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

SKRIPSI

ALAT PELACAK POSISI DENGAN GPS DAN PENONAKTIFAN MESIN MOBIL DENGAN KENDALI HANDPHONE PADA PERUSAHAAN PERSEWAAN MOBIL

Oleh :
MUHAMMAD INDRA WIJAYA
NIM : 02.17.066

SEPTEMBER 2007

LEMBAR PERSETUJUAN

ALAT PELACAK POSISI DENGAN GPS DAN PENONAKTIFAN MESIN MOBIL DENGAN KENDALI HANDPHONE PADA PERUSAHAAN PERSEWAAN MOBIL

SKRIPSI

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

Disusun oleh :

MUHAMMAD INDRA WIJAYA
NIM : 02.17.066

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
Joseph Dedy Irawan,ST, MT	**M. Ashar, ST, MT**
NIP. 132315178	**NIP.P. 1030500408**

Mengetahui
Ketua Jurusan Teknik Elektro S-1

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

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

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

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa : Muhammad Indra Wijaya
NIM : 02.17.066
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Elektronika
Judul Skripsi : "ALAT PELACAK POSISI DENGAN GPS DAN PENONAKTIFAN MESIN MOBIL DENGAN KENDALI HANDPHONE PADA PERUSAHAAN PERSEWAAN MOBIL"

Dipertahankan dihadapan Tim Penguji Skripsi Jenjang Strata Satu (S-1) pada:
Hari : Senin
Tanggal : 3 September 2007
Dengan Nilai : 92,4 (A)

Panitia Ujian Skripsi

Ketua	Sekretaris
(Ir. Mochtar Asroni, MSME)	**(Ir.F.Yudi Limpraptono, MT)**
NIP.Y. 1018100036	**NIP.Y. 1039500274**

Anggota Penguji

Penguji Pertama	Penguji Kedua
(Ir. Widodo Pudji M., MT.)	**(DR. Cahyo Crysdian, Msc)**
NIP.Y. 102870017	**NIP. 1030400412**

ALAT PELACAK POSISI DENGAN GPS DAN PENONAKTIFAN MESIN MOBIL DENGAN KENDALI HANDPHONE PADA PERUSAHAAN PERSEWAAN MOBIL

M. Indra Wijaya

Jurusan Teknik Elektronika, Fakultas Teknologi Industri, Institut Teknologi Nasional,

Jl. Bendungan Sigura-gura No. 2 Malang, *indr@programmer.net*

Abstrak

Umumnya perusahaan persewaan mobil menyewakan mobil pada pihak penyewa dengan persyaratan yang mudah sehingga banyak pihak penyewa yang berminat menyewa. Sayangnya kemudahan tersebut banyak dimanfaatkan oleh pihak-pihak yang tidak bertanggung jawab untuk melakukan tindak pencurian. Untuk meminimalkan resiko terjadinya tindak pencurian oleh pihak penyewa maka dibuat alat yang dapat melacak posisi dari mobil yang disewakan dan dapat menonaktifkan mesin mobil apabila terjadi tindak pencurian. Alat ini dibuat dengan menggunakan GPS untuk melacak posisi dan handphone digunakan untuk mengirimkan data kooordinat dari GPS dan sebagai pengendali penonaktifan mesin mobil dari jarak jauh. Untuk menampilkan posisi mobil yang didapat oleh GPS digunakan program Google Earth. Dengan alat ini maka pihak perusahaan tidak khawatir akan resiko kehilangan mobil karena pencarian dengan Google Earth cukup akurat. Dan untuk pihak penyewa akan lebih mudah karena ada informasi tentang masa persewaannya bila akan berakhir.

Abstract

Generally company of car's rental, rent the car on the side of lodger's with easy clauses so that many enthusiastic lodger's rent. Unhappily the amenity exploited many by irresponsible's to act the theft. For the minimization of risk the happening of acting the theft by lodger's, hence made appliance able to trace position of rented car and can deactivate the car machine in the event of acting the theft. This appliance is made by using GPS to trace the position and cell phone used to deliver the data from GPS and as controller to deactivate car machine from long distance. Google Earth used to present the car position got by GPS. With this application company not worry with risk to lose the car's because searching position with Google Earth is accurate. And the lodger's will easier because information about rent status already present.

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat menyelesaikan skripsi yang berjudul "Alat Pelacak Posisi Dengan GPS Dan Penonaktifan Mesin Mobil Dengan Kendali Handphone Pada Perusahaan Persewaan Mobil" ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Elektronika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

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

1. Bapak Prof. DR. Ir. Abraham Lomi, MSEE selaku Rektor ITN Malang.
2. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
3. Bapak Joseph Dedy Irawan, ST, MT selaku Dosen Pembimbing I.
4. Bapak M Ashar, ST, MT selaku Dosen Pembimbing II.
5. Ayah dan Ibu serta keluarga besar yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
6. Rekan-rekan mahasiswa/i Elektronika 2002 yang telah memberikan semangat dan motivasi dalam penyelesaian skripsi ini.
7. Semua pihak yang telah membantu dalam penyelesaian penyusunan skripsi ini.

Penulis telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelasaikan laporan ini. Untuk itu penyusun mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

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

Malang, September 2007

Penulis

DAFTAR ISI

BAB III. PERANCANGAN DAN PEMBATAN ALAT

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Menjamurnya usaha persewaan mobil saat ini, mengakibatkan tingkat persaingan semakin tinggi. Sehingga tidak sedikit perusahaan persewaan mobil yang memberikan kemudahan-kemudahan dalam menyewakan mobil, seperti tarif yang murah dan persyaratan yang mudah. Karena tarif yang murah dan persyaratan yang mudah ini, maka Perusahaan persewaan mobil seperti ini menjadi sasaran empuk bagi sindikat pencurian mobil. Oleh karena itu sistem ini akan kami aplikasikan pada perusahaan persewaan mobil untuk mengurangi tingkat pencurian yang terjadi.

Sistem ini bekerja dengan cara menerima data lokasi yang dikirimkan oleh satelit ke GPS *receiver*, kemudian data tersebut diproses dengan menggunakan mikrokontroler. Jika *central unit* meminta data lokasi dari mobil yang menggunakan sistem ini, maka mikrokontroler akan mengirimkan data lokasi terakhir berdasarkan pantauan satelit melalui *handphone* yang berbentuk SMS. *Central unit* juga dapat mengirimkan perintah untuk menonaktifkan mobil (mematikan mesin) jika terjadi sesuatu hal yang tidak diinginkan.

Sistem pengaman ini dapat mengetahui lokasi mobil walau berada pada posisi yang jauh dari *central unit* dengan cara data lokasi berupa koordinat yang diterima GPS *receiver* dimasukkan dalam program Google Earth yang dapat menampilkan gambar Bumi secara keseluruhan yang diambil dari satelit. Alat ini

juga dapat mengirimkan informasi penting kepada konsumen tentang status persewaan mobil dengan cepat mengunakan layanan jaringan GSM yang telah melingkupi hampir diseluruh daerah di indonesia.

1.2. Rumusan Masalah

Mengacu pada permasalahan yang ada, maka rumusan masalah ini adalah :

Bagaimana membuat alat pengaman mobil yang dapat melacak posisi dan dapat menonaktifkan mesin mobil dari jarak jauh.

1.3. Batasan Masalah

Agar permasalahan yang ada dapat di jelaskan secara tepat dan terhindar dari pembahasan yang tidak sesuai dengan topik yang di bahas maka dianggap perlu adanya batasan masalah. Adapun batasan masalah pada tugas akhir ini antara lain:

1. GPS dan Google Earth digunakan sebagai pelacak posisi.
2. *Handphone* hanya digunakan fasilitas SMS nya saja.
3. Jaringan GSM dianggap konstan dalam keadaan stabil (tidak ada gangguan).

1.4. Tujuan

Tujuan pembuatan alat ini adalah :

Membuat alat pengaman mobil yang dapat melacak posisi dan dapat menonaktifkan mesin mobil dari jarak jauh.

1.5. Metodologi

Metodologi yang dipakai dalam pembuatan skripsi ini adalah:

1. Studi Literatur

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

2. *Field Research*

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

3. Perencanaan Dan Pembuatan Alat

   Dalam pembuatan alat ini menggunakan konsep sebagai berikut :

   - Perencanaan sistem secara keseluruhan (pembuatan blok diagram sistem).
   - Mendeskripsikan fungsi dari masing-masing blok diagram.
   - Membuat perangkat keras *(hardware)* dan perangkat lunaknya *(software).*

4. Pengujian Alat

   Dengan melakukan pengujian perblok rangkaian dan kerja seluruh sistem pada alat tersebut.

5. Penyusunan Laporan Skripsi

   Membuat laporan yang terdiri dari: Pendahuluan, Landasan Teori, Perencanaan dan Pembuatan Alat, Pengujian Alat dan Penutup.

1.6. Sistematika Penulisan

Penulisan skripsi ini terbagi menjadi lima bab dengan sistematika sebagai berikut:

BAB I PENDAHULUAN

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

BAB II LANDASAN TEORI

Membahas teori dasar penunjang perancangan dan pembuatan alat.

BAB III PERENCANAAN DAN PEMBUATAN ALAT

Membahas tentang perancangan alat yang terdiri dari perancangan perangkat keras dan perangkat lunak.

BAB IV PENGUJIAN ALAT

Membahas tentang pengujian peralatan secara keseluruhan dan analisa hasil pengujian.

BAB V PENUTUP

Berisikan kesimpulan dan saran.

BAB II

LANDASAN TEORI

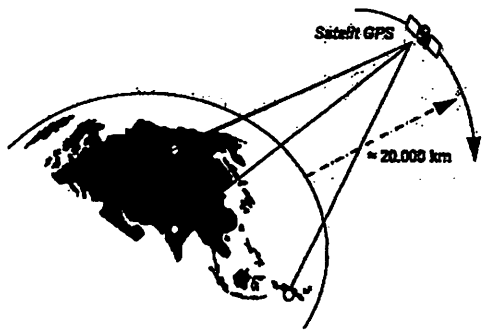
Dalam bab ini akan dibahas mengenai dasar-dasar teori yang mendukung dalam pembuatan alat pelacak posisi dengan GPS dan penonaktifan mesin mobil dengan kendali handphone.

2.1. *Global Position Systems*

Global Positioning System (GPS) merupakan sebuah sistem informasi yang dapat digunakan untuk menunjukkan posisi secara nyata suatu obyek pada permukaan bumi. Dimana informasi GPS didapatkan dari 24 satelit GPS Yang mempunyai ketinggian orbit sekitar 20.000 km di atas permukaan bumi yang sampai saat ini dikontrol oleh stasiun bumi. Dimana dari *receiver* GPS akan dapat dideteksi sinyal yang dipancarkan dari satelit dengan tingkat ketepatan yang tinggi. Untuk lebih memperjelas mengenai sistem GPS ini, maka akan diperjelas mengenai Navigasi, elemen GPS dan cara kerja GPS.

2.1.1. Navigasi GPS

Pada sistem GPS penunjukan navigasi ditentukan berdasarkan posisi garis lintang dan bujur objek pada permukaan bumi yang terpetakan berdasarkan sudut satelit yang menerima sinyal dari *receiver* GPS.



Gambar 2.1. Cakupan GPS di bumi[1]

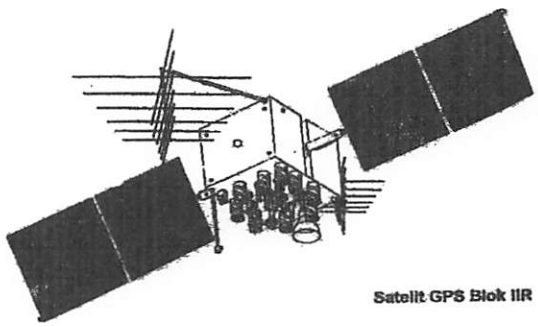
2.1.2. Elemen GPS

GPS memiliki 3 elemen, yaitu : bagian satelit, bagian pengguna (*receiver*) dan bagian kontrol (stasiun bumi). Ketiga elemen ini saling berinteraksi untuk memberikan informasi, dimana bagian satelit berinteraksi dengan bagian pengguna untuk mengirimkan sinyal yang dideteksi, sedangkan bagian kontrol berfungsi untuk memonitor pergerakan satelit pada orbitnya.

2.1.2.1. Segmen Satelit

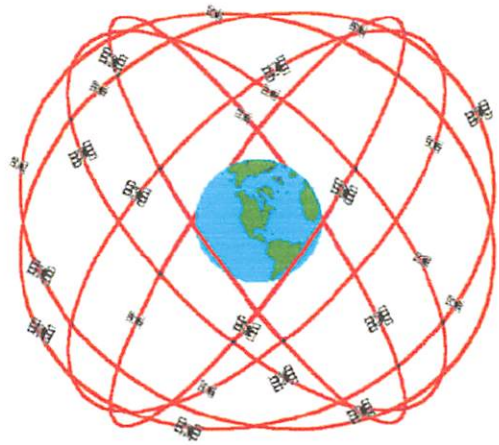
Pada bagian elemen satelit, terdiri dari 24 satelit yang masing-masing berada pada orbit dengan ketinggian sekitar 20.000 km diatas permukaan bumi seperti terlihat pada gambar 2.1. Dimana masing-masing satelit memiliki 12 jam waktu orbit untuk mengelilingi bumi. Setiap satelit diposisiskan dengan clock akurasi pada sinyal broadcast

dengan waktu pesan yang tepat. Oleh karenanya dengan diposisikan satelit dengan posisi yang mampu menjangkau setiap daerah dibumi dari antartika sampai artik maka sinyal dapat diterima mendekati tingkat keberhasilan 100% dari setiap sinyal disetiap tempat dan disetiap waktu dibumi. Dimana akurasi ketepatan dijaga dengan skala 3 nano second. Ketepatan waktu ini penting karena penerima *receiver* harus tahu secara nyata seberapa lama dia mengirimkan sinyal dan kemudian dapat diterima oleh satelit, kemudian dari waktu tersebut pada posisi setiap satelit dapat dihitung posisi keberadaan *receiver*.



Gambar 2.2. Bentuk tipikal Satelit GPS Blok IIR[1]

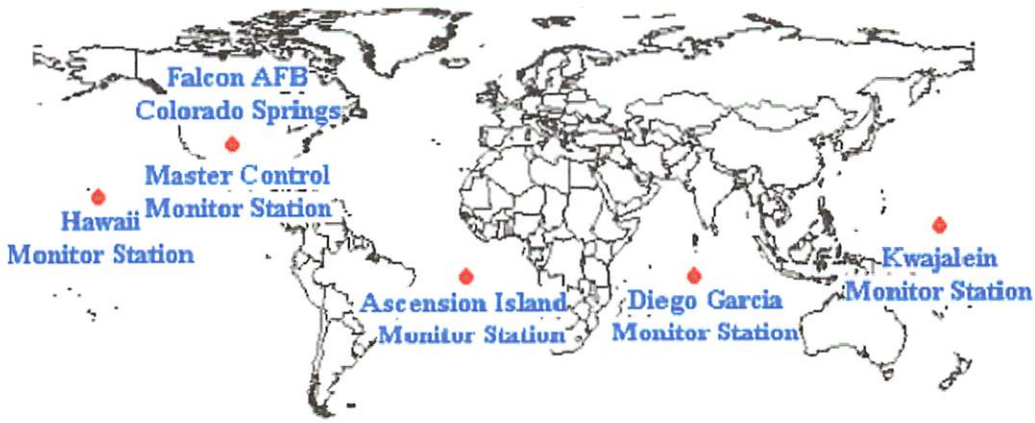
Konstelasi standar dari satelit GPS terdiri dari 24 satelit yang menempati 6 (enam) bidang orbit yang bentuknya sangat mendekati lingkaran, dengan eksentrisitas orbit umumnya lebih kecil dari 0,02, seperti yang diilustrasikan pada gambar 2.3. Keenam bidang orbit satelit GPS mempunyai spasi sudut yang sama antar sesamanya.



Gambar 2.3. Konfigurasi Orbit[1]

2.1.2.2. Segmen Stasiun Kontrol Bumi

Stasiun kontrol bumi, merupakan stasiun yang bertugas untuk memonitor dan mengontrol pergerakan satelit pada orbitnya serta memastikan bahwa satelit berfungsi sebagaimana mestinya. Kelayakgunaan satelit-satelit GPS tersebut dimonitor dan dikontrol oleh segmen sistem kontrol yang terdiri dari beberapa stasiun pemonitor dan pengontrol yang tersebar diseluruh dunia. Segmen kontrol ini juga berfungsi menentukan orbit dari seluruh satelit GPS yang merupakan informasi vital untuk penentuan posisi dengan satelit. Secara spesifik, segmen sistem kontrol terdiri dari *Ground Antenna Stations* (GAS), *Monitor Station* (MS), *Prelaunch Compability Station* (PCS), dan *Master Control Station* (MCS).



Gambar 2.4. Lokasi stasiun sistem kontrol GPS[1]

2.1.2.3. Segmen Pengguna

Segmen pengguna terdiri dari para pengguna satelit GPS, baik di darat, laut udara. Dalam hal ini alat penerima sinyal GPS (*GPS receiver*) diperlukan untuk menerima dan memproses sinyal dari satelit GPS untuk digunakan dalam penentuan posisi, kecepatan maupun waktu.

GPS *Receiver* yang digunakan adalah EG-T10 buatan Elink, GPS ini hanya berupa modul *receiver* data dari satelit tanpa display dan memori layaknya GPS – GPS *portable* yang banyak berada di pasaran bebas.

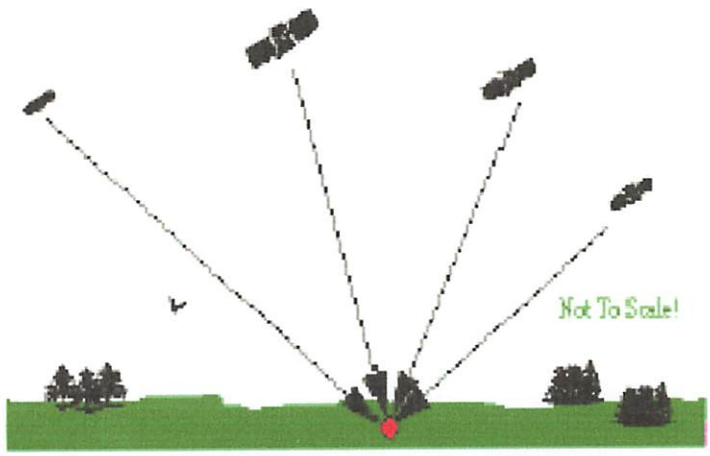
Karena berupa modul dasar, GPS ini masih memerlukan antena eksternal dan *power suply* tambahan untuk mengoperasikannya. Antena berfungsi untuk menangkap *signal* yang dikirim oleh satelit , namun berhubung lemahnya *signal* yang dikirim, *signal* mudah rusak dan tidak dapat

menembus halangan yang berupa bagunanan atau kondisi cuaca buruk.

Untuk menerima dan mengolah data dari GPS digunakan *microcontroller* yang berkomunikasi data secara serial dengan level TTL.

2.1.3. Cara Kerja GPS

prinsip dasar GPS dalam melakukan pengukuran suatu lokasi *receiver* adalah dengan melakukan pengukuran sudut antara *receiver* dengan satelit yang menjangkaunya, kemudian satelit akan memberikan informasi dimana posisi orbitnya.



Gambar 2.5. Prinsip kerja sistem GPS[6]

Pesawat penerima GPS menggunakan sinyal satelit untuk melakukan triangulasi posisi yang hendak ditentukan dengan cara mengukur lama perjalanan waktu sinyal dikirimkan dari satelit, kemudian mengalikannya dengan kecepatan rambat gelombang untuk

menentukan secara tepat berapa jauh pesawat penerima GPS dari setiap satelit. Dengan mengunci sinyal yang ditransmit oleh satelit minumum 3 sinyal dari satelit yang berbeda, pesawat penerima GPS dapat menghitung posisi tetap sebuah titik yaitu posisi Lintang dan Bujur bumi (Latitude & Longitude) atau sering disebut dengan 2D fix. Penguncian sinyal satelit yang keempat membuat pesawat penerima GPS dapat menghitung posisi ketinggian titik tersebut terhadap muka laut rata-rata (Mean Sea /Level) atau disebut 3D fix dan keadaan ini yang ideal untuk melakukan navigasi.

2.1.4. NMEA 0813 (*National Marine Electronics Association*)

Dengan beraneka ragamnya peralatan-peralatan yang dapat dihubungkan dengan sebuah GPS *receiver*, maka haruslah dibentuk suatu protocol komunikasi yang standar, sehingga suatu GPS *receiver* dapat berhubungan dengan peralatan-peralatan yang lain. Dan suatu peralatan dapat berhubungan dengan berbagai macam GPS *receiver* yang ada di pasaran. Standar komunikasi untuk *interface* dengan GPS *receiver* adalah NMEA 0813.

Format NMEA 0813 kompatibel dengan RS-232 dan komponen IC TTL, juga kompatibel dengan standar RS-422 yang merupakan standar komunikasi dengan sistem *differential*. Sistem ini bekerja pada 4800 bps dengan sistem pengiriman secara data paket. Selain itu semua data yang dikirimkan sudah bukan berupa data *byte* langsung melainkan

berupa kode-kode ASCII teks yang dapat dicetak. Format dari protocol komunikasi ini dapat dilihat pada table 2.1.

Tabel 2.1. Format Protokol NMEA 0813[11]

$	Talker ID	Sentences ID	Data Fields	*	Check Sum	<CR>	<LF>

Keterangan :

"$" Merupakan karakter awal yang dikirimkan yang menandakan awal dari pengiriman suatu paket data.

Talker ID Terdiri dari dua buah huruf yang menunjukan jenis GPS *receiver* yang sedang dipakai.

Contoh:

GP *Global Positioning System receiver*

LC *Loran-C receiver*

OM *Omega Navigation Receiver*

Sentence ID Terdiri dari tiga huruf yang menunjukan jenis paket yang sedang dikiriman.

Contoh:

GLL *Geographic Position*

GSA GPS DOP and *active satellite*

GSV *Satellite in View*

RMB *Recommended minimum navigation information*

Data Fields Merupakan data dari jenis paket yang telah disebutkan pada sentences ID. Apabila pada *segment*

ini terdiri dari beberapa jenis data lagi maka perlu dipisah-pisahkan dengan koma. Apabila data yang bersangkutan tidak dapat diberikan oleh GPS *receiver* karena misalnya GPS *receiver* tidak menerima sinyal informasi apapun, maka yang dikirimkan hanya komanya saja tanpa diisi apapun.

Check Sum — Bagian ini hanyalah optional (boleh ada/boleh tidak), *Check Sum* ini selalu didahului dengan karakter ' * ' dan terdiri dari dua digit *hexadecimal*. *Check Sum* ini digunakan sebagai pendeteksi *error* apakah paket data yang diterima itu sudah benar dan sesuai dengan saat dikirimkan. *Check Sum* ini merupakan XOR dari semua karakter, tetapi tidak termasuk ' $ ' dan ' * '.

<CR> <LF> — ASCII *Carried Return dan Line Feed* ini dikirimkan sebagai tanda bahwa transmisi untuk paket yang bersangkutan sudah berakhir.

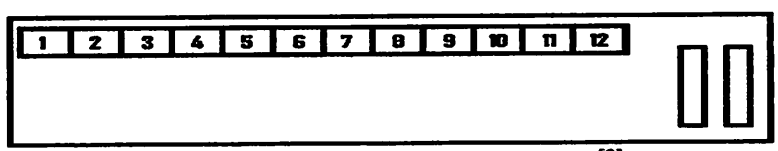
2.2. Telepon Seluler

Telepon Seluler (Ponsel) adalah suatu jenis telepon bergerak yang menggunakan teknologi sel sebagai akses komunikasinya sehingga memudahkan seseorang untuk berkomunikasi dimanapun dan dalam kondisi apapun. Sebuah ponsel dapat mengirim dan menerima data suara menggunakan pemancar RF.

Dengan adanya ponsel, maka komunikasi lebih mudah dan lebih efisien tetapi sedikit lebih mahal. Seiring perkembangan teknologi di bidang *mobile* sistem, ponsel tidak hanya mengirim data suara tetapi juga data karakter yang biasa disebut SMS *(Short Message Service).* Disamping itu, pesatnya perkembangan teknologi saat ini menyebabkan ponsel tidak hanya dapat berkomunikasi antara sesama ponsel, tetapi juga dapat berkomunikasi dengan mikrokontroler. Hal ini menyebabkan kita dapat melakukan pentransferan data antara ponsel dan mikrokontroler.

2.2.1. Siemens M-35

Siemens merupakan salah satu merek ponsel yang dapat berkomunikasi dengan mikrokontroler melalui port serial, sehingga suatu pentransferan data dapat terjadi antara mikrokontroler dengan ponsel merk Siemens tersebut. Melalui pentransferan data ini penggguna ponsel dapat mengirim atau menerima suatu pesan singkat (SMS). Untuk dapat berkomunikasi dengan mikrokontroler, Siemens dilengkapi dengan *internal* modem yang dapat mengenali *AT Command.*



Gambar 2.6. Konektor Siemens M-35[9]

Tabel 2.2. Keterangan Konektor Siemens M-35[9]

Nama	Fungsi	In/Out
GND	*Ground*	
SELF SERVICE	*Recognition / control battery charger*	In/Out
LOAD	*Charging Voltage*	In
BATTERY	*Battery*	Out
DATA OUT	*Data Sent*	Out
DATA IN	*Data eceived*	In
Z CLK	*Recognition / control accessories*	
Z DATA	*Recognition / control accessories*	
MIC3	*Ground for microphone*	In
MIC	*Microphone input*	
AUD	*Loudspeaker*	Out
AUDG	*Ground for eksternal speaker*	

2.2.2. Format Data SMS

SMS *(Short Message Service)* merupakan sebuah mekanisme pengiriman pesan singkat melalui jaringan bergerak *(mobile network).* Panjang maksimum dari sebuah pesan singkat adalah 160 karakter, fasilitas ini disediakan oleh jaringan telepon seluler. Sebenarnya panjang pesan maksimum yang dapat dikirimkan melalui SMS adalah 140 karakter. Teknik ini bertumpu pada keadaan bahwa kode karakter ASCII alfanumerik yang mempunyai lebar data 7 bit (bit ke-7 selalu bernilai 0 sehingga bisa

diabaikan). Teknik kompresi *septet to oktet* dilakukan dengan menyisipkan bit-bit LSB karakter selanjutnya ke dalam bit-bit MSB dari data sebelumnya secara berkesinambungan.

Ada dua Cara pengiriman dan penerimaan SMS, yaitu dengan menggunakan mode teks dan mode PDU *(Protocol Data Unit).* Mode teks tidak terdapat pada beberapa telepon seluler, maka dalam laporan skripsi ini digunakan pesan SMS dengan mode PDU dan tidak membahas mode yang lain. PDU berisi bilangan-bilangan heksadesimal yang terdiri atas: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.

2.2.3. Prinsip Kerja SMS *(Short Message Service)*

Dibalik tampilan menu-menu SMS dalam telepon seluler sebenarnya adalah *AT Command* yang bertugas mengirim, menerima dan menghapus data dari dan ke SMS *centre. AT Command* tiap-tiap perangkat SMS bisa berbeda-beda tergantung dari jenis telepon seluler, akan tetapi pada dasarnya sama berdasarkan fungsinya. Data-data yang mengalir ke dan dari *SMS-centre* berbentuk PDU *(Protocol Data Unit).* PDU ini berisi bilangan-bilangan heksadesimal yang mencerminkan bahasa input output. PDU tersebut terdiri dari beberapa *header. Header* untuk mengirim SMS ke *SMS-centre* berbeda dengan SMS yang diterima dari *SMS-centre.*

2.2.4. PDU untuk Kirim SMS ke SMS-Centre

Fomat PDU untuk mengirim SMS terdiri atas delapan *header,* yaitu sebagai berikut :

1. Nomor *SMS-Centre*

   Header pertama ini terbagi atas tiga *subheader,* yaitu :

   a. Jumlah pasangan heksadesimal *SMS-Centre* dalam bilangan heksa.

   b. Kode *National/International.*

   - Untuk *National,* kode *subheader-nya* yaitu 81
   - Untuk *International,* kode *subheader-nya* yaitu 91

   c. Nomor *SMS-Centre-nya* sendiri, dalam pasangan heksa dibalik-balik. Jika tertinggal satu angka heksa yang tidak memiliki pasangan, angka tersebut akan dipasangkan dengan huruf F di depannya. Contoh: untuk nomor *SMS-Centre* Indosat-M3 dapat ditulis dengan 2 cara yaitu :

   - **Cara 1, 0855000000 diubah menjadi:**

     1) 06 → ada 6 pasang

     2) 81 → 1 pasang

     3) 80-55-00-00-00 → 5 pasang

   Total ada 6 pasang. Digabung menjadi : **<u>06818055000000</u>**

   - **Cara 2, 62855000000 diubah menjadi:**

     1) 07 → ada 7 pasang

     2) 91 → 1 pasang

3) 26-58-05-00-00-F0 → 6 pasang

Total ada 7 pasang. Digabung menjadi:**07912658050000F0**

Berikut ini beberapa nomor SMS-*Centre* operator seluler di Indonesia :

Tabel 2.3. Beberapa nomor *SMS-Centre National Code*[4]

No	Operator Seluler	SMS-Centre No	Kode PDU
1.	Telkomsel	0811000000	06818011000000
2.	Satelindo	0816124	0581806121F4
3.	Excelcom	0818445009	06818081440590
4.	Indosat-M3	0855000000	0855000000

Tabel 2.4. Beberapa nomor *SMS-Centre International Code*[4]

No	Operator Seluler	SMS-Centre No	Kode PDU
1.	Telkomsel	62811000000	07912618010000F0
2.	Satelindo	62816124	059126181642
3.	Excelcom	62818445009	07912618485400F9
4.	Indosat-M3	62855000000	07912658050000F0

2. Tipe SMS

   Untuk tipe SMS kirim = 1, jadi bilangan heksanya adalah 01 atau 11

3. Nomor Referensi

Nomor referensi ini dibiarkan dulu 0, jadi bilangan heksanya adalah 00. Nomor referensi ini akan diberikan secara otomatis oleh ponsel tersebut.

4. Nomor Ponsel Penerima

   Sama seperti cara menulis PDU *Header* untuk *SMS-Center, header* ini juga terbagi atas tiga bagian, sebagai berikut :

   a. Jumlah bilangan desimal nomor ponsel yang dituju dalam bilangan heksa

   b. National/International Code

   - Untuk *National,* kode *subheader-nya* yaitu 81
   - Untuk *International,* kode *subheader-nya* yaitu 91

   c. Nomor ponsel yang dituju, dalam pasangan heksa dibalik. Jika tertinggal satu angka heksa atau tidak memiliki pasangan, angka tersebut dipasangkan dengan huruf 'F' di depannya.

   Contoh :

   Untuk nomor ponsel yang dituju 628129573337 dapat ditulis dengan dua cara, yaitu sebagai berikut :

   - Cara 1, 08129573337 diubah menjadi :

     a. 0B → ada 11 angka

     b. 81

     c. 80-21-59-37-33-F7

     Sehingga digabung menjadi : **0B818021593733F7**

- Cara 2, 628129573337 diubah menjadi :

  a. 0C → ada 12 angka

  b. 91

  c. 26-18-92-75-33-73

  Sehingga digabungkan menjadi : **0C91261892753373**

5. Bentuk SMS, antara lain :

   0 → 00→ dikirim sebagai SMS.

   1 → 01→ dikirim sebagai *telex.*

   2 → 02→ dikirim sebagai *fax.*

   Dalam hal ini, untuk mengirim dalam bentuk SMS tentu saja harus memakai kode heksa 00.

6. Skema *Encoding Data I/O*

   Ada dua skema, yaitu :

   a. Skema 7 bit, ditandai dengan angka → 00

   b. Skema 8 bit, ditandai dengan angka lebih besar dari 0, kemudian diubah ke heksa.

   Kebanyakan ponsel/SMS *Gateway yang* ada di pasaran sekarang menggunakan skema 7 bit sehingga kita menggunakan kode 00.

7. Jangka waktu/waktu validitas

   Jika jangka waktu validitas diabaikan (kode 00), itu berarti tidak membatasi waktu berlakunya SMS. Sedangkan jika diisi dengan suatu bilangan integer yang kemudian diubah ke

pasangan heksa tertentu, bilangan yang diberikan tersebut akan mewakili jumlah waktu validitas SMS tersebut. Rumus untuk menghitung jangka waktu validitas SMS adalah sebagai berikut:

Tabel 2.5. Rumus untuk Menghitung Jangka Waktu Validitas SMS[4]

Integer (INT)	Jangka Waktu Validitas SMS
0 – 143	(INT + 1) x 5 menit (berarti : 5 menit s/d 12 jam)
144 – 167	12 jam + ((INT – 143) x 30 menit)
168 – 196	(INT – 166) x 1 hari
197 – 255	(INT – 192) x 1 minggu

8. Isi SMS

Header ini terdiri atas dua *subheader,* yaitu :

a. Panjang isi (jumlah huruf dari isi)

Misalnya : untuk kata "bahaya", ada 6 huruf maka penulisannya 06

b. Isi berupa pasangan bilangan heksa

Untuk ponsel/SMS *gateway* berskema *encoding* 7 bit, jika mengetik suatu huruf dari keypadnya, berarti telah membuat 7 angka I/O berturutan. Skema 7 bit tersebut diperlihatkan pada Tabel 2.7.

Ada 2 langkah yang harus dilakukan untuk

mengkonversikan isi SMS, yaitu :

Langkah pertama : mengubah menjadi kode *7* bit.

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

Contoh : untuk kata "bahaya"

Langkah pertama :

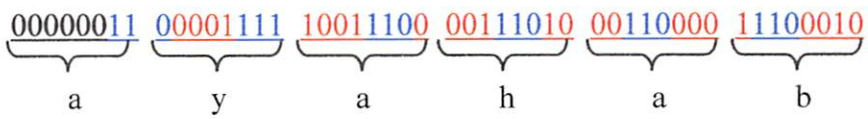
Isi	Bit ke						
	7	**6**	**5**	**4**	**3**	**2**	**1**
B	1	1	0	0	0	1	0
A	1	1	0	0	0	0	1
H	1	1	0	1	0	0	0
A	1	1	0	0	0	0	1
Y	1	1	1	1	0	0	1
A	1	1	0	0	0	0	1

Langkah Kedua:

Mengubah kode 7 bit menjadi 8 bit, yaitu oleh karena total 7 bit x 6 huruf = 42 bit, sedangkan yang diperlukan adalah 8 bit x 6 huruf = 48 bit, maka diperlukan 6 bit *dummy* yang diisi dengan bilangan 0, ditambahkan pada MSB huruf terakhir. Setiap 8 bit mewakili suatu pasangan heksa. Tiap 4 bit mewakili suatu angka heksa, tentu saja karena secara logika $2^4 = 16$

Susunan menjadi :

Susunan 4 bit dimulai dari MSB menjadi :



Isi	Bit ke							
	8	**7**	**6**	**5**	**4**	**3**	**2**	**1**
B	1	1	1	0	0	0	1	0
	E				**2**			
A	0	0	1	1	0	0	0	0
	3				**0**			
H	0	0	1	1	1	0	1	0
	3				**A**			
A	1	0	0	1	1	1	0	0
	9				**C**			
y	0	0	0	0	1	1	1	1
	0				**F**			
a	0	0	0	0	0	0	1	1
	0				**3**			

Dengan demikian kata "bahaya" hasil konversinya menjadi:

E2303A9C0F03

Tabel 2.6. Skema 7 Bit SMS pada Telepon Seluler[4]

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	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		Λ		4	D	T	d	t
0	1	0	1	5		Ω	%	5	E	U	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	Ξ	*	:	J	Z	j	z
1	0	1	1	11			+	;	K	Ä	k	ä
1	1	0	0	12			,	<	L	Ö	l	ö
1	1	0	1	13	CR		-	=	M		m	
1	1	1	0	14		ß	.	>	N	Ü	n	ü
1	1	1	1	15			/	?	O		o	

Cara penggunaan tabel :

Misalnya untuk karakter `N', yaitu :

1. Cari posisi karakter `N' pada tabel.

2. Baca nilai biner B1 – B4 pada sisi kiri tabel dan B5 – B7 pada sisi atas tabel.
3. Pembacaan nilai biner dimulai dari kiri ke kanan yaitu B7 – B1, untuk karakter `N' bernilai 1001110.

Setelah mempelajari masing-masing header maupun sub-header untuk mengirim SMS maka untuk menggabungkan kedelapan header tersebut menjadi PDU yang lengkap adalah sebagai berikut :

Contoh: Untuk mengirimkan kata "bahaya" ke handphone nomor 6285664408185 lewat SMS-center Indosat-M3, tanpa membatasi jangka waktu valid, maka PDU lengkapnya adalah:

07	91	2658050000F0	01	00	0D	91	265866448081F5	00	00	06	E2303A9C0F03

Penjelasan dari format tersebut dapat dijelaskan pada tabel berikut :

Tabel 2.7. Format SMS Kirim dalam PDU[4]

Heksa	Penjelasan
07	Panjang pasangan nomor SMS-Center termasuk tipe Kode
91	Tipe kode Nasional atau Internasional (91 = kode internasional)
2658050000F0	Nomor Service-Center (62855000000)
01	Tipe SMS (01 = tipe untuk pengiriman)
00	Nomor referensi
0D	Panjang nomor handphone penerima
91	Tipe kode Nasional atau internasional
265866448081F5	Nomor Handphone penerima (6285664408185)

00	Tipe bentuk SMS (00 = dikirim sebagai SMS)
00	Tipe data coding
06	Panjang pasangan dari isi SMS
E2303A9C0F03	Isi SMS (bahaya)

2.2.4.1. PDU untuk SMS Terima dari SMS-Centre

Terdapat *8 header* untuk menerima SMS *yang* pada prinsipnya hampir sama dengan *8 header* untuk mengirim *SMS. Header* tersebut meliputi:

1. Nomor *SMS-Centre*
2. Tipe SMS untuk SMS-terima = 4 menjadi **04**
3. Nomor ponsel pengirim
4. Bentuk SMS
5. Skema *Encoding*
6. Tanggal dan waktu SMS di *SMS-Centre*

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

   Misalnya : 207022512380 yang berarti 02/07/22 15:32:08 atau 22 Juli 2002 15:32:08 WIB.
7. Batas waktu validitas, jika tidak dibatasi dilambangkan dengan 00.
8. Isi SMS.

   Isi SMS yang berupa tulisan disini nantinya dilakukan pengkonversian yaitu mengubahnya menjadi 8 bit dan kemudian menjadi 7 bit.

Contoh: Format SMS Terima dalam PDU :

07	91	2658050000F0	04	0D	91	265866448081F4	00	00	701022512380	00	06	E2303A9C0F03

Penjelasan dari format tersebut dapat dijelaskan pada tabel berikut :

Tabel 2.8. Format SMS Terima dalam PDU[4]

Heksa	Penjelasan
07	Panjang pasangan nomor SMS-Center termasuk tipe Kode
91	Tipe kode Nasional atau Internasional (91 = kode internasional)
2658050000F0	Nomor Service-Center (62855000000)
04	Tipe SMS (04= tipe untuk terima)
0D	Panjang nomor handphone pengirim
91	Tipe kode Nasional atau internasional
265866448081F4	Nomor Handphone pengirim (6285664408184)
00	Tipe bentuk SMS (00 = diterima sebagai SMS)
00	Tipe data coding
701022512380	Waktu SMS sampai di SMS center yaitu tanggal 22-01-07, pukul 15:32:08 WIB.
00	Jangka waktu SMS expired (00 = tidak memiliki batas)
06	Panjang pasangan dari isi SMS
E2303A9C0F03	Isi SMS (bahaya)

Setelah mengupas satu demi satu header untuk SMS terima ini, maka untuk PDU di bawah ini :

07912658050000F0, 04, 0D91265866448081F4, 00, 00, 701022512380, 00, 06, E2303A9C0F03

Dapat kita artikan sebagai berikut :

1. SMS tersebut dikirim lewat SMS-center 62855000000

2. SMS tersebut merupakan SMS terima
3. SMS tersebut dikirim dari *handphone* no. 6285664408184
4. SMS tersebut diterima dalam bentuk SMS
5. SMS tersebut memiliki skema encoding 7 bit
6. SMS tersebut sampai di SMS-center pada tanggal: 22-01-07, pukul: 15:32:08 WIB
7. SMS tersebut tidak memiliki batas waktu valid
8. SMS tersebut isinya adalah "bahaya"

2.2.5. AT - Command

Komunikasi data antara telepon seluler dengan periperal lain seperti mikrokontroler dilakukan secara serial menggunakan perintah-perintah AT *(AT Command)*. Dengan mengirimkan perintah-perintah AT yang spesifik dapat memerintahkan telepon seluler untuk melakukan apa yang kita inginkan.

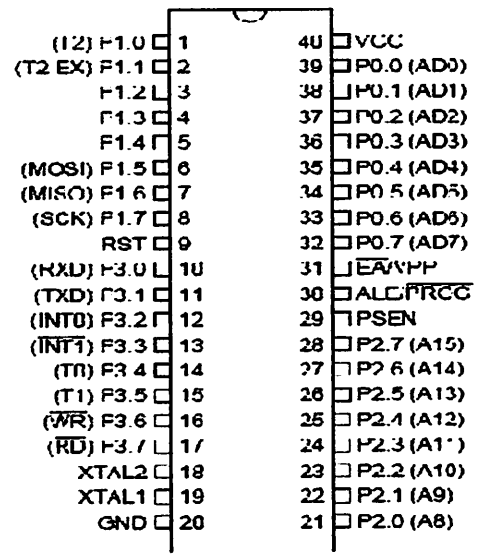
Tabel 2.9. *AT Command* pada SMS[9]

Perintah	Fungsi
AT+CMGC	Mengirim sebuah perintah SMS
AT+CMGD	Menghapus sebuah SMS dalam SMS memori
AT+CMGF	SMS Format
AT+CMGL	Daftar SMS
AT+CMGR	Membaca dalam sebuah SMS
AT+CMGS	Mengirim sebuah SMS

AT+CSCA	Alamat dari pusat SMS servis

2.3. Mikrokontroler AT89S52

AT89S52 adalah *mikrokontroller* keluaran Atmel dengan 8K *byte* Flash PEROM *(Programmable and Erasable Read Only Memory)*, AT89S52 merupakan memori dengan teknologi *nonvolatile memory*, isi memori tersebut dapat diisi ulang ataupun dihapus berkali-kali. Memori ini biasa digunakan untuk menyimpan instruksi (perintah) berstandar MCS-51 *code* sehingga memungkinkan *mikrokontroller* ini untuk bekerja dalam *mode single chip operation* (mode operasi keping tunggal) yang tidak memerlukan *external memory* (memori luar) untuk menyimpan *source code* tersebut.



Gambar 2.7. Konfigurasi Pin Mikrokontroler AT89S52[2]

2.3.1. Deskripsi Mikrokontroller AT89S52

Berikut deskripsi dari mikrokontroler keluarga ATMEL AT89S52 :

- VCC (*power supply*)

- GND (*ground*)
- Port 0, yaitu pin p0.0...p0.7

  Port 0 dapat berfungsi sebagai I/O biasa, *low order multiplex addres/data* ataupun menerima kode bye pada saat *Flash Programming*. Pada saat sebagai I/O biasa port ini dapat memberikan *output sink* ke delapan buah *Transistor Transistor Logic* (TTL) input atau dapat diubah sebagai input dengan memberikan logika 1 pada port tersebut.
- Port 1, yaitu pin p1.0...p1.7 Port 1 berfungsi sebagai I/O biasa atau menerima *low order address bytes* selama pada saat *Flash Programming*. Port ini mempunyai *internal pull up* dan berfungsi sebagai *input* dengan memberikan logika 1. Sebagai *output* port ini dapat memberikan *output sink* keempat buah input TTL. Fasilitas khusus dari port 1 ini adalah adanya *In-System Programming*, yaitu port 1.5 sebagai MOSI, port 1.6 sebagai MISO, port 1.7 sebagai SCK.
- Port 2, yaitu mulai pin p2.0...p2.7

  Port 2 berfungsi sebagai I\O biasa atau *high order address*, pada saat mengakses memori secara 16 bit (Movx @DPTR). Pada saat mengakses memori secara 8 bit (Mov @Rn), port ini akan mengeluarkan sisi dari *Special Function Register*. Port ini mempunyai *pull up* dan berfungsi sebagai *input* dengan

memberikan logika 1. Sebagai *output*, port ini dapat memberikan *output sink* keempat buah input TTL.

- Pin 3.0, sebagai RXD *(Port Serial Input)*.
- Pin 3.1, sebagai TXD (Port Seial Output).
- Pin 3.2, sebagai INT0 (Port External Interupt 0).
- Pin 3.3, sebagai INT1 (Port External Interupt 1).
- Pin 3.4, sebagai T0 (Port External Timer 0).
- Pin 3.5, sebagai T1 (Port External Timer 1).
- Pin 3.6, sebagai WR (External Data Memory Write Strobe).
- Pin 3.7, sebagai RD (External Data Memory Read Strobe).
- Pin 9, sebagai RST

  Reset akan aktif dengan memberikan *input high* selama 2 *cycle*.
- Pin 30, sebagai ALE/PROG

  Pin ini dapat berfungsi sebagai *Address Latch Enable* (ALE) yang me-latch low *byte* address pada saat mengakses memori *external*. Sedangkan pada saat Flash Programming (PROG) berfungsi sebagai pulsa *input*. Pada operasi normal ALE akan mengeluarkan sinyal *clock* sebesar 1/16 *frekwensi oscillator*, kecuali pada saat mengakses memori *external*. Sinyal *clock* pada saat ini dapat pula di-*disable* dengan men-set bit 0 *Special Function Register*.
- Pin 29, sebagai PSEN

  Pin ini berfungsi pada saat mengeksekusi program yang terletak pada memori *external*. PSEN akan aktif dua kali setiap cycle.

- Pin 31, Sebagai EA/VPP

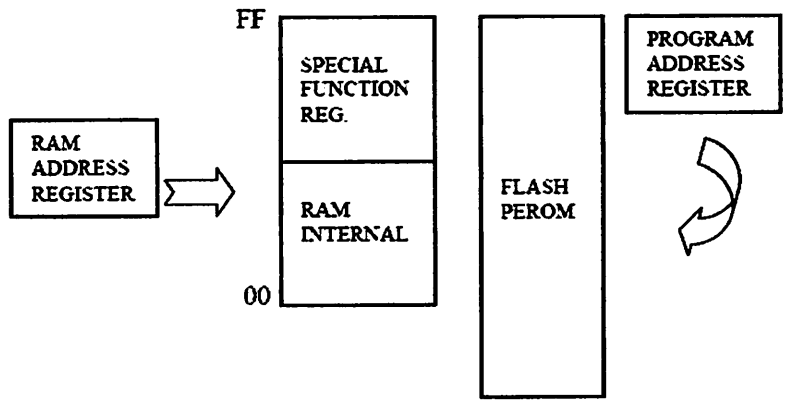
  Pada kondisi *low*, pin ini akan berfungsi sebagai EA yaitu *mikrokontroller* akan menjalankan program yang ada pada memori *external* setelah sistem di reset. Jika berkondisi *high*, pin ini akan berfungsi untuk menjalankan program yang ada pada memori *internal*. Pada saat Flash Programming pin ini akan mendapat tegangan 12 Volt (VPP).

- Pin 19, sebagai XTALL1 *(Input Oscillator).*
- Pin 18, sebagai XTALL2 *(Output Oscillator).*

2.3.2. Struktur Memori

AT89S52 mempunyai stuktur memori yang terdiri atas :

- *RAM Internal*, memori sebesar 128 *byte* yang biasanya digunakan untuk menyimpan variabel atau data yang bersifat sementara.
- *Special Function Register* (Register Fungsi Khusus), memori yang berisi register-register yang mempunyai fungsi-fungsi khusus yang disediakan oleh *mikrokontroller* tersebut, seperti timer, serial dan lain-lain.
- *Flash PEROM*, memori yang digunakan untuk menyimpan instruksi-instruksi MCS51.



Gambar 2.8. Struktur Memori AT89S52[2]

AT89S52 mempunyai struktur memori yang terpisah antara RAM *Internal* dan Flash PEROM nya. RAM *Internal* dialamati oleh *RAM Address Register* (*Register* Alamat RAM) sedangkan *Flash PEROM* yang menyimpan perintah-perintah MCS-51 dialamati oleh *Program Adderss Register* (Register Alamat Program). Dengan adanya struktur memori yang terpisah tersebut, walaupun RAM *Internal* dan Flash PEROM mempunyai alamat yang sama, yaitu alamat 00, namun secara fisiknya kedua memori tidak saling berhubungan.

2.3.3. Antarmuka Serial

Port serial pada AT89S52 bersifat dupleks-penuh atau *fullduplex*, artinya port serial bisa menerima dan mengirim secara bersamaan. 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 juga dibaca, maka salah satu *byte* akan hilang). Penerimaan dan pengiriman data port serial melalui register

SBUF. Penulisan ke SBUF berarti mengisi register pengiriman SBUF sedangkan pembacaan dari SBUF berarti membaca register penerimaan SBUF yang memang terpisah secara fisik (secara perangkat lunak namanya menjadi satu yaitu SBUF).

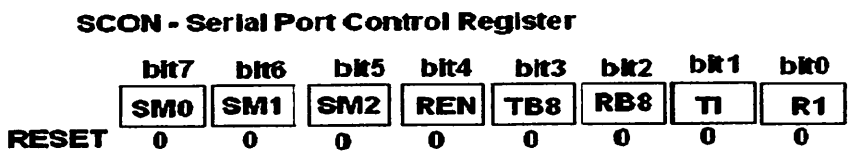
Port serial pada AT89S52 bisa digunakan dalam 4 mode kerja yang berbeda. Dari 4 mode tersebut, 1 mode diantaranya bekerja secara sinkron dan 3 lainnya bekerja secara asinkron. Keempat mode kerja tersebut adalah :

- 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 AT89S52. 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 (bit 7). Kecepatan pengiriman data (*baudrate*) adalah 1/12 frekuensi kristal yang digunakan.
- 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/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 AT89S52 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 UART (*Universal Asynchronous Receiver/Transmiter*).

- Mode 2 : Data dikirim/diterima 11 bit sekaligus, diawali dengan 1 bit *start*, disusul 8 bit data yang dimulai dari bit yang bobotnya paling kecil (bit 0), kemudian bit ke 9 yang bisa diatur lebih lanjut, diakhiri dengan 1 bit *stop*. Pada AT89S51 yang berfungsi sebagai penerima, bit 9 ditampung pada bit RB8 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 saja kecepatan pengiriman data (*baud rate*) bisa diatur sesuai dengan keperluan, seperti halnya Mode 1. Pada mode asinkron (mode 1, mode 2, mode 3), port AT89S51 bekerja secara *fullduplex*.

Register kontrol dan status untuk port serial dalam SCON register ini mengandung bit-bit pemilihan mode kerja port serial, bit data ke-9 pengiriman dan (TB8 dan RB8) serta bit-bit interupsi port serial (TI dan RI).



Gambar 2.9. Register SCON[2]

Keterangan:

- SM0: Serial port mode bit 0, bit pengubah mode serial.
- SM1: Serial port mode bit 1, bit pengatur mode serial.
- SM2: Serial port mode bit 2, bit untuk mengaktifkan komunikasi multiprosessor pada kondisi set.
- REN: *Receive* Enable, bit untuk mengaktifkan penerimaan data dari port serial pada kondisi set.
- TB8: Transmit bit 8, bit ke-9 yang akan dikirim pada mode 2 atau mode 3.
- RB8: *Receive* bit 8, bit ke-9 yang diterima pada mode 2 atau mode 3. Pada mode 1 bit ini berfungsi sebagai *stop* bit.
- TI: *Transmit Interupt Flag*, bit yang akan di set pada akhir pengiriman karakter.
- RI: *Receive Interupt Flag*, bit yang akan diset pada akhir penerimaan karakter.

Tabel 2.10. Register Kontrol Port Serial[2]

SM0	SM1	MODE	Keterangan	Baudrate
0	0	0	Register Geser	Tetap (fosc/12)
0	1	1	UART 8-Bit	Bias Diubah-ubah (dengan timer)
1	0	2	UART 9-Bit	Tetap (fosc/64 atau fosc/32)
1	1	3	UART 9-Bit	Bisa Diubah-ubah (dengan timer)

- Bit SM0 dan SM1 (bit7 dan 6 pada register SCON) dipakai untuk menentukan mode kerja port serial. Setelah reset kedua bit ini bernilai '0' dan penentuan mode kerja port serial mengikuti tabel di atas.
- Bit REN (bit4) dipakai untuk mengaktifkan kemampuan port serial untuk menerima data. Pada mode 0 kaki RxD (P3.0) dipakai untuk mengirim data serial. Dan juga untuk menerima data serial. Sifat ini terbawa pula pada saat port serial bekerja pada mode 1,2 dan 3, meskipun pada mode–mode tersebut kaki RxD hanya dipakai untuk mengirim data, agar kaki RxD bisa dipakai untuk menerima data, terlebih dulu harus dibuat REN = '1'. Setelah reset bit REN bernilai '0'.
- Pada mode 2 dan mode 3, port serial bekerja dengan 9 bit data (dari 11 bit, 1 bit untuk *start* dan 1 bit untuk *stop*), SBUF yang kapasitasnya 8 bit tidak cukup untuk keperluan ini. Bit ke-sembilan yang akan dikirim terlebih dulu diletakkan di TB8 (bit 3), sedangkan bit RB8 (bit2) merupakan bit yang dipakai untuk menampung bit ke-sembilan yang diterima port serial.
- Pada mode 1, RB8 dipakai untuk menampung bit *stop* yang diterima, dengan demikian apabila RB8 bernilai '1' maka data diterima dengan benar, sebaliknya apabila RB8 = '0' berarti terjadi kesalahan *frame* (*framing error*). Kalau bit SM2(bit 5) bernilai '1' pada mode 1, jika terjadi kesalahan frame, RI tidak akan menjadi

'1' (aktif) meskipun SBUF sudah berisi data dari port serial (bit *stop* diterima dengan benar). Bit ke 9 ini bisa dipakai sebagai bit paritas, hanya saja bit paritas yang dikirim harus ditentukan sendiri dengan program dan diletakkan pada TB8 dan bit paritas yang diterima pada RB8 dipakai untuk menentukan integritas data secara program pula. Tidak seperti dalam UART standart. Semuanya dikerjakan oleh perangakat keras dalam IC UART.

- Bit T1 (bit 1) merupakan sinyal yang setara dengan sinyal *Transmiter Holding Register Empty* (THRE) yang umum dijumpai pada UART standard. Setelah port seial selesai mengirim data yang tersimpan dalam SBUF, bit T1 akan bernilai '1' dengan sendirinya, kemudian bit ini harus di-nol-kan dengan program agar bisa dipakai untuk memantau keadaan SBUF dalam pengiriman data berikutnya.
- Bit RI (bit0) merupakan sinyal yang setara dengan sinyal RDA (*Receiver Data Available*) yang umum dijumpai pada UART standart. Setelah SBUF menerima data dari port serial, bit RI akan bernilai '1' dengan sendirinya, bit ini harus di-nol-kan dengan program agar bisa dipakai untuk memantau keadaan SBUF dalam penerimaan data berikutnya.

2.3.4. Pengaturan Baud Rate Serial

Baud rate dari port serial dapat diatur pada mode 1 dan mode 3, namun pada Mode 0 dan Mode 2 *baud rate* tersebut mempunyai

kecepatan yang permanen yaitu untuk mode 0 adalah 1/12 frekwensi osilator dan mode 2 adalah 1/32 frekwensi osilator.

Dengan mengubah bit SMOD yang terletak pada register PCON menjadi set (kondisi awal saat sistem reset adalah clear), *baud rate* pada mode 1,2, dan 3 akan berubah menjadi dua kali lipat.

Tabel 2.11. Rumus Perhitungan Baudrate Pada Komunikasi Serial[2]

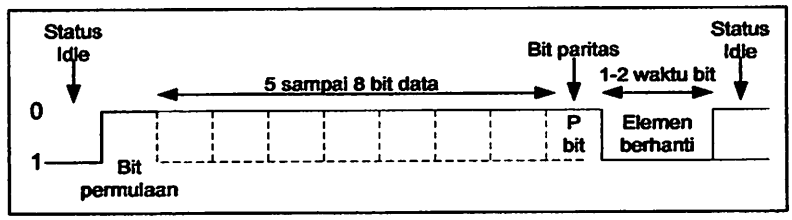
Mode	Baud Rate	
0	½ f Oscilator	
1	SMOD = 0 $Baudrate = \frac{f\ osc}{12 \times [256 - TH1] \times 32}$	SMOD = 1 $Baudrate = \frac{f\ osc}{12 \times [256 - TH1] \times 16}$
2	1/32 f	1/32 f
3	$Baudrate = \frac{f\ osc}{12 \times [256 - TH1] \times 32}$	$Baudrate = \frac{f\ osc}{12 \times [256 - TH1] \times 16}$

2.3.5. Komunikasi serial *Asynchronous*

Transmisi *asynchronous* digunakan untuk menghindari permasalahan yang berkaitan dengan waktu yaitu dengan cara tidak mengirimkan deretan bit yang panjang dan tidak putus-putus, tetapi data ditransmisikan satu karakter sekaligus, dimana setiap karakter panjangnya lima sampai delapan bit. Waktu atau sinkronisasi harus dipertahankan hanya didalam setiap karakter.

Gambar 2.12 memberi penjelasan mengenai teknik ini. Status *idle* diberikan bila tidak ada karakter yang ditransmisikan yang ekivalen dengan biner 1. permulaan karakter ditandai dengan suatu start bit

dengan nilai biner 0. Ini diikuti dengan lima sampai delapan bit yang merupakan suatu karakter. Kemudian diikuti dengan bit paritas yang tersusun berdasarkan posisi bit yang paling signifikan. Jumlah total bit-bit dalam karakter, termasuk bit paritas (ganjil maupun genap) tergantung ketentuan yang digunakan. Yang dimaksud elemen akhir adalah stop elemen yang berupa biner 1. Panjang minimum untuk elemen akhir ditentukan biasanya 1, 1,5 atau 2 kali durasi bit biasa.



Gambar 2.10. Format Frame *Asynchronous*[]

2.4. LCD (*Liquid Crystal Display)* M1632

LCD (liquid Crystal Display) adalah suatu jenis piranti output yang menggunakan daya rendah dengan pengontrol kontras dan kecerahan. Pengontrol utamanya dan karakter ada pada ROM (Read Only Memory) generator dan display data RAM (Random Access Memory) yang akan menghasilkan extended key codes (kode tombol/keyboard standart internasional dalam Hexsa) jika padanya diberikan inputan. Untuk mendapatkan fungsi dengan baik maka perlu diperhatikan proses inisialisasi yang telah ditentukan oleh pabrik pembuatnya. Timming penginisialisasian sangat perlu dipertimbangkan, karena jika meleset sampai orde millisecond, maka dapat dipastikan LCD itu tidak dapat berfungsi.

Ada dua jenis register yang terdapat dalam LCD M1632 ini, yaitu data register dan instruction register. Dengan menggunakan pin RS (Register Select)

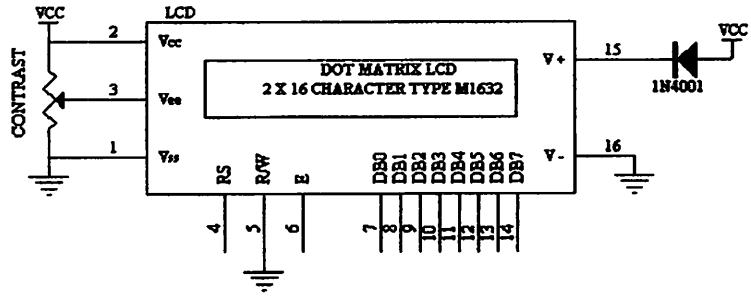
pada LCD, pemakaian kedua register dapat dipilih. Pemilihan register dapat dipilih. Pemilihan register pada LCDditunjukkan dalam tabel berikut ini :

Tabel 2.12. Pemilihan Register Pada LCD M1632[8]

Nama Sinyal	No. Terminal	I/O	Tujuan	Keterangan Sinyal
RS	4	Input	MPU	0 : Instruction Register 0 : Data Register

Jika bagian yang dipilih adalah instruction register maka output yang dihasilkan adalah meliputi operasional dari LCD, misalnya fungsi display clear, cursor home, entry mode set, display on/off, cursor shift, dan sejenisnya. Sebaliknya, jika bagian yang dipilih adalah data register, output yang dihasilkan adalah meliputi karakter yang tabelnya terdapat pada lampiran data sheet LCD.

Berikut adalah gambar dari LCD dengan pin- pin yang terhubung dengan mikrokontroller AT89S52 :



Gambar 2.11. Modul LCD 2 ×16 karakter[8]

LCD M1632 mempunyai spesifikasi sebagai berikut :

- 16 karakter 2 baris dalam bentuk dot matrik 5 × 7 dan kursor.
- *Duty ratio* 1/16.
- Memiliki ROM pembangkit karakter untuk 192 jenis karakter.
- RAM untuk data *display* sebanyak 80 × 8 bit (80 karakter maksimum).
- Dapat dirangkai dengan MPU (*Mikroprocessor Unit*) 8 bit atau 4 bit.
- RAM data *display* dan RAM pembangkit karakter dibaca oleh MPU.
- Memiliki fungsi intruksi : *display ON/OFF, cursor ON/OFF, display character blink, cursor shift dan display shift.*
- Memiliki rangkaian oscillator sendiri.
- Sumber tegangan tunggal +5 volt.
- Memiliki rangkaian reset otomatis pada catu daya dihidupkan.
- Temperature operasi 0^{o}-50^{o} C.

LCD modul M1632 mempunyai 16 pin dengan fungsi sebagai berikut :

Tabel 2.13. Fungsi Pin – Pin LCD[8]

No. PIN	Nama PIN	Fungsi
1	Vss	Terminal Ground
2	Vcc	Tegangan Catu + 5 volt
3	Vee	Mengendalikan kecerahan LCD
4	RS	Sinyal pemilihan register 0 = Tulis 1 = Baca
5	R/W	Sinyal seleksi tulis atau baca

		0 = Tulis 1 = Baca
6	E	Sinyal operasi awal yang mengaktifkan data tulis atau baca
7 – 14	DB0 – DB7	Merupakan saluran data berisi perintah data yang akan ditampilkan
15	V + BL	Back Light Supply
16	V – BL	Back Ligth Supply (Ground)

Pada LCD juga terdapat instruksi – instruksi sebagai berikut :

- *Display clear* : membersihkan tampilan yang ada pada LCD serta menyimpan, sedangkan kursor kembali ke posisi semula.
- *Cursor home* : hanya membersihkan tampilan dan kursor kembali ke semula.
- *Empty mode Set* : layar beraksi sebagai tampilan tulis.

  S : 1/0 = menggeser layar.

  1/0 : 1 = kursor bergerak ke kanan dan layar bergerak ke kiri.

  1/0 : 0 = kursor bergerak ke kiri dan layar bergerak ke kanan

- *Display On/Off* kontrol.

  D : 1 = layar on

  D : 0 = layar off

  C : 1 = kursor on

C : 0 = kursor off

B : 1 = kursor berkedip-kedip

B : 0 = kursor tidak berkedip-kedip

- *Cursor Display Shift*

  S/C : 1 = LCD diidentifikasikan sebagai layar

  S/C : 0 = LCD diidentifikasikan sebagai kursor

  R/L : 1 = menggeser satu spasi ke kanan

  R/L : 0 = menggeser satu spasi ke kiri

- *Fuction Set*

  DL : 1 = panjang data LCD pada 8 bit

  DL : 0 = panjang data LCD pada 4 bit

  Bit upper ditransfer terlebih dahulu kemudian diikuti dengan 4 bit lower.

  N : 1/0 = LCD menggunakan 2 atau 1 baris karakter

  P : 1/0 = LCD menggunakan 5 x 10 dot matrik

- *CG RAM address set* : menulis alamat RAM ke karakter
- *DD RAM address set* : menulis alamat RAM ke tampilan
- *BF/address set* : BF = 1/0, LCD dalam keadaan sibuk atau tidak sibuk.
- *Data write to CG RAM or DD RAM* : membaca byte dari alamat terakhir RAM yang dipilih.

2.4.1. Sinyal Interface M1632

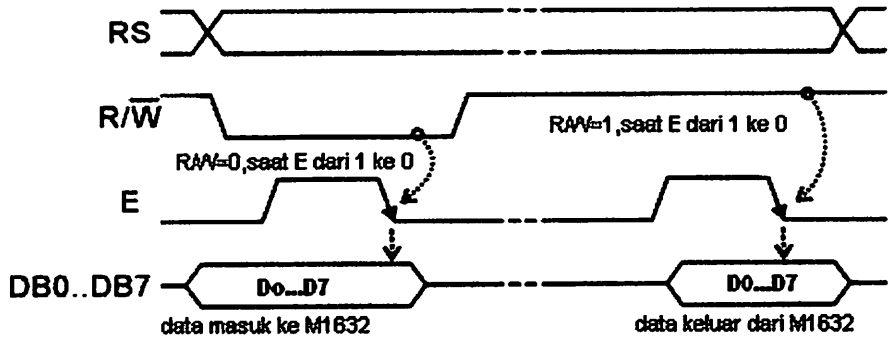
Untuk berhubungan dengan mikrokontroler pemakai, M1632 dilengkapi dengan 8 jalur data (**DB0..DB7**) yang dipakai untuk menyalurkan kode ASCII maupun perintah pengatur kerjanya M1632. Selain itu dilengkapi pula dengan **E**, **R/W** dan **RS** seperti layaknya komponen yang kompatibel dengan mikroprosesor.

Kombinasi lainya **E** dan **R/W** merupakan sinyal standar pada komponen buatan Motorola. Sebaliknya sinyal-sinyal dari MCS51 merupakan sinyal khas Intel dengan kombinasi sinyal **WR** dan **RD**..

RS, singkatan dari *Register Select,* dipakai untuk membedakan jenis data yang dikirim ke M1632, kalau **RS=0** data yang dikirim adalah perintah untuk mengatur kerja M1632, sebaliknya kalau **RS=1** data yang dikirim adalah kode ASCII yang ditampilkan.

Demikian pula saat pengambilan data, saat **RS=0** data yang diambil dari M1632 merupakan data status yang mewakili aktivitas M1632, dan saat **RS=1** maka data yang diambil merupakan kode ASCII dari data yang ditampilkan.

Proses mengirim/mengambil data ke/dari M1632 digambarkan dalam gambar 2-13 bisa dijabarkan sebagai berikut :



Gambar 2.12. Mengirim/Mengambil Data Ke/Dari M1632[8]

1. **RS** harus dipersiapkan dulu, untuk menentukan jenis data seperti yang telah dibicarakan di atas.
2. **R/W** di-nol-kan untuk menandakan akan diadakan pengiriman data ke M1632. Data yang akan dikirim disiapkan di **DB0..DB7**, sesaat kemudian sinyal **E** di-satu-kan dan di-nol-kan kembali. Sinyal **E** merupakan sinyal sinkronisasi, saat **E** berubah dari 1 menjadi 0 data di **DB0 .. DB7** diterima oleh M1632.
3. Untuk mengambil data dari M1632 sinyal **R/W** di-satu-kan, menyusul sinyal **E** di-satu-kan. Pada saatu **E** menjadi 1, M1632 akan meletakkan datanya di **DB0 .. DB7**, data ini harus diambil sebelum sinyal **E** di-nol-kan kembali.

2.5. Transistor

Transistor merupakan salah satu komponen aktif karena dapat memperkuat sinyal masukan dan menghasilkan suatu sinyal keluaran yang lebih besar. Untuk mengoperasikan sebuah transistor dalam suatu rangkaian linear diperlukan beberapa syarat sebagai berikut :

1. diode emitor harus bias maju
2. diode kolektor harus dibias balik

Untuk membuat transistor berfungsi dengan baik (perlu mengetahui karakteristik transistor dengan mengetahui bentuk kurva transistor dan garis bebannya. Dalam hal ini akan diketahui fungsi transistor sebagai saklar.

2.5.1. Transistor Sebagai Saklar

Cara yang paling mudah menggunakan sebuah transistor adalah sebagai saklar, artinya transistor dioperasikan pada salah satu dari saturasi atau *cut off* Jika transistor berada pada titik saturasi, transistor tersebut seperti sebuah saklar yang tertutup dari kolektor ke emitor. Jika transistor *cut* off 'maka transistor akan seperti sebuah saklar yang terbuka.

Transistor yang difungsikan sebagai saklar diperlihatkan dalam Gambar 2.14. Pada rangkaian tersebut merupakan penjumlahan tegangan disekitar *loop input,* sehingga diperoleh persamaan:

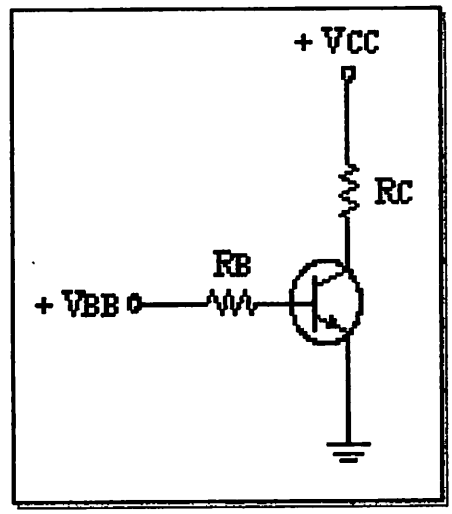
$I_B.R_B+V_{BE}-V_{BB}O$ .....(2-1)

$k.Rc + V_{CE} - Vcc = 0$ .....(2-2)

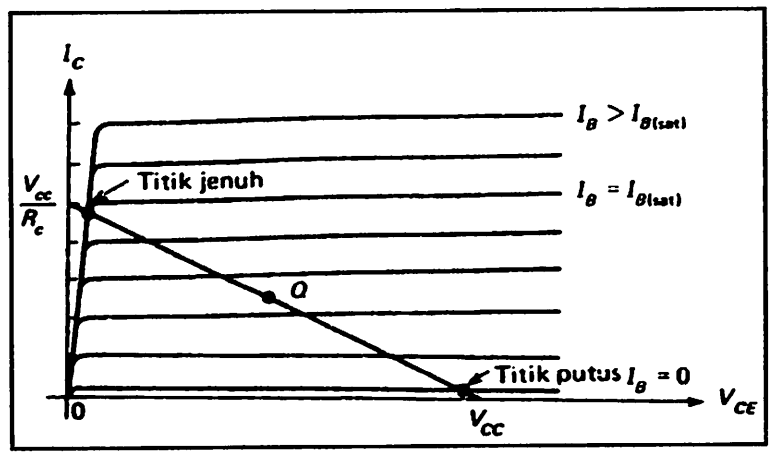
Sehingga dengan persamaan (2-1) didapat persamaan untuk mengetahui besar arus pada kutub basis (I_B). Maka persamaan untuk arus pada basis dalam rangkaian transistor sebagai saklar adalah:

$$I_B = \frac{V_{BB} - V_{BE}}{R_B} \quad \text{.....(2-3)}$$

Jika arus basis lebih besar atau sama besar dengan I_B (sat), titik kerja Q berada pada ujung atas dari garis beban (Gambar 2.7). Dalam hal ini transistor kelihatan seperti sebuah saklar yang tertutup. Sebaliknya jika arus basis nol, transistor bekerja pada ujung bawah dari garis beban, dan transistor kelihatan seperti sebuah saklar yang terbuka.



Gambar 2.13. Rangkaian transistor sebagai saklar[12]



Gambar 2.14. Titik jenuh dan titik putus pada garis beban dc [12]

Titik potong antara garis beban dc dan kurva $I_B = 0$ dikenal sebagai

titik putus *(cut off)*. Pada titik ini, arus basis sama dengan nol, dan arus kolektor juga sangat kecil. Temyata dalam keadaan ini, dioda emiter tidak lagi mendapat prategangan maju dan fungsi normal dari transistor menjadi terhenti. Perhatikan bahwa titik putus dalam Gambar 2.15 sangat berdekatan dengan intersep datar dari garis beban dc.

Titik jenuh *(saturasion)* adalah titik potong dengan kurva I_B pada ujung teratas dari garis beban dc. Pada titik ini, diode kolektor tepat berada pada batas kehilangan prategangan-balik. Untuk mudahnya, titik jenuh akan diberi koordinat Ic(.t) dan Dengan nilai resistansi relay yang diketahui sebesar R_{relay} maka didapat :

$$I_{relay} = \frac{Vcc - Vce}{R_{relay}} \quad \text{.....(2-4)}$$

Jika *Irelay = Ic*, dengan nilai hfe minimum maka nilai Ib minimum yang diperlukan agar transistor dalam keadaan saturasi adalah :

$$I_b = \frac{I_C}{hfe} \quad \text{..... (2-5)}$$

Dengan *Vbe = 0,7 V* maka nilai tahanan Rb

dapat dicari yaitu :

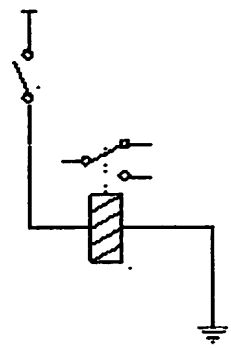
$$R_b = \frac{Vbb - Vbe}{I_b} \quad \text{.....(2-6)}$$

Bila arus basis melebihi $I_{B\,(sat)}$, arus kolektor tidak dapat bertambah lagi karena diode kolektor tidak lagi berfungsi (tidak mendapat prategangan balik). Dengan kata lain, merupakan harga maksimum

dari arus kolektor yang dapat dihasilkan oleh rangkaian dengan prategangan basis pada tegangan catu dan hambatan kolektor tertentu.

2.6. Relay

Relay adalah komponen elektronika yang terdiri dari sebuah lilitan kawat (kumparan/koil) yang terlilit pada sebuah besi lunak. Jika kumparan dialiri arus listrik maka inti besi akan menjadi magnet dan menarik pegas sehingga kotak AB terhubung dan BC terputus.



Gambar 2.15. Cara Kerja Relay[12]

Relay merupakan suatu alat untuk menghubungkan atau memerlukan kontak antara komponen yang satu dengan yang lain. Dalam memutus atau menghubungkan kontak digerakkan oleh *fluksi* yang ditimbulkan dari adanya medan magnet listrik yang dihasilkan oleh kumparan yang melilit pada besi lunak.

BAB III

PERANCANGAN DAN PEMBUATAN ALAT

3.1. Pendahuluan

Dalam bab ini akan dibahas perancangan dan pembuatan alat. Pembahasan akan dilakukan pada setiap blok rangkaian, cara kerja masing-masing blok rangkaian, perhitungan dan juga fungsi masing-masing blok rangkaian tersebut. Secara garis besar terdapat dua bagian perangkat yang ada yaitu :

- Perancangan perangkat keras *(Hardware).*
- Perancangan perangkat lunak *(Software).*

Pada perancangan perangkat keras akan meliputi *peripheral-peripheral* yang digunakan pada sistem ini. Sedangkan pada perancangan perangkat lunak akan meliputi diagram alir dan *software* secara umum. Akan tetapi kedua perangkat ini dalam kerjanya saling menunjang satu sama lain..

3.1.1. Spesifikasi Alat

Spesifikasi "Alat Pelacak Posisi Dengan GPS Dan Penonaktifan Mesin Mobil Dengan Kendali Handphone Pada Perusahaan Persewaan Mobil" yang dirancang adalah sebagai berikut:

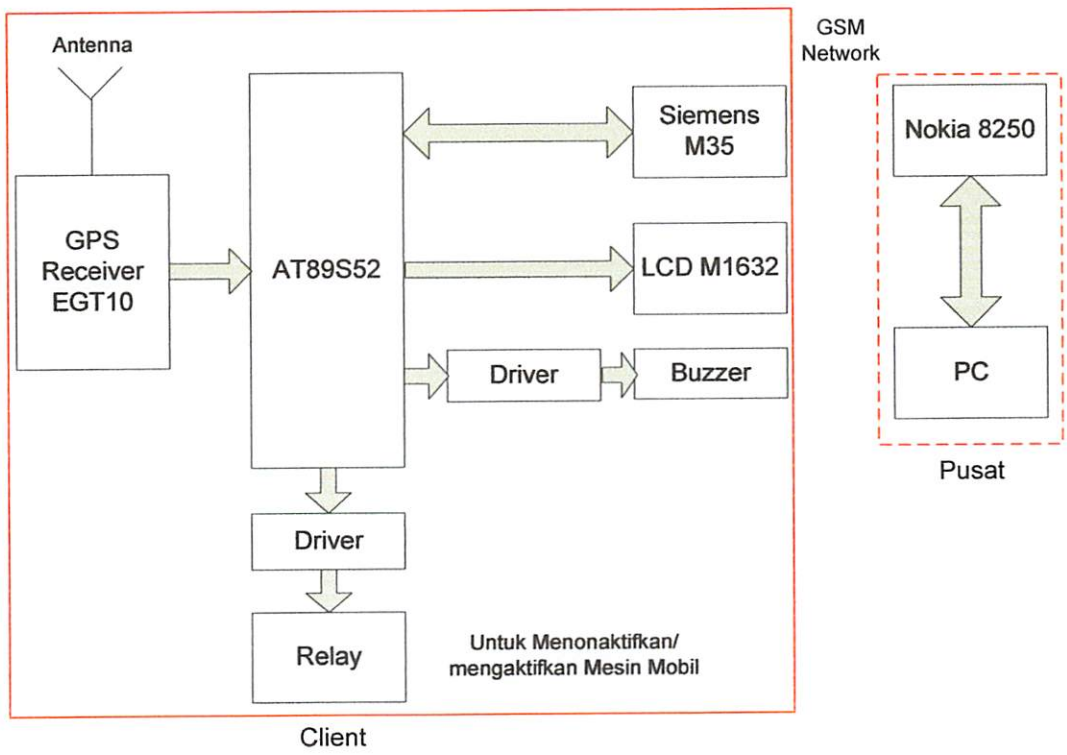
1. Alat ini menggunakan sistem minimum Mikrokontroller AT89S52 sebagai pengontrol utama.
2. Menggunakan telepon selular Siemens M35 dan Nokia 8250 sebagai pengirim/penerima informasi.
3. Komunikasi telepon selular dengan unit mikrokontroller menggunakan komunikasi serial asinkron dengan *baudrate* 19200 bps. Sedangkan

komunikasi modul GPS *receiver* dengan mikrokontroller menggunakan komunikasi serial asinkron dengan *baudrate* 4800 bps.

4. Menggunakan *Relay* untuk menonaktifkan/mengaktifkan mesin mobil.
5. Catu daya untuk alat, diperoleh dari baterai/accu pada kendaraan sebesar 12 volt.

3.1.2. Blok Diagram Keseluruhan Sistem

Perancangan dan pembuatan alat agar dapat dilakukan secara sistematis dan terstruktur maka perlu dibuat blok diagram yang menjelaskan dari sistem yang dirancang. Secara garis besar sistem perancangan ditunjukkan pada diagram blok dari Gambar 3.1.



Gambar 3.1 Blok Diagram Keseluruhan Sistem

Keterangan fungsi dari masing-masing blok di atas adalah sebagai berikut :

1. Handphone (Siemens M35)

   Berfungsi sebagai penerima perintah berupa SMS dari Pusat dan mengirimkan informasi Koordinat posisi yang berupa SMS ke pusat.

2. Handphone (Nokia 8250)

   Berfungsi sebagai pengirim perintah berupa SMS ke Client dan menerima informasi Koordinat posisi yang berupa SMS dari Client.

3. GPS *Receiver* EG-T10

   Berfungsi untuk menerima data koordinat posisi dari satelit.

4. LCD M1632

   Berfungsi untuk menampilkan pesan yang berisi informasi tentang masa persewaan mobil yang dikirimkan dari pusat.

5. Mikrokontroller AT89S52

   Mikrokontroller ini berfungsi sebagai pengontrol utama semua system.

6. Driver Relay

   Rangkaian ini berfungsi untuk mengaktifkan relay sesuai dengan data dari mikrokontroler.

7. Relay

   Relay ini berfungsi untuk memutus/menghubungkan aliran listrik dari accu mobil ke *Ignition Coil.*

8. Driver Buzzer

   Berfungsi untuk mengaktifkan buzzer sesuai dengan data dari mikrokontroller.

9. Buzzer

   Berfungsi untuk memberikan tanda apabila ada pesan tentang masa persewaan mobil yang masuk.

10. PC (*Personal Computer*)

    Berfungsi sebagai pengolah data perusahaan persewaan dan untuk menampilkan posisi mobil berdasarkan koordinat yang diterima dengan program Google Earth.

3.1.3. Rancangan Proses Kerja Alat

Pada perancangan dan pembuatan alat ini, sistem ini akan siap bekerja bila keseluruhan komponen pembentuknya yang meliputi *software* aplikasi utama, *modem (handphone)* dan Mikrokontroler sudah diaktifkan semua. Berdasarkan gambar 3.1, maka didapatkan suatu langkah kerja sistem yaitu :

a. Sistem Pada Pusat

   Sistem pada pusat ini terdiri dari sebuah PC dan Handphone yang dihubungkan dengan komunikasi Infra Red atau kabel data. PC dioperasikan oleh operator dengan sebuah program interface yang bisa mengirimkan perintah-perintah ke sistem Client melalui handphone. Perintah-perintah yang dapat dikirimkan ke Client ada empat macam perintah yaitu:

1. Perintah Permintaan Lokasi.

   Pusat dapat mengirimkan perintah berupa SMS dengan format (M<spasi>PoS) ke sistem Client yang berisi permintaan untuk mengirimkan lokasi terakhir dari Client ke Pusat.

2. Perintah Pengiriman Pesan.

   Pusat dapat mengirimkan pesan yang berisi informasi status persewaan kepada Client. Isi pesan yang dikirimkan adalah :

   - M<spasi>Kr1 (isi pesan "Peringatan Waktu Sisa 3 Jam)
   - M<spasi>Kr2 (isi pesan "Peringatan Waktu Sisa 2 Jam)
   - M<spasi>Kr3 (isi pesan "Peringatan Waktu Sisa 3 Jam)

3. Perintah Untuk Menonaktifkan Mesin Mobil.

   Pusat dapat mengirimkan perintah berupa SMS dengan format (M<spasi>OfF) ke sistem Client yang berisi perintah untuk menonaktifkan mesin mobil.

4. Perintah Untuk Mengaktifkan Mesin Mobil.

   Pusat dapat mengirimkan perintah berupa SMS dengan format (M<spasi>OnN) ke sistem Client yang berisi perintah untuk mengaktifkan kembali mesin mobil.

Format penulisan SMS harus benar, penulisan mengunakan huruf kecil atau besar dibedakan. Penulisan SMS yang tidak benar menyebabkan perintah tidak dapat dikenali oleh mikrokontroller.

Selain dapat mengirimkan perintah, Pusat juga dapat menerima data yang dikirimkan oleh Client. Data yang dikirimkan oleh Client

tersebut berupa SMS yang berisi informasi koordinat lokasi terakhir dari Client yang diminta. Informasi koordinat lokasi tersebut kemudian dimasukkan kedalam program Google Earth untuk melihat lokasi Client melalui citra satelit.

b. Sistem Client

Sistem Client ini terdiri dari: (1) Modul GPS, sebagai penerima data koordinat dari satelit, (2) minimum sistem Mikrokontroler AT89S52, sebagai pemroses utama, (3) handphone Siemens M35, digunakan untuk menerima perintah dan mengirimkan data, (4) LCD M1632, sebagai penampil pesan secara visual, (5) Buzzer, Sebagai pemberi tanda apabila ada pesan tentang masa persewaan masuk, (6) Relay, dihubungkan dengan sistem kelistrikan mobil yang digunakan untuk mengaktifkan dan menonaktifkan mesin mobil.

Sistem Client ini bekerja berdasarkan perintah dari Pusat, dimana perintah tersebut dikirimkan dalam format SMS. SMS yang diterima akan di proses oleh Mikrokontroler untuk diseleksi apakah SMS yang diterima formatnya sesuai atau tidak. Jika SMS yang diterima formatnya sesuai, maka SMS yang diterima akan di proses oleh Mikrokontroler untuk diseleksi kembali apa yang diperintahkan oleh Pusat. Perintah yang dapat diproses oleh Sistem Client ada 4 (empat) macam yaitu :

1. Perintah Untuk Mengirimkan Koordinat Lokasi Client.

   Jika perintah yang diterima adalah perintah untuk mengirimkan koordinat lokasi (M<spasi>PoS), maka mikrokontroller akan mengambil data koordinat lokasi terakhir dari modul GPS kemudian mengubahnya dalam format SMS dan mengirimkannya melalui handphone ke Pusat.

2. Perintah Untuk Menampilkan Pesan Pada LCD.

   Jika perintah yang diterima adalah perintah untuk menampilkan pesan pada LCD (M<spasi>Kr1/M<spasi>Kr2/M<spasi>Kr3), maka mikrokontroller akan memproses data SMS yang diterima kemudian menampilkanya pada LCD sehingga dapat dibaca oleh penyewa, dan juga menyalakan Buzzer sehingga konsumen dapat mengetahui ada pesan yang masuk.

3. Perintah Untuk Menonaktifkan Mesin Mobil.

   Jika perintah yang diterima adalah perintah untuk menonaktifkan mesin mobil (M<spasi>OfF), maka mikrokontroller akan mengaktifkan Relay. Dengan aktifnya relay tersebut maka sistem kelistrikan pada mobil akan terputus (mati), sehingga mesin mobil akan mati.

4. Perintah Untuk Mengaktifkan Mesin Mobil.

   Jika perintah yang diterima adalah perintah untuk mengaktifkan mesin mobil (M<spasi>OnN), maka mikrokontroller akan menonaktifkan Relay. Dengan tidak aktifnya relay tersebut maka

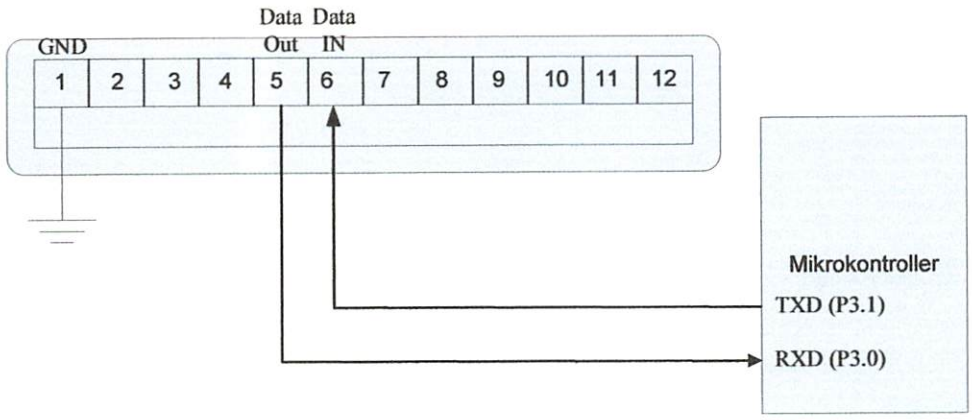
sistem kelistrikan pada mobil akan terhubung kembali (hidup), sehingga mesin mobil dapat dihidupkan kembali.

3.2. Perancangan Perangkat Keras (*Hardware)*

3.2.1. Perancangan komunikasi data Handphone dengan Mikrokontroller

Pada sistem digunakan *handphone* Siemens tipe M35. *Handphone* ini berperan sebagai konverter yang menjadi perantara pertukaran data dalam bentuk SMS antara HP (*handphone*) dengan mikrokontroller maupun sebaliknya, sehingga data dapat dikomunikasikan melalui jaringan GSM dan dapat diolah oleh mikrokontroller.

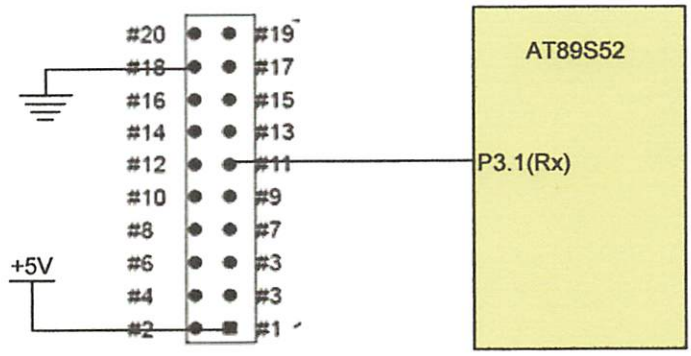
Pada *handphone* ini memiliki suatu *pin-out* yang berfungsi sebagai *interface* ke perangkat lain seperti mikrokontroller. Untuk pengiriman dan penerimaan data pada HP Siemens M35 menggunakan pin ke-5 untuk *Transmit* (Tx) dan pin ke-6 untuk *Receive* (Rx) serta pin ke-1 untuk *Ground* sesuai dengan gambar 3.2.



Gambar 3.2 Rangkaian Interface antara Handphone dengan Mikrokontroller

3.2.2. Perancangan komunikasi data GPS Receiver dengan Mikrokontroller

Pada sistem digunakan modul GPS Receiver EGT10 yang memiliki keluaran berupa data serial dengan level TTL dengan Baudrate sebesar 4800 bps. Untuk pengiriman data ke mikrokontroller digunakan pin ke-11 *Transmit* (Tx) yang dihubungkan dengan port 3.0 (Rxd) pada mikrokontroller dan pin ke-2 untuk VCC sebesar 5 volt serta pin ke-18 dihubungkan ke Ground seperti pada gambar 3.3.

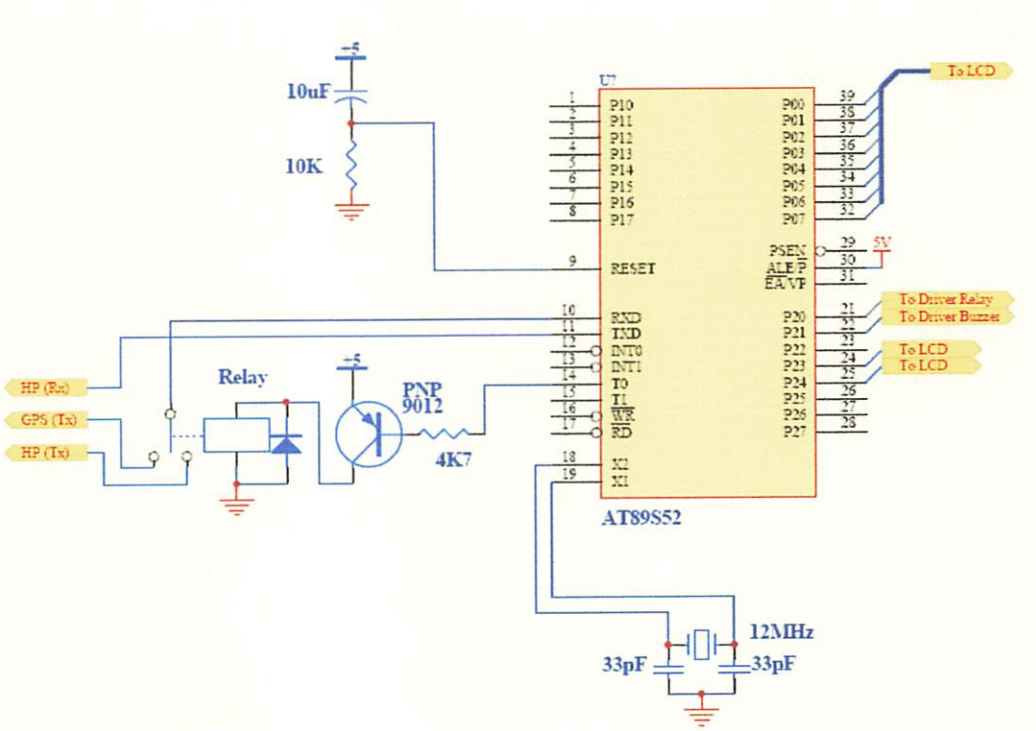


Gambar 3.3 Rangkaian interface GPS Receiver dengan AT89S52

3.2.3. Mikrokontroller AT89S52

3.2.3.1. Perancangan Penggunaan Port pada Mikrokontroller AT89S52

Rangkaian minimum dari mikrokontroler AT89S52 terdiri dari 3 kapasitor, 1 IC mikrokontroler, 1 resistor, dan 1 kristal. Dengan rangkaian yang sederhana ini mikrokontroler dibuat sebagai sistem minimum menjadi pengontrol alat, disamping itu rangkaian ini dapat dibuat bermacam-macam alat dengan menambah sedikit komponen tambahan lainnya. Dari rangkaian tersebut yang berpengaruh terhadap kecepatan proses menjalankan program adalah kristal. Adapun rangkaiannya ditunjukkan seperti pada Gambar 3.4.



Gambar 3.4 Minimum sistem AT89S52

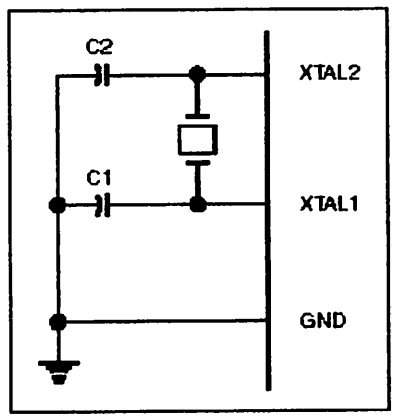
Agar sebuah mikrokontroler dapat bekerja sebagai pengontrol, maka kaki-kaki/*port* mikrokontroler dihubungkan dalam rangkaian-rangkaian eksternal. Dalam perancangan ini, *port* yang digunakan adalah sebagai berikut:

1. Port 0.0 – Port 0.7 sebagai output ke LCD M1632.
2. Port 2.0 digunakan sebagai output ke driver relay.
3. Port 2.1 digunakan sebagai output ke driver buzzer.
4. Port 3.0 digunakan sebagai input data serial dari HP dan GPS.
5. Port 3.1 digunakan sebagai output data serial ke HP.
6. Port 3.4 digunakan sebagai pengontrol relay untuk memilih input data serial apakah dari HP atau GPS.

7. X1 dan X2 digunakan sebagai input dari rangkaian osilator kristal. Rangkaian osilator kristal terdiri dari kristal osilator 11,0592 MHz, kapasitor C1 dan C2 yang masing-masing bernilai 33 pF, akan membangkitkan pulsa *clock* yang menjadi penggerak bagi seluruh operasi internal MCU.
8. VCC dihubungkan dengan tegangan sebesar +5V
9. GND dihubungkan ke *ground* catu daya.

3.2.3.2. Pengaturan *Baud Rate*

Di dalam proses komunikasi serial antara mikrokontroler dengan HP dan GPS terlebih dahulu ditentukan *baud rate* yang digunakan. Pada sistem ini digunakan *baud rate* sebesar 19200 bps dan 4800 bps dengan menggunakan f_{osc} = 11,0592 MHz.



Gambar 3.5 Rangkaian Osilator[2]

Cara menghitung pada register TH1: untuk *baudrate* 19200 bps

$$Baud\ Rate = \frac{fosc}{12 \times (256 - TH_1) \times 16}$$

$$19200 = \frac{11{,}0592Mhz}{12 \times (256 - TH_1) \times 16}$$

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

$$256 - TH_1 = \frac{11{,}0592x10^6}{3686400}$$

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

$$= 253$$

$$= FD_H$$

Cara menghitung pada register TH1: untuk *baudrate* 4800 bps

$$Baud\ Rate = \frac{fosc}{12 \times (256 - TH_1) \times 32}$$

$$4800 = \frac{11{,}0592Mhz}{12 \times (256 - TH_1) \times 32}$$

$$1843200 = \frac{11{,}0592Mhz}{256 - TH_1}$$

$$256 - TH_1 = \frac{11{,}0592x10^6}{1843200}$$

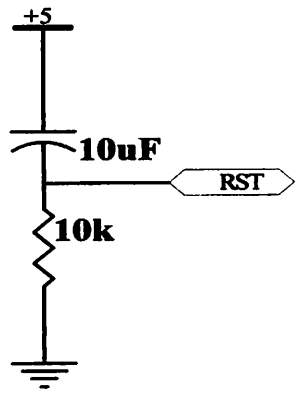
$$TH_1 = 256\text{-}6$$

$$= 244$$

$$= FA_H$$

3.2.3.3. Perancangan Rangkaian Reset

Pin reset pada mikrokontroller merupakan masukan aktif high (1). Pulsa transisi dari low (0) dan high (1) akan mereset mikrokontroller menuju alamat 0000H. pin reset dihubungkan dengan rangkaian power on reset yang diperlihatkan pada gambar 3-6 berikut ini:



Gambar 3.6 Rangkaian Reset untuk MCU AT89S52[2]

Karena kristal yang digunakan mempunyai frekuensi sebesar 11,0592 MHz maka satu periode membutuhkan waktu sebesar:

$$T = \frac{1}{f_{XTAL}}$$

$$= \frac{1}{11{,}0592MHz} S$$

$$= 9{,}042 \times 10^{-8} \text{ S}$$

Sehingga waktu minimal logika tinggi yang dibutuhkan untuk m*ereset* mikrokontroller adalah :

Reset (min) $= \tau \times$ periode yang dibutuhkan

$$= 9{,}0424 \times 10^{-8} \times 24 = 2{,}170 \ \mu s$$

Jadi mikrokontroller membutuhkan waktu minimal 2,170 μs untuk *mereset*. Waktu minimal inilah yang dijadikan pedoman untuk menentukan nilai R dan C. Dari persamaan konstanta waktu $\tau = R \times C$ (William H Hyat, 1998, h132 [1]) dan jika nilai R ditentukan sebesar 10 kΩ, maka nilai C adalah :

$$C = \frac{\tau}{R}$$

$$= \frac{2{,}170 \times 10^{-6}}{10 \times 10^{3}}$$

$$= 2{,}170 \times 10^{-12}\,\text{F}$$

Kapasitor minimal yang dibutuhkan adalah 2,170 pF. Dengan menggunakan kapasitor sebesar 10 μ F, maka akan menjamin waktu reset di atas nilai minimal waktu yang dibutuhkan untuk mereset mikrokontroller.

3.2.4. Perancangan Rangkaian LCD Dengan Mikrokontroller AT89S52

Untuk berhubungan dengan mikrokontroller, pemakai LCD M1632 dilengkapi dengan 8 jalur data (**DB0..DB7**) yang dipakai untuk menyalurkan kode ASCII maupun perintah pengatur kerjanya M1632. Selain itu dilengkapi pula dengan **E**, **R/W*** dan **RS** seperti layaknya komponen yang kompatibel dengan mikroprosesor.

RS, singkatan dari Register Select, dipakai untuk membedakan jenis data yang dikirim ke M1632, kalau **RS=0** data yang dikirim adalah perintah untuk mengatur kerja M1632, sebaliknya kalau **RS=1** data yang dikirim adalah kode ASCII yang ditampilkan.

Demikian pula saat pengambilan data, saat **RS=0** data yang diambil dari M1632 merupakan data status yang mewakili aktivitas M1632, dan saat **RS=1** maka data yang diambil merupakan kode ASCII dari data yang ditampilkan.

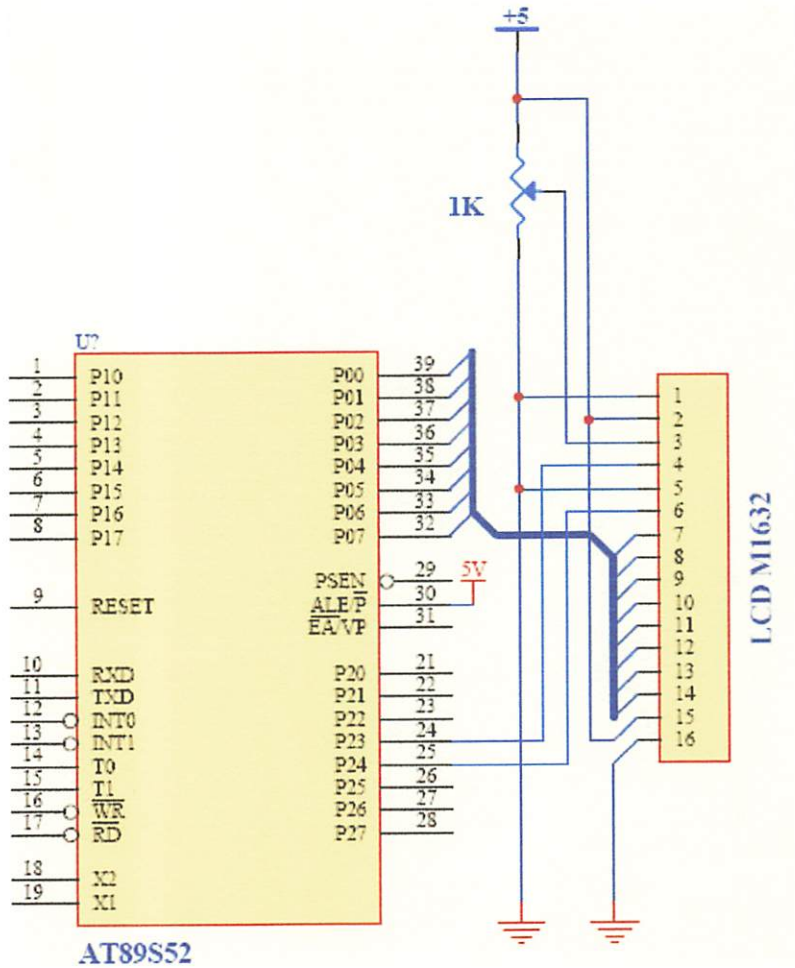
Proses mengirim/mengambil data ke/dari M1632 bisa dijabarkan sebagai berikut :

- **RS** harus dipersiapkan dulu, untuk menentukan jenis data seperti yang telah dibicarakan di atas.
- **R/W*** di-nol-kan untuk menandakan akan diadakan pengiriman data ke M1632. Data yang akan dikirim disiapkan di **DB0..DB7**, sesaat kemudian sinyal **E** di-satu-kan dan di-nol-kan kembali. Sinyal **E** merupakan sinyal sinkronisasi, saat **E** berubah dari 1 menjadi 0 data di **DB0 .. DB7** diterima oleh M1632.
- Untuk mengambil data dari M1632 sinyal **R/W*** di-satu-kan, menyusul sinyal **E** di-satu-kan. Pada saat **E** menjadi 1, M1632 akan meletakkan datanya di **DB0 .. DB7**, data ini harus diambil sebelum sinyal **E** di-nol-kan kembali.

M1632 mempunyai seperangkat perintah untuk mengatur tata kerjanya, perangkat perintah tersebut meliputi perintah untuk menghapus tampilan, meletakkan kembali cursor pada baris huruf pertama baris pertama, menghidup/matikan tampilan dan lain sebagainya, semua itu dibahas secara terperinci dalam Lembar Data M1632.

Untuk tampilan dipergunakan LCD Dot Matrik 2 x 16 karakter. Sinyal-sinyal yang diperlukan oleh LCD adalah RS dan Enable, sinyal RS dan Enable

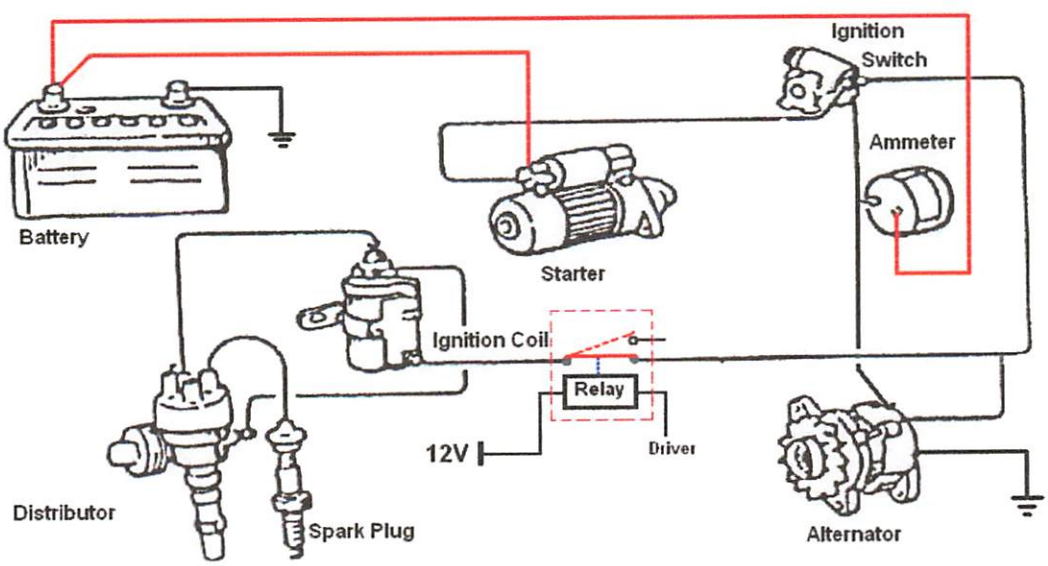
dipergunakan sebagai input yang outputnya dipakai untuk mengaktifkan LCD. LCD akan aktif apabila mikrokontroller memberikan instruksi tulis pada LCD. Saat kondisi RS don't care dan Enable 0 maka LCD tetap pada kondisi semula, pengiriman data ke LCD dilakukan saat RS berlogika 1 dan enable berlogika 1. Instruksi dikirim pada LCD bila keadaan RS 0 dan Enable 1. Pin LCD ini untuk data terkoneksi pada *Port 0* mikrokontoller. Kemudian untuk RS dihubungkan pada *Port 2.1*, tulis/baca (*Read/Write)* diberikan logika *low* karena disini LCD bersifat menulis data, dan yang terakhir *Enable* (E) dikendalikan dengan *Port 2.0*. Gambar rangkaian LCD ditunjukkan pada gambar 3.7 sebagai berikut :



Gambar 3.7. Rangkaian *interface* LCD M1632 dengan AT89S52

3.2.5. Perancangan Relay Sebagai Pemutus Arus Listrik Pada Mobil

Relay merupakan sebuah alat elektromagnetik yang dapat mengubah kontak-kontak saklar pada saat menerima aliran listrik. Relay terdiri dari suatu kumparan dan inti, jika kumparan tersebut dialiri listrik akan menimbulkan medan magnet sehingga dapat membuka atau menutup kontak-kontak. Kontak-kontaknya ada dua macam, yaitu NO (*Normally Open*) dan NC (*Normally Close*). Untuk *Normally Close* , kontak relay yang terhubung pada saat belum ada arus listrik yang mengalir ke kumparan relay. *Normally open,* kontak relay terhubung pada saat mendapat arus listrik. Pada sistem ini digunakan relay dengan anak kontak *Normally Close.*

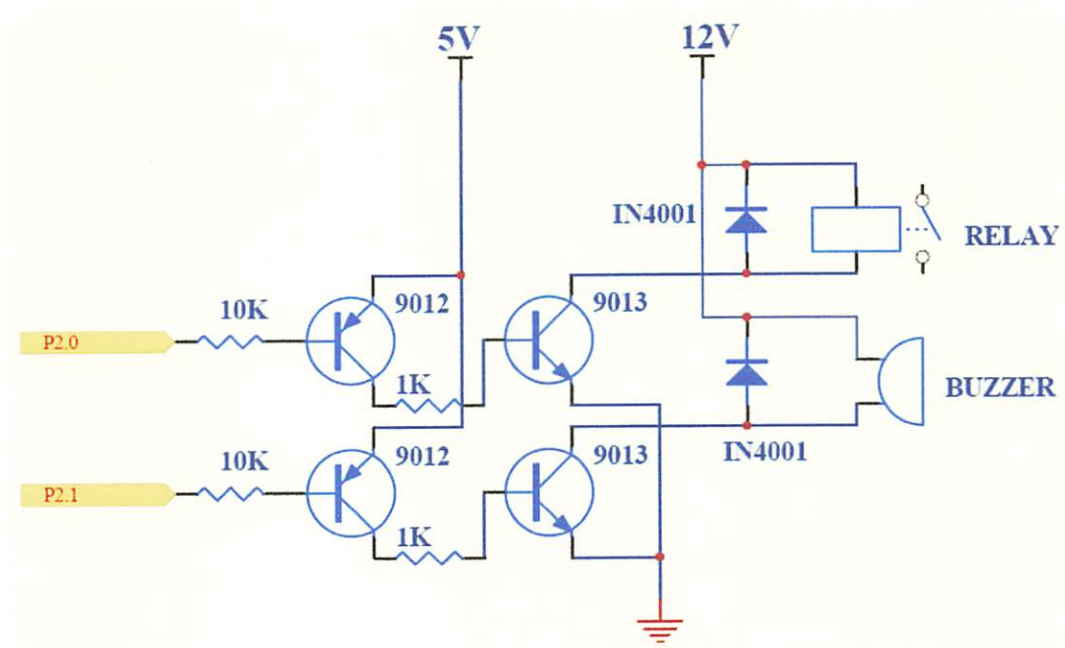


Gambar 3.8. Relay Sebagai Pemutus Arus Pada Mobil[13]

3.2.6. Perancangan Rangkaian Driver Relay Dan Buzzer

Rangkaian driver yang digunakan untuk menggerakkan relay dan buzzer terdiri dari transistor NPN 9013 dan transistor PNP 9012 yang difungsikan

sebagai transistor *switching*. Relay dan buzzer diberi dioda yang dirangkai paralel dengan relay dengan tujuan untuk menghindari tegangan mundur yang dibangkitkan oleh relay sehingga tidak merusak transistor. Rangkaian *driver* relay dan buzzer ditunjukkan dalam Gambar 3.10.



Gambar 3.9 Rangkaian Driver Relay dan Buzzer

. Diketahui dari data sheet transistor 9012 besarnya I_C = 50 mA, hfe = 120, V_{BE} = 0,7 Volt, sehingga :

$$I_C = I_B * hfe$$

$$I_B = \frac{I_C}{hfe}$$

$$I_B = \frac{50mA}{120} = 0{,}416mA$$

Maka R_B adalah :

$$R_B = \frac{Vcc - Vbe}{I_B}$$

$$R_B = \frac{5 - 0{,}7}{0{,}416mA}$$

$$= \frac{4{,}3}{0{,}416mA} = 10{,}33 \text{ x } 103\,\Omega$$

Jadi dari hasil perhitungan diatas diperoleh nilai R_B sebesar 10,33 x 10^3 Ω maka disesuaikan dengan resistor yang ada dipasaran yaitu 10 KΩ.

Dari transistor C9013 diketahui:

I_C = 500 mA

Hfe = 110

$V_{C(SAT)9012}$ = 0,16 Volt

V_{BE} = 0,7 Volt

Sehingga:

$$Ic = Hfe.I_B$$

$$I_B = \frac{Ic}{Hfe}$$

$$I_B = \frac{500.10^{-3}}{110}$$

$$I_B = 4{,}545mA$$

R_B untuk Transistor 9013:

$$5 - V_{C(SAT)} - I_B.R_B - V_{BE} = 0$$

$$5 - 0{,}16 - 4{,}545.10^{-3}.R_B - 0{,}7 = 0$$

$$4{,}84 - 4{,}545.R_B - 0{,}7 = 0$$

$$R_B = \frac{4,84 - 0,7}{4,545.10^{-3}}$$

$$R_B = 910,89\Omega$$

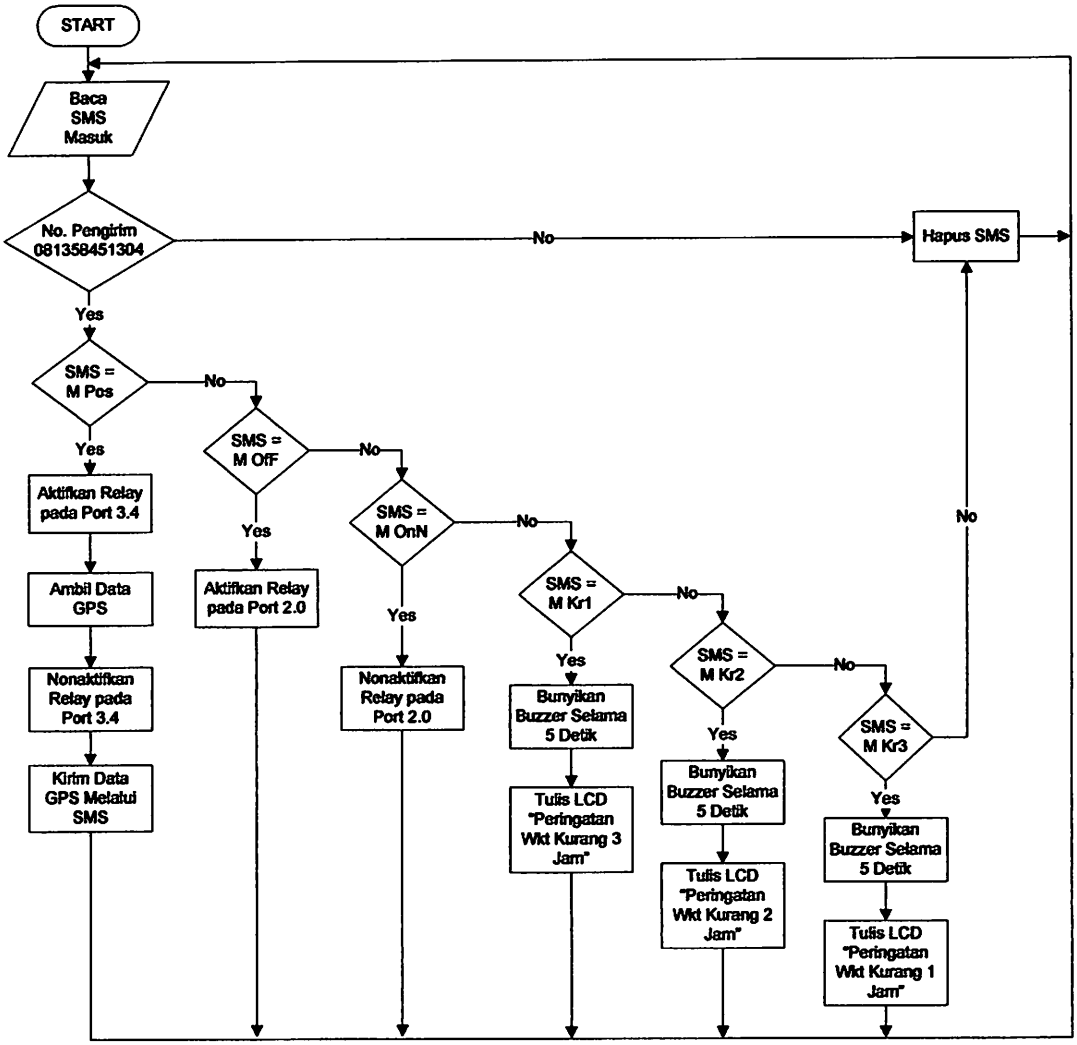
Karena nilai resistor 910,89 Ω tidak ada dipasaran maka digunakan resistor yang mendekati yaitu 1 KΩ.

3.3. Perancangan Perangkat Lunak *(Software)*

Untuk mendukung agar perangkat keras berfungsi sesuai dengan perencanaan, maka diperlukan perangkat lunak sebagai penunjangnya. Perangkat lunak ini sendiri maksudnya adalah suatu program yang kita buat yang nantinya akan ditanam kedalam mikrokontroller AT89S52. setelah mikrokontroller tersebut diprogram, maka akan diketahui apakah program yang telah kita buat bekerja sesuai dengan yang kita rencanakan ataukah masih memiliki kesalahan.

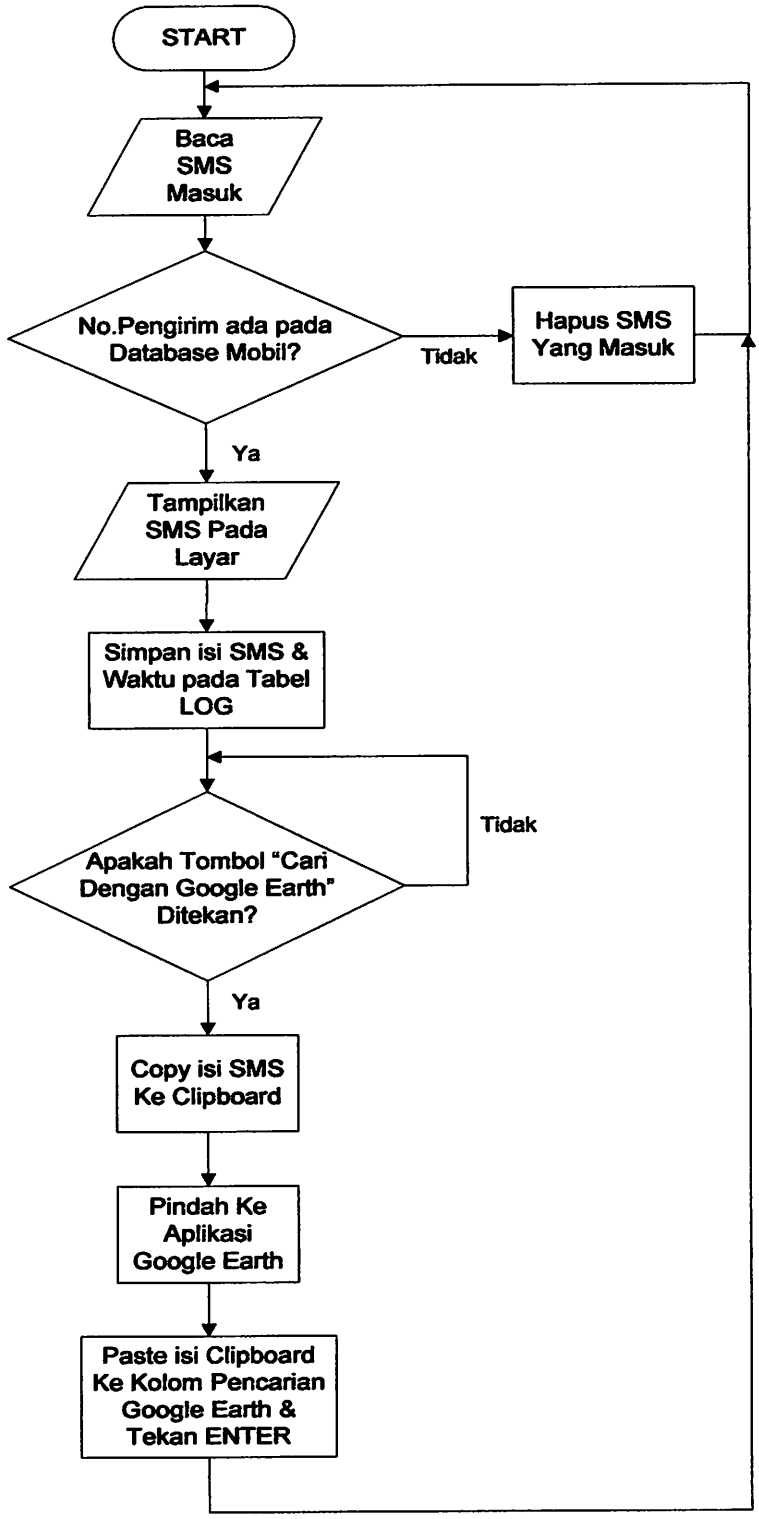
Sistem aplikasi mikrokontroller AT89S52 ini dapat mengatur dan mengendalikan keseluruhan sistem apabila ada urutan instruksi yang mendefinisikan secara jelas urutan tugas yang harus dikerjakan. Urutan instruksi ini sangat penting untuk didefinisikan, karena mikrokontroller dapat bekerja secara pasti sesuai dengan instruksi yang telah dibuat.

3.3.1.Flowchart Pada Client



Gambar 3.10. Flowchart Pada Client

3.3.2. Flowchart Untuk Program Delphi



Gambar 3.11. Flowchart Pada Pusat Untuk Pencarian Dengan Google Earth

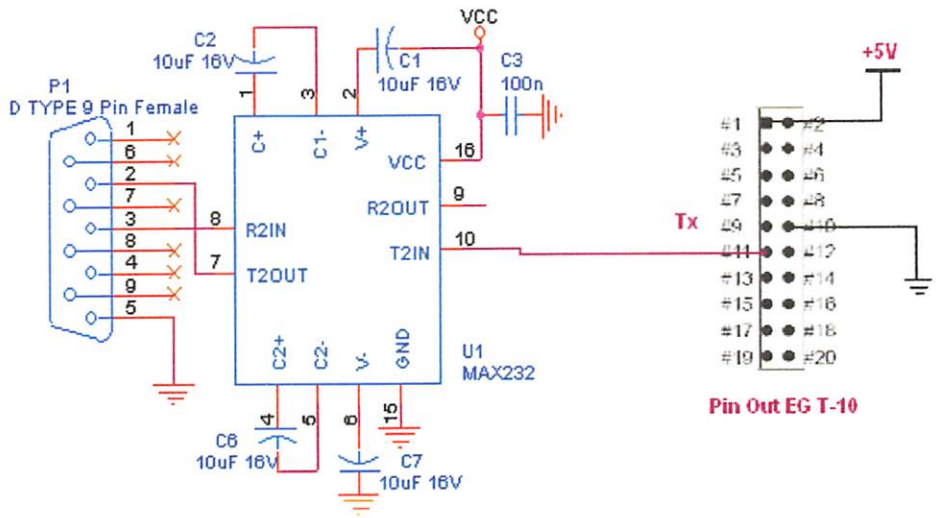
BAB IV

PENGUJIAN DAN ANALISA SISTEM

Sebagai cara untuk mengetahui unjuk kerja dari sistem yang dirancang dan dibuat, maka dilakukan pengujian alat. Pengujian ini meliputi pengujian modul GPS, pengujian handphone, pengujian driver, dan pengujian keseluruhan sistem.

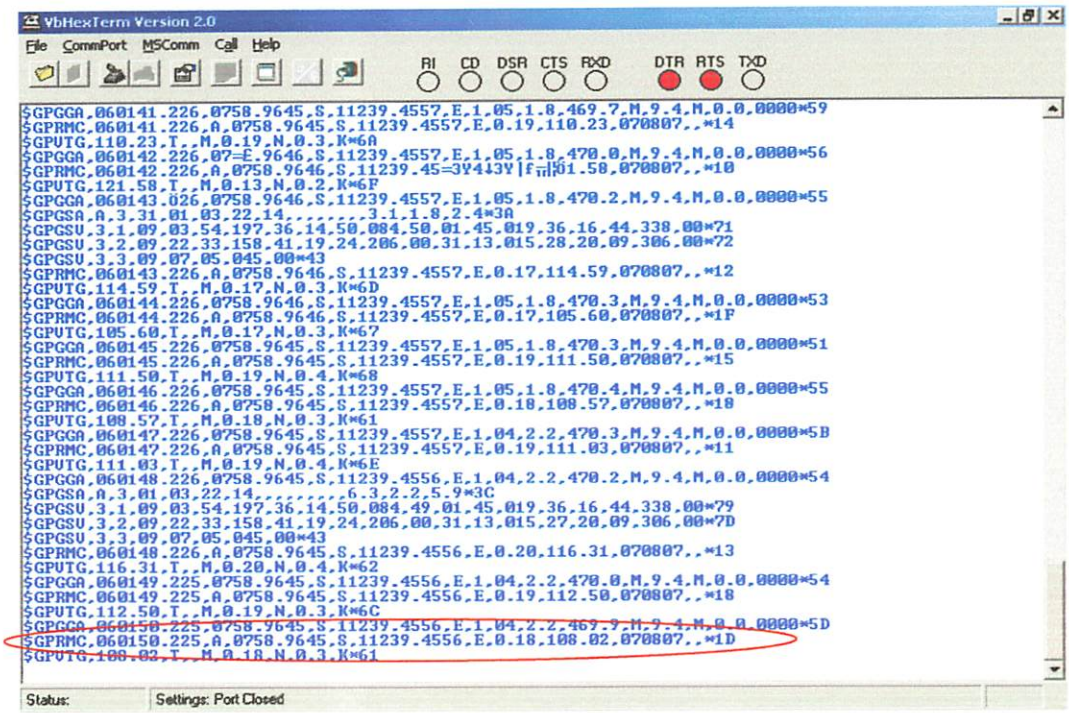
4.1. Pengujian Modul GPS

Pengujian modul GPS dilakukan dengan menggunakan PC dengan komunikasi RS232. Data yang diterima oleh PC ditampilkan dengan menggunakan program Serial Tester.



Gambar 4.1. Rangkaian Pengujian Modul GPS

Dari hasil pengujian didapat data yang dikirimkan oleh modul GPS sebagai berikut :



Gambar 4.2. Hasil Pengujian Modul GPS

$GPRMC,060150.225,A,0758.9645,S,11239.4556,E,0.18,108.02,070807,,*1D

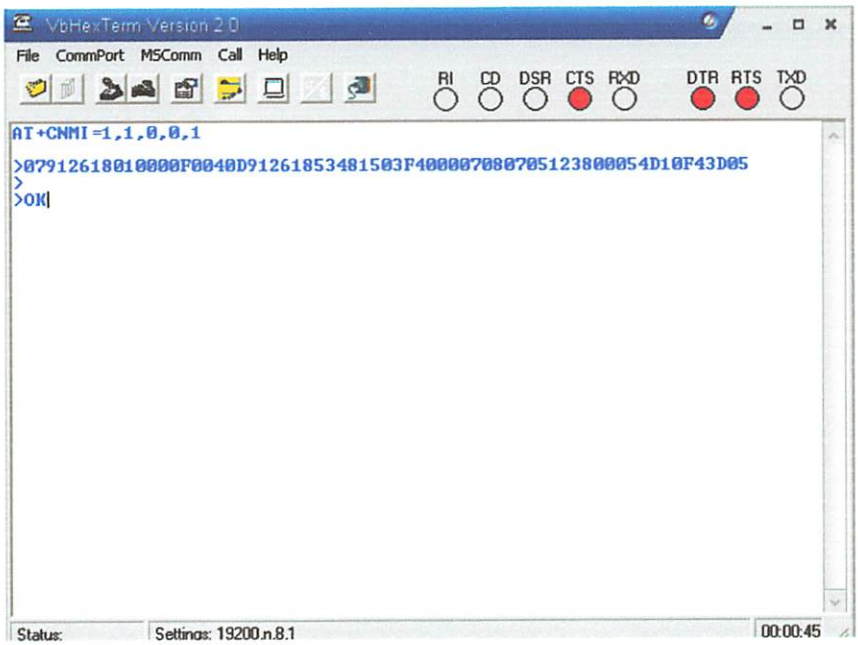
Tabel 4.1. Hasil Pengujian Modul GPS EG T-10

$GPRMC	RMC Protocol Header
060150.225	Diambil : 06:15:00:225 UTC
A	A, Data Valid ; V, Data Not Valid
0758.9645,S	Lattitude : 07°58'.9645 S
11239.4556,E	Longitude : 112°39'.4556 E
0.18	Kecepatan diatas tanah 0,18 Knot

108,02	Track made good (heading) 108.02°
070807	Tanggal dalam format ddmmyy (07 Agustus 2007)
*1D	Checksum

4.2. Pengujian Handphone

Pengujian handphone dilakukan dengan cara menghubungkan handphone dengan PC dengan menggunakan kabel data. Hasil pengujian ditampilkan dengan program Serial Tester.



Gambar 4.3. Hasil Pengujian Handphone

07912618010000F0040D91261853481503F40000708070512380005 4D10F43D0

5

Tabel 4.2. Hasil Pengujian Handphone

Heksa	Penjelasan
07	Panjang pasangan nomor SMS-Center termasuk tipe Kode
91	Tipe kode Nasional atau Internasional (91 = kode internasional)
2618010000F0	Nomor Service-Center Simpati (6281100000)
04	Tipe SMS (04= tipe untuk terima)
0D	Panjang nomor handphone pengirim
91	Tipe kode Nasional atau internasional
261853481503F4	Nomor Handphone pengirim (6281358451304)
00	Tipe bentuk SMS (00 = diterima sebagai SMS)
00	Tipe data coding
708070512380	Waktu SMS sampai di SMS center yaitu tanggal 07-08-07, pukul 15:32:08 WIB.
00	Jangka waktu SMS expired (00 = tidak memiliki batas)
06	Panjang pasangan dari isi SMS
4D10F43D05	Isi SMS (M PoS)

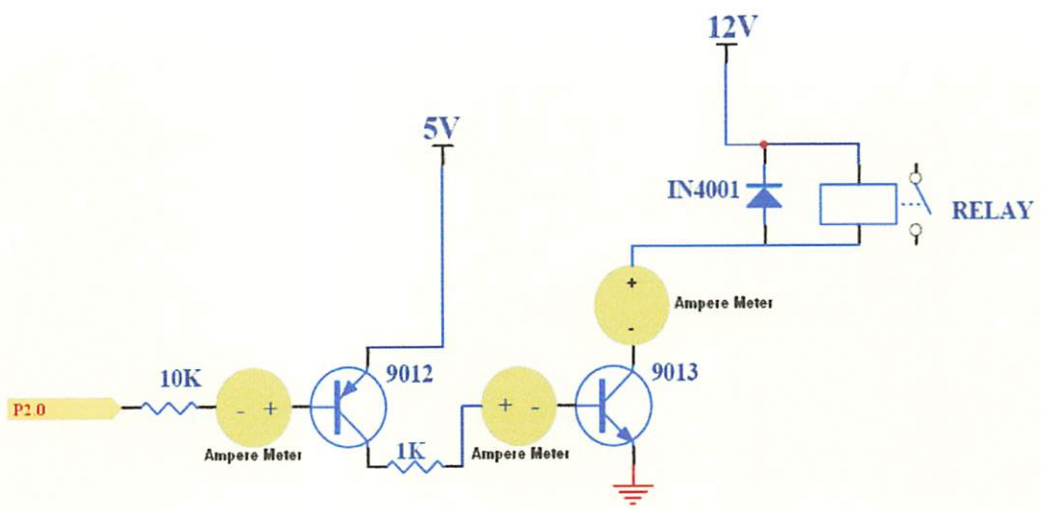
4.3. Pengujian Driver Relay

a. Peralatan yang digunakan

- Rangkaian pengujian (Driver relay)
- Sumber tegangan +5 dan +12 Volt
- Relay 12 Volt
- 3 buah Multimeter

b. Langkah-langkah Pengujian

1. Merangkai rangkaian driver seperti pada gambar dibawah ini:



Gambar 4 .4. Rangkaian Pengujian Driver Relay

2. Melakukan pengukuran dan pengamatan.

c. Analisa

Pada saat relay aktif, nilai yang didapatkan dari hasil pengukuran pada setiap multimeter di atas adalah sebagai berikut :

I_B pada 9012 = 0,44 mA.

I_B pada 9013 = 4,17 mA.

I_C pada 9013 = 68,3 mA.

Dan diketahui :

R_{Relay} = 150 ohm.

RB_{9013} = 1 Kohm

RB_{9012} = 10 Kohm

VCE_{9013} = 0,6 Volt

VCE_{9012} = 0,6 Volt

V_{OL} = 0,5 Volt

Maka, untuk menentukan nilai Arus Kolektor pada 9013 adalah:

$V_{CC} - I_C.R_C - V_{CE} = 0$

$$I_C = \frac{Vcc - V_{CE}}{R_C}$$

$$= \frac{12 - 0,6}{150}$$

$$= 76 \text{ mA}$$

dan Arus Basis pada 9013 adalah :

$$I_B = \frac{V_{EE} - V_{CE(sat)} - 0,7}{R_B}$$

$$= \frac{5 - 0,6 - 0,7}{1000}$$

$$= 3,7 \text{ mA}$$

Arus Basis pada 9012 adalah :

$$I_B = \frac{V_{EE} - 0,7 - V_{OL}}{R_B}$$

$$= \frac{5 - 0,7 - 0,5}{10000}$$

$$= 0,38 \text{ mA}$$

Untuk Menghitung Error digunakan Rumus :

$$\%Error = \frac{(Pengukuran - Perhitungan)}{Pengukuran} x100\%$$

Tabel 4.3. Hasil Pengukuran dan Perhitungan Driver Relay

Parameter	Perhitungan	Pengukuran	Error (%)
I_C 9013	76 mA	68,8 mA	10,46
I_B 9013	3,7 mA	4,17 mA	11,27
I_B 9012	0,38 mA	0,44 mA	13,63
Error rata-rata			11,78%

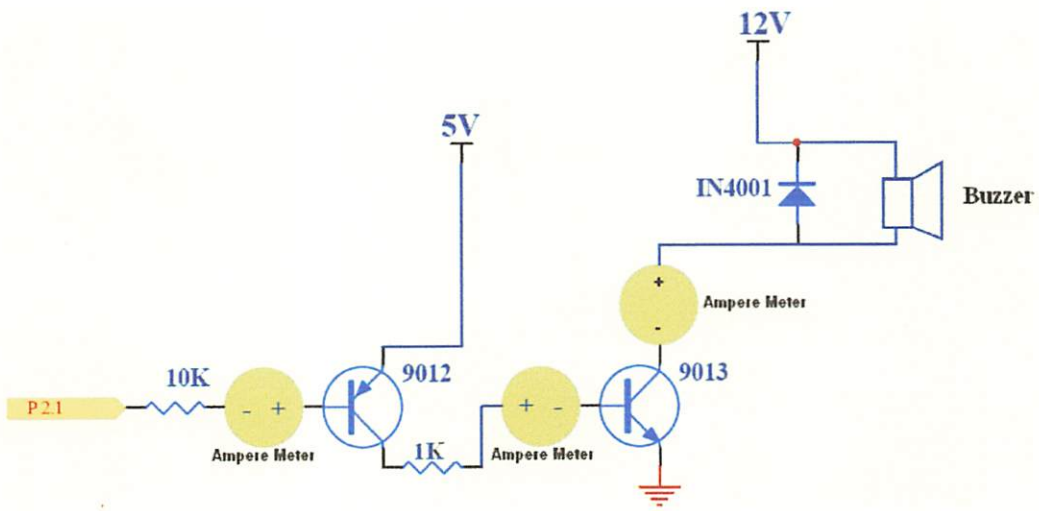
4.4. Pengujian Driver Buzzer

a. Peralatan yang digunakan

- Rangkaian pengujian (Driver Buzzer)
- Sumber tegangan +5 dan +12 Volt
- Buzzer 12 Volt
- 3 buah Multimeter

b. Langkah-langkah Pengujian

1. Merangkai rangkaian driver seperti pada gambar dibawah ini:



Gambar 4.5. Rangkaian Pengujian Driver Buzzer

2. Melakukan pengukuran dan pengamatan.

c. Analisa

Pada saat buzzer aktif, nilai yang didapatkan dari hasil pengukuran pada setiap multimeter di atas adalah sebagai berikut :

I_B pada 9012 = 0,44 mA.

I_B pada 9013 = 4,2 mA.

I_C pada 9013 = 18,7 mA.

Dan diketahui :

R_{Buzzer} = 600 ohm.

R_B9013 = 1 Kohm

R_B9012 = 10 Kohm

V_{CE}9013 = 0,6 Volt

V_{CE}9012 = 0,6 Volt

V_{OL} = 0,5 Volt

Maka, untuk menentukan nilai Arus Kolektor pada 9013 adalah:

$$V_{CC} - I_C.R_C - V_{CE} = 0$$

$$I_C = \frac{Vcc - V_{CE}}{R_C}$$

$$= \frac{12 - 0,6}{600}$$

$$= 19 \text{ mA}$$

dan Arus Basis pada 9013 adalah :

$$I_B = \frac{V_{EE} - V_{CE(sat)} - 0,7}{R_B}$$

$$= \frac{5-0,6-0,7}{1000}$$

$$= 3,7 \text{ mA}$$

Arus Basis pada 9012 adalah :

$$I_B = \frac{V_{EE}-0,7-V_{OL}}{R_B}$$

$$= \frac{5-0,7-0,5}{10000}$$

$$= 0,38 \text{ mA}$$

Tabel 4.4. Hasil Pengukuran dan Perhitungan Driver Buzzer

Parameter	Perhitungan	Pengukuran	% Error
I_C 9013	19 mA	18,7 mA	1,60
I_B 9013	3,7 mA	4,2 mA	11,90
I_B 9012	0,38mA	0,44 mA	13,63
Error Rata-rata			9,04%

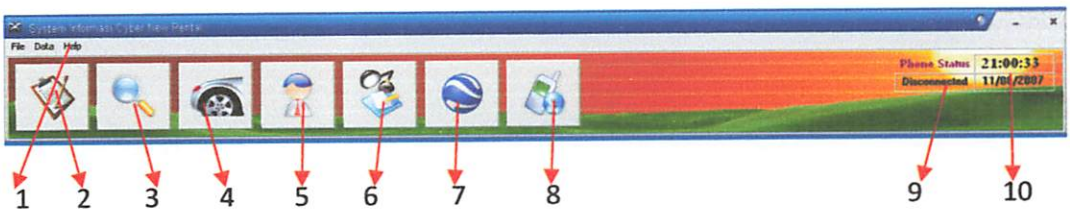
4.5. Pengujian Keseluruhan Sistem

Dalam pengujian keseluruhan sistem ini dilakukan dalam empat tahap, yaitu pengujian untuk permintaan lokasi (kondisi mobil diam dan bergerak), pengujian

pengiriman pesan masa persewaaan, pengujian menonaktifkan mesin mobil dan pengujian pengaktifan mesin mobil.

Unit kendaraan menggunakan kendaraan tertutup roda empat dengan *receiver* GPS yang diletakkan di dalam mobil dan antena GPS yang diletakkan di luar mobil dan Sistem Client menggunakan *power* dari *accu* mobil. Pusat menggunakan Laptop yang telah dilengkapi program Google Earth dan telah dihubungkan dengan *Handphone.*

4.5.1 Pengujian Penentuan Posisi



Gambar 4.6. Tampilan Awal Program

Keterangan :

1. Menu

   Digunakan untuk membuka form-form yang telah disediakan.

2. Tombol Transaksi

   Digunakan untuk membuka form transaksi yang digunakan untuk proses persewaan atau pengembalian mobil.

3. Tombol Tracking

   Digunakan untuk membuka form tracking yang berguna untuk proses pencarian posisi mobil atau pengiriman pesan maupun menon-aktifkan/mengaktifkan mesin mobil.

4. Tombol Database Mobil

   Digunakan untuk membuka form Database Mobil yang berisi informasi mobil (No. Plat, No. handphone, Merk, Warna)

5. Tombol Data Pelanggan

   Digunakan untuk membuka form Database Pelanggan yang berisi informasi pelanggan (Nama, No. Telepon, Alamat, No. Identitas, dll)

6. Tombol Data transaksi

   Digunakan untuk membuka form Data Transaksi yang berisi data-data transaksi persewaan mobil.

7. Tombol Run Google Earth

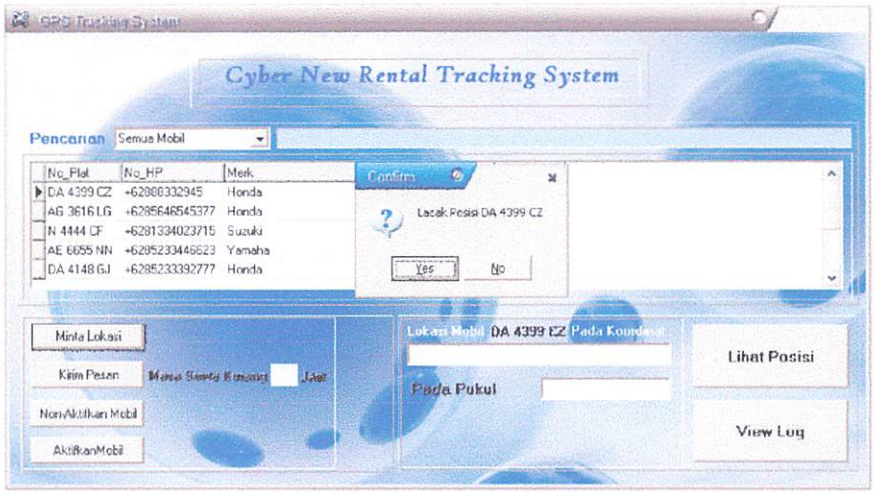
   Digunakan untuk menjalankan aplikasi Google Earth.

8. Tombol Connect/Disconnect Handphone

   Digunakan untuk menghubungkan atau memutuskan koneksi handphone dengan PC.

9. Label Status Handphone (Connect/Disconnected).

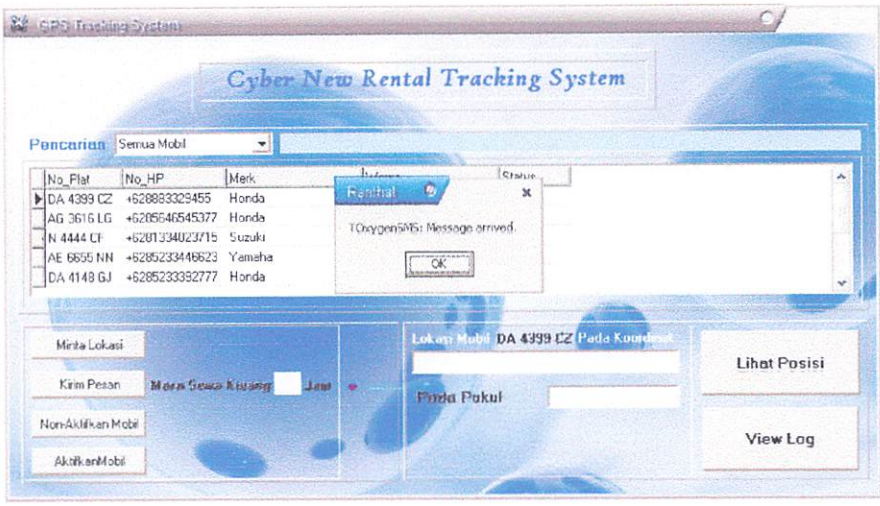
10. Informasi Jam dan Tanggal.



Gambar 4.7. Proses Permintaan Lokasi

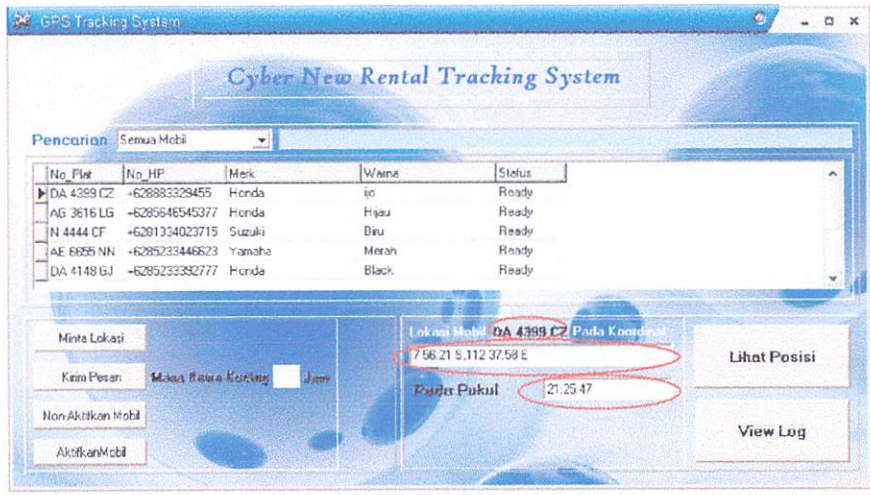
Untuk meminta lokasi dari client maka langkah-langkah yang harus dilakukan adalah sebagai berikut :

1. Buka form Tracking
2. Pilih tipe pencarian (Mobil Disewa/Semua Mobil)
   - Jika dipilih "Mobil Disewa" maka akan ditampilkan data mobil yang sedang disewa saja.
   - Jika dipilih "Semua Mobil" maka akan ditampilkan data semua mobil.
3. Setelah itu pilih mobil yang akan dilacak.
4. Tekan tombol "Minta Lokasi".
5. Jika Client yang dilacak merespon dan mengirimkan koordinatnya maka akan tampil seperti gambar 4.9. dibawah.



Gambar 4.8. Proses Menerima Pesan

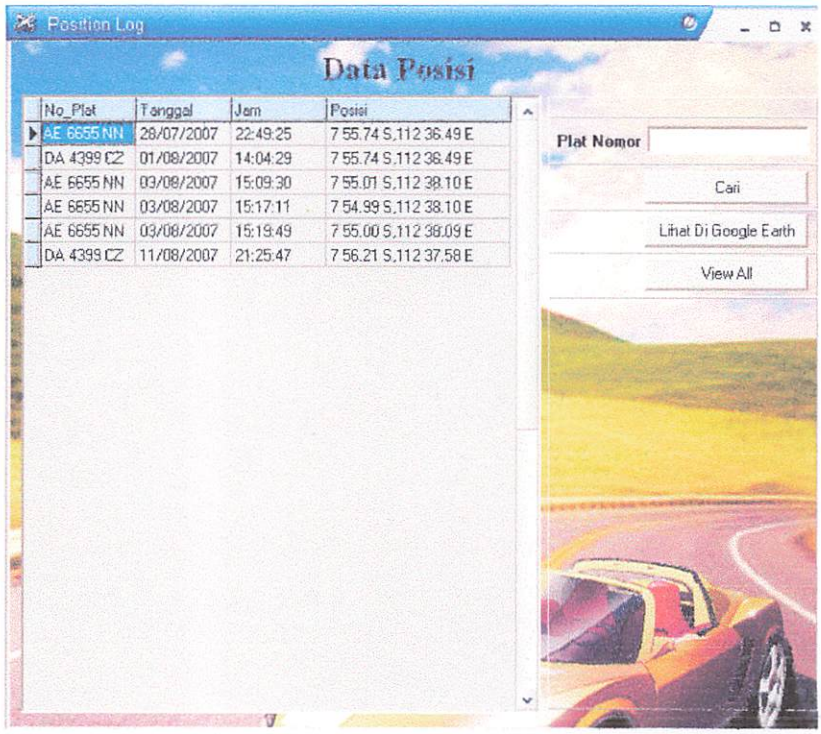
6. Click OK, maka data koordinat akan ditampilkan seperti pada gambar 4.10. dibawah ini.



Gambar 4.9. Data Koordinat Yang Diterima

7. Data yang ditampilkan adalah No. Plat Mobil, Koordinat, dan Waktu pada saat koordinat dikirimkan.
8. Untuk menampilkan pada program Google Earth click tombol "Lihat Posisi", pastikan program Google Earth telah dijalankan!!.

9. Data koordinat yang telah dikirimkan akan disimpan pada tabel Log, untuk membuka tabel log click tombol "View Log"



Gambar 4.10. Rekapitulasi Data Koordinat Yang Diterima

4.5.1.1. Pengujian Pertama

Pengujian dilakukan mulai dari Jl. Saxofone No.10 sampai dengan Dinoyo kemudian kembali ke tempat semula. Pengujian dilakukan mulai pukul 05.00 s/d 06.00. Titik-titik pengambilan koordinat dilakukan pada saat mobil dalam posisi berhenti. Titik-titik pengambilan koordinat yaitu :

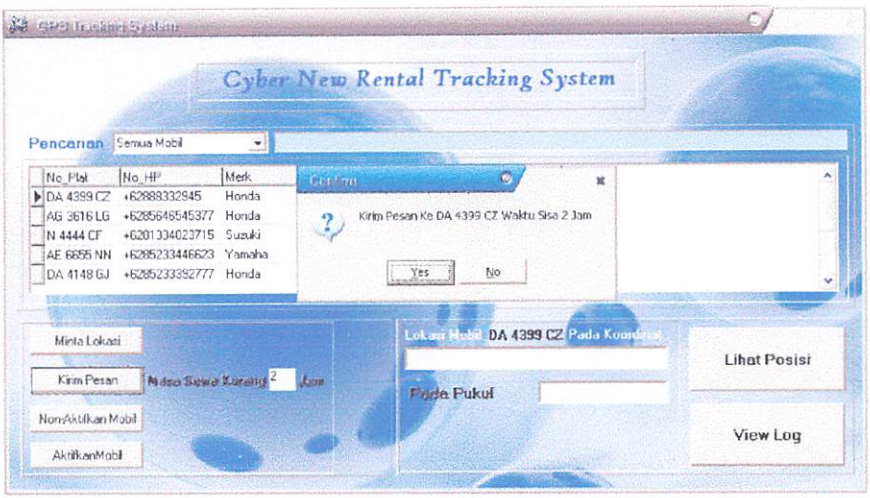
- Didepan Perumahan Bumi Palapa
- Didepan RRI
- Di Tugu MIG

Gambar 4.19. Selisih Pengujian Pertama dan Kedua

4.5.2. Pengujian Pengiriman Pesan Ke-Client

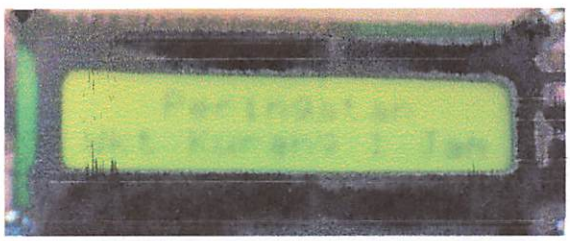
Untuk mengirim pesan ke client maka langkah-langkah yang harus dilakukan adalah sebagai berikut :

1. Buka form Tracking.
2. Pilih tipe pencarian (Mobil Disewa/Semua Mobil)
   - Jika dipilih "Mobil Disewa" maka akan ditampilkan data mobil yang sedang disewa saja.
   - Jika dipilih "Semua Mobil" maka akan ditampilkan data semua mobil.
3. Setelah itu pilih mobil yang akan dikirim pesan.
4. Isi kolom "Masa Sewa Kurang (1-3) Jam.
5. Tekan tombol "Kirim Pesan".

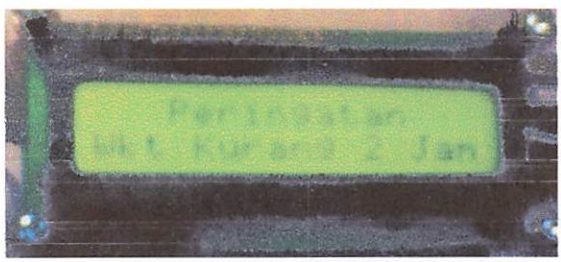


Gambar 4.20. Proses Pengiriman Pesan

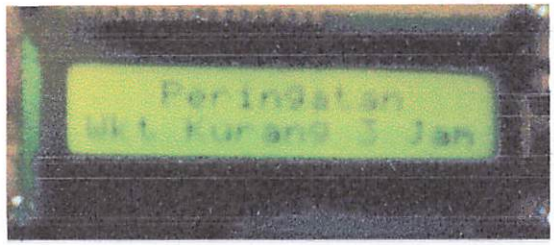
6. Pesan yang tampil pada LCD client seperti pada gambar 4.19 berikut



Gambar 4.21. Pesan yang tampil pada LCD Client Untuk Waktu Kurang 1 Jam



Gambar 4.22. Pesan yang tampil pada LCD Client Untuk Waktu Kurang 2 Jam

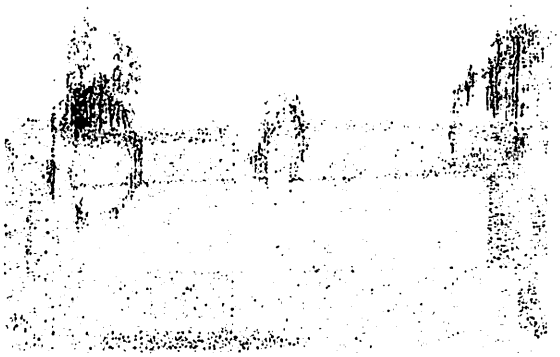


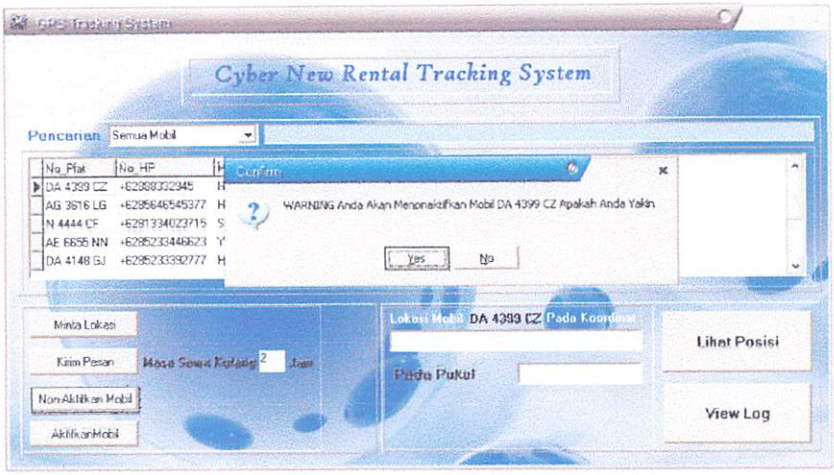
Gambar 4.23. Pesan yang tampil pada LCD Client Untuk Waktu Kurang 3 Jam

4.5.3. Pengujian Penonaktifan dan Pengaktifan mesin Mobil

Pada pengujian ini hanya dilakukan pengamatan pada relay pemutus aliran listrik yang terpasang pada mobil. Langkah-langkah yang harus dilakukan adalah sebagai berikut :

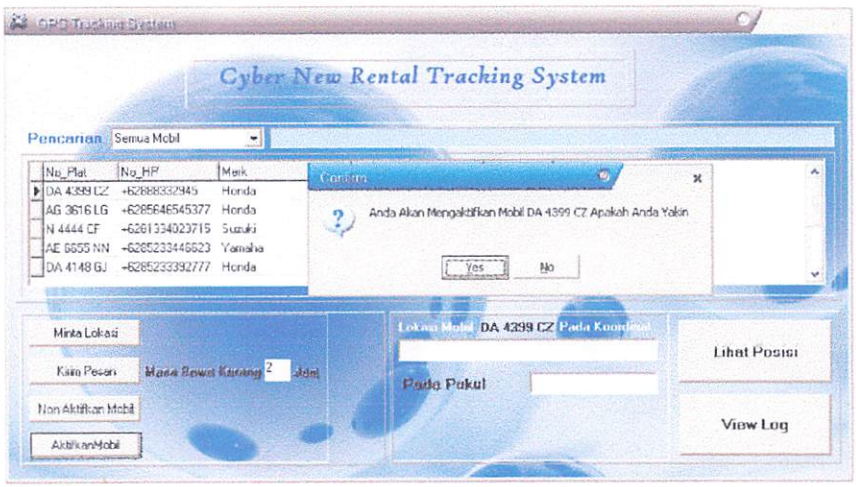
1. Buka form Tracking.
2. Pilih tipe pencarian (Mobil Disewa/Semua Mobil)
   - Jika dipilih "Mobil Disewa" maka akan ditampilkan data mobil yang sedang disewa saja.
   - Jika dipilih "Semua Mobil" maka akan ditampilakan data semua mobil.
3. Setelah itu pilih mobil yang akan dinonaktifkan (untuk menonaktifkan)
4. Tekan tombol "Non_Aktifkan Mobil".





Gambar 4.24. Proses Penonaktifan Mobil

5. Untuk mengaktifkan kembali tekan tombol "Aktifkan Mobil".



Gambar 4.25. Proses Pengaktifan Mobil

Tabel 4.8. Hasil Pengamatan Relay

No.	Tombol Yang Ditekan	SMS Yang Dikirim	Keadaan Relay
1.	Non-Aktifkan Mobil	M<spasi>OfF	Aktif
2.	Aktifkan Mobil	M<spasi>OnN	Tidak Aktif

3.	Non-Aktifkan Mobil	M<spasi>OfF	Aktif
4.	Aktifkan Mobil	M<spasi>OnN	Tidak Aktif
5.	Non-Aktifkan Mobil	M<spasi>OfF	Aktif
6.	Aktifkan Mobil	M<spasi>OnN	Tidak Aktif
7.	Non-Aktifkan Mobil	M<spasi>OfF	Aktif
8.	Aktifkan Mobil	M<spasi>OnN	Tidak Aktif
9.	Non-Aktifkan Mobil	M<spasi>OfF	Aktif
10.	Aktifkan Mobil	M<spasi>OnN	Tidak Aktif

Dari hasil pengujian pada tabel 4.8. dapat dilihat bahwa dalam 10 kali percobaan posisi relay sesuai dengan yang diharapkan.

BAB V

PENUTUP

5.1. Kesimpulan

1. Saat pelacakan mobil bergerak posisi yang diterima tidak sesuai, selisih posisi bisa mencapai 10-50m tergantung dari kecepatan mobil dan waktu pengiriman SMS.

2. Pada pengujian penonaktifan mesin mobil persentase keberhasilan relay untuk memutus aliran listrik mobil adalah 100 persen. Demikian pula pada waktu pengaktifan mesin mobil.

3. Rata-rata selisih waktu antara pengiriman dan penerimaan SMS pada saat pengujian adalah 11,94 detik.

5.2. Saran-saran

1. Untuk mengetahui keadaan mobil setelah dinonaktifkan sebaiknya ditambahkan sensor gerak atau sensor lainnya untuk mendeteksi apakah mesin benar-benar dalam kondisi mati.

2. Pada saat client mengirimkan informasi koordinat sebaiknya ditambahkan informasi apakah mobil dalam kondisi bergerak atau tidak.

DAFTAR PUSTAKA

[1] Abidin,Hasanudin. "*Penetuan Posisi Dengan GPS*".Jakarta: P.T. Elex Media Komputindo, 2000

[2] Atmel Corporation, *MCS-51 Datasheet.* <http://www.atmel.com>

[3] Elink-Tech Co.,Ltd, "*EG-T10 GPS Module Receiver specsheet*" <http://www.elink-tech.com.tw/products/manual/>

[4] Khang, Bustam, "*Trik Pemrograman Aplikasi Berbasis SMS*". Jakarta: P.T. Elex Media Komputindo, 2002.

[5] ITN Malang, "*Buku Panduan Belajar Borland Delphi*". Malang: Laboratorium Elektronika Digital, 2006

[6] Nizam, Khairul. "*Global Positioning Sistem (GPS)*." Malaysia: Diktat Kuliah Ukur Kejuruteraan, 2006

[7] Putra, Agfianto Eko. "*Belajar Mikrokontroller AT89S51 (Teori Dan Aplikasi)*". Yogyakarta : Gava Media,2002

[8] Seiko Instruments Inc, *Liquid Crystal Display Modul M1632 User Manual.* Japan: Jan. 1987

[9] Siemens, *AT Command Set Reference Manual.* Agustus 1999.

[10] Marcus Teddy Zakaria dan Josef Widiadhi, "*Aplikasi SMS untuk Berbagai Keperluan*." Bandung: Informatika 2006

[11] http://vancouver-webpages.com/peter/nmeafaq.txt

[12] Wasito S, 1990, *Vandemekum Elektronika, Edisi Kedua*, Jakarta: PT. Gramedia Pustaka Utama

[13] Training Center Department "*Training Manual.*" Jakarta: P.T. Krama Yudha Tiga Berlian Motor 2002


LAMP IRAN

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

LEMBAR PERBAIKAN SKRIPSI

Nama : Muhammad Indra Wijaya
Nim : 02.17.066
Jurusan : Teknik Elektro S1
Konsentrasi : Teknik Elektronika
Judul : "ALAT PELACAK POSISI DENGAN GPS DAN PENONAKTIFAN MESIN MOBIL DENGAN KENDALI HANDPHONE PADA PERUSAHAAN PERSEWAAN MOBIL"
Hari/Tgl Skripsi : Senin, 3 September 2007

No.	Materi Perbaikan	Paraf
1.	Supaya dicantumkan mekanisme terperinci tentang proses pengiriman data dari GPS →Mikrokontroler→Handphone	
2.	Supaya dirincikan uraian tentang mekanisme komunikasi data antara handphone Nokia dengan Laptop.	

Diperiksa / Disetujui
Penguji

Ir. Widodo Pudji M., MT
NIP.Y. 102870017

Mengetahui

Dosen Pembimbing I

Joseph Dedy Irawan, ST, MT
NIP. 132315178

Dosen Pembimbing II

M. Askar, ST, MT
NIP.P. 1030500408

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No. 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Muhammad Indra Wijaya
NIM : 02.17.066
Masa Bimbingan : 17 April 2007 s/d 17 Oktober 2007
Judul Skripsi : Alat Pelacak Posisi Dengan GPS Dan Penonaktifan Mesin Mobil Dengan Kendali Handphone Pada Perusahaan Persewaan Mobil

No.	Tanggal	Uraian	Paraf Pembimbing
1	02/08 07	Bab I, II (revisi)	
2	03/08 07	Bab III (revisi)	
3	06/08 07	Perancangan system	
4	10/08 07	Bab IV (revisi)	
5	13/08 07	pengujian google earth	
6	14/08 07	pengujian Non aktif mesin	
7	27/08 07	pengujian database	
8	30/08 07	Makalah Seminar	
9	31/08 07	Bab V (revisi)	
10	1/08 07	Acc kompre	

Malang,
Dosen Pembimbing II

M. Ashar, ST, MT

Form S-4a

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No. 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Muhammad Indra Wijaya
NIM : 02.17.066
Masa Bimbingan : 17 April 2007 s/d 17 Oktober 2007
Judul Skripsi : Alat Pelacak Posisi Dengan GPS Dan Penonaktifan Mesin Mobil Dengan Kendali Handphone Pada Perusahaan Persewaan Mobil

No.	Tanggal	Uraian	Paraf Pembimbing
1	02/08 07	BAB I - III Revisi flowChart.	
2	03/08 07	BAB IV Pengujian Gps dan Hp.	
3	06/08 07	BAB IV Pengujian Keseluruhan.	
4	10/08 07	BAB V. Saran - saran ditambahkan	
5	13/08 07	BAB I - V Revisi	
6	14/08 07	Kesimpulan Ditambahkan	
7	27/08 07	lengkapi Diagram/Skematik	
8	1/9 2007	ACC KOMPRE	
9			
10			

Malang,
Dosen Pembimbing I

Joseph Dedy Irawan, ST, MT
NIP. 132.315.178

Form S-4a

256

atures

- ›mpatible with MCS-51® Products
- ‹ Bytes of In-System Programmable (ISP) Flash Memory
  - – Endurance: 1000 Write/Erase Cycles
-)V to 5.5V Operating Range
- ılly Static Operation: 0 Hz to 33 MHz
- ıree-level Program Memory Lock
- ;6 x 8-bit Internal RAM
- ! Programmable I/O Lines
- ıree 16-bit Timer/Counters
- ght Interrupt Sources
- ıll Duplex UART Serial Channel
- ›w-power Idle and Power-down Modes
- terrupt Recovery from Power-down Mode
- atchdog Timer
- ıal Data Pointer
- ›wer-off Flag

scription

AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K ›s of in-system programmable Flash memory. The device is manufactured using ;el's high-density nonvolatile memory technology and is compatible with the indus-;tandard 80C51 instruction set and pinout. The on-chip Flash allows the program nory to be reprogrammed in-system or by a conventional nonvolatile memory pro-nmer. By combining a versatile 8-bit CPU with in-system programmable Flash on onolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a ıly-flexible and cost-effective solution to many embedded control applications.

AT89S52 provides the following standard features: 8K bytes of Flash, 256 bytes ;AM, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, clock circuitry. In addition, the AT89S52 is designed with static logic for operation /n to zero frequency and supports two software selectable power saving modes. Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and rrupt system to continue functioning. The Power-down mode saves the RAM con-;s but freezes the oscillator, disabling all other chip functions until the next interrupt ıardware reset.

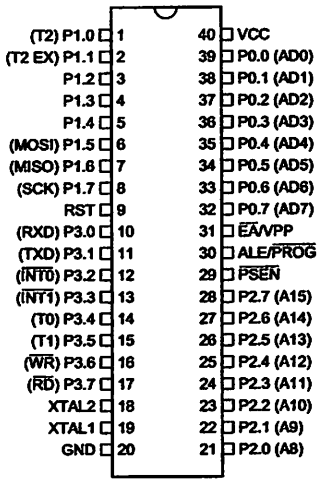
8-bit Microcontroller with 8K Bytes In-System Programmable Flash

AT89S52

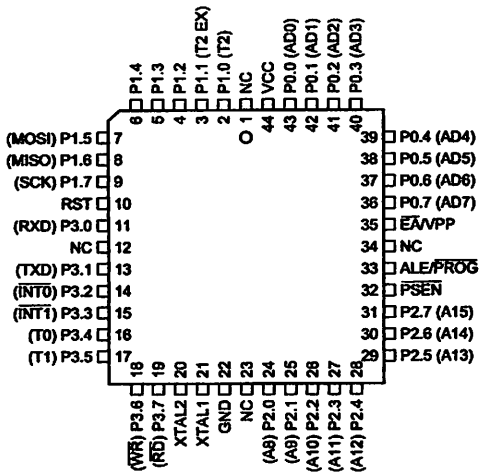
Rev. 1919A-07/01

n Configurations

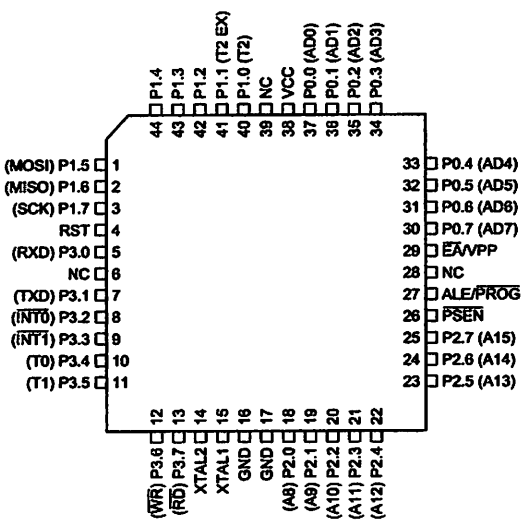
PDIP



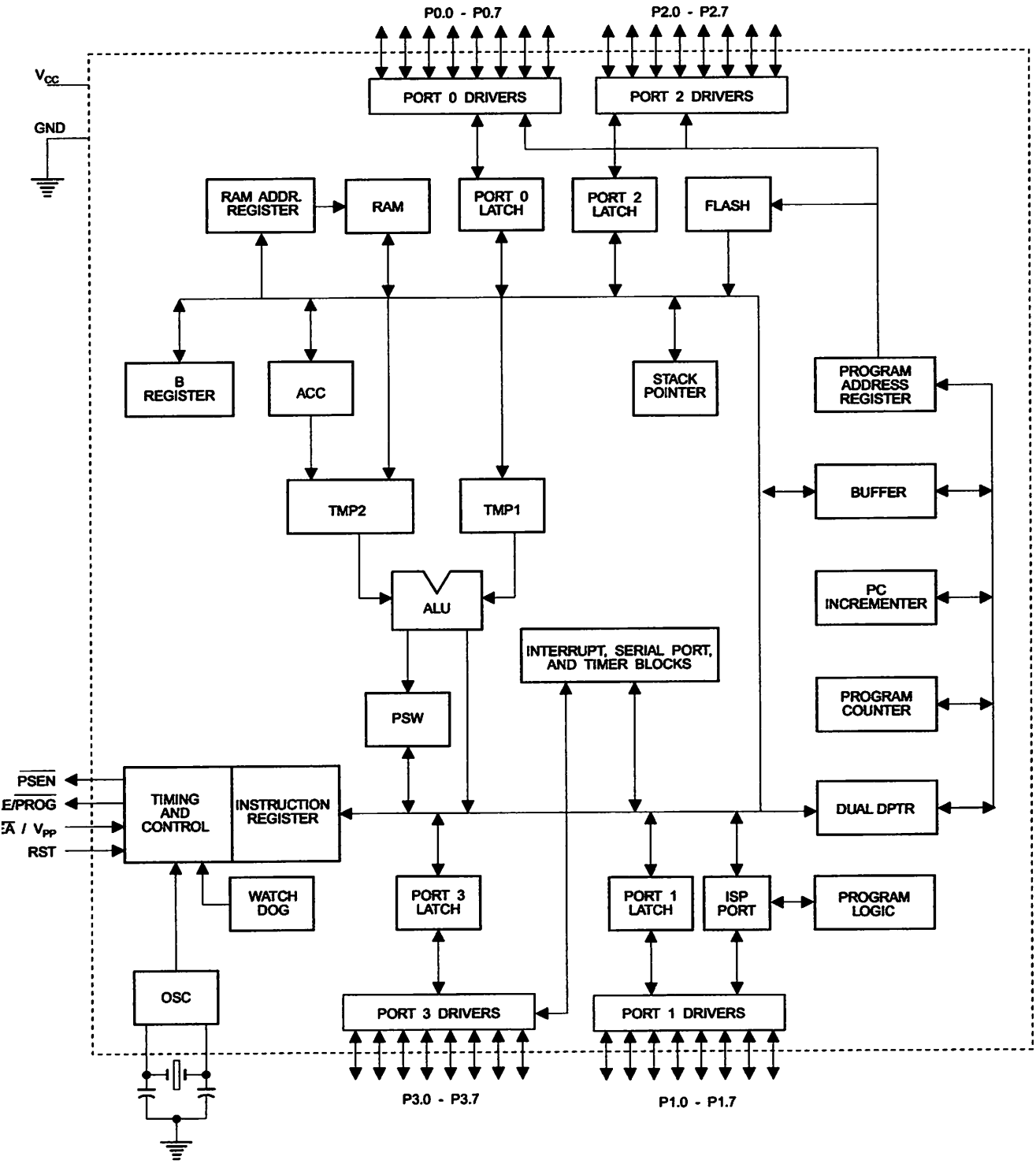
PLCC



TQFP



ɔck Diagram


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

ı Description

ɔ

ıply voltage.

D

ıund.

t 0

t 0 is an 8-bit open drain bidirectional I/O port. As an ɔut port, each pin can sink eight TTL inputs. When 1s written to port 0 pins, the pins can be used as high-edance inputs.

t 0 can also be configured to be the multiplexed low-er address/data bus during accesses to external gram and data memory. In this mode, P0 has internal ups.

t 0 also receives the code bytes during Flash program-g and outputs the code bytes during program verifica-ı. External pullups are required during program fication.

t 1

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

ıddition, P1.0 and P1.1 can be configured to be the ər/counter 2 external count input (P1.0/T2) and the ər/counter 2 trigger input (P1.1/T2EX), respectively, as wn in the following table.

t 1 also receives the low-order address bytes during ɜh programming and verification.

ırt Pin	Alternate Functions
.0	T2 (external count input to Timer/Counter 2), clock-out
.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
.5	MOSI (used for In-System Programming)
.6	MISO (used for In-System Programming)
.7	SCK (used for In-System Programming)

t 2

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

t 2 emits the high-order address byte during fetches n 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 pullups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

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

Port 3

Port 3 is an 8-bit bidirectional I/O port with internal pullups. The Port 3 output buffers can sink/source four TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the internal pullups and can be used as inputs. As inputs, Port 3 pins that are externally being pulled low will source current (I_{IL}) because of the pullups.

Port 3 also serves the functions of various special features of the AT89S52, as shown in the following table.

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

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

RST

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 96 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

ALE/$\overline{PROG}$

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

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

If desired, ALE operation can be disabled by setting bit 0 of SFR location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is

akly pulled high. Setting the ALE-disable bit has no ct if the microcontroller is in external execution mode.

EN

gram Store Enable ($\overline{PSEN}$) is the read strobe to exter- program memory.

en the AT89S52 is executing code from external pro- m memory, $\overline{PSEN}$ is activated twice each machine le, except that two $\overline{PSEN}$ activations are skipped during h access to external data memory.

/VPP

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

XTAL1

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

XTAL2

Output from the inverting oscillator amplifier.

ble 1. AT89S52 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000							0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000							0BFH
0B0H	P3 11111111							0B7H
0A8H	IE 0X000000							0AFH
0A0H	P2 11111111		AUXR1 XXXXXXX0				WDTRST XXXXXXXX	0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111							97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0	8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	PCON 0XXX0000	87H

ıecial Function Registers

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

:e that not all of the addresses are occupied, and unoc- ıied addresses may not be implemented on the chip. ad accesses to these addresses will in general return dom data, and write accesses will have an indetermi- e effect.

ɜr software should not write 1s to these unlisted loca- ıs, 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 3) 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.

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

ıle 2. T2CON – Timer/Counter 2 Control Register

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

Bit Addressable

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

ɣmbol	Function
⁼2	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.
XF2	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).
CLK	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 overflow to be used for the receive clock.
⁻LK	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.
XEN2	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.
R2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
/T2	Timer or counter select for Timer 2. C/$\overline{T2}$ = 0 for timer function. C/$\overline{T2}$ = 1 for external event counter (falling edge triggered).
P/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.

le 3a. AUXR: Auxiliary Register

JXR	Address = 8EH						Reset Value = XXX00XX0B	
	Not Bit Addressable							
		–	–	–	WDIDLE	DISRTO	–	–	DISALE
	Bit	7	6	5	4	3	2	1	0
	Reserved for future expansion							
SALE	Disable/Enable ALE							
	DISALE	Operating Mode						
	0	ALE is emitted at a constant rate of 1/6 the oscillator frequency						
	1	ALE is active only during a MOVX or MOVC instruction						
SRTO	Disable/Enable Reset out							
	DISRTO							
	0	Reset pin is driven High after WDT times out						
	1	Reset pin is input only						
DIDLE	Disable/Enable WDT in IDLE mode							
	WDIDLE							
	0	WDT continues to count in IDLE mode						
	1	WDT halts counting in IDLE mode						

al Data Pointer Registers: To facilitate accessing both ernal and external data memory, two banks of 16-bit a Pointer Registers are provided: DP0 at SFR address ations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 SFR AUXR1 selects DP0 and DPS = 1 selects DP1. e user should always initialize the DPS bit to the appropriate value before accessing the respective Data Pointer Register.

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

ble 3b. AUXR1: Auxiliary Register 1

JXR1	Address = A2H						Reset Value = XXXXXXX0B	
	Not Bit Addressable							
		–	–	–	–	–	–	–	DPS
	Bit	7	6	5	4	3	2	1	0
	Reserved for future expansion							
PS	Data Pointer Register Select							
	DPS							
	0	Selects DPTR Registers DP0L, DP0H						
	1	Selects DPTR Registers DP1L, DP1H						

ɜmory Organization

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

ɔgram Memory

ıe $\overline{EA}$ pin is connected to GND, all program fetches are
ıcted to external memory.

the AT89S52, if $\overline{EA}$ is connected to V_{CC}, program
ıes to addresses 0000H through 1FFFH are directed to
ırnal memory and fetches to addresses 2000H through
FFH are to external memory.

ta Memory

ɜ AT89S52 implements 256 bytes of on-chip RAM. The
ıer 128 bytes occupy a parallel address space to the
ɜcial Function Registers. This means that the upper 128
ɜs have the same addresses as the SFR space but are
ʻsically 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 which use direct addressing access of the 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.

ıtchdog Timer
ne-time Enabled with Reset-out)

WDT is intended as a recovery method in situations ere the CPU may be subjected to software upsets. The T consists of a 13-bit counter and the Watchdog Timer et (WDTRST) SFR. The WDT is defaulted to disable n exiting reset. To enable the WDT, a user must write EH and 0E1H in sequence to the WDTRST register R location 0A6H). When the WDT is enabled, it will ement every machine cycle while the oscillator is run-. The WDT timeout period is dependent on the external ck frequency. There is no way to disable the WDT ept through reset (either hardware reset or WDT over- reset). When WDT overflows, it will drive an output SET HIGH pulse at the RST pin.

ing the WDT

enable the WDT, a user must write 01EH and 0E1H in uence to the WDTRST register (SFR location 0A6H). en the WDT is enabled, the user needs to service it by ing 01EH and 0E1H to WDTRST to avoid a WDT over-. The 13-bit counter overflows when it reaches 8191 FFH), and this will reset the device. When the WDT is bled, it will increment every machine cycle while the illator is running. This means the user must reset the T at least every 8191 machine cycles. To reset the T the user must write 01EH and 0E1H to WDTRST. TRST is a write-only register. The WDT counter cannot ead or written. When WDT overflows, it will generate an put RESET pulse at the RST pin. The RESET pulse ation is 96xTOSC, where TOSC=1/FOSC. To make the t use of the WDT, it should be serviced in those sec- s of code that will periodically be executed within the required to prevent a WDT reset.

DT During Power-down and Idle

Power-down mode the oscillator stops, which means the T also stops. While in Power-down mode, the user s not need to service the WDT. There are two methods xiting Power-down mode: by a hardware reset or via a l-activated external interrupt which is enabled prior to ering Power-down mode. When Power-down is exited hardware reset, servicing the WDT should occur as it mally does whenever the AT89S52 is reset. Exiting ver-down with an interrupt is significantly different. The rrupt is held low long enough for the oscillator to stabi- . When the interrupt is brought high, the interrupt is viced. To prevent the WDT from resetting the device le the interrupt pin is held low, the WDT is not started il the interrupt is pulled high. It is suggested that the T be reset during the interrupt service for the interrupt d to exit Power-down mode.

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

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S52 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode.

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

UART

The UART in the AT89S52 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 '8051-Architecture Flash Microcontroller', then 'Product Overview'.

Timer 0 and 1

Timer 0 and Timer 1 in the AT89S52 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 '8051-Architecture Flash Microcontroller', then 'Product Overview'.

Timer 2

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/$\overline{T2}$ in the SFR T2CON (shown in Table 2). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 3. 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.

Table 3. Timer 2 Operating Modes

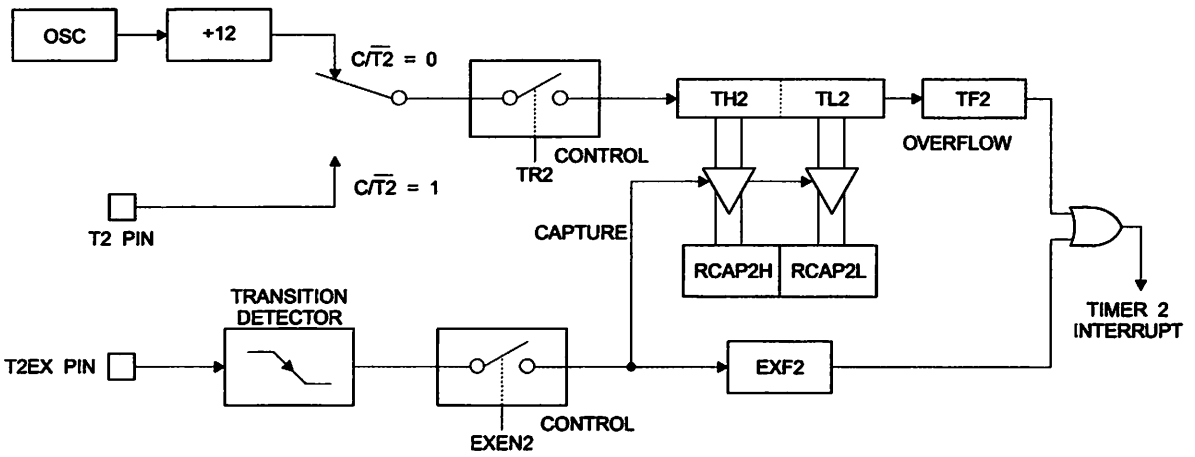
RCLK +TCLK	CP/$\overline{RL2}$	TR2	MODE
0	0	1	16-bit Auto-reload
0	1	1	16-bit Capture
1	X	1	Baud Rate Generator
X	X	0	(Off)

he Counter function, the register is incremented in ɔonse to a 1-to-0 transition at its corresponding external ɹt pin, T2. In this function, the external input is sampled ing S5P2 of every machine cycle. When the samples ɔw a high in one cycle and a low in the next cycle, the ɹnt is incremented. The new count value appears in the ɟister during S3P1 of the cycle following the one in which transition was detected. Since two machine cycles (24 ːillator periods) are required to recognize a 1-to-0 transi-ɳ, the maximum count rate is 1/24 of the oscillator fre-ɘncy. To ensure that a given level is sampled at least ːe before it changes, the level should be held for at least ɜ full machine cycle.

ɹpture Mode

the capture mode, two options are selected by bit EN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer counter which upon overflow sets bit TF2 in T2CON.

ɹure 5. Timer in Capture Mode



ure 6 shows Timer 2 automatically counting up when EN=0. In this mode, two options are selected by bit EN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to FFFH and then sets the TF2 bit upon overflow. The ɘrflow also causes the timer registers to be reloaded with 16-bit value in RCAP2H and RCAP2L. The values in ɳer in Capture ModeRCAP2H and RCAP2L are preset software. If EXEN2 = 1, a 16-bit reload can be triggered ɳer by an overflow or by a 1-to-0 transition at external ut T2EX. This transition also sets the EXF2 bit. Both the 2 and EXF2 bits can generate an interrupt if enabled.

ːting the DCEN bit enables Timer 2 to count up or down, shown in Figure 6. 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.

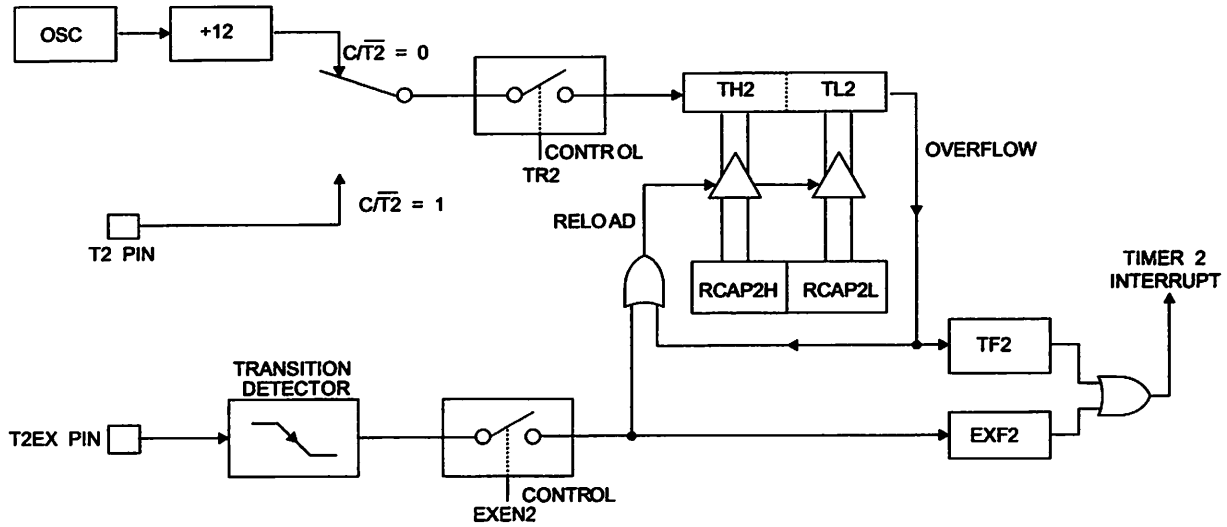
The EXF2 bit toggles whenever Timer 2 overflows or underflows and can be used as a 17th bit of resolution. In this operating mode, EXF2 does not flag an interrupt.

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 5.

Auto-reload (Up or Down Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 4). 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.

ure 6. Timer 2 Auto Reload Mode (DCEN = 0)



ble 4. T2MOD – Timer 2 Mode Control Register

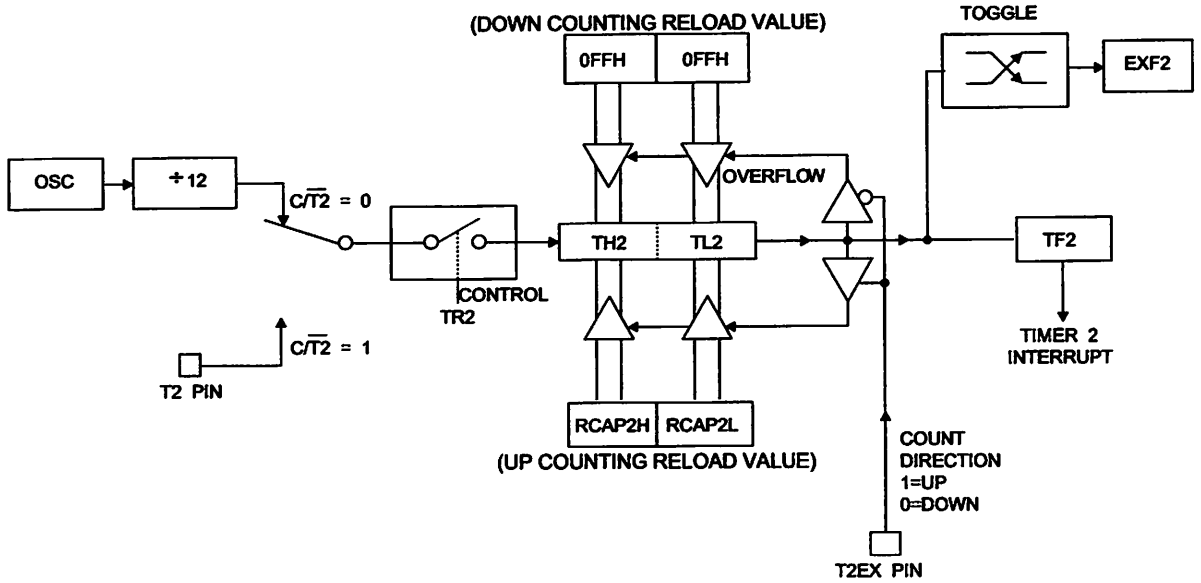
T2MOD Address = 0C9H — Reset Value = XXXX XX00B

Not Bit Addressable

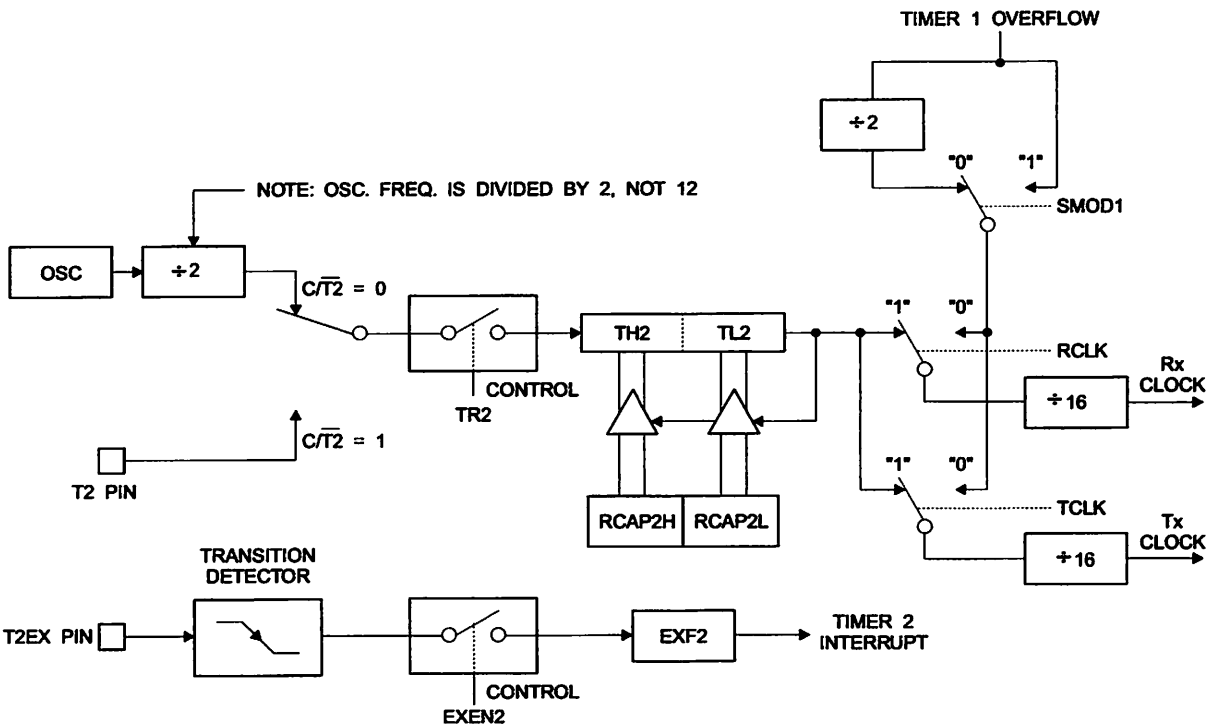
Bit	7	6	5	4	3	2	1	0
	–	–	–	–	–	–	T2OE	DCEN

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

ure 7. Timer 2 Auto Reload Mode (DCEN = 1)



ure 8. Timer 2 in Baud Rate Generator Mode



ıd Rate Generator

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

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

baud rates in Modes 1 and 3 are determined by Timer ›verflow rate according to the following equation.

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

Timer can be configured for either timer or counter ration. In most applications, it is configured for timer ration ($C/\overline{T2}$ = 0). The timer operation is different for er 2 when it is used as a baud rate generator. Normally, ı timer, it increments every machine cycle (at 1/12 the llator 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.

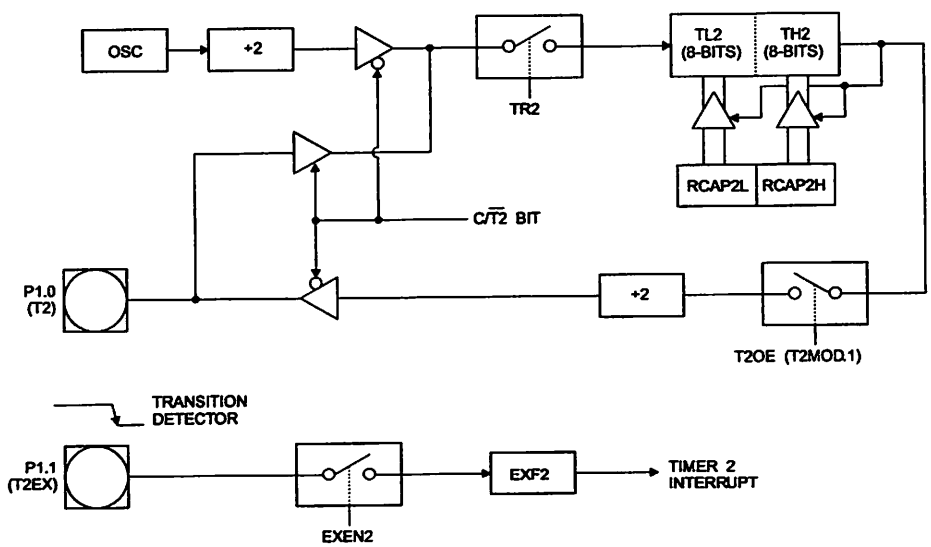
$$\frac{\text{Modes 1 and 3}}{\text{Baud Rate}} = \frac{\text{Oscillator Frequency}}{32 \times [65536\text{-RCAP2H,RCAP2L})]}$$

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 8. 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.

ıre 9. Timer 2 in Clock-Out Mode



ɔgrammable Clock Out

)% duty cycle clock can be programmed to come out on), as shown in Figure 9. This pin, besides being a regu- I/O pin, has two alternate functions. It can be pro- nmed to input the external clock for Timer/Counter 2 or ıutput a 50% duty cycle clock ranging from 61 Hz to 4 z at a 16 MHz operating frequency.

ɔonfigure the Timer/Counter 2 as a clock generator, bit 2 (T2CON.1) must be cleared and bit T2OE (T2MOD.1) ɜt be set. Bit TR2 (T2CON.2) starts and stops the timer.

ɨ clock-out frequency depends on the oscillator fre- ıncy and the reload value of Timer 2 capture registers ;AP2H, RCAP2L), as shown in the following equation.

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

ıe clock-out mode, Timer 2 roll-overs will not generate interrupt. This behavior is similar to when Timer 2 is d as a baud-rate generator. It is possible to use Timer 2 ə baud-rate generator and a clock generator simulta- usly. Note, however, that the baud-rate and clock-out uencies cannot be determined independently from one ther since they both use RCAP2H and RCAP2L.

errupts

AT89S52 has a total of six interrupt vectors: two exter- interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Tim- 0, 1, and 2), and the serial port interrupt. These rrupts are all shown in Figure 10.

h of these interrupt sources can be individually enabled lisabled by setting or clearing a bit in Special Function ister IE. IE also contains a global disable bit, EA, which ıbles all interrupts at once.

e that Table 5 shows that bit position IE.6 is unimple- ıted. In the AT89S52, bit position IE.5 is also unimple- ıted. User software should not write 1s to these bit itions, since they may be used in future AT89 products.

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

Timer 0 and Timer 1 flags, TF0 and TF1, are set at '2 of the cycle in which the timers overflow. The values then polled by the circuitry in the next cycle. However, Timer 2 flag, TF2, is set at S2P2 and is polled in the ıe cycle in which the timer overflows.

Table 5. Interrupt Enable (IE) Register

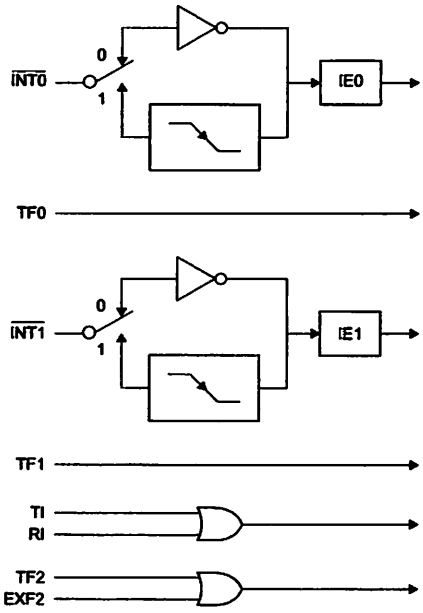
(MSB)							(LSB)
EA	–	ET2	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
–	IE.6	Reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	Serial Port interrupt enable bit.
ET1	IE.3	Timer 1 interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.
User software should never write 1s to unimplemented bits, because they may be used in future AT89 products.		

Figure 10. Interrupt Sources



;cillator Characteristics

\L1 and XTAL2 are the input and output, respectively, ın inverting amplifier that can be configured for use as ɔn-chip oscillator, as shown in Figure 11. Either a quartz stal or ceramic resonator may be used. To drive the ice from an external clock source, XTAL2 should be left :onnected while XTAL1 is driven, as shown in Figure 12. ere are no requirements on the duty cycle of the external :k signal, since the input to the internal clocking circuitry rrough a divide-by-two flip-flop, but minimum and maxi- m voltage high and low time specifications must be ierved.

le Mode

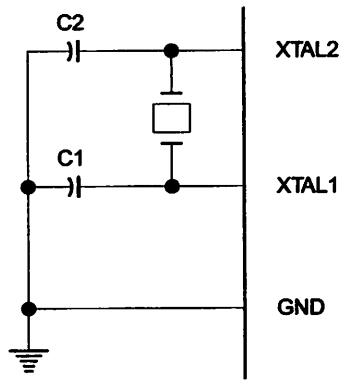
dle mode, the CPU puts itself to sleep while all the on- p peripherals remain active. The mode is invoked by tware. The content of the on-chip RAM and all the spe- l functions registers remain unchanged during this de. The idle mode can be terminated by any enabled rrupt or by a hardware reset.

te that when idle mode is terminated by a hardware et, the device normally resumes program execution n where it left off, up to two machine cycles before the rnal reset algorithm takes control. On-chip hardware ibits access to internal RAM in this event, but access to port pins is not inhibited. To eliminate the possibility of unexpected write to a port pin when idle mode is termi- :ed by a reset, the instruction following the one that ɔkes idle mode should not write to a port pin or to exter- memory.

ɔwer-down Mode

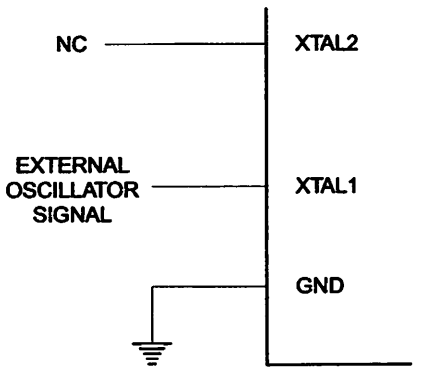
he Power-down mode, the oscillator is stopped, and the truction that invokes Power-down is the last instruction ecuted. The on-chip RAM and Special Function Regis- s retain their values until the Power-down mode is termi- ed. Exit from Power-down mode can be initiated either a hardware reset or by an enabled external interrupt. set redefines the SFRs but does not change the on-chip M. The reset should not be activated before V_{CC} is tored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

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



ɔle 6. Status of External Pins During Idle and Power-down Modes

ɔde	Program Memory	ALE	PSEN	PORT0	PORT1	PORT2	PORT3
e	Internal	1	1	Data	Data	Data	Data
e	External	1	1	Float	Data	Address	Data
wer-down	Internal	0	0	Data	Data	Data	Data
wer-down	External	0	0	Float	Data	Data	Data

›gram Memory Lock Bits

AT89S52 has three lock bits that can be left unpro-nmed (U) or can be programmed (P) to obtain the addi-al features listed in the following table.

le 7. Lock Bit Protection Modes

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

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

ogramming the Flash – Parallel Mode

AT89S52 is shipped with the on-chip Flash memory ly ready to be programmed. The programming interface ds a high-voltage (12-volt) program enable signal and ompatible with conventional third-party Flash or ROM programmers.

AT89S52 code memory array is programmed byte-by-e.

›gramming Algorithm: Before programming the 89S52, the address, data, and control signals should be up according to the Flash programming mode table and ures 13 and 14. To program the AT89S52, take the fol-ing steps:

Input the desired memory location on the address lines.

Input the appropriate data byte on the data lines.

Activate the correct combination of control signals.

Raise $\overline{EA}/V_{PP}$ to 12V.

Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 50 µs.

Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

$\overline{Data}$ Polling: The AT89S52 features $\overline{Data}$ Polling to indicate the end of a byte write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P0.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{Data}$ Polling may begin any time after a write cycle has been initiated.

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

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

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

(000H) = 1EH indicates manufactured by Atmel
(100H) = 52H indicates 89S52
(200H) = 06H

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

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

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

Programming the Flash – Serial Mode

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

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

Either an external system clock can be supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2. The maximum serial clock (SCK)

quency should be less than 1/16 of the crystal frency. With a 33 MHz oscillator clock, the maximum SCK quency is 2 MHz.

rial Programming Algorithm

program and verify the AT89S52 in the serial programng mode, the following sequence is recommended:

Power-up sequence:

Apply power between VCC and GND pins.

Set RST pin to "H".

If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.

Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less than the CPU clock at XTAL1 divided by 16.

The Code array is programmed one byte at a time by supplying the address and data together with the appropriate Write instruction. The write cycle is self-timed and typically takes less than 1 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 device operation.

Power-off sequence (if needed):

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

Set RST to "L".

Turn V_{CC} power off.

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

Serial Programming Instruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 10.

ogramming Interface – Parallel Mode

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

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

ble 8. Flash Programming Modes

ode	V_{CC}	RST	$\overline{PSEN}$	ALE/ $\overline{PROG}$	$\overline{EA}$/ V_{PP}	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.4-0 Address	P1.7-0 Address
rite Code Data	5V	H	L	(2)	12V	L	H	H	H	H	D_{IN}	A12-8	A7-0
ead Code Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A12-8	A7-0
rite Lock Bit 1	5V	H	L	(3)	12V	H	H	H	H	H	X	X	X
rite Lock Bit 2	5V	H	L	(3)	12V	H	H	H	L	L	X	X	X
rite Lock Bit 3	5V	H	L	(3)	12V	H	L	H	H	L	X	X	X
ead Lock Bits 2, 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
hip Erase	5V	H	L	(1)	12V	H	L	H	L	L	X	X	X
ead Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	X 0000	00H
ead Device ID	5V	H	L	H	H	L	L	L	L	L	52H	X 0001	00H
ead Device ID	5V	H	L	H	H	L	L	L	L	L	06H	X 0010	00H

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

ure 13. Programming the Flash Memory (Parallel Mode)

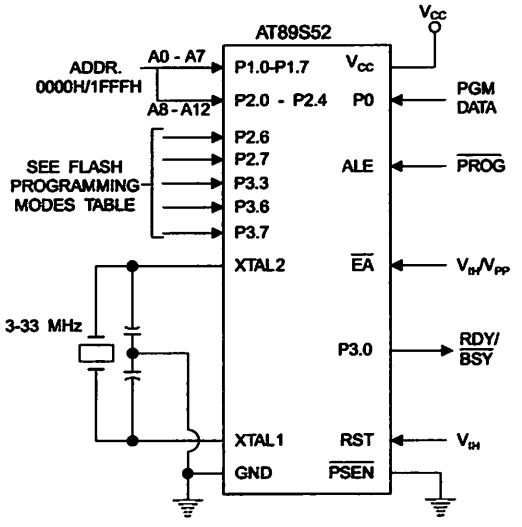
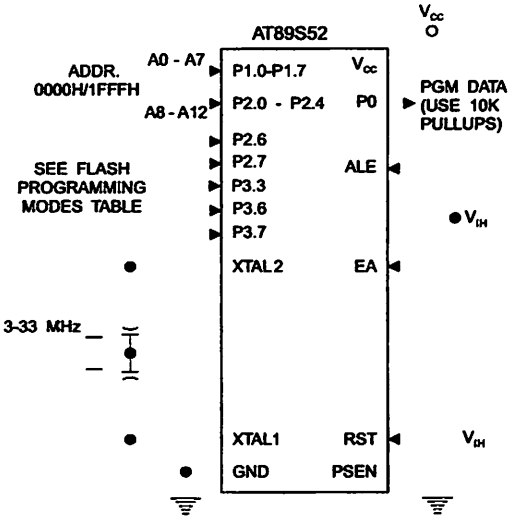


Figure 14. Verifying the Flash Memory (Parallel Mode)

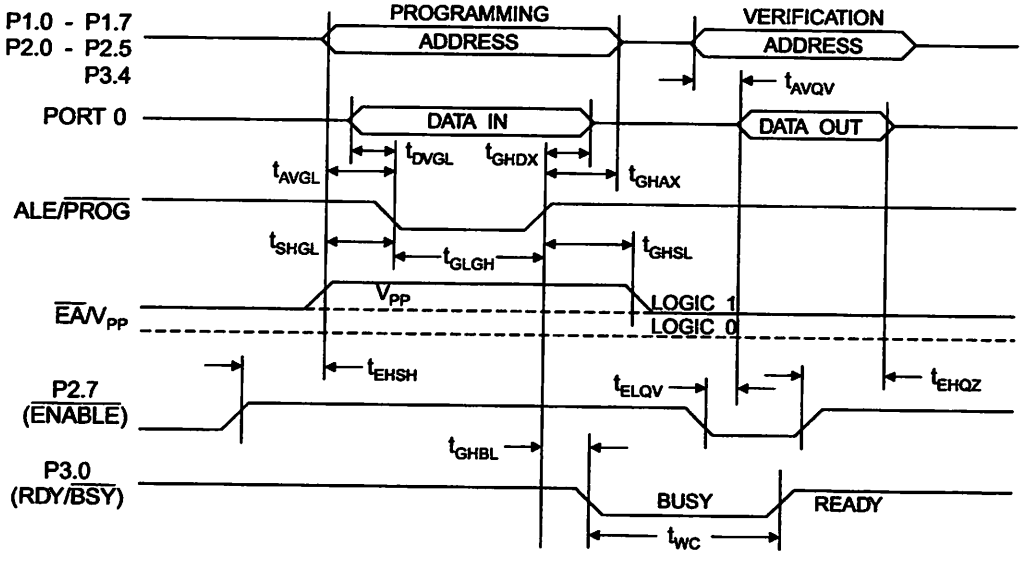


ısh Programming and Verification Characteristics (Parallel Mode)

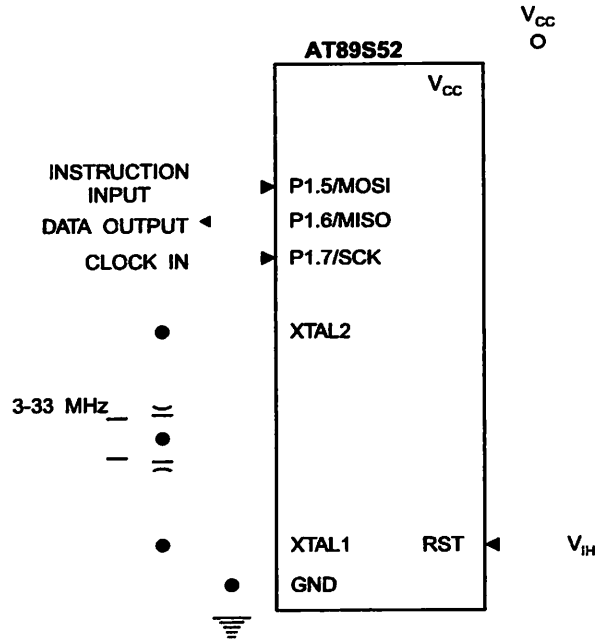
: 20°C to 30°C, V_{CC} = 4.5 to 5.5V

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

re 15. Flash Programming and Verification Waveforms – Parallel Mode

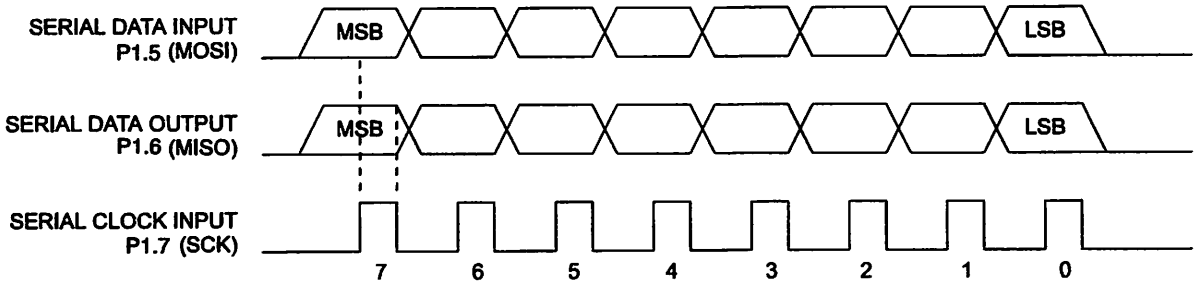


ıre 16. Flash Memory Serial Downloading



ash Programming and Verification Waveforms – Serial Mode

ıure 17. Serial Programming Waveforms



le 9. Serial Programming Instruction Set

	Instruction Format				
truction	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Operation**
›gramming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output)	Enable Serial Programming while RST is high
ip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
ad Program Memory yte Mode)	0010 0000	xxx A12 A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Read data from Program memory in the byte mode
ite Program Memory yte Mode)	0100 0000	xxx A12 A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Write data to Program memory in the byte mode
ite Lock Bits[2]	1010 1100	1110 00 B1 B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (2).
ad Lock Bits	0010 0100	xxxx xxxx	xxxx xxxx	xx LB3 LB2 LB1 xx	Read back current status of the lock bits (a programmed lock bit reads back as a '1')
ad Signature Bytes[1]	0010 1000	xxx A5 A4 A3 A2 A1 A0	xxx xxxx	Signature Byte	Read Signature Byte
ad Program Memory age Mode)	0011 0000	xxx A12 A11 A10 A9 A8	Byte 0	Byte 1... Byte 255	Read data from Program memory in the Page Mode (256 bytes)
ite Program Memory age Mode)	0101 0000	xxx A12 A11 A10 A9 A8	Byte 0	Byte 1... Byte 255	Write data to Program memory in the Page Mode (256 bytes)

es: 1. The signature bytes are not readable in Lock Bit Modes 3 and 4.

2. B1 = 0, B2 = 0 —> Mode 1, no lock protection
   B1 = 0, B2 = 1 —> Mode 2, lock bit 1 activated
   B1 = 1, B2 = 0 —> Mode 3, lock bit 2 activated
   B1 = 1, B1 = 1 —> Mode 4, lock bit 3 activated

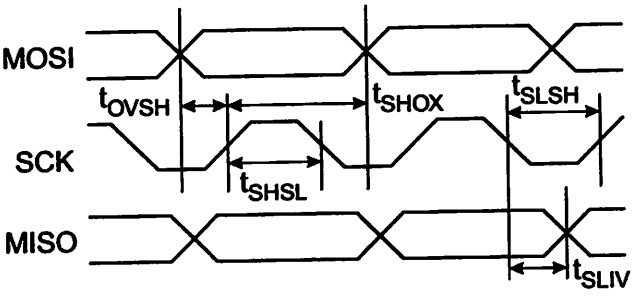
   } Each of the lock bits needs to be activated sequentially before Mode 4 can be executed.

r Reset signal is high, SCK should be low for at least 64 tem clocks before it goes high to clock in the enable a bytes. No pulsing of Reset signal is necessary. SCK uld be no faster than 1/16 of the system clock at AL1.

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

al Programming Characteristics

re 18. Serial Programming Timing



e 10. Serial Programming Characteristics, T_A = -40° C to 85° C, V_{CC} = 4.0 - 5.5V (Unless otherwise noted)

nbol	Parameter	Min	Typ	Max	Units
:LCL	Oscillator Frequency	0		33	MHz
:L	Oscillator Period	30			ns
SL	SCK Pulse Width High	2 t_{CLCL}			ns
:H	SCK Pulse Width Low	2 t_{CLCL}			ns
SH	MOSI Setup to SCK High	t_{CLCL}			ns
OX	MOSI Hold after SCK High	2 t_{CLCL}			ns
V	SCK Low to MISO Valid	10	16	32	ns
ASE	Chip Erase Instruction Cycle Time			500	ms
/C	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	µs

;olute Maximum Ratings*

;rating Temperature	-55°C to +125°C
;age Temperature	-65°C to +150°C
age on Any Pin ; Respect to Ground	-1.0V to +7.0V
;imum Operating Voltage	6.6V
Output Current	15.0 mA

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

Characteristics

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

;bol	Parameter	Condition	Min	Max	Units
	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
;1	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ±10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ±10%	2.4		V
;11		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
;ST	Reset Pulldown Resistor		10	30	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[1]	V_{CC} = 5.5V		50	µA

;es: 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.

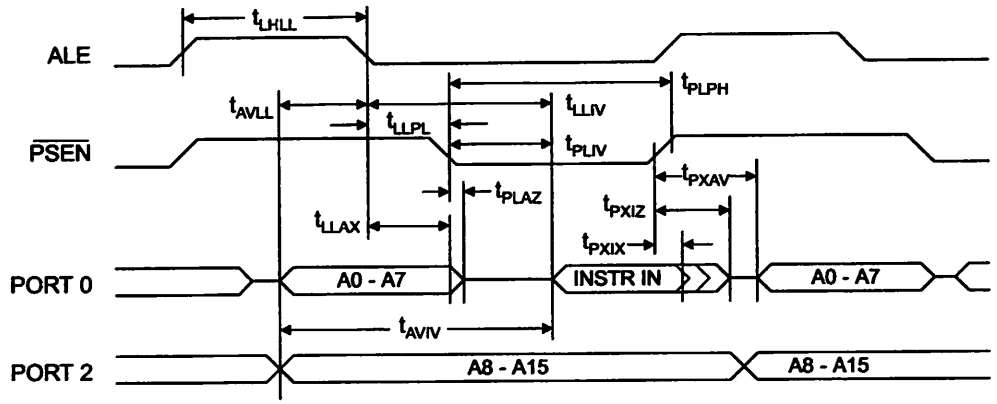
; Characteristics

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

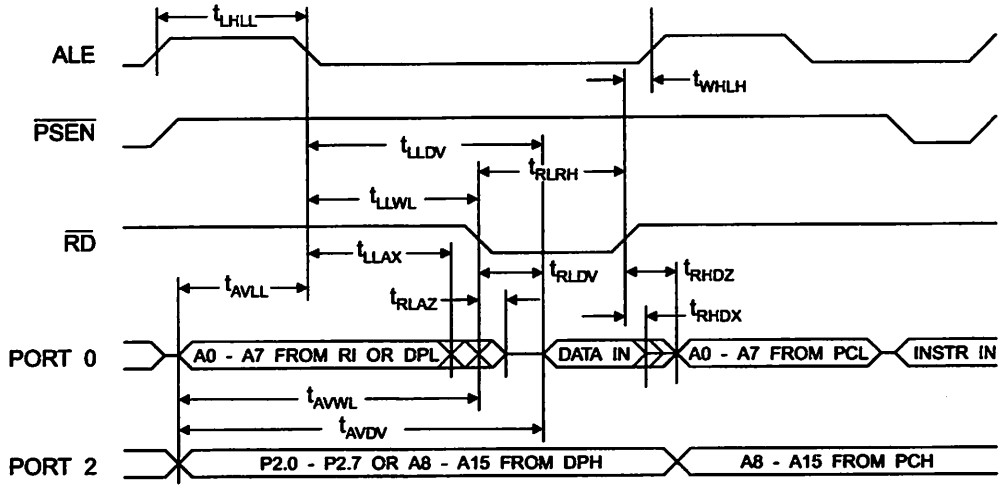
‹ternal Program and Data Memory Characteristics

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

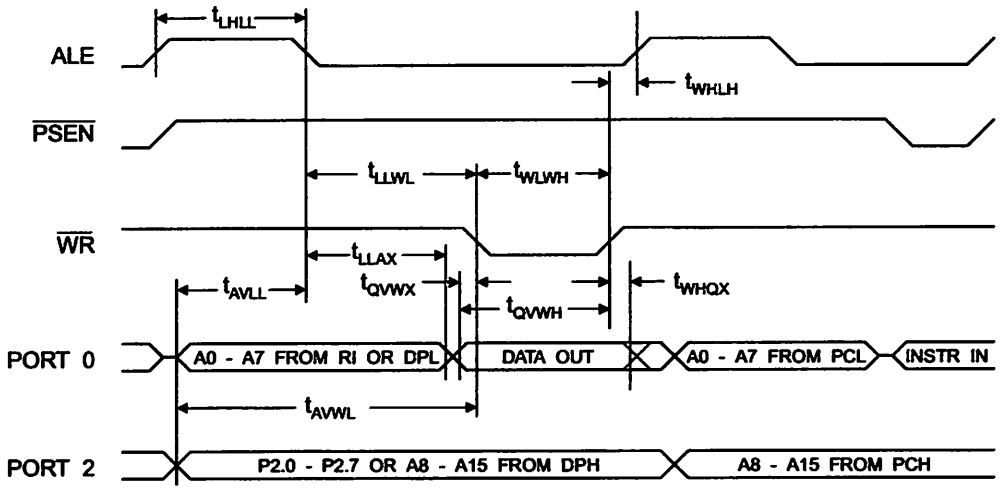
ternal Program Memory Read Cycle



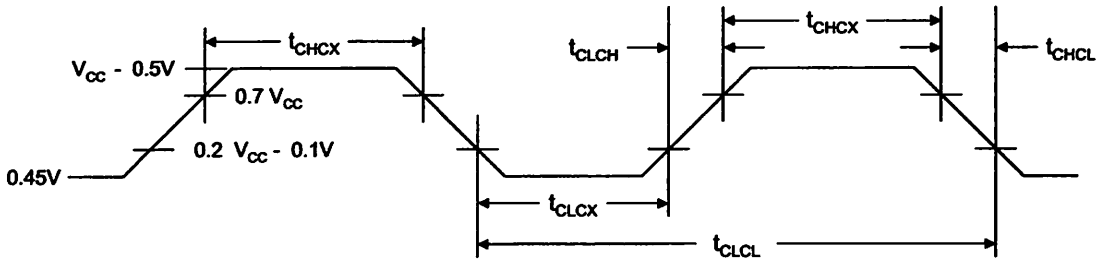
ternal Data Memory Read Cycle



ternal Data Memory Write Cycle



kternal Clock Drive Waveforms



kternal Clock Drive

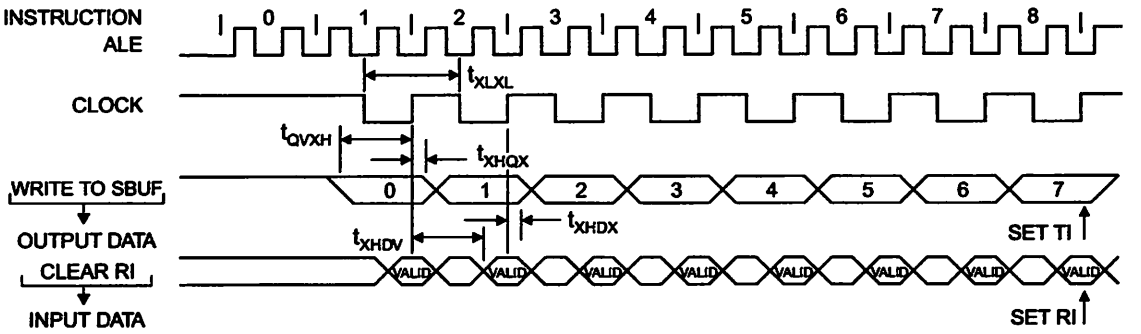
ymbol	Parameter	Min	Max	Units
t_{CLCL}	Oscillator Frequency	0	33	MHz
LCL	Clock Period	30		ns
HCX	High Time	12		ns
LCX	Low Time	12		ns
LCH	Rise Time		5	ns
HCL	Fall Time		5	ns

ırial Port Timing: Shift Register Mode Test Conditions

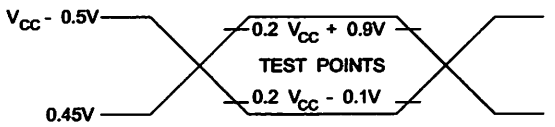
ı values in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

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

hift Register Mode Timing Waveforms

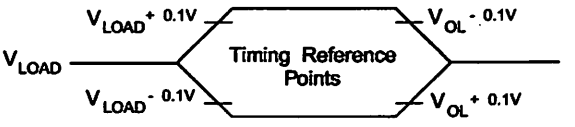


C Testing Input/Output Waveforms[1]



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

Float Waveforms[1]



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

lering Information

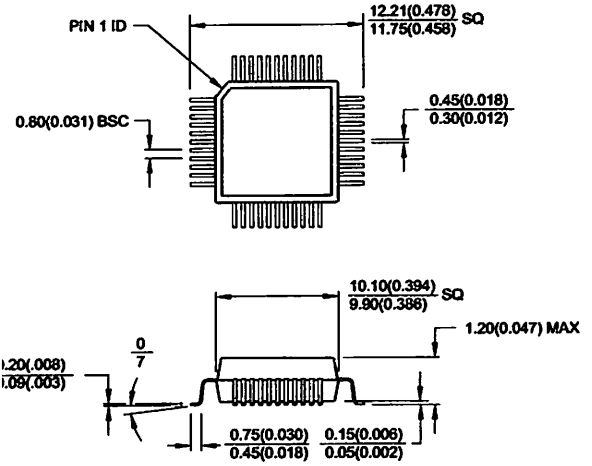
peed MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S52-24AC AT89S52-24JC AT89S52-24PC	44A 44J 40P6	Commercial (0° C to 70° C)
		AT89S52-24AI AT89S52-24JI AT89S52-24PI	44A 44J 40P6	Industrial (-40° C to 85° C)
33	4.5V to 5.5V	AT89S52-33AC AT89S52-33JC AT89S52-33PC	44A 44J 40P6	Commercial (0° C to 70° C)

= Preliminary Availability

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

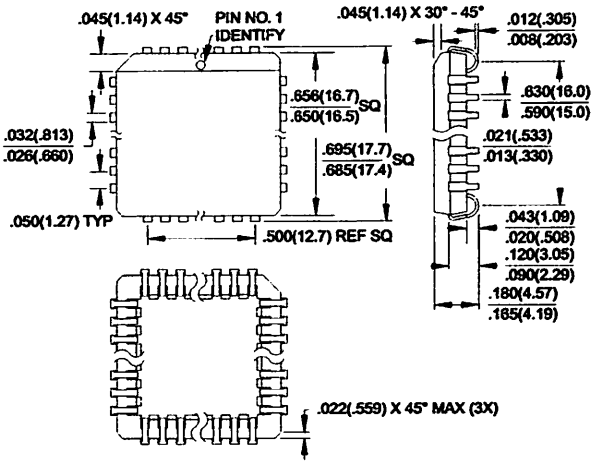
:kaging Information

:4A, 44-lead, Thin (1.0 mm) Plastic Gull Wing Quad
:lat Package (TQFP)
)imensions in Millimeters and (Inches)*

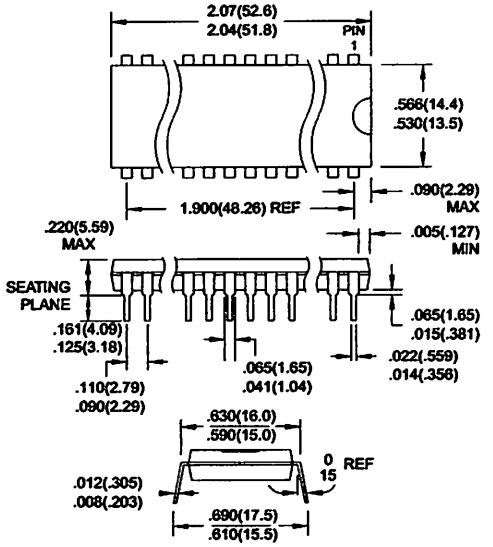


Controlling dimension: millimeters

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



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



tmel Headquarters

ɔrporate Headquarters
2325 Orchard Parkway
San Jose, CA 95131
TEL (408) 441-0311
FAX (408) 487-2600

ɪrope
Atmel SarL
Route des Arsenaux 41
Casa Postale 80
CH-1705 Fribourg
Switzerland
TEL (41) 26-426-5555
FAX (41) 26-426-5500

ɜia
Atmel Asia, Ltd.
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimhatsui
East Kowloon
Hong Kong
TEL (852) 2721-9778
FAX (852) 2722-1369

pan
Atmel Japan K.K.
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
TEL (81) 3-3523-3551
FAX (81) 3-3523-7581

Atmel Product Operations

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

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

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

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

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

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

Fax-on-Demand
North America:
1-(800) 292-8635
International:
1-(408) 441-0732

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

BBS
1-(408) 436-4309

Rev.1919A-07/01/xM

ELINK TECHNOLOGIES CO., LTD
2F, 6 Alley 8, Land 120, Baogao Rd., Xindan City, Taipei 221, Taiwan, R.O.C
TEL:+886-2-2911-1465 FAX:+886-2-2913-7216

1. Introduction

1.1. Overview

ELINK EG-T10 is a high performance, GPS module board designed for easy integration into a broad spectrum of OEM system applications. This product is based on the proven SiRFstarII chipset solution technology found in other ELINK 12 channel GPS receivers. The GPS module receiver tracks up to 12 satellites at a time while providing fast time-to-first-fix and 1 Hz navigation updates. Its far reaching capability meets the sensitivity requirements of car navigation as well as other location-based applications.

The EG-T10 design utilizes the latest surface mount technology and high level circuit integration to achieve superior performance while minimizing space and power requirements. This hardware capability combined with software intelligence makes the board easy to be integrated and used in all kinds of navigation applications or products. The application system may communicate with the module board via two UART channels with CMOS/TTL voltage level.

1.2. Main Features

◆Build on high performance SiRFstarII chipset.
◆Average Cold/Warm/Hot Start time under 45/38/8 seconds.
◆12 channels "All-in-View" tracking.
◆Trickle Power enabled for Power Saving.
◆Integrated ARM7TDMI CPU with software engineering services available for embedded customer defined applications.
◆On chip 1Mb SRAM, optional on board SRAM up to 8Mb
◆On board 4Mb Flash, can up to 16Mb.
◆Dual TTL level serial ports with one for GPS receiver command message interface, and one for RTCM-104 DGPS input.
◆Mechanical Size 2.8" x1.6" x0.57" (71.12x 40.64x14.4mm)
◆Reacquisition Time：0.1 seconds
◆Support Standard NMEA-0183 and SiRF Binary protocol
◆Multi-path Mitigation Hardware
◆On-board RTCM SC-104 DGPS and WAAS Demodulator

2. Technical Specifications

2.1. Electrical Characteristics

2.1.1 General

Frequency L1, 1575.42 MHz

C/A code	1.023 MHz chip rate
Channels	12

2.1.2 Accuracy

Position	15 meters, 2D RMS
	7 meters 2D RMS, WAAS corrected
	1-5 meters, DGPS corrected
Velocity	0.1 meters/second Time
	1 microsecond synchronized to GPS time

2.1.3 Datum

Default	WGS-84
Other	Support different datum by request

2.1.4 Acquisition Rate (Open sky, stationary requirements)

Reacquisition	0.1 sec., average
Snap start	2 sec., average
Hot start	8 sec., average
Warm start	38 sec., average
Cold start	45 sec., average

2.1.5 Dynamic Conditions

Altitude	18,000 meters (60,000 feet) max.
Velocity	515 meters/second (1000 knots) max.
Acceleration	4g, max.
Jerk	20 meters/second^3, max.

2.1.6 Power

EG-T10 3.3V Version (TTL)	
Main Power	3.3VDC ± 10%
Supply Current	125mA Typical (Without antenna at 3.3V operation)
Backup Power	+2.5V to 3.6V
Backup Current	10μA Typical

Note：Optional 5V regulator on board

2.1.7 Serial Port

Electrical interface	Two full duplex serial communication, TTL interface
Protocol messages	SiRF binary and NMEA-0183, version 2.20 with a baud rate selection.
	SiRF binary-position, velocity, altitude, status, and control NMEA -GGA, GLL, GSA,GSV, RMC and VTG
DGPS protocol	RTCM SC-104

2.1.8 Time-1PPS Pulse

Level	TTL
Pulse duration	100ms
Time reference	At the pulse positive edge
Measurements	Aligned to GPS second, ±1 microsecond

2.2. Environmental Characteristics

Operating temperature range	-40 deg. C to +85 deg. C
Storage temperature range	-55 deg. C to +100 deg. C

2.3. Physical Characteristics

Length	2.8" (71.12 mm)
Width	1.6"(40.64mm)
Height	0.57"(14.4 mm)
Weight	25g
Antenna connector	MCX type (Optional RMCX, RSMA type)
Interface connector	20-pin straight header, 2mm pitch board-to-board and 4mm height, for example, TMM-110-03-L-D of Samtec's connector
Matched connector	standard socket, TLE-110-01-G-DV of Samtec's connector

2.4. Trickle Power Description

The EG-T10 design supports the SiRF Trickle Power™ mode of operation. In this mode, the lowest average power dissipation is achieved by powering down the board (after a position is determined) in such a manner that when it is turned back on it can recomputed a position fix in the shortest amount of time. Standard Trickle Power operates in three states:

2.4.1 Tracking State

In this state, the board is fully powered, tracking satellites, and gathering data.

2.4.2 CPU State

In this mode, the RF front end has been turned off which removes the clock to the baseband. Without a clock, the RF front end is effectively powered down (although the RTC keeps running). The CPU would keep running to process the GPS data until a position fix is determined and the result has been transmitted by the serial communication interface.

2.4.3 Trickle State

In this state, the CPU is in a low power standby state and the receiver clocks are off with only the RTC clock active. After a set amount of time, the RTC generates an NMI signal to wake up the CPU and reset the receiver back to tracking state.

2.5. Push-to-Fix Description

The purpose of Push-to-Fix mode is to support applications where a position fix is only required when requested by the user (or the application). To support this the board is left in the Trickle state until commanded to generate a fix by toggle reset.

2.5.1 Power-on State

In this state, the receiver calculates the position once, collects the ephemeris, and calibrates the RTC before going back to the Trickle State.

2.5.2 Trickle State

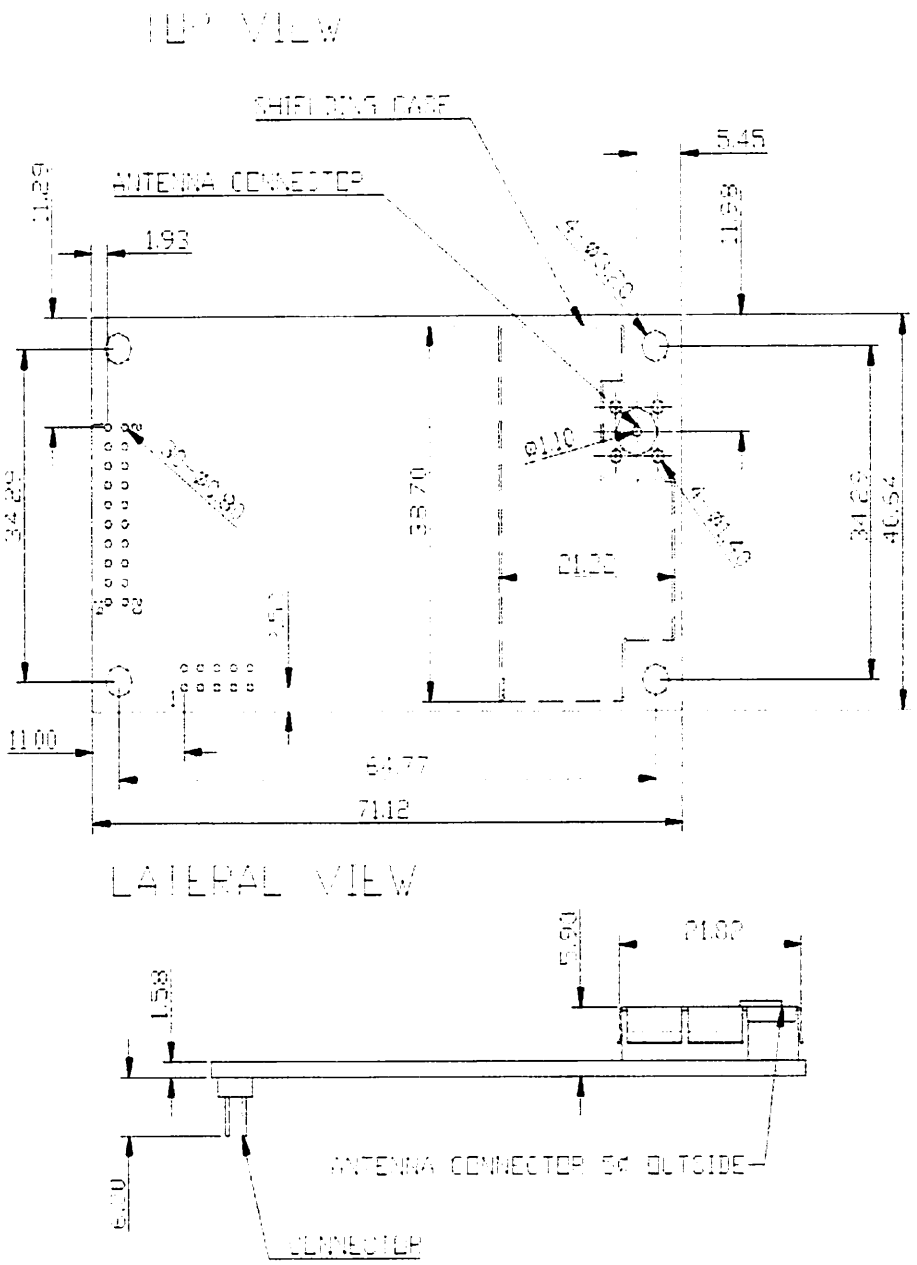
In this state only the RTC is running. The supply current is typically <500uA, which includes the standby current of the baseband IC.

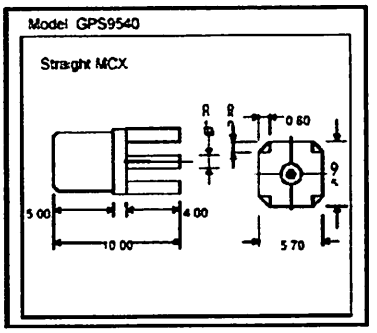
There are three events that can happen which effectively return the CPU to normal operation:

1 Power-on: If power is removed, then re-applied to the board a reset signal is generated by the CPU supervisor. After the reset has been removed, the CPU will start up, get a fix and return to Trickle State. This typically takes two to six seconds.
2 Ephemeris Collection: Every 30 minutes the receiver wakes up the CPU to calculate a fix, collect a new ephemeris, calibrate the RTC and then go to the Trickle State.
3 User Requested Fix: With each user request of a fix, the CPU will wake up by toggling PBRES low (pin 5 of the digital interface connector). The CPU is restarted and (following a Snap Start) a fix is calculated. Before going back to Trickle State the CPU will check the ephemeris and the RTC calibration.

ELINK TECHNOLOGIES CO., LTD
2F, 6 Alley 8, Land 120, Baogao Rd., Xindan City, Taipei 221, Taiwan, R.O.C
TEL:+886-2-2911-1465 FAX:+886-2-2913-7216

3. Mechanical Dimensions





4. Pin Assignment of Connector

Table 4-1 Pin list of the 20-Pin Digital Interface Connector (J1)

4. Pin Assignment of Connector

Pin Number	Name	Type	Description
1	ANT_PWR	PWR	Antenna DC Voltage
2	VCC_5V	PWR	+5 DC Power Input(note1)
3	BAT	PWR	Backup Battery
4	VCC_3V	PWR	+3.3V DC Power Input(note2)
5	PBRES	I	Push Button Reset Input. Active Low
6	GPIO3	I/O	SW dependent functions (note3)
7	SK/GPIO7	I/O	SW dependent functions (note3)(note5)
8	SO/GPIO6	I/O	SW dependent functions (note3)(note5)
9	SI/GPIO5	I/O	SW dependent functions (note3)(note5)
10	GND	PWR	Ground
11	TXA	O	Serial Data Output A
12	RXA	I	Serial Data Input A
13	GND	PWR	Ground
14	TXB	O	Serial Data Output B
15	RXB	I	Serial Data Input B
16	GND	PWR	Ground
17	BOOTSEL	I	Booting Mode Select(note6)
18	GND	PWR	Ground
19	TIMEMARK/GPIO9	I/O	1PPS Time Mark Output (note3)
20	GPIO15	I/O	SW dependent functions (note4)

Note: 1) If the module is 3.3V type, the pin is not used.
2) If the module is 5V type, the pin in no used.
3) Pulled low (GND) through on-board resister and please float this pin if not use.
4) Pulled high (VCC/VDD) through on-board resister.
5) SI, SO, SK are synchronous serial interface (in, out and clock).
6) The pin is active high and float when not use

4.1 VCC_5V (+5V DC Power Input)

This is the main DC power supply for a +5V powered board. (Required for the EG-T10 5V version)

4.2 VCC_3V (+3.3V DC Power Input)

This is the main DC power supply for a +3.3V powered board. (Required for the EG-T10 3.3V version)

4.3 ANT_PWR

DC voltage for an active antenna. The antenna power may be supplied through the interface connector (CN1). The current limiting (<200mA) should be supplied externally. Therefore, DC voltage is dependent on active antenna that you choose.

4.4 GND

GND provides the ground for the board. Connect all grounds.

4.5 Serial Data: RXA, RXB, TXA, and TXB

The board supports two full duplex serial channels. All four connections are at TTL levels, all support variable baud rates, and all can be controlled from the appropriate screens in GPS Monitor software.
（TTL level should be converted to RS-232 level when communicate with PC serial port.）

4.6 RXA

This is the main receiving channel and is used to receive software commands to the board from GPS Monitor software or from user written software.

4.7 RXB

This is the auxiliary receiving channel and is used to input differential corrections to the board to enable DGPS navigation.

4.8 TXA

This is the main transmitting channel and is used to output navigation and measurement data to GPS Monitor or user written software.

4.9 TXB

Reserved

4.10 PBRES

This pin provides an active-low reset input to the board. It causes the board to reset and start searching for satellites. PB Reset is an optional input and, if not utilized, it may be left open.

4.11 Timemark

This pin provides 1 pulse per second output from the board, which is synchronized to within 1 microsecond of GPS time. The output is a TTL positive level signal. Only upon a situation of tracking or navigating will output once per second.

The 1 PPS function is not available in Trickle Power mode.

4.12 GPIO Functions

Several I/Os of CPU are connected to the digital interface connector for customized applications.

4.13 Battery (BAT)

This is the battery backup supply that powers the SRAM and RTC when main power is off.

4.14 Booting Mode Select

When you want to update your development firmware, you have two methods to finish this procedure. First, you can use the GMonitor.exe to update firmware to the GPS module and we call software download for general users. When using "force download" function, you need to co-work with pulling high BOOTSEL PIN. Please refer to the manual of GMonitor. **Therefore, when you make the BOOTSEL pin high at power on, the GPS will get into the waiting for burn-in firmware.**

5. Applications

ELINK EG-T10 module board receiver is a high performance, low power consumption, easy integration and not expensive product. It can be now in widespread use about global positioning system. These applications are as follow.

◆Car Navigation
◆Marine Navigation
◆Fleet Management
◆AVL and Location-Based Services
◆Hand-Held Device for Personal Positioning and Navigation
◆Other Computing Devices at GPS Application

6. Operation and Test

The customers can use ELINK GMonitor.exe to test the smart antenna. You can change the data protocol and communication data baud rate for your application using this software. The software and manual are available for download from ELINK website, www.elink-tech.com.tw.

ELINK TECHNOLOGIES CO., LTD
2F, 6 Alley 8, Land 120, Baogao Rd., Xindan City, Taipei 221, Taiwan, R.O.C
TEL:+886-2-2911-1465 FAX:+886-2-2913-7216

Appendix A Other Electrical Specifications

Table A-1 Absolute Maximum Ratings

Parameter	Symbol	Min	Max	Units
Power Supply Voltage	VCC_5V	-0.3	6	V
Power Supply Voltage	VCC_3V	-0.3	3.6	V
Input Pin Voltage	VIN	-0.3	5	V
Output Pin Voltage	VOUT	-0.3	3.6	V
Storage Temperature	TSTG	-45	100	ºC

Table A-2 Operating Conditions

Parameter	Symbol	Min	Typ	Max	Units
Power Supply Voltage	VCC_5V	3.5	5	6	V
Power Supply Voltage	VCC_3V	3.2	3.3	3.4	V
Operating Temperature	TOPR	-40		85	ºC
Operating Current	ICC		160		mA

Table A-3 Backup Battery Conditions

Parameter	Symbol	Min	Typ	Max	Units
RTC(Battery) Power	BAT	2.5	3	3.3	V
Supply Current			10		µA

Table A-4 DC Characteristics

Parameter	Symbol	Min	Max	Conditions	Units
Input High Level	Vih	2.0	5		V
Input Low Level	Vil	0	0.8		V
Output High Level	Voh	2.4	3.4		V
Out Low Level	Vol	0	0.4		V
Input Leakage Current	Ii	2	2		µA
Input Capacitance			3		pF
Output Capacitance			3		pF
Bi-directional Buffer Capacitance			3		pF

Table A-5 Average Current Calculations for GPSCLK TricklePower, 200msec

On Time (VCC_3V Version)

Item	Current(mA)	Time (ms)		
Update Period		1080	6000	10080
Tracking State	170	280	280	280
CPU State	55	260	260	260
Sat Positioning	55	600	700	700
Trickle State	5	540	4760	8840
Average Current (mA)		59.8	20.7	14.3
Average Power (mW)		197.4	68.3	47.3

<Note>

1 Update Period is greater than programmed value by approximately 80-100 msec.

2 With all development data messages turned on the CPU state time increases.

3 Satellite positioning task occurs during CPU State once per 6 second interval. With one second update interval, the satellite positioning task takes about 100 msec less, since it uses the available throughput during the following task state.

4 Total Trickle State time computed over 6 second interval for 1000 and 6000 msec update period and over 10 second interval for 10000 msec update period.

```
        mov     DPTR,#univ
        acall   line1
        mov     Hurf,#16
        acall   tulis
        acall   line2
        mov     Hurf,#16
        acall   tulis
        acall   delay1
        clr     Bkgl
        acall   blank
;
        lcall   sr_in0                  ; serial 19200,8,N,1
        lcall   noecho                  ; no respon character
        lcall   smsfrm                  ; sms format
;
mulai:  lcall   bc_fbk
        acall   delays
        acall   delays
        lcall   bc_sms
        sjmp    mulai
;
bc_sms: lcall   bcnohp                  ;\
        mov     A,Stts                  ; | baca no HP
        cjne    A,#1,bcsmlp             ; | cek status
        lcall   bccmnd                  ; | salah = hapus sms
        lcall   hpssms                  ; | benar = baca command
        ljmp    bcsms0                  ; | hapus sms
bcsmlp: lcall   hpssms                  ; | cek command
        ret                             ;/
;
bcsms0: mov     DPTR,#comnd0            ;\
        lcall   ckcmnd                  ; | ambil data command
        mov     A,Stts                  ; | cek command
        cjne    A,#1,bcsms1             ; | valid -> kirim posisi
        lcall   sr_in1                  ; serial 4800,8,N,1
        acall   bcdgga
        lcall   sr_in0                  ; serial 19200,8,N,1
        acall   krmpos                  ; | tidak -> lompat ke label mulai
        ljmp    mulai                   ;/
;
bcsms1: mov     DPTR,#comnd1            ;\
        lcall   ckcmnd                  ; | ambil data command
        mov     A,Stts                  ; | cek command
        cjne    A,#1,bcsms2             ; | valid -> matikan mesin
        clr     Rly1                    ; | tidak -> lompat ke label mulai
        ljmp    mulai                   ;/
;
bcsms2: mov     DPTR,#comnd2            ;\
        lcall   ckcmnd                  ; | ambil data command
        mov     A,Stts                  ; | cek command
        cjne    A,#1,bcsms3             ; | valid -> nyalakan mesin
        setb    Rly1                    ; | tidak -> lompat ke label mulai
        ljmp    mulai                   ;/
;
bcsms3: mov     DPTR,#comnd3            ;\
        lcall   ckcmnd                  ; |
        mov     A,Stts                  ; |
        cjne    A,#1,bcsms4             ; |
        clr     Rly0                    ; |
        setb    Bkgl                    ; |
        mov     DPTR,#prgtn1            ; |
        acall   line1                   ; | ambil data command
        mov     Hurf,#16                ; | cek command
        acall   tulis                   ; | valid -> kirim pesan
        acall   line2                   ; | peringatan 1
        mov     Hurf,#16                ; |
        acall   tulis                   ; |
        acall   delay1                  ; |
        setb    Rly0                    ; |
        acall   delayp                  ; |
        clr     Bkgl                    ; |
        acall   blank                   ; |
        ljmp    mulai                   ;/
;
bcsms4: mov     DPTR,#comnd4            ;\
        lcall   ckcmnd                  ; |
        mov     A,Stts                  ; |
```

```
        cjne    A,#1,bcsms5             ; |
        clr     Rly0                    ; |
        setb    Bkgl                    ; |
        mov     DPTR,#prgtn2            ; |
        acall   line1                   ; | ambil data command
        mov     Hurf,#16                ; | cek command
        acall   tulis                   ; | valid -> kirim pesan
        acall   line2                   ; | peringatan 2
        mov     Hurf,#16                ; |
        acall   tulis                   ; |
        acall   delayl                  ; |
        acall   delayl                  ; |
        acall   delayl                  ; |
        clr     Bkgl                    ; |
        setb    Rly0                    ; |
        acall   blank                   ; |
        ljmp    mulai                   ;/
;
bcsms5: mov     DPTR,#comnd5            ;\
        lcall   ckcmnd                  ; |
        mov     A,Stts                  ; |
        cjne    A,#1,bcsms6             ; |
        clr     Rly0                    ; |
        setb    Bkgl                    ; |
        mov     DPTR,#prgtn3            ; |
        acall   line1                   ; | ambil data command
        mov     Hurf,#16                ; | cek command
        acall   tulis                   ; | valid -> kirim pesan
        acall   line2                   ; | peringatan 3
        mov     Hurf,#16                ; |
        acall   tulis                   ; |
        acall   delayl                  ; |
        acall   delayl                  ; |
        acall   delayl                  ; |
        clr     Bkgl                    ; |
        setb    Rly0                    ; |
        acall   blank                   ; |
        ljmp    mulai                   ;/
;
bcsms6: ljmp    mulai
;
krmpos: lcall   jd_pdu                  ;\
        mov     DPTR,#smskrm            ; |
        mov     Char,#10                ; |
        lcall   kr_ins                  ; |
        lcall   bc_fbk                  ; |
        lcall   delays                  ; |
        mov     DPTR,#dtapos            ; | rubah character jadi PDU &
kirim
        mov     Char,#44                ; |
        lcall   kr_hlf                  ; |
        lcall   kr_pdu                  ; |
        lcall   bc_fbk                  ; |
        lcall   delays                  ; |
        ret                             ;/
;
bcnohp: mov     DPTR,#smsred            ;\
        mov     Char,#9                 ; |
        lcall   kr_ins                  ; | baca no hp
        lcall   bc_fbk                  ; | header sms + service center
        mov     Cnt0,#35                ; | dihilangkan
bcnolp: lcall   bc_srl                  ; |
        djnz    Cnt0,bcnolp             ; |
        lcall   bc_dta                  ;/
;
cknohp: mov     DPTR,#nohpku            ;\
        mov     Cnt0,#14                ; | jumlah no hp = 14
        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    Cnt0,cknhp0             ; | cek sampai no hp habis
        mov     Stts,#1                 ; | set status = 1
        ljmp    cknhp2                  ; | jika tidak
cknhp1: mov     Stts,#0                 ; | set status = 0
cknhp2: ret                             ;/
;
dtanhp: mov     A,Cndt                  ;\
        movc    A,@A+DPTR               ; | ambil data pointer yang ke cndt
        ret                             ;/
;
bccmnd: lcall   bc_hp                   ;\
        mov     DPTR,#smsred            ; |
        mov     Char,#9                 ; |
        lcall   kr_ins                  ; | baca command
        lcall   bc_fbk                  ; | header sms + service center +
no HP
        mov     Cnt0,#69                ; | dihilangkan
bccmlp: lcall   bc_srl                  ; |
        djnz    Cnt0,bccmlp             ; |
        lcall   bc_dta                  ; |
        ret                             ;/
;
ckcmnd: mov     Cnt0,#10                ;\
        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    Cnt0,ckcnd0             ; | cek sampai command habis
        mov     Stts,#1                 ; | set status = 1
        ljmp    ckcnd2                  ; | jika tidak
ckcnd1: mov     Stts,#0                 ; | set status = 0
ckcnd2: ret                             ;/
;
dtacmd: mov     A,Cndt                  ;\
        movc    A,@A+DPTR               ; | ambil data pointer yang ke cndt
        ret                             ;/
;
bcdgga: lcall   bc_gps                  ;\
bcglp0: lcall   bc_fbk                  ; | baca gps
        cjne    A,#'R',bcglp0           ; | yang diawali character R
        mov     Cnt0,#17                ; | 17 character dihilangkan
bcglp1: lcall   bc_srl                  ; | baca data serial gps
        djnz    Cnt0,bcglp1             ;/
        lcall   bc_srl
        mov     Ch10,A
        lcall   bc_srl
        mov     Ch11,A
        lcall   bc_srl
        mov     Ch12,A
        lcall   bc_srl
        mov     Ch13,A                  ; asli data titik
        lcall   bc_srl
        mov     Ch14,A
        lcall   bc_srl
        mov     Ch15,A
        lcall   bc_srl
        lcall   bc_srl
        lcall   bc_srl
        mov     Ch16,#' '               ; tambahan data spasi
        lcall   bc_srl
        mov     Ch17,A                  ; asli data lintang (S)
        lcall   bc_srl
        mov     Ch18,A                  ; asli data koma
        lcall   bc_srl
;
        mov     Ch19,A
        lcall   bc_srl
        mov     Ch1A,A
        lcall   bc_srl
```

```
        mov     Ch1B,A
        lcall   bc_srl
        mov     Ch1C,A
        lcall   bc_srl
        mov     Ch1D,A
        lcall   bc_srl
        mov     Ch1E,A                  ; asli data titik
        lcall   bc_srl
        mov     Ch1F,A
        lcall   bc_srl
        mov     Ch20,A
        lcall   bc_srl
        lcall   bc_srl
        lcall   bc_srl
        mov     Ch21,#' '               ; tambah data spasi
        lcall   bc_srl
        mov     Ch22,A                  ; asli data bujur (E)
        mov     Ch23,#' '               ; tambah data spasi
        lcall   delays
        lcall   delays
        ret
;
dtasms: clr     C                       ;\
        mov     A,Cnsm                  ; |
dtsm00: cjne    A,#00,dtsm01            ; |
        mov     B,Ch00                  ; |
dtsm01: cjne    A,#01,dtsm02            ; |
        mov     B,Ch01                  ; |
dtsm02: cjne    A,#02,dtsm03            ; |
        mov     B,Ch02                  ; |
dtsm03: cjne    A,#03,dtsm04            ; |
        mov     B,Ch03                  ; |
dtsm04: cjne    A,#04,dtsm05            ; |
        mov     B,Ch04                  ; |
dtsm05: cjne    A,#05,dtsm06            ; |
        mov     B,Ch05                  ; |
dtsm06: cjne    A,#06,dtsm07            ; |
        mov     B,Ch06                  ; |
dtsm07: cjne    A,#07,dtsm08            ; |
        mov     B,Ch07                  ; | samakan data counter
dtsm08: cjne    A,#08,dtsm09            ; | dengan character
        mov     B,Ch08                  ; |
dtsm09: cjne    A,#09,dtsm0A            ; |
        mov     B,Ch09                  ; |
dtsm0A: cjne    A,#10,dtsm0B            ; |
        mov     B,Ch0A                  ; |
dtsm0B: cjne    A,#11,dtsm0C            ; |
        mov     B,Ch0B                  ; |
dtsm0C: cjne    A,#12,dtsm0D            ; |
        mov     B,Ch0C                  ; |
dtsm0D: cjne    A,#13,dtsm0E            ; |
        mov     B,Ch0D                  ; |
dtsm0E: cjne    A,#14,dtsm0F            ; |
        mov     B,Ch0E                  ; |
dtsm0F: cjne    A,#15,dtsmsx            ; |
        mov     B,Ch0F                  ; |
dtsmsx: ret                             ;/
;
bc_dta: lcall   bc_srl                  ;\
        mov     Ch00,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch01,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch02,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch03,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch04,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch05,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch06,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch07,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch08,A                  ; | baca serial & simpan
        lcall   bc_srl                  ; | ke memory character
```

```
        mov     Ch09,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0A,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0B,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0C,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0D,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0E,A                  ; |
        lcall   bc_srl                  ; |
        mov     Ch0F,A                  ; |
        lcall   delays                  ; |
        lcall   delays                  ; |
        lcall   delays                  ; |
        ret                             ;/
;
noecho: mov     DPTR,#smseco            ;\
        mov     Char,#4                 ; |
        lcall   kr_ins                  ; |
        acall   bc_fbk                  ; | kirim instruksi no-echo ke hp
        acall   delays                  ; |
        acall   delays                  ; |
        ret                             ;/
;
smsfrm: mov     DPTR,#smsfmt            ;\
        mov     Char,#17                ; |
        acall   kr_ins                  ; |
        acall   bc_fbk                  ; | kirim instruksi display sms ke
hp
        acall   delays                  ; |
        acall   delays                  ; |
        ret                             ;/
;
hpssms: mov     DPTR,#smshps            ;\
        mov     Char,#9                 ; |
        lcall   kr_ins                  ; |
        lcall   bc_fbk                  ; | hapus sms
        lcall   delays                  ; |
        acall   delays                  ; |
        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_srl: mov     A,#0FFh                 ;\
        mov     Tmo0,#10                ; |
tmlp:   mov     Tmo1,#255               ; | 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                             ;/
;
kr_ins: clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim instruksi ke hp
        djnz    Char,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 pdu
        inc     DPTR                    ; | yang sudah pasti
        djnz    Char,kr_hlf             ; |
        ret                             ;/
;
kr_dta: clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim data olahan
        djnz    Char,kr_dta             ; | dari character ke pdu
```

```
        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   jeda                    ; |
        ret                             ;/
;
sr_in0: lcall   bc_hp                   ;\
        setb    EA                      ; |
        mov     TMOD,#20h               ; |
        mov     TH1,#0FDH               ; |
        setb    TR1                     ; | set relay posisi hp
        mov     SCON,#50h               ; | set baudrate 19200bps
        mov     A,#80h                  ; | double baudrates
        orl     87h,A                   ; |
        setb    ES                      ; |
        lcall   delays                  ; |
        ret                             ;/
;
sr_in1: lcall   bc_gps                  ;\
        setb    EA                      ; |
        mov     TMOD,#20h               ; |
        mov     TH1,#0FAH               ; |
        setb    TR1                     ; | set relay posisi gps
        mov     SCON,#50h               ; | set baudrate 4800bps
        mov     A,#80h                  ; | single baudrates
        xrl     87h,A                   ; |
        setb    ES                      ; |
        lcall   delays                  ; |
        ret                             ;/
;
bc_hp:  clr     Slct                    ;\
        lcall   delays                  ; | relay posisi ke hp
        ret                             ;/
;
bc_gps: setb    Slct                    ;\
        lcall   delays                  ; | relay posisi ke gps
        ret                             ;/
;
blank:  mov     DPTR,#kosong
        acall   line1
        mov     Hurf,#16
        acall   tulis
        mov     DPTR,#kosong
        acall   line2
        mov     Hurf,#16
        acall   tulis
        ret
;
line1:  mov     R0,#80h                 ;\
        lcall   w_ins                   ; | instruksi lcd line 1
        ret                             ;/
;
line2:  mov     R0,#0C0h                ;\
        lcall   w_ins                   ; | instruksi lcd line 2
        ret                             ;/
;
tulis:  clr     A                       ;\
        movc    A,@A+DPTR               ; |
        mov     R0,A                    ; | tulis chraracter ke lcd
        inc     DPTR                    ; | dari data pointer
        lcall   w_chr                   ; | sebanyak huruf
        djnz    Hurf,tulis              ; |
        ret                             ;/
;
wr_chr: movc    A,@A+DPTR               ;\
        mov     R0,A                    ; | tulis character
        lcall   w_chr                   ; | ke lcd
        ret                             ;/
;
w_ins:  clr     Enbl                    ;\
        clr     Rest                    ; |
```

```
        mov     P0,R0                   ; |
        setb    Enbl                    ; | kirim instruksi ke lcd
        clr     Enbl                    ; |
        lcall   jeda                    ; |
        ret                             ;/
;
w_chr:  clr     Enbl                    ;\
        setb    Rest                    ; |
        mov     P0,R0                   ; |
        setb    Enbl                    ; | kirim character ke lcd
        clr     Enbl                    ; |
        lcall   jeda                    ; |
        ret                             ;/
;
lcd_op: lcall   delays                  ;\
        mov     R0,#01h                 ; |
        lcall   w_ins                   ; |
        mov     R0,#38h                 ; | Display Clear
        lcall   w_ins                   ; | Function Set
        mov     R0,#0Dh                 ; | Display On, Cursor, Blink
        lcall   w_ins                   ; | Entry Mode
        mov     R0,#06h                 ; | Cursor Home
        lcall   w_ins                   ; |
        mov     R0,#02h                 ; |
        lcall   w_ins                   ; |
        lcall   delays                  ; |
        ret                             ;/
;
jeda:   djnz    Dly0,$                  ;\ jeda selama
        ret                             ;/ 2 inst x 256 = 512us
;
delays: lcall   jeda                    ;\
        djnz    Dly1,delays             ; | delay selama 256 x panggil jeda
        ret                             ;/
;
delay1: mov     Dly2,#20                ;\
dely1:  lcall   delays                  ; | delay selama
        djnz    Dly2,dely1              ; | 20 x panggil delays
        ret                             ;/
;
delayp: mov     Dly2,#200               ;\
dlyp:   djnz    Dly0,$                  ; |
        djnz    Dly1,dlyp               ; | delay pesan
        djnz    Dly2,dlyp               ; |
        ret                             ;/
;
nama:   DB      ' M.Indra Wijaya '      ;\
        DB      '  NIM. 0217066  '      ; | karakter
univ:   DB      ' Teknik Elektro '      ; | nama & universitas
        DB      '   ITN Malang   '      ;/
prgtn1: DB      '   Peringatan   '      ; peringatan 1
        DB      'Wkt Kurang 3 Jam'      ; waktu kurang 3 jam
prgtn2: DB      '   Peringatan   '      ; peringatan 2
        DB      'Wkt Kurang 2 Jam'      ; waktu kurang 2 jam
prgtn3: DB      '   Peringatan   '      ; peringatan 3
        DB      'Wkt Kurang 1 Jam'      ; waktu kurang 1 jam
smseco: DB      'ATE0'                  ; instruksi no-echo
smsfmt: DB      'AT+CNMI=1,1,0,0,1'     ; display incoming sms
smsred: DB      'AT+CMGR=1'             ; instruksi baca sms memory 1
smshps: DB      'AT+CMGD=1'             ; instruksi hapus sms memory 1
smskrm: DB      'AT+CMGS=38'            ; instruksi kirim pdu 38 oktet
nohpku: DB      '261853481503F4'        ; no HP tujuan
comnd0: DB      '4D10F43D05'            ; M PoS (command posisi)
comnd1: DB      '4DD0D36C04'            ; M OfF (command off mesin)
comnd2: DB      '4DD0D3ED04'            ; M OnN (command on mesin)
comnd3: DB      '4DD0521E03'            ; M Kr1 (command kurang 1 jam)
comnd4: DB      '4DD0522E03'            ; M Kr2 (command kurang 2 jam)
comnd5: DB      '4DD0523E03'            ; M Kr3 (command kurang 3 jam)
dtapos: DB      '0001000D91261853481503F400001BD0E734394D8240'
angka:  DB      '0123456789ABCDEF'      ; karakter angka
kosong: DB      '                '      ; karakter kosong
;
kr_pdu: mov     DPTR,#angka             ;\
        mov     R1,Ch10                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch11                 ; |
        lcall   cchpdu                  ; |
```

```
        mov     R1,Ch12                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch13                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch14                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch15                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch16                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch17                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch18                 ; |
        lcall   cchpdu                  ; | cacah pdu
        mov     R1,Ch19                 ; | kirim pdu
        lcall   cchpdu                  ; |
        mov     R1,Ch1A                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch1B                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch1C                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch1D                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch1E                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch1F                 ; |
        lcall   cchpdu                  ; |
        mov     R1,Ch20                 ; |
        lcall   cchpdu                  ; |
        mov     A,#26                   ; | character 26 = CTRL+Z
        lcall   kr_srl                  ; | kirim
        ret                             ;/
;
cchpdu: mov     A,R1                    ;\
        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,R1                    ; |
        anl     A,#00Fh                 ; |
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        ret                             ;/
;
jd_pdu: mov     A,Ch11                  ;\
        RR      A                       ; | data1 geser 7x
        anl     A,#080h                 ; | and dg 1000 0000
        orl     A,Ch10                  ; | or dg data0
        mov     Ch10,A                  ;/
;
        mov     A,Ch11                  ;\
        anl     A,#11111110b            ; |
        RR      A                       ; | data1 hilangkan bit0
        mov     Bufr,A                  ; | data1 geser kanan 1x
        mov     A,Ch12                  ; | simpan sementara
        RR      A                       ; | and dg 1100 0000
        RR      A                       ; | or dg bufr
        anl     A,#0C0h                 ; |
        orl     A,Bufr                  ; |
        mov     Ch11,A                  ;/
;
        mov     A,Ch12                  ;\
        anl     A,#11111100b            ; |
        RR      A                       ; |
        RR      A                       ; | data1 hilangkan bit0 & bit1
        mov     Bufr,A                  ; | data1 geser kanan 2x
        mov     A,Ch13                  ; | simpan sementara
        RR      A                       ; | and dg 1110 0000
        RR      A                       ; | or dg bufr
        RR      A                       ; |
        anl     A,#0E0h                 ; |
        orl     A,Bufr                  ; |
        mov     Ch12,A                  ;/
```

```
;
        mov     A,Ch13
        anl     A,#11111000b
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch14
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F0h
        orl     A,Bufr
        mov     Ch13,A
;
        mov     A,Ch14
        anl     A,#11110000b
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch15
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F8h
        orl     A,Bufr
        mov     Ch14,A
;
        mov     A,Ch15
        anl     A,#11100000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch16
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FCh
        orl     A,Bufr
        mov     Ch15,A
;
        mov     A,Ch16
        anl     A,#11000000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch17
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FEh
        orl     A,Bufr
        mov     Ch16,A
;
        mov     A,Ch19
        RR      A
        anl     A,#080h
        orl     A,Ch18
```

```
        mov     Ch17,A
;
        mov     A,Ch19
        anl     A,#11111110b
        RR      A
        mov     Bufr,A
        mov     A,Ch1A
        RR      A
        RR      A
        anl     A,#0C0h
        orl     A,Bufr
        mov     Ch18,A
;
        mov     A,Ch1A
        anl     A,#11111100b
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch1B
        RR      A
        RR      A
        RR      A
        anl     A,#0E0h
        orl     A,Bufr
        mov     Ch19,A
;
        mov     A,Ch1B
        anl     A,#11111000b
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch1C
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F0h
        orl     A,Bufr
        mov     Ch1A,A
;
        mov     A,Ch1C
        anl     A,#11110000b
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch1D
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0F8h
        orl     A,Bufr
        mov     Ch1B,A
;
        mov     A,Ch1D
        anl     A,#11100000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch1E
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FCh
        orl     A,Bufr
        mov     Ch1C,A
;
```

```
        mov     A,Ch1E
        anl     A,#11000000b
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch1F
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        RR      A
        anl     A,#0FEh
        orl     A,Bufr
        mov     Ch1D,A
;
        mov     A,Ch21
        RR      A
        anl     A,#080h
        orl     A,Ch20
        mov     Ch1E,A
;
        mov     A,Ch21
        anl     A,#11111110b
        RR      A
        mov     Bufr,A
        mov     A,Ch22
        RR      A
        RR      A
        anl     A,#0C0h
        orl     A,Bufr
        mov     Ch1F,A
;
        mov     A,Ch22
        anl     A,#11111100b
        RR      A
        RR      A
        mov     Bufr,A
        mov     A,Ch23
        RR      A
        RR      A
        RR      A
        anl     A,#0E0h
        orl     A,Bufr
        mov     Ch20,A
;
        ret
;
        end
```

```
interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, DBCtrls, DB, ADODB, SMSComp, Grids, DBGrids, ExtCtrls,
  Mask,clipbrd, jpeg, ComCtrls;

type
  TForm2 = class(TForm)
    OxygenSMS1: TOxygenSMS;
    DataSource1: TDataSource;
    ADOQuery1: TADOQuery;
    DBGrid1: TDBGrid;
    Button1: TButton;
    Button2: TButton;
    Button3: TButton;
    Button4: TButton;
    ADOQuery1No_Plat: TWideStringField;
    ADOQuery1No_Hp: TWideStringField;
    ADOQuery1Nama: TWideStringField;
    ADOQuery1Tujuan: TWideStringField;
    ADOQuery1jam_selesai: TWideStringField;
    ADOQuery1Tgl_selesai: TDateTimeField;
    DBEdit1: TDBEdit;
    Edit2: TEdit;
    ADOQuery2: TADOQuery;
    ADOQuery2no_plat: TWideStringField;
    DBText5: TDBText;
    Label7: TLabel;
    Label8: TLabel;
    Button5: TButton;
    Image1: TImage;
    ComboBox1: TComboBox;
    Label2: TLabel;
    Button7: TButton;
    Label1: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    Bevel1: TBevel;
    Bevel2: TBevel;
    Bevel3: TBevel;
    Bevel4: TBevel;
    DataSource2: TDataSource;
    Bevel5: TBevel;
    Bevel6: TBevel;
    Bevel7: TBevel;
    Bevel8: TBevel;
    Edit3: TEdit;
    Label5: TLabel;
```

```
    ComboBox2: TComboBox;
    procedure FormShow(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure Button3Click(Sender: TObject);
    procedure Button4Click(Sender: TObject);
    procedure Button2Click(Sender: TObject);
    procedure OxygenSMS1SMSMessageReceived(Index: Integer; Time: TDateTime;
      Text, Send: String; Pict: TBitmap);
    procedure Button5Click(Sender: TObject);
    procedure ComboBox1Change(Sender: TObject);
    procedure Button7Click(Sender: TObject);
    procedure Timer1Timer(Sender: TObject);

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

var
  Form2: TForm2;

implementation

uses Database, snckey32, F_Log, StrUtils, F_Main;

{$R *.dfm}


procedure TForm2.FormShow(Sender: TObject);
begin
  ADOQuery1.Close;
  ADOQuery1.Open;
  ADOQuery2.Close;
  ADOQuery2.Open;
end;

procedure TForm2.Button1Click(Sender: TObject);
begin
if form1.Label3.Caption='Connected' then
  begin
  if messagedlg('Lacak Posisi '+DBEdit2.Text,mtconfirmation,[mbyes,mbno],0)=mryes  then
  begin
  Form2.OxygenSMS1.SendSMSMessage(dbedit1.Text,'M PoS',167,true,false,nil);
  Timer1.Enabled:=true;
  end;
  end
else
  ShowMessage('Handphone Belum Terhubung');
end;
```

```
procedure TForm2.Button3Click(Sender: TObject);
begin
  if messagedlg('WARNING Anda Akan Menonaktifkan Mobil '+DBEdit2.Text+' Apakah Anda Yakin',mtconfirmation,[mbyes,mbno],0)=mr
  begin
  Form2.OxygenSMS1.SendSMSMessage(
  dbedit1.Text,'M OfF',167,true,false,nil);
  Timer1.Enabled:=true;
  end;
end;

procedure TForm2.Button4Click(Sender: TObject);
begin
  if messagedlg('Anda Akan Mengaktifkan Mobil '+DBEdit2.Text+' Apakah Anda Yakin',mtconfirmation,[mbyes,mbno],0)=mryes  then
  begin
  Form2.OxygenSMS1.SendSMSMessage(
  dbedit1.Text,'M OnN',167,true,false,nil);
  Timer1.Enabled:=true;
  end;
end;

procedure TForm2.Button2Click(Sender: TObject);
begin
IF ComboBox2.ItemIndex=0 then
  begin
    if messagedlg('Kirim Pesan Ke '+DBEdit2.Text+' Waktu Sisa 1 Jam',mtconfirmation,[mbyes,mbno],0)=mryes  then
    begin
    Form2.OxygenSMS1.SendSMSMessage(
    dbedit1.Text,'M Kr3',167,true,false,nil);
    Timer1.Enabled:=true;
    end;
  end;
if ComboBox2.ItemIndex=1 then
  begin
    if messagedlg('Kirim Pesan Ke '+DBEdit2.Text+' Waktu Sisa 2 Jam',mtconfirmation,[mbyes,mbno],0)=mryes  then
    begin
    Form2.OxygenSMS1.SendSMSMessage(
    dbedit1.Text,'M Kr2',167,true,false,nil);
    Timer1.Enabled:=true;
    end;
  end;
if ComboBox2.ItemIndex=2 then
  begin
    if messagedlg('Kirim Pesan Ke '+DBEdit2.Text+' Waktu Sisa 3 Jam',mtconfirmation,[mbyes,mbno],0)=mryes  then
    begin
    Form2.OxygenSMS1.SendSMSMessage(
    dbedit1.Text,'M Kr1',167,true,false,nil);
    Timer1.Enabled:=true;
    end;
  end;
end;
```

```
procedure TForm2.OxygenSMS1SMSMessageReceived(Index: Integer;
  Time: TDateTime; Text, Send: String; Pict: TBitmap);
begin
  OxygenSMS1.AutoDeleteMessages:=false;
  ADOQuery2.SQL.Clear;
  ADOQuery2.SQL.Add(' select No_Plat from T_Mobil ');
  ADOQuery2.SQL.Add(' where No_hp = :hp ');
  ADOQuery2.Parameters.ParamByName('hp').Value:=send;
  ADOQuery2.Active:=true;
  ADOQuery2.RecordCount;
  if ADOQuery2.RecordCount>0 then
    begin
    edit2.Clear;
    edit3.Clear;
    edit2.Text:=text;
    edit2.Text:=MidStr(edit2.Text,9,1)+' '+MidStr(edit2.Text,10,11)+' '+MidStr(edit2.Text,21,7);
    edit3.Text:=TimeToStr(time);
    DataModule1.ADOLog.Append;
    DataModule1.ADOLogNo_Plat.Value:=DBText5.Caption;
    DataModule1.ADOLogTanggal.Value:=DateToStr (time);
    DataModule1.ADOLogJam.Value:=TimeToStr(time);
    DataModule1.ADOLogPosisi.Value:=edit2.Text;
    DataModule1.ADOLog.Post;
    DataModule1.ADOLog.Close;
    DataModule1.ADOLog.Open;
    end;
  if ADOQuery2.RecordCount=0 then
    begin
    ShowMessage('SMS Bukan Dari Client');
    OxygenSMS1.AutoDeleteMessages:=true;
    end;
end;

procedure TForm2.Button5Click(Sender: TObject);
begin
  if edit2.Text='' then
  begin
    ShowMessage('Kordinat Tidak Ditemukan');
  end;
  if edit2.Text<>'' then
  begin
    edit2.SelectAll;
    Edit2.CopyToClipboard;
    OxygenSMS1.AutoDeleteMessages:=true;
    AppActivate('Google Earth');
    SendKeys('%f {Escape}{tab}{Delete}^v~',false);
  end;
end;

procedure TForm2.ComboBox1Change(Sender: TObject);
```

```
begin
  ADOQuery1.SQL.Clear;
  ADOQuery1.SQL.Add (' select T_Mobil.No_Plat,T_Mobil.No_Hp,T_Pelanggan.Nama,T_Sewa.Tujuan,T_sewa.jam_selesai,T_sewa.Tgl_sel
  ADOQuery1.SQL.Add (' from ( T_mobil inner join T_Sewa ON T_Mobil.No_Plat = T_sewa.No_Plat ) ');
  ADOQuery1.SQL.Add (' inner join T_pelanggan ON T_Sewa.No_ID = T_Pelanggan.No_ID ');
  ADOQuery1.SQL.Add (' where T_mobil.status = "Disewa" ');
  ADOQuery1.Active:=true;
  DataSource1.DataSet:=ADOQuery1;
end;
if ComboBox1.ItemIndex=1 then
begin
  DataSource1.DataSet:=DataModule1.ADOmobil;
end;
end;

procedure TForm2.Button7Click(Sender: TObject);
begin
  form7.Show;
end;

procedure TForm2.Timer1Timer(Sender: TObject);
begin
  if ProgressBar1.Position < ProgressBar1.Max then
  begin
    ProgressBar1.Step:=ProgressBar1.Max div 10;
    ProgressBar1.StepIt;
  end
else
  begin
    timer1.Enabled:=false;
    ShowMessage('SMS Sudah Dikirim');
    ProgressBar1.Position:=0;
  end;
end;

end.
```

- Di Jembatan Kembar Soekarno Hatta
- Pertigaan Dinoyo

Tabel 4.5. Hasil Pengujian Penentuan Posisi

Posisi	Koordinat
Didepan Perumahan Bumi Palapa	7°55'.88 S, 112°36'.66 E
Didepan RRI	7°56'.18 S, 112°37'.13 E
Di Tugu MIG	7°56'.21 S, 112°37'.58 E
Di Jembatan Kembar Soekarno Hatta	7°56'.93 S, 112°36'.98 E
Pertigaan Dinoyo	7°56'.59 S, 112°36'.61 E

Gambar 4.11. Screenshoot GoogleEarth Pada koordinat 7°55'.88 S, 112°36'.66 E

Gambar 4.12 Screenshoot Google Earth Pada koordinat 7°56'.18 S, 112°37'.13 E

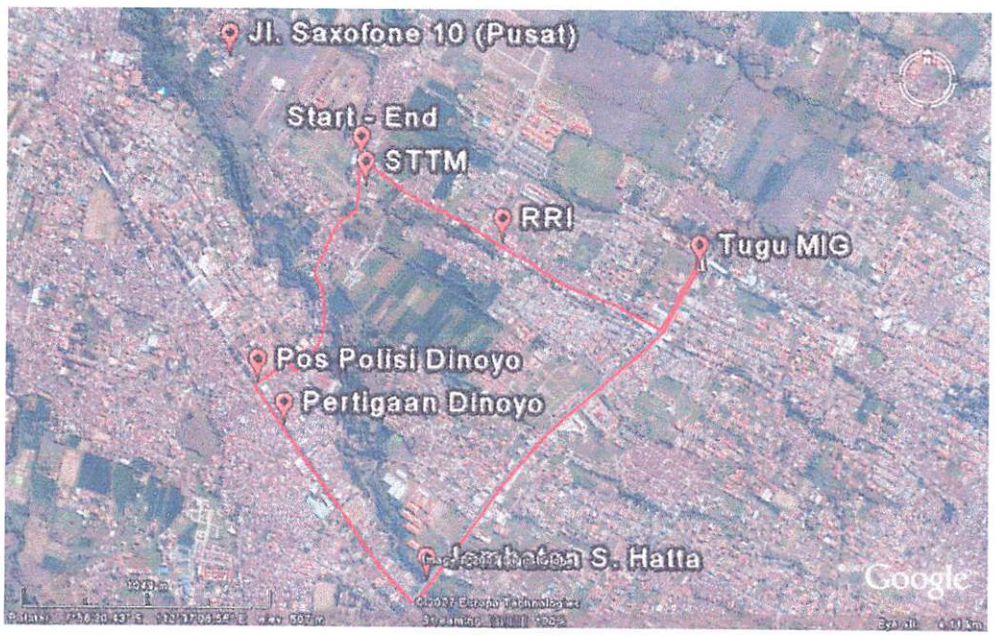
Gambar 4.13 Screenshoot Google Earth Pada koordinat 7°56'.21 S, 112°37'.58 E

Gambar 4.14 Screenshoot Google Earth Pada koordinat 7°56'.93 S, 112°36'.98 E

Gambar 4.15 Screenshoot Google Earth Pada koordinat 7°56'.59 S, 112°36'.61 E

4.5.1.2. Pengujian Kedua

Pengujian kedua dilakukan mulai dari Simpang Empat STTM sampai dengan Dinoyo kemudian kembali ke tempat semula. Pengujian dilakukan mulai pukul 15.00 s/d selesai dengan kecepatan rata-rata 20-40 Km/Jam. Pengambilan koordinat dilakukan pada saat mobil bergerak dengan cara mengirim SMS permintaan lokasi dari Pusat dengan interval pengiriman ± 2 menit sekali. Pengujian dilakukan sebanyak 2 Putaran.



Gambar 4.16. Rute Pengujian Kedua

Tabel 4.6. Hasil Pengujian 2 Putaran Pertama

Waktu Pengiriman	Waktu Terima	Selisih (detik)	Koordinat
3:35:54	3:36:07	13	7 55.98 S, 112 36.80 E
3:38:04	3:38:17	13	7 56.16 S, 112 37.09 E

3:40:06	3:40:18	12	7 56.35 S, 112 37.49 E
3:42:06	3:42:17	11	7 56.44 S, 112 37.41 E
3:44:05	3:44:16	11	7 56.80 S, 112 37.06 E
3:46:10	3:46:22	12	7 56.85 S, 112 36.80 E
3:48:10	3:48:20	10	7 56.51 S, 112 36.56 E
3:50:02	3:50:15	13	7 56.39 S, 112 36.70 E
3:52:04	3:52:20	16	7 56.08 S, 112 36.79 E
Rata-rata		12,33 Detik	

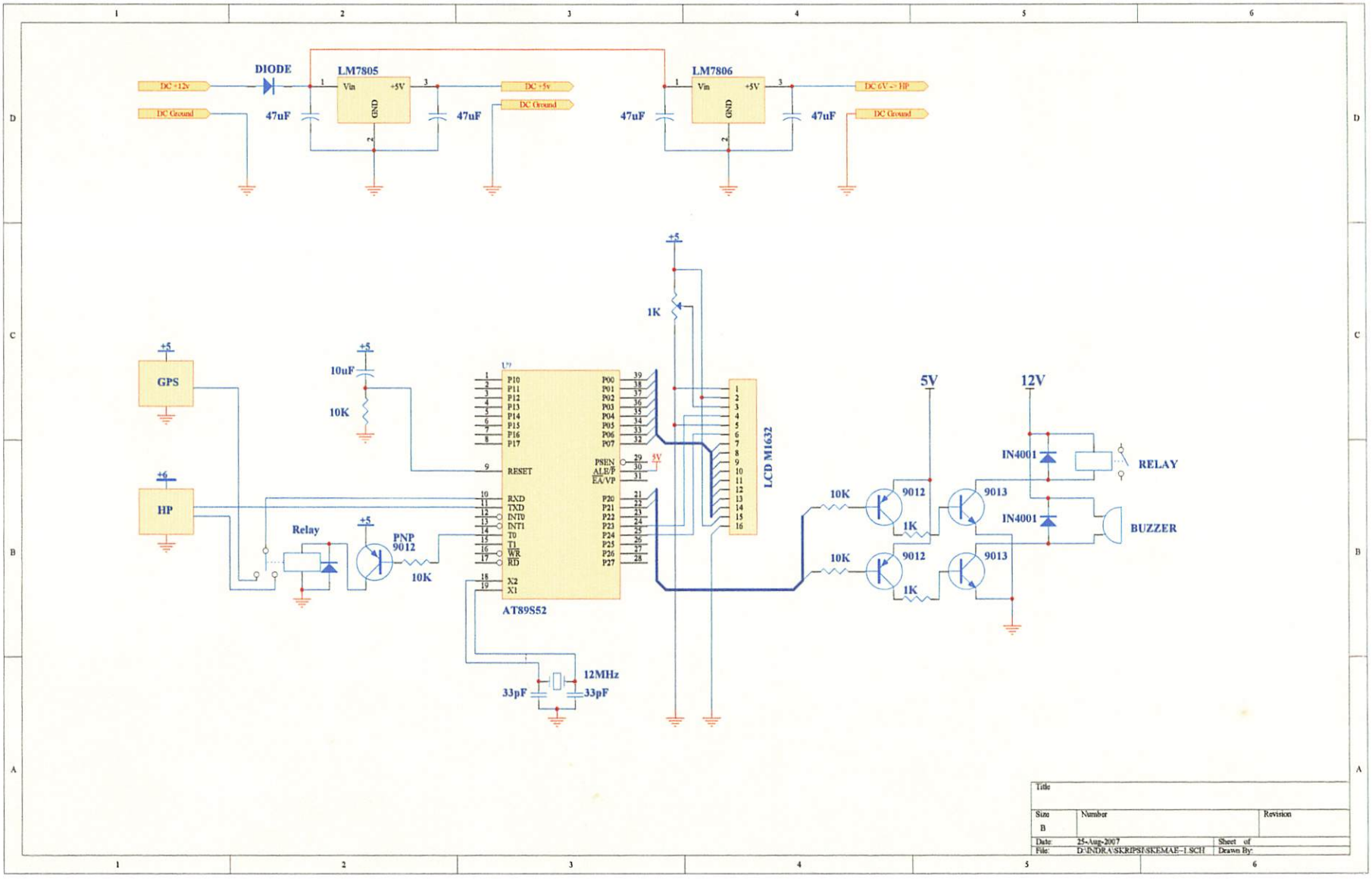
Gambar 4.17. Hasil Pengujian 2 Putaran Pertama

Tabel 4.7. Hasil Pengujian 2 Putaran Kedua

Waktu Pengiriman	Waktu Terima	Selisih (Detik)	Koordinat
3:54:02	3:54:12	10	7 55.98 S, 112 36.80 E
3:56:06	3:56:18	12	7 56.13 S, 112 37.04 E

3:58:17	3:58:19	12	7 56.33 S, 112 37.43 E
4:00:08	4:00:21	13	7 56.37 S, 112 37.49 E
4:02:10	4:02:22	12	7 56.75 S, 112 37.10 E
4:04:08	4:04:18	10	7 56.88 S, 112 36.82 E
4:06:11	4:06:21	10	7 56.58 S, 112 36.61 E
4:08:04	4:08:17	13	7 56.43 S, 112 36.68 E
4:10:08	4:10:20	12	7 56.10 S, 112 36.78 E
Rata-rata		11,55 Detik	
Total Rata-rata		11,94 Detik	

Gambar 4.18. Hasil Pengujian 2 Putaran Kedua


DIODE
LM7805
Vin
+5V
GND
DC +12v
DC Ground
47uF
47uF
DC +5v
DC Ground
LM7806
Vin
+5V
GND
47uF
47uF
DC 6V - HP
DC Ground
+5
1K
GPS
+5
10uF
10K
HP
+6
Relay
+5
PNP
9012
10K
U?
P10
P11
P12
P13
P14
P15
P16
P17
RESET
RXD
TXD
INT0
INT1
T0
T1
WR
RD
X2
X1
P00
P01
P02
P03
P04
P05
P06
P07
PSEN
ALE/P
EA/VP
P20
P21
P22
P23
P24
P25
P26
P27
5V
AT89S52
12MHz
33pF
33pF
LCD M1632
5V
12V
10K
9012
9013
1K
IN4001
RELAY
IN4001
BUZZER
10K
9012
9013
1K
Title
Size
B
Number
Revision
Date:
25-Aug-2007
Sheet of
File:
D:\INDRA\SKRIPSI\SKEMAE~1.SCH
Drawn By: