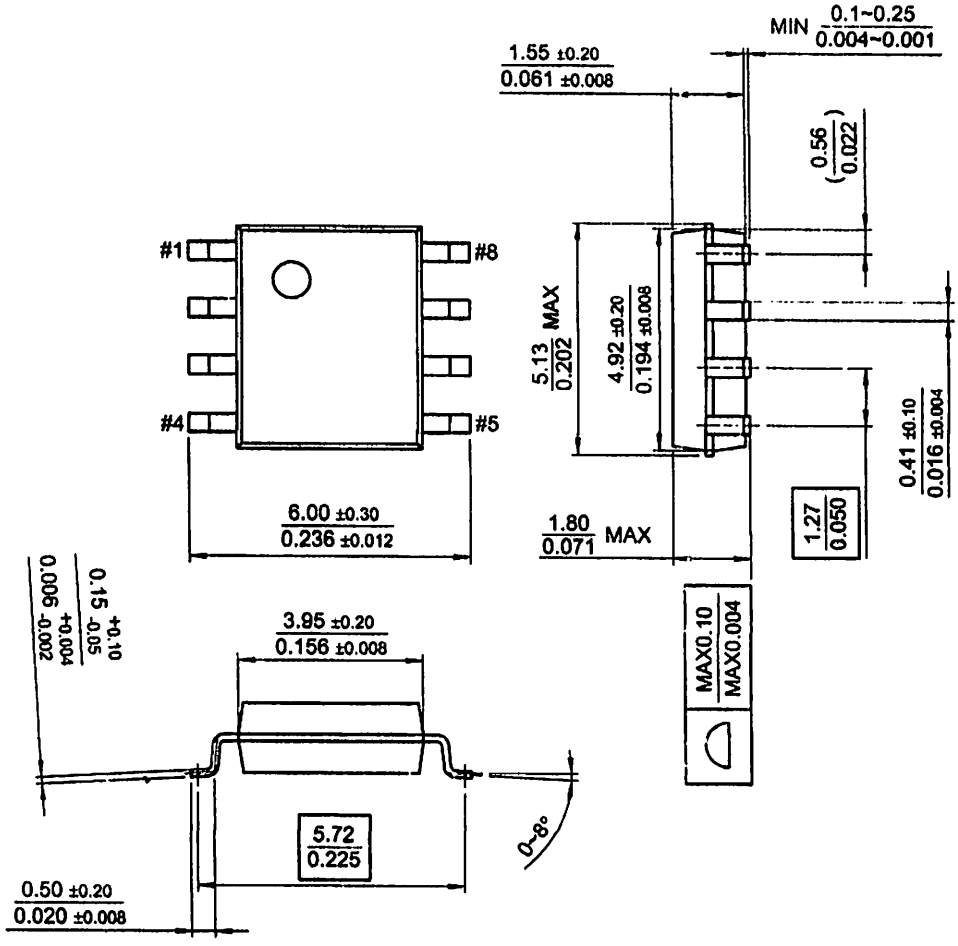
8-DIP



Mechanical Dimensions (Continued)

Package

8-SOP



Ordering Information

Product Number	Package	Operating Temperature
LM311N	8-DIP	0 ~ +70°C
LM311M	8-SOP	

6/1/01 0.0m 001
Stock#DSxxxxxxxx

ures

patible with MCS®51 Products
ytes of In-System Reprogrammable Downloadable Flash Memory
SPI Serial Interface for Program Downloading
Endurance: 1,000 Write/Erase Cycles
ytes EEPROM
Endurance: 100,000 Write/Erase Cycles
6V Operating Range
Static Operation: 0 Hz to 24 MHz
e-level Program Memory Lock
x 8-bit Internal RAM
rogrammable I/O Lines
e 16-bit Timer/Counters
Interrupt Sources
rammable UART Serial Channel
Serial Interface
-power Idle and Power-down Modes
rupt Recovery from Power-down
rammable Watchdog Timer
Data Pointer
er-off Flag

cription

T89S8252 is a low-power, high-performance CMOS 8-bit microcontroller with 8K of downloadable Flash programmable and erasable read-only memory and 2K of EEPROM. The device is manufactured using Atmel's high-density nonvolatile ry technology and is compatible with the industry-standard 80C51 instruction d pinout. The on-chip downloadable Flash allows the program memory to be rammed In-System through an SPI serial interface or by a conventional nonvol- nemory programmer. By combining a versatile 8-bit CPU with downloadable on a monolithic chip, the Atmel AT89S8252 is a powerful microcontroller, which es a highly-flexible and cost-effective solution to many embedded control ations.

T89S8252 provides the following standard features: 8K bytes of downloadable 2K bytes of EEPROM, 256 bytes of RAM, 32 I/O lines, programmable watchdog two data pointers, three 16-bit timer/counters, a six-vector two-level interrupt cture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, 89S8252 is designed with static logic for operation down to zero frequency and rts two software selectable power saving modes. The Idle Mode stops the CPU allowing the RAM, timer/counters, serial port, and interrupt system to continue ning. The Power-down mode saves the RAM contents but freezes the oscillator, ng all other chip functions until the next external interrupt or hardware reset.

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

ATMEL®

8-bit Microcontroller with 8K Bytes Flash

AT89S8252

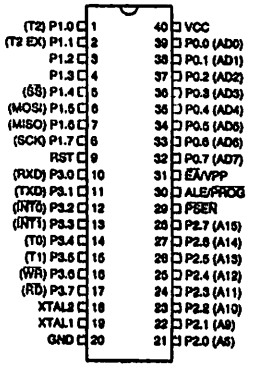
Not Recommended for New Designs. Use AT89S8253.

0401G–MICRO–3/06

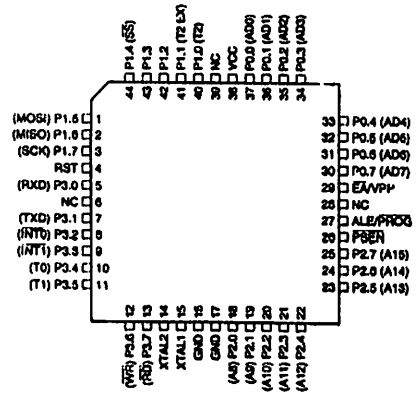
ATMEL

Configurations

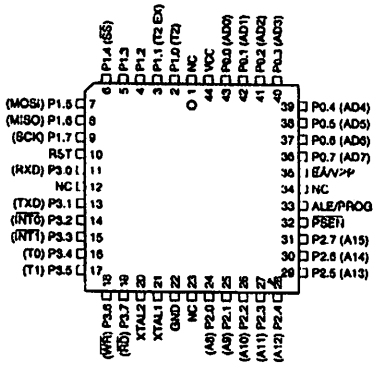
PDIP



TQFP



PLCC



Description

Supply voltage.

Ground.

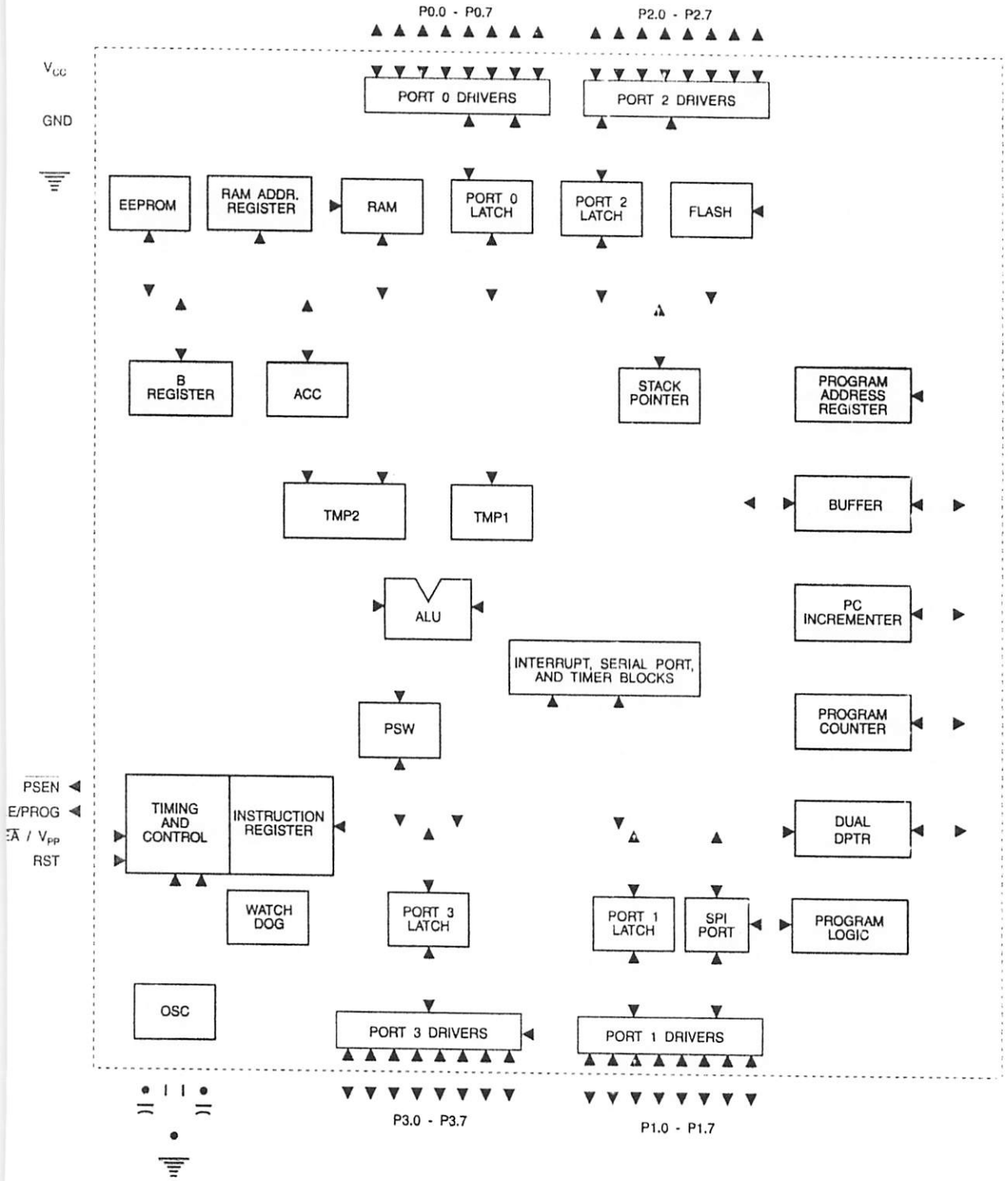
Port 0 is an 8-bit open drain bi-didirectional I/O port. As an output port, each pin can sink eight TTL inputs. When 1s are written to port 0 pins, the pins can be used as high-impedance inputs.

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 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.

k Diagram


P0.0 - P0.7
P2.0 - P2.7
V_{CC}
GND
PORT 0 DRIVERS
PORT 2 DRIVERS
EEPROM
RAM ADDR. REGISTER
RAM
PORT 0 LATCH
PORT 2 LATCH
FLASH
B REGISTER
ACC
STACK POINTER
PROGRAM ADDRESS REGISTER
TMP2
TMP1
BUFFER
ALU
PC INCREMENTER
INTERRUPT, SERIAL PORT, AND TIMER BLOCKS
PSW
PROGRAM COUNTER
PSEN
E/PROG
$\overline{EA}$ / V_{PP}
RST
TIMING AND CONTROL
INSTRUCTION REGISTER
DUAL DPTR
WATCH DOG
PORT 3 LATCH
PORT 1 LATCH
SPI PORT
PROGRAM LOGIC
OSC
PORT 3 DRIVERS
PORT 1 DRIVERS
P3.0 - P3.7
P1.0 - P1.7

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.

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.

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)

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

'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.

N

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.

'PP

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

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

-1

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

-2

Output from the inverting oscillator amplifier.

cial Function isters

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

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

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.

1. AT89S8252 SFR Map and Reset Values

								0FFH
B 00000000								0F7H
								0EFH
ACC 00000000								0E7H
								0DFH
PSW 00000000					SPCR 000001XX			0D7H
T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			0CFH
								0C7H
IP XX000C00								0BFH
P3 11111111								0B7H
IE 0X000000		SPSR 00XXXXXX						0AFH
P2 11111111								0A7H
SCON 00000000	SBUF XXXXXXXX							8FH
P1 11111111						WMCON 00000010		97H
TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000			8FH
P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR XXXXXXXX	PCON 0XXX0000	87H

2. T2CON – Timer/Counter 2 Control Register

)N Address = 0C8H — Reset Value = 0000 0000B

ddressable

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

ool	Function
	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.
2	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).
K	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.
K	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.
N2	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.
	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
	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).
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.

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

3. WMCON—Watchdog and Memory Control Register

:ON Address = 96H

Reset Value = 0000 0010B

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

›ol	Function
	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.
WE	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.
EN	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.
	Data Pointer Register Select. DPS = 0 selects the first bank of Data Pointer Register, DP0, and DPS = 1 selects the second bank, DP1
RST $\overline{BSY}$	Watchdog Timer Reset and EEPROM Ready/$\overline{Busy}$ 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/$\overline{BSY}$ flag in a Read-Only mode during EEPROM write. RDY/$\overline{BSY}$ = 1 means that the EEPROM is ready to be programmed. While programming operations are being executed, the RDY/$\overline{BSY}$ bit equals "0" and is automatically reset to "1" when programming is completed.
EN	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.

4. SPCR – SPI Control Register

R Address = D5H

Reset Value = 0000 01XXB

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

bol	Function
:	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.
	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.
ID	Data Order. DORD = 1 selects LSB first data transmission. DORD = 0 selects MSB first data transmission.
R	Master/Slave Select. MSTR = 1 selects Master SPI mode. MSTR = 0 selects Slave SPI mode.
L	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.
A	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.
0 1	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

5. SPSR – SPI Status Register

R Address = AAH

Reset Value = 00XX XXXXB

SPIF	WCOL	–	–	–	–	–	–
7	6	5	4	3	2	1	0

ool	Function
	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.
)L	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.

6. SPDR – SPI Data Register

R Address = 86H

Reset Value = unchanged

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

a Memory – ROM 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.

grammable chdog 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

r 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".

r 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.

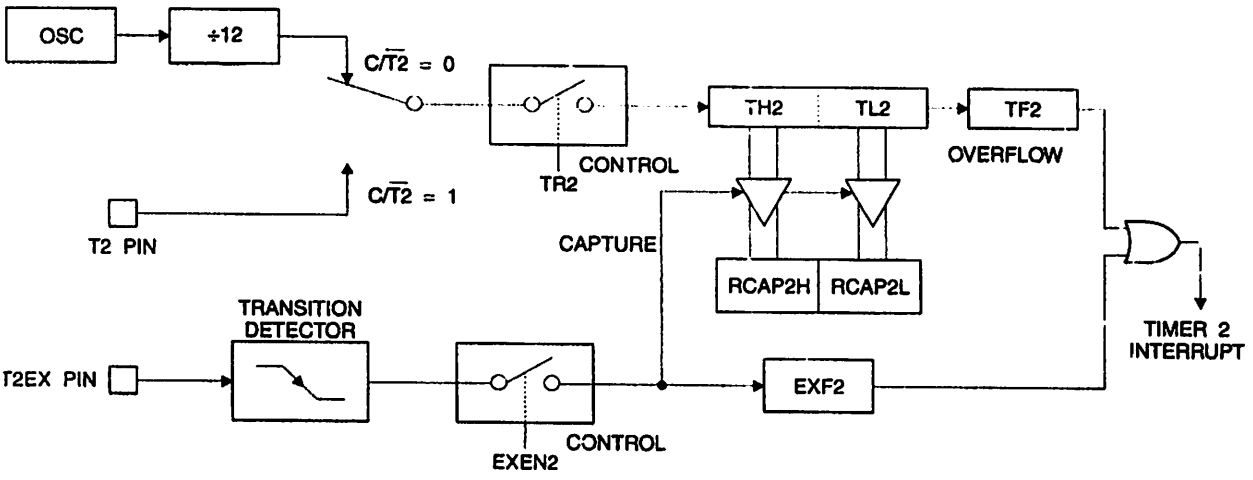
Table 8. Timer 2 Operating Modes

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)

ture Mode

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

'e 1. Timer 2 in Capture Mode



-reload (Up or Down nter)

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.

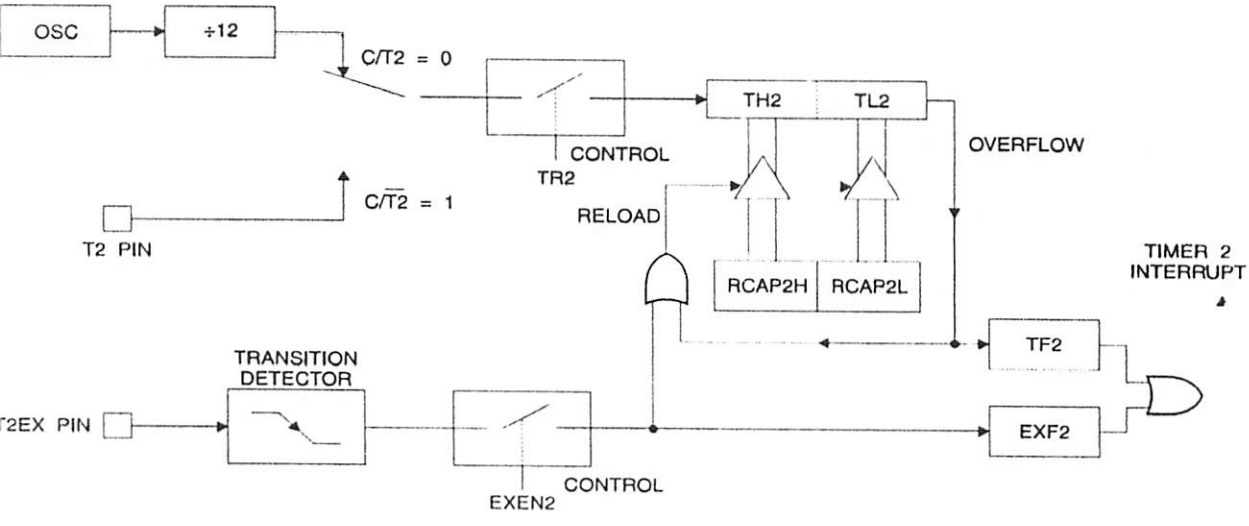
Figure 2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H 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.

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



9. T2MOD – Timer 2 Mode Control Register

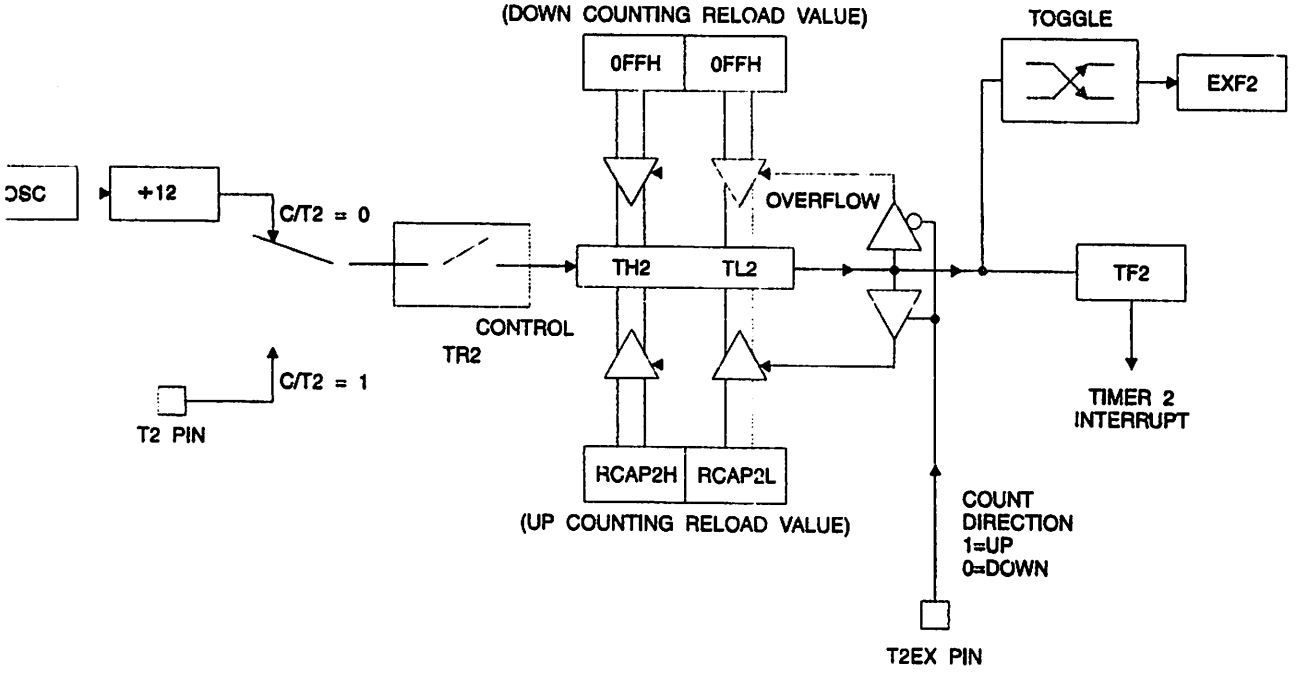
OD Address = 0C9H — Reset Value = XXXX XX00B

Bit Addressable

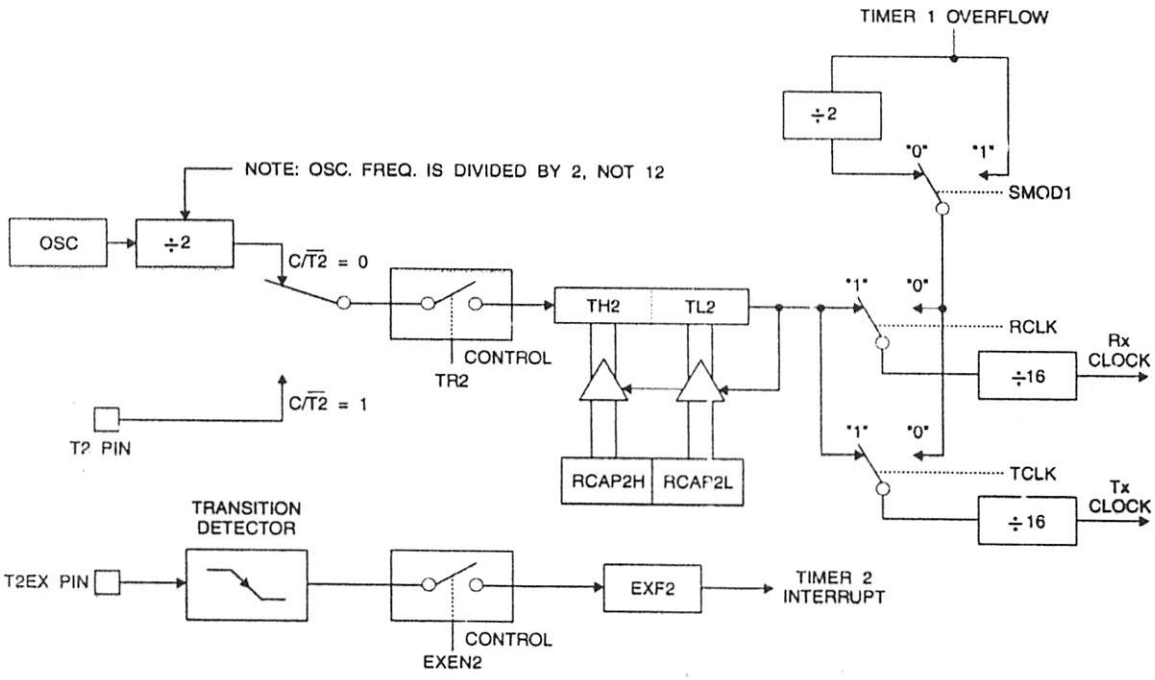
–	–	–	–	–	–	T2OE	DCEN
7	6	5	4	3	2	1	0

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

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



e 4. Timer 2 in Baud Rate Generator Mode



id 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.

grammable ck 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).

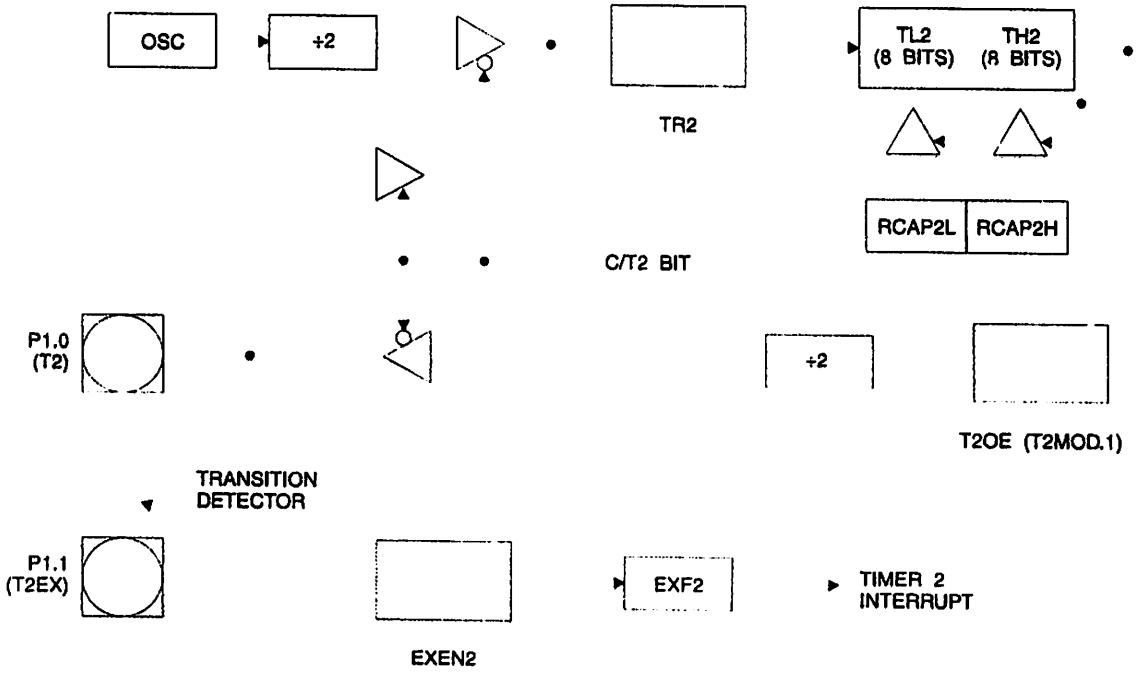
To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

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.

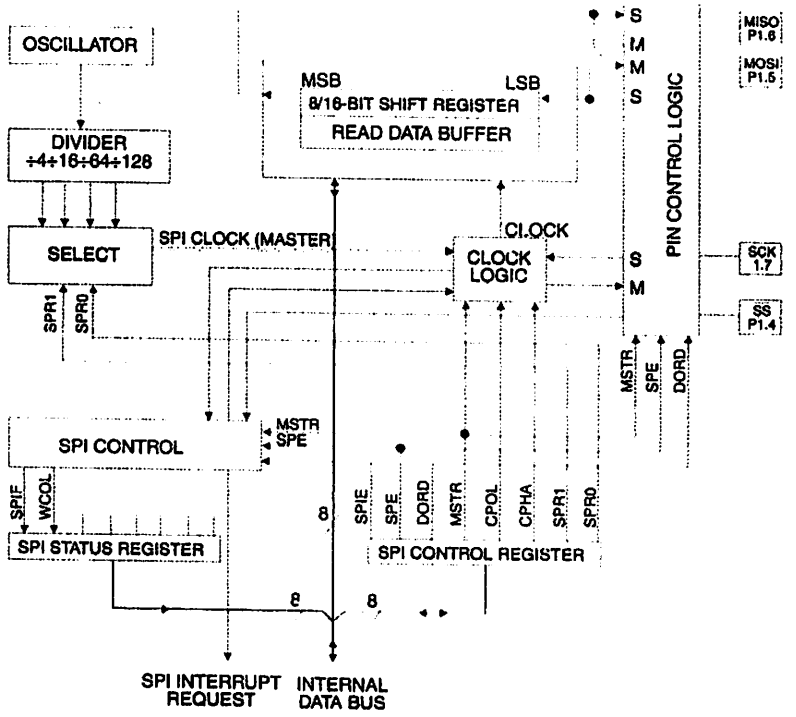
$$\text{Clock Out Frequency} = \frac{\text{Oscillator Frequency}}{4 \times [65536 - (\text{RCAP2H,RCAP2L})]}$$

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.

re 5. Timer 2 in Clock-out Mode



re 6. SPI Block Diagram



RT

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".

ial Peripheral rface

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:

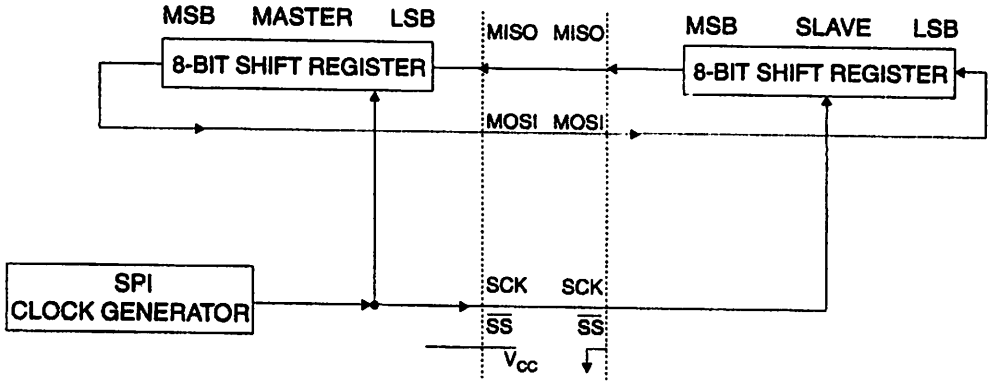
- Full-Duplex, 3-Wire Synchronous Data Transfer
- Master or Slave Operation
- 1.5 MHz Bit Frequency (max.)
- LSB First or MSB First Data Transfer
- Four Programmable Bit Rates
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Wakeup from Idle Mode (Slave Mode Only)

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.

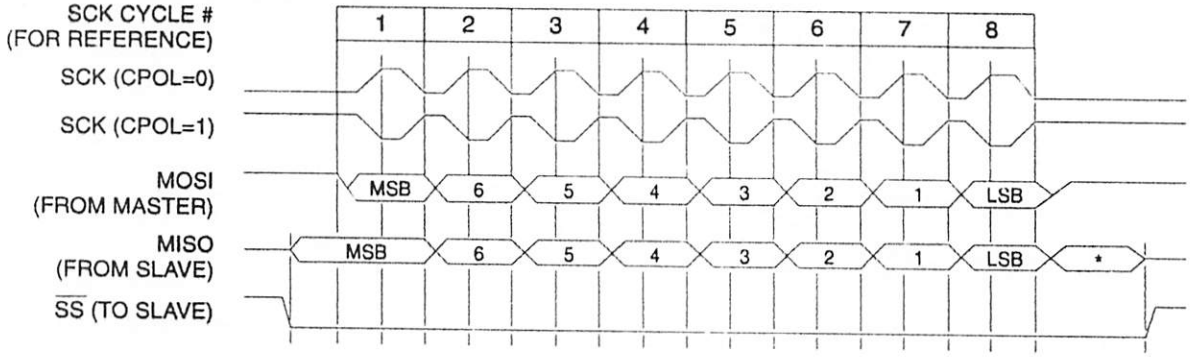
The Slave Select input, $\overline{SS}$/P1.4, is set low to select an individual SPI device as a slave. When $\overline{SS}$/P1.4 is set high, the SPI port is deactivated and the MOSI/P1.5 pin can be used as an input.

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.

re 7. SPI Master-slave Interconnection

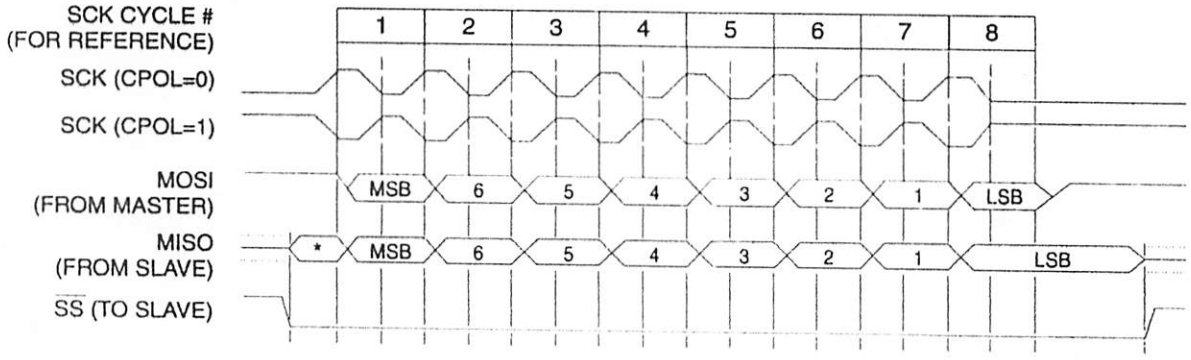


re 8. SPI transfer Format with CPHA = 0



*Not defined but normally MSB of character just received

re 9. SPI Transfer Format with CPHA = 1



*Not defined but normally LSB of previously transmitted character.

errupts

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.

e 10. Interrupt Enable (IE) Register

B)(LSB)

EA	–	ET2	ES	ET1	EX1	ET0	EX0

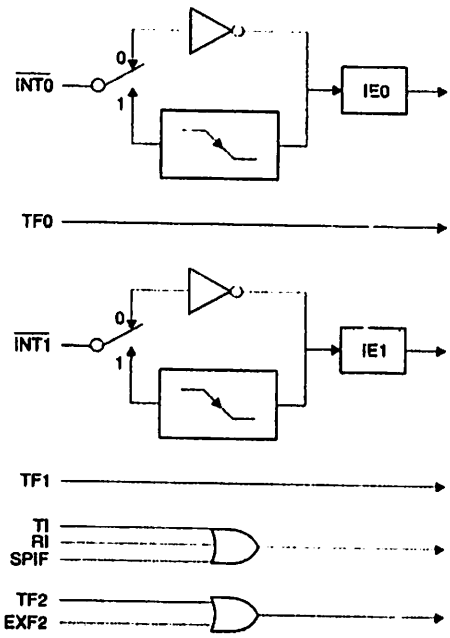
Enable Bit = 1 enables the interrupt.

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.

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

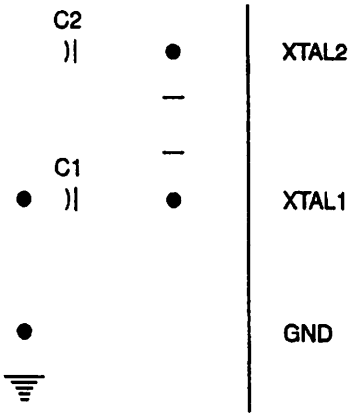
re 10. Interrupt Sources



illator racteristics

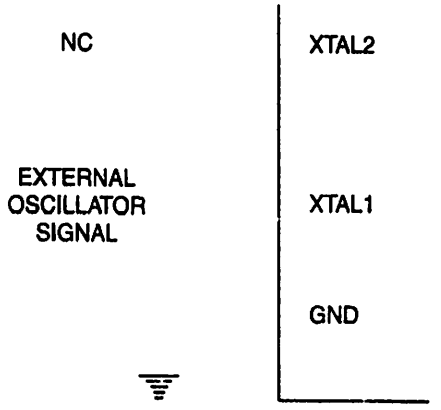
XTAL1 and XTAL2 are the input and output, respectively, of an invertirg amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 11. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected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



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

?er-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.

;ram Memory ‹ 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.

Bit Protection Modes[1][2]

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

1. U = Unprogrammed
2. P = Programmed

gramming the sh 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 $\overline{\text{DATA}}$ 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. $\overline{\text{DATA}}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The progress of byte programming in the parallel programming mode can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. Pin P3.4 is pulled Low after ALE goes High during programming to indicate $\overline{\text{BUSY}}$. 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/$\overline{\text{PROG}}$ 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

gramming
rface

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.

al 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.

ıl Programming rithm

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.

al Programming
uction

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

uction Set

uction	Input Format			Operation
	Byte 1	Byte 2	Byte 3	
ramming Enable	1010 1100	0101 0011	xxxx xxxx	Enable serial programming interface after RST goes high.
Erase	1010 1100	xxxx x100	xxxx xxxx	Chip erase both 8K & 2K memory arrays.
d 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.
e 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 togetner with the second byte.
d 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.
e Data Memory	00aa a110	low addr	data in	Write data to Data memory location at selected address.
e Lock Bits	1010 1100	LB1 LB2 LB3 x x111	xxxx xxxx	Write lock bits. Set LB1, LB2 or LB3 = "0" to program lock bits.

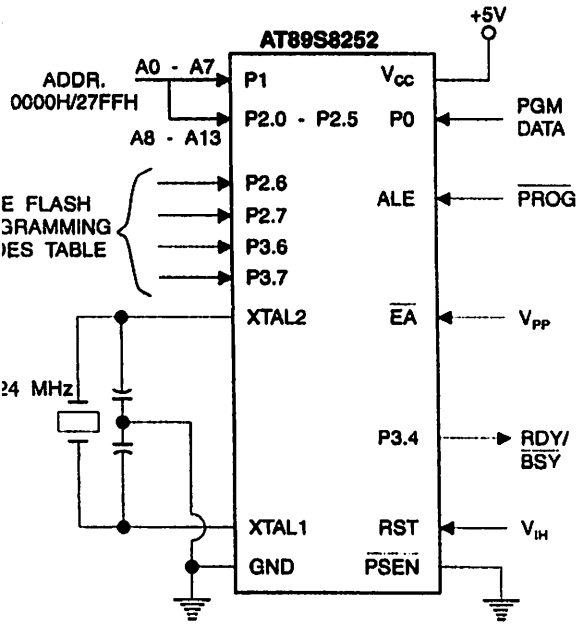
1. DATA polling is used to indicate the end of a byte write cycle which typically takes less than 2.5 ms at 5V.
2. "aaaaa" = high order address.
3. "x" = don't care.

h and EEPROM Parallel Programming Modes

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

1. "h" = weakly pulled "High" internally.
2. Chip Erase and Serial Programming Fuse require a 10 ms 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/BSY.
4. "X" = don't care

13. Programming the Flash/EEPROM Memory



e 14. Verifying the Flash/EEPROM Memory

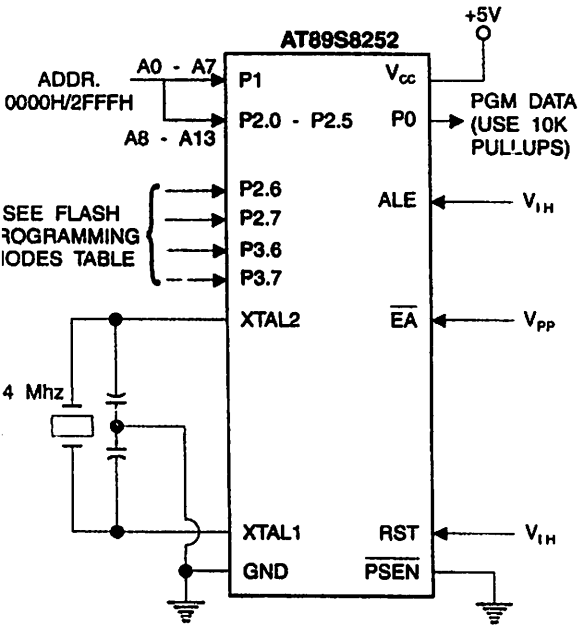
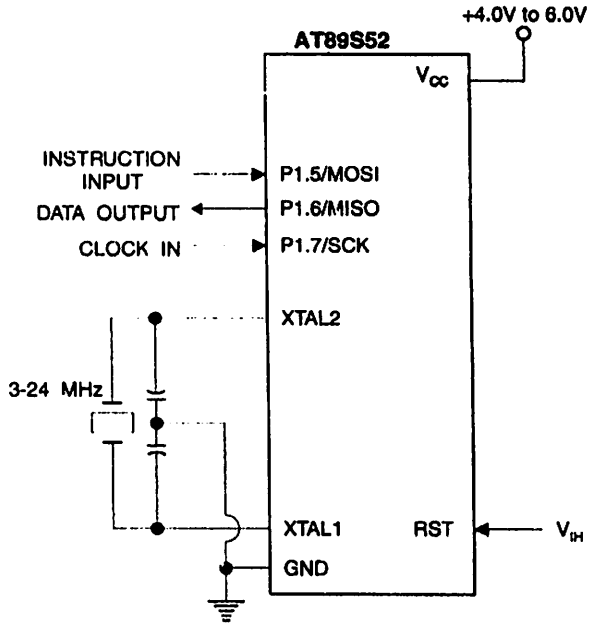


Figure 15. Flash/EEPROM Serial Downloading

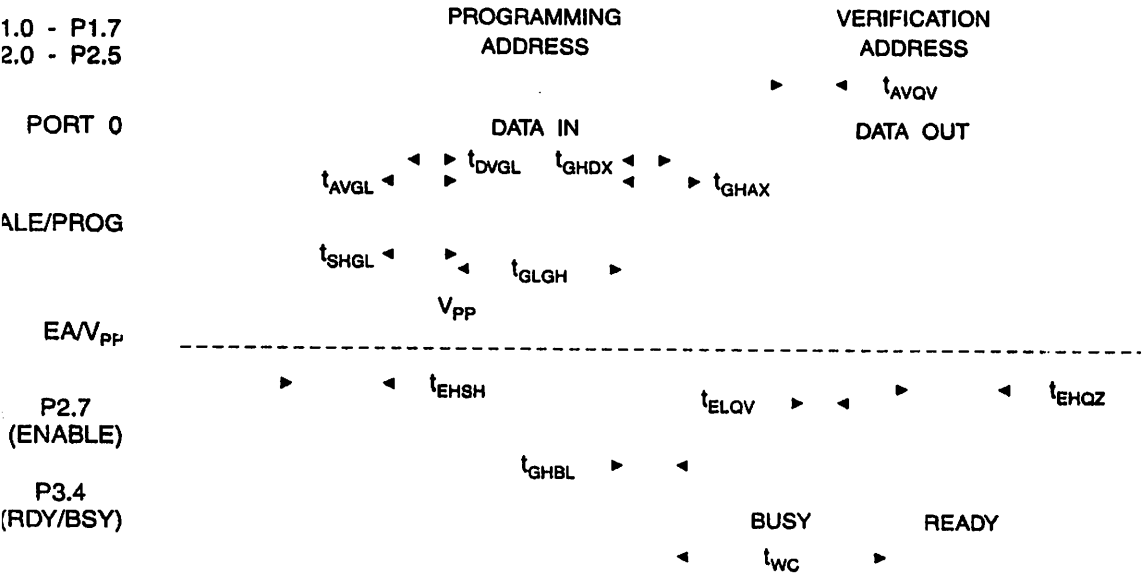


h Programming and Verification Characteristics – Parallel Mode

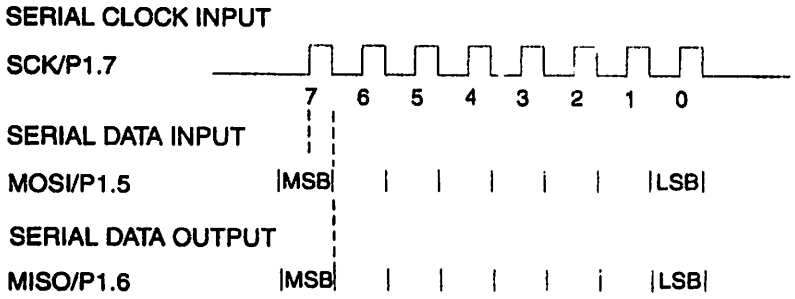
°C to 70°C, V_{CC} = 5.0V ± 10%

ol	Parameter	Min	Max	Units
	Programming Enable Voltage	11.5	12.5	V
	Programming Enable Current		1.0	mA
L	Oscillator Frequency	3	24	MHz
	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
	Address Hold after $\overline{PROG}$	$48t_{CLCL}$		
	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
	Data Hold after $\overline{PROG}$	$48t_{CLCL}$		
	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
	$\overline{PROG}$ Width	1	110	µs
	Address to Data Valid		$48t_{CLCL}$	
	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
	Data Float after $\overline{ENABLE}$	0	$48t_{CLCL}$	
	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
	Byte Write Cycle Time		2.0	ms

h/EEPROM Programming and Verification Waveforms – Parallel Mode

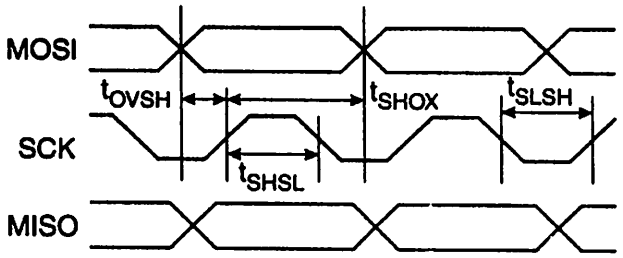


al Downloading Waveforms



al Programming Characteristics

e 16. Serial Programming Timing



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

bol	Parameter	Min	Typ	Max	Units
CL	Oscillator Frequency	0		24	MHz
L	Oscillator Period	41.6			ns
L	SCK Pulse Width High	24 t_{CLCL}			ns
H	SCK Pulse Width Low	24 t_{CLCL}			ns
H	MOSI Setup to SCK High	t_{CLCL}			ns
X	MOSI Hold after SCK High	2 t_{CLCL}			ns

olute Maximum Ratings*

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

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

Characteristics

alues shown in this table are valid for T_A = -40°C to 85°C and V_{CC} = 5.0V ± 20%, unless otherwise noted.

bol	Parameter	Condition	Min	Max	Units
	Input Low-voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC} - 0.1	V
	Input Low-voltage ($\overline{EA}$)		-0.5	0.2 V_{CC} - 0.3	V
	Input High-voltage	(Except XTAL1, RST)	0.2 V_{CC} + 0.9	V_{CC} + 0.5	V
	Input High-voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC} + 0.5	V
	Output Low-voltage (1) (Ports 1,2,3)	I_{OL} = 1.6 mA		0.5	V
	Output Low-voltage (1) (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.5	V
	Output High-voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
	Output High-voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-650	µA
	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
T	Reset Pull-down Resistor		50	300	KΩ
	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
	Power Supply Current	Active Mode, 12 MHz		25	mA
		Idle Mode, 12 MHz		6.5	mA
	Power-down Mode (2)	V_{CC} = 6V		100	µA
		V_{CC} = 3V		40	µA

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

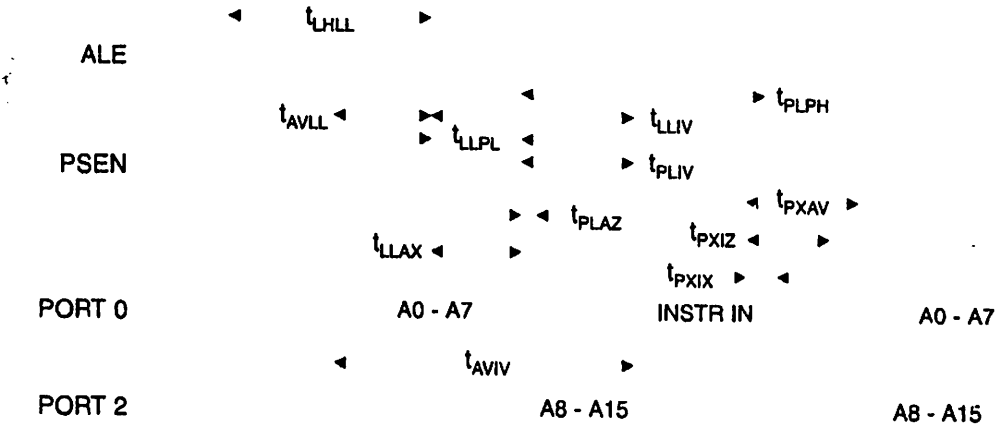
Characteristics

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

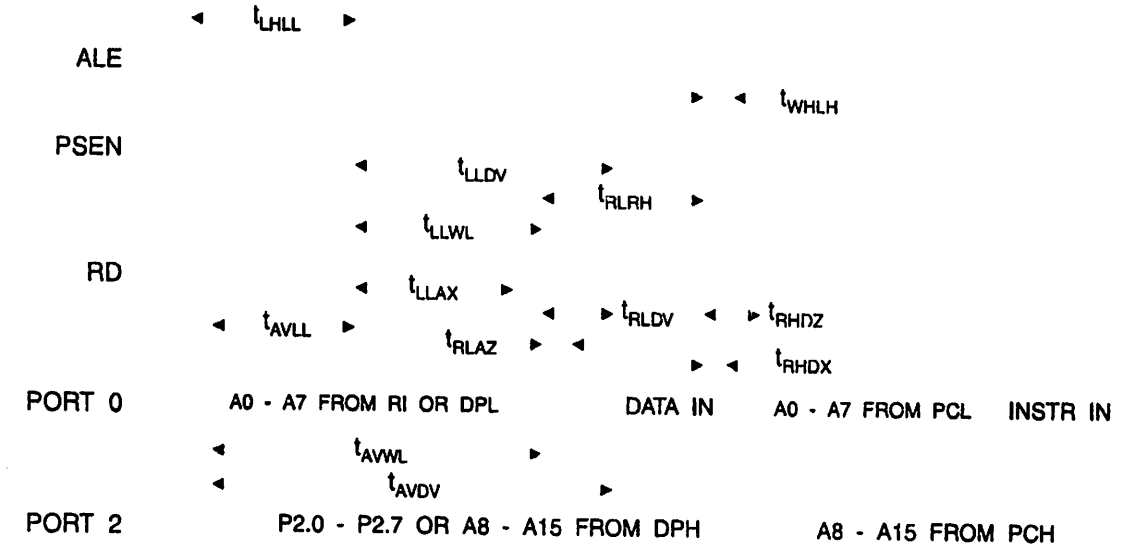
nal Program and Data Memory Characteristics

ol	Parameter	Variable Oscillator		Units
		Min	Max	
L	Oscillator Frequency	0	24	MHz
	ALE Pulse Width	$2t_{CLCL}$ - 40		ns
	Address Valid to ALE Low	t_{CLCL} - 13		ns
	Address Hold after ALE Low	t_{CLCL} - 20		ns
	ALE Low to Valid Instruction In		$4t_{CLCL}$ - 65	ns
	ALE Low to $\overline{PSEN}$ Low	t_{CLCL} - 13		ns
	$\overline{PSEN}$ Pulse Width	$3t_{CLCL}$ - 20		ns
	$\overline{PSEN}$ Low to Valid Instruction In		$3t_{CLCL}$ - 45	ns
	Input Instruction Hold after $\overline{PSEN}$	0		ns
	Input Instruction Float after $\overline{PSEN}$		t_{CLCL} - 10	ns
	$\overline{PSEN}$ to Address Valid	t_{CLCL} - 8		ns
	Address to Valid Instruction In		$5t_{CLCL}$ - 55	ns
	$\overline{PSEN}$ Low to Address Float		10	ns
	$\overline{RD}$ Pulse Width	$6t_{CLCL}$ - 100		ns
	$\overline{WR}$ Pulse Width	$6t_{CLCL}$ - 100		ns
	$\overline{RD}$ Low to Valid Data In		$5t_{CLCL}$ - 90	ns
	Data Hold after $\overline{RD}$	0		ns
	Data Float after $\overline{RD}$		$2t_{CLCL}$ - 28	ns
	ALE Low to Valid Data In		$8t_{CLCL}$ - 150	ns
	Address to Valid Data In		$9t_{CLCL}$ - 165	ns
	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	$3t_{CLCL}$ - 50	$3t_{CLCL}$ + 50	ns
	Address to $\overline{RD}$ or $\overline{WR}$ Low	$4t_{CLCL}$ - 75		ns
	Data Valid to $\overline{WR}$ Transition	t_{CLCL} - 20		ns
	Data Valid to $\overline{WR}$ High	$7t_{CLCL}$ - 120		ns
	Data Hold after $\overline{WR}$	t_{CLCL} - 20		ns
	$\overline{RD}$ Low to Address Float		0	ns
	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{CLCL} - 20	t_{CLCL} + 25	ns

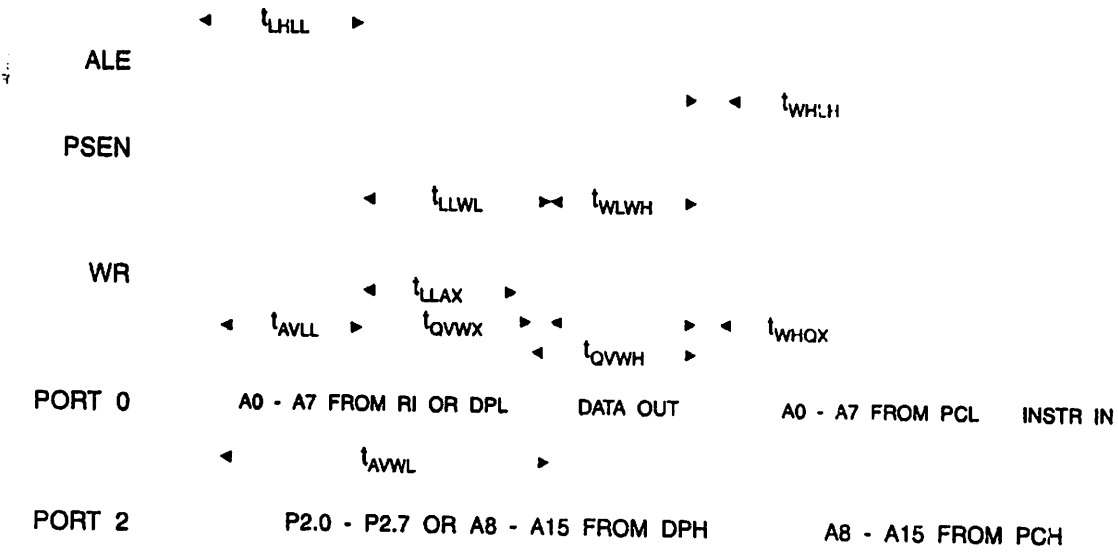
ernal Program Memory Read Cycle



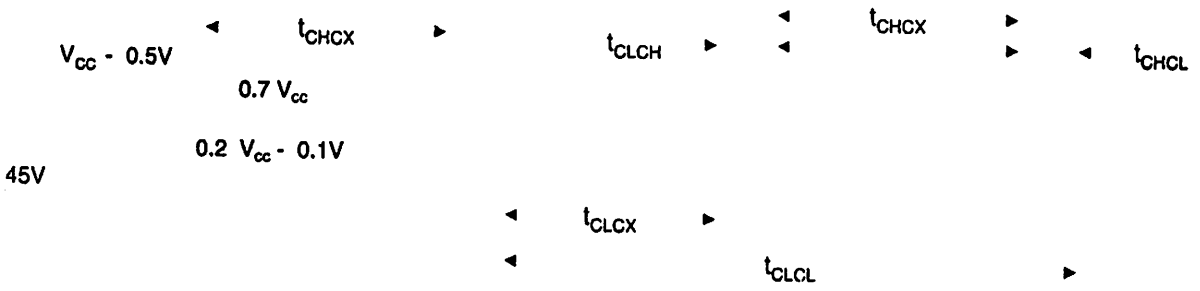
ernal Data Memory Read Cycle



ernal Data Memory Write Cycle



ernal Clock Drive Waveforms



ernal Clock Drive

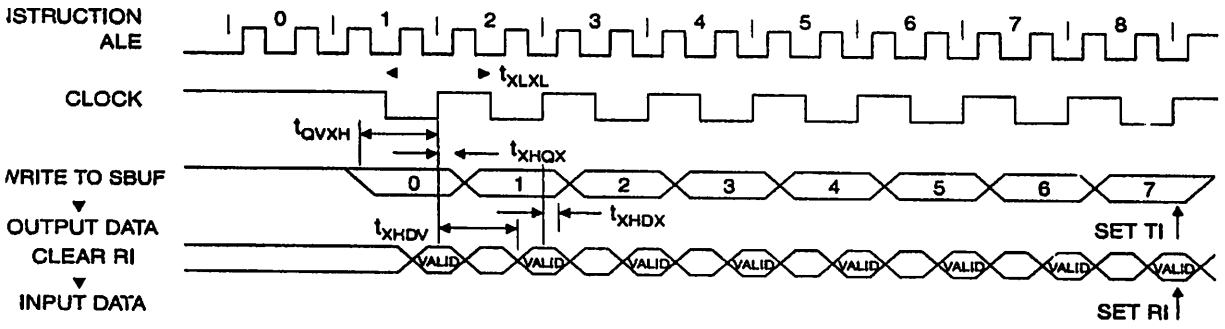
ool	Parameter	V_{CC} = 4.0V to 6.0V		Units
		Min	Max	
:L	Oscillator Frequency	0	24	MHz
	Clock Period	41.6		ns
	High Time	15		ns
	Low Time	15		ns
	Rise Time		20	ns
	Fall Time		20	ns

al Port Timing: Shift Register Mode Test Conditions

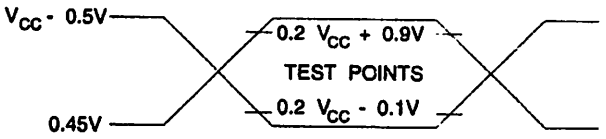
alues in this table are valid for V_{CC} = 4.0V to 6V and Load Capacitance = 80 pF.

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

t Register Mode Timing Waveforms

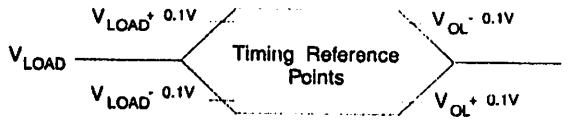


Testing Input/Output Waveforms[1]



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

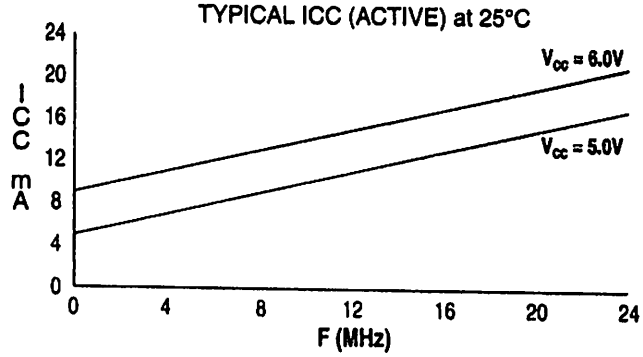
t Waveforms[1]



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

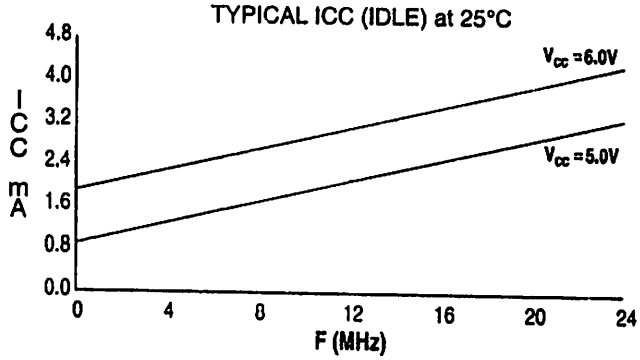
AT89S8252

TYPICAL ICC (ACTIVE) at 25°C



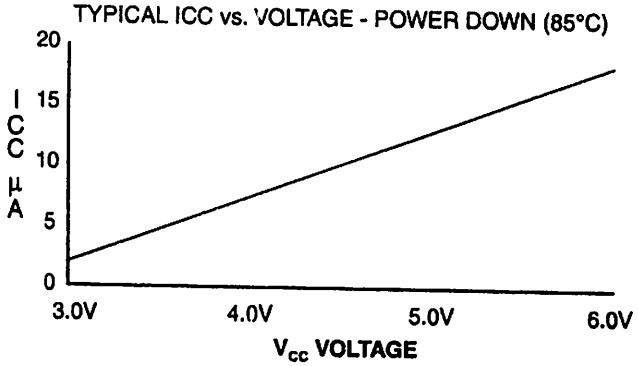
AT89S8252

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

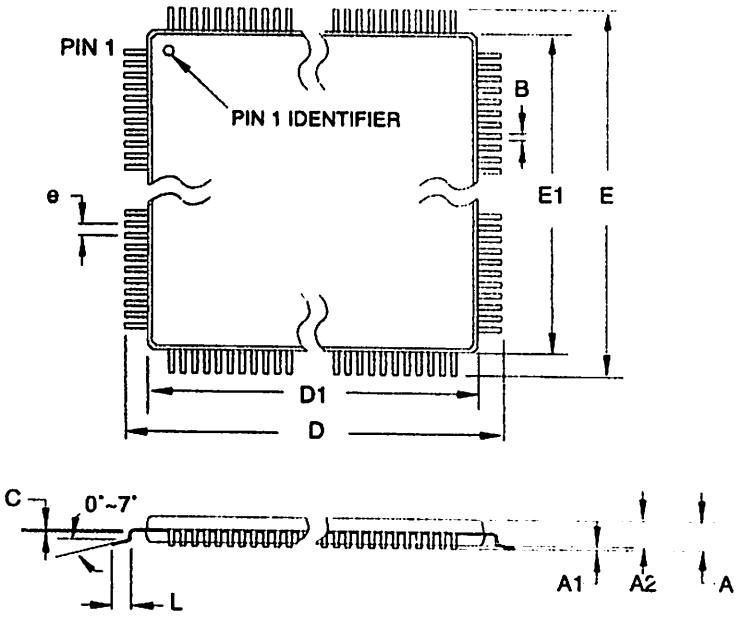
ering Information

eed Hz)	Power Supply	Ordering Code	Package	Operation Range
:4	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
	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
	44-lead, Plastic J-leaded Chip Carrier (PLCC)
	40-lead, 0.600" Wide, Plastic Dual Inline Package (PDIP)

kaging Information

– TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

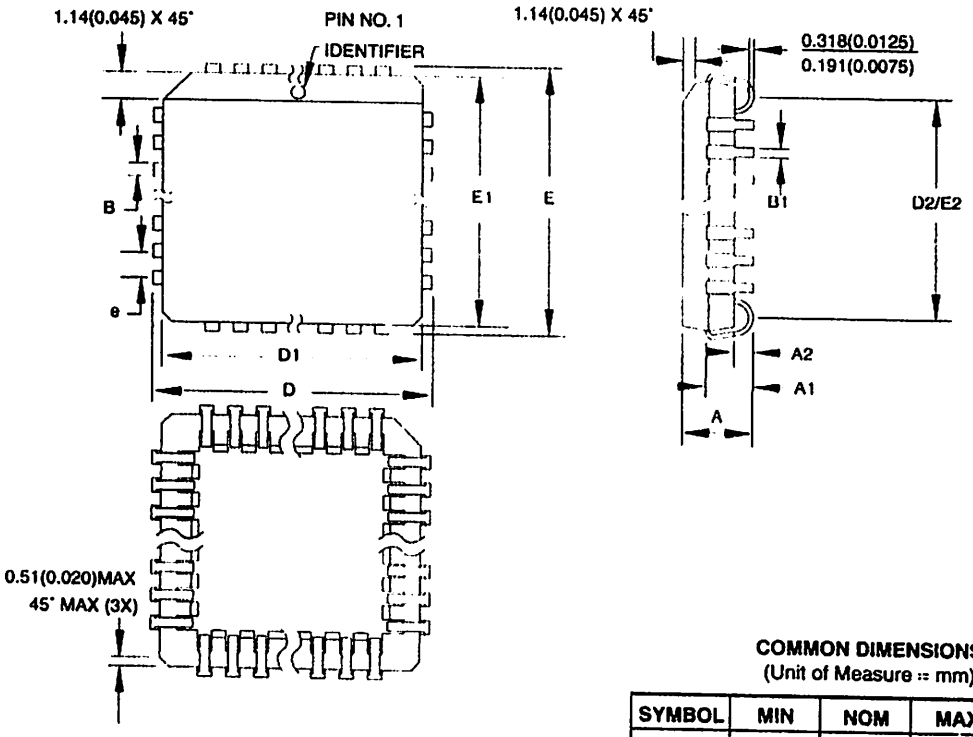
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

lotes:
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

– PLCC



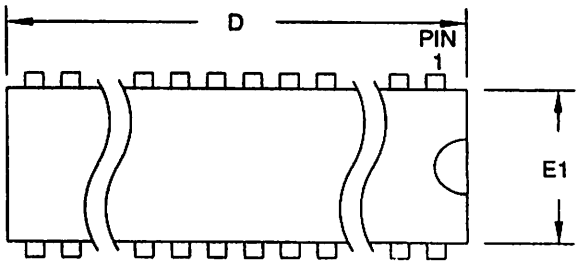
COMMON DIMENSIONS
(Unit of Measure = mm)

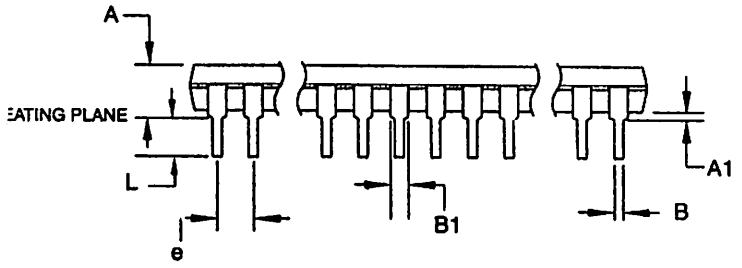
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

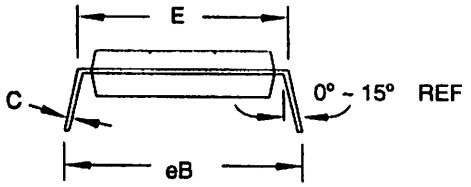
Notes:
1. This package conforms to JEDEC reference MS-018, Variation AC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is .010"(0.254 mm) per side. Dimension D1 and E1 include mold mismatch and are measured at the extreme material condition at the upper or lower parting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







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

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

el Corporation

:5 Orchard Parkway
ı Jose, CA 95131, USA
: 1(408) 441-0311
:: 1(408) 487-2600

ional Headquarters

pe
ıel Sarl
ıte des Arsenaux 41
ɜe Postale 80
-1705 Fribourg
itzerland
: (41) 26-426-5555
:: (41) 26-426-5500

ɔm 1219
nachem Golden Plaza
Mody Road Tsimshatsui
ɜt Kowloon
ıg Kong
: (852) 2721-9778
:: (852) 2722-1369

ı
Tonetsu Shinkawa Bldg.
4-8 Shinkawa
ɹo-ku, Tokyo 104-0033
ɐan
: (81) 3-3523-3551
:: (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