PERANCANGAN DAN PEMBUATAN PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

SKRIPSI

Disusun oleh :

Andri Dwi Arianto

NIM. 0912902

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ELEKTRONIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2011

PERANCANGAN DAN PEMBUATAN
PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR
PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

SKRIPSI

Disusun oleh :

Andri Dwi Arfianto
NIM. 0912302

JURUSAN TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ELEKTRONIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2011

HALAMAN JUDUL

PERANCANGAN DAN PEMBUATAN
PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR
PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

SKRIPSI

Disusun oleh :

Andri Dwi Arianto

NIM. 0912902

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK ELEKTRONIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2011

LEMBAR PERSETUJUAN

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

SKRIPSI

Disusun dan diajukan untuk melengkapi dan memenuhi persyaratan
guna mencapai gelar Sarjana Teknik

Disusun oleh :
Andri Dwi Arianto
NIM. 0912902

Mengetahui,
Ketua Program Studi Teknik Elektro S-1

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

Diperiksa dan Disetujui,

Dosen Pembimbing I	Dosen Pembimbing II
I Komang Somawirata, ST, MT	**Irmalia Suryani F. ST, MT**
NIP. Y. 1030100361	**NIP. P. 1030000365**

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

LEMBAR
BERITA ACARA
UJIAN SKRIPSI

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	: Andri Dwi Arianto
NIM	: 09.12.902
Jurusan	: Teknik Elektro S1
Konsentrasi	: Teknik Elektronik
Judul	: PERANCANGAN DAN PEMBUATAN PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

Dipertahankan dihadapan Majelis Penguji Skripsi Jenjang Strata Satu (S1) pada :

Hari	: Kamis
Tanggal	: 17 Februari 2011
Dengan Nilai	: 73,45 (B+)

Panitia Ujian Skripsi :

Ketua Majelis Penguji

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

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y. 1030800417

Anggota Penguji :

Penguji Pertama

Sotyohadi, ST
NIP.Y. 1039700309

Penguji Kedua

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y. 1030800417

LEMBAR PERSEMBAHAN

بسم الله الرحمن الرحيم

" Dengan Menyebut Nama Allah Yang Maha Pengasih Lagi Maha Penyayang"

Sripsi Meniko Kulo Persembahaken Dateng :

Gusti ALLAH

Ingkang Sampun Maringi Barokah Dateng Kulo Ngantos Dados Mekaten Meniko.

Lan Hasil Kuliah Niki Ikhlas Kulo Persembahaken Dateng Panjenengan Gusti.

Kulo Pasrah Marang Panjenengan Gusti, Kathah Lepating Kulo Nanging

Luwih Kathah Pangapuraning Panjenengan

Damel Kawulomu Niki.

Sholawat Kalian Salam Boten Kesupen Kulo AturakenDateng Junjungan Kulo Nabi MUHAMMAD SAW. Mugi-Mugi Kulo Saget Tepang Kaleh Panjenengan Kanjeng. Amiin.nn..

Kagem Ibu Kalian Almr. Ayah

Andri Mboten Saget Ngewangsulaken Utang Budi Lan Jasa Ingkang Penjenengan Kekaleh Paringaken Dateng Kang Putro, Nanging Kulo Namung Saget Ngaturaken Matur Suwun Ingkang Mboten Kekiro,Lan Nyuwun Gunging Pangaksami Dateng Kekalih. Mugi-Mugi Ibu Panjang Umur, Sehat Sekalu Saget Ngemong Putu Ngantos Ageng. Mugi-Mugi Almr. Ayah Diparingi Ketentreman Lan Damai Dateng Alam Kubur. Lan Panjenengan Kekalih Angsal Barokah Ndumya Lan Akhirod Ingkang Mboten Kekiro Saking Gusti ALLAH. Amiiin.nn...

Kagem Mbah Putri Kalian Almr. Mbah Kakung

Matur Suwun Engkang Kathah Dateng Kekaleh, Sampun Nyukani Kasih Sayange Dateng Kula, Putu Jaler Sampek Dinten Meniko. Mugi-Mugi Mbah Putri Diparingi Panjang Umur Lan Kesehatan. Mugi-Mugi Mbah Kakung Diparingi Kedamaian Dateng Alam Kubur.

Kagem Sedanten Dosen Elektronika Lan Karyawan ITN Malang

Semoga ALLAH membalas Kebaikan Mereka Amin.nn..

Yo Ra' Lali Aku Ngucapno Matur Suwun Gawé Masku Seng Lan Adik Kandungku
Seng Ngganteng Lan Ayuuu Dewé, Sampun Maringi Semangat Lan Kasih Sayange.
Suwun Yo.oo..

Ra' Lali Gawé Tunanganku Tie.Tie..
Sayangku Lan Hatiku Mong Kanggo Sampean. Aku Matur Suwunuun Lan Suwunuun Banget
Anggéné Sampeyan Ndukung Gawene Aku Ngrampungno Skripsi Lan Kuliah Iki.. Hi.Hi.ii.. Suwun Pisan Yo.oo..

Gawe Arek-Arek Lab.Rangkaian Listrik dan Pengukuran
Suwuuun Anggéné Nyemangati Aku Pas Nggarap Skripsi Lan Katé Kompré.
Kompak Yo Rek Lan Terus Semangat Di Lab. "RALIPE"

Gawe Arek-Arek Transferan Elka. Angkatan 2009
Iki Koncoku Séng Khutuk-Khutuk He.He..He...He....
Suwun Banget Dadi Konco Séng Iso Mbangun Pikiranku Pas Ruwet. Kuliahe Ndang
Dimarekno Bos.. Semangat.Semangatt..

Naah Gawé Bocah-Bocah Kabeh Séng Ra Iso Ta' Sebutno Siji-Sijine
Ojo Putus Semangat, Urip Iku Mong Sepisan, Kudu Dilakoni Nganti Kabeh Rampung.
Ilingo Kabéh Marang Gusti ALLAH, Yo Iku Kunciné Uripmu Néng Ndunyo.
Aku Ra Iso Mbales Jasa-Jasamu Kabéh Cah, Aku Mong Iso Njaluk Sepuro Nék Aku Ono
Salah Séng Sengojo Opo Ra Sengojo, Yooo Mugo-Mugo Gusti Nglimpahi Awakmu
Rejekin-Ĕ Lan Kasempatan Ngrampungno Kuliah..Amiiin.nn.

ANDRI DWI ARIANTO

ABSTRACT

ABSTRAK

PERANCANGAN DAN PEMBUATAN PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

Andri Dwi Arianto
Fakultas Teknologi Industri, Jurusan Teknik Elektro S1, Institut Teknologi Nasional Malang
Email : Andrye14_SweetColl@yahoo.co.id

Abstrak

Komunikasi data pada pengontrol pintu air bendungan yang diaplikasikan oleh beberapa bendungan, masih menggunakan cara konvensional. Pada alat-alat tertentu yang mempunyai jarak sangat jauh dari sistem kontrol sangat riskan terjadi kesalahan pengiriman dan penerimaan data, hal itu diakibatkan oleh kerusakan pada instalasi kabel komunikasi. Sama halnya dengan menggunakan system komunikasi HT , akan tetapi sering terjadi kesalahan dalam pengiriman seperti rusaknya *receiver* dan *transmiter* pada pos penjagaan sampai kesalahan yang sering terjadi yaitu kesalahan manusia. Untuk meningkatkan efisiensi waktu, harga dan penggunaan, maka pada tugas akhir ini penulis membuat suatu alat pengontrol dan monitoring ketinggian level air pada suatu bendungan menggunakan *SMS gateway*. Alat ini memakai handphone sebagai alat pengirim dan penerima data ketinggian air bendungan menggunakan sms. Alat dirancang untuk mengirim data dan dapat melakukan buka tutup pintu air secara otomatis. Dengan adanya alat ini kesalahan yang terjadi pada cara-cara konvensional sebelumnya dapat diatasi.

Kata kunci : Handphone, Receiver, Transmiter, SMS Gateway.

Abstrack

Data communications on the control of dams water gates that are applied by several dams, are still using conventional means. In the tool - a specific tool that has a very far distance from the system control is very risky error occurs sending and receiving data, it is caused by damage to the installation of communication cables. Similarly, using the communication system of HT, but it often happened mistake in delivery such as damaged receivers and transmitters at the checkpoint until the frequent mistakes are human error. To improve the efficiency of time, cost and usage, so in this thesis the author makes a device height control equipment and monitoring water levels at a dam using SMS gateway. This tool uses mobile phones as a means of sending and receiving data using sms dam water levels. Equipment designed to transmit data and can perform opening and closing of water gates automatically. With the existence of this tool is expected that the error occurred on the way - the previous conventional method can be overcome.

Keyword : Handphone, Receiver, Transmiter, SMS Gateway.

KATA
PENGANTAR

KATA PENGANTAR

Atas Berkat Rahmat Allah Yang Maha Kuasa, sehingga penulis dapat menyelesaikan laporan Skripsi dengan judul :

PERANCANGAN DAN PEMBUATAN
PENGONTROL DAN MONITORING KETINGGIAN LEVEL AIR PADA SUATU BENDUNGAN MENGGUNAKAN SMS GATEWAY

Pembuatan Skripsi ini disusun guna memenuhi syarat akhir kelulusan pendidikan jenjang Strata-1 di Institut Teknologi Nasional Malang. Laporan Skripsi ini merupakan tanggung jawab tertulis atas ilmu pengetahuan yang didapat selama penyusun mengikuti kuliah.

Atas terselesaikannya Skripsi ini, penulis mengucapkan terima kasih kepada :

- Bapak Prof. Dr. Eng Ir. Abraham Lomi, MSEE. selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. Sidik Noetjahjono, MT selaku Dekan Fakultas Teknologi Industri Institut Teknologi Nasional Malang
- Bapak Ir. Yusuf Ismail Nahkoda, MT selaku Ketua Jurusan Teknik Elektro S1 Konsentrasi Teknik Elektronika.
- Bapak Dr. Eng, Aryuanto Soetedjo ST, MT selaku Sekretaris Jurusan Teknik Elektro S1 Konsentrasi Teknik Elektronika.
- Bapak I Komang Somawirata, ST, MT dan Ibu Irmalia Suryani F. ST, MT selaku Dosen Pembimbing I dan II yang telah memberikan bimbingan, pengarahan, serta ilmu-ilmu yang sangat berharga sehingga skripsi ini dapat terselesaikan.

Penulis menyadari bahwa laporan ini masih banyak yang perlu disempurnakan. Oleh sebab itu kritik dan saran yang membangun sangat diharapkan.

Malang, Pebruari 2011

Penulis

DAFTAR
ISI

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR GAMBAR

Halaman

DARTAR
TABEL

DAFTAR TABEL

Halaman

DAFTAR GRAFIK

DAFTAR GRAFIK

Halaman

DAFTAR
LAMPIRAN

DAFTAR LAMPIRAN

Halaman

BAB I
PENDAHULUAN

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Pada saat ini pemakaian alat untuk mendeteksi ketinggian air masih menggunakan cara konvensional. Jadi terdapat sensor yang mendeteksi ketinggian air pada pos penjagaan pada pintu air, kemudian jika terjadi peluapan air secara besar-besaran sensor mengirimkan data melalui frekuensi radio menggunakan HT, akan tetapi sering terjadi kesalahan dalam pengiriman seperti rusaknya *receiver* dan *transmiter* pada pos penjagaan sampai kesalahan yang sering terjadi yaitu *human error*. Setelah data diterima operator atau penjaga pintu, akan memberitahukan kepada Dinas Pekerjaan Umum setempat untuk selanjutnya memberikan informasi kepada warga sekitar. Sebenarnya aplikasi ini sudah dapat membantu jika penjaga pintu air bisa menggunakan fasilitas tersebut dan dapat merawat dengan sebaik-baiknya. Kenyataanya fasilitas pengiriman data ketinggian air secara berkala pada sungai dan bendungan sudah tidak dapat digunakan secara maksimal atau sudah rusak. Itu terjadi karena besarnya biaya untuk inspeksi atau perawatan alat tersebut. Dan juga kurang pahamnya cara kerja keseluruhan para operator baru menggunakan fasilitas tersebut. Tapi sistem menggunakan *frekuensi radio* mempunyai beberapa kelemahan, sistem tersebut membutuhkan aliran listrik besar dan antena tinggi, yang sewaktu-waktu dapat rusak karena tersambar petir.

Sebagian besar banjir disebabkan karena berbagai faktor, seperti : penebangan hutan secara liar sehingga mengakibatkan hutan gundul, membuang sampah tidak pada tempatnya, tidak ada langkah penghijauan dan lain-lain. Terdapat beberapa penelitian pada skripsi sebelumnya yang berkaitan dalam mengantisipasi dan penginformasian banjir.

Arief Budi Setiawan (03.17.101) dalam penelitiannya dalam judul "*Perancangan dan Pembuatan Sistem Peringatan Dini Ketinggian Air Sungai Via SMS Berbasis Mikrokontroler AT89AS51*", menetapkan sensor mendeteksi ketinggian air dan mendeteksi setiap adanya perubahan ketinggian air sungai. Sensor yang digunakan adalah sensor VR (*Variable Resistor*) yang jenisnya potensio meter. Dalam penelitian tersebut informasi yang disampaikan melalui SMS dari HP sangat efisien, tetapi informasi yang disampaikan hanya ketinggian air saja dan tidak ada tindakan lain dalam mengantisipasi ketinggian level air. Penelitian diatas dikembangkan oleh Nanang Imam Setya Wahyudi (03.17.048) dengan judul "*Perancangan dan Pembuatan Sistem*

Monitoring ketinggian Air Sungai Melalui WEB Berbasis Mikrokontroler ATMEGA 16", menetapkan sensor pelampung bengsin yang di dalamnya ada potensio meter. Cara kerjanya seperti kita akan browsing di internet dengan memasukkan alamat yang dituju dan mencangkup internet LAN lokal saja. Penelitian diatas memerlukan sambungan internet yang belum tentu kita dapatkan dengan mudah dan cepat.

Melalui "Perancangan dan Pembuatan Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway". Penulis dapat meminimalisir kelemahan tersebut, dengan menggunakan SMS sebagai *interface* data yang akan dikirimkan lebih murah, *efisien* dan *real time*. Juga dengan perubahan level air maka akan mempengaruhi buka tutup pintu dam di bendungan sungai.

1.2 Rumusan Masalah

Berdasarkan latar belakang dari masalah tersebut di atas, maka dapat dirumuskan beberapa permasalahan yaitu :

a. Bagaimana cara membangun *system* "Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway" ?
b. Bagaimana desain mekanik "Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway" yang tepat ?
c. Bagaimana proses komunikasi data antara *handphone* ke mikrokontroler ?
d. Bagaimana proses komunikasi data antara mikrokontroler ke *handphone* ?
e. Bagaimana proses pengiriman data ketinggian air ke *handphone user* secara berkala setiap keadaan sensor berubah ?
f. Bagaimana tanggapan (*respon*) motor penggerak pintu air terhadap sensor Ketinggian ?

1.3 Batasan Masalah

Untuk memfokuskan pembahasan dalam tugas akhir ini, dibatasi oleh beberapa hal yaitu :

a. Jenis Hp Siemens C35 atau M35.
b. Handphone hanya digunakan fasilitas SMSnya saja.
c. Gangguan layanan operator dalam pengiriman informasi berupa SMS diabaikan.
d. Pembahasan ditekankan pada buka tutup pintu air dan ketinggian level air.
e. Tidak membahas tentang proses pengiriman dan penerimaan SMS.
f. Sistem pengolah data menggunakan mikrokontroler AT89S52.

g. Tidak membahas tentang cara kerja *provider selular.*

h. Sistem yang dibuat berupa *prototype.*

i. Tidak membahas Catu Daya.

1.4 Tujuan Penalitian

Tujuan yang ingin dibuat untuk skripsi ini adalah :

a. Mengetahui cara membangun system "Rancang Bangun Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway".

b. Mengetahui konsep dan proses komunikasi data *handphone* ke mikrokontroler.

c. Mengetahui konsep dan proses komunikasi data mikrokontroler ke *handphone.*

d. Mengetahui proses pengiriman data ketinggian air ke *handphone user* secara berkala setiap keadaan sensor berubah.

e. Menganalisa respon *actuator* atau motor penggerak pintu air berdasarkan sensor.

f. Membuat simulator buka tutup pintu air menggunakan SMS yang *effisien.*

1.5 Metodologi Penulisan

Prosedur yang dilakukan untuk menyelesaikan skripsi ini dapat dijelaskan sebagai berikut :

a. Observasi Masalah dan Penentuan Judul

Proses awal dari penyusunan skripsi ini adalah observasi dari beberapa kendala yang dihadapi oleh operator atau penjaga pintu air sungai, waduk atau bendungan kemudian penulis menemukan gagasan yang bisa menjadi judul skripsi.

b. Studi Literatur

Studi literatur dilakukan untuk mempelajari dasar-dasar teori yang digunakan untuk :

- Sistem pengontrolan berbasis mikrokontroler AT89S52 pada proses operasi dari alat yang akan dirancang.
- *Driver* motor dan *sensor* ketinggian untuk pengontrolan *hardware.*
- Pengiriman data *sensor* yang menggunakan SMS *GateWay.*

c. Perancangan dan Pembuatan Alat (*Hardware*)

Langkah-langkah yang dilakukan dalam pembuatan alat ini :

- Pembuatan mekanik, mencari referensi bahan yang kuat dan *efisien.*

- Perancangan pada tiap-tiap bloknya, perhitungan dan penentuan komponen yang digunakan kemudian dilanjutkan dengan mendisain rangkaian dan lalu membuatnya.

  Rangkaian meliputi :

  - *Driver* motor DC, *driver sensor* ketinggian, mikrokontroler, modul pengiriman data menggunakan *Handphone* dan LCD untuk mengetahui keadaan tinggi air.

d. Uji Coba dan Analisa

Pengujian ini dilaksanakan secara bertahap seperti berikut :

- Pengujian pengiriman SMS data ketinggian air ke *user* secara otomatis.
- Pengujian respon *sensor* ketinggian.
- Pengujian pengiriman data menggunakan SMS *Gate Way.*
- Pengujian respon pintu air berdasarkan data *sensor.*
- Pengujian *system* secara keseluruhan.

e. Penyempurnaan

Penyempurnaan rangkaian yang tidak bekerja seperti yang diharapkan dan penyempurnaan terhadap *error* yang didapat pada proses pengujian.

f. Penyusunan Skripsi

Pembuatan dan penyusunan buku dari keseluruhan skripsi dengan sistematika yang telah ditemukan.

g. Presentasi Skripsi

Mempresentasikan hasil skripsi yang telah dibekukan sebagai syarat pengujian secara tertulis.

1.6 Sistematika Penulisan

Dalam penulisan skripsi ini terbagi dalam lima bab dengan sistematika sebagai berikut :

BAB I Pendahuluan

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

BAB II Landasan Teori

Membahas teori dasar penunjang, perancangan dan pembuatan alat.

BAB III Perancangan dan Pembuatan Alat

Membahas tentang perancangan alat baik perangkat keras maupun perangkat lunak, serta cara kerja blok diagram.

BAB IV Pengujian dan Analisa

Mencakup pembahasan tentang proses pengujian alat yang terdiri dari peralatan yang digunakan, langkah kerja dan analisa hasil pengujian.

BAB V Penutup

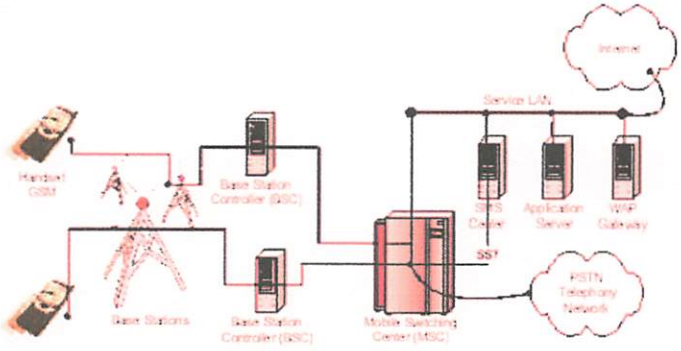
Penulis menarik kesimpulan dari apa yang telah diuraikan pada bab-bab sebelumnya dan mengemukakan saran saran yang mungkin akan bermanfaat bagi laporan akhir ini.

BAB II
LANDASAN TEORI

BAB II
LANDASAN TEORI

2.1 GSM (*Global System for Mobile Communication*)

GSM (*Global System for Mobile Communication*) merupakan sebuah sistem telepon bergerak generasi ke-2 yang dikembangkan di wilayah Eropa sejak tahun 1980-an. Sistem ini bekerja pada frekuensi 1800 MHz dan 1900 MHz. Jaringan GSM dapat dibagi menjadi 3 bagian besar. *Mobile Station* merupakan perangkat *handset* yang dibawa oleh pelanggan yang disebut *handphone*. *Subsystem base station* merupakan bagian sistem yang mengendalikan *radio link* dengan *mobile station* di dalam terdapat BSC (*Base Station Controller*) dan BTS (*Base Transceiver Station*). Subsistem jaringan merupakan bagian terpenting karena di dalamnya terdapat *Mobile Services Switching Center* (MSC) yang melakukan penyambungan komunikasi antara *mobile station* dan antara *mobile station* dengan *fixed phone*. MSC juga menangani operasi pengaturan mobilitas. *Mobile station* dan *base station subystem* berkomunikasi melalui *interface.*



Gambar 2.1 : *Jaringan GSM*

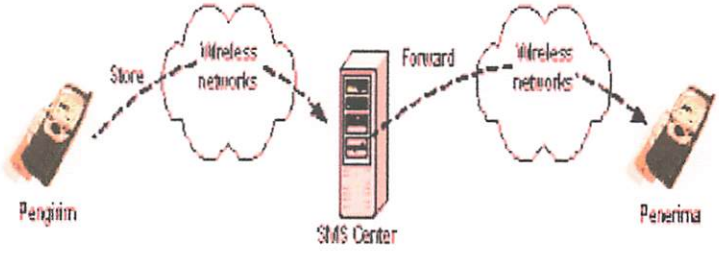
2.2 Mode Pengiriman dan Penerimaan SMS

Short Messaging Service (SMS) merupakan salah satu fitur dari GSM yang dikembangkan dan distandarisasi oleh ETSI. SMS merupakan salah satu layanan yang dimungkinkan *Handphone* bisa mengirimkan pesan pendek dalam format teks dari dan menuju *handphone*. Kapasitas karakter SMS adalah sebanyak 160 karakter *alpha-numeric*. Pada pengiriman pesan terdapat dua jenis *mobile*, yaitu *mobile terminated* (*handphone* penerima) dan *mobile originated* (*handphone* pengirim). Saat sebuah pesan terkirim, pesan tersebut diterima oleh *Short Message Service Center* (SMSC), yang setelah itu memberikan kepada *handphone* yang tepat (nomor tujuan). Untuk melakukan hal ini, SMSC mengirimkan *SMS request* kepada *Home Location Register* (HLR) untuk

mengetahui keberadaan pelanggan. Begitu HLR menerima *request* tersebut, HLR akan memberikan *respond* kepada SMSC berupa status *subscriber* :

- Aktif atau tidak aktif
- Dimana *subscriber* berada

Jika responnya adalah "tidak aktif", SMSC akan menyimpan pesan tersebut beberapa waktu, lalu SMSC akan memberikan informasi kepada pengirim "*Not Delivered*", saat *subscriber* mengakses *device*-nya, HLR mengirimkan SMS pemberitahuan kepada SMSC dan SMSC akan mencoba kembali pengiriman. Sistem ini kemudian menghubungi *handphone* dan jika *handphone* memberikan *respond*, pesan tersebut dikirimkan. SMSC menerima verifikasi bahwa pesan tersebut telah diterima oleh pelanggan dan mengategorikan pesan telah terkirim dan tidak lagi mencoba untuk melakukan pengiriman ulang. Kemudian SMSC akan memberikan informasi kepada pengirim "*delivered*". Gambar ini menunjukan mekanisme pengiriman SMS melalui SMSC.



Gambar 2.2 : *Mekanisme Pengiriman SMS Melalui SMSC*

Terdapat dua mode untuk pengiriman dan penerimaan SMS, yaitu mode teks dan mode PDU (*Protocol Data Unit*). Namun sistem mode teks tidak didukung oleh semua operator GSM maupun *handphone*. Untuk mengetahui mode yang digunakan pada operator dan *handphone* dapat dikirimkan AT*Command* "AT-CMGF=1" menggunakan program emulasi *hyperterminal*, jika hasilnya *error*, berarti *handphone* tersebut tidak didukung mode teks. Mode teks adalah format pesan dalam bentuk teks asli yang dituliskan pada saat akan mengirimkan pesan. Mode ini adalah cara termudah untuk mengirimkan pesan. Pada mode teks pesan yang dikirimkan tidak melakukan proses konversi melainkan sesuai dengan apa yang diketik dari keyboard.

2.3 Telephone Selular (*Handphone*)

Handphone merupakan suatu jenis perangkat telepon bergerak data komunikasi, sehingga memudahkan seseorang berkomunikasi dimanapun, kapanpun selama dalam cakupan suatu jaringan dari *SIM Card* yang digunakan. *Handphone* merupakan suatu perangkat dalam mengirim atau menerima data suara. Seiring dengan perkembangan teknologi dibidang *mobile system*, *handphone* tidak hanya dapat mengirim atau menerima data suara tetapi juga data karakter atau disebut dengan SMS (*Short Message Service*) . Dengan kecanggihan *Handphone* pada jaman sekarang maka *Handphone* juga dilengkapi dengan fasilitas lainnya. Seperti EMS (*Enhaced Message Service*), MMS (*Multimedia Message Service*) dan *polyphonic*. Dalam perancangan ini menggunakan *HP Siemens* tipe *M35* atau *C35* seperti pada gambar di bawah ini.

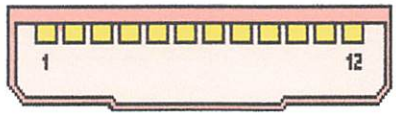
Gambar 2.3 : *HP Siemens M35 dan C35*

2.4 Kabel Data HP *Siemens*

HP Siemens dilengkapi dengan kabel data untuk melakukan komunikasi data *serial* dengan terminal lain. Kecepatan *transfer* data (*Baud Rate*) sebesar 19200. Kabel data *Siemens C35* juga cocok digunakan untuk *Siemens M35*. Untuk mengadakan komunikasi *serial*, pin-pin yang digunakan adalah :

Tabel 2.1 : *Kabel Data Siemens C45*

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



Gambar 2.4 : *Pin Out Siemens C45*

Tabel 2.2 : *Pin Out Handphone Siemens C45*

Pin	Nama	Fungsi	In / Out
1	GND	*Ground*	
2	*Self Service*	*Recognition / Control Battery Charger*	In / Out
3	*Load*	*Charging Voltage*	In
4	*Battery*	*Battery* (Vcc)	Out
5	*Data Out* (TX)	*Data Sent*	Out
6	*Data In* (RX)	*Data Received*	In
7	Z_CLK	*Recognition / control accessories Use as* DCD *in data operation*	
8	Z_DATA	*Recognition / control accessories Use as* CTS *in data operation*	
9	MICG	*Ground for Microphone*	In
10	MICG	*Microphone Input*	
11	AUD	*Lodspeaker*	Out
12	AUDG	*Ground for External Speaker*	

2.5 Format SMS

Menurut spesifikasi ETSI (*European Telcommunications Standards Institute*) panjang maksimum sebuah pesan adalah 160 karakter. Fasilitas ini disediakan oleh jaringan telepon selular. Teknologi yang mendukung layanan SMS antara lain adalah GSM, *Time Division Multiple Access* (TDMA) dan *Code Division Multiple Access* (CDMA)[1].

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

[1] Ir. Bustam Khang. "Trik Pemrograman Aplikasi Berbasis SMS". Jakarta: PT Elex Media Komputindo. Jakarta, 2002.

2.6 PDU (*Protocol Data Unit*)

PDU mode adalah format pesan dalam heksa decimal *octet* dan *semi-oktet* dengan panjang mencapai 160 (7bit) atau 140 (8bit) karakter. Sehingga untuk mengirimkan pesan SMS, pesan yang ditulis dalam teks harus dikonversikan terlebih dahulu ke dalam format PDU dan semua SMS yang diterima oleh *handphone* agar bisa dibaca maka harus dikonversikan dari PDU mengirimkan SMS dan menerima SMS.

2.6.1 Format Data Kirim SMS dengan Mode PDU

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

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

- Jumlah pasangan *heksa decimal* SMS_*Centre* (Bilangan Heksa).
- *National atau International Code.*

  Untuk *National, Code subheader*nya adalah 81.

  Untuk *International, Code subheader*nya adalah 91
- Nomor SMS_*Centre*nya sendiri, dalam pasangan *heksa* dibolak-balik. Jika tertinggal satu angka *heksa* yang tidak memiliki pasangan, angka tersebut akan dipasangkan dengan huruf F di depannya.

Contoh untuk SMS_*Centre Exelcom* atau XL dapat ditulis dengan dua cara, sebagai berikut :

Cara 1 : (National)

0818445009 diubah menjadi :

- 06 → 6 pasang
- 81 → 1 pasang
- 80-81-44-05-90 → 5 pasang

Digabung menjadi : 06818081440590

Cara 2 : (Internasional)

- 07 → 7 pasang
- 91 → 1 pasang
- 26-18-48-54-00-F9 → 6 pasang

Digabung menjadi : 07912618485400F9

Berikut ini beberapa nomor SMS_*Centre* operator selular di Indonesia :

Tabel 2.3 : *Nomor SMS-Operator Selular di Indonesia*

No	Operator Selular	No SMSC	Kode PDU
1	Telkomsel	081100000	0691801100000
2	Satelindo	0816124	058106121F5
3	Excelcom / XL	0818445009	06818081440590
4	Indosat-M3	085500000	0681805500000

Tabel 2.4 : *Nomor SMSC Operator Seluler di Indonesia yang Telah Dikonversi ke Dalam PDU Kode*

No.	Operator Seluler	No SMSC	Kode PDU
1	Telkomsel	081100000	07912618010000F0
2	Satelindo	0816124	059126181652
3	Exelcom / XL	0818445009	07912618485400F9
4	Indosat-M3	085500000	07912658050000F0

➢ Tipe SMS

Untuk *send* tipe SMS=1, jadi bilangan heksanya adalah 01.

a. Nomor Referensi

Nomor referensi ini dibiarkan dulu 0, jadi bilangan heksanya adalah 00. Nantinya akan diberikan sebuah nomor referensi otomatis oleh.

b. Nomor Ponsel Penerima

Sama seperti cara menulis PDU *header* untuk SMSC, *header* ini terdiri atas dua bagian.

c. Nasional atau Internasional Kode

- Untuk nasional, kode *subheader*nya = 81
- Untuk internasional, kode *subheader*nya = 91
- Nomor ponsel yang dituju, dalam pasangan heksa dibolak-balik. Jika tertinggal 1 angka heksa yang tidak memiliki pasangan, maka angka tersebut dipasangkan dengan huruf F di depannya.

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

Cara 1 :

628129573337 diubah menjadi :

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

Diganti menjadi : 0B818021593733F7

Cara 2 :

628129573337 diubah menjadi :

- 0C → ada 12 angka
- 91
- 26-18-92-75-33-73

Digabung menjadi 0C91261892753373

d. Bentuk SMS

Terdiri dari : 0 → 00 → dikirim sebagai SMS.

1 → 01 → dikirim sebagai telex.

2 → 02 → dikirim sebagai fax.

e. Skema *Encoding Data I/O*

Terdapat dua skema, yaitu :

- Skema 7 bit, ditandai dengan angka 0 □ 00.
- Skema 8 bit, ditandai dengan angka lebih besar dari 0 □ diubah ke heksa.

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

f. Jangka Waktu Sebelum SMS *Expired*

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

g. Isi SMS

Header ini terdiri dari dua *subheader*, yaitu :

- Panjang isi (jumlah huruf dari isi)
  Misalnya untuk kata "hello"□ada lima huruf □05
- Isi berupa pasangan bilangan heksa
  Untuk ponsel, SMS berskema *encoding* 7 bit, jika kita mengetikan huruf dari *keyboard*-nya berarti kita telah membuat tujuh angka I/O berurutan.

Dua langkah yang harus lakukan untuk mengkonversi isi SMS, yaitu :

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

Contoh : untuk kata "hello"

Langkah pertama :

Bit	7	I
H	110	1000
E	110	0101
L	110	1100
L	110	1100
O	110	1111

Tabel 2.5 : *Skema 7 Bit SMS pada Telepon Selular*

				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**		^		4	D	T	d	T
0	1	0	1	**5**		Ω	%	5	E	U	e	U
0	1	1	0	**6**		Π	-	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	g	X
1	0	0	1	**9**		Θ	)	9	I	Y	i	Y
1	0	1	0	**10**	LF	≡	•	:	J	Z	j	Z
1	0	1	1	**11**			+	;	K	Ä	k	Â
1	1	0	0	**12**			,	<	L	Ö	l	Ô
1	1	0	1	**13**	C R		_	=	M		m	
1	1	1	0	**14**		B	.	>	N	Ü	n	Ü
1	1	1	1	**15**			/	?	O		o	

2.6.2 Format Data Terima SMS Dengan Mode PDU

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

a. No SMSC
b. Tipe SMS
c. No ponsel pengirim
d. Bentuk SMS
e. Skema *Encoding*
f. Tanggal dan waktu SMS di-*stamp* di SMSC

Diwakili oleh 12 pasangan bilangan heksa (6 pasang) yang berarti :

yy/mm/dd hh:mm:ss 9020023512380□23/02/2009 15:32:08

Batas waktu validitas□ jika tidak dibatasi dilambangkan dengan 00

g. Isi SMS

Isi SMS yang berupa tulisan di sini nantinya akan dilakukan pengkonversian yaitu mengubahnya menjadi 8 bit, kemudian menjadi 7 bit.

Contoh format SMS diterima dalam format PDU:

0791265805000F0040C9126581610739800002070225123800005E8329BFD06

Penjelasan dari format di atas adalah sebagai berikut :

Heksa	Penjelasan
07	Panjang pesan nomor SMSC
91	Tipe kode nasional/inernasional
265805000F0	Nomor *Service Center*
04	Tipe SMS (01= tipe untuk penerimaan)
0C	Panjang nomor ponsel penerima
91	Tipe kode nasional/internasional
265816107398	Nomor ponsel penerima
00	Tipe bentuk SMS (00-dikirim sebagai SMS)
00	Tipe data *Coding*
207022512380	Waktu SMS sampai di SMSC yaitu tanggal 23/02/1009, 15:32:08
00	Jangka waktu SMS *expired* (00 = tidak memiliki batas)
05	Panjang pesan dari SMS
E8329BFD06	Isi pesan

2.7 Perintah *AT-Command*

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

Sebagai contoh, jika mengirimkan "AT" ke telepon selular, maka ia akan menjawab "OK" seperti perintah di bawah ini :

PC/µC AT

HP OK

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

PC/µC	**AT+CMMS=1**
HP	**+CMMS:96**
	OK

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

Selain perintah diatas juga masih ada perintah yang lain misalnya :

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

Perintah AT+CMGR=1 berarti memerintahkan kepada ponsel untuk membaca pesan nomor 1 yang tersimpan dalam memori ponsel. Perintah AT+CMGD digunakan untuk menghapus pesan yang tersimpan di dalam memori ponsel. Dalam skripsi ini tidak membahas seluruh perintah AT yang tersedia akan tetapi hanya membahas perintah-perintah AT yang berkaitan dengan operasi SMS seperti mengirim SMS dan menghapus SMS.

2.8 Sensor Level Air

Sensor yang digunakan untuk mengukur ketinggian air sungai adalah potensiometer. Sensor ketinggian level air ini digunakan sebagai pendeteksi ketinggian air sungai, yang disimulasikan menggunakan pelampung analog pada tangki bensin kendaraan motor roda dua. Karena pada pelampung tangki bensin kendaraan motor roda dua sudah dilengkapi VR (*Variabel Resistor*) jenis potensio meter, yang digerakkan oleh pelampung untuk mendeteksi adanya perubahan ketinggian air. Yang mana tahanannya dapat berubah-ubah sesuai dengan letak kontak gesek dengan terminal-terminal ujungnya. Sehingga dapat pula dikatakan bahwa potensiometer memiliki perubahan nilai resistansi sebagai fungsi rotasi kontak gesek.

Gambar 2.5 : *Potensiometer*

Untuk mengkonversi gerakan naik-turun ketinggian air menjadi gerakan rotasi yang akan memutar potensiometer, digunakan sebuah pelampung dan pemberat. Potensiometer mengubah ketinggian air menjadi tegangan.

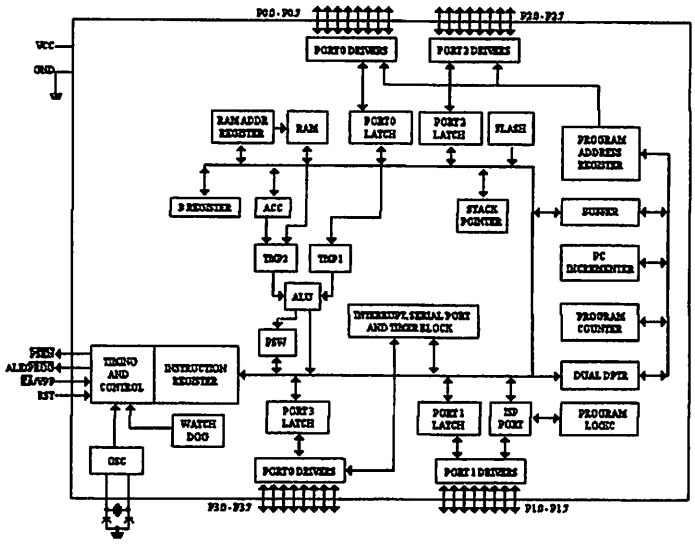
2.9 Mikrokontroler

Pada alat ini yang pertama mikrokontroler berguna sebagai penerima dan pengolah data sensor. Setelah diolah, data akan dikirim ke *handphone* secara serial. Yang kedua mikrokontroler berguna mengendalikan putaran motor atau pintu air.

2.9.1 Arsitektur Mikrokontroler AT89S52

Arsitektur dari AT89S52 mempunyai struktur yang serupa dengan IC mikrokontroler 8051 dari Intel (keluarga MCS51). Mikrokontroler dengan arsitektur 89XX merupakan salah satu jenis arsitektur mikrokontroler yang paling lama dan paling banyak digunakan di dunia. Seri mikrokontroler beraksitektur 8052, baik dari keluarga Atmel 89XX maupun dari vendor-vendor lain, memiliki beragam tipe dan fasilitas, namun kesemuanya memiliki arsitektur yang sama, dan juga set instruksi yang relatif tidak berbeda.

Mikrokontroler merupakan mikroprosesor (CPU) yang dilengkapi dengan komponen-komponen pendukung seperti : RAM, ROM, port I/O (*Input/Output*), bus control seperti EA (*External Access*), RST (*Reset*), ALE (*Address Latch Enable*), PSEN (*Program Store Enable*) dan lain-lain. Hal tersebut dapat kita lihat dalam arsitektur AT89S52 pada gambar diagram blok AT89S52 di bawah ini :

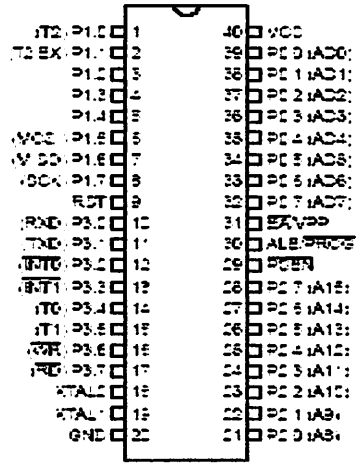


Gambar 2.6 : *Diagram Blok AT89S52*[2]

Semua produk mikrokontroler *flash* AT89S51, AT89C51 ataupun AT89S52 dari Atmel memiliki ruang alamat memori data dan program yang terpisah. Pemisahan memori program dan data tersebut membolehkan memori data diakses dengan alamat 8-bit, sehingga dapat dengan cepat dan mudah disimpan dan dimanipulasi oleh CPU 8-bit. Namun demikian, alamat memori data 16-bit juga dihasilkan melalui register DPTR (*Data Pointer*).

2.9.2 Konfigurasi Pin AT89S52

Konfigurasi untuk IC mikrokontroler dapat kita lihat pada gambar konfigurasi kaki IC mikrokontroler AT89S52 di bawah ini :



Gambar 2.7 : *Konfigurasi Kaki IC Mikrokontroler AT89S52*

[2] Totok Budioko,"Belajar dengan mudah dan cepat Pemrograman Bahasa C dengan SDCC pada Mikrokontroller AT89S52", GAVA MEDIA Yogyakarta, Maret 2005.

Adapun konfigurasi pin IC AT89S52 adalah sebagai berikut :

1. **VCC**

   Pin VCC terdapat pada pin 40. VCC berfungsi sebagai pemberi tegangan suplai. AT89S51 beroperasi pada tegangan 5 volt.

2. **GND**

   Sebagai *ground* atau pentanahan, pin ini dihubungkan ke tegangan 0 volt.

3. **Port 0**

   Merupakan *dual-purpose port* (port yang memiliki dua kegunaan). Pada desain minimum (sederhana) digunakan sebagai port I/O (*input* atau *output*). Pada desain yang lebih lanjut pada perancangan dengan memori eksternal digunakan sebagai data dan *address* yang di-*multiplex*. *Port* 0 terdapat pada pin 32-39.

4. **Port 1**

   Merupakan *open drain bidirectional*. Maksudnya ialah *port* ini dapat berfungsi sebagai masukan atau keluaran atau port I/O (*input* atau *output*). *Port* 1 terdapat pada pin 1-8.

5. **Port 2**

   Merupakan *dual-purpose port*. Pada desain minimum digunakan sebagai *port* I/O. Pada desain lebih lanjut digunakan sebagai *high byte* dari *address*. *Port* 2 terdapat pada pin 21-28.

6. **Port 3**

   Merupakan *dual-purpose port*. Selain berfungsi sebagai I/O juga mempunyai fungsi khusus yang ditunjukan pada tabel di bawah ini :

Tabel 2.6: *Fungsi Khusus Port 3*[3]

Port Pin	Alternate Functions
P3.0	RXD (*serial input port*)
P3.1	TXD (*serial output port*)
P3.2	_INT0 (*external interrupt 0*)
P3.3	_INT1 (*external interrupt 1*)
P3.4	T0 (*timer 0 external input*)
P3.5	T1 (*timer 1 external input*)
P3.6	_WR (*external data memory write strobe*)
P3.7	_RD (*external data memory read strobe*)

[3] IPutra, Agfianto Eko, "Belajar Mikrokontroller AT89C51/C52/C55" ANDI, 2003.

7. **PSEN (*Program Store Enable*)**

PSEN adalah kontrol sinyal yang mengijinkan untuk mengakses program (*code*) memori eksternal. Pin ini dihubungkan ke pin OE (*Output Enable*) dari EPROM. Sinyal PSEN akan 0 pada tahap *fetch* (penjemputan) instruksi. PSEN akan selalu bernilai 0 pada pembacaan program memori internal. PSEN terdapat pada pin 29.

8. **ALE (*Address Latch Enable*)**

ALE digunakan untuk men-*demultiplex address* dan *data bus.* Ketika menggunakan program memori eksternal *port* 0 akan berfungsi sebagai *address* dan *data bus*. Pada setengah paruh pertama *memory cycle* ALE akan bernilai 1 sehingga mengijinkan penulisan alamat pada register eksternal dan pada setengah paruh berikutnya akan bernilai satu sehingga port 0 dapat digunakan sebagai data bus. ALE terdapat pada pin 30. Keluaran ALE menghasilkan pulsa-pulsa untuk mengancing alamat *byte* rendah (*low byte*) selama mengakses memori eksternal.

9. **EA (*External Access*)**

Jika EA diberi masukan 1 maka AT89S52 menjalankan program memori internal saja. Jika EA diberi masukan 0 (*ground*) maka AT89S52 hanya akan menjalankan program memori eksternal (PSEN akan bernilai 0). EA terdapat pada pin 31.

10. **RST (*Reset*)**

RST pada pin 9 merupakan reset dari AT89S52. Jika pin ini diberi masukan 1 selama minimal 2 *machine cycle* maka sistem akan di-*reset.*

2.9.3 Program Memori Internal

AT89S52 memiliki program memori internal sebesar 8 Kbyte dengan ruang alamat 0000H-0FFFH. Jika alamat program lebih tinggi dari 0FFFH, yang melebihi kapasitas ROM internal menyebabkan AT89S52 secara otomatis mengambil Code byte juga dapat diambil hanya dari eksternal memori dengan alamat 0000H-FFFFH, menghubungkan pin EA ke ground.

2.9.4 Data Memori (RAM) Internal

Ruang alamat bawah memori data (RAM) internal dengan kapasitas 128 byte yaitu 00H-7FH yang terbagi atas 3 daerah, yaitu :

1. **Empat Bank Register**

   Setiap *bank* terdiri dari 8 *register* (R0-R7) sehingga jumlah *register* untuk keempat bank *register* (bank 0-bank 3) menjadi 32 buah *register* yang menempati ruang alamat 00H-1FH. Mengaktifkan salah satu *bank register* dapat dilakukan dengan mengatur RS0-RS1 pada PSW (*Program Status Word*).

2. **Bit Addressable**

   Terdiri dari 16 *byte* yang berada pada alamat 20H-2FH. Masing-masing 128 bit lokasi ini dapat dialamati secara langsung. Sehingga hanya dengan sebuah *instruksi* saja setiap bit dalam area ini dapat diset, clear, AND dan OR. Dengan adanya sistem bit *addressable* RAM, proses yang seharusnya dijalankan dengan tiga *cycle* seperti pada listing dapat digantikan dengan sebuah instruksi yang hanya membutuhkan satu *cycle* saja. Dalam aplikasinya, lokasi yang dapat diakses dapat juga digunakan untuk menandai suatu lokasi bit tertentu baik berupa *register* fungsi khusus yang dapat dialamati secara bit ataupun lokasi-lokasi tertentu yang dapat dialamati secara bit. Terdiri atas 80 *byte* yang menempati alamat 30H-7FH. Yang dapat dialamati secara langsung dan digunakan untuk keperluan umum (*general purpose*) misalnya digunakan untuk lokasi *stack.*

2.10 EEPROM (AT24C16)

Serial EEPROM tipe 24xx merupakan memori serial yang menggunakan teknologi I2C dimana dengan adanya penggunaan teknologi tersebut, jumlah I/O yang digunakan untuk meng-akses memori tersebut semakin sedikit. Hal ini sangat bermanfaat bagi sebuah sistem yang memerlukan banyak I/O. Penggunaan I/O yang semakin sedikit untuk mengakses memori, akan menyediakan lebih banyak I/O yang dapat digunakan untuk keperluan lain.

I2C adalah teknologi komunikasi serial. Teknologi ini hanya menggunakan 2 buah jalur I/O yaitu SDA dan SCLSDA merupakan jalur data pada komunikasi I2C sedangkan SCL merupakan jalur clock dimana sinyal clock akan selalu muncul untuk setiap bit dari pengiriman data. Komunikasi I2C bukan hanya untuk serial EEPROM melainkan juga diperuntukkan bagi komponen-komponen lain yang mempunyai kemampuan untuk diakses secara I2C.

Oleh karena itu, untuk membedakan antara Serial EEPROM dengan komponen yang lain digunakan *Slave Address* yang menunjukkan identitas dari komponen tersebut. Dalam hal ini Serial EEPROM mempunyai kode 1010. Serial EEPROM I2C berdasarkan pengalamatannya terdiri dari 2 jenis yaitu :

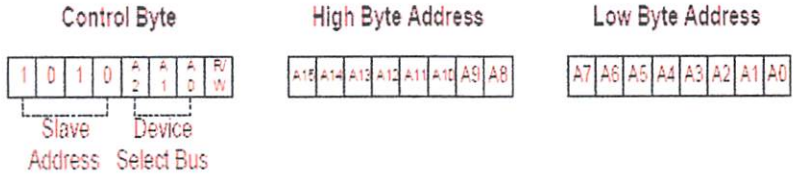
1. Pengalamatan 8 bit yang digunakan untuk Serial EEPROM dengan kapasitas memori sebesar 128 byte hingga 2Kb.
2. Pengalamatan 16 bit untuk Serial EEPROM 4Kb hingga 512Kb.

Pada pengalamatan 16 bit terdapat 3 buah paket 8 bit data yang harus dikirimkan ke Serial EEPROM yaitu :

1. Control Byte
2. High Byte Address
3. Low Byte Address

Pada pengalamatan 8 bit hanya diperlukan 2 buah paket 8 bit data yaitu :

1. Control Byte
2. Byte Address



Gambar 2.8 : *Pengalamatan 16 Bit Serial EEPROM*

Tabel 2.7 : *Jenis Serial EEPROM AT24Cxx*

Type	Kapasitas
AT24C01	128 byte
AT24C02	256 byte
AT24C04	512 byte
AT24C08	1 Kbyte
AT24C16	2 Kbyte
AT24C32	4 Kbyte
AT24C64	8 Kbyte
AT24C128	16 Kbyte
AT24C256	32 Kbyte
AT24C512	64 Kbyte

Control byte terdiri dari Slave Address, Device Select Bus bit-bit pengatur alamat dari Serial EEPROM dalam satu jalur bus, R/W bit penentu proses penulisan atau pembacaan data dari Serial EEPROM. Pembacaan maupun penulisan data ke dalam Serial EEPROM, selalu diawali dengan pengiriman *Control Byte* dan *Address Byte*. Hanya pada penulisan data akan dilanjutkan dengan pengiriman data 8 bit, sedangkan sebaliknya pada pembacaan akan dilanjutkan dengan pengambilan data 8 bit. Bit R/W pada *Control Byte* akan berlogika 1 untuk pembacaan data dan berlogika 0 untuk penulisan data.

2.10.1 Penulisan Data EEPROM

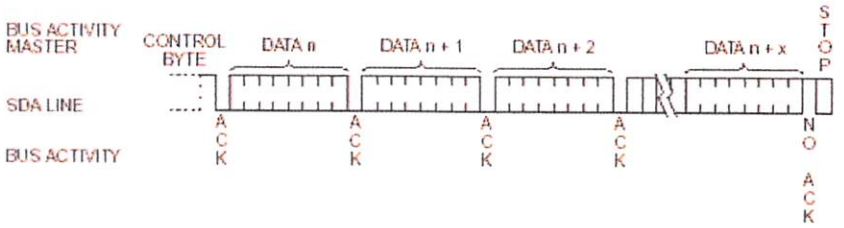
Penulisan data pada Serial EEPROM I2C dapat dilakukan secara byte maupun secara page. Pada penulisan secara byte dilakukan dengan mengirimkan control byte, alamat tujuan dan data sedangkan pada penulisan secara page dilakukan hanya dengan mengirimkan alamat tujuan awal saja yang kemudian dilanjutkan dengan 32 byte data yang akan menempati lokasi secara berurutan mulai dari alamat tujuan awal.

Gambar 2.9 : *Penulisan Secara Byte EEPROM*[4]

2.10.2 Pembacaan Data EEPROM

Pembacaan data dapat dilakukan secara *Current Address Read* (pembacaan alamat saat ini) maupun *random read (*pembacaan secara acak*)*. Pada *Current Address Read*, data yang dibaca adalah data pada alamat yang terakhir kali diakses saat itu, sedangkan pada pembacaan secara acak dilakukan dengan mengirimkan *control byte* dan alamat tujuan terlebih dahulu. Untuk pembacaan secara sequential, dilakukan dengan *control byte* dan dilanjutkan dengan data-data yang berada mulai dari alamat yang terakhir diakses saat itu berturut-turut hingga sinyal *stop* bit dikeluarkan.

[4] ST Microelectronics (1999) "16 Kbit Serial I2C Bus EEPROM with User-Defined Block Write Protection", Diakses 14 Januari 2011 dari All datasheet
http://www.alldatasheet.com/datasheet-pdf/pdf/23725/STMICROELECTRONICS/24C16.html



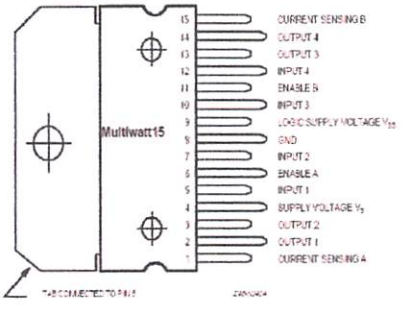
Gambar`2.10 : *Pembacaan Secara Sequential Read EEPROM*

2.11 L298 (*Dual Full - Bridge Driver*)

IC ini adalah sebuah *dual full-bridge driver* untuk arus dan tegangan DC yang cukup besar (max 4A dan max 46Volt) didesain untuk menerima level sinyal *standart* dan sebagai piranti pengendali beban lilitan, motor DC dan motor *Stepper*. Satu IC terdiri atas 2 rangkaian *full-bridge*, dua input *enable* (A dan B) disediakan untuk mengaktif atau menonaktifkan masing-masing *input*.

Gambar 2.11`: *IC L298*[5]

Tegangan masukan IC L298 (Vs) maksimum sampai 50 VDC, sedangkan tegangan kerja IC L298 (Vss) memiliki nilai maksimum 7 VDC. Tegangan untuk kedua *enable* adalah -0,3 – 7 VDC (-0,3 – 1,5 aktif low, 2,3 – Vss aktif high). Arus keluar pada masing-masing *full-bridge* sebesar 2A. Daya total dengan suhu 75°C sebesar 25 Watt, suhu operasi antara -25°C - 130°C.



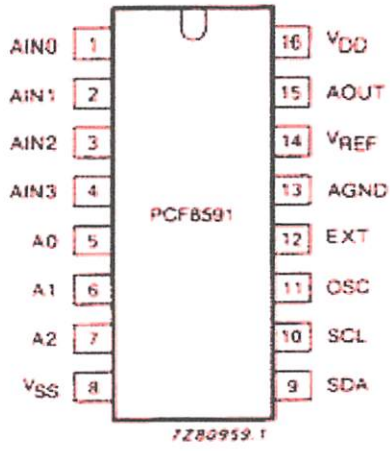
Gambar 2.12 : *Konfigurasi Pin IC L298*

[5] ST Microelectronics (2000) "DUAL FULL-BRIDGE DRIVER", Diakses 14 Januari 2011 dari All Datasheet. http://www.alldatasheet.com/datasheet-pdf/pdf/22437/STMICROELECTRONICS/ L298.html

Bila ingin menggunakan PWM sebagai pengatur kecepatan motor maka input PWM bisa diberikan pada pin *enable* IC L298.

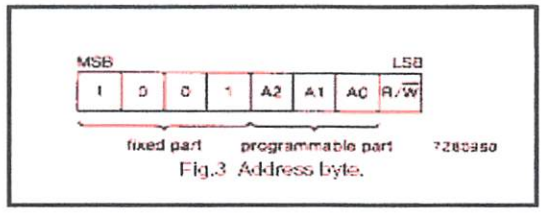
2.12 PCF 8591 (*8 Bit A/D and D/A Converter*)

PCF8591 ialah AD/DA *Converter* 8 bit dengan 4 input analog menggunakan interface I2C. Chip ini sudah ramai digunakan saat ini menggantikan ADC/DAC standar seperti 0808, 0809, MC1408 dan 0804. DT51 I2C AD/DA merupakan Analog Input Output Add-on board untuk DT51 menggunakan teknologi I2C-bus dimana IC utama yang digunakan ialah PCF8591. I2C AD/DA digunakan untuk mengubah sinyal analog seperti tegangan atau arus ke data biner dan sebaliknya. Aplikasinya antara lain kontrol kecepatan motor, pengaturan suhu ruang, akurasi data jarak jauh dan lainnya. Pin AIN0-AIN3 ialah pin input analog, sedangkan AOUT ialah sinyal output analog.



Gambar 2.13 : *Konfigurasi Pin PCF8591*[6]

PCF8591 yang memiliki teknik akses I2C maka untuk mengakses terlebih dahulu dilakukan pengalamatan sebagai berikut :



Gambar 2.14 : *Pengalamatan ADC PCF 8591*

[6] NXP Semiconductors (2 juli 1998) "8-Bit A/D and D/A converter ", Diakses 14 Januari 2011 dari All Datasheet. http://www.alldatasheet.com/datasheet-pdf/pdf/18222/PHILIPS/PCF8591.html

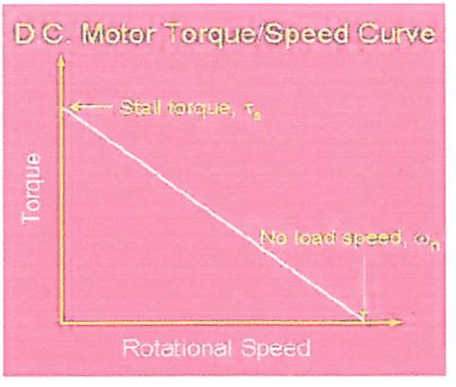
PCF8591 diaktifkan dengan mengirimkan Address Byte yang terdiri dari fix part dan programmable part dimana programmable part ini akan menyesuaikan dengan kondisi logika pada A0, A1 dan A2 di konektor ADC Address. Bit terakhir merupakan mode akses dimana logika 1 adalah Read dan logika 0 adalah Write. Akses I2C ini dapat dilakukan dari Port I2C yang telah disediakan dan dihubungkan ke I/O mikrokontroler sesuai program yang dirancang.

2.13 Motor DC

Motor DC pada skripsi ini berfungsi untuk menggerakkan pintu air. Motor DC adalah motor yang memerlukan *supply* tegangan searah pada kumparan jangkar dan kumparan medan untuk diubah menjadi energi mekanik. Berdasarkan karakteristiknya, motor arus searah ini mempunyai daerah pengaturan putaran yang luas dibandingkan dengan motor arus bolak-balik, sehingga sampai sekarang masih banyak digunakan pada pabrik-pabrik yang mesin produksinya memerlukan pengaturan putaran yang luas.

Ada dua kondisi yang dibutuhkan untuk menghasilkan gaya pada konduktor prinsip kerja motor DC, pertama konduktor harus dialiri arus, kedua konduktor harus berada di dalam medan magnet. Ketika dua kondisi terpenuhi, maka konduktor akan menerima gaya yang kanan menggerakkan konduktor sesuai dengan arah medan magnet. Prinsip ini merupakan dasar operasi motor DC. Setiap konduktor yang mengalirkan arus memiliki aliran medan magnet di sekitarnya. Jika konduktor berarus ditempatkan pada medan magnet, akan dihasilkan gaya yang akan menggerakkan konduktor.

Karakteristik yang dimiliki suatu motor DC dapat digambarkan melalui kurva daya dan kurva torsi atau kecepatannya, dari kurva tersebut dapat dianalisa batasan-batasan kerja dari motor serta daerah kerja optimum dari motor tersebut.



Grafik 2.1 : *Hubungan Antara Torsi dan Kecepatan*

Dari grafik diatas terlihat hubungan antara torsi dan kecepatan untuk suatu motor DC tertentu. Dari grafik terlihat bahwa torsi berbanding terbalik dengan kecepatan putaran, dengan kata lain terdapat *trade off* antara besar torsi yang dihasilkan motor dengan kecepatan putaran motor. Dua karakteristik penting terlihat dari grafik yaitu :

a. Stall torque (Ts), menunjukkan titik pada grafik dimana torsi maksimum, tetapi tidak ada putaran pada motor.
b. No load speed (GƆn), menunjukkan titik pada grafik dimana terjadi kecepatan putaran maksimum, tetapi tidak ada beban pada motor.

Analisa terhadap grafik dilakukan dengan menghubungkan kedua titik tersebut dengan sebuah garis, dimana persamaan garis tersebut dapat ditulis di dalam fungsi torsi atau kecepatan sudut.

$$\tau_{motor} = \tau_s - {}^{\omega\tau_s}/_{\omega_n}$$

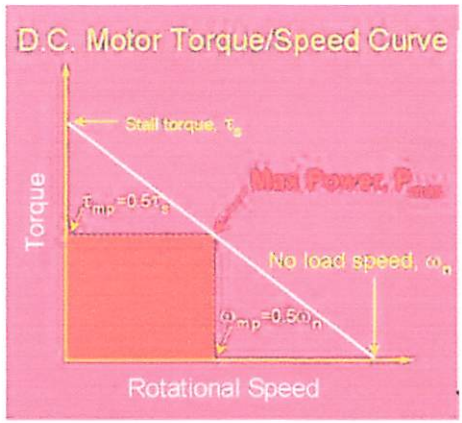
$$\omega_{motor} = (\tau_s - \tau)\,{}^{\omega_n}/_{\tau_s}$$

Dengan mensubstitusikan persamaan (torsi dan kecepatan) kedalam persamaan (daya) diperoleh :

$$P_{motor}(\omega) = -({}^{\tau_s}/_{\omega_n})\omega^2 + \tau_s\omega$$

$$P_{motor}(\tau) = -({}^{\omega_n}/_{\tau_s})\tau^2 + \omega_n\tau$$

Dari persamaan (daya) terlihat bahwa daya merupakan perkalian antara torsi dan kecepatan sudut, dimana di dalam grafik ditunjukkan oleh luas daerah segiempat dibawah kurva torsi atau kecepatan.

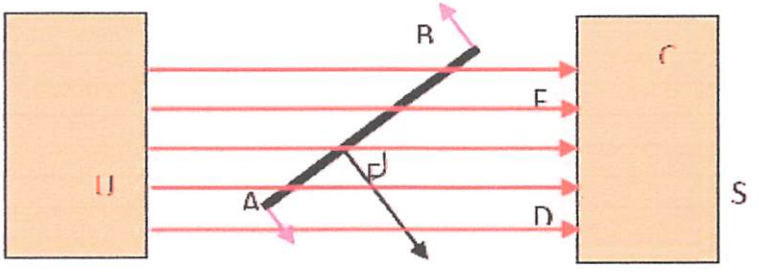


Grafik 2.2 : *Torsi dan Kecepatan Motor*

2.13.1 Cara Kerja Motor DC

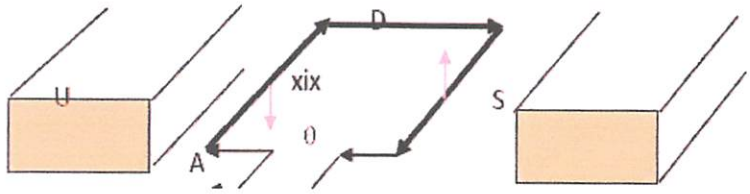
Motor DC atau motor arus searah adalah suatu mesin yang berfungsi untuk mengubah tenaga listrik arus searah menjadi tenaga gerak atau tenaga mekanik, yang tenaga gerak tersebut berupa putaran dari rotor. Prinsip kerja dari motor DC hampir sama dengan generator AC, perbedaannya hanya terletak dalam konversi daya. Prinsip dasarnya adalah apabila suatu kawat berarus diletakkan diantara kutub-kutub magnet (U-S), maka pada kawat itu akan bekerja suatu gaya yang menggerakkan kawat tersebut.

Motor DC terdapat dalam berbagai ukuran dan kekuatan, masing- masing didesain untuk keperluan yang berbeda-beda namun secara umum memiliki fungsi dasar yang sama yaitu mengubah energi elektrik menjadi energi mekanik. Sebuah motor DC sederhana dibangun dengan menempatkan kawat yang dialiri arus di dalam medan magnet. Kawat yang membentuk loop ditempatkan sedemikian rupa diantara dua buah magnet permanen. Bila arus mengalir pada kawat, arus akan menghasilkan medan magnet sendiri yang arahnya berubah-ubah terhadap arah medan magnet permanen sehingga menimbulkan putaran.



Gambar 2.15 : *Karakteristik Loop ABCD Dalam Satu Medan Magnet*

Pada gambar diatas sebuah loop ABCD berada dalam satu medan magnet. Jika arah flux magnet B berasal dari kutub U ke kutub S dari magnet permanen dan pada loop dialiri arus listrik dengan arah ABCD maka pada sisi AB akan terjadi gaya F1 yang mengarah kebawah, dan pada sisi CD juga terjadi gaya F2 yang mengarah keatas sesuai dengan aturan tangan kanan. Gaya F1 dan F2 tersebut menyebabkan loop berputar berlawanan dengan arah jarum jam. Proses tersebut terjadi terus-menerus dan merupakan dasar dari pembentukan sebuah motor.



Gambar 2.16 : *Torsi Pada Loop*

Gambar di atas torsi pada loop torsi yang dihasilkan oleh gaya F1 dan F2 sehingga menyebabkan loop berputar dapat dihitung dengan persamaan berikut :

τ = BIAN sinθ

B = Densitas flux magnetic yang berasal dri kutub U ke kutub S magnet (Wb/m^2)

I = Besar arus yang mengalir pada loop ABCD (A)

A = Luas loop ABCD yang memotong arah garis-garis flux B (m^2)

θ = Sudut antara bidang normal loop ABCD dengan B

N = Jumlah lilitan yang membentuk loop

Dengan K_b adalah konstanta yang diukur dari tegangan yang dihasilkan oleh motor ketika berputar setiap satuan kecepatan (Volt.det/rad). Magnitude dan polaritas K_b adalah fungsi kecepatan angular, ω dan arah putar poros motor. Persamaan tersebut dikenal sebagai persamaan DC motor secara umum.

2.14 LCD (*Liquid Crystal Display*)

LCD merupakan suatu piranti yang digunakan untuk menampilkan suatu huruf ataupun yang lain tanpa banyak memakan tempat dan dengan tampilan yang bagus, tetapi LCD ini juga mempunyai kode untuk mengakses tampilan yang ada, sama juga dengan *seven segment* juga mempunyai kode akses tersendiri. LCD juga mempunyai kode alamat untuk mengakses setiap karakter dari pada LCD. Pada alat ini LCD berfungsi sebagai alat monitoring yang akan menampilkan ketingian air berdasarkan sensor ketinggian air. Dalam skripsi ini digunakan LCD type 2x16 yang berarti memiliki karakter 2 baris dan 16 kolom, Adapun fungsi pin-pin LCD sebagai berikut :

Gambar 2.17 : *LCD M1632*

Tabel 2.8 : *Fungsi Pin-Pin LCD*

Pin no	Sinyal	I/O	Fungsi
1	Vss	Power	Ground
2	Vcc	Power	2,7V sampai 5,5V
3	V_{EE}	Power	Penggerak LCD
4	RS	Input	0: Instruktion register (write) dan address counter (read) 1: Data register (write dan read)
5	R/W	Input	Memilih operasi write (0) read (1)
6	E	Input	Memilih operasi write/read data
7...10	DB3-DB0	Input/Output	Empat high data bus three state bidirectional
11...14	DB7-DB4	Input/Output	Empat high data bus three state bidirectional

Keterangan dan fungsi masing-masing pin sebagai berikut :

- Pin 1 dihubungkan ke Gnd
- Pin 2 dihubungkan ke Vcc +5V
- Pin 3 dihubungkan ke bagian tengah potensiometer 10KOhm sebagai pengatur kontras
- Pin 4 untuk memberitahu LCD bahwa sinyal yang dikirim adalah data, jikaPin 4 ini diset ke logika 1 (high, +5V), atau memberitahu bahwa sinyal yang dikirim adalah perintah jika pin ini di set ke logika 0 (low, 0V)
- Pin 5 digunakan untuk mengatur fungsi LCD. Jika di set ke logika 1 (high,+5V) maka LCD berfungsi untuk menerima data (membaca data). Dan berfungsi untuk mengeluarkan data,jika pin ini di set ke logika 0 (low,0V). Namun kebanyakan aplikasi hanya digunakan untuk menerima data,sehingga pin 5 ini selalu dihubungkan ke Gnd.
- Pin 6 adalah terminal enable. Berlogika 1 setiap kali pengiriman atau pembacaan data.
- Pin 7 – Pin 14 adalah data 8 bit data bus (Aplikasi ini menggunakan 4 bit MSB saja, sehingga pin data yang digunakan hanya pin 11 - pin 14)
- Pin 15 dan Pin 16 adalah tegangan untuk menyalakan lampu LCD

Pada lembaran *datasheet* modul LCD M1632 disebutkan bahwa :

Power supply LCD meliputi :

Vss = 0

Vcc = 5V ±% (2mA)

Power supply back light :

V + BL = 4 – 4,2V (50 sampai 200mA)

V BL = 0V (GND)

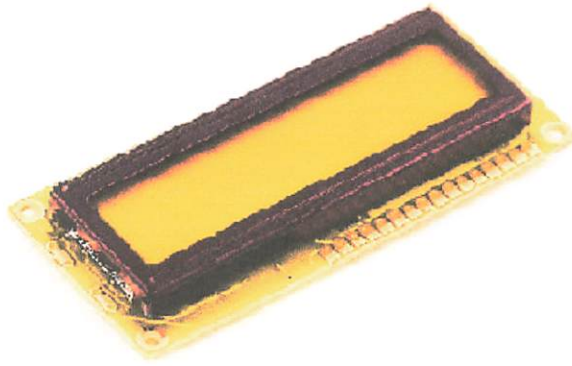
Pada input V + BL dipasang sebuah dioda IN4001 (bahan silicon dengan Vd = 0,65V sampai 0,7V), tujuannya adalah didapatkan tegangan V + BL sebesar 4 – 4,2V dengan perhitungan sebagai berikut :

Vcc = Vd +(V + BL)

5 = 0,7 + (V +BL)

(V + BL) = 5 – 0,7 = 4,3Volt

Dipilih dioda IN4001 karena arus maksimum yang biasa dilewatkan oleh dioda ini sebesar 1A. Adapun bentuk fisik dari LCD 2x16 bisa dilihat pada gambar berikut :



Gambar 2.18 : *Bentuk Fisik LCD 2x16*

2.15 Keypad 4x4

Pembuatan keypad, disini dimaksudkan untuk memberikan inputan ke mikrokontroler. Scanning keypad yang dilakukan oleh minimum sistem harus mampu menentukan posisi dari tombol yang ditekan. Setelah posisi keypad yang aktif dapat ditemukan maka data tersebut diolah menjadi data-data tombol yang ditekan. Dalam aplikasi ini digunakan satu macam scanning yang digunakan untuk mendeteksi keypad 4x4.

Keypad yang digunakan dalam aplikasi ini adalah keypad matriks 4x4 (4 baris dan 4 kolom). Berisikan angka 0 sampai 9 dan tombol tambahan berupa karakter CAN, ENT, DOWN, UP, COR, serta MEN. Untuk mengenali bagian kolom dan baris yang aktif maka keypad ini dihubungkan dengan minimum sistem AT89S52. Kemudian dibuat program yang dapat mengenali tombol yang sedang ditekan. Program yang dibuat harus mampu mengenali setiap tombol yang ditekan sesuai perencanaan.

Berikut merupakan bagian-bagian dari keypad tersebut :

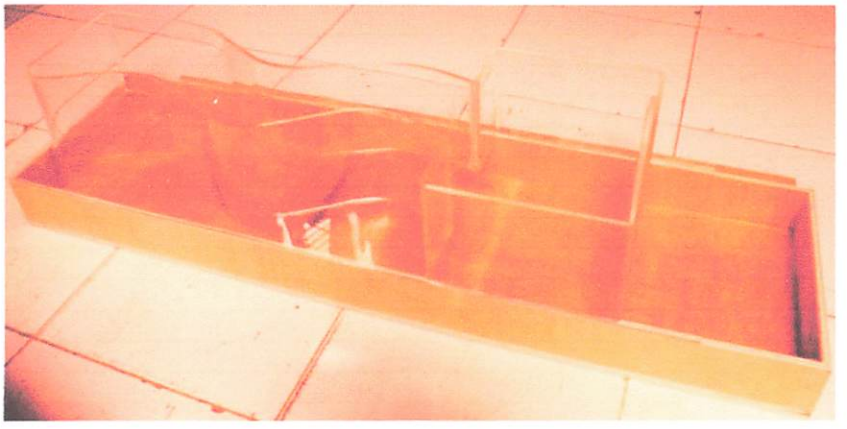
Gambar 2.19 : *Susunan Tombol*

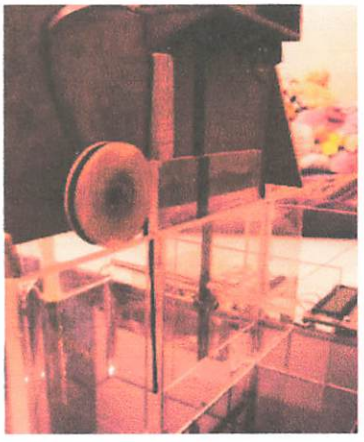
BAB III
PERANCANGAN
DAN
PEMBUATAN ALAT

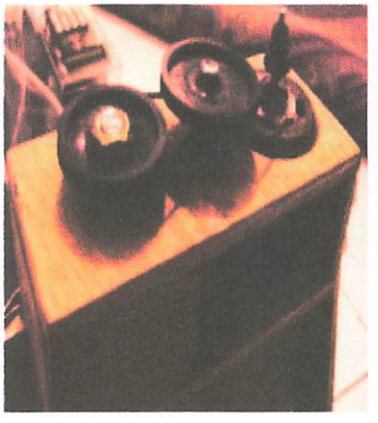
BAB III
PERANCANGAN DAN PEMBUATAN ALAT

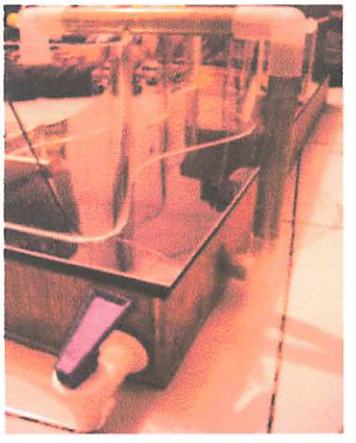
Perancangan merupakan proses yang kita lakukan terhadap alat, mulai dari rancangan kerja rangkaian hingga hasil jadi yang akan difungsikan. Perancangan dan pembuatan alat merupakan bagian yang terpenting dari seluruh pembuatan skripsi. Pada prinsipnya perancangan dengan sistematika yang baik akan memberikan kemudahan dalam proses pembuatan alat.

Pada bab ini membahas tentang perancangan dan pembuatan sistem mekanik dan elektronika. Alat ini bahan dasar mekaniknya menggunakan multiplex dengan tebal 5mm untuk bak air dan akrilik dengan tebal 5mm untuk konstruksi simulasi sungai. Desain dibuat semaksimal mungkin sama dengan keadaan sungai dan bendungan.



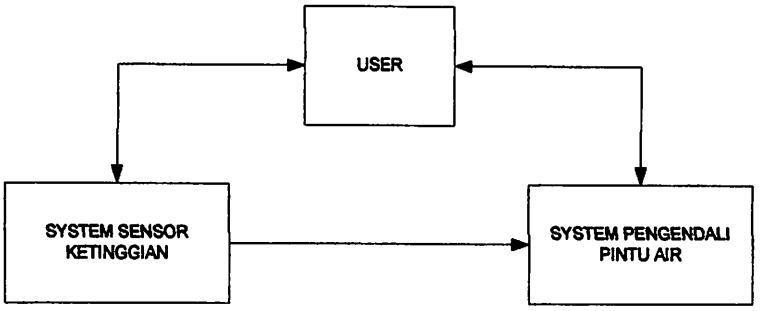




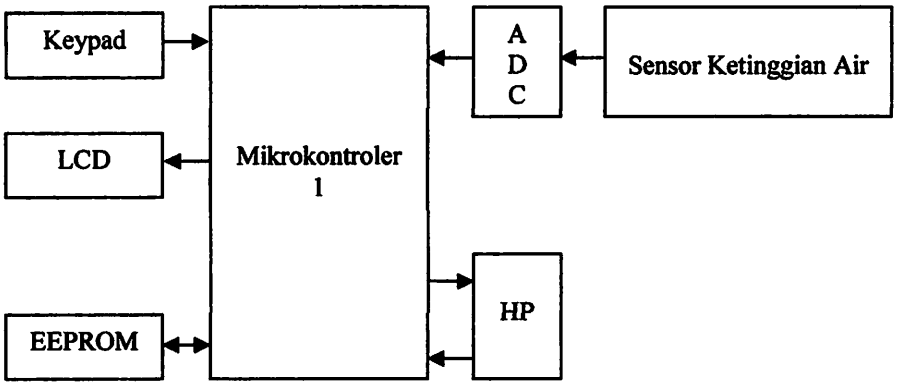


Gambar 3.1 : *Pembentukan Dasar Mekanik*

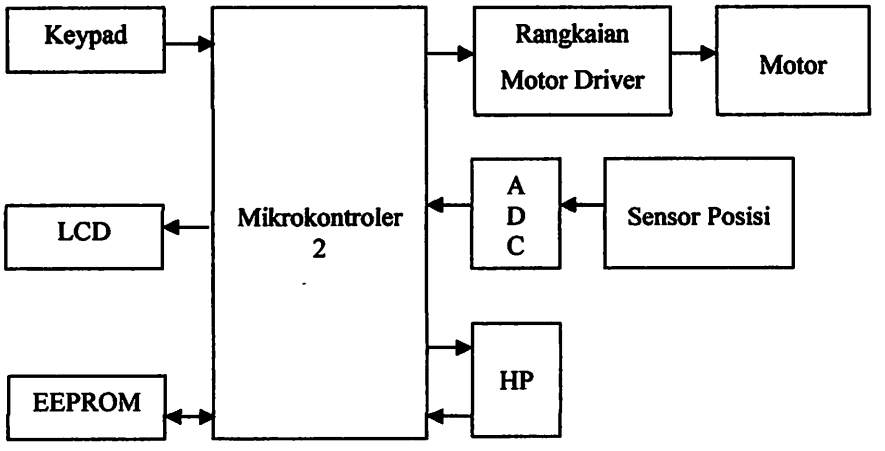
3.1 Blok Diagram



Gambar 3.2 : *Blok Diagram Konsep Pengiriman Data*



Gambar 3.3 : *Blok Diagram System Sensor Ketinggian Air*



Gambar 3.4 : *Blok Diagram System Pintu Air*

Dari gambar blok diagram di atas dapat dijelaskan secara umum dan fungsi dari masing-masing blok sebagai berikut :

- **Sensor Ketinggian Air**

  Sensor menggunakan potensiometer dimana pelampung bengsin kendaraan bermotor sebagai pendeteksi ketinggian level air.

- **ADC (*Analog to Digital Converter*)**

  Mengubah sinyal analog yang dihasilkan dari sensor ketinggian dan dikonversikan oleh ADC PCF8591 menjadi data digital.

- **Mikrokontroler AT89S52**

  Mikrokontroler AT89S52 digunakan sebagai pengolah data hasil pembacaan dari instruksi yang diterima dari ADC PCF8591 kemudian ditampilkan ke LCD.

- **LCD (*Liquid Crystal Display*)**

  Digunakan untuk menampilkan data yang telah diproses oleh Mikrokontroler AT89S52.

- **HP (*HandPhone)***

  Jenis HP yang digunakan adalah Siemen M35 dan C35, berfungsi sebagai pengirim dan penerima pesan dari Mikrokontroler.

- **Motor**

  Menggunakan motor DC dengan type L298 dimana di dalamnya sudah terdapat rangkaian *transistor* sebagai *driver* motor.

- **Sensor Posisi**

  Sensor yang digunakan jenis potensio mono 10kΩ yang berfungsi untuk membuka tutup pintu air sesuai dengan sensor ketinggian level air.

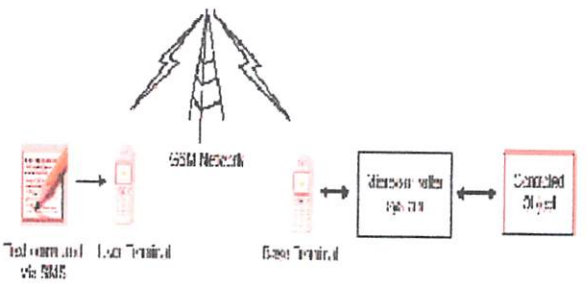
- **EEPROM**

  Menggunakan IC AT24C16 yang berfungsi sebagai memori eksternal.

- **Keypad**

  Menggunakan keypad bentuk matrik 4x4 yang berfungsi sebagai masukan dan mereset sistem mikrokontroler.

3.1.1 Komunikasi HP



Gambar 3.5 : *Alur Komunikasi*

Dalam perancangan sistem komunikkasi ini, GSM *network* merupakan operator penyedia jaringan GSM yang mendukung layanan pengiriman pesan pendek (*Short Message Service*). *User* Terminal merupakan peralatan *Mobile Station* yang berupa telepon selular yang berfungsi untuk melakukan pengiriman dan penerimaan pesan pendek (*Short Message Service*). Base Terminal juga merupakan peralatan *Mobile Station* yang berupa telepon seluler yang mampu melakukan fungsi pengiriman dan penerimaan pesan-pesan pendek melewati gerbang kabel data serial dengan menggunakan mode PDU (*Protocol Data Unit*). *Microcontroller System* merupakan perangkat keras yang terdiri dari sebuah mikrokontroler yang dilengkapi dengan perangkat lunak dan beberapa komponen tambahan yang berfungsi untuk melakukan pembacaan dan pengolahan data PDU yang diterima dari *Base Terminal*, serta melakukan pengendalian. *Controlled Object* merupakan perangkat-perangkat yang akan dikendalikan oleh sistem pengendali jarak jauh ini.

Sistem pengendali ini dirancang untuk memiliki proses kerja sebagai berikut :

1. User mengetikkan perintah-perintah berupa teks pada*User Terminal.*
2. Pesan perintah dalam bentuk SMS dikirimkan dari terminal pengguna (*User Terminal*) menuju ke terminal utama (*Base Terminal*) melewati jaringan GSM.
3. Pesan yang diterima oleh terminal utama (*Base Terminal*) yang berupa PDU SMS langsung diberitahukan ke sistem mikrokontroler dan dibaca.
4. Sistem mikrokontroler menterjemahkan pesan pada PDU SMS.
5. Sistem mikrokontroler membaca perintah-perintah teks yang ada dan menterjemahkan perintah tersebut menjadi suatu tindakan pengendalian.

3.1.2 Prinsip Kerja Alat

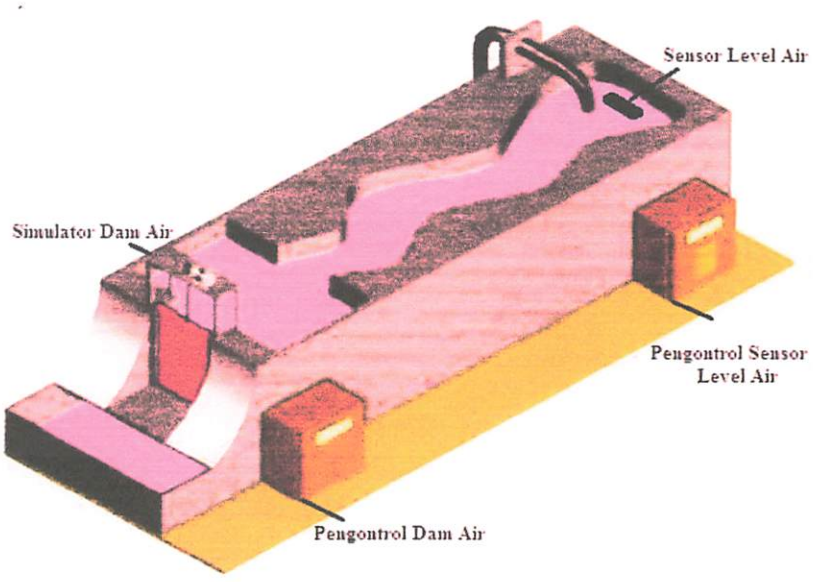
Dari gambar diagram blok sensor ketinggian air dan system pintu air, komunikasi antara kedua bagian ini dengan SMS *Gate Way*. Pada sensor ketinggian level air menggunakan sensor *potensiometer*. Sensor ketinggian air diletakkan jauh dari pintu air, jika terjadi hujan lebat pada daerah sensor atau daerah sebelum sensor diletakkan dan kenyataanya pada pintu air hanya terjadi gerimis atau sama sekali tidak terjadi hujan, otomatis sensor akan mendeteksi level air. Sensor mendeteksi keadaan ketingian air mencapai status tinggi dan akan mendekati status sangat tinggi. Untuk menanggulangi terjadinya kiriman air dari berbagai tempat, mikrokontroler akan mengirimkan data dengan menggunakan SMS ke pintu air dan penjaga pintu atau operator. Motor akan membuka pintu air secara otomatis sesuai dengan keadaan sensor ketinggian air. Jadi sebelum terjadinya peluapan air pada pintu air karena kiriman air dari berbagai tempat, pintu air telah membuka untuk mengembalikan ke keadaan normal. Pintu air akan berjalan otomatis mengirim berdasarkan kondisi sensor ketinggian air. *User*, operator atau penjaga air juga dapat mengirimkan sms untuk meminta status keadaan sensor ketinggian air. Contoh *user* mengirim SMS "RQST" *(Request)*, kemudian secara otomatis Hp yang berfungsi sebagai *interface* akan membalas operator "SR" (tinggi .... cm).

Mikrokontroler sebagai pengendelai *system* akan mengirim data dan Hp akan mengirim melalui SMS ke *user* dan penjaga pintu air, pintu air akan merespon secara otomatis dengan membuka dan menutup pintu air sesuai dengan kondisi sensor ketinggian air. Pada alat ini user juga dapat mengirimkan perintah untuk buka tutup pintu air, jika ada peluapan air secara berlebihan user dapat lansung mengirimkan SMS untuk membuka atau menutup sesuai keinginan.

Dari cara kerja alat di atas, maka tegangan keluaran sensor dibaca oleh ADC PCF8591 untuk merubah tegangan analog dari sensor menjadi digital untuk selanjutnya diproses oleh mikrokontroler AT98S52, kemudian mikrokontroler mengirimkan data ke HP (1) dan dikirim melalui SMS ke HP (2) yang ditampilkan pada LCD.

3.2 Perancangan Mekanik

Pada alat ini bahan dasar mekanik menggunakan multiplex dengan tebal 5mm untuk bak air dan akrilik dengan tebal 5mm untuk konstruksi simulasi sungai. Desain dibuat semaksimal mungkin sama dengan keadaan sungai dan bendungan.



Gambar 3.6 : *3D (Tiga Dimensi)*

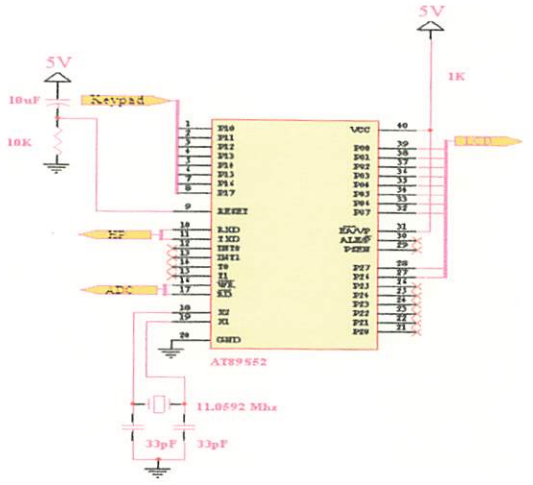
Dalam pembuatan penampung air menggunakan multiplex yang dilapisi dengan cairan fiber agar penampung air tidak mengalami kebocoran. Dan membekok akrilik menggunakan hiter 24V, akrilik dipanasi terlebih dahulu kemudian dibentuk sesuain dengan desain.

Tabel 3.1 : *Kondisi Pintu Air Terhadap Sensor Ketinggian Air*

NO	Ketinggian Dam	Kondisi Pintu Air	Motor
1	0 cm - 2 cm (Sangat Rendah)	Normal	Off
2	2 cm - 4 cm (Rendah)	Normal	Off
3	4 cm - 6 cm (Normal)	Normal	Off
4	6 cm - 8 cm (Tinggi)	Waspada	On
5	8 cm - 10 cm (Sangat Tinggi)	Siaga	On
6	≥ 11 cm	Banjir	On

3.3 Perancangan Mikrokontroler

Pembuatan minimum sistem AT89S52 didesain agar mudah digunakan termasuk penggunaan *port* yang menggunakan konektor *dip plug*. Mengingat konektor sangat kecil dan koneksi antara mikrokontroler dengan rangkaian yang lain sangatlah penting.



Gambar 3.7 : *Schematic Rangkaian Minimum System AT89S52*

Tabel 3.2 : *Konfigurasi Port*

Port	Port Pin	Function
1	P1.0 – P1.7	*Keypad*
2	P2.0	*Input 1 to motor driver*
	P2.1	*Input 2 to motor driver*
3	P3.0	RXD *(Serial Input Port) to Handphone C35*
	P3.1	TXD *(Serial Output Port) to Handphone C35*
	P3.2	ADC *(Analog Digital Converter)*
	P3.3	ADC *(Analog Digital Converter)*

3.3.1 Rangkaian Reset

Rangkaian reset ini diperlukan agar AT89S52 dapat direset secara otomatis pada saat pertama kali power diaktifkan, atau disebut power-on reset. Saat catu daya dinyalakan rangkaian reset akan menahan logika tinggi pada pin RST untuk jangka waktu tertentu. Jangka waktu tersebut ditentukan oleh pengosongan muatan pada kondensator, waktu penundaan yang diperlukan adalah 2μs (2kali siklus) ditambah dengan waktu penundaan saat kristal mulai on. Karena kristal yang digunakan mempunyai frekuensi sebesar 11Mhz maka satu periode membutuhkan waktu :

$$T = \frac{1}{f\,XTAL} = \frac{1}{11}$$

$$= 9.09 \times 10^{-8}\text{s}$$

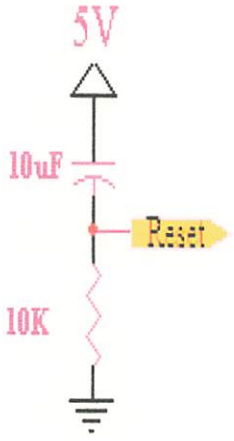
Sehingga waktu minimal logika tinggi yang dibutuhkan untuk mereset mikrokontroler adalah :

Reset (min) = T x Periode yang dibutuhkan

$= 9.09 \times 10^{-8}s \times 22$

$= 2\mu s$

Dari sini dapat disimpulkan bahwa mikrokontroler membutuhkan waktu minimal 2μs untuk mereset, waktu minimal inilah yang dijadikan pedoman untuk menentukan nilai R dan C. Dalam perancangan digunakan R = 10K, C = 10μF dan Vcc = 5V. Rangkaian reset ini juga ditambah dengan saklar agar dapat juga dilakukan secara manual. Rangkaian reset ditunjukkan dalam gambar 3.6 :

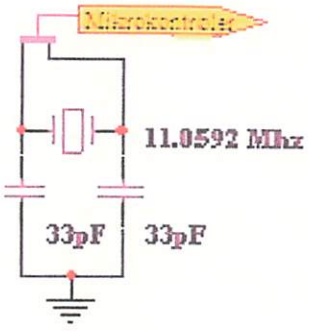


Gambar 3.8 : *Rangkaian Reset*

3.3.2 Rangkaian Clock

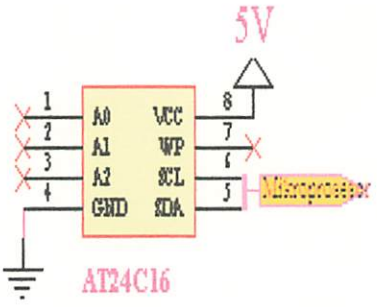
Mikrokontroler AT89S52 membutuhkan sinyal clock untuk berpindah dari satu kondisi ke kondisi yang lain, sebab dalam perancangan perangkat lunaknya berpindah dari satu state ke state yang lain dieksekusi bila ada clock.

Perencanaan clock ini menggunakan crystal 11Mhz, dua buah capasitor 33pF yang terhubung dengan kaki dari kristal dan rangkaian ini dihubungkan ke kaki 18-19 pada mikrokontroler. Dalam perencanaan alat ini clock digunakan sebagai pemicunya. Rangkaian clock seperti gambar berikut :



Gambar 3.9 : *Rangkaian Clock*

3.4 Perancangan EEPROM



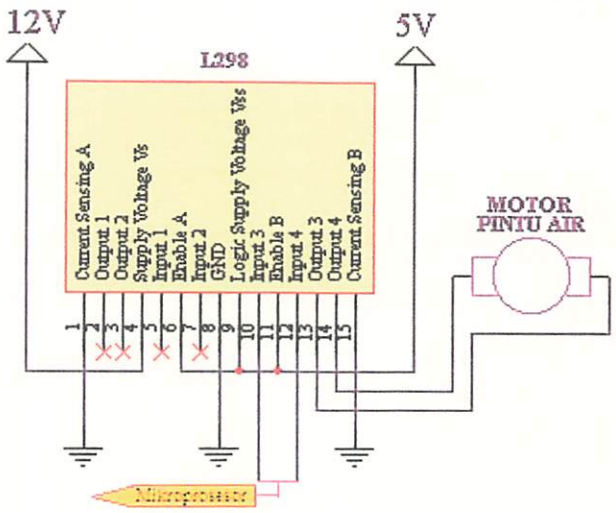
Gambar 3.10 : *Schematic Rangkaian Memori EEPROM*

EEPROM berfungsi sebagai memori *eksternal* dengan kapasitas 16Kbit. Pada rangkaian ini menggunakan IC AT24C16. EEPROM digunakan untuk menyimpan data *real* yang telah dikirim oleh ADC. Jika user meminta keadaan atau *request* keadaan EEPROM akan mengirim keadaan *real* kepada *user* menggunakan komunikasi SMS menggunkan *handphone*.

Penjelasan dalam perancangan dan pembuatan alat ini menggunakan EPPROM AT24C16 dikarenakan hal-hal di bawah ini :

- Perancangan dan pembuatan alat ini memerlukan memori yang cukup banyak untuk menyimpan data-data selain di memori internal mikrokontroler.
- EEPROM menggunakan teknologi I2C, bekerja secara byte maupun page yang bisa dipergunakan oleh komponen-komponen lain yang bisa diakses dengan I2C.
- Data pada EPPROM tersebut disimpan tidak secara tumpang tindih jadi membutuhkan banyak memori untuk dipakai.

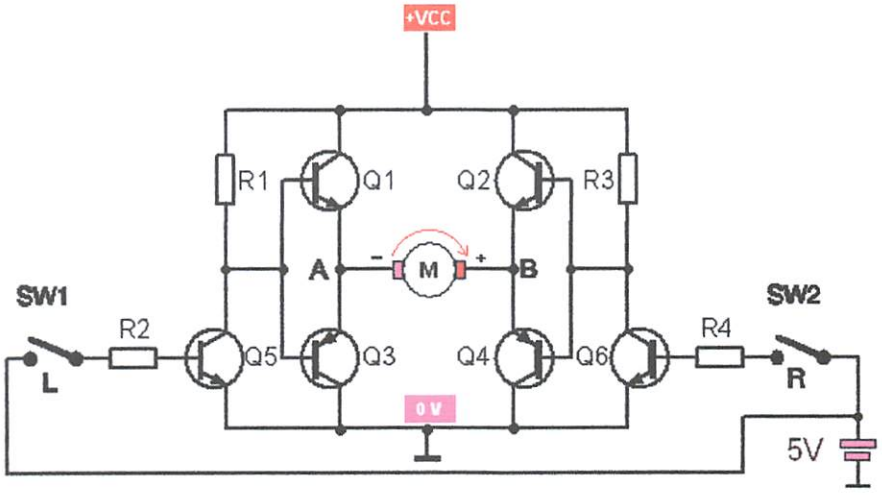
3.5 Perancangan Driver Motor



Gambar 3.11 : *Schematic Rangkaian Driver Motor*

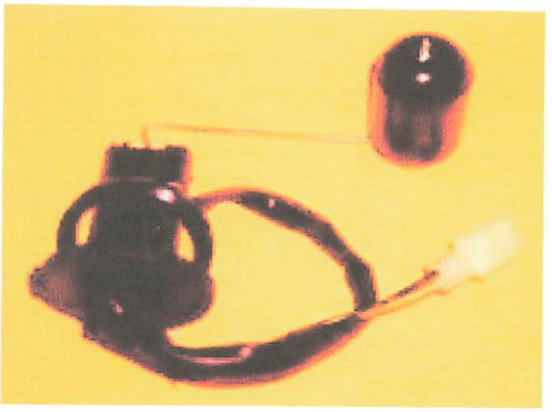
Driver motor alat ini menggunakan L298 dengan tegangan *input* (Vs) 12V untuk *power supply* dan (Vss) 5V untuk *logic power supply. Logic power supply* digunakan untuk mengaktifkan *enable, enable* berfungsi seperti kunci untuk *input* data yang masuk dari mikrokontroler *(1 dan 0).* Dipasang *diode* atau penyearah sebagai *input enable* dan agar data dari *input* 1 dan *input* 2 tidak saling bertabrakan. Pada rangkaian driver motor ini motor yang digunakan adalah motor DC 1A.

Dalam rangkaian *driver* motor penulis menggunakan L298 karena memanfaatkan komponen yang sudah ada. Dapat menggunakan rangkaian *transistor* sebagai *driver* motor yang berguna sebagai sakelar karena motor hanya memiliki kapasitas tegangan sebesar 12V dan arus sebesar 1A.



Gambar 3.12 : *Rangkaian Driver Motor Menggunkan Transistor Sebagai Sakelar.*

3.6 Perancangan Sensor Ketinggian Air



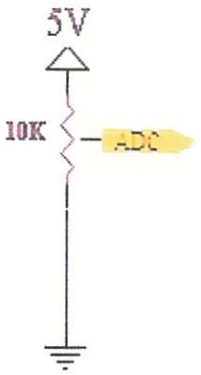
Gambar 3.13: *Pelampung Bensin*

Untuk sensor yang digunakan dalam perancangan monitoring level air adalah potensiometer. Potensiometer termasuk kedalam kelas *variabel resistor* yang nilai tahanannya dapat berubah-ubah sesuai dengan letak kontak gesek dengan terminal-terminal ujungnya. Sehingga dapat pula dikatakan bahwa potensiometer memiliki perubahan nilai resistansi sebagai fungsi rotasi kontak gesek. Potensiometer dapat dibedakan menjadi dua jenis yaitu jenis linier dan jenis logaritmik. Nilai resistansi potensiometer ditandai dengan huruf A (untuk jenis linier) dan B (untuk jenis logaritmik), misalnya A5 kΩ, B500Ω.

Hal-hal yang perlu diperhatikan dalam pemilihan potensiometer :

1. Resolusi, adalah perubahan terkecil dalam nilai resistansi yang dapat direalisasikan bila kontak gesek diputar.
2. Perlawanan ujung, adalah harga perlawanan antara kontak gesek dan terminal-terminal ujung, sementara kontak gesek berada di titik ujung salah satu terminal yang akan diukur nilai resistansinya.
3. Perlawanan kontak, adalah perlawanan antara terminal kontak gesek dan unsur perlawanan yang saling berkontak.
4. Arus kontak gesek (*wipper current*) adalah arus maksimum yang boleh mengalir di terminal kontak gesek.
5. Stabilitas setelan (*setting stability*) adalah kesamaan nilai resistansi pada perulangan (*repeatibility*) suatu setelan potensiometer pada suatu nilai resistansi.

Adapun rangkaian sensor level air ini adalah sebagai berikut :



Gambar 3.14 : *Sensor Ketinggian Menggunakan VR*

Dari gambar di atas dapat dianalisa dengan perhitungan 0-10kΩ dengan tinggi 0-12cm, maka dapat dihitung dengan rumus sebagai berikut :

$$Posisi\ Sensor = \frac{\text{Ketinggian}}{\text{Ketinggian Max}} \text{ x Posisi Sensor Max}$$

$$\text{R2} = \frac{\text{Posisi Sensor}}{\text{Posisi Sensor Max}} \text{ x R Total}$$

$$\text{R1} = \text{R Total} - \text{R2}$$

$$\text{R Total} = \text{R1} + \text{R2}$$

$$\text{V Out} = \frac{\text{R1}}{\text{R1} + \text{R2}} \text{ x Vin}$$

Dari hasil perhitungan yang telah dilakukan, diketahui :

Batas ketinggian = 0-12cm

Vin = 5 Volt

- Untuk batas ketinggian normal = 6 cm

  Posisi sensor = $\frac{6}{12}$ x 100 % = 50 %

  Maka :

  $$\text{R2} = \frac{50}{100} \text{ x } 10\ \text{k}\Omega$$
  $$= 5\ \Omega$$

  $$\text{R1} = 10\ \text{k}\Omega - 5$$
  $$= 5\ \Omega$$

$$R\ Total = 5\ \Omega + 5\ \Omega$$
$$= 10\ \Omega$$

$$V\ Out = \frac{5}{5+5} \times 5\ Volt$$
$$= 2{,}5\ Volt$$

❖ Untuk batas ketinggian tinggi = 8 cm

Posisi sensor = $\frac{8}{12}$ x 100 % = 66,66 %

Maka :

$$R2 = \frac{66{,}66}{100} \times 10\ k\Omega$$
$$= 6{,}6\ \Omega$$

$$R1 = 10\ k\Omega - 6{,}6\ \Omega$$
$$= 3{,}4\ \Omega$$

$$R\ Total = 6{,}6\ \Omega + 3{,}4\ \Omega$$
$$= 10\ \Omega$$

$$V\ Out = \frac{3{,}4}{3{,}4+6{,}6} \times 5\ Volt$$
$$= 1{,}7\ Volt$$

❖ Untuk batas ketinggian sangat tinggi = 10 cm

Posisi sensor = $\frac{10}{12}$ x 100 % = 83,33 %

Maka :

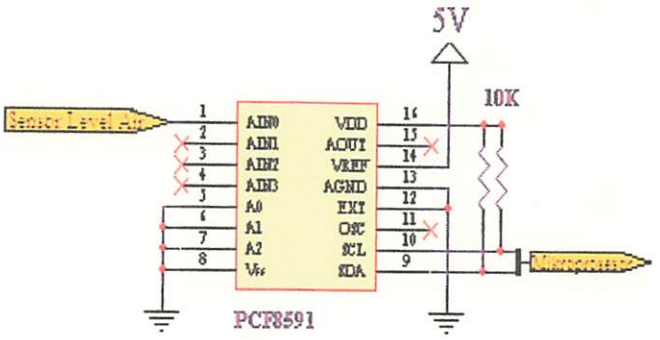
$$R2 = \frac{83{,}33}{100} \times 10\ k\Omega$$
$$= 8{,}3\ \Omega$$

$$R1 = 10\ k\Omega - 8{,}3\ \Omega$$
$$= 1{,}7\ \Omega$$

$$R\ Total = 8{,}3\ \Omega + 1{,}7\ \Omega$$
$$= 10\ \Omega$$

$$V\ Out = \frac{1{,}7}{1{,}7+8{,}3} \times 5\ Volt$$
$$= 0{,}85\ Volt$$

3.7 Perancangan ADC *(Analog to Digital)*



Gambar 3.15 : *Schematic Rangkaian ADC*

Pada alat ini ADC berfungsi sebagai *converter* dari sensor posisi (potensio) dan *converter* dari sensor ketinggian level air. ADC menggunakan PCF8591, terdapat resistor yang berfungsi sebagai *pull up*. *Output* dari PCF8591 adalah data yang akan dibaca oleh mikrokontroler.

ADC dan DAC diatas mempunyai keungggulan dibanding yang lain karena menggunakan komunikasi I2C untuk komunikasinya ke mikrokontroler. Fitur – fitur yang dimiliki IC tersebut :

- Single power supply
- Operating supply voltage 2.5 V to 6 V
- Arus standby kecil
- Serial input/output dengan I2C-bus
- Set alamat pada 3 pin ic
- Sampling rate given by I2C-bus speed
- 4 analog inputs programmable as single-ended or differential inputs
- Auto-incremented channel selection
- 8-bit successive approximation A/D conversion
- Multiplying DAC with one analog output.

(The I2C bus specification version 2.1 january 2000, philips semiconductor)

Komponen diatas merupakan komponen yang kompotibel dengan komunikasi I2C. Pemilihan dari komponen ini selain karena komunikasinya menggunakan I2C juga karena komponen ini telah terintegrasi adc dan dac dalam satu chip sehingga akan mudah dalam perancanganya.

3.8. Perancangan LCD M1632 (*Liquid Crystal Display*)

LCD diperlukan untuk menampilkan nilai karakter *input* yang akan diproses dan data karakter *output* dari hasil pengukuran supaya hasil proses dan pengukuran bisa dipahami. Rangkaian LCD ini dalam pengoperasiannya memerlukan 8 bit data dan 3 bit kontrol. Bagian utama dari rangkaian ini adalah penampil karakter LCD 16x2 baris. RS (*Register Select*) dan *enabel* pada pin 4 dan pin 6 yang merupakan kontrol dari LCD. Saluran data (*data bus*) dihubungkan ke *port* 0 *mikrokontroler*. Untuk pin R atau W akan berlogika *low* (0) apabila dihubungkan dengan *ground* maka LCD difungsikan hanya untuk menuliskan program atau data ke *display*. Untuk mengambil data dari *mikrokontroler* maka pin-pin data dihubungkan dengan *port* dari *mikrokontroler*.

Pada lembaran *datasheet* modul LCD M1632 disebutkan bahwa :

Power supply LCD meliputi :

Vss = 0

Vcc = 5V ±% (2mA)

Power supply back light :

V + BL = 4 – 4,2V (50 sampai 200mA)

V BL = 0V (GND)

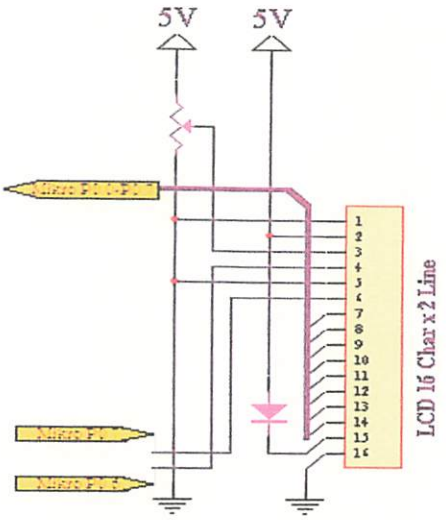
Pada input V + BL dipasang sebuah dioda IN4001 (bahan silicon dengan Vd = 0,65V sampai 0,7V), tujuannya adalah didapatkan tegangan V + BL sebesar 4 – 4,2V dengan perhitungan sebagai berikut :

Vcc = Vd +(V + BL)

5 = 0,7 + (V +BL)

(V + BL) = 5 – 0,7 = 4,3Volt.

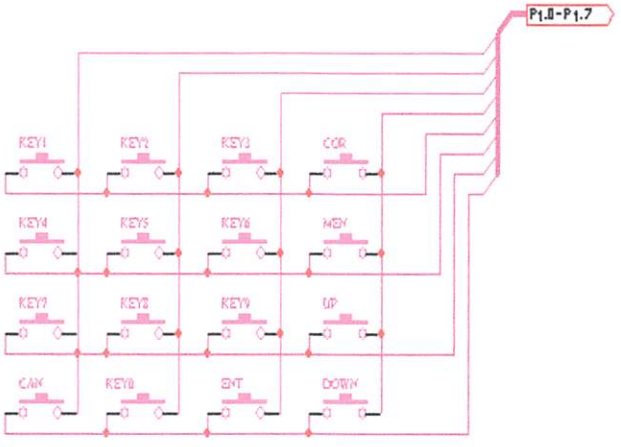
Dipilih dioda IN4001 karena arus maksimum yang biasa dilewatkan oleh dioda ini sebesar 1A. Berikut ini adalah rangkaian lengkap LCD yang digunakan pada perancangan :



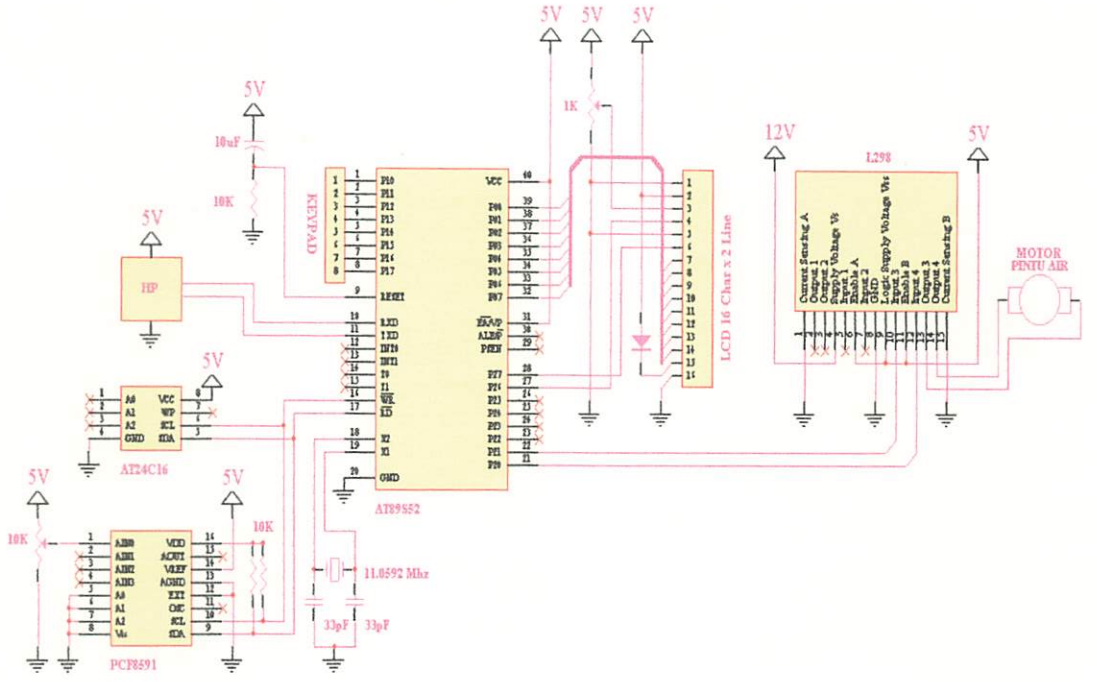
Gambar 3.16 : *Rangkaian LCD*

3.9 Perancangan Rangkaian Keypad

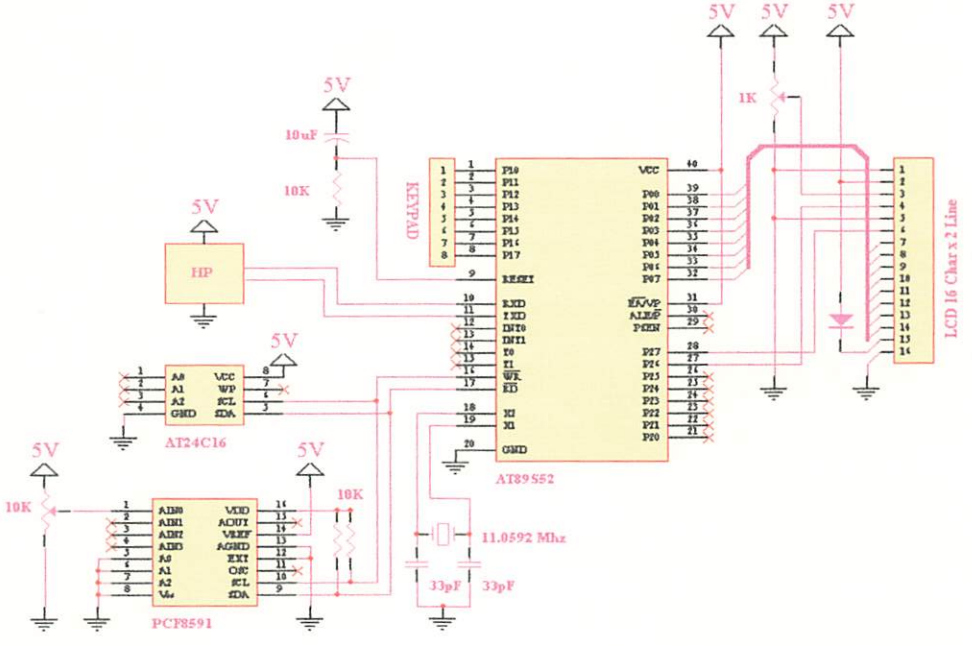
Keypad berfungsi untuk memasukkan input data base, rangkaian keypad menggunakan keypad 4 X 4 yaitu 12 buah saklar tekan (*push button*) yang dirangkai dalam bentuk matrik. Rangkaian keypad dihubungkan langsung ke mikrokontroller pin 1, 2 dan 3 digunakan sebagai scanning sedangkan pin 4, 5, 6 dan 7 sebagai data hasil penekanan keypad proses scaning dari penekanan keypad.



Gambar 3.17 : Rangkaian Keypad 4 X 4



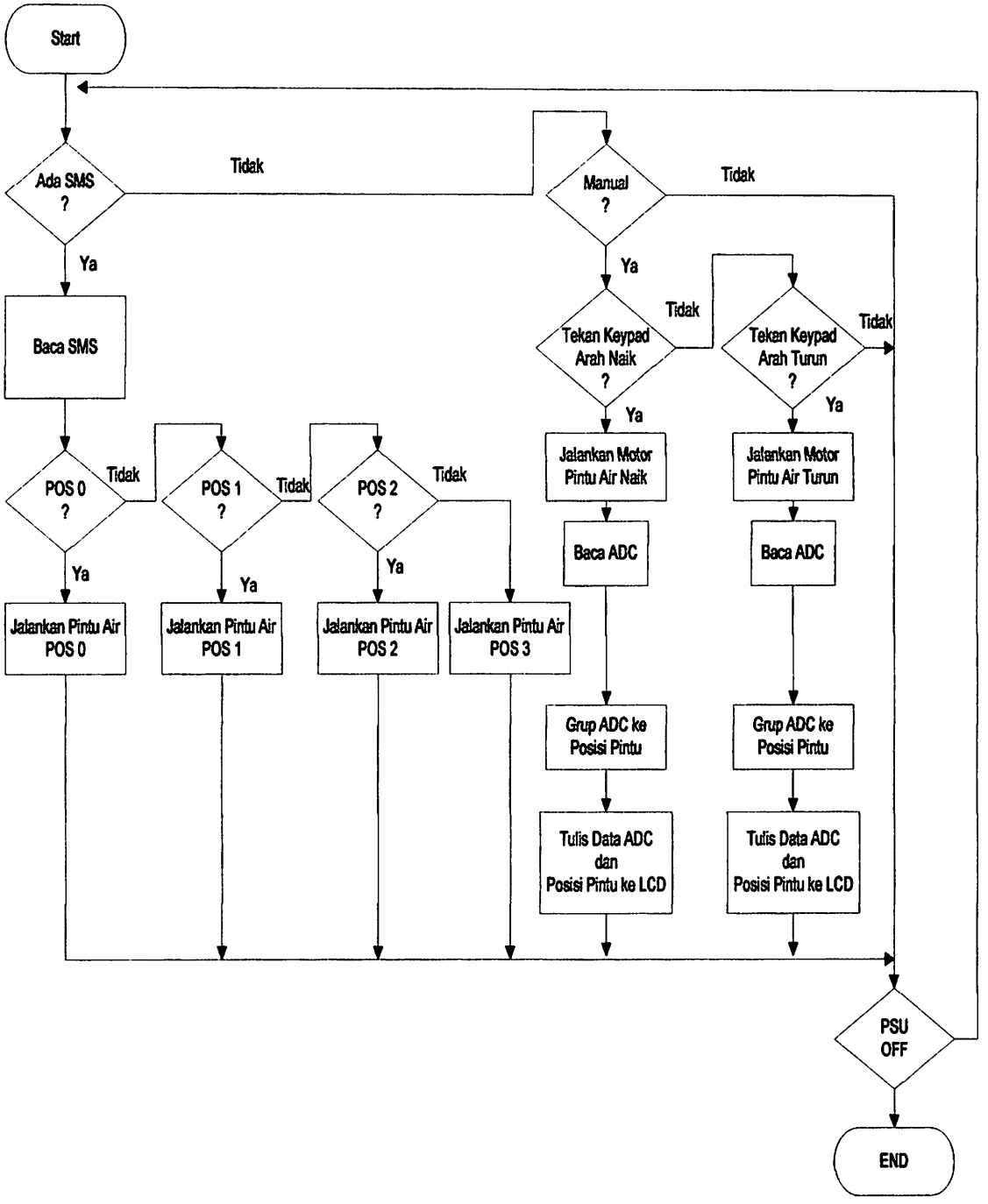
Gambar 3.18 : *Schematic Rangkaian System Keseluruhan "Pintu Air"*



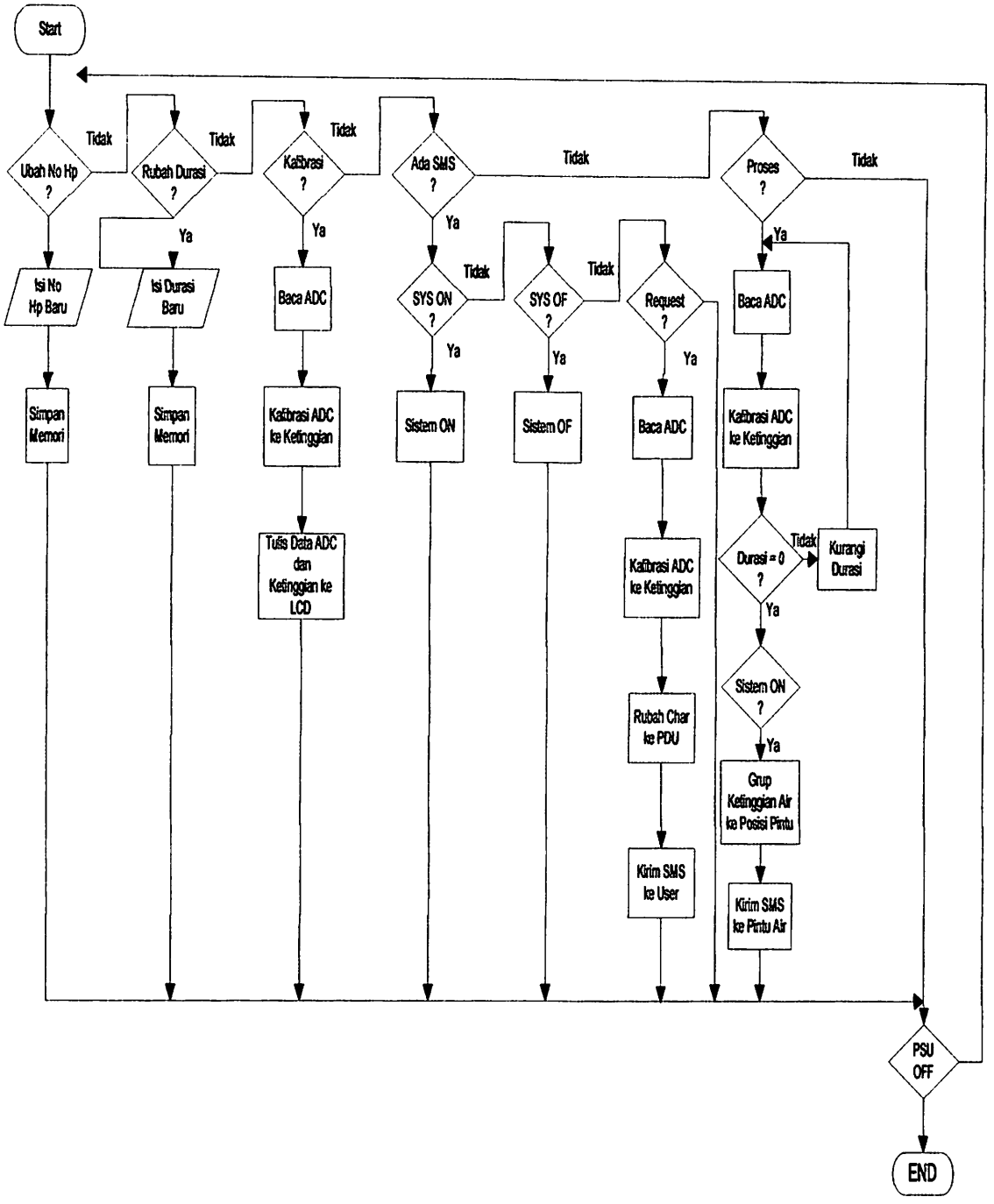
Gambar 3.19 : *Schematic Rangkaian System Keseluruhan "Sensor Ketinggian Air"*

3.10 Perancangan Perangkat Lunak (*Software*)

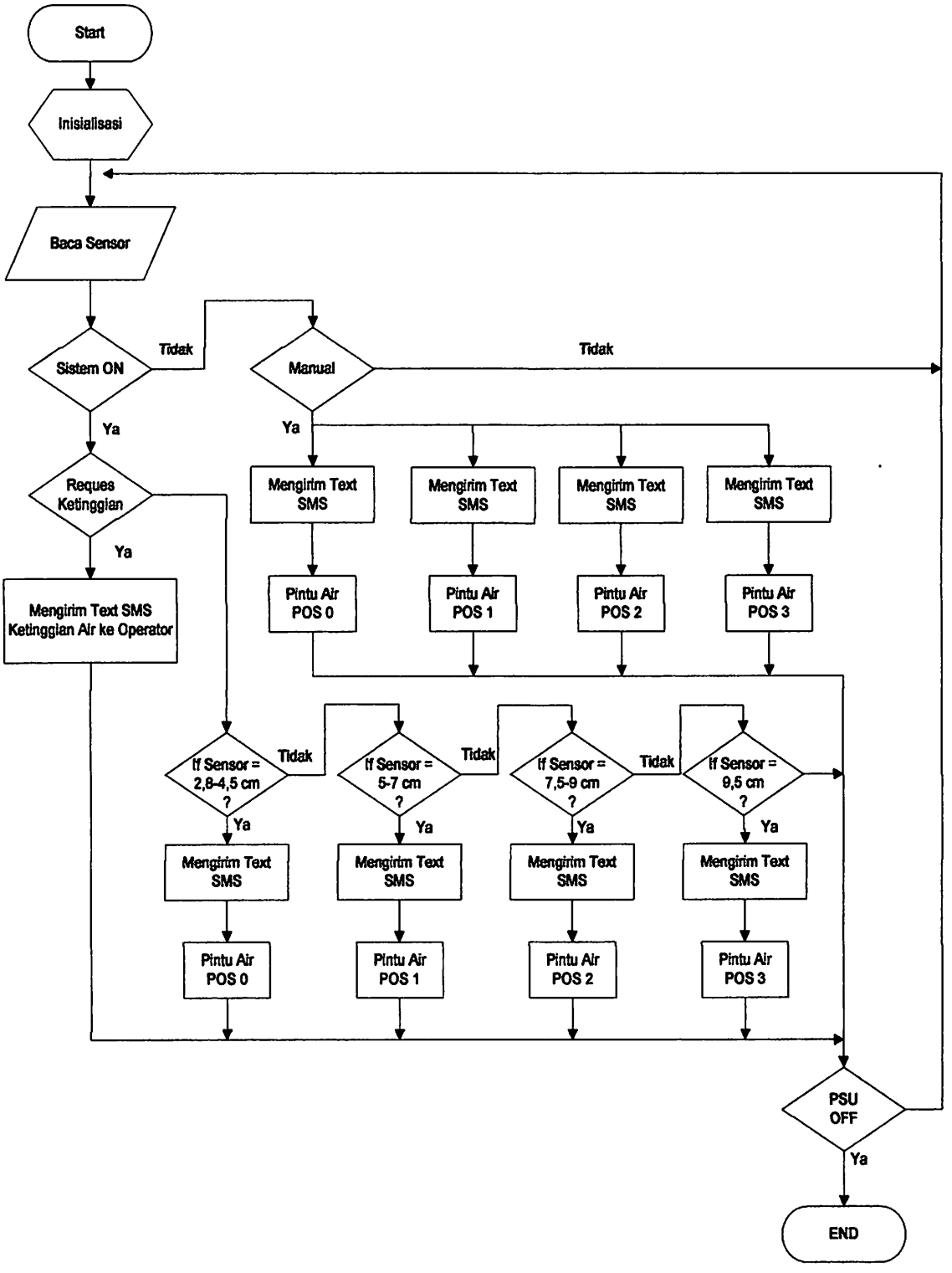
Pada perancangan perangkat lunak (*software*) dipaparkan dalam diagram alir secara keseluruhan dari semua system.



Gambar 3.20 : *Flowchart Sistem Pintu Air*



Gambar 3.21 : *Flowchart Sistem Sensor Ketinggian Air*



Gambar 3.22 : *Flowchart Sistem Keseluruhan*

BAB IV
PENGUJIAN
DAN
ANALISA

BAB IV
PENGUJIAN DAN ANALISA

Pada bab ini membahas cara pengujian dan analisa dari alat yang dirancang, sehingga dapat diketahui apakah alat tersebut dapat bekerja sesuai dengan yang telah direncanakan. Dalam rangka pengujian alat tersebut, diuraikan percobaan yang dilakukan untuk mengetahui respon dari keseluruhan alat yang dapat dirancang.

Setelah semua perancangan alat telah selesai dirancang secara keseluruhan, maka perlu dilakukan suatu pengujian sistem yang dilakukan pada tiap-tiap blok maupun secara keseluruhan, dimana pengujian sistem ini bertujuan agar :

1. Mengetahui sejauh mana *system* dapat bekerja secara maksimal.
2. Mencari dan menemukan beberapa permasalahan yang mungkin timbul pada saat alat ini beroperasi untuk kemudian diperbaiki sampai pada tingkat kesalahan sekecil mungkin sehingga didapatkan hasil yang baik.
3. Menganalisa hasil dari rangkaian dengan menghitung menggunakan rumus yang sesuai.
4. Mengetahui unjuk kerja alat secara keseluruan.

4.1 Pengujian Sensor Ketinggian Air

4.1.1 Tujuan

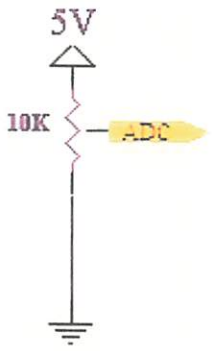
Untuk mengetahui ketinggian air sungai dan kemampuan rangkaian yang sudah dibuat, apakah dapat mendukung sistem yang direncanakan.

4.1.2 Peralatan yang Digunakan

- Catu Daya 5 Volt
- Sensor menggunakan pelampung bengsin kendaraan bermontor yang sudah terdapat VR
- Multimeter Digital
- Rangkaian yang akan diuji
- Menyiapkan tabel pengukuran

4.1.3 Prosedur Pengujian

- Menyusun rangkaian seperti pada gambar 4.1
- Menjalankan program untuk mengetahui tegangan keluaran (Vout) pada setiap batas ketinggian normal sampai sangat tinggi.

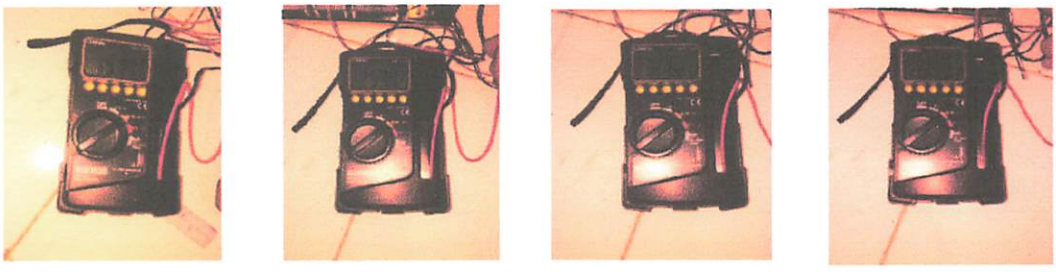


Gambar 4.1 : *Pengujian Rangkaian Variabel Resistor (VR)*

4.1.4 Hasil dan Analisa Pengujian

Tabel 4.1 : *Hasil Pengujian dan Pengukuran Sensor Batas Ketinggian*

No	Teks SMS	Tinggi Gerbang	VR (Ω)	Respon Pintu Setelah SMS Diterima	Derajat (°) Sudut Pelampung
1	POS 0	3 cm	151,4 Ω	5.42 detik	18°
2	POS 1	5 cm	253 Ω	6.48 detik	32°
3	POS 2	8 cm	325 Ω	4.68 detik	48°
4	POS 3	9.2 cm	333,3 Ω	6.68 detik	52°



Gambar 4.2 : *Pengukuran Dengan Multimeter Digital*

4.1.5 Analisa Respon Buka Tutup Pintu Air

Dalam percobaan buka tutup pintu air dan pengambilan data respon buka tutup pintu air terjadi *error* atau penyimpangan pada waktu respon pintu air setelah menerima SMS. Dari data yang diperoleh dengan menggunakan rumus matematis :

$$\sigma = \sqrt{\sigma^2}$$

$$\sigma^2 = (\sum (x_1 - \text{rerata})^2) / (n-1)$$

$$\text{rerata} = \sum xi / n$$

$$\text{rerata} = (5{,}42 + 6{,}42 + 4{,}68 + 6{,}68) / 4$$

$$\text{rerata} = 5{,}81$$

$\sigma^2 = (\sum (x_1 - \text{rerata})^2) / (n-1)$

$\sigma^2 = ((x1 - \text{rerata})^2 + (x2 - \text{rerata})^2 + (x3 - \text{rerata}) + (x4 - \text{rerata})^2) / (n-1)$

$\sigma^2 = ((5,42-5,81)^2 + (6,48-5,81)^2 + (4,68-5,81)^2 + (6,68-5,81)^2) / (n-1)$

$\sigma^2 = (0,15 + 0,44 + 1,27 + 0,75) / 4,18$

$\sigma^2 = 2,61 / 4$

$\sigma^2 = 0,65$

$\sigma = \sqrt{\sigma^2}$

$\sigma = \sqrt{0,65}$

$\sigma = 0,8$ detik

σ = *Deviasi* / penyimpangan / *error*

σ^2 = *Variansi*

4.2 Pengujian Respon Buka Tutup Pintu Air

Pengujian buka tutup setiap posisi bertujuan untuk mengetahui respon waktu (detik) setiap perpindahan posisi pintu air. Peralatan yang diperlukan sebagai berikut :

- Multimeter Digital.
- Rangkaian Driver Motor
- *Power Supply* DC 24 Volt
- *Handphone*
- Program Essembler
- Menyiapkan Tabel Pengukuran.

Tabel 4.2 : *Respon Waktu BukaTtutup Pintu Air Setiap Posisi*

No.	Jarak Buka Tutup Pintu Air	Waktu
1	POS 0 – POS 1	2.65 detik
2	POS 0 – POS 2	4.81 detik
3	POS 0 – POS 3	5.63 detik
4	POS 1 – POS 0	1.53 detik
5	POS 1 – POS 2	4.35 detik
6	POS 1 – POS 3	4.15 detik
7	POS 2 – POS 0	3.79 detik
8	POS 2 – POS 1	2.91 detik
9	POS 2 – POS 3	2.35 detik
10	POS 3 – POS 0	4.67 detik
11	POS 3 – POS 1	3.72 detik
12	POS 3 – POS 2	3.36 detik

4.3 Pengujian Sensor Potensio

Pada tegangan kondisi awal 0V. Tegangan output sensor potensio memiliki range 0V-3,20V. Data sebenarnya sampai pada posisi ADC 240 = 4,68V. Data tegangan output sensor dapat diperoleh dengan menggunakan rumus matematis :

$$V_{out} = V_{in} \times \frac{nilai\ ADC\ kondisi\ saat\ ini}{nilai\ ADC\ kondisi\ puncak}$$

Ket : V_{out} = Tegangan output

V_{in} = Tegangan input

ADC kondisi puncak = 240

Tabel 4.3 : *Hasil Perhitungan Percobaan Hubungan Tegangan dengan ADC*

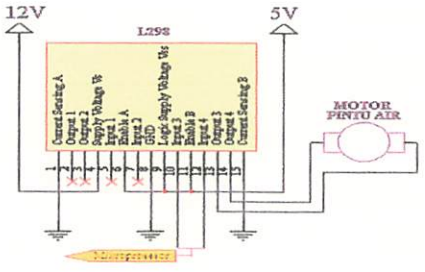
ADC	Tegangan (V)
0	0
29	0,56
50	0.97
81	1.59
108	2,10
135	2,62
160	3.11
170	3,31
194	3,77
214	4.17
240	4,68

4.4 Pengujian Rangkaian *Driver* Motor DC

4.4.1 Tujuan

Mengetahui apakah *driver* motor DC dapat bekerja dengan baik sesuai dengan apa yang diharapkan dan mengetahui tegangan dan arus pada motor DC.

4.4.2 Rangkain Pengujian Motor DC



Gambar 4.3 : *Pengujian Rangkaian Driver Motor DC*

4.4.3 Peralatan Yang Digunakan

- Multimeter Digital.
- Power supply 12 Volt dan 5 Volt.
- Rangkaian Driver Motor.

4.4.4 Langkah-Langkah Pengujian

Langkah- langkah pengujian antara lain :

- Rangkaian *driver* diberi tegangan 5 Volt Dc.
- Motor dc diberi tegangan 12 Dolt DC.
- Dilakukan Pengukuran tegangan dan arus pada motor dc pada saat ON.

4.4.5 Hasil Pengujian Motor DC

Tabel 4.4 : *Hasil Pengujian Motor DC*

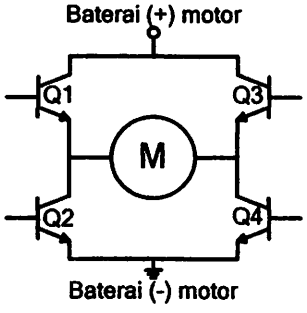
Spesifikasi		Motor DC ON	
V = 12 Volt	I = 1A	V = 11,93 Volt	I = 296,7 mA

Gambar 4.4 : *Pengujian Motor DC*

4.4.6 Analisa Perancangan dan Pengujian Driver Motor DC

Nilai input *driver* motor yang diberikan berasal dari Mikrokontroler AT89S52. Motor *gear* DC tidak dapat dikendalikan secara langsung oleh mikrokontroler, karena kebutuhan arus yang besar sedangkan keluaran arus dari mikrokontroler sangat kecil. *Driver* motor merupakan alternatif yang dapat digunakan untuk mengatur gerakan arah dan kecepatan motor DC dengan mudah.

Motor DC dikontrol menggunakan konfigurasi transistor yang dikenal dengan istilah *H-Bridge*. Konfigurasi ini menggunakan 4 buah transistor NPN atau dua transistor NPN dan dua transistor PNP.



Gambar 4.5 : *Konfigurasi H-Bridge*

Tabel 4.5 : *Kebenaran konfigurasi H-Bride*

Tabel Kebenaran			
	Maju	Mundur	Stop
Q1	1	0	0
Q2	0	1	0
Q3	0	1	0
Q4	1	0	0

Gambar diatas menunjukkan konfigurasi transistor NPN yang digunakan sebagai pengontrol motor DC. Arus yang mengalir ke motor DC polaritasnya dapat diatur dengan memberikan logika ke transistor Q1 sampai Q4. Pengaturannya seperti tabel kebenaran diatas. Transistor Q1 dan Q2 atau Q3 dan Q4 tidak diperbolehkan kondisi keduanya dalam keadaan *high* karena akan menyebabkan *short circuit* terhadap baterai.

Pada IC L298, kaki 1 dan 15 adalah kaki *sensing voltage*, kaki ini akan mengendalikan besarnya arus beban. Dalam keadaan tidak terhubung ke ground, tegangan sensing (Vsens) pada kaki ini sebesar 3 Volt. Agar motor dapat berputar, tegangan pada kaki 1 dan 15 harus sebesar -1 Volt sampai +2 Volt. Apabila

diinginkan arus yang mengalir pada beban sebesar 4 A (dari baterai 12V/4A), maka :

$$I = \frac{Vsens}{Rs}$$

$$4 = \frac{2}{Rs}$$

$$Rs = \frac{2}{4}$$

$$Rs = 0.5\Omega$$

Komponen utama rangkaian penggerak motor DC ini adalah IC L298, merupakan IC yang dipasang khusus untuk keperluan pengendalian motor DC. IC L298 ini di dalamnya terdapat 4 penggerak *push-pull* yang dapat mengalirkann arus pada keluarannya sampai sebesar 4 A setiap jalurnya. Masing-masing jalur dikendalikan oleh sebuah TTL yang kompatibel.

Hasil pengujian untuk mengendalikan motor DC yang terhubung dengan driver motor DC dapat dilihat pada tabel berikut ini :

Tabel 4.6 : *Analisa Perancangan Rangkaian Driver Motor*

Enable	IN+	IN-	Keadaan Motor	Hasil uji
0	X	X	Tidak berputar	OK
1	0	0	Tidak berputar	OK
	0	1	Berputar Searah Jarum Jam	OK
	1	0	Berputar Berlawanan Jarum Jam	OK
	1	1	Tidak berputar	OK

Dari hasil pengujian di atas dapat diketahui bahwa pada saat motor kondisi ON maka motor mempunyai tegangan sebesar 11,93 V dan arus 296,7 mA. Hasil ini memiliki toleransi pada kondisi spesifikasi motor, yang mana tegangan sebesar 12 V dan arus 1 A. Untuk motor DC ini mempunyai kecepatan putarannya adalah 36 rpm (*Revolutions Per Minute)*. Kecepatan putar tersebut didapat dengan mengukur secara manual, yaitu menghitung berapa putaran per menit.

4.5 Pengujian Sensor Ketinggian Air

Pengujian sensor ketinggian air bertujuan untuk mengetahui apakah sensor potensiometer dapat berfungsi dengan baik. Peralatan yang digunakan sebagai berikut :

- Multimeter Digital
- Rangkaian Driver Sensor
- Power Supply 12 V
- Air
- Menyiapkan Table Pengukuran

Tabel 4.7 : *Hasil Pengukuran ADC dan Ketinggian Sensor*

NO	ADC	Ketinggian Sensor	Respon Pintu Air
1	29	2,5 cm	POS 0
2	50	3 cm	POS 0
3	60	3,5 cm	POS 0
4	81	4 cm	POS 0
5	93	4,5 cm	POS 0
6	108	5 cm	POS 1
7	135	5,5 cm	POS 1
8	146	6 cm	POS 1
9	160	6,5 cm	POS 1
10	170	7 cm	POS 1
11	180	7,5 cm	POS 2
12	194	8 cm	POS 2
13	214	8,5 cm	POS 2
14	240	9 cm	POS 2
15	275	9,5 cm	POS 3

4.6 Display dengan LCD (*Liquid Crystal Display*)

Pada alat ini LCD berguna sebagai alat *monitoring system* ketinggian air dan *system* pintu air. Pada system ketinggian air berfungsi sebagai alat monitoring ketinggian air dan ADC yang dihasilkan secara manual, LCD juga menampilkan nomor *handphone* yang akan dituju dan menampilkan berjalannya timer sebelum mengirimkan data melalui SMS kepada pintu air.

Pada *system* pintu air LCD menampilkan ketinggian pintu air dan ADC yang dihasilkan.

Gambar 4.6 : *Tampilan LCD Initialisasi Handphone Saat System Dinyalakan*

Gambar 4.7 : *LCD Menampilkan Nama dan Nim*

Gambar 4.8 : *Tampilan LCD Jurusan dan Kampus*

Gambar 4.9 : *Tampilan LCD Saat Memasukkan Nomor Handphone*

Gambar 4.10 : *Tampilan LCD saat Timer Hitung Mundur untuk Mengirim SMS*

Gambar 4.11 : *Tampilan LCD saat Timer Tidak Aktif*

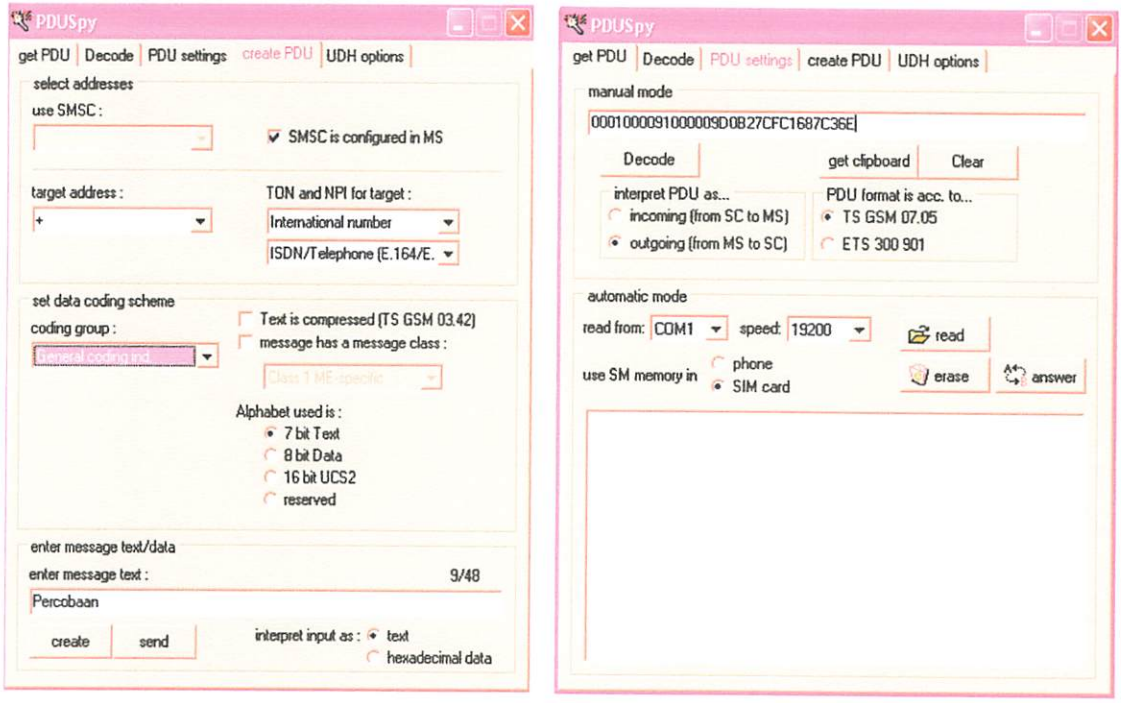
Gambar 4.12 : *Tampilan LCD Indikator ADC dan Ketinggian Air*

Gambar 4.13 : *Tampilan LCD Indikator ADC dan Posisi Pintu Air*

Gambar 4.14 : *Tampilan LCD saat Setting Timer*

4.7 Komunikasi Data

Komunikasi pada alat ini menggunakan *handphone* dengan format SMS, sebelum mengirim dan menerima, format teks SMS dirubah terlebih dahulu menjadi format PDU. Dengan format PDU data sudah siap dikirim dan diterima.



Gambar 4.15 : *Software Converter Format Teks SMS Menjadi Format PDU*

Tabel 4.8 : *Konfigurasi Pengiriman SMS*

No.	Format Teks	Format PDU	Keterangan
1	REQST	0001000C91261857048978 000005D262744A05	*User* meminta status terbaru ketinggian air.
2	SYSON	0001000C91261857048978 000005D3ECF4E904	Mengaktifkan *system* ketinggian agar dapat mengirim data ke pintu air secara otomatis.
3	SYSOF	0001000C91261857048978 000005D3ECF46904	Menonaktifkan *system* ketinggian agar tidak mengirim data ke pintu air secara otomatis. *User* dapat memberikan SMS melalui SMS untuk buka dan tutup pintu air.
4	POS 0	0001000C91261857048978 000005D0E7140403	Perintah *user* terhadap pintu air untuk buka atau tutup pintu air sampai posisi 0.
5	POS 1	0001000C91261857048978 000005D0E7141403	Perintah *user* terhadap pintu air untuk buka atau tutup pintu air sampai posisi 1.
6	POS 2	0001000C91261857048978 000005D0E7142403	Perintah *user* terhadap pintu air untuk buka atau tutup pintu air sampai posisi 2.
7	POS 3	0001000C91261857048978 000005D0E7143403	Perintah *user* terhadap pintu air untuk buka atau tutup pintu air sampai posisi 3.
8	Tinggi	0001000C91261857048978 000007D4B4FB7C4EEB00	SMS balasan dari *system* ketinggian air, jika user member perintah untuk meminta status ketinggian air (REQST)

BAB V
PENUTUP

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan pembuatan alat "Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS *Gateway*" dapat diambil kesimpulan sebagai berikut :

1. Konsep komunikasi data menggunakan *SMS gateway* dapat dijadikan referensi pada keadaan sesungguhnya karena pegiriman dan penerimaan data dalam masa percobaan tidak mengalami kesalahan.
2. Pengiriman data secara otomatis tidak effisien, karena pada kenyataanya perubahan ketinggian air bisa terjadi dalam waktu sangat cepat karena kondisi sekitar yang berubah-ubah.
3. Pengujian pada sensor ketinggian air sungai menggunakan pelampung yang menggunakan pelampung yang mempunyai variabel resistor sebagai pendeteksinya karena pelampung ini memiliki sensitifitas yang cukup baik, apabila air naik ataupun turun maka pelampung tentunya juga ikut bergerak dan mempengaruhi perubahan kondisi air.
4. Respon pintu air berdasarkan sensor ketinggian sangat dipengaruhi oleh kecepatan pengiriman dan penerimaan data. Akan tetapi respon dengan menggunakan komunikasi data *SMS gateway* sangat effisien daripada dengan cara konvesional, karena tidak membutuhkan waktu dan tenaga yang cukup besar, *error* yang terjadi setelah sms diterima sebesar 0,8 detik.

5.2 Saran

Saran-saran yang dapat diberikan sehubungan dengan isi dari laporan skripsi ini adalah sebagai berikut :

1. Pemakaian sensor ketinggian air menggunakan potensiometer dinilai kurang effisien, karena perubahan resistansi sangat banyak terpengaruh dari suhu, kelembaban dan kebersihan pada air itu sendiri.
2. Pemilihan *provider telephone* yang tepat sangat berpengaruh dengan kecepatan pengiriman dan penerimaan data menggunakan *SMS*.
3. Untuk penerapan dengan kompleksitas permasalahan yang lebih besar (lebih tinggi) belum bisa maksimal.
4. Masih belum bisa diterapkan pada kenyataannya, karena butuh pembenahan (masih *prototype*).

DAFTAR PUSTAKA

DAFTAR PUSTAKA

- Ir. Bustam Khang. "*Trik Pemrograman Aplikasi Berbasis SMS*". Jakarta : PT Elex Media Komputindo. Jakarta, 2002.

- Putra, Agfianto Eko, *"Belajar Mikrokontroller AT89C51/C52/C55"*. ANDI, 2003.

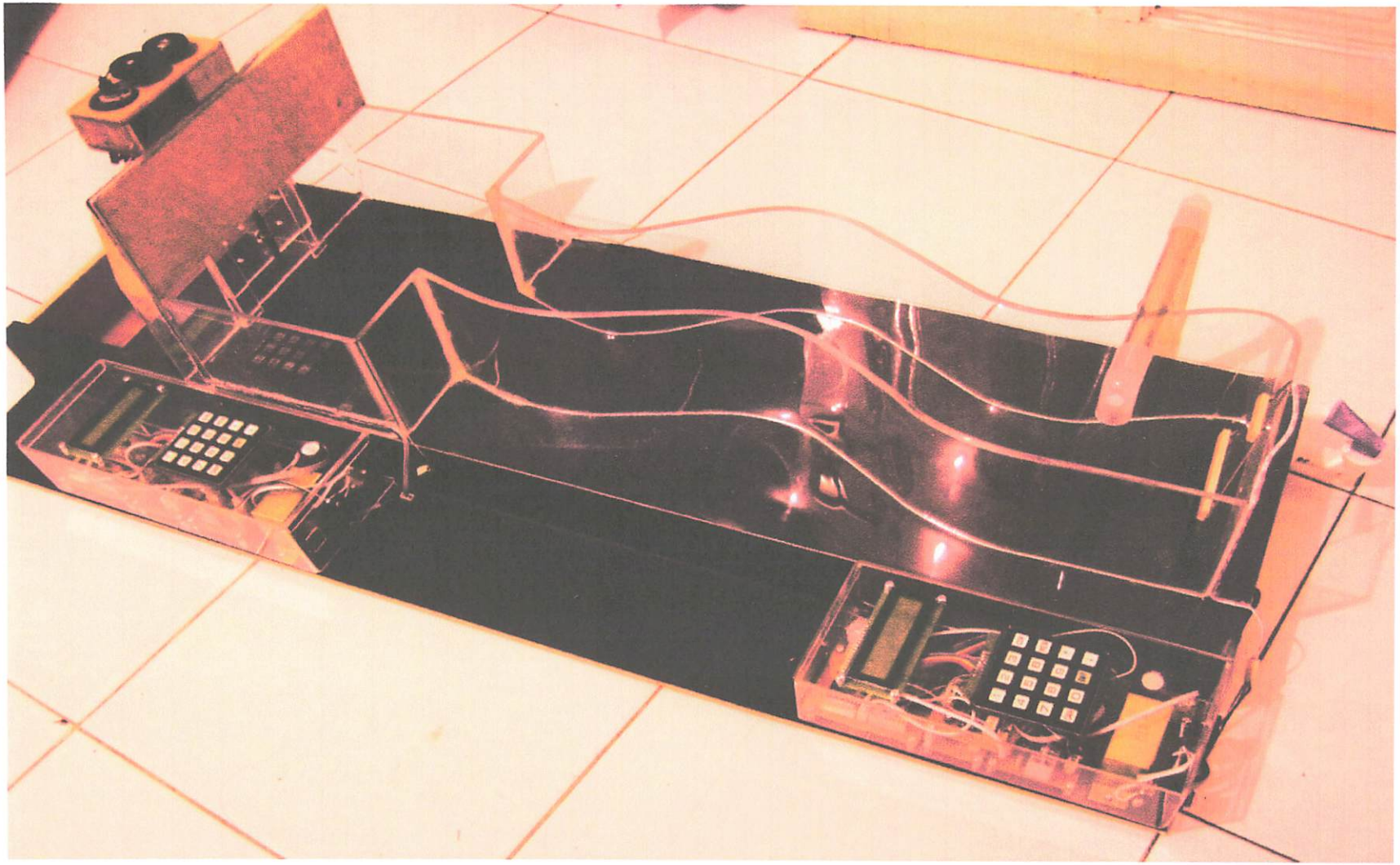
- Totok Budioko, *"Belajar dengan mudah dan cepat Pemrograman Bahasa C dengan SDCC pada Mikrokontroller AT89S52"*. GAVA MEDIA Yogyakarta, Maret 2005.

- ST Microelectronics (1999) "*16 Kbit Serial I2C Bus EEPROM with User-Defined Block Write Protection*", Diakses 19 Januari 2011 dari All datasheet. URL:http://www.alldatasheet.com/datasheetpdf/pdf/23725/STMICROELECTRONICS/24C16.html.

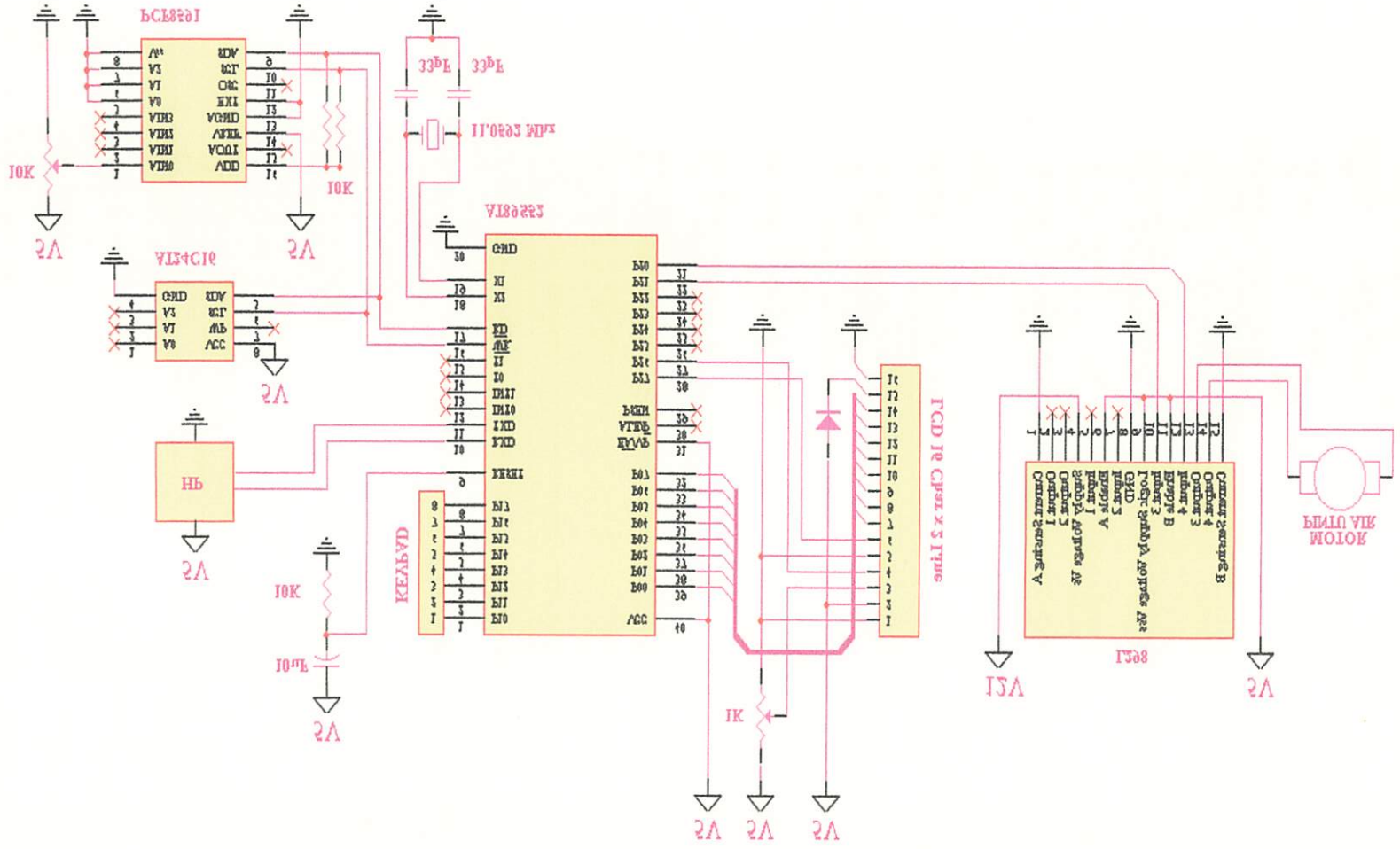
- ST Microelectronics (2000) "*DUAL FULL-BRIDGE DRIVER*". Diakses 19 Januari 2011 dari Alldatasheet.
URL:http://www.alldatasheet.com/datasheetpdf/pdf/22437/STMICROELECTRONICS/ L298.html.

- NXP Semiconductors (2 juli 1998) "*8-bit A/D and D/A converter*". 19 Januari 2011 dari All datasheet.
URL:http://www.alldatasheet.com/datasheetpdf/pdf/18222/PHILIPS/PCF8591.html.

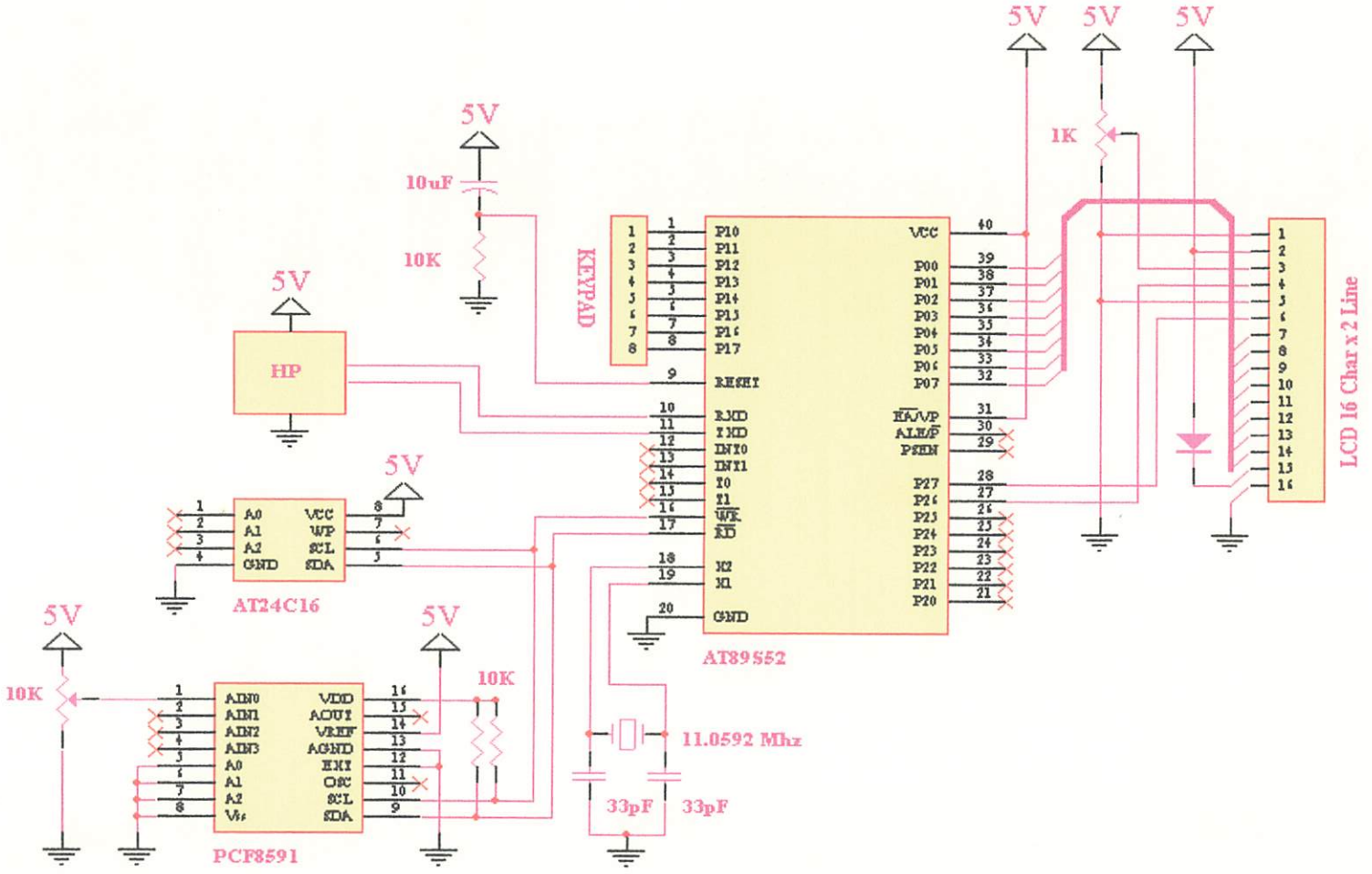
LAMPIRAN



Pengontrol Dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway



Schematic System Pintu Air



Schematic Sensor Ketinggian Air

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

LEMBAR PERBAIKAN SKRIPSI

Nama Mahasiswa : Andri Dwi Arianto
NIM : 09.12.902
Jurusan : Teknik Elektro S1
Konsentrasi : Teknik Elektronika
Hari / Tanggal : Selasa / 15 Maret 2011

No	Materi Perbaikan	Paraf
1.	Rancangan Driver Motor	
2.	Analisa Rancangan Driver Motor	

Telah Diperiksa / Disetujui :

Penguji Pertama

Sotyohadi, ST
NIP.Y. 1039700309

Mengetahui,

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP.Y. 1030100361

Dosen Pembimbing II

Irmalia Suryani F. ST, MT
NIP. P. 1030000365

LEMBAR PERBAIKAN SKRIPSI

Nama Mahasiswa : Andri Dwi Arianto

NIM : 09.12.902

Jurusan : Teknik Elektro S1

Konsentrasi : Teknik Elektronika

Hari / Tanggal : Selasa / 14 Maret 2011

No	Materi Perbaikan	Paraf
1.	Rancangan Driver Motor	
2.	Analisa Rancangan Driver Motor	

Telah Diperiksa / Disetujui :

Penguji Pertama

Sotyohadi, ST
NIP.Y. 1039700309

Mengetahui,

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP.Y. 1030100361

Dosen Pembimbing II

Irmalia Suryani F, ST, MT
NIP.P. 1030000365

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

LEMBAR PERBAIKAN SKRIPSI

Nama Mahasiswa	: Andri Dwi Arianto
NIM	: 09.12.902
Jurusan	: Teknik Elektro S1
Konsentrasi	: Teknik Elektronika
Hari / Tanggal	: Selasa / 15 Maret 2011

No	Materi Perbaikan	Paraf
1	Penjelasan Kenapa Perlu Eksternal EPPROM	
2.	Pengujian Perlu Detail, Prosedur dan Penjelasan Pengujian	
3.	Memperbaiki Gambar 3.13	

Telah Diperiksa / Disetujui :

Penguji Kedua

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y. 1030800417

Mengetahui,

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP. Y. 1030100361

Dosen Pembimbing II

Irmalia Suryani F. ST, MT
NIP. P. 1030000365

LEMBAR PERBAIKAN SKRIPSI

Nama Mahasiswa : Andri Dwi Arianto

NIM : 06.12.902

Jurusan : Teknik Elektro S1

Konsentrasi : Teknik Elektronika

Hari / Tanggal : Selasa / 14 Maret 2011

No	Materi Perbaikan	Paraf
1	Penjelasan Kenapa Perlu Eksternal EPPROM	
2.	Pengujian Perlu Detail, Prosedur dan Penjelasan Pengujian	
3.	Memperbaiki Gambar 3.13	

Telah Diperiksa / Disetujui :

Penguji Kedua

Dr. Eng. Aryuanto Soetedjo, ST, MT
NIP.Y. 1030800417

Mengetahui,

Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP. Y. 1030100361

Dosen Pembimbing II

Irmalia Suryani F, ST, MT
NIP. P. 1030000365

INSTITUT TEKNOLOGI NASIONAL
Jl. Raya Karanglo Km 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Andri Dwi Arianto
NIM : 09.12.902
Masa Bimbingan : 20 Desember 2010 s/d 20 Juni 2011
Judul : Perancangan dan Pembuatan Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway

NO	Tanggal	Uraian	Paraf Pembimbing
1	26-01-2011	Gambar Alat Dilampirkan	
2	26-01-2011	Memperbaiki Diagram Blok	
3	26-01-2011	Memperbaiki dan Menambah Flowchart	
4	26-01-2011	Melengkapi I/O Dalam Perancangan Rangkaian	
5	27-01-2011	Menambah Skema Provider Dengan Komunikasi HP	
6	27-01-2011	Memperbaiki Schematic Rangkaian LCD	
7	27-01-2011	Memperbaiki Kesimpulan	
8			
9			

Malang, 2011
Dosen Pembimbing I

I Komang Somawirata, ST, MT
NIP. Y. 1030100361

Form S-4b

INSTITUT TEKNOLOGI NASIONAL
Jl. Raya Karanglo Km 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Andri Dwi Arianto
NIM : 09.12.902
Masa Bimbingan : 20 Desember 2010 s/d 20 Juni 2011
Judul : Perancangan dan Pembuatan Pengontrol dan Monitoring Ketinggian Level Air Pada Suatu Bendungan Menggunakan SMS Gateway

NO	Tanggal	Uraian	Paraf
1	20-01-2011	Perencanaan Awal Sistem Kerja	
2	20-01-2011	Memperbaiki Susunan Antar Sub Bab	
3	20-01-2011	Diagram Blok Sistem	
4	24-01-2011	Menjelaskan Fungsi Masing-Masing Blok Diagram	
5	24-01-2011	Memperbaiki Cara Kerja Keseluruhan Sistem	
6	24-01-2011	Melengkapi I/O Dalam Perancangan Rangkaian	
7			
8			
9			

Malang, 2011
Dosen Pembimbing II

Irmalia Suryani F. ST, MT
NIP. P. 1030000365

Form S-4b

LISTING PROGRAM

Program Buka Tutup Pintu Air


```
org     00h
ljmp    init

org     23h
clr     ES
jnb     RI,$
clr     RI
mov     R7,SBUF
setb    ES
reti

ISDA    Bit P2.2                ; I2C data
ISCL    Bit P2.3                ; I2C clock
Rest    Bit P2.6
Enbl    Bit P2.7

Sttc    Bit 20h.0               ; status command

Ch00    Equ 30h                 ;\
Ch01    Equ 31h                 ; |
Ch02    Equ 32h                 ; |
Ch03    Equ 33h                 ; |
Ch04    Equ 34h                 ; |
Ch05    Equ 35h                 ; |
Ch06    Equ 36h                 ; |
Ch07    Equ 37h                 ; | nama memory (character)
Ch08    Equ 38h                 ; | untuk baca no hp & sms
Ch09    Equ 39h                 ; |
Ch0A    Equ 3Ah                 ; |
Ch0B    Equ 3Bh                 ; |
Ch0C    Equ 3Ch                 ; |
Ch0D    Equ 3Dh                 ; |
Ch0E    Equ 3Eh                 ; |
Ch0F    Equ 3Fh                 ;/

Nhp0    Equ 40h                 ;\
Nhp1    Equ 41h                 ; |
Nhp2    Equ 42h                 ; |
Nhp3    Equ 43h                 ; |
Nhp4    Equ 44h                 ; | nama memory (character)
Nhp5    Equ 45h                 ; | untuk no HP
Nhp6    Equ 46h                 ; |
Nhp7    Equ 47h                 ; |
Nhp8    Equ 48h                 ; |
Nhp9    Equ 49h                 ; |
NhpA    Equ 4Ah                 ; |
NhpB    Equ 4Bh                 ;/
Tmo0    Equ 4Ch                 ; data time out 0
Tmo1    Equ 4Dh                 ; data time out 1
Cnsm    Equ 4Eh                 ; counter sms
Cndt    Equ 4Fh                 ; counter data sms

ChrL    Equ 50h
ChrH    Equ 51h
Dsts    Equ 52h
Dadc    Equ 53h
Dktg    Equ 54h
Dtmo    Equ 55h

Anhp    Equ 60h
Admm    Equ 61h
Dtmm    Equ 62h
Cntr    Equ 63h
Bufr    Equ 64h
Buf0    Equ 65h
Buf1    Equ 66h
Buf2    Equ 67h
Dly0    Equ 68h
```

```
        Dly1    Equ 69h
        Dly2    Equ 6Ah
        Dly3    Equ 6Bh

nit:    lcall   lcd_in
        mov     DPTR,#tpinls
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tphnph
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   rdmsts                  ; baca memory status
        lcall   rdmtmo                  ; baca memory time out
        mov     Anhp,#010h              ; baca memory no hp
        lcall   rdmnhp
        mov     Dly3,#10
        lcall   delay3

        lcall   srl_in                  ; serial 19200,8,N,1
        lcall   noecho                  ; no respon character
        lcall   dsminc                  ; display incoming sms
        lcall   rstcmd                  ; reset command

ulai:   mov     DPTR,#tpnama
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tpnims
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   delay2
        mov     DPTR,#tpjurs
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tpuniv
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   delay2

ksts0:  mov     A,Dsts
        cjne    A,#1,cksts1
        mov     DPTR,#tptdak
ksts1:  cjne    A,#2,cksts2
        mov     DPTR,#tpsdak
ksts2:  lcall   line1
        mov     ChrL,#16
        lcall   tulis

nttmo:  mov     DPTR,#tptmdt
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        mov     Dly0,#0
        mov     Dly1,#249
        mov     Dly2,#7
        mov     Dly3,Dtmo
ttmo0:  mov     DPTR,#angka
        mov     P0,#0C9h
        lcall   w_ins
        mov     A,Dly3
        lcall   nilai
        mov     P0,#0D0h
        lcall   w_ins
```

```
:ttmo1: djnz    Dly0,cttmo1
        mov     Dly0,#0
        djnz    Dly1,cttmo1
        mov     Dly1,#249
        djnz    Dly2,cttmo0
        mov     Dly2,#7

:ksms0: cjne    R7,#0FFh,cksms1         ;\
        ljmp    cksms2                  ; | cek
:ksms1: mov     Dly3,#10                ; |
        lcall   delay3                  ; |
        lcall   bc_sms                  ; | ada request
        lcall   rstcmd                  ;/
        ljmp    cksts0
:ksms2: djnz    Dly3,cttmo0
        lcall   tkrsms                  ;\
        lcall   bc_sns                  ; | baca ketinggian
        lcall   cm_air                  ; | kirim sms ketinggian
         lcall   kr_ktg                 ;/
:kprb0: mov     A,Dsts
        cjne    A,#2,ckprb1
        lcall   ckprbh
:kprb1: ljmp    cksts0

:kprbh: mov     A,Dktg                  ; cek perubahan
        mov     B,#48                   ;\
        div     AB                      ; | if dibawah 48
        jnz     cprbh0                  ;/
        lcall   krknd0                  ; kirim sms
        ljmp    cprbh3                  ; lompat
prbh0: mov      A,Dktg                  ; cek perubahan
        mov     B,#64                   ;\
        div     AB                      ; | if dibawah 64
        jnz     cprbh1                  ;/
        lcall   krknd1                  ; kirim sms
        ljmp    cprbh3                  ; lompat
prbh1: mov      A,Dktg                  ; cek perubahan
        mov     B,#80                   ;\
        div     AB                      ; | if dibawah 80
        jnz     cprbh2                  ;/
        lcall   krknd2                  ; kirim sms
        ljmp    cprbh3                  ; lompat
prbh2: mov      A,Dktg                  ; cek perubahan
        mov     B,#96                   ;\
        div     AB                      ; | if dibawah 96
        jnz     cprbh1                  ;/
        lcall   krknd3                  ; kirim sms
prbh3: ret

sinhp: lcall    lcdclr
        mov     DPTR,#tpnohp
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
snhp0: mov      DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        mov     A,Nhp0
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp1
        lcall   cknhpn
```

```
        lcall   wr_chr
        mov     A,Nhp2
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp3
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp4
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp5
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp6
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp7
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp8
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp9
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpA
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpB
        lcall   cknhpn
        lcall   wr_chr
        mov     P0,#0D0h
        lcall   w_ins
        lcall   tg_lps
snhp1:  lcall   scnkpd
        cjne    R0,#11,isnhp2
        mov     SP,#07h
        ljmp    mulai
snhp2:  cjne    R0,#12,isnhp3
        ljmp    isnhp4
snhp3:  cjne    R0,#16,isnhp1
        ljmp    stgtmo

snhp4:  mov     DPTR,#kosong
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   rsnohp
        lcall   tg_lps
        mov     DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        lcall   tginhp
        mov     Nhp0,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp1,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
```

```
        lcall   tginhp
        mov     Nhp2,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp3,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp4,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp5,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp6,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp7,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp8,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp9,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpA,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpB,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
snhp5:  lcall   wrmnhp
        ljmp    isnhp0

ginhp:  lcall   scnkpd
        lcall   delay0
inhp0:  cjne    R0,#16,tinhp1
        ljmp    tginhp
inhp1:  cjne    R0,#15,tinhp2
        ljmp    tginhp
inhp2:  cjne    R0,#14,tinhp3
        ljmp    tginhp
inhp3:  cjne    R0,#13,tinhp4
        ljmp    tginhp
inhp4:  cjne    R0,#12,tinhp5
        ljmp    isnhp5
inhp5:  cjne    R0,#11,tinhp6
        mov     SP,#07h
        ljmp    mulai
```

```
cinhp6: cjne    R0,#10,tinhp7
        ljmp    tginhp
cinhp7: ret

rsnohp: mov     Nhp0,#00Fh
        mov     Nhp1,#00Fh
        mov     Nhp2,#00Fh
        mov     Nhp3,#00Fh
        mov     Nhp4,#00Fh
        mov     Nhp5,#00Fh
        mov     Nhp6,#00Fh
        mov     Nhp7,#00Fh
        mov     Nhp8,#00Fh
        mov     Nhp9,#00Fh
        mov     NhpA,#00Fh
        mov     NhpB,#00Fh
        ret

cknhpn: cjne    A,#00Fh,cnhpn           ; cek angka no hp
        mov     A,#16                   ; if F then chr ' '
cnhpn:  ret

stgtmo: lcall   lcdclr                  ; seting time out
        mov     DPTR,#tpstmo
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
sttmo0: mov     DPTR,#angka
        mov     P0,#0C5h
        lcall   w_ins
        mov     A,Dtmo
        lcall   nilai
        mov     P0,#' '
        lcall   w_chr
        mov     P0,#'d'
        lcall   w_chr
        mov     P0,#'t'
        lcall   w_chr
        mov     P0,#'k'
        lcall   w_chr
        mov     P0,#0D0h
        lcall   w_ins
        lcall   tg_lps
sttmo1: lcall   scnkpd
        cjne    R0,#10,sttmo2
        ljmp    sttmo1
sttmo2: cjne    R0,#11,sttmo3
        mov     SP,#07h
        ljmp    mulai
sttmo3: cjne    R0,#12,sttmo4
        ljmp    sttmo1
sttmo4: cjne    R0,#13,sttmo5
        ljmp    sttmo1
sttmo5: cjne    R0,#14,sttmo6
        ljmp    sttmo1
sttmo6: cjne    R0,#15,sttmo7
        ljmp    isinhp
sttmo7: cjne    R0,#16,sttmo8
        ljmp    sttmo1
sttmo8: mov     P0,#0C5h
        lcall   w_ins
        mov     Buf0,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tg_tkn
        mov     Buf1,R0
        mov     A,R0
```

```
        lcall   wr_chr
        lcall   tg_lps
        lcall   tg_tkn
        mov     Buf2,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        mov     A,Buf0
        mov     B,#100
        mul     AB
        mov     Buf0,A
        mov     A,Buf1
        mov     B,#10
        mul     AB
        mov     B,Buf2
        add     A,B
        mov     B,Buf0
        add     A,B
        mov     Dtmo,A
        lcall   wrmtmo
        ljmp    sttmo0

:albrs: lcall   lcdclr
        mov     DPTR,#tpadkt
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        lcall   tg_lps
:albr0: lcall   bc_sns
        mov     DPTR,#angka
        mov     P0,#0C2h
        lcall   w_ins
        mov     A,Dadc
        lcall   nilai
        mov     P0,#0CAh
        lcall   w_ins
        mov     A,Dktg
        mov     B,#100
        div     AB
        mov     A,B
        mov     B,#10
        div     AB
        lcall   wr_chr
        mov     P0,#'.'
        lcall   w_chr
        mov     A,B
        lcall   wr_chr
        mov     P0,#'c'
        lcall   w_chr
        mov     P0,#'m'
        lcall   w_chr
        mov     P0,#0D0h
        lcall   w_ins

        mov     Dly3,#1
        lcall   delay3
albr1:  lcall   scnkpd
        cjne    R0,#11,kalbr0
        mov     SP,#07h
        ljmp    mulai

rl_in:  mov     Dly3,#1                 ;\
        lcall   delay3                  ; |
        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    EA                      ; |
        setb    ES                      ; |
        ret                             ;/

:r_ins: clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim instruksi ke hp
        djnz    ChrH,kr_ins             ; | sebagai awal pengiriman data
        mov     A,#0Dh                  ; |
        lcall   kr_srl                  ; |
        ret                             ;/

:r_dta: clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim data olahan (PDU)
        djnz    ChrH,kr_dta             ; | ke HP
        mov     A,#26                   ; |
        lcall   kr_srl                  ; |
        ret                             ;/

:r_hlf: clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; | kirim setengah data olahan (PDU)
        inc     DPTR                    ; | yang sudah pasti ke HP
        djnz    ChrH,kr_hlf             ; |
        ret                             ;/

:r_srl: clr     ES                      ;\
        mov     SBUF,A                  ; |
        jnb     TI,$                    ; |
        clr     TI                      ; | kirim serial ke hp
        setb    ES                      ; |
        djnz    Dly0,$                  ; |
        ret                             ;/

oecho:  mov     Dly3,#1                 ;\
        lcall   delay3                  ; |
        mov     DPTR,#smseco            ; | kirim
        mov     ChrH,#4                 ; | instruksi
        lcall   kr_ins                  ; | no-echo ke hp
        lcall   bc_fbk                  ; |
        ret                             ;/

sminc:  mov     Dly3,#1                 ;\
        lcall   delay3                  ; |
        mov     DPTR,#smsfrm            ; | kirim instruksi
        mov     ChrH,#17                ; | display-incoming-sms
        lcall   kr_ins                  ; | ke hp
        lcall   bc_fbk                  ; |
        ret                             ;/

pssms:  mov     DPTR,#smshps            ;\
        mov     ChrH,#9                 ; |
        lcall   kr_ins                  ; |
        lcall   bc_fbk                  ; | hapus sms pada memory 1
        mov     Dly3,#1                 ; |
        lcall   delay3                  ; |
        ret                             ;/

c_fbk:  lcall   rstcmd                  ;\
c_fb0:  cjne    R7,#0FFh,bc_fb1         ; | tunggu feedback
        ljmp    bc_fb0                  ; | data dari hp
c_fb1:  ret                             ;/

c_srl:  lcall   rstcmd                  ;\
```

```
        mov     Tmo0,#25                ;  |
        mov     Tmo1,#0                 ;  | baca serial
)c_sr0: cjne    R7,#0FFh,bc_sr1         ;  | tunggu data bukan FF
        djnz    Tmo1,bc_sr0             ;  | sebelum time out
        djnz    Tmo0,bc_sr0             ;  |
)c_sr1: ret                             ;/

'stcmd: mov     R7,#0FFh
        ret

(rnohp: mov     A,Nhp1
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp0
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp3
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp2
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp5
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp4
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp7
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp6
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp9
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,Nhp8
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,NhpB
        lcall   ck_ksg
        mov     A,NhpA
        lcall   ck_ksg
        ret

k_ksg:  cjne    A,#00Fh,ckksg0
        mov     A,#'F'
        ljmp    ckksg1
kksg0:  mov     B,#30h
        add     A,B
kksg1:  lcall   kr_srl
        ret

krsms:  lcall   lcdclr                  ; tulis kirim sms
        mov     DPTR,#tpkrsm
        lcall   line1
        mov     ChrL,#16
```

```
        lcall   tulis
        mov     Dly3,#1
        lcall   delay3
        ret

c_sms:  lcall   bcnohp                  ;\
        jnb     Sttc,bcsms0             ; | baca no HP
        lcall   bccmnd                  ; | cek status
        lcall   hpssms                  ; | 0=hapus sms, 1=baca command
        ljmp    bcsms1                  ; | hapus sms
csms0:  mov     A,Ch00                  ; |
        cjne    A,#10,bcsmsx            ; | if tdk ada no -> tdk hps
        ljmp    bcsmsB                  ; |
csmsx:  lcall   hpssms                  ; | cek command
        ljmp    bcsmsB                  ;/

csms1:  mov     DPTR,#comnd0            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsms2             ; | cek command
        lcall   bc_sns                  ; |
        lcall   cm_air                  ; |
        lcall   tkrsms                  ; | kirim sms ketinggian
        lcall   kr_ktg                  ; |
        ljmp    bcsmsB                  ;/

csms2:  mov     DPTR,#comnd1            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsms3             ; | cek command
        mov     Dsts,#2                 ; | status = 2
        lcall   wrmsts                  ; |
        ljmp    bcsmsB                  ;/

csms3:  mov     DPTR,#comnd2            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsmsB             ; | cek command
        mov     Dsts,#1                 ; |
        lcall   wrmsts                  ; | status = 1
        ljmp    bcsmsB                  ;/

csmsB:  ret

cnohp:  mov     DPTR,#smsred            ;\
        mov     ChrH,#9                 ; |
        lcall   kr_ins                  ; |
        mov     Cntr,#30                ; | baca command
        lcall   bc_fbk                  ; |
cnhp0:  lcall   bc_srl                  ; | baca serial
        cjne    R7,#0Dh,bcnhp1          ; | sampai character '_|'
        ljmp    bcnhp2                  ; |
cnhp1:  djnz    Cntr,bcnhp0             ; |
        ljmp    cknhp2                  ;/
cnhp2:  lcall   bc_srl                  ;\
        lcall   bc_srl                  ; |
        lcall   bc_srl                  ; |
        clr     C                       ; |
        mov     A,R7                    ; |
        mov     B,#30h                  ; | lewati service center
        subb    A,B                     ; |
        mov     B,#2                    ; | ambil character -> decimal
        mul     AB                      ; | lewati character sebanyak
        mov     B,#8                    ; | hasil kali 2 tambah 8
        add     A,B                     ; |
        mov     Cntr,A                  ; |
cnhp3:  lcall   bc_srl                  ; |
        djnz    Cntr,bcnhp3             ;/
        lcall   bc_dta                  ; baca data
         lcall   line1                  ; optional
         lcall   tlsdta                 ; optional
```

```
;        lcall   delay2                  ; optional
;
cknohp: mov     DPTR,#angka             ;\
        mov     Cntr,#10                ; | jumlah no hp = 10
        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 selanjutnya
        djnz    Cntr,cknhp0             ; | cek sampai no hp habis
        setb    Sttc                    ; | set status = 1
        ljmp    cknhp2                  ; | jika tidak
cknhp1: clr     Sttc                    ; | set status = 0
cknhp2: ret                             ;/

ltanhp: mov     A,Cndt
        cjne    A,#00,dnhp00
        mov     A,Nhp1
        ljmp    dnhp11
lnhp00: cjne    A,#01,dnhp01
        mov     A,Nhp0
        ljmp    dnhp11
lnhp01: cjne    A,#02,dnhp02
        mov     A,Nhp3
        ljmp    dnhp11
lnhp02: cjne    A,#03,dnhp03
        mov     A,Nhp2
        ljmp    dnhp11
lnhp03: cjne    A,#04,dnhp04
        mov     A,Nhp5
        ljmp    dnhp11
nhp04:  cjne    A,#05,dnhp05
        mov     A,Nhp4
        ljmp    dnhp11
nhp05:  cjne    A,#06,dnhp06
        mov     A,Nhp7
        ljmp    dnhp11
nhp06:  cjne    A,#07,dnhp07
        mov     A,Nhp6
        ljmp    dnhp11
nhp07:  cjne    A,#08,dnhp08
        mov     A,Nhp9
        ljmp    dnhp11
nhp08:  cjne    A,#09,dnhp09
        mov     A,Nhp8
        ljmp    dnhp11
nhp09:  cjne    A,#10,dnhp10
        mov     A,NhpB
        ljmp    dnhp11
nhp10:  cjne    A,#11,dnhp12
        mov     A,NhpA
nhp11:  cjne    A,#00Fh,dnhp12
        mov     A,#15
nhp12:  movc    A,@A+DPTR
        ret

ccmnd:  mov     DPTR,#smsred            ;\
        mov     ChrH,#9                 ; |
        lcall   kr_ins                  ; |
        mov     Cntr,#50                ; | baca command
        lcall   bc_fbk                  ; |
ccmd0:  lcall   bc_srl                  ; | baca serial
        cjne    R7,#0Dh,bccmd1          ; | if character '_|'
        ljmp    bccmd2                  ; |
```

```
)ccmd1: djnz    Cntr,bccmd0          ; |
        ljmp    bccmdB               ;/
)ccmd2: lcall   bc_srl               ;\
        lcall   bc_srl               ; |
        lcall   bc_srl               ; |
        clr     C                    ; |
        mov     A,R7                 ; |
        mov     B,#30h               ; | lewati service center
        subb    A,B                  ; |
        mov     B,#2                 ; | ambil character -> decimal
        mul     AB                   ; | lewati character sebanyak
        mov     B,#3                 ; | hasil kali 2 tambah 3
        add     A,B                  ; |
        mov     Cntr,A               ; |
)ccmd3: lcall   bc_srl               ; |
        djnz    Cntr,bccmd3          ;/
        lcall   bc_srl               ;\
)ccmd4: cjne    R7,#'A',bccmd5       ; |
        mov     Cntr,#30             ; |
)ccmd5: cjne    R7,#'B',bccmd6       ; | cek jumlah no HP
        mov     Cntr,#32             ; |
)ccmd6: cjne    R7,#'C',bccmd7       ; | if A -> lewati 30 character
        mov     Cntr,#32             ; | if B -> lewati 32 character
)ccmd7: cjne    R7,#'D',bccmd8       ; | if C -> lewati 32 character
        mov     Cntr,#34             ; | if D -> lewati 34 character
)ccmd8: cjne    R7,#'E',bccmd9       ; | if E -> lewati 34 character
        mov     Cntr,#34             ; |
)ccmd9: lcall   bc_srl               ; |
        djnz    Cntr,bccmd9          ;/
)ccmdA: lcall   bc_srl               ;
        lcall   bc_srl               ;
        lcall   bc_dta               ; baca data
         lcall   line2               ; optional
         lcall   tlsdta              ; optional
         lcall   delay2              ; optional
)ccmdB: ret

:kcmnd: mov     Cntr,#10             ;\
        mov     Cnsm,#0              ; | jumlah command
        mov     Cndt,#0              ; | reset counter sms
:kcnd0: 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
elanjutnya
        djnz    Cntr,ckcnd0          ; | cek sampai command habis
        setb    Sttc                 ; | set status = 1
        ljmp    ckcnd2               ; | jika tidak
kcnd1:  clr     Sttc                 ; | set status = 0
kcnd2:  ret                          ;/

tacmd:  mov     A,Cndt               ;\
        movc    A,@A+DPTR            ; | ambil data pointer yang ke cndt
        ret                          ;/

tasms:  clr     C                    ;\
        mov     A,Cnsm               ; |
tsm00:  cjne    A,#00,dtsm01         ; |
        mov     B,Ch00               ; |
tsm01:  cjne    A,#01,dtsm02         ; |
        mov     B,Ch01               ; |
tsm02:  cjne    A,#02,dtsm03         ; |
        mov     B,Ch02               ; |
tsm03:  cjne    A,#03,dtsm04         ; |
        mov     B,Ch03               ; |
```

```
ltsm04: cjne    A,#04,dtsm05            ;  |
        mov     B,Ch04                  ;  |
ltsm05: cjne    A,#05,dtsm06            ;  |
        mov     B,Ch05                  ;  |
ltsm06: cjne    A,#06,dtsm07            ;  |
        mov     B,Ch06                  ;  |
ltsm07: cjne    A,#07,dtsm08            ;  |
        mov     B,Ch07                  ;  |  samakan data counter
ltsm08: cjne    A,#08,dtsm09            ;  |  dengan character
        mov     B,Ch08                  ;  |
ltsm09: cjne    A,#09,dtsm0A            ;  |
        mov     B,Ch09                  ;  |
ltsm0A: cjne    A,#10,dtsm0B            ;  |
        mov     B,Ch0A                  ;  |
ltsm0B: cjne    A,#11,dtsm0C            ;  |
        mov     B,Ch0B                  ;  |
ltsm0C: cjne    A,#12,dtsm0D            ;  |
        mov     B,Ch0C                  ;  |
ltsm0D: cjne    A,#13,dtsm0E            ;  |
        mov     B,Ch0D                  ;  |
ltsm0E: cjne    A,#14,dtsm0F            ;  |
        mov     B,Ch0E                  ;  |
ltsm0F: cjne    A,#15,dtsmsx            ;  |
        mov     B,Ch0F                  ;  |
ltsmsx: ret                             ;/

c_dta:  lcall   bc_srl                  ;\
        mov     Ch00,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch01,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch02,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch03,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch04,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch05,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch06,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch07,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch08,R7                 ;  |  baca serial & simpan
        lcall   bc_srl                  ;  |  ke memory character
        mov     Ch09,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0A,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0B,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0C,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0D,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0E,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0F,R7                 ;  |
        mov     Dly3,#5                 ;  |
        lcall   delay3                  ;  |
        ret                             ;/

lsdta:  mov     P0,Ch00                 ;\
        lcall   w_chr                   ;  |
        mov     P0,Ch01                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch02                 ;  |
        lcall   w_chr                   ;  |
```

```
        mov     P0,Ch03                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch04                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch05                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch06                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch07                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch08                 ;  | tulis character ke lcd
        lcall   w_chr                   ;  |
        mov     P0,Ch09                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0A                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0B                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0C                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0D                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0E                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0F                 ;  |
        lcall   w_chr                   ;  |
        ret                             ;/

rknd0:  mov     DPTR,#smskrm            ; ambil data PDU kirim 20 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        mov     DPTR,#cmdpn0            ; ambil data PDU "POS 0"
        mov     ChrH,#40                ; ATCommand 44 character
        lcall   kr_dta                  ; kirim data setengah yg sudah pasti
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        ret

rknd1:  mov     DPTR,#smskrm            ; ambil data PDU kirim 20 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        mov     DPTR,#cmdpn1            ; ambil data PDU "POS 1"
        mov     ChrH,#40                ; ATCommand 44 character
        lcall   kr_dta                  ; kirim data setengah yg sudah pasti
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        ret

rknd2:  mov     DPTR,#smskrm            ; ambil data PDU kirim 20 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        mov     DPTR,#cmdpn2            ; ambil data PDU "POS 2"
        mov     ChrH,#40                ; ATCommand 44 character
        lcall   kr_dta                  ; kirim data setengah yg sudah pasti
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
```

```
        ret

:rknd3: mov     DPTR,#smskrm            ; ambil data PDU kirim 20 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        mov     DPTR,#cmdpn3            ; ambil data PDU "POS 2"
        mov     ChrH,#40                ; ATCommand 44 character
        lcall   kr_dta                  ; kirim data setengah yg sudah pasti
        lcall   bc_fbk                  ; tunggu feedback dari HP
        lcall   delay1
        lcall   delay1
        ret

r_ktg:  mov     DPTR,#smskr0            ; ambil data PDU kirim 47 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        mov     Dly3,#1                 ;
        lcall   delay3                  ; wait
        mov     DPTR,#header            ;\
        mov     ChrH,#12                ; | kirim header
        lcall   kr_hlf                  ;/
        lcall   krnohp                  ; kirim no HP
        mov     DPTR,#tinggi            ; ambil data PDU
        mov     ChrH,#20                ; ATCommand 20 character
        lcall   kr_hlf                  ; kirim setengah data PDU
        lcall   kr_pdu                  ; kirim data diakhiri CTRL+Z
        lcall   bc_fbk                  ; tunggu feedback dari HP
        mov     Dly3,#2                 ;
        lcall   delay3                  ; wait
        ret

n_air:  mov     DPTR,#angka
        mov     A,Dktg
        mov     B,#10
        div     AB
        movc    A,@A+DPTR
        mov     Ch00,A
        mov     A,B
        movc    A,@A+DPTR
        mov     Ch01,A
        mov     Ch02,#'m'
        mov     Ch03,#'m'
        mov     Ch04,#' '
        lcall   jd_pdu
        ret

:_sns:  mov     A,#90h                  ; address PCF8591 write address
        lcall   adrtx                   ; kirim
        mov     A,#00h                  ; adc ch-0
        lcall   dtatx                   ; kirim
        mov     A,#91h                  ; address PCF8591 read address
        lcall   adrtx                   ; kirim
        lcall   dtarx                   ; baca data
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; baca data
        lcall   i2cstp                  ; i2c stop
        mov     Dadc,A                  ; simpan data sensor
        mov     DPTR,#lookup
        movc    A,@A+DPTR
        mov     Dktg,A                  ; simpan data kalibrasi
        ret

'msts:  mov     Admm,#000h              ; address memory
        mov     Dtmm,Dsts               ; data memory
```

```
        lcall   wr_mem
        ret

;
'dmsts: mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,#000h                 ; address hardware memory
        lcall   dtatx
        mov     A,#0A1h                 ; AT24C16 read address
        lcall   adrtx
        lcall   dtarx                   ; terima data
'dmst0: cjne    A,#1,rdmst1
        ljmp    rdmst3
'dmst1: cjne    A,#2,rdmst2
        ljmp    rdmst3
'dmst2: mov     A,#1
'dmst3: mov     Dsts,A
        ret

'rmtmo: mov     Admm,#001h              ; address memory
        mov     Dtmm,Dtmo               ; data memory
        lcall   wr_mem
        ret

'dmtmo: mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,#001h                 ; address hardware memory
        lcall   dtatx
        mov     A,#0A1h                 ; AT24C16 read address
        lcall   adrtx
        lcall   dtarx                   ; terima data
        mov     Dtmo,A
        ret

'rmnhp: mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp1               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp0               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp3               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp2               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp5               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp4               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp7               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp6               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp9               ; data memory
        lcall   wr_mem
```

```
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp8               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,NhpB               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,NhpA               ; data memory
        lcall   wr_mem
        ret

dmnhp:  mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,Anhp                  ; address memory
        lcall   dtatx
        mov     A,#0A1h                 ; AT24C16 read address
        lcall   adrtx
        lcall   dtarx                   ; terima data
        mov     Nhp1,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp0,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp3,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp2,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp5,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp4,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp7,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp6,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp9,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp8,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     NhpB,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     NhpA,A
        lcall   i2cstp                  ; i2c stop
        ret

r_mem:  mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,Admm                  ; address hardware memory
        lcall   dtatx
        mov     A,Dtmm                  ; data memory
        lcall   dtatx
        lcall   i2cstp                  ; i2c stop
        lcall   wt_wr
        ret
```

```
rt_wr:  mov     Dly0,#000
        mov     Dly1,#075
rtwr:   lcall   delay0
        djnz    Dly1,wtwr
        ret

idrtx:  lcall   i2cstr                  ; kirim address
        lcall   putbit                  ; kirim data
        ret                             ; back

ltatx:  lcall   putbit                  ; kirim data
        ret                             ; back

ltarx:  lcall   getbit                  ; terima data
        ret                             ; back

utbit:  mov     R6,#8                   ;\
utbt:   RLC     A                       ; |
        mov     ISDA,C                  ; |
        setb    ISCL                    ; |
        clr     ISCL                    ; | kirim bit
        djnz    R6,putbt                ; |
        setb    ISDA                    ; |
        lcall   getack                  ; |
        ret                             ;/

etbit:  mov     R6,#8                   ;\
etbt:   setb    ISCL                    ; |
        mov     C,ISDA                  ; |
        RLC     A                       ; | terima bit
        clr     ISCL                    ; |
        djnz    R6,getbt                ; |
        setb    ISDA                    ; |
        ret                             ;/

etack:  setb    ISDA                    ;\
        setb    ISCL                    ; |
ckbit:  mov     C,ISDA                  ; | tunggu ack
        jc      ackbit                  ; | D=1, C=1D=0, C=0
        clr     ISCL                    ; |
        ret                             ;/

ivack:  clr     ISDA                    ;\
        setb    ISCL                    ; |
        clr     ISCL                    ; | kirim ack -> D=0, C=1, C=0, D=1
        setb    ISDA                    ; |
        ret                             ;/

2cstr:  setb    ISCL                    ;\
        setb    ISDA                    ; |
        clr     ISDA                    ; | i2c start -> C=1, D=1, D=0, C=0
        clr     ISCL                    ; |
        ret                             ;/

2cstp:  clr     ISDA                    ;\
        setb    ISCL                    ; |
        setb    ISDA                    ; | i2c stop -> D=0, C=1, D=1, C=0
        clr     ISCL                    ; |
        ret                             ;/

ilai:   mov     B,#100
        div     AB
        lcall   wr_chr
        mov     A,B
        mov     B,#10
        div     AB
        lcall   wr_chr
        mov     A,B
```

```
        lcall   wr_chr
        ret

ine1:   mov     P0,#080h
        lcall   w_ins
        ret

ine2:   mov     P0,#0C0h
        lcall   w_ins
        ret

ulis:   clr     A
        lcall   wr_chr
        inc     DPTR
        djnz    ChrL,tulis
        ret

r_chr:  movc    A,@A+DPTR
        mov     P0,A
        lcall   w_chr
        ret

_ins:   clr     Enbl
        clr     Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret

_chr:   clr     Enbl
        setb    Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret

cd_in:  mov     Dly3,#1
        lcall   delay3
        mov     P0,#01h                 ; Display Clear
        lcall   w_ins
        mov     P0,#38h                 ; Function Set
        lcall   w_ins
        mov     P0,#0Dh                 ; Display On, Cursor, Blink
        lcall   w_ins
        mov     P0,#06h                 ; Entry Mode
        lcall   w_ins
        mov     P0,#02h                 ; Cursor Home
        lcall   w_ins
        ret

cdclr:  mov     P0,#01h                 ; Display Clear
        lcall   w_ins
        lcall   delay0
        lcall   delay0
        lcall   delay0
        ret

cnkpd:  mov     R0,#10
        lcall   delay0
ol1:    mov     P1,#11111110b
        mov     A,P1
1b1:    cjne    A,#11101110b,c1b2
        mov     R0,#1
1b2:    cjne    A,#11011110b,c1b3
        mov     R0,#2
1b3:    cjne    A,#10111110b,c1b4
        mov     R0,#3
1b4:    cjne    A,#01111110b,col2
```

```
        mov     R0,#13

:ol2:   mov     P1,#11111101b
        mov     A,P1
:2b1:   cjne    A,#11101101b,c2b2
        mov     R0,#4
:2b2:   cjne    A,#11011101b,c2b3
        mov     R0,#5
:2b3:   cjne    A,#10111101b,c2b4
        mov     R0,#6
:2b4:   cjne    A,#01111101b,col3
        mov     R0,#14

:ol3:   mov     P1,#11111011b
        mov     A,P1
:3b1:   cjne    A,#11101011b,c3b2
        mov     R0,#7
:3b2:   cjne    A,#11011011b,c3b3
        mov     R0,#8
:3b3:   cjne    A,#10111011b,c3b4
        mov     R0,#9
:3b4:   cjne    A,#01111011b,col4
        mov     R0,#15

:ol4:   mov     P1,#11110111b
        mov     A,P1
:4b1:   cjne    A,#11100111b,c4b2
        mov     R0,#11
:4b2:   cjne    A,#11010111b,c4b3
        mov     R0,#0
:4b3:   cjne    A,#10110111b,c4b4
        mov     R0,#12
:4b4:   cjne    A,#01110111b,back
        mov     R0,#16
ack:    ret

g_tkn:  lcall   scnkpd
        lcall   delay0
g_tk0:  cjne    R0,#16,tg_tk1
        ljmp    tg_tkn
g_tk1:  cjne    R0,#15,tg_tk2
        ljmp    tg_tkn
g_tk2:  cjne    R0,#14,tg_tk3
        ljmp    tg_tkn
g_tk3:  cjne    R0,#13,tg_tk4
        ljmp    tg_tkn
g_tk4:  cjne    R0,#12,tg_tk5
        ljmp    tg_tkn
g_tk5:  cjne    R0,#11,tg_tk6
        ljmp    tg_tkn
g_tk6:  cjne    R0,#10,tg_tk7
        ljmp    tg_tkn
g_tk7:  ret

g_lps:  lcall   scnkpd
        lcall   delay0
        cjne    R0,#10,tg_lps
        ret

elay0:  djnz    Dly0,delay0
        ret

elay1:  lcall   scnkpd
        cjne    R0,#13,dely10
        ljmp    isinhp
ely10:  cjne    R0,#14,dely11
        ljmp    kalbrs
ely11:  djnz    Dly1,delay1
```

```
        ret

lelay2: mov     Dly2,#20
lely2:  lcall   delay1
        djnz    Dly2,dely2
        ret

lelay3: lcall   delay0
        djnz    Dly1,delay3
        djnz    Dly3,delay3
        ret

pnama: DB       '   Andri  Dwi   '
pnims: DB       '  NIM: 0912902  '
pjurs: DB       ' Teknik Elektro '
puniv: DB       '   ITN Malang   '
pinls: DB       '  Inisialisasi  '
phnph: DB       '   Hand Phone   '
psdak: DB       '  Sedang Aktif  '
ptdak: DB       '  Tidak  Aktif  '
ptmdt: DB       'Time Out     dtk'
pnohp: DB       '    Nomor HP    '
pstmo: DB       'Setting Time Out'
padkt: DB       '  ADC     Ktgg  '
pkrsm: DB       '   Kirim  SMS   '

mseco: DB       'ATE0'                  ; instruksi tanpa echo
msfrm: DB       'AT+CNMI=1,1,0,0,1'     ; instruksi display incoming sms
msred: DB       'AT+CMGR=1'             ; instruksi baca SMS
mshps: DB       'AT+CMGD=1'             ; instruksi hapus sms
mskrm: DB       'AT+CMGS=19'            ; instruksi kirim respon
mskr0: DB       'AT+CMGS=25'            ; instruksi kirim kendali
inggi: DB       '00000CD4B4FB7C4EEB40'  ; tinggi:

mdpn0: DB       '0001000D91265852294403F8000005D0E7140403'      ; POS 0
mdpn1: DB       '0001000D91265852294403F8000005D0E7141403'      ; POS 1
mdpn2: DB       '0001000D91265852294403F8000005D0E7142403'      ; POS 2
mdpn3: DB       '0001000D91265852294403F8000005D0E7143403'      ; POS 3

omnd0: DB       'D262744A05'            ; REQST
omnd1: DB       'D3ECF4E904'            ; SYSON
omnd2: DB       'D3ECF46904'            ; SYSOF

eader: DB       '0001000D9126'

ngka:  DB       '0123456789ABCDEF'
osong: DB       '                '

r_pdu: mov      DPTR,#angka             ;\
       mov      R0,Ch00                 ; |
       lcall    cchpdu                  ; |
       mov      R0,Ch01                 ; |
       lcall    cchpdu                  ; |
       mov      R0,Ch02                 ; |
       lcall    cchpdu                  ; | kirim PDU
       mov      R0,Ch03                 ; |
       lcall    cchpdu                  ; |
       mov      A,#26                   ; |
       lcall    kr_srl                  ; |
       ret                              ;/

chpdu: mov      A,R0                    ;\
       anl      A,#0F0h                 ; |
       RR       A                       ; |
       RR       A                       ; |
       RR       A                       ; |
       RR       A                       ; | cacah character
       movc     A,@A+DPTR               ; | menjadi data hexa
```

```
        lcall   kr_srl                  ;|
        mov     A,R0                    ;|
        anl     A,#00Fh                 ;|
        movc    A,@A+DPTR               ;|
        lcall   kr_srl                  ;|
        ret                             ;/

d_pdu:  mov     A,Ch01                  ;\
        RR      A                       ;| data1 geser 7x
        anl     A,#080h                 ;| and dg 1000 0000
        orl     A,Ch00                  ;| or dg data0
        mov     Ch00,A                  ;/
        mov     A,Ch01                  ;\
        anl     A,#11111110b            ;|
        RR      A                       ;| data1 hilangkan bit0
        mov     Bufr,A                  ;| data1 geser kanan 1x
        mov     A,Ch02                  ;| simpan sementara
        RR      A                       ;| and dg 1100 0000
        RR      A                       ;| or dg bufr
        anl     A,#0C0h                 ;|
        orl     A,Bufr                  ;|
        mov     Ch01,A                  ;/
        mov     A,Ch02                  ;\
        anl     A,#11111100b            ;|
        RR      A                       ;|
        RR      A                       ;| data1 hilangkan bit0 & bit1
        mov     Bufr,A                  ;| data1 geser kanan 2x
        mov     A,Ch03                  ;| simpan sementara
        RR      A                       ;| and dg 1110 0000
        RR      A                       ;| or dg bufr
        RR      A                       ;|
        anl     A,#0E0h                 ;|
        orl     A,Bufr                  ;|
        mov     Ch02,A                  ;/
        mov     A,Ch03                  ;\
        anl     A,#11111000b            ;|
        RR      A                       ;|
        RR      A                       ;|
        RR      A                       ;| data1 hilangkan bit0,bit1 & bit2
        mov     Bufr,A                  ;| data1 geser kanan 3x
        mov     A,Ch04                  ;| simpan sementara
        RR      A                       ;| and dg 1111 0000
        RR      A                       ;| or dg bufr
        RR      A                       ;|
        RR      A                       ;|
        anl     A,#0F0h                 ;|
        orl     A,Bufr                  ;|
        mov     Ch03,A                  ;/
        ret

                  0   1   2   3   4   5   6   7   8   9
ookup:  DB      000,015,015,015,016,016,016,017,017,017         ; 000-009
        DB      017,018,018,018,019,019,019,020,020,020         ; 010-019
        DB      021,021,021,022,022,022,023,023,023,024         ; 020-029
        DB      024,024,025,025,025,025,026,026,026,027         ; 030-039
        DB      027,027,028,028,028,029,029,029,030,030         ; 040-049
        DB      030,031,031,031,032,032,032,033,033,033         ; 050-059
        DB      033,034,034,034,035,035,035,036,036,036         ; 060-069
        DB      037,037,037,038,038,038,039,039,039,040         ; 070-079
        DB      040,040,041,041,041,041,042,042,042,043         ; 080-089
        DB      043,043,044,044,044,045,045,045,046,046         ; 090-099
        DB      046,047,047,047,048,048,048,049,049,049         ; 100-109
        DB      049,050,050,050,051,051,051,052,052,052         ; 110-119
        DB      053,053,053,054,054,054,055,055,055,056         ; 120-129
        DB      056,056,057,057,057,057,058,058,058,059         ; 130-139
        DB      059,059,060,060,060,061,061,061,062,062         ; 140-149
        DB      062,063,063,063,064,064,064,065,065,065         ; 150-159
        DB      065,066,066,066,067,067,067,068,068,068         ; 160-169
```

```
DB      069,069,069,070,070,070,071,071,071,072         ; 170-179
DB      072,072,073,073,073,073,074,074,074,075         ; 180-189
DB      075,075,076,076,076,077,077,077,078,078         ; 190-199
DB      078,079,079,079,080,080,080,081,081,081         ; 200-209
DB      081,082,082,082,083,083,083,084,084,084         ; 210-219
DB      085,085,085,086,086,086,087,087,087,088         ; 220-229
DB      088,088,089,089,089,089,090,090,090,091         ; 230-239
DB      091,091,092,092,092,093,093,093,093,094         ; 240-249
DB      094,094,094,095,095,095                         ; 250-255

end
```

Program Level Ketinggian Air


```
org     00h
ljmp    init

org     23h
clr     ES
jnb     RI,$
clr     RI
mov     R7,SBUF
setb    ES
reti

Mttr    Bit P2.2
Mtna    Bit P2.3
Rest    Bit P2.6
Enbl    Bit P2.7

ISDA    Bit P3.6                ; I2C data
ISCL    Bit P3.7                ; I2C clock

Sttc    Bit 20h.0               ; status command

Ch00    Equ 30h                 ;\
Ch01    Equ 31h                 ; |
Ch02    Equ 32h                 ; |
Ch03    Equ 33h                 ; |
Ch04    Equ 34h                 ; |
Ch05    Equ 35h                 ; |
Ch06    Equ 36h                 ; |
Ch07    Equ 37h                 ; | nama memory (character)
Ch08    Equ 38h                 ; | untuk baca no hp & sms
Ch09    Equ 39h                 ; |
Ch0A    Equ 3Ah                 ; |
Ch0B    Equ 3Bh                 ; |
Ch0C    Equ 3Ch                 ; |
Ch0D    Equ 3Dh                 ; |
Ch0E    Equ 3Eh                 ; |
Ch0F    Equ 3Fh                 ;/

Nhp0    Equ 40h                 ;\
Nhp1    Equ 41h                 ; |
Nhp2    Equ 42h                 ; |
Nhp3    Equ 43h                 ; |
Nhp4    Equ 44h                 ; | nama memory (character)
Nhp5    Equ 45h                 ; | untuk no HP
Nhp6    Equ 46h                 ; |
Nhp7    Equ 47h                 ; |
Nhp8    Equ 48h                 ; |
Nhp9    Equ 49h                 ; |
NhpA    Equ 4Ah                 ; |
NhpB    Equ 4Bh                 ;/
Tmo0    Equ 4Ch                 ; data time out 0
Tmo1    Equ 4Dh                 ; data time out 1
Cnsm    Equ 4Eh                 ; counter sms
Cndt    Equ 4Fh                 ; counter data sms

ChrL    Equ 50h
ChrH    Equ 51h
Dadc    Equ 52h
Dpnt    Equ 53h
Dpos    Equ 54h

Anhp    Equ 60h
Admm    Equ 61h
Dtmm    Equ 62h
Cntr    Equ 63h
Dpwm    Equ 64h
Dly0    Equ 65h
Dly1    Equ 66h
```

```
        Dly2    Equ 67h
        Dly3    Equ 68h

nit:    lcall   lcd_in
        mov     DPTR,#tpinls
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tphnph
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        mov     Anhp,#010h              ; baca memory no hp
        lcall   rdmnhp
        mov     Dly3,#10
        lcall   delay3

        lcall   srl_in                  ; serial 19200,8,N,1
        lcall   noecho                  ; no respon character
        lcall   dsminc                  ; display incoming sms
        lcall   rstcmd                  ; reset command

ulai:   mov     DPTR,#tpnama
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tpnims
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   delay2
        mov     DPTR,#tpjurs
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     DPTR,#tpuniv
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   delay2

ksms0:  cjne    R7,#0FFh,cksms1         ;\
        ljmp    cksms2                  ; |
ksms1:  mov     Dly3,#5                 ; |
        lcall   delay3                  ; | cek ada request
        lcall   bc_sms                  ; |
        lcall   rstcmd                  ;/
ksms2:  sjmp    mulai

sinhp:  lcall   lcdclr
        mov     DPTR,#tpnohp
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
snhp0:  mov     DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        mov     A,Nhp0
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp1
        lcall   cknhpn
```

```
        lcall   wr_chr
        mov     A,Nhp2
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp3
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp4
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp5
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp6
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp7
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp8
        lcall   cknhpn
        lcall   wr_chr
        mov     A,Nhp9
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpA
        lcall   cknhpn
        lcall   wr_chr
        mov     A,NhpB
        lcall   cknhpn
        lcall   wr_chr
        mov     P0,#0D0h
        lcall   w_ins
        lcall   tg_lps
snhp1:  lcall   scnkpd
        cjne    R0,#11,isnhp2
        mov     SP,#07h
        ljmp    mulai
snhp2:  cjne    R0,#12,isnhp3
        ljmp    isnhp4
snhp3:  cjne    R0,#15,isnhp1

snhp4:  mov     DPTR,#kosong
        lcall   line2
        mov     ChrL,#16
        lcall   tulis
        lcall   rsnohp
        lcall   tg_lps
        mov     DPTR,#angka
        mov     P0,#0C1h
        lcall   w_ins
        mov     P0,#'+'
        lcall   w_chr
        mov     P0,#'6'
        lcall   w_chr
        mov     P0,#'2'
        lcall   w_chr
        lcall   tginhp
        mov     Nhp0,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp1,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
```

```
        mov     Nhp2,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp3,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp4,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp5,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp6,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp7,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp8,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     Nhp9,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpA,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
        lcall   tginhp
        mov     NhpB,R0
        mov     A,R0
        lcall   wr_chr
        lcall   tg_lps
snhp5:  lcall   wrmnhp
        ljmp    isnhp0

ginhp:  lcall   scnkpd
        lcall   delay0
inhp0:  cjne    R0,#16,tinhp1
        ljmp    tginhp
inhp1:  cjne    R0,#15,tinhp2
        ljmp    tginhp
inhp2:  cjne    R0,#14,tinhp3
        ljmp    tginhp
inhp3:  cjne    R0,#13,tinhp4
        ljmp    tginhp
inhp4:  cjne    R0,#12,tinhp5
        ljmp    isnhp5
inhp5:  cjne    R0,#11,tinhp6
        mov     SP,#07h
        ljmp    mulai
inhp6:  cjne    R0,#10,tinhp7
```

```
        ljmp    tginhp
inhp7:  ret

snohp:  mov     Nhp0,#00Fh
        mov     Nhp1,#00Fh
        mov     Nhp2,#00Fh
        mov     Nhp3,#00Fh
        mov     Nhp4,#00Fh
        mov     Nhp5,#00Fh
        mov     Nhp6,#00Fh
        mov     Nhp7,#00Fh
        mov     Nhp8,#00Fh
        mov     Nhp9,#00Fh
        mov     NhpA,#00Fh
        mov     NhpB,#00Fh
        ret

knhpn:  cjne    A,#00Fh,cnhpn           ; cek angka no hp
        mov     A,#16                   ; if F then chr ' '
nhpn:   ret

albrs:  lcall   lcdclr
        mov     DPTR,#tpadkt
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        lcall   tg_lps
albr0:  lcall   bc_sns
        mov     DPTR,#angka
        mov     P0,#0C2h
        lcall   w_ins
        mov     A,Dadc
        lcall   nilai

ngpos:  mov     A,Dadc
        mov     B,#56
        div     AB
        jnz     rgpos0
        mov     Dpnt,#0
        ljmp    rgpos3
gpos0:  mov     A,Dadc
        mov     B,#100
        div     AB
        jnz     rgpos1
        mov     Dpnt,#1
        ljmp    rgpos3
gpos1:  mov     A,Dadc
        mov     B,#170
        div     AB
        jnz     rgpos2
        mov     Dpnt,#2
        ljmp    rgpos3
gpos2:  mov     A,Dadc
        mov     B,#210
        div     AB
        jnz     rgpos3
        mov     Dpnt,#3
gpos3:  mov     P0,#0CBh
        lcall   w_ins
        mov     A,Dpnt
        lcall   nilai
        mov     P0,#0D0h
        lcall   w_ins

        mov     Dly3,#1
        lcall   delay3
albr1:  lcall   scnkpd
        cjne    R0,#10,kalbr2
```

```
        setb    Mtna
        setb    Mttr
albr2:  cjne    R0,#11,kalbr3
        mov     SP,#07h
        ljmp    mulai
albr3:  cjne    R0,#15,kalbr4
        clr     Mtna
        setb    Mttr
        ljmp    kalbr0
albr4:  cjne    R0,#16,kalbr0
        setb    Mtna
        clr     Mttr
        ljmp    kalbr0

rl_in:  mov     Dly3,#1                 ;\
        lcall   delay3                  ; |
        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    EA                      ; |
        setb    ES                      ; |
        ret                             ;/

r_ins:  clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim instruksi ke hp
        djnz    ChrH,kr_ins             ; | sebagai awal pengiriman data
        mov     A,#0Dh                  ; |
        lcall   kr_srl                  ; |
        ret                             ;/

r_dta:  clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; |
        inc     DPTR                    ; | kirim data olahan (PDU)
        djnz    ChrH,kr_dta             ; | ke HP
        mov     A,#26                   ; |
        lcall   kr_srl                  ; |
        ret                             ;/

r_hlf:  clr     A                       ;\
        movc    A,@A+DPTR               ; |
        lcall   kr_srl                  ; | kirim setengah data olahan (PDU)
        inc     DPTR                    ; | yang sudah pasti ke HP
        djnz    ChrH,kr_hlf             ; |
        ret                             ;/

r_srl:  clr     ES                      ;\
        mov     SBUF,A                  ; |
        jnb     TI,$                    ; |
        clr     TI                      ; | kirim serial ke hp
        setb    ES                      ; |
        djnz    Dly0,$                  ; |
        ret                             ;/

)echo:  mov     Dly3,#1                 ;\
        lcall   delay3                  ; |
        mov     DPTR,#smseco            ; | kirim
        mov     ChrH,#4                 ; | instruksi
        lcall   kr_ins                  ; | no-echo ke hp
        lcall   bc_fbk                  ; |
        ret                             ;/

smminc: mov     Dly3,#1                 ;\
```

```
        lcall   delay3                  ;  |
        mov     DPTR,#smsfrm            ;  | kirim instruksi
        mov     ChrH,#17                ;  | display-incoming-sms
        lcall   kr_ins                  ;  | ke hp
        lcall   bc_fbk                  ;  |
        ret                             ;/

pssms:  mov     DPTR,#smshps            ;\
        mov     ChrH,#9                 ;  |
        lcall   kr_ins                  ;  |
        lcall   bc_fbk                  ;  | hapus sms pada memory 1
        mov     Dly3,#1                 ;  |
        lcall   delay3                  ;  |
        ret                             ;/

c_fbk:  lcall   rstcmd                  ;\
c_fb0:  cjne    R7,#0FFh,bc_fb1         ;  | tunggu feedback
        ljmp    bc_fb0                  ;  | data dari hp
c_fb1:  ret                             ;/

c_sr1:  lcall   rstcmd                  ;\
        mov     Tmo0,#15                ;  |
        mov     Tmo1,#0                 ;  | baca serial
c_sr0:  cjne    R7,#0FFh,bc_sr1         ;  | tunggu data bukan FF
        djnz    Tmo1,bc_sr0             ;  | sebelum time out
        djnz    Tmo0,bc_sr0             ;  |
c_sr1:  ret                             ;/

stcmd:  mov     R7,#0FFh
        ret

rnohp:  mov     A,Nhp1
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp0
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp3
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp2
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp5
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp4
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp7
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp6
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp9
        mov     B,#30h
        add     A,B
        lcall   kr_sr1
        mov     A,Nhp8
```

```
        mov     B,#30h
        add     A,B
        lcall   kr_srl
        mov     A,NhpB
        lcall   ck_ksg
        mov     A,NhpA
        lcall   ck_ksg
        ret

k_ksg:  cjne    A,#00Fh,ckksg0
        mov     A,#'F'
        ljmp    ckksg1
kksg0:  mov     B,#30h
        add     A,B
kksg1:  lcall   kr_srl
        ret

krsms:  lcall   lcdclr                  ; tulis kirim sms
        mov     DPTR,#tpkrsm
        lcall   line1
        mov     ChrL,#16
        lcall   tulis
        mov     Dly3,#1
        lcall   delay3
        ret

:_sms:  ;lcall   bcnohp                 ;\
        ;jnb     Sttc,bcsms0            ; | baca no HP
        lcall   bccmnd                  ; | cek status
        ;lcall   hpssms                 ; | 0=hapus sms, 1=baca command
        ;ljmp    bcsms1                 ; | hapus sms
:sms0:  mov     A,Ch00                  ; |
        cjne    A,#10,bcsmsx           ; | if tdk ada no -> tdk hps
        ljmp    bcsmsB                  ; |
:smsx:  lcall   hpssms                  ; | cek command
        ;ljmp    bcsmsB                 ;/

:sms1:  mov     DPTR,#comnd0            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsms2            ; | cek command
        mov     Cntr,#0                 ; |
        mov     Dpos,#56                ; | pintu posisi 0
        lcall   ps_pnt                  ; |
        ljmp    bcsmsB                  ;/

:sms2:  mov     DPTR,#comnd1            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsms3            ; | cek command
        mov     Cntr,#0                 ; |
        mov     Dpos,#100               ; | pintu posisi 1
        lcall   ps_pnt                  ; |
        ljmp    bcsmsB                  ;/

:sms3:  mov     DPTR,#comnd2            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsms4            ; | cek command
        mov     Cntr,#0                 ; |
        mov     Dpos,#170               ; | pintu posisi 2
        lcall   ps_pnt                  ; |
        ljmp    bcsmsB                  ;/

:sms4:  mov     DPTR,#comnd3            ;\
        lcall   ckcmnd                  ; | ambil data command
        jnb     Sttc,bcsmsB            ; | cek command
        mov     Cntr,#0                 ; |
        mov     Dpos,#210               ; | pintu posisi 3
        lcall   ps_pnt                  ; |
        ljmp    bcsmsB                  ;/
```

```
csmsB: ret

cnohp: mov     DPTR,#smsred            ;\
       mov     ChrH,#9                 ; |
       lcall   kr_ins                  ; |
       mov     Cntr,#30                ; | baca command
       lcall   bc_fbk                  ; |
cnhp0: lcall   bc_srl                  ; | baca serial
       cjne    R7,#0Dh,bcnhp1          ; | sampai character '_|'
       ljmp    bcnhp2                  ; |
cnhp1: djnz    Cntr,bcnhp0             ; |
       ljmp    cknhp2                  ;/
cnhp2: lcall   bc_srl                  ;\
       lcall   bc_srl                  ; |
       lcall   bc_srl                  ; |
       clr     C                       ; |
       mov     A,R7                    ; |
       mov     B,#30h                  ; | lewati service center
       subb    A,B                     ; |
       mov     B,#2                    ; | ambil character -> decimal
       mul     AB                      ; | lewati character sebanyak
       mov     B,#8                    ; | hasil kali 2 tambah 8
       add     A,B                     ; |
       mov     Cntr,A                  ; |
cnhp3: lcall   bc_srl                  ; |
       djnz    Cntr,bcnhp3             ;/
       lcall   bc_dta                  ; baca data
        lcall   line1                  ; optional
        lcall   tlsdta                 ; optional
        lcall   delay2                 ; optional

knohp: mov     DPTR,#angka             ;\
       mov     Cntr,#10                ; | jumlah no hp = 10
       mov     Cnsm,#0                 ; | reset counter sms
       mov     Cndt,#0                 ; | reset counter no hp
knhp0: 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 selanjutnya
       djnz    Cntr,cknhp0             ; | cek sampai no hp habis
       setb    Sttc                    ; | set status = 1
       ljmp    cknhp2                  ; | jika tidak
knhp1: clr     Sttc                    ; | set status = 0
knhp2: ret                             ;/

tanhp: mov     A,Cndt
       cjne    A,#00,dnhp00
       mov     A,Nhp1
       ljmp    dnhp11
nhp00: cjne    A,#01,dnhp01
       mov     A,Nhp0
       ljmp    dnhp11
nhp01: cjne    A,#02,dnhp02
       mov     A,Nhp3
       ljmp    dnhp11
nhp02: cjne    A,#03,dnhp03
       mov     A,Nhp2
       ljmp    dnhp11
nhp03: cjne    A,#04,dnhp04
       mov     A,Nhp5
       ljmp    dnhp11
nhp04: cjne    A,#05,dnhp05
       mov     A,Nhp4
       ljmp    dnhp11
```

```
nhp05: cjne    A,#06,dnhp06
       mov     A,Nhp7
       ljmp    dnhp11
nhp06: cjne    A,#07,dnhp07
       mov     A,Nhp6
       ljmp    dnhp11
nhp07: cjne    A,#08,dnhp08
       mov     A,Nhp9
       ljmp    dnhp11
nhp08: cjne    A,#09,dnhp09
       mov     A,Nhp8
       ljmp    dnhp11
nhp09: cjne    A,#10,dnhp10
       mov     A,NhpB
       ljmp    dnhp11
nhp10: cjne    A,#11,dnhp12
       mov     A,NhpA
nhp11: cjne    A,#00Fh,dnhp12
       mov     A,#15
nhp12: movc    A,@A+DPTR
       ret

ccmnd: mov     DPTR,#smsred            ;\
       mov     ChrH,#9                 ; |
       lcall   kr_ins                  ; |
       mov     Cntr,#50                ; | baca command
       lcall   bc_fbk                  ; |
ccmd0: lcall   bc_srl                  ; | baca serial
       cjne    R7,#0Dh,bccmd1          ; | if character '_|'
       ljmp    bccmd2                  ; |
ccmd1: djnz    Cntr,bccmd0             ; |
       ljmp    bccmdB                  ;/
ccmd2: lcall   bc_srl                  ;\
       lcall   bc_srl                  ; |
       lcall   bc_srl                  ; |
       clr     C                       ; |
       mov     A,R7                    ; |
       mov     B,#30h                  ; | lewati service center
       subb    A,B                     ; |
       mov     B,#2                    ; | ambil character -> decimal
       mul     AB                      ; | lewati character sebanyak
       mov     B,#3                    ; | hasil kali 2 tambah 3
       add     A,B                     ; |
       mov     Cntr,A                  ; |
ccmd3: lcall   bc_srl                  ; |
       djnz    Cntr,bccmd3             ;/
       lcall   bc_srl                  ;\
ccmd4: cjne    R7,#'A',bccmd5          ; |
       mov     Cntr,#30                ; |
ccmd5: cjne    R7,#'B',bccmd6          ; | cek jumlah no HP
       mov     Cntr,#32                ; |
ccmd6: cjne    R7,#'C',bccmd7          ; | if A -> lewati 30 character
       mov     Cntr,#32                ; | if B -> lewati 32 character
ccmd7: cjne    R7,#'D',bccmd8          ; | if C -> lewati 32 character
       mov     Cntr,#34                ; | if D -> lewati 34 character
ccmd8: cjne    R7,#'E',bccmd9          ; | if E -> lewati 34 character
       mov     Cntr,#34                ; |
ccmd9: lcall   bc_srl                  ; |
       djnz    Cntr,bccmd9             ;/
ccmdA: lcall   bc_srl                  ;
       lcall   bc_srl                  ;
       lcall   bc_dta                  ; baca data
        lcall   line2                  ; optional
        lcall   tlsdta                 ; optional
        lcall   delay2                 ; optional
ccmdB: ret

ccmnd: mov     Cntr,#10                ;\
```

```
        mov     Cnsm,#0                 ;\ | jumlah command
        mov     Cndt,#0                 ; | reset counter sms
kcnd0:  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
elanjutnya
        djnz    Cntr,ckcnd0             ; | cek sampai command habis
        setb    Sttc                    ; | set status = 1
        ljmp    ckcnd2                  ; | jika tidak
kcnd1:  clr     Sttc                    ; | set status = 0
kcnd2:  ret                             ;/

tacmd:  mov     A,Cndt                  ;\
        movc    A,@A+DPTR               ; | ambil data pointer yang ke cndt
        ret                             ;/

tasms:  clr     C                       ;\
        mov     A,Cnsm                  ; |
tsm00:  cjne    A,#00,dtsm01            ; |
        mov     B,Ch00                  ; |
tsm01:  cjne    A,#01,dtsm02            ; |
        mov     B,Ch01                  ; |
tsm02:  cjne    A,#02,dtsm03            ; |
        mov     B,Ch02                  ; |
tsm03:  cjne    A,#03,dtsm04            ; |
        mov     B,Ch03                  ; |
tsm04:  cjne    A,#04,dtsm05            ; |
        mov     B,Ch04                  ; |
tsm05:  cjne    A,#05,dtsm06            ; |
        mov     B,Ch05                  ; |
tsm06:  cjne    A,#06,dtsm07            ; |
        mov     B,Ch06                  ; |
tsm07:  cjne    A,#07,dtsm08            ; |
        mov     B,Ch07                  ; | samakan data counter
tsm08:  cjne    A,#08,dtsm09            ; | dengan character
        mov     B,Ch08                  ; |
tsm09:  cjne    A,#09,dtsm0A            ; |
        mov     B,Ch09                  ; |
tsm0A:  cjne    A,#10,dtsm0B            ; |
        mov     B,Ch0A                  ; |
tsm0B:  cjne    A,#11,dtsm0C            ; |
        mov     B,Ch0B                  ; |
tsm0C:  cjne    A,#12,dtsm0D            ; |
        mov     B,Ch0C                  ; |
tsm0D:  cjne    A,#13,dtsm0E            ; |
        mov     B,Ch0D                  ; |
tsm0E:  cjne    A,#14,dtsm0F            ; |
        mov     B,Ch0E                  ; |
tsm0F:  cjne    A,#15,dtsmsx            ; |
        mov     B,Ch0F                  ; |
tsmsx:  ret                             ;/

t_dta:  lcall   bc_srl                  ;\
        mov     Ch00,R7                 ; |
        lcall   bc_srl                  ; |
        mov     Ch01,R7                 ; |
        lcall   bc_srl                  ; |
        mov     Ch02,R7                 ; |
        lcall   bc_srl                  ; |
        mov     Ch03,R7                 ; |
        lcall   bc_srl                  ; |
        mov     Ch04,R7                 ; |
        lcall   bc_srl                  ; |
        mov     Ch05,R7                 ; |
```

```
        lcall   bc_srl                  ;  |
        mov     Ch06,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch07,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch08,R7                 ;  |  baca serial & simpan
        lcall   bc_srl                  ;  |  ke memory character
        mov     Ch09,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0A,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0B,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0C,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0D,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0E,R7                 ;  |
        lcall   bc_srl                  ;  |
        mov     Ch0F,R7                 ;  |
        mov     Dly3,#5                 ;  |
        lcall   delay3                  ;  |
        ret                             ;/

lsdta:  mov     P0,Ch00                 ;\
        lcall   w_chr                   ;  |
        mov     P0,Ch01                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch02                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch03                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch04                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch05                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch06                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch07                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch08                 ;  |  tulis character ke lcd
        lcall   w_chr                   ;  |
        mov     P0,Ch09                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0A                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0B                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0C                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0D                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0E                 ;  |
        lcall   w_chr                   ;  |
        mov     P0,Ch0F                 ;  |
        lcall   w_chr                   ;  |
        ret                             ;/

r_sms:  mov     DPTR,#smskrm            ; ambil data PDU kirim 47 oktet
        mov     ChrH,#10                ; ATCommand 10 character
        lcall   kr_ins                  ; kirim data diakhiri enter
        lcall   bc_fbk                  ; tunggu feedback dari HP
        mov     Dly3,#1                 ;
        lcall   delay3                  ; wait
        mov     DPTR,#header            ;\
        mov     ChrH,#12                ;  | kirim header
        lcall   kr_hlf                  ;/
        lcall   krnohp                  ; kirim no HP
```

```
        mov     DPTR,#sysonn            ; ambil data PDU
        mov     ChrH,#20                ; ATCommand 20 character
        lcall   kr_dta                  ; kirim data PDU yg sudah pasti
        lcall   bc_fbk                  ; tunggu feedback dari HP
        mov     Dly3,#2                 ;
        lcall   delay3                  ; wait
        ret

c_sns:  mov     A,#90h                  ; address PCF8591 write address
        lcall   adrtx                   ; kirim
        mov     A,#00h                  ; adc ch-0
        lcall   dtatx                   ; kirim
        mov     A,#91h                  ; address PCF8591 read address
        lcall   adrtx                   ; kirim
        lcall   dtarx                   ; baca data
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; baca data
        lcall   i2cstp                  ; i2c stop
        mov     Dadc,A                  ; simpan data sensor
        ret

rmnhp:  mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp1               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp0               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp3               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp2               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp5               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp4               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp7               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp6               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp9               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,Nhp8               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,NhpB               ; data memory
        lcall   wr_mem
        inc     Anhp
        mov     Admm,Anhp               ; address memory
        mov     Dtmm,NhpA               ; data memory
        lcall   wr_mem
```

```
        ret

dmnhp:  mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,Anhp                  ; address memory
        lcall   dtatx
        mov     A,#0A1h                 ; AT24C16 read address
        lcall   adrtx
        lcall   dtarx                   ; terima data
        mov     Nhp1,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp0,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp3,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp2,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp5,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp4,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp7,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp6,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp9,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     Nhp8,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     NhpB,A
        lcall   givack                  ; beri ack
        lcall   dtarx                   ; terima data
        mov     NhpA,A
        lcall   i2cstp                  ; i2c stop
        ret

r_mem:  mov     A,#0A0h                 ; AT24C16 write address
        lcall   adrtx
        mov     A,Admm                  ; address hardware memory
        lcall   dtatx
        mov     A,Dtmm                  ; data memory
        lcall   dtatx
        lcall   i2cstp                  ; i2c stop
        lcall   wt_wr
        ret

t_wr:   mov     Dly0,#000
        mov     Dly1,#050
twr:    lcall   delay0
        djnz    Dly1,wtwr
        ret

irtx:   lcall   i2cstr                  ; kirim address
        lcall   putbit                  ; kirim data
        ret                             ; back

tatx:   lcall   putbit                  ; kirim data
        ret                             ; back
```

```
tarx:   lcall   getbit              ; terima data
        ret                         ; back

utbit:  mov     R6,#8               ;\
utbt:   RLC     A                   ; |
        mov     ISDA,C              ; |
        setb    ISCL                ; |
        clr     ISCL                ; | kirim bit
        djnz    R6,putbt            ; |
        setb    ISDA                ; |
        lcall   getack              ; |
        ret                         ;/

etbit:  mov     R6,#8               ;\
etbt:   setb    ISCL                ; |
        mov     C,ISDA              ; |
        RLC     A                   ; | terima bit
        clr     ISCL                ; |
        djnz    R6,getbt            ; |
        setb    ISDA                ; |
        ret                         ;/

etack:  setb    ISDA                ;\
        setb    ISCL                ; |
ckbit:  mov     C,ISDA              ; | tunggu ack
        jc      ackbit              ; | D=1, C=1D=0, C=0
        clr     ISCL                ; |
        ret                         ;/

ivack:  clr     ISDA                ;\
        setb    ISCL                ; |
        clr     ISCL                ; | kirim ack -> D=0, C=1, C=0, D=1
        setb    ISDA                ; |
        ret                         ;/

?cstr:  setb    ISCL                ;\
        setb    ISDA                ; |
        clr     ISDA                ; | i2c start -> C=1, D=1, D=0, C=0
        clr     ISCL                ; |
        ret                         ;/

?cstp:  clr     ISDA                ;\
        setb    ISCL                ; |
        setb    ISDA                ; | i2c stop -> D=0, C=1, D=1, C=0
        clr     ISCL                ; |
        ret                         ;/

ilai:   mov     B,#100
        div     AB
        lcall   wr_chr
        mov     A,B
        mov     B,#10
        div     AB
        lcall   wr_chr
        mov     A,B
        lcall   wr_chr
        ret

s_pnt:  lcall   bc_sns
        mov     A,Dadc
        mov     B,Dpos
        div     AB
        jnz     pspnt0
        mov     A,Dpos
        mov     B,Dadc
        clr     C
        subb    A,B
```

```
        mov     Dpwm,A
        clr     Mtna
        setb    Mttr
spwm0:  lcall   delay0
        djnz    Dpwm,pspwm0
        setb    Mtna
        setb    Mttr
        mov     Dpwm,#5
spwm1:  lcall   delay0
        djnz    Dpwm,pspwm1
        ljmp    ps_pnt

spnt0:  mov     A,Dadc
        mov     B,Dpos
        clr     C
        subb    A,B
        jz      pspnt1
        mov     A,Dadc
        mov     B,Dpos
        clr     C
        subb    A,B
        mov     Dpwm,A
        setb    Mtna
        clr     Mttr
spwm2:  lcall   delay0
        djnz    Dpwm,pspwm2
        setb    Mtna
        setb    Mttr
        mov     Dpwm,#5
spwm3:  lcall   delay0
        djnz    Dpwm,pspwm3
        ljmp    ps_pnt

spnt1:  djnz    Cntr,ps_pnt
        setb    Mtna
        setb    Mttr
spnt2:  ret

ine1:   mov     P0,#080h
        lcall   w_ins
        ret

ine2:   mov     P0,#0C0h
        lcall   w_ins
        ret

ulis:   clr     A
        lcall   wr_chr
        inc     DPTR
        djnz    ChrL,tulis
        ret

r_chr:  movc    A,@A+DPTR
        mov     P0,A
        lcall   w_chr
        ret

_ins:   clr     Enbl
        clr     Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret

_chr:   clr     Enbl
        setb    Rest
        setb    Enbl
        clr     Enbl
```

```
        lcall   delay0
        ret

cd_in:  mov     Dly3,#1
        lcall   delay3
        mov     P0,#01h                 ; Display Clear
        lcall   w_ins
        mov     P0,#38h                 ; Function Set
        lcall   w_ins
        mov     P0,#0Dh                 ; Display On, Cursor, Blink
        lcall   w_ins
        mov     P0,#06h                 ; Entry Mode
        lcall   w_ins
        mov     P0,#02h                 ; Cursor Home
        lcall   w_ins
        ret

cdclr:  mov     P0,#01h                 ; Display Clear
        lcall   w_ins
        lcall   delay0
        lcall   delay0
        lcall   delay0
        ret

cnkpd:  mov     R0,#10
        lcall   delay0
>11:    mov     P1,#11111110b
        mov     A,P1
1b1:    cjne    A,#11101110b,c1b2
        mov     R0,#1
1b2:    cjne    A,#11011110b,c1b3
        mov     R0,#2
1b3:    cjne    A,#10111110b,c1b4
        mov     R0,#3
1b4:    cjne    A,#01111110b,col2
        mov     R0,#13

>12:    mov     P1,#11111101b
        mov     A,P1
2b1:    cjne    A,#11101101b,c2b2
        mov     R0,#4
2b2:    cjne    A,#11011101b,c2b3
        mov     R0,#5
2b3:    cjne    A,#10111101b,c2b4
        mov     R0,#6
2b4:    cjne    A,#01111101b,col3
        mov     R0,#14

>13:    mov     P1,#11111011b
        mov     A,P1
3b1:    cjne    A,#11101011b,c3b2
        mov     R0,#7
3b2:    cjne    A,#11011011b,c3b3
        mov     R0,#8
3b3:    cjne    A,#10111011b,c3b4
        mov     R0,#9
3b4:    cjne    A,#01111011b,col4
        mov     R0,#15

>14:    mov     P1,#11110111b
        mov     A,P1
4b1:    cjne    A,#11100111b,c4b2
        mov     R0,#11
4b2:    cjne    A,#11010111b,c4b3
        mov     R0,#0
4b3:    cjne    A,#10110111b,c4b4
        mov     R0,#12
4b4:    cjne    A,#01110111b,back
```

```
        mov     R0,#16
ack:    ret

g_tkn: lcall   scnkpd
       lcall   delay0
g_tk0: cjne    R0,#16,tg_tk1
       ljmp    tg_tkn
g_tk1: cjne    R0,#15,tg_tk2
       ljmp    tg_tkn
g_tk2: cjne    R0,#14,tg_tk3
       ljmp    tg_tkn
g_tk3: cjne    R0,#13,tg_tk4
       ljmp    tg_tkn
g_tk4: cjne    R0,#12,tg_tk5
       ljmp    tg_tkn
g_tk5: cjne    R0,#11,tg_tk6
       ljmp    tg_tkn
g_tk6: cjne    R0,#10,tg_tk7
       ljmp    tg_tkn
g_tk7: ret

g_lps: lcall   scnkpd
       lcall   delay0
       cjne    R0,#10,tg_lps
       ret

elay0: djnz    Dly0,delay0
       ret

elay1: lcall   scnkpd
       cjne    R0,#13,dely10
        ljmp    isinhp
ely10: cjne    R0,#14,dely11
       ljmp    kalbrs
ely11: djnz    Dly1,delay1
       ret

elay2: mov     Dly2,#20
ely2:  lcall   delay1
       djnz    Dly2,dely2
       ret

elay3: lcall   delay0
       djnz    Dly1,delay3
       djnz    Dly3,delay3
       ret

nama: DB       '   Andri  Dwi   '
nims: DB       '  NIM: 0912902  '
jurs: DB       ' Teknik Elektro '
univ: DB       '   ITN Malang   '
inls: DB       '  Inisialisasi  '
hnph: DB       '   Hand Phone   '
nohp: DB       '    Nomor HP    '
adkt: DB       '  ADC      POS  '
krsm: DB       '   Kirim  SMS   '

nseco: DB      'ATE0'                    ; instruksi tanpa echo
nsfrm: DB      'AT+CNMI=1,1,0,0,1'       ; instruksi display incoming sms
nsred: DB      'AT+CMGR=1'               ; instruksi baca SMS
nshps: DB      'AT+CMGD=1'               ; instruksi hapus sms
nskrm: DB      'AT+CMGS=21'              ; instruksi kirim respon

mnd0: DB       'D0E7140403'              ; POS 0
mnd1: DB       'D0E7141403'              ; POS 1
mnd2: DB       'D0E7142403'              ; POS 2
mnd3: DB       'D0E7143403'              ; POS 3
```

```
eader: DB      '0001000D9126'
ysonn: DB      '000008D3EC14F4748240'   ; SYS ON
ngka:  DB      '0123456789ABCDEF'
osong: DB      '                '

       end
```

DATA SHEET

SIEMENS

Standard EEPROM ICs

SLx 24C164/P

16 Kbit (2048 × 8 bit)
Serial CMOS-EEPROM with
I^2C Synchronous 2-Wire Bus
and Page Protection Mode™

Data Sheet 1998-07-27

SLx 24C164/P Revision History:		Current Version: 1998-07-27
Previous Version:		06.97
Page (in previous Version)	Page (in current Version)	Subjects (major changes since last revision)
3	3	Text was changed to "Typical programming time 5 ms for up to 16 bytes".
5	5	WP = V_{CC} protects the ~~upper-half~~ entire memory.
11, 12	11, 12	The erase/write cycle is finished latest after ~~10~~ 8 ms.
15	15	Figure 11: second command byte is a **CSR** and not CSW.
21	21	The write or erase cycle is finished latest after ~~10~~ 4 ms.
19	24	"Capacitive load ..." were added.
25	25	Some timings were changed.
25	25	The line "erase/write cycle" was removed.
25	25	Chapter 8.4 "Erase and Write Characteristics" has been added.

²C Bus

'urchase of Siemens I²C components conveys the license under the Philips I²C patent to use the components in he I²C system provided the system conforms to the I²C specifications defined by Philips.

dition 1998-07-27

ublished by Siemens AG,
ereich Halbleiter, Marketing-
ommunikation, Balanstraße 73,
1541 München

ttention please!

s far as patents or other rights of third parties are concerned, liability is only assumed for components, not for applications, processes nd circuits implemented within components or assemblies.

he information describes the type of component and shall not be considered as assured characteristics.

erms of delivery and rights to change design reserved.

or questions on technology, delivery and prices please contact the Semiconductor Group Offices in Germany or the Siemens Companies nd Representatives worldwide (see address list).

ue to technical requirements components may contain dangerous substances. For information on the types in question please contact our nearest Siemens Office, Semiconductor Group.

iemens AG is an approved CECC manufacturer.

acking

lease use the recycling operators known to you. We can also help you – get in touch with your nearest sales office. By agreement we ill take packing material back, if it is sorted. You must bear the costs of transport.

or packing material that is returned to us unsorted or which we are not obliged to accept, we shall have to invoice you for any costs in- urred.

omponents used in life-support devices or systems must be expressly authorized for such purpose!

ritical components[1] of the Semiconductor Group of Siemens AG, may only be used in life-support devices or systems[2] with the express ritten approval of the Semiconductor Group of Siemens AG.

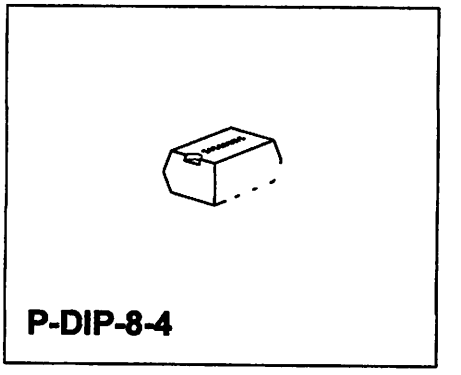
1 A critical component is a component used in a life-support device or system whose failure can reasonably be expected to cause the failure of that life-support device or system, or to affect its safety or effectiveness of that device or system.

2 Life support devices or systems are intended (a) to be implanted in the human body, or (b) to support and/or maintain and sustain human life. If they fail, it is reasonable to assume that the health of the user may be endangered.

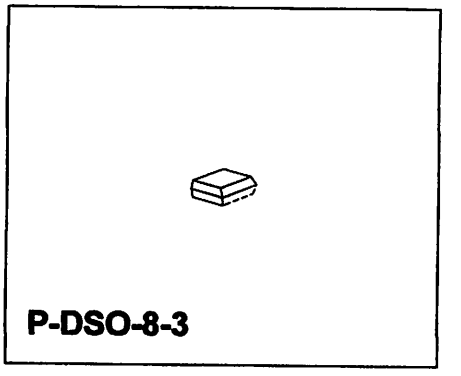
16 Kbit (2048 × 8 bit) Serial CMOS EEPROMs, I²C Synchronous 2-Wire Bus, Page Protection Mode™

SLx 24C164/P

Features



P-DIP-8-4



P-DSO-8-3

- **Data EEPROM internally organized as 2048 bytes and 128 pages × 16 bytes**
- **Page protection mode, flexible page-by-page hardware write protection**
  - Additional protection EEPROM of 128 bits, 1 bit per data page
  - Protection setting for each data page by writing its protection bit
  - Protection management without switching WP pin
- **Low power CMOS**
- **V_{CC} = 2.7 to 5.5 V operation**
- **Two wire serial interface bus, I²C-Bus compatible**
- **Three chip select pins to address 8 devices**
- **Filtered inputs for noise suppression with Schmitt trigger**
- **Clock frequency up to 400 kHz**
- **High programming flexibility**
  - Internal programming voltage
  - Self timed programming cycle including erase
  - Byte-write and page-write programming, between 1 and 16 bytes
  - Typical programming time 5 ms for up to 16 bytes
- ***High reliability***
  - Endurance 10^6 cycles[1)]
  - Data retention 40 years[1)]
  - ESD protection 4000 V on all pins
- **8 pin DIP/DSO packages**
- **Available for extended temperature ranges**
  - Industrial: – 40 °C to + 85 °C
  - Automotive: – 40°C to + 125 °C

Values are temperature dependent, for further information please refer to your Siemens Sales office.

Ordering Information

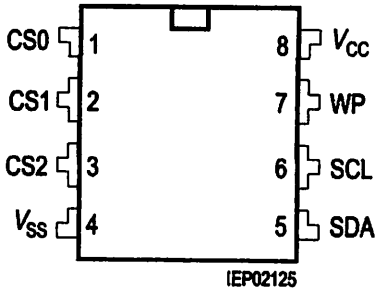
Type	Ordering Code	Package	Temperature	Voltage
SLA 24C164-D/P	Q67100-H3504	P-DIP-8-4	– 40 °C ... + 85 °C	4.5 V...5.5 V
SLA 24C164-S/P	Q67100-H3499	P-DSO-8-3	– 40 °C ... + 85 °C	4.5 V...5.5 V
SLA 24C164-D-3/P	Q67100-H3502	P-DIP-8-4	– 40 °C ... + 85 °C	2.7 V...5.5 V
SLA 24C164-S-3/P	Q67100-H3498	P-DSO-8-3	– 40 °C ... + 85 °C	2.7 V...5.5 V
SLE 24C164-D/P	Q67100-H3503	P-DIP-8-4	– 40°C ... + 125 °C	4.5 V...5.5 V
SLE 24C164-S/P	Q67100-H3497	P-DSO-8-3	– 40°C ... + 125 °C	4.5 V...5.5 V

Other types are available on request

- Temperature range (– 55 °C ... + 150 °C)
- Package (die, wafer delivery)

Pin Configuration

P-DIP-8-4



P-DSO-8-3

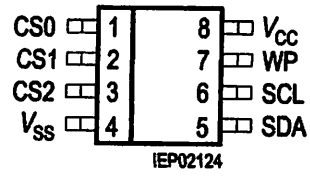


Figure 1
Pin Configuration (top view)

Pin Definitions and Functions

Table 1

Pin No.	Symbol	Function
1, 2, 3	CS0, CS1, CS2	Chip select inputs
4	V_{SS}	Ground
5	SDA	Serial bidirectional data bus
6	SCL	Serial clock input
7	WP	Write protection input
8	V_{CC}	Supply voltage

Pin Description

Serial Clock (SCL)

The SCL input is used to clock data into the device on the rising edge and to clock data out of the device on the falling edge.

Serial Data (SDA)

SDA is a bidirectional pin used to transfer addresses, data or control information into the device or to transfer data out of the device. The output is open drain, performing a wired AND function with any number of other open drain or open collector devices. The SDA bus requires a pull-up resistor to V_{CC}.

Chip Select (CS0, CS1, CS2)

The CS0, CS1 and CS2 pins are chip select inputs either hard wired or actively driven to V_{CC} or V_{SS}. These inputs allow the selection of one of eight possible devices sharing a common bus.

Write Protection (WP)

WP switched to V_{SS} allows normal read/write operations.

WP switched to V_{CC} protects the entire EEPROM against changes (hardware write protection).

Additionally write protection is managed by a protection bit associated to each page. (refer to **chapter 7** Page Protection Mode™)

2 Description

The SLx 24C164/P device is a serial electrically erasable and programmable read only memory (EEPROM), organized as 2048 × 8 bit. The data memory is divided into 128 pages. The 16 bytes of a page can be programmed simultaneously. Each page may be protected individually against changes by its associated protection bit.

The device conforms to the specification of the 2-wire serial I^2C-Bus. Three chip select pins allow the addressing of 8 devices on the I^2C-Bus. Low voltage design permits operation down to 2.7 V with low active and standby currents. All devices have a minimum endurance of 10^6 erase/write cycles.

The device operates at 5.0 V ± 10% with a maximum clock frequency of 400 kHz and at 2.7 ... 4.5 V with a maximum clock frequency of 100 kHz. The device is available as 5 V type (V_{CC} = 4.5 ... 5.5 V) with two temperature ranges for industrial and automotive applications and as 3 V type (V_{CC} = 2.7 ... 5.5 V) for industrial applications. The EEPROMs are mounted in eight-pin DIP and DSO packages or are also supplied as chips.

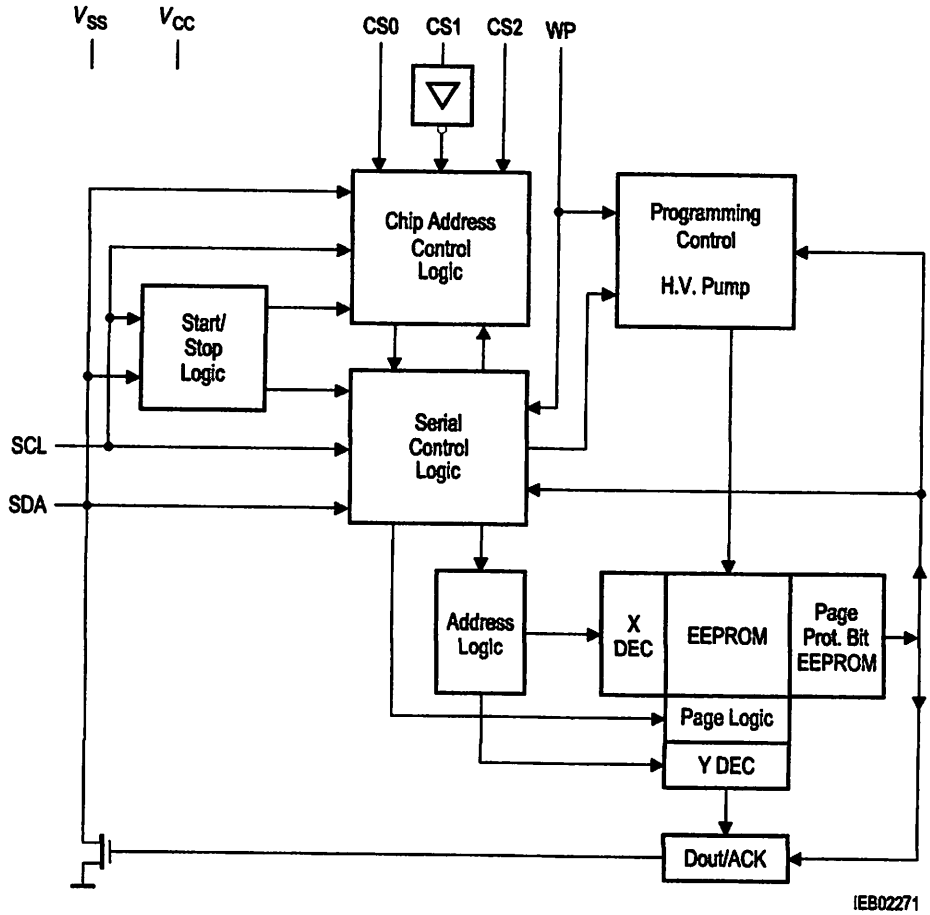


Figure 2
Block Diagram

I²C-Bus Characteristics

The SLx 24C164/P devices support a master/slave bidirectional bus oriented protocol in which the EEPROM always takes the role of a slave.

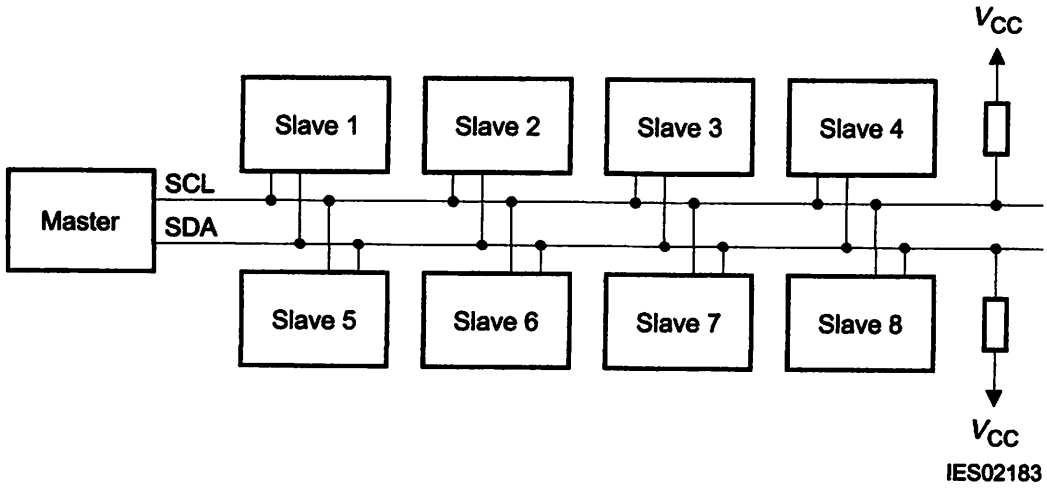


Figure 3
Bus Configuration

Master Device that initiates the transfer of data and provides the clock for both transmit and receive operations.

Slave Device addressed by the master, capable of receiving and transmitting data.

Transmitter The device with the SDA as output is defined as the transmitter. Due to the open drain characteristic of the SDA output the device applying a low level wins.

Receiver The device with the SDA as input is defined as the receiver.

The conventions for the serial clock line and the bidirectional data line are shown in Figure 4.

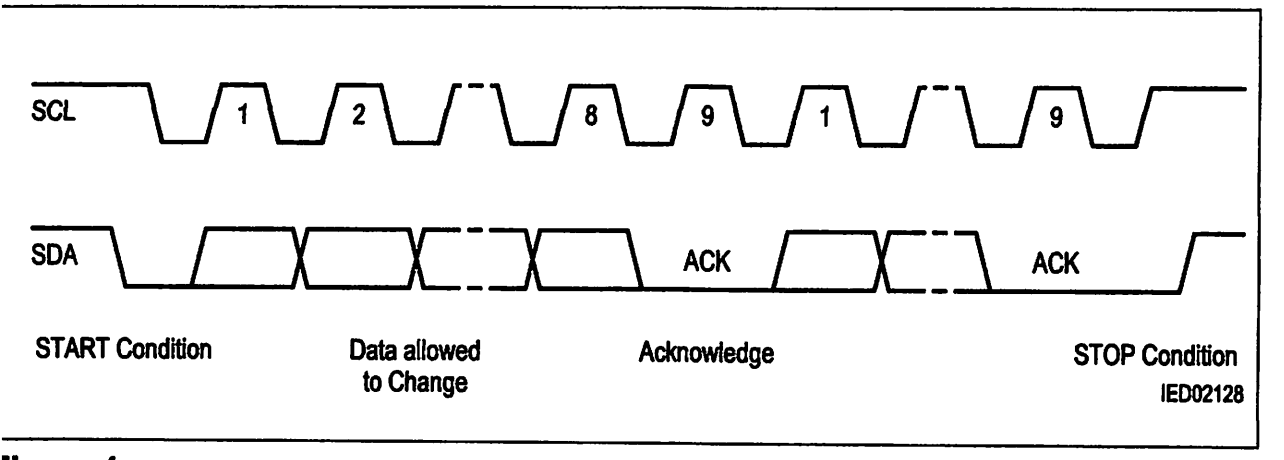


Figure 4
I²C-Bus Timing Conventions for START Condition, STOP Condition, Data Validation and Transfer of Acknowledge ACK

Standby Mode in which the bus is not busy (no serial transmission, no programming): both clock (SCL) and data line (SDA) are in high state. The device enters the standby mode after a STOP condition or after a programming cycle.

START Condition High to low transition of SDA when SCL is high, preceding all commands.

STOP Condition Low to high transition of SDA when SCL is high, terminating all communications. A STOP condition initiates an EEPROM programming cycle. A STOP condition after reading a data byte from the EEPROM initiates the Standby mode.

Acknowledge A successful reception of eight data bits is indicated by the receiver by pulling down the SDA line during the following clock cycle of SCL (ACK). The transmitter on the other hand has to release the SDA line after the transmission of eight data bits.
The EEPROM as the receiving device responds with an acknowledge, when addressed. The master, on the other side, acknowledges each data byte transmitted by the EEPROM and can at any time end a read operation by releasing the SDA line (no ACK) followed by a STOP condition.

Data Transfer Data must change only during low SCL state, data remains valid on the SDA bus during high SCL state. Nine clock pulses are required to transfer one data byte, the most significant bit (MSB) is transmitted first.

Device Addressing and EEPROM Addressing

After a START condition, the master always transmits a Command Byte CSW or CSR. After the acknowledge of the EEPROM a Control Byte follows, its content and the transmitter depend on the previous Command Byte. The description of the Command and Control Bytes is shown in **table 2.**

Command Byte

Selects one of the 8 addressable devices: the chip select bits c2, $\overline{c1}$ and c0 (bit positions b6 to b4) are compared to their corresponding hard wired input pins CS2, CS1 and CS0, respectively ($\overline{c1}$ is the complement of CS1 pin).

Selects operation: the least significant bit b0 is low for a write operation (Chip Select Write Command Byte CSW) or set high for a read operation (Chip Select Read Command Byte CSR).

Contains address information: in the CSW Command Byte, the bit positions b3 to b1 are decoded for the three uppermost EEPROM address bits A10, A9, A8 (in the CSR Command Byte, the bit positions b3 to b1 are left undefined).

Control Byte

Following CSW (b0 = 0): contains the eight lower bits of the EEPROM address (EEA) bit A7 to A0, or an additional command byte for the handling of the protection bit.

Following CSR (b0 = 1): contains the data read out, transmitted by the EEPROM. The EEPROM data are read as long as the master pulls down SDA after each byte in order to acknowledge the transfer. The read operation is stopped by the master by releasing SDA (no acknowledge is applied) followed by a STOP condition.

Table 2
Command and Control Byte for I^2C-Bus Addressing of Chip and EEPROM

	Definition								Function
	b7	b6	b5	b4	b3	b2	b1	b0	
CSW	1	c2	$\overline{c1}$	c0	A10	A9	A8	0	Chip Select for Write
CSR	1	c2	$\overline{c1}$	c0	x	x	x	1	Chip Select for Read
EEA	A7	A6	A5	A4	A3	A2	A1	A0	EEPROM address

The device has an internal address counter which points to the current EEPROM address.

The address counter is incremented

after a data byte to be written has been acknowledged, during entry of further data byte

during a byte read, thus the address counter points to the following address after reading a data byte.

The timing conventions for read and write operations are described in **figures 5** and **6**.

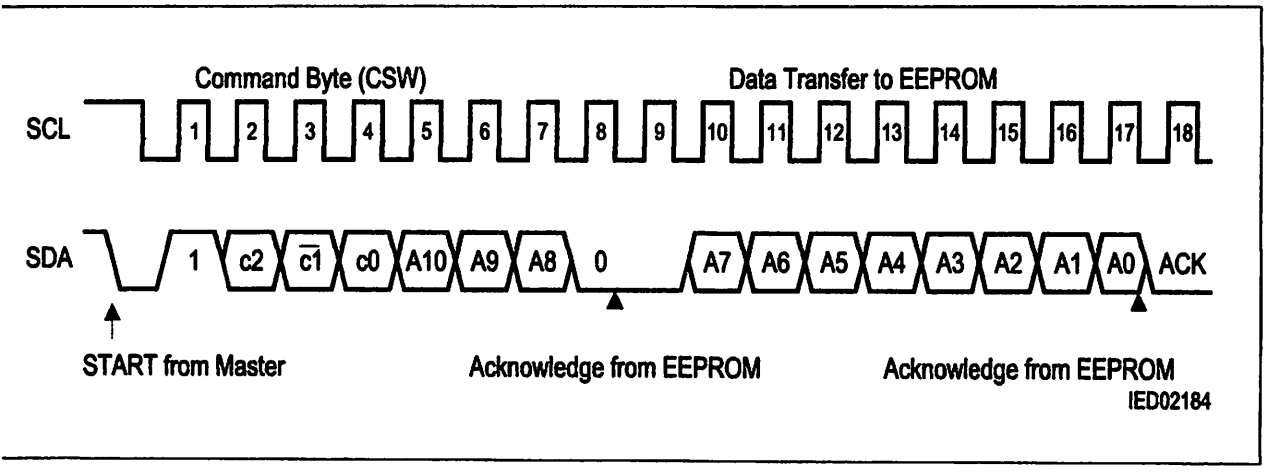


Figure 5
Timing of the Command Byte CSW

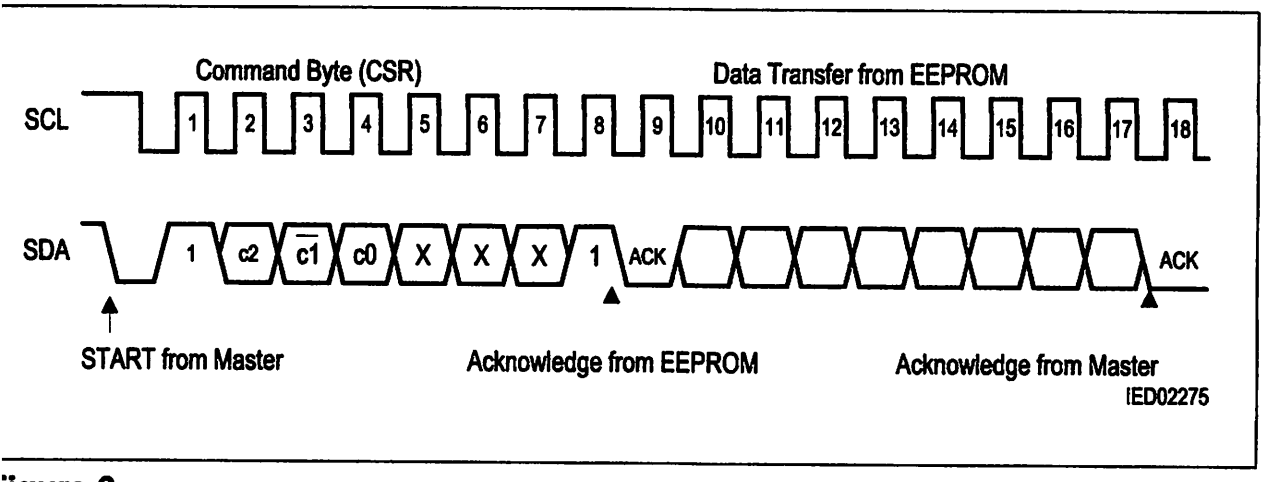


Figure 6
Timing of the Command Byte CSR

5 Write Operations

Changing of the EEPROM data is initiated by the master with the command byte CSW. Depending on the state of the Write Protection pin WP and of the Protection Bits (refer to **chapter 7** Page Protection Mode™) either one byte (Byte Write) or up to 16 byte (Page Write) are modified in one programming procedure.

5.1 Byte Write

Address Setting After a START condition the master transmits the Chip Select Write byte CSW. The EEPROM acknowledges the CSW byte during the ninth clock cycle. The following byte with the EEPROM address (A0 to A7) is loaded into the address counter of the EEPROM and acknowledged by the EEPROM.

Transmission of Data Finally the master transmits the data byte which is also acknowledged by the EEPROM into the internal buffer.

Programming Cycle Then the master applies a STOP condition which starts the internal programming procedure. The data bytes are written in the memory location addressed in the EEA byte (A0 to A7) and the CSW byte (A8 to A10). The programming procedure consists of an internally timed erase/write cycle. In the first step, the selected byte is erased to "1". With the next internal step, the addressed byte is written according to the contents of the buffer.

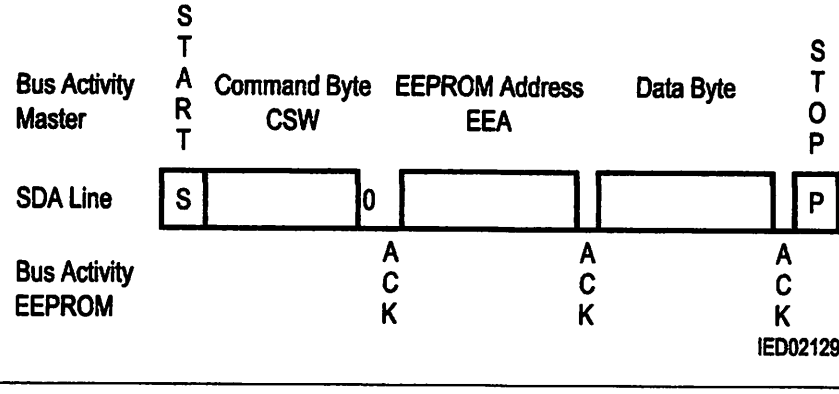


Figure 7
Byte Write Sequence

The erase/write cycle is finished latest after 8 ms. Acknowledge polling may be used for speed enhancement in order to indicate the end of the erase/write cycle (refer to **chapter 5.3** Acknowledge Polling).

5.2 Page Write

Address Setting The page write procedure is the same as the byte write procedure up to the first data byte. In a page write instruction however, entry of the EEPROM address byte EEA is followed by a sequence of one to maximum sixteen data bytes with the new data to be programmed. These bytes are transferred to the internal page buffer of the EEPROM.

Transmission of Data The first entered data byte will be stored according to the EEPROM address n given by EEA (A0 to A7) and CSW (A8 to A10). The internal address counter is incremented automatically after the entered data byte has been acknowledged. The next data byte is then stored at the next higher EEPROM address. EEPROM addresses within the same page have common page address bits A4 through A10. Only the respective four least significant address bits A0 through A3 are incremented, as all data bytes to be programmed simultaneously have to be within the same page.

Programming Cycle The master stops data entry by applying a STOP condition, which also starts the internally timed erase/write cycle. In the first step, all selected bytes are erased to "1". With the next internal step, the addressed bytes are written according to the contents of the page buffer.

Those bytes of the page that have not been addressed are not included in the programming.

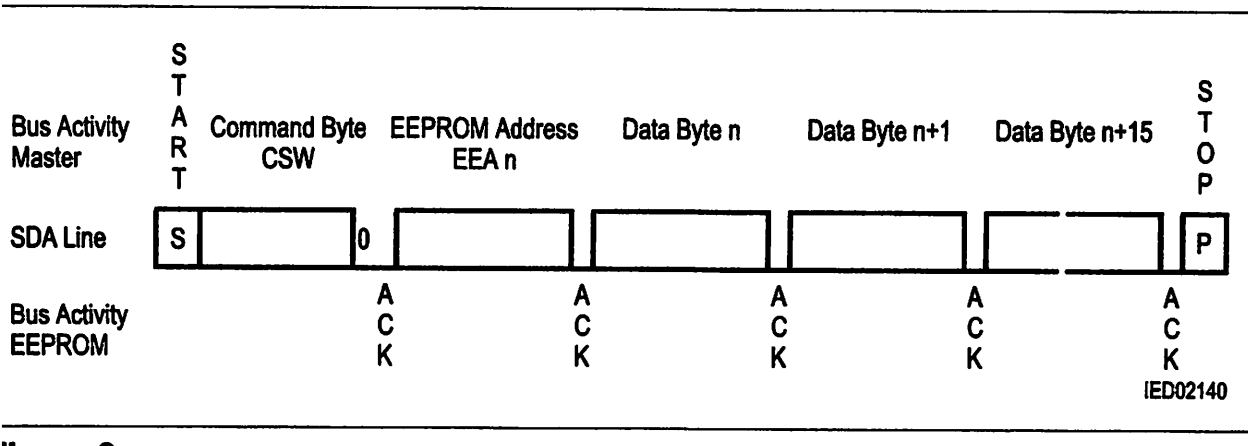


Figure 8
Page Write Sequence

The erase/write cycle is finished latest after 8 ms. Acknowledge polling may be used for speed enhancement in order to indicate the end of the erase/write cycle (refer to **Chapter 5.3** Acknowledge Polling).

5.3 Acknowledge Polling

During the erase/write cycle the EEPROM will not respond to a new command byte until the internal write procedure is completed. At the end of active programming the chip returns to the standby mode and the last entered EEPROM byte remains addressed by the address counter. To determine the end of the internal erase/write cycle acknowledge polling can be initiated by the master by sending a START condition followed by a command byte CSR or CSW (read with b0 = 1 or write with b0 = 0). If the internal erase/write cycle is not completed, the device will not acknowledge the transmission. If the internal erase/write cycle is completed, the device acknowledges the received command byte and the protocol activities can continue.

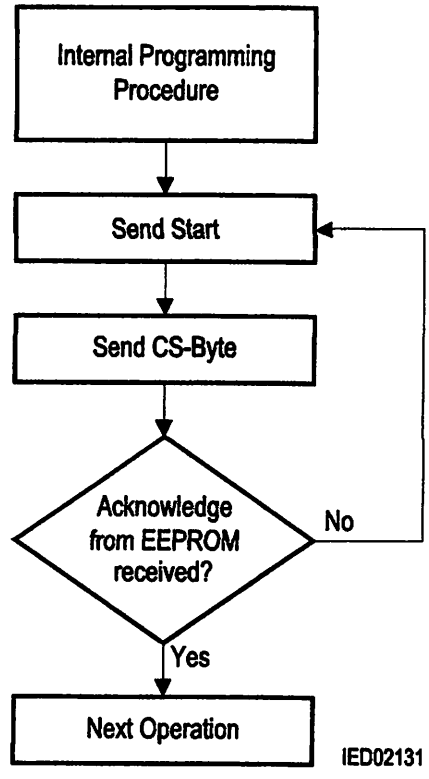


Figure 9
Flow Chart "Acknowledge Polling"

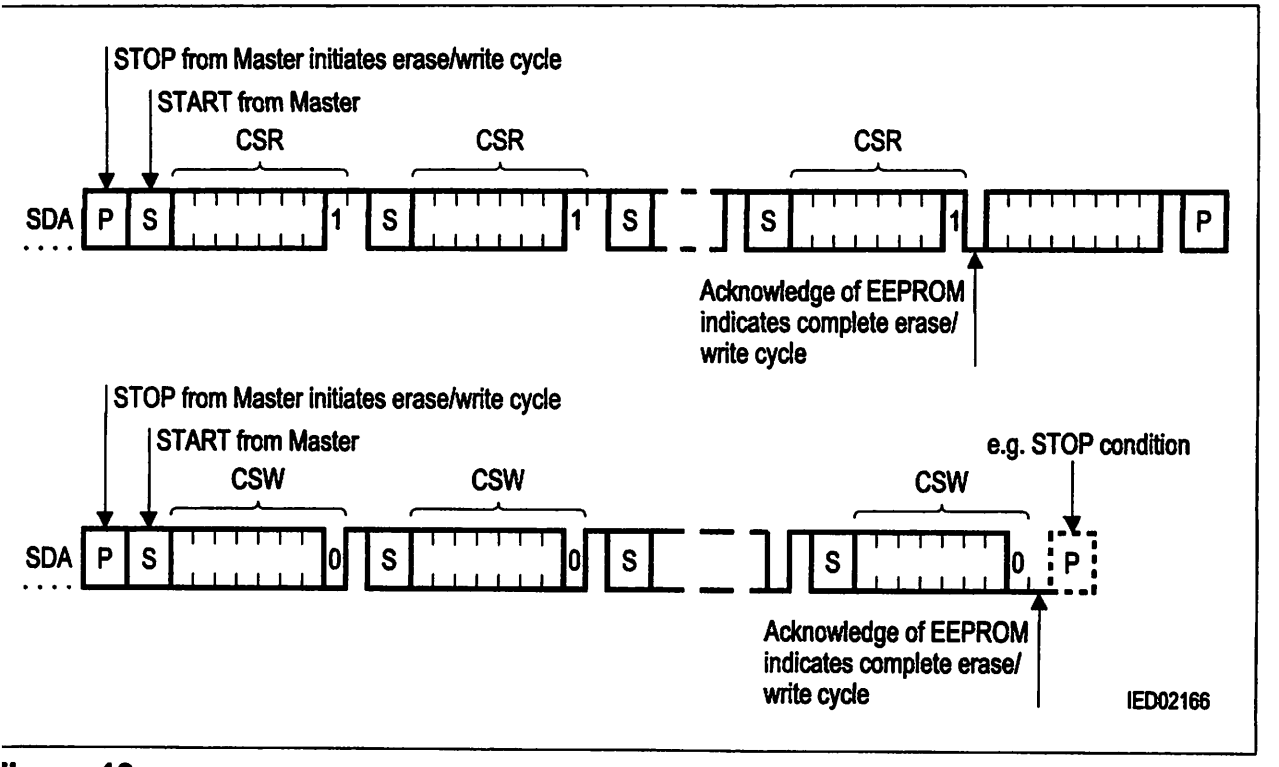


Figure 10
Principle of Acknowledge Polling

Read Operations

Reading of the EEPROM data is initiated by the Master with the command byte CSR.

Random Read

Random read operations allow the master to access any memory location.

Address Setting The master generates a START condition followed by the command byte CSW. The receipt of the CSW-byte is acknowledged by the EEPROM with a low on the SDA line. Now the master transmits the EEPROM address (EEA) to the EEPROM and the internal address counter is loaded with the desired address.

Transmission of CSR After the acknowledge for the EEPROM address is received, the master generates a START condition, which terminates the initiated write operation. Then the master transmits the command byte CSR for read, which is acknowledged by the EEPROM.

Transmission of EEPROM Data During the next eight clock pulses the EEPROM transmits the data byte and increments the internal address counter.

STOP Condition from Master During the following clock cycle the masters releases the bus and then transmits the STOP condition.

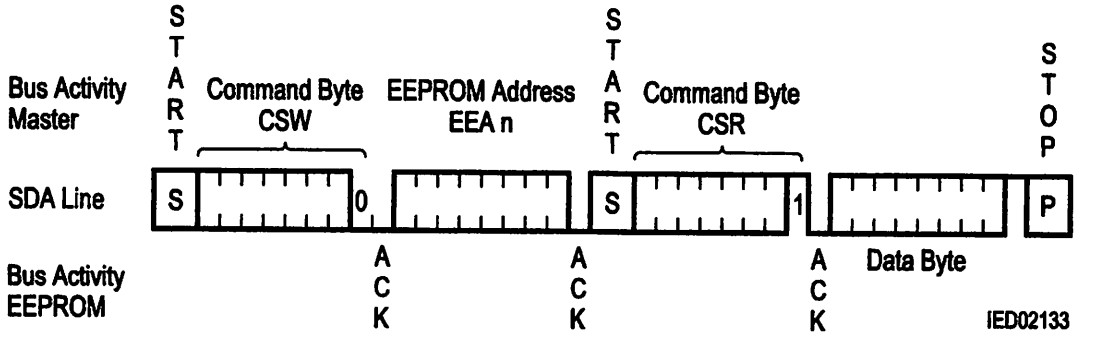


Figure 11
Random Read

6.2 Current Address Read

The EEPROM content is read without setting an EEPROM address, in this case the current content of the address counter will be used (e.g. to continue a previous read operation after the Master has served an interrupt).

Transmission of CSR For a current address read the master generates a START condition, which is followed by the command byte CSR (chip select read). The receipt of the CSR-byte is acknowledged by the EEPROM with a low on the SDA line.

Transmission of EEPROM Data During the next eight clock pulses the EEPROM transmits the data byte and increments the internal address counter.

STOP Condition from Master During the following clock cycle the masters releases the bus and then transmits the STOP condition.

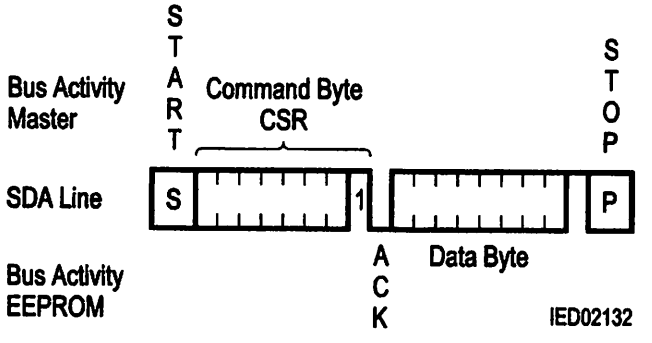


Figure 12
Current Address Read

6.3 Sequential Read

A sequential read is initiated in the same way as a current read or a random read except that the master acknowledges the data byte transmitted by the EEPROM. The EEPROM then continues the data transmission. The internal address counter is incremented by one during each data byte transmission.

A sequential read allows the entire memory to be read during one read operation. After the highest addressable memory location is reached, the internal address pointer "rolls over" to the address 0 and the sequential read continues.

The transmission is terminated by the master by releasing the SDA line (no acknowledge) and generating a STOP condition (see **figure 13**).

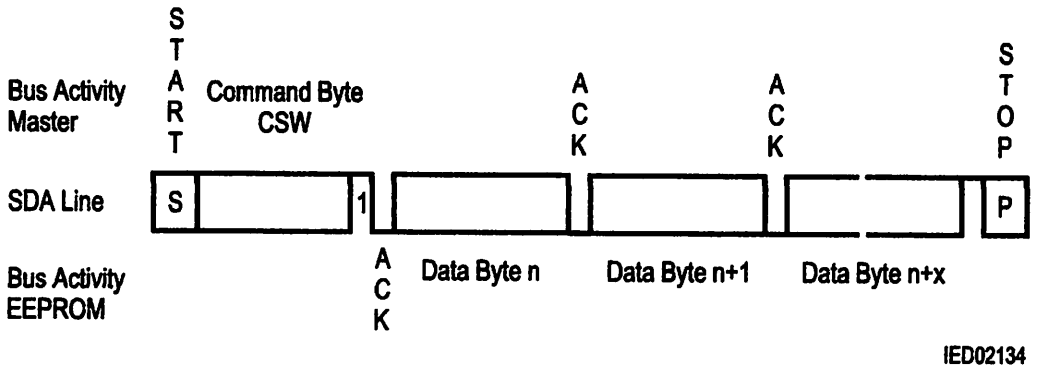


Figure 13
Sequential Read

Page Protection Mode™

Each page (16 byte) in the Data Memory can be protected against unintended data changes by an associated protection bit. The protection bit memory consists of an additional EEPROM of 128 bit (**figure 14**).

Data in the Data Memory can be modified only if the assigned protection bit is erased logical state "1"). After writing the data bytes to a page, the protection is achieved by writing the associated protection bit (logical state "0"). Further changes in the data in a protected page is possible only after erasing the protection bit.

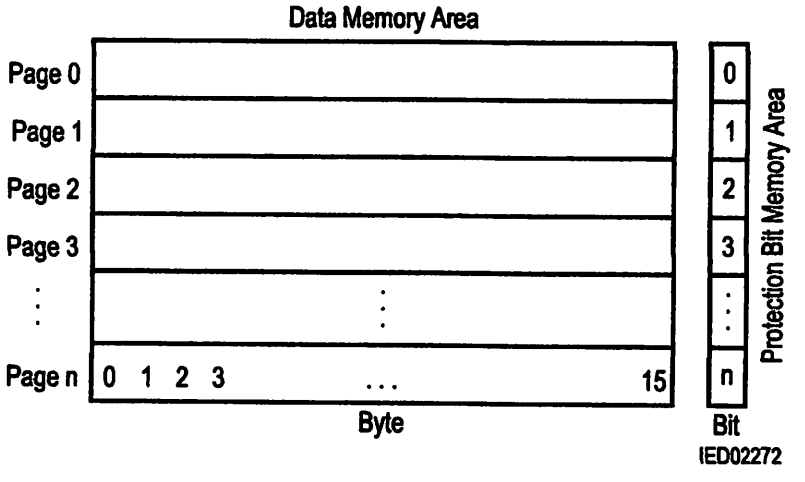


Figure 14
Data Page and Assigned Protection Memory

A special procedure to write or erase a protection bit guarantees proper activation or deactivation respectively of page protection. For protection bit write or erase, all 16 data bytes of the respective page have to be entered for a second time. The data then are compared internally with the data to be protected, and in case of identity the protection bit is written or erased respectively.

.1 Protection Bit Handling

he bits of the protection memory can be addressed directly for reading or programming. protection bit address corresponds to the lowest address within the respective page A4 to A10, A0 to A3 = zero). The status of each protection bit is sensed internally. A ritten state ("0") prevents programming in the associated page. If an already protected emory page is accidentally addressed for programming, the programming procedure s suppressed.

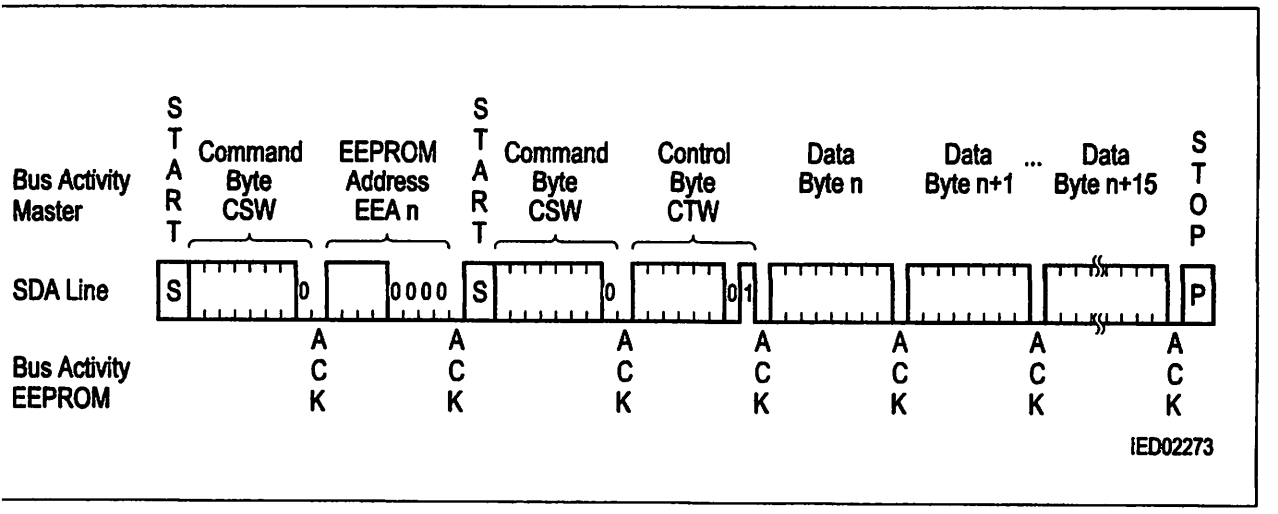
he conventional I^2C-Bus protocol allows data bytes to be read and programmed only. herefore an independent instruction sequence for addressing and manipulation of rotection bits is implemented. For protection bit instructions, the command byte CSW ith its preceding START condition followed by the associated control byte has to be ntered twice (**figures 15** through **17**). The first command byte CSW (with A8 to A10) is llowed by the control byte EEA with the bit/page address A0 through A3 always at zero. he second CSW is required for entering a control byte CTx for protection bit anipulation. The three control bytes for read, write or erase of a protection bit are listed elow (**table 3**):

able 3
ontrol Byte for Protection Bit Manipulation

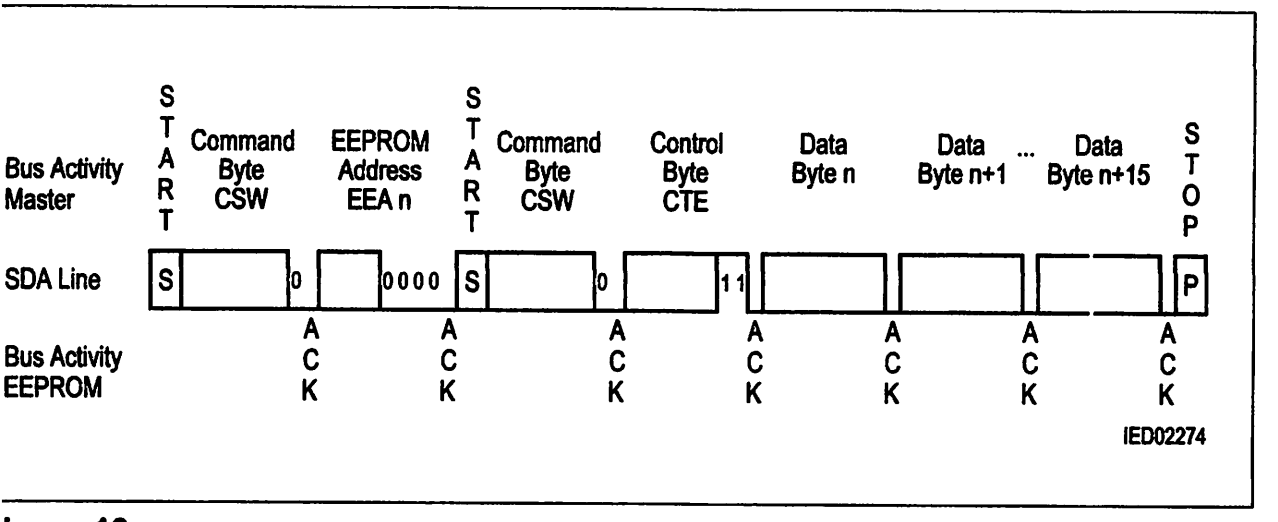
ddress lame	Definition								Function
	b7	b6	b5	b4	b3	b2	b1	b0	
TR	x	x	x	x	x	x	0	0	Protection bit read
TW	x	x	x	x	x	x	0	1	Protection bit write
TE	x	x	x	x	x	x	1	1	Protection bit erase

'.2 Protection Bit Write and Erase

For writing or erasing a protection bit, the data of the respective page have to be known by the master. The data of the page are not affected by the write or erase procedure of he protection bit. The I^2C-Bus protocol is shown in **figure 15** for protection bit write and **igure 16** for protection bit erase.



igure 15
equence for Protection Bit Write



igure 16
equence for Protection Bit Erase

he first command byte CSW followed by the control byte EEA addresses the page to e protected. The second command byte CSW (identical content of first CSW) is llowed by the control byte CTW = 01_H for protection bit write or CTE = 03_H for protection it erase. Depending on CTx, the addressed protection bit will be either written or rased.

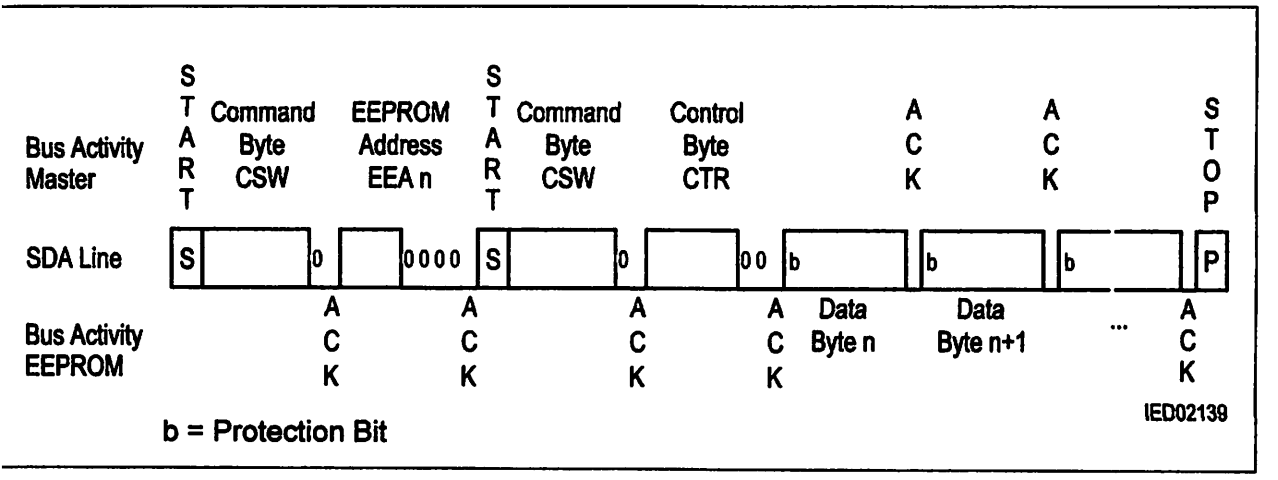
The control byte CTx is followed by 16 parameter bytes identical to the 16 data bytes of the page to be protected or unprotected. The data of the first entered byte must be identical to the data byte stored at the lowest address of the current page. The other 15 bytes have to be identical to the bytes stored in ascending address order within the same page.

A successful verification of each byte is indicated by the EEPROM by pulling the SDA line to low (acknowledge ACK).

After verification of the last byte, the bit programming procedure is initiated by the STOP condition. Programming is started only if all 128 bits of a page have been verified successfully. If bit programming has taken place, the address counter points to the uppermost address of the respective page. The write or erase cycle is finished latest after 4 ms. Acknowledge polling may be used for speed enhancement in order to indicate the end of the write or erase cycle (refer to **chapter 5.3** Acknowledge Polling).

.3 Protection Bit Read

he byte sequence for random bit read is shown in **figure 17**.



igure 17
yte Sequence for Protection Bit Read

he first command byte CSW followed by the control byte EEA addresses the protection it to be read. The second command byte CSW is followed by the control byte 00_H for rotection bit read. The first bit (MSB) of the transferred byte is the protection bit of the ddressed page. The other 7 bits are not valid. The page protection status is indicated s following

rotection Bit = 1: A normal write operation changes the data in the associated page
rotection Bit = 0: The data in the associated page are protected against changes.

the master acknowledges a byte with a low state of the SDA line, the protection bit of ie next page can be read as the first bit of the following byte. If the master releases the DA line, a STOP condition has to complete the read procedure. Any number of bytes ith a page protection status at the first bit position can be requested by the master. If ie bit of the uppermost page has been addressed, the counter has its overflow to the west address according to the first page.

Electrical Characteristics

The listed characteristics are ensured over the operating range of the integrated circuit. Typical characteristics specify mean values expected over the production spread. If not otherwise specified, typical characteristics apply at T_A = 25 °C and the given supply voltage.

.1 Absolute Maximum Ratings

Stresses above those listed here 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 above those indicated in the operational section of this data sheet is not implied. Exposure to absolute maximum ratings for extended periods may affect device reliability.

Parameter		Limit Values	Units
Operating temperature	range 1 (industrial) range 2 (automotive)	– 40 to + 85 – 40 to + 125	°C °C
Storage temperature		– 65 to + 150	°C
Supply voltage		– 0.3 to + 7.0	V
All inputs and outputs with respect to ground		– 0.3 to V_{CC} + 0.5	V
ESD protection (human body model)		4000	V

.2 DC Characteristics

Parameter	Symbol	Limit Values			Units	Test Condition
		min.	typ.	max.		
Supply voltage	V_{CC}	4.5		5.5	V	5 V type
	V_{CC}	2.7		5.5	V	3 V type
Supply current[1] (write)	I_{CC}		1	3	mA	V_{CC} = 5 V; f_c = 100 kHz
Standby current[2]	I_{SB}			50	µA	Inputs at V_{CC} or V_{SS}
Input leakage current	I_{LI}		0.1	10	µA	V_{IN} = V_{CC} or V_{SS}
Output leakage current	I_{LO}		0.1	10	µA	V_{OUT} = V_{CC} or V_{SS}
Input low voltage	V_{IL}	– 0.3		0.3 × V_{CC}	V	

.2 DC Characteristics (cont'd)

Parameter	Symbol	Limit Values			Units	Test Condition
		min.	typ.	max.		
nput high oltage	V_{IH}	$0.7 \times V_{CC}$		V_{CC} + 0.5	V	
)utput low oltage	V_{OL}			0.4	V	I_{OL} = 3 mA; V_{CC} = 5 V I_{OL} = 2.1 mA; V_{CC} = 3 V
nput/output apacitance SDA)	$C_{I/O}$			8[3)]	pF	V_{IN} = 0 V; V_{CC} = 5 V
nput apacitance ther pins)	C_{IN}			6[3)]	pF	V_{IN} = 0 V; V_{CC} = 5 V
apacitive load r each bus line	C_b			400	pF	

The values for I_{cc} are maximum peak values

Valid over the whole temperature range

This parameter is characterized only

.3 AC Characteristics

Parameter	Symbol	Limit Values V_{CC} = 2.7-5.5 V		Limit Values V_{CC} = 4.5-5.5 V		Units
		min.	max.	min.	max.	
CL clock frequency	f_{SCL}		100		400	kHz
lock pulse width low	t_{low}	4.7		1.2		µs
lock pulse width high	t_{high}	4.0		0.6		µs
DA and SCL rise time	t_R		1000	1)	300	ns
DA and SCL fall time	t_F		300	1)	300	ns
tart set-up time	$t_{SU.STA}$	4.7		0.6		µs
tart hold time	$t_{HD.STA}$	4.0		0.6		µs
ata in set-up time	$t_{SU.DAT}$	200		100		ns
ata in hold time	$t_{HD.DAT}$	0		0		µs
CL low to SDA data out valid	t_{AA}	0.1	4.5	0.1	0.9	µs
ata out hold time	t_{DH}	100		50		ns
top set-up time	$t_{SU.STO}$	4.0		0.6		µs
ime the bus must be free before new transmission can start	t_{BUF}	4.7		1.2		µs
DA and SCL spike suppression me at constant inputs	t_I	50	100	50	100	ns

The minimum rise and fall times can be calculated as follows: 20 + (0.1/pF) × C_b [ns]
Example: C_b = 100 pF → t_R = 20 + 0.1 × 100 [ns] = 30 ns

.4 Erase and Write Characteristics

arameter	Symbol	Limit Values V_{CC} = 2.7-5.5 V		Limit Values V_{CC} = 4.5-5.5 V		Units
		typ.	max.	typ.	max.	
rase + write cycle (per page)	t_{WR}	5	8	5	8	ms
rase page protection bit		2.5	4	2.5	4	ms
/rite page protection bit		2.5	4	2.5	4	ms

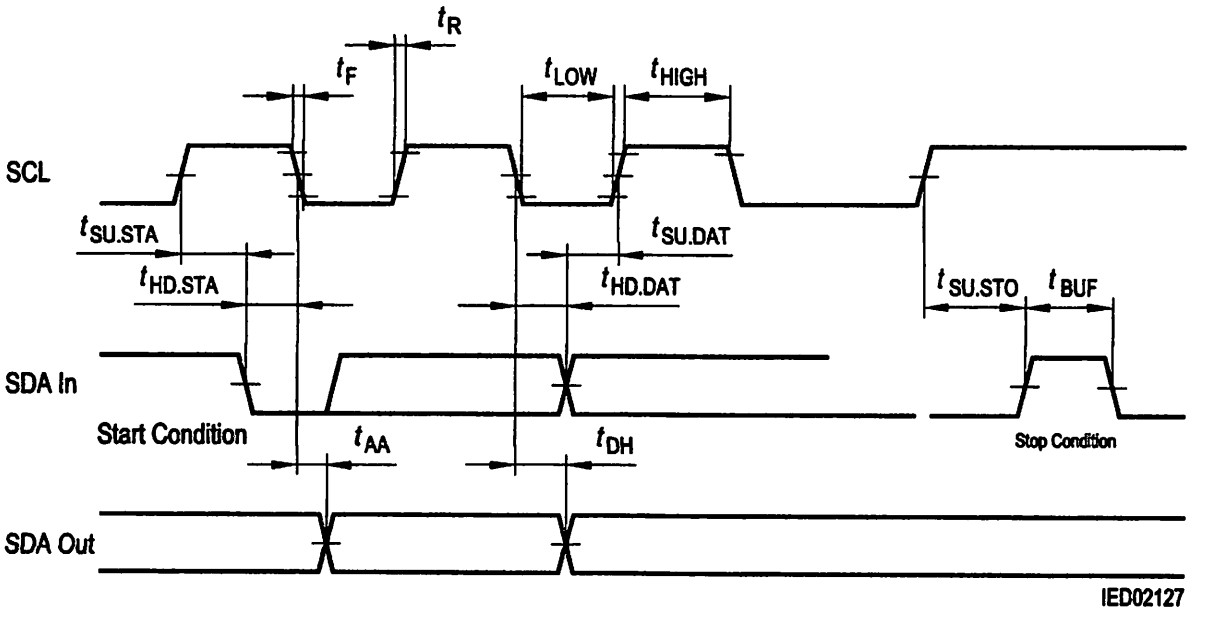
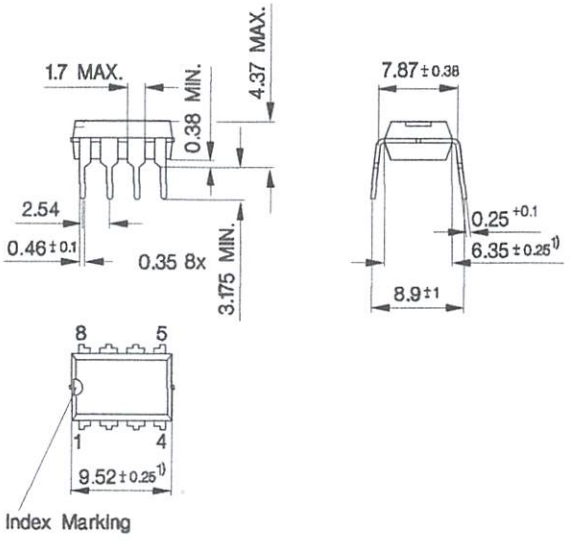


Figure 18
Bus Timing Data

Package Outlines

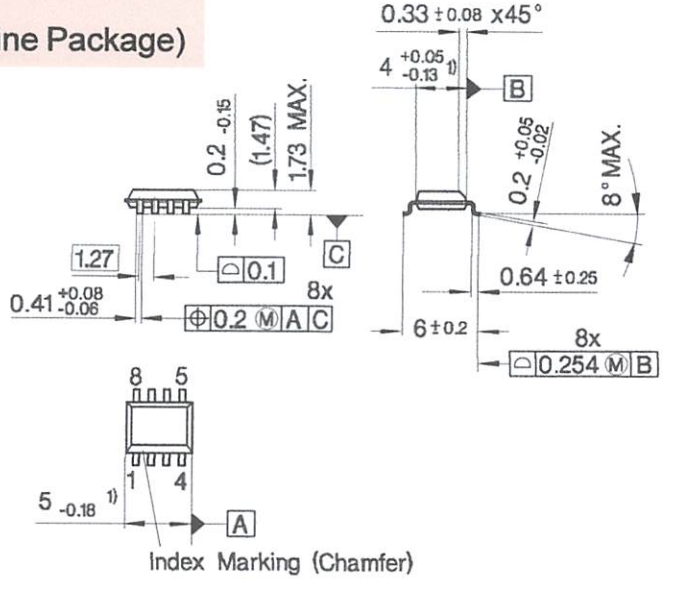
P-DIP-8-4
(Plastic Dual In-line Package)



1) Does not include plastic or metal protrusion of 0.25 max. per side

GPD05583

P-DSO-8-3
(Plastic Dual Small Outline Package)



1) Does not include plastic or metal protrusion of 0.15 max. per side

GPS09032

orts of Packing

ackage outlines for tubes, trays etc. are contained in our
ata Book "Package Information".

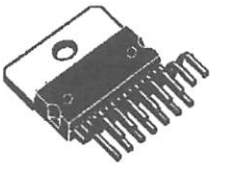
MD = Surface Mounted Device

Dimensions in mm

L298

DUAL FULL-BRIDGE DRIVER

- OPERATING SUPPLY VOLTAGE UP TO 46 V
- TOTAL DC CURRENT UP TO 4 A
- LOW SATURATION VOLTAGE
- OVERTEMPERATURE PROTECTION
- LOGICAL "0" INPUT VOLTAGE UP TO 1.5 V (HIGH NOISE IMMUNITY)



Multiwatt15

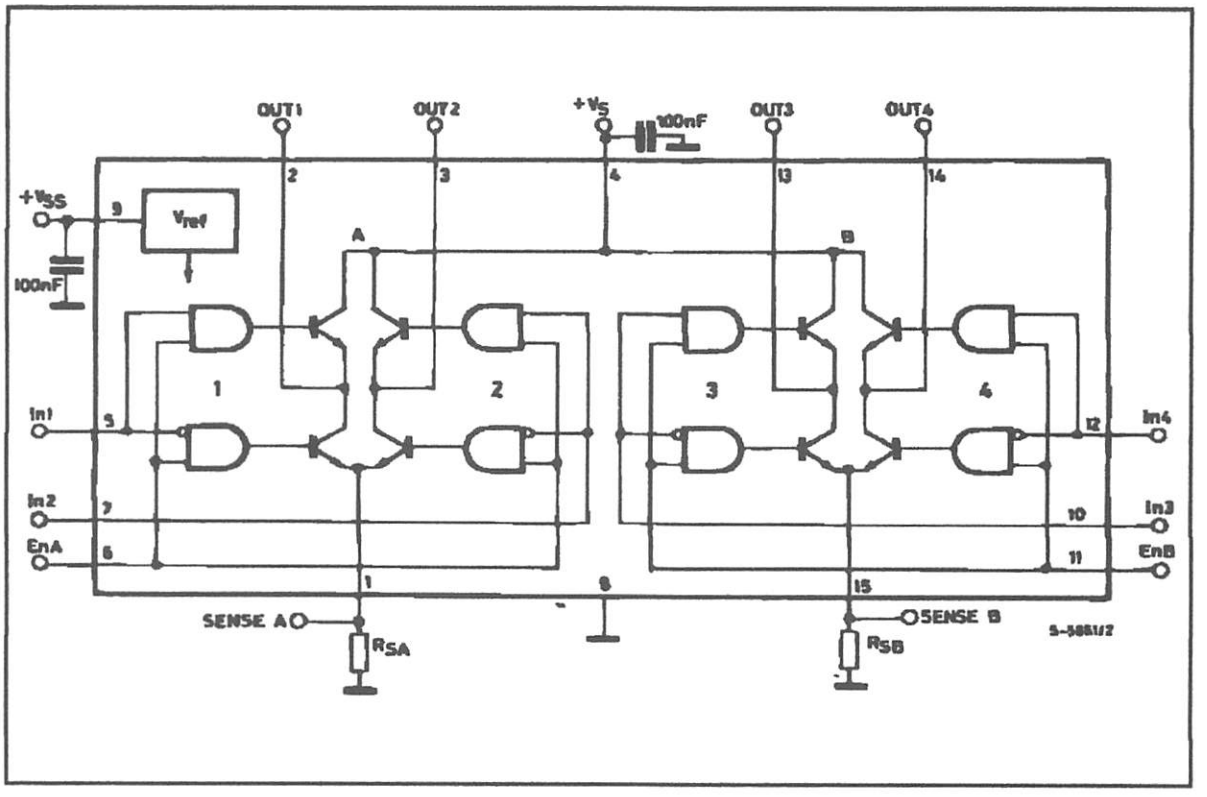
PowerSO20

ORDERING NUMBERS : L298N (Multiwatt Vert.)
L298HN (Multiwatt Horiz.)
L298P (PowerSO20)

DESCRIPTION

The L298 is an integrated monolithic circuit in a 15-lead Multiwatt and PowerSO20 packages. It is a high voltage, high current dual full-bridge driver designed to accept standard TTL logic levels and drive inductive loads such as relays, solenoids, DC and stepping motors. Two enable inputs are provided to enable or disable the device independently of the input signals. The emitters of the lower transistors of each bridge are connected together and the corresponding external terminal can be used for the connection of an external sensing resistor. An additional supply input is provided so that the logic works at a lower voltage.

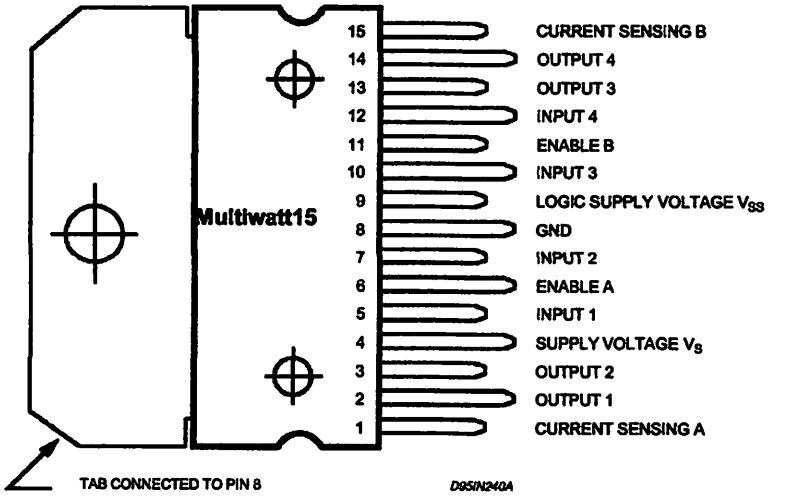
BLOCK DIAGRAM

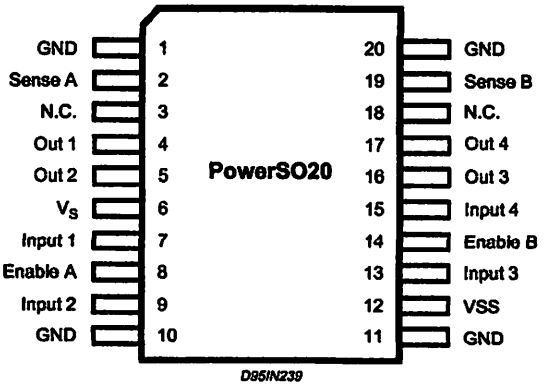


ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_S	Power Supply	50	V
V_{SS}	Logic Supply Voltage	7	V
V_i, V_{en}	Input and Enable Voltage	–0.3 to 7	V
I_O	Peak Output Current (each Channel) – Non Repetitive (t = 100µs) –Repetitive (80% on –20% off; t_{on} = 10ms) –DC Operation	 3 2.5 2	 A A A
V_{sens}	Sensing Voltage	–1 to 2.3	V
P_{tot}	Total Power Dissipation (T_{case} = 75°C)	25	W
T_{op}	Junction Operating Temperature	–25 to 130	°C
T_{stg}, T_j	Storage and Junction Temperature	–40 to 150	°C

PIN CONNECTIONS (top view)





THERMAL DATA

Symbol	Parameter		PowerSO20	Multiwatt15	Unit
$R_{th\ j\text{-}case}$	Thermal Resistance Junction-case	Max.	–	3	°C/W
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	13 (*)	35	°C/W

(*) Mounted on aluminum substrate

PIN FUNCTIONS (refer to the block diagram)

MW.15	PowerSO	Name	Function
1;15	2;19	Sense A; Sense B	Between this pin and ground is connected the sense resistor to control the current of the load.
2;3	4;5	Out 1; Out 2	Outputs of the Bridge A; the current that flows through the load connected between these two pins is monitored at pin 1.
4	6	V_S	Supply Voltage for the Power Output Stages. A non-inductive 100nF capacitor must be connected between this pin and ground.
5;7	7;9	Input 1; Input 2	TTL Compatible Inputs of the Bridge A.
6;11	8;14	Enable A; Enable B	TTL Compatible Enable Input: the L state disables the bridge A (enable A) and/or the bridge B (enable B).
8	1,10,11,20	GND	Ground.
9	12	VSS	Supply Voltage for the Logic Blocks. A100nF capacitor must be connected between this pin and ground.
10; 12	13;15	Input 3; Input 4	TTL Compatible Inputs of the Bridge B.
13; 14	16;17	Out 3; Out 4	Outputs of the Bridge B. The current that flows through the load connected between these two pins is monitored at pin 15.
–	3;18	N.C.	Not Connected

ELECTRICAL CHARACTERISTICS (V_S = 42V; V_{SS} = 5V, T_j = 25°C; unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
V_S	Supply Voltage (pin 4)	Operative Condition	V_{IH} +2.5		46	V
V_{SS}	Logic Supply Voltage (pin 9)		4.5	5	7	V
I_S	Quiescent Supply Current (pin 4)	V_{en} = H; I_L = 0 V_i = L V_i = H		13 50	22 70	mA mA
		V_{en} = L V_i = X			4	mA
I_{SS}	Quiescent Current from V_{SS} (pin 9)	V_{en} = H; I_L = 0 V_i = L V_i = H		24 7	36 12	mA mA
		V_{en} = L V_i = X			6	mA
V_{iL}	Input Low Voltage (pins 5, 7, 10, 12)		–0.3		1.5	V
V_{iH}	Input High Voltage (pins 5, 7, 10, 12)		2.3		VSS	V
I_{iL}	Low Voltage Input Current (pins 5, 7, 10, 12)	V_i = L			–10	µA
I_{iH}	High Voltage Input Current (pins 5, 7, 10, 12)	Vi = H ≤ V_{SS} –0.6V		30	100	µA
V_{en} = L	Enable Low Voltage (pins 6, 11)		–0.3		1.5	V
V_{en} = H	Enable High Voltage (pins 6, 11)		2.3		V_{SS}	V
I_{en} = L	Low Voltage Enable Current (pins 6, 11)	V_{en} = L			–10	µA
I_{en} = H	High Voltage Enable Current (pins 6, 11)	V_{en} = H ≤ V_{SS} –0.6V		30	100	µA
$V_{CEsat(H)}$	Source Saturation Voltage	I_L = 1A I_L = 2A	0.95	1.35 2	1.7 2.7	V V
$V_{CEsat(L)}$	Sink Saturation Voltage	I_L = 1A (5) I_L = 2A (5)	0.85	1.2 1.7	1.6 2.3	V V
V_{CEsat}	Total Drop	I_L = 1A (5) I_L = 2A (5)	1.80		3.2 4.9	V V
V_{sens}	Sensing Voltage (pins 1, 15)		–1 (1)		2	V

ELECTRICAL CHARACTERISTICS (continued)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit
T_1 (V_i)	Source Current Turn-off Delay	0.5 V_i to 0.9 I_L (2); (4)		1.5		µs
T_2 (V_i)	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		0.2		µs
T_3 (V_i)	Source Current Turn-on Delay	0.5 V_i to 0.1 I_L (2); (4)		2		µs
T_4 (V_i)	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.7		µs
T_5 (V_i)	Sink Current Turn-off Delay	0.5 V_i to 0.9 I_L (3); (4)		0.7		µs
T_6 (V_i)	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.25		µs
T_7 (V_i)	Sink Current Turn-on Delay	0.5 V_i to 0.9 I_L (3); (4)		1.6		µs
T_8 (V_i)	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.2		µs
fc (V_i)	Commutation Frequency	I_L = 2A		25	40	KHz
T_1 (V_{en})	Source Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (2); (4)		3		µs
T_2 (V_{en})	Source Current Fall Time	0.9 I_L to 0.1 I_L (2); (4)		1		µs
T_3 (V_{en})	Source Current Turn-on Delay	0.5 V_{en} to 0.1 I_L (2); (4)		0.3		µs
T_4 (V_{en})	Source Current Rise Time	0.1 I_L to 0.9 I_L (2); (4)		0.4		µs
T_5 (V_{en})	Sink Current Turn-off Delay	0.5 V_{en} to 0.9 I_L (3); (4)		2.2		µs
T_6 (V_{en})	Sink Current Fall Time	0.9 I_L to 0.1 I_L (3); (4)		0.35		µs
T_7 (V_{en})	Sink Current Turn-on Delay	0.5 V_{en} to 0.9 I_L (3); (4)		0.25		µs
T_8 (V_{en})	Sink Current Rise Time	0.1 I_L to 0.9 I_L (3); (4)		0.1		µs

1) 1)Sensing voltage can be –1 V for t ≤ 50 µsec; in steady state V_{sens} min ≥ –0.5 V.
2) See fig. 2.
3) See fig. 4.
4) The load must be a pure resistor.

Figure 1 : Typical Saturation Voltage vs. Output Current.

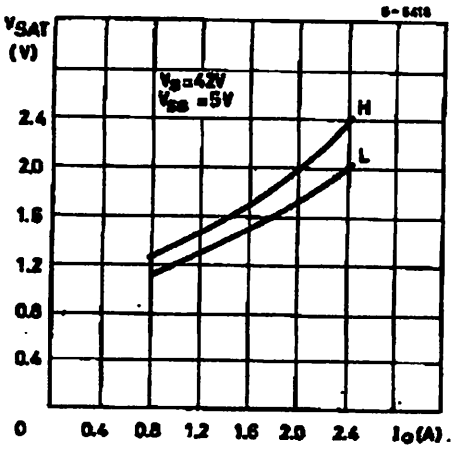
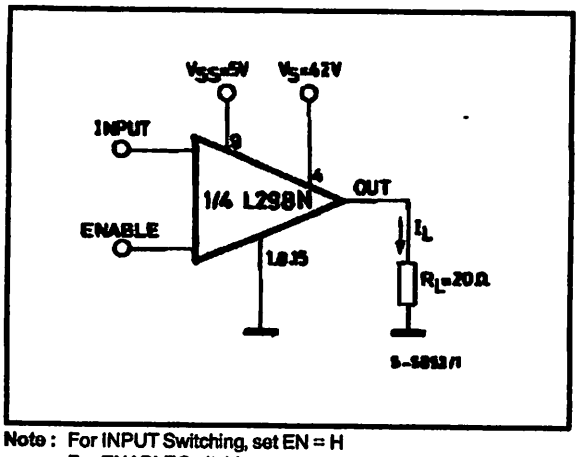


Figure 2 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = H

Figure 3 : Source Current Delay Times vs. Input or Enable Switching.

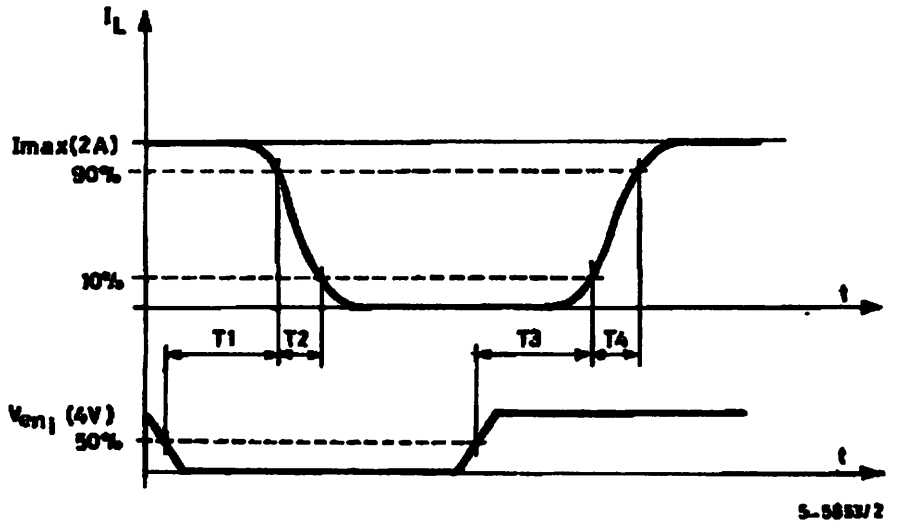
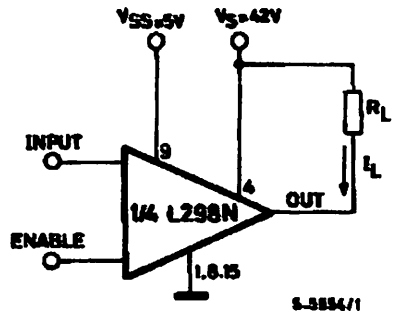


Figure 4 : Switching Times Test Circuits.



Note : For INPUT Switching, set EN = H
For ENABLE Switching, set IN = L

Figure 5 : Sink Current Delay Times vs. Input 0 V Enable Switching.

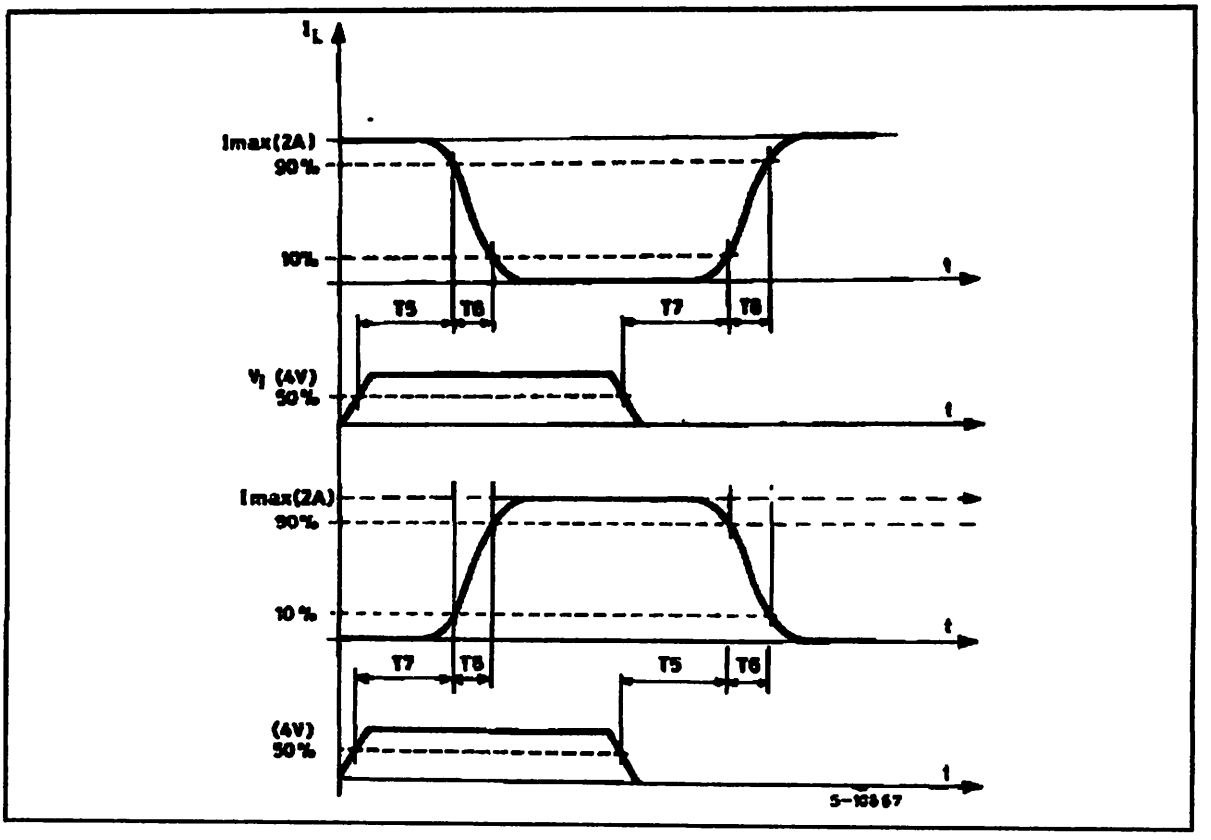
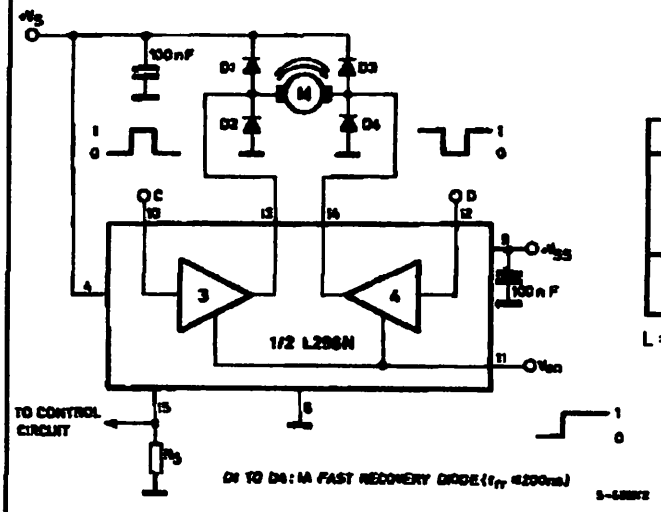


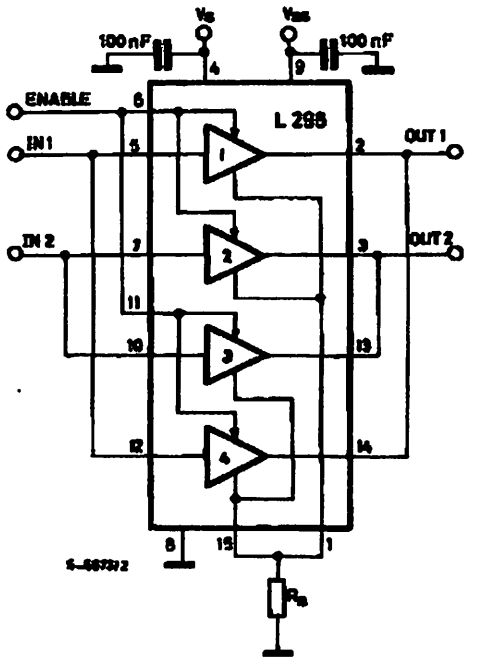
Figure 6 : Bidirectional DC Motor Control.



Inputs		Function
V_{en} = H	C = H ; D = L	Forward
	C = L ; D = H	Reverse
	C = D	Fast Motor Stop
V_{en} = L	C = X ; D = X	Free Running Motor Stop

L = Low H = High X = Don't care

Figure 7 : For higher currents, outputs can be paralleled. Take care to parallel channel 1 with channel 4 and channel 2 with channel 3.



APPLICATION INFORMATION (Refer to the block diagram)

1.1. POWER OUTPUT STAGE

The L298 integrates two power output stages (A; B). The power output stage is a bridge configuration and its outputs can drive an inductive load in common or differenzial mode, depending on the state of the inputs. The current that flows through the load comes out from the bridge at the sense output: an external resistor (R_{SA} ; R_{SB}.) allows to detect the intensity of this current.

1.2. INPUT STAGE

Each bridge is driven by means of four gates the input of which are In1 ; In2 ; EnA and In3 ; In4 ; EnB. The In inputs set the bridge state when The En input is high ; a low state of the En input inhibits the bridge. All the inputs are TTL compatible.

2. SUGGESTIONS

A non inductive capacitor, usually of 100 nF, must be foreseen between both Vs and Vss, to ground, as near as possible to GND pin. When the large capacitor of the power supply is too far from the IC, a second smaller one must be foreseen near the L298.

The sense resistor, not of a wire wound type, must be grounded near the negative pole of Vs that must be near the GND pin of the I.C.

Each input must be connected to the source of the driving signals by means of a very short path.

Turn-On and Turn-Off : Before to Turn-ON the Supply Voltage and before to Turn it OFF, the Enable input must be driven to the Low state.

3. APPLICATIONS

Fig 6 shows a bidirectional DC motor control Schematic Diagram for which only one bridge is needed. The external bridge of diodes D1 to D4 is made by four fast recovery elements ($t_{rr} \leq 200$ nsec) that must be chosen of a VF as low as possible at the worst case of the load current.

The sense output voltage can be used to control the current amplitude by chopping the inputs, or to provide overcurrent protection by switching low the enable input.

The brake function (Fast motor stop) requires that the Absolute Maximum Rating of 2 Amps must never be overcome.

When the repetitive peak current needed from the load is higher than 2 Amps, a paralleled configuration can be chosen (See Fig.7).

An external bridge of diodes are required when inductive loads are driven and when the inputs of the IC are chopped; Shottky diodes would be preferred.

This solution can drive until 3 Amps in DC operation and until 3.5 Amps of a repetitive peak current.

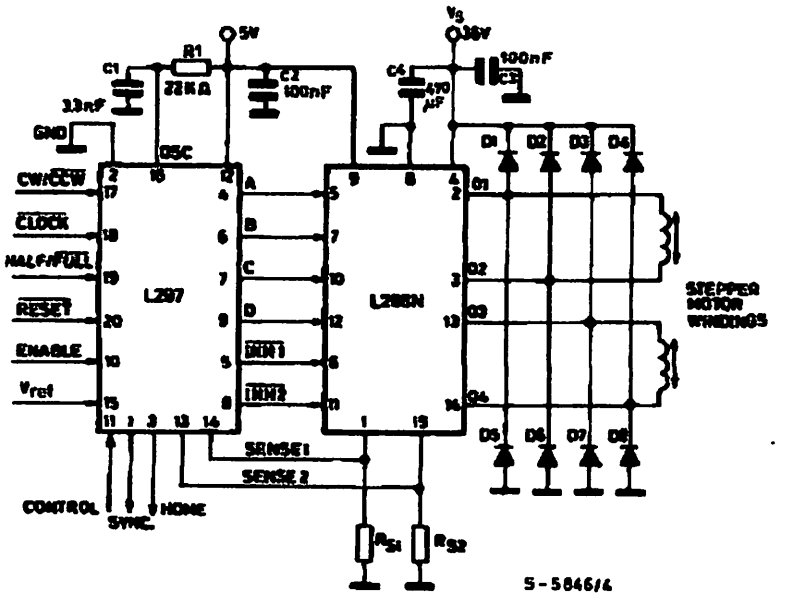
On Fig 8 it is shown the driving of a two phase bipolar stepper motor ; the needed signals to drive the inputs of the L298 are generated, in this example, from the IC L297.

Fig 9 shows an example of P.C.B. designed for the application of Fig 8.

Fig 10 shows a second two phase bipolar stepper motor control circuit where the current is controlled by the I.C. L6506.

Figure 8 : Two Phase Bipolar Stepper Motor Circuit.

This circuit drives bipolar stepper motors with winding currents up to 2 A. The diodes are fast 2 A types.



$R_{S1} = R_{S2} = 0.5\,\Omega$

D1 to D8 = 2 A Fast diodes { $V_F \leq 1.2\,V$ @ $I = 2\,A$; $t_{rr} \leq 200\,ns$

Figure 9 : Suggested Printed Circuit Board Layout for the Circuit of fig. 8 (1:1 scale).

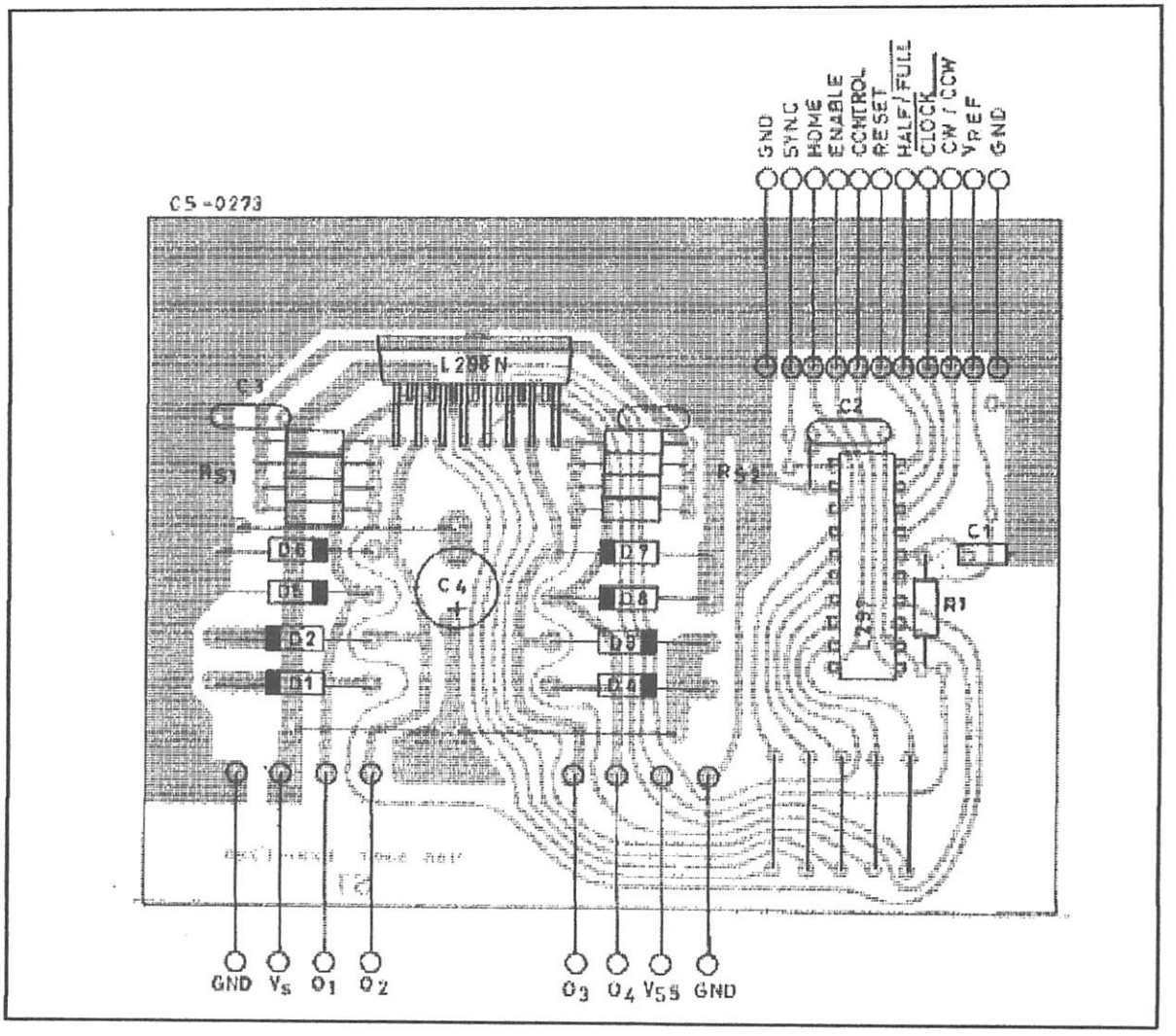
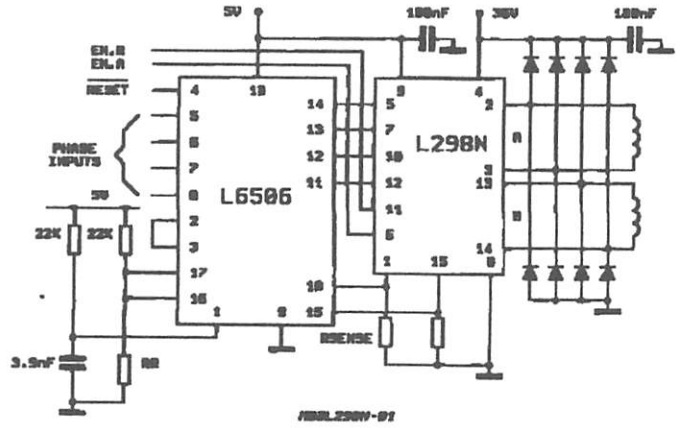


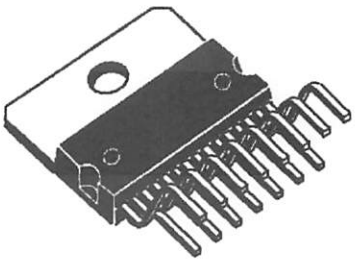
Figure 10 : Two Phase Bipolar Stepper Motor Control Circuit by Using the Current Controller L6506.



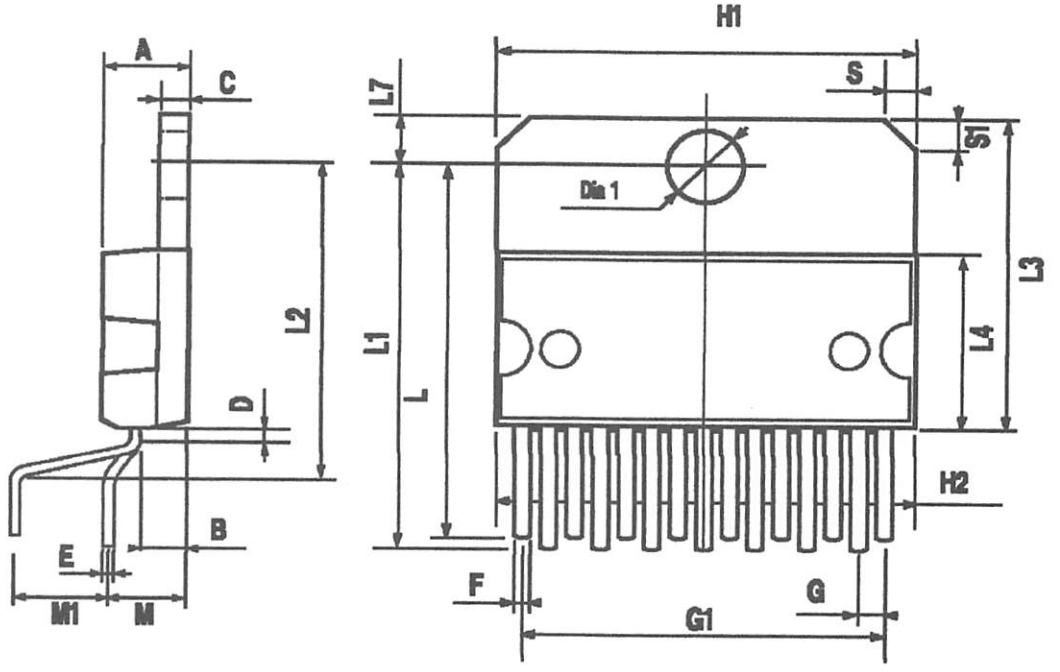
R_R and R_{sense} depend from the load current

DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
D		1			0.039	
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.02	1.27	1.52	0.040	0.050	0.060
G1	17.53	17.78	18.03	0.690	0.700	0.710
H1	19.6			0.772		
H2			20.2			0.795
L	21.9	22.2	22.5	0.862	0.874	0.886
L1	21.7	22.1	22.5	0.854	0.870	0.886
L2	17.65		18.1	0.695		0.713
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L7	2.65		2.9	0.104		0.114
M	4.25	4.55	4.85	0.167	0.179	0.191
M1	4.63	5.08	5.53	0.182	0.200	0.218
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA

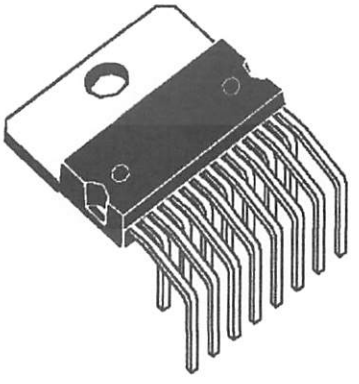


Multiwatt15 V

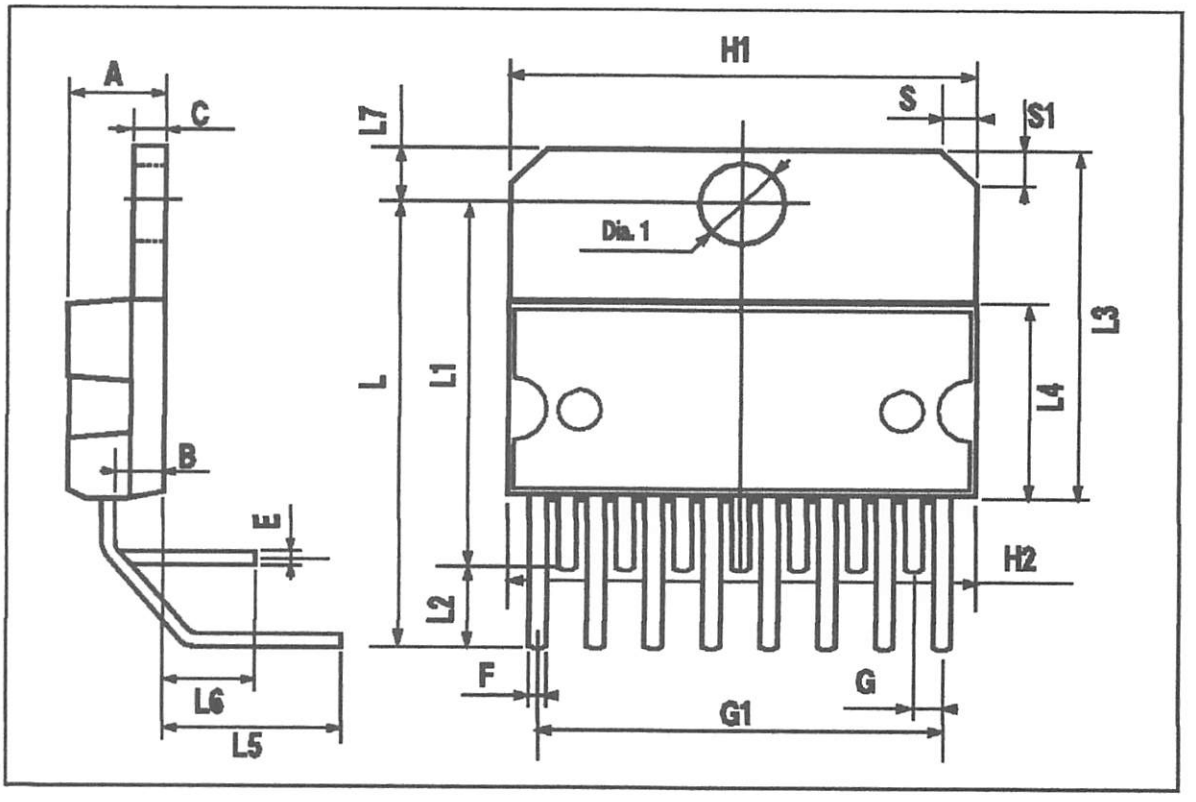


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			5			0.197
B			2.65			0.104
C			1.6			0.063
E	0.49		0.55	0.019		0.022
F	0.66		0.75	0.026		0.030
G	1.14	1.27	1.4	0.045	0.050	0.055
G1	17.57	17.78	17.91	0.692	0.700	0.705
H1	19.6			0.772		
H2			20.2			0.795
L		20.57			0.810	
L1		18.03			0.710	
L2		2.54			0.100	
L3	17.25	17.5	17.75	0.679	0.689	0.699
L4	10.3	10.7	10.9	0.406	0.421	0.429
L5		5.28			0.208	
L6		2.38			0.094	
L7	2.65		2.9	0.104		0.114
S	1.9		2.6	0.075		0.102
S1	1.9		2.6	0.075		0.102
Dia1	3.65		3.85	0.144		0.152

OUTLINE AND MECHANICAL DATA



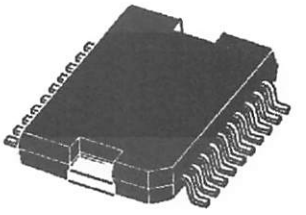
Multiwatt15 H



DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			3.6			0.142
a1	0.1		0.3	0.004		0.012
a2			3.3			0.130
a3	0		0.1	0.000		0.004
b	0.4		0.53	0.016		0.021
c	0.23		0.32	0.009		0.013
D (1)	15.8		16	0.622		0.630
D1	9.4		9.8	0.370		0.386
E	13.9		14.5	0.547		0.570
e		1.27			0.050	
e3		11.43			0.450	
E1 (1)	10.9		11.1	0.429		0.437
E2			2.9			0.114
E3	5.8		6.2	0.228		0.244
G	0		0.1	0.000		0.004
H	15.5		15.9	0.610		0.626
h			1.1			0.043
L	0.8		1.1	0.031		0.043
N	10° (max.)					
S	8° (max.)					
T		10			0.394	

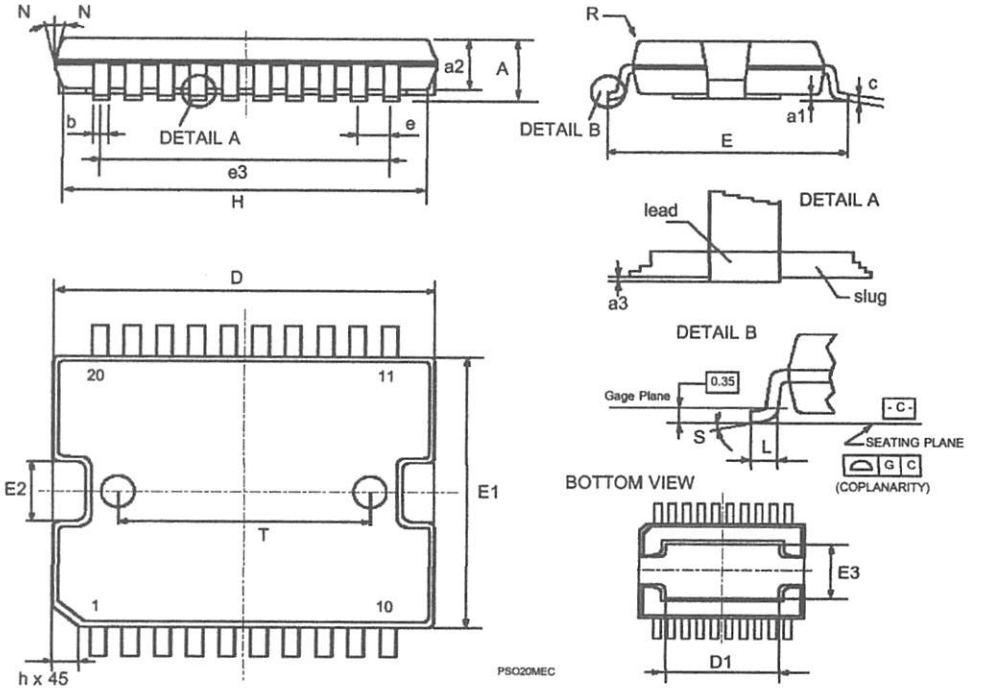
(1) "D and F" do not include mold flash or protrusions.
- Mold flash or protrusions shall not exceed 0.15 mm (0.006").
- Critical dimensions: "E", "G" and "a3"

OUTLINE AND MECHANICAL DATA

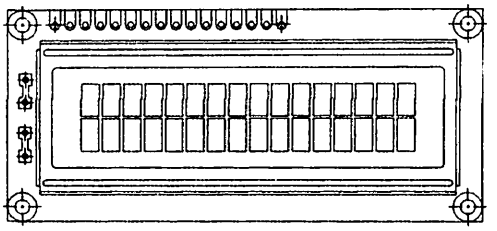


JEDEC MO-166

PowerSO20



16 x 2 Character LCD



FEATURES

- 5 x 8 dots with cursor
- Built-in controller (KS 0066 or Equivalent)
- + 5V power supply (Also available for + 3V)
- 1/16 duty cycle
- B/L to be driven by pin 1, pin 2 or pin 15, pin 16 or A.K (LED)
- N.V. optional for + 3V power supply

MECHANICAL DATA

ITEM	STANDARD VALUE	UNIT
Module Dimension	80.0 x 36.0	mm
Viewing Area	66.0 x 16.0	mm
Dot Size	0.56 x 0.66	mm
Character Size	2.96 x 5.56	mm

ABSOLUTE MAXIMUM RATING

ITEM	SYMBOL	STANDARD VALUE			UNIT
		MIN.	TYP.	MAX.	
Power Supply	VDD-VSS	- 0.3	–	7.0	V
Input Voltage	VI	- 0.3	–	VDD	V

NOTE: VSS = 0 Volt, VDD = 5.0 Volt

ELECTRICAL SPECIFICATIONS

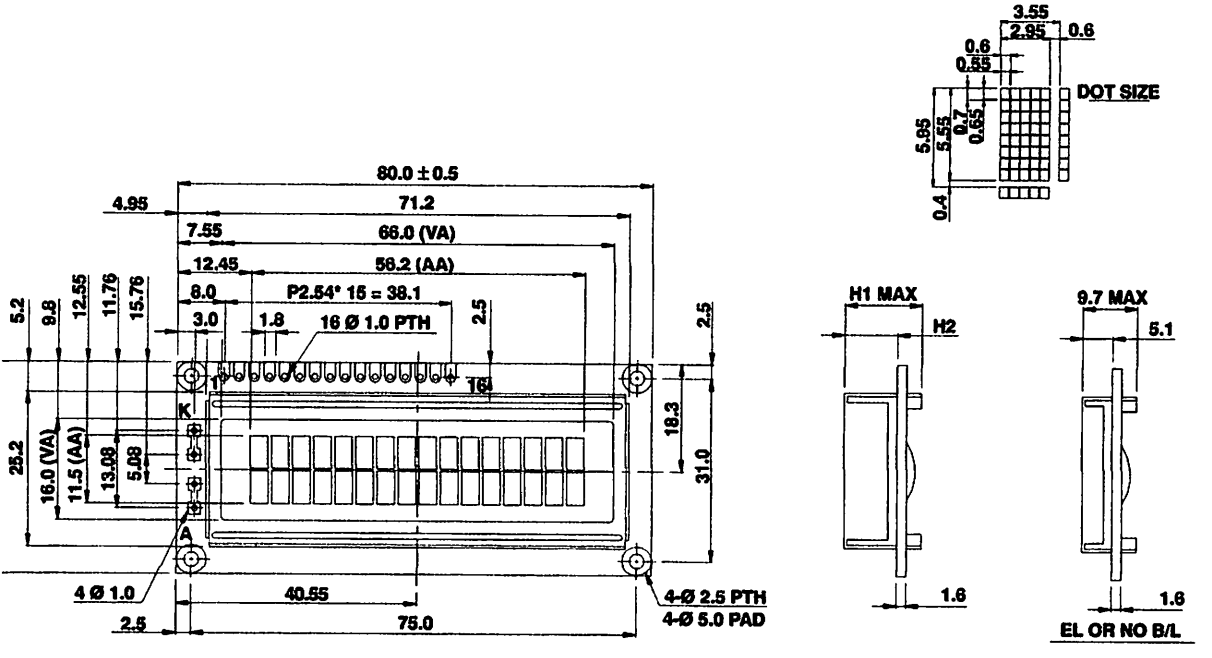
ITEM	SYMBOL	CONDITION		STANDARD VALUE			UNIT
				MIN.	TYP.	MAX.	
Input Voltage	VDD	VDD = + 5V		4.7	5.0	5.3	V
		VDD = + 3V		2.7	3.0	5.3	V
Supply Current	IDD	VDD = 5V		–	1.2	3.0	mA
Recommended LC Driving Voltage for Normal Temp. Version Module	VDD - V0	- 20 °C		–	–	–	V
		0°C		4.2	4.8	5.1	
		25°C		3.8	4.2	4.6	
		50°C		3.6	4.0	4.4	
		70°C		–	–	–	
LED Forward Voltage	VF	25°C		–	4.2	4.6	V
LED Forward Current	IF	25°C	Array	–	130	260	mA
			Edge	–	20	40	
EL Power Supply Current	IEL	Vel = 110VAC:400Hz		–	–	5.0	mA

DISPLAY CHARACTER ADDRESS CODE:

Display Position	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DD RAM Address	00	01														0F
DD RAM Address	40	41														4F

IN NUMBER	SYMBOL	FUNCTION
	Vss	GND
	Vdd	+ 3V or + 5V
	Vo	Contrast Adjustment
	RS	H/L Register Select Signal
	R/$\overline{W}$	H/L Read/Write Signal
	E	H →L Enable Signal
	DB0	H/L Data Bus Line
	DB1	H/L Data Bus Line
	DB2	H/L Data Bus Line
)	DB3	H/L Data Bus Line
	DB4	H/L Data Bus Line
2	DB5	H/L Data Bus Line
3	DB6	H/L Data Bus Line
4	DB7	H/L Data Bus Line
5	A/Vee	+ 4.2V for LED/Negative Voltage Output
6	K	Power Supply for B/L (OV)

IMENSIONS in millimeters



LED - H/L B/L		
	HIGH	LOW
H1	13.2	12.1
H2	8.6	7.5

DATA SHEET

PCF8591
8-bit A/D and D/A converter

Product specification
Supersedes data of 1997 Apr 02
File under Integrated Circuits, IC12

1998 Jul 02

Philips
Semiconductors

PHILIPS

ONTENTS

FEATURES

Single power supply

Operating supply voltage 2.5 V to 6 V

Low standby current

Serial input/output via I^2C-bus

Address by 3 hardware address pins

Sampling rate given by I^2C-bus speed

4 analog inputs programmable as single-ended or differential inputs

Auto-incremented channel selection

Analog voltage range from V_{SS} to V_{DD}

On-chip track and hold circuit

8-bit successive approximation A/D conversion

Multiplying DAC with one analog output.

APPLICATIONS

Closed loop control systems

Low power converter for remote data acquisition

Battery operated equipment

Acquisition of analog values in automotive, audio and TV applications.

3 GENERAL DESCRIPTION

The PCF8591 is a single-chip, single-supply low power 8-bit CMOS data acquisition device with four analog inputs, one analog output and a serial I^2C-bus interface. Three address pins A0, A1 and A2 are used for programming the hardware address, allowing the use of up to eight devices connected to the I^2C-bus without additional hardware. Address, control and data to and from the device are transferred serially via the two-line bidirectional I^2C-bus.

The functions of the device include analog input multiplexing, on-chip track and hold function, 8-bit analog-to-digital conversion and an 8-bit digital-to-analog conversion. The maximum conversion rate is given by the maximum speed of the I^2C-bus.

ORDERING INFORMATION

TYPE NUMBER	PACKAGE		
	NAME	DESCRIPTION	VERSION
CA8591P	DIP16	plastic dual in-line package; 16 leads (300 mil); long body	SOT38-1
CA8591T	SO16	plastic small outline package; 16 leads; body width 7.5 mm	SOT162-1

5 BLOCK DIAGRAM

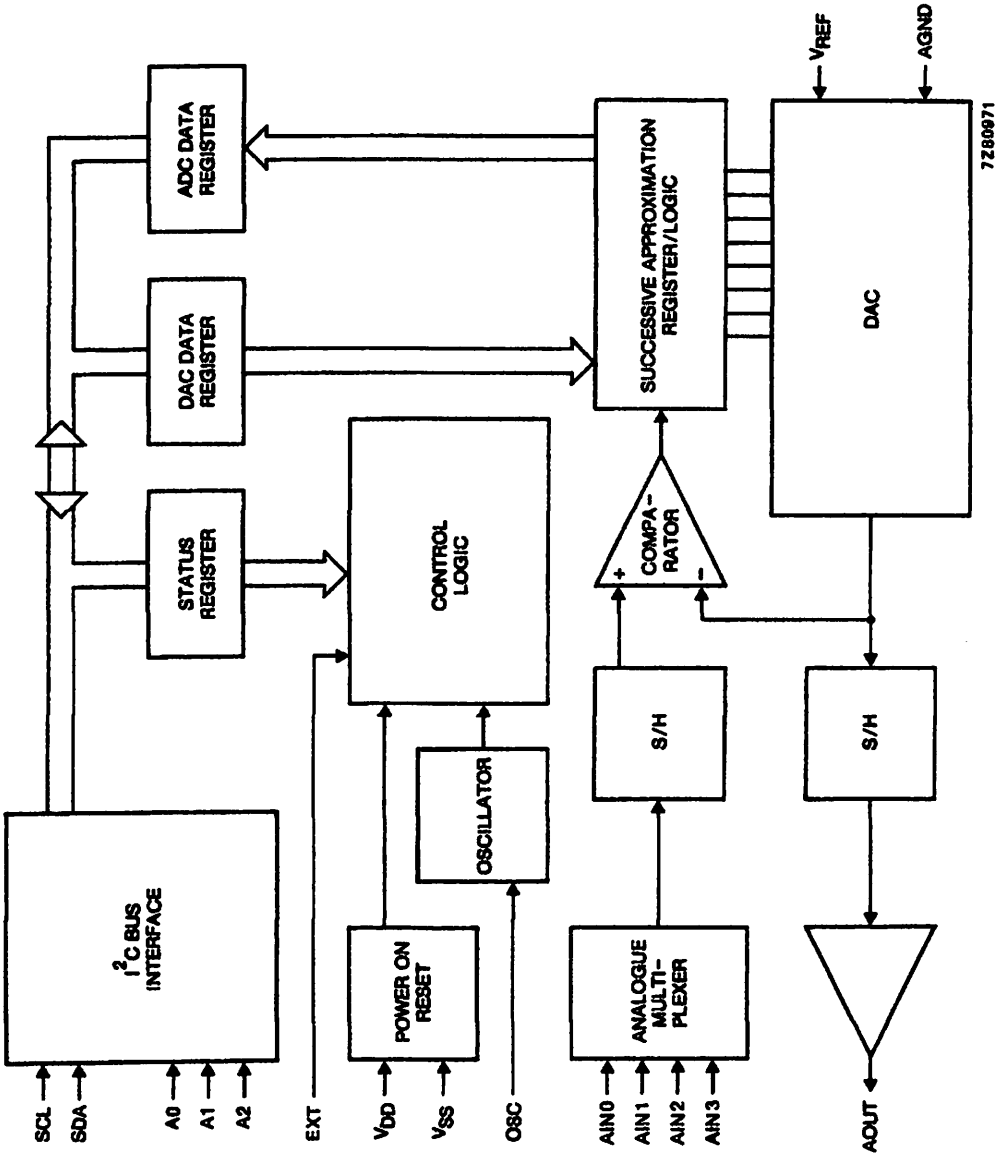


Fig.1 Block diagram.

PINNING

YMBOL	PIN	DESCRIPTION
NO	1	analog inputs (A/D converter)
N1	2	
N2	3	
N3	4	
)	5	hardware address
1	6	
2	7	
SS	8	negative supply voltage
DA	9	I²C-bus data input/output
CL	10	I²C-bus clock input
SC	11	oscillator input/output
KT	12	external/internal switch for oscillator input
GND	13	analog ground
REF	14	voltage reference input
OUT	15	analog output (D/A converter)
DD	16	positive supply voltage

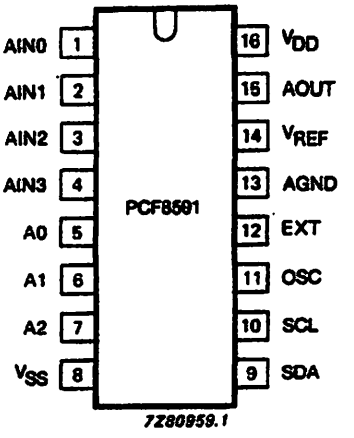


Fig.2 Pinning diagram.

FUNCTIONAL DESCRIPTION

Addressing

ch PCF8591 device in an I²C-bus system is activated by nding a valid address to the device. The address nsists of a fixed part and a programmable part. e programmable part must be set according to the dress pins A0, A1 and A2. The address always has to sent as the first byte after the start condition in the -bus protocol. The last bit of the address byte is the *id/write-bit which sets the direction of the following data* nsfer (see Figs 3, 15 and 16).

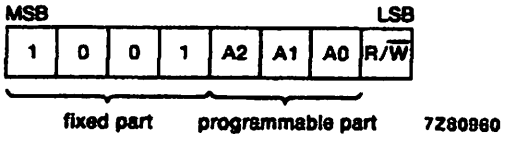


Fig.3 Address byte.

7.2 Control byte

The second byte sent to a PCF8591 device will be stored in its control register and is required to control the device function.

The upper nibble of the control register is used for enabling the analog output, and for programming the analog inputs as single-ended or differential inputs. The lower nibble selects one of the analog input channels defined by the upper nibble (see Fig.4). If the auto-increment flag is set the channel number is incremented automatically after each A/D conversion.

If the auto-increment mode is desired in applications where the internal oscillator is used, the analog output enable flag in the control byte (bit 6) should be set. This allows the internal oscillator to run continuously, thereby preventing conversion errors resulting from oscillator start-up delay. The analog output enable flag may be reset at other times to reduce quiescent power consumption.

The selection of a non-existing input channel results in the highest available channel number being allocated. Therefore, if the auto-increment flag is set, the next selected channel will be always channel 0. The most significant bits of both nibbles are reserved for future functions and have to be set to 0. After a Power-on reset condition all bits of the control register are reset to 0. The D/A converter and the oscillator are disabled for power saving. The analog output is switched to a high-impedance state.

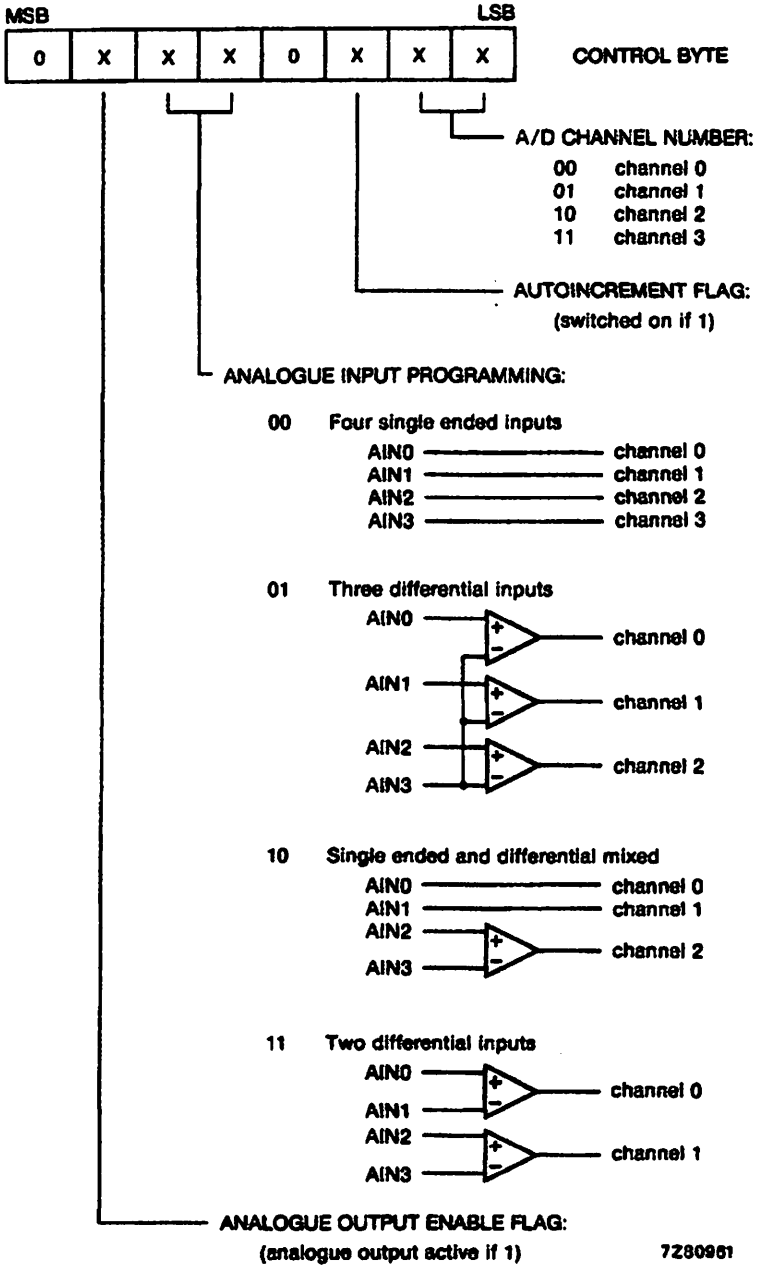


Fig.4 Control byte.

} D/A conversion

e third byte sent to a PCF8591 device is stored in the \C data register and is converted to the corresponding alog voltage using the on-chip D/A converter. This D/A nverter consists of a resistor divider chain connected to } external reference voltage with 256 taps and selection itches. The tap-decoder switches one of these taps to } DAC output line (see Fig.5).

e analog output voltage is buffered by an auto-zeroed ity gain amplifier. This buffer amplifier may be switched or off by setting the analog output enable flag of the ntrol register. In the active state the output voltage is ld until a further data byte is sent.

The on-chip D/A converter is also used for successive approximation A/D conversion. In order to release the DAC for an A/D conversion cycle the unity gain amplifier is equipped with a track and hold circuit. This circuit holds the output voltage while executing the A/D conversion.

The output voltage supplied to the analog output AOUT is given by the formula shown in Fig.6. The waveforms of a D/A conversion sequence are shown in Fig.7.

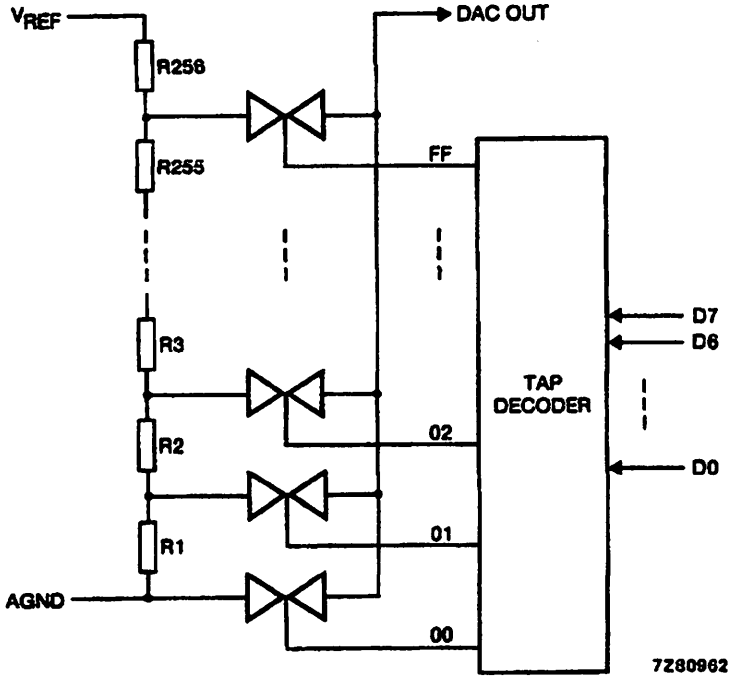


Fig.5 DAC resistor divider chain.

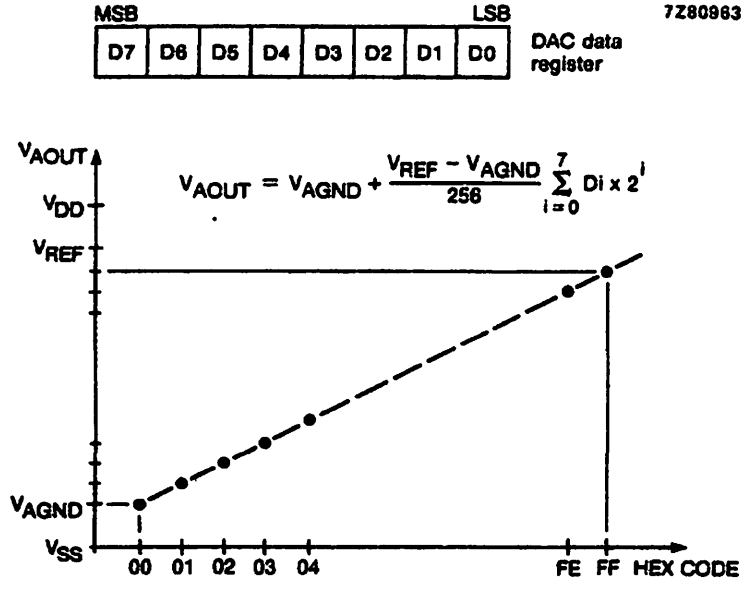


Fig.6 DAC data and DC conversion characteristics.

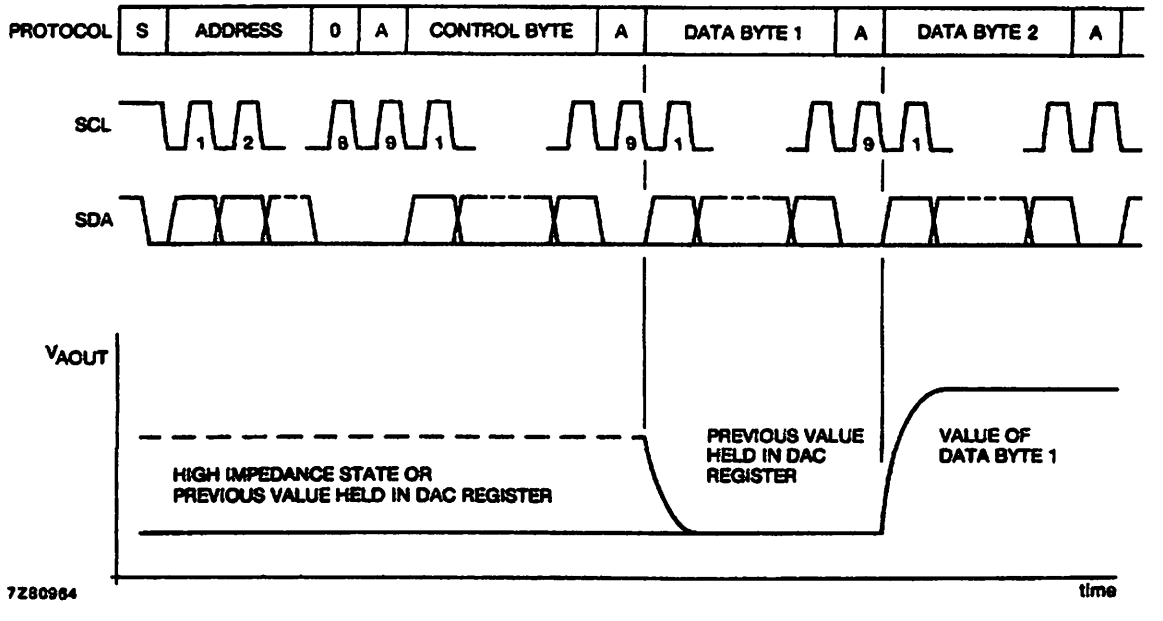


Fig.7 D/A conversion sequence.

A/D conversion

e A/D converter makes use of the successive proximation conversion technique. The on-chip D/A nverter and a high-gain comparator are used nporarily during an A/D conversion cycle.

A/D conversion cycle is always started after sending a lid read mode address to a PCF8591 device. The A/D nversion cycle is triggered at the trailing edge of the knowledge clock pulse and is executed while nsmitting the result of the previous conversion (see g.8).

ice a conversion cycle is triggered an input voltage mple of the selected channel is stored on the chip and is nverted to the corresponding 8-bit binary code. Samples ked up from differential inputs are converted to an 8-bit o's complement code (see Figs 9 and 10).

The conversion result is stored in the ADC data register and awaits transmission. If the auto-increment flag is set the next channel is selected.

The first byte transmitted in a read cycle contains the conversion result code of the previous read cycle. After a Power-on reset condition the first byte read is a hexadecimal 80. The protocol of an I^2C-bus read cycle is shown in Chapter 8, Figs 15 and 16.

The maximum A/D conversion rate is given by the actual speed of the I^2C-bus.

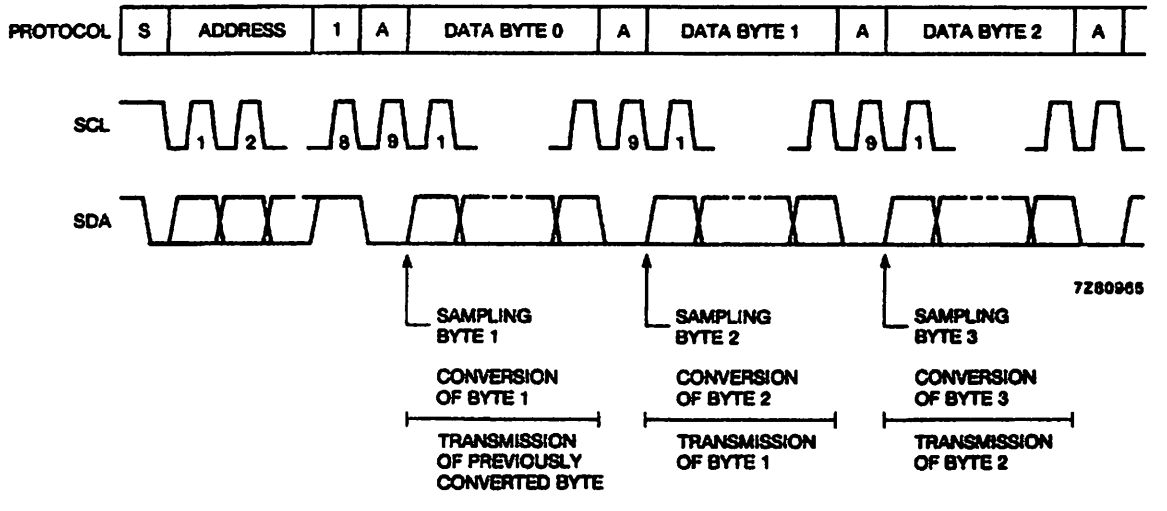


Fig.8 A/D conversion sequence.

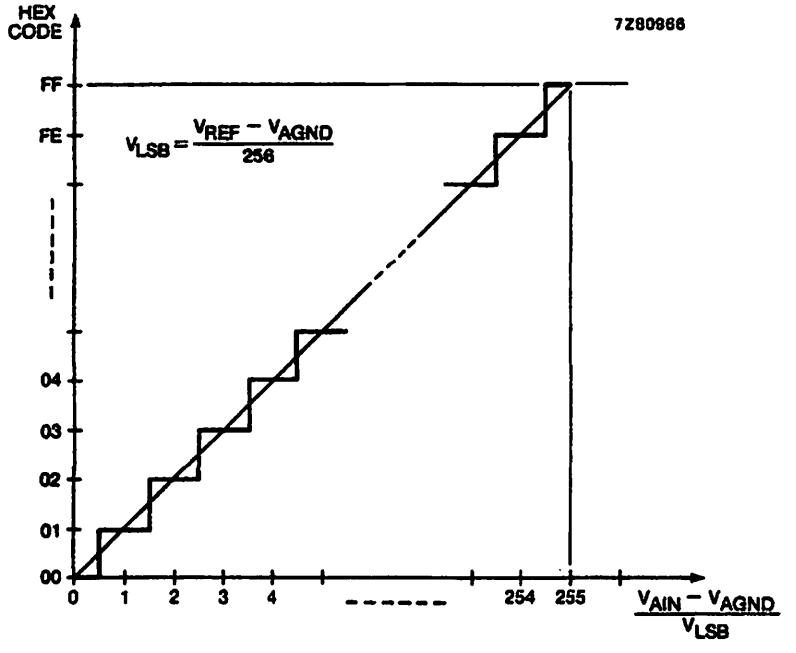


Fig.9 A/D conversion characteristics of single-ended inputs.

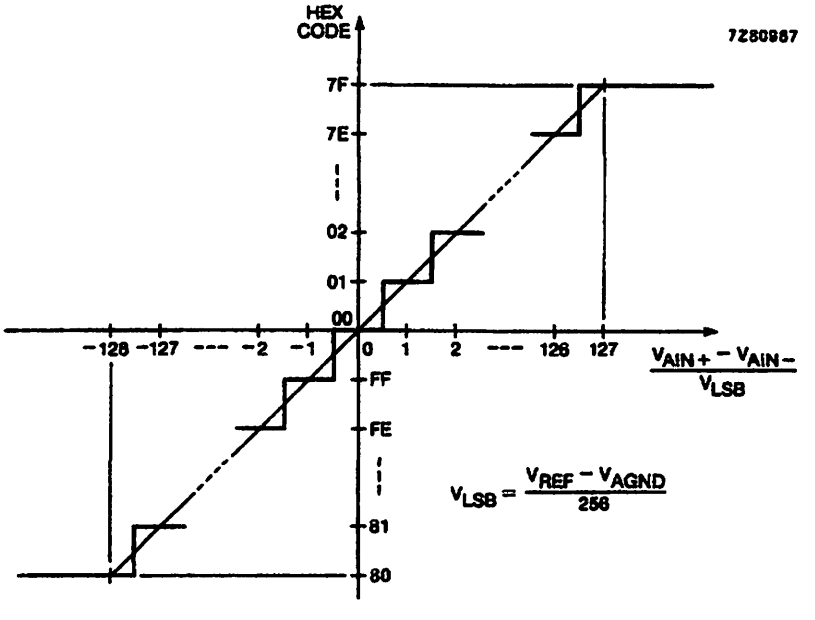


Fig.10 A/D conversion characteristics of differential inputs.

; Reference voltage

r the D/A and A/D conversion either a stable external ltage reference or the supply voltage has to be applied the resistor divider chain (pins V_{REF} and AGND). e AGND pin has to be connected to the system analog)und and may have a DC off-set with reference to V_{SS}.

ow frequency may be applied to the V_{REF} and AGND is. This allows the use of the D/A converter as a e-quadrant multiplier; see Chapter 15 and Fig.6.

e A/D converter may also be used as a one or two adrant analog divider. The analog input voltage is ided by the reference voltage. The result is converted to)inary code. In this application the user has to keep the erence voltage stable during the conversion cycle.

7.6 Oscillator

An on-chip oscillator generates the clock signal required for the A/D conversion cycle and for refreshing the auto-zeroed buffer amplifier. When using this oscillator the EXT pin has to be connected to V_{SS}. At the OSC pin the oscillator frequency is available.

If the EXT pin is connected to V_{DD} the oscillator output OSC is switched to a high-impedance state allowing the user to feed an external clock signal to OSC.

CHARACTERISTICS OF THE I²C-BUS

e I²C-bus is for bidirectional, two-line communication between different ICs or modules. The two lines are a serial data e (SDA) and a serial clock line (SCL). Both lines must be connected to a positive supply via a pull-up resistor. Data nsfer may be initiated only when the bus is not busy.

Bit transfer

e data bit is transferred during each clock pulse. The data on the SDA line must remain stable during the HIGH period he clock pulse as changes in the data line at this time will be interpreted as a control signal.

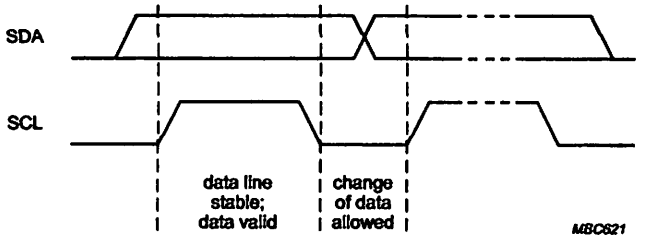


Fig.11 Bit transfer.

Start and stop conditions

th data and clock lines remain HIGH when the bus is not busy. A HIGH-to-LOW transition of the data line, while the ck is HIGH, is defined as the start condition (S). A LOW-to-HIGH transition of the data line while the clock is HIGH, is ined as the stop condition (P).

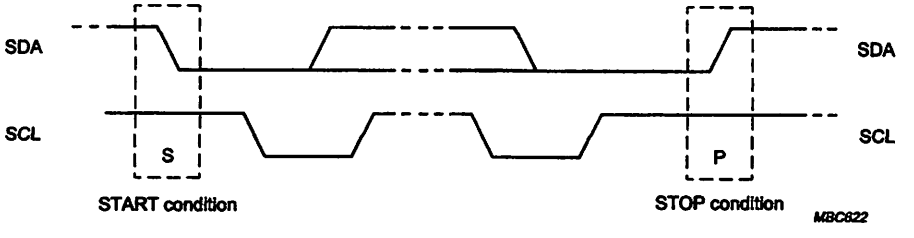


Fig.12 Definition of START and STOP condition.

System configuration

levice generating a message is a 'transmitter', a device receiving a message is the 'receiver'. The device that controls message is the 'master' and the devices which are controlled by the master are the 'slaves'.

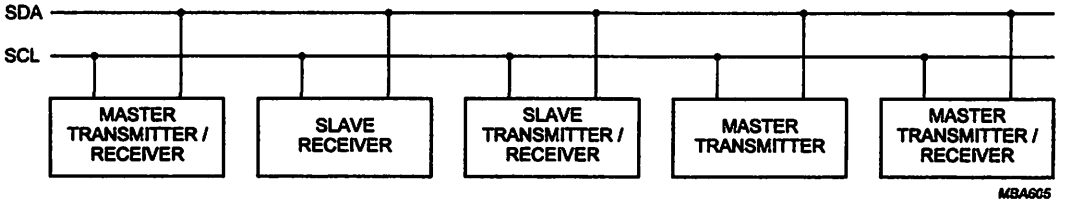


Fig.13 System configuration.

Acknowledge

e number of data bytes transferred between the start and stop conditions from transmitter to receiver is not limited. ch data byte of eight bits is followed by one acknowledge bit. The acknowledge bit is a HIGH level put on the bus by transmitter whereas the master also generates an extra acknowledge related clock pulse. A slave receiver which is dressed must generate an acknowledge after the reception of each byte. Also a master must generate an knowledge after the reception of each byte that has been clocked out of the slave transmitter. The device that knowledges has to pull down the SDA line during the acknowledge clock pulse, so that the SDA line is stable LOW ring the HIGH period of the acknowledge related clock pulse. A master receiver must signal an end of data to the nsmitter by **not** generating an acknowledge on the last byte that has been clocked out of the slave. In this event the nsmitter must leave the data line HIGH to enable the master to generate a stop condition.

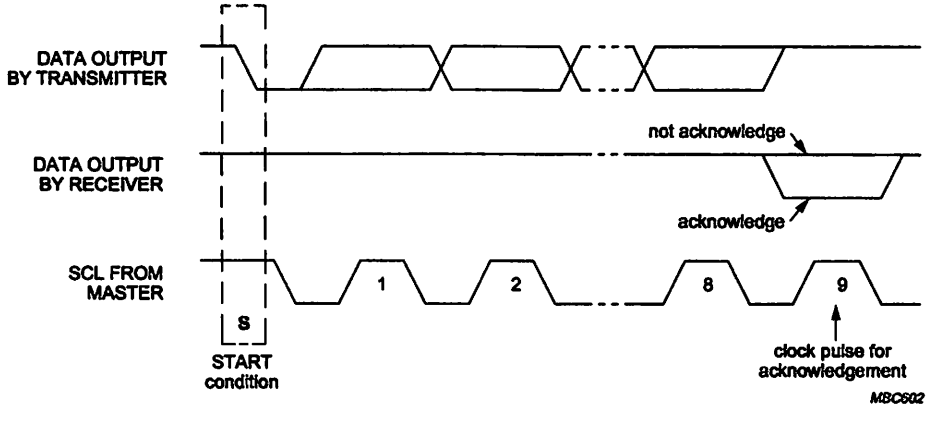


Fig.14 Acknowledgement on the I^2C-bus.

I²C-bus protocol

er a start condition a valid hardware address has to be sent to a PCF8591 device. The read/write bit defines the ection of the following single or multiple byte data transfer. For the format and the timing of the start condition (S), the p condition (P) and the acknowledge bit (A) refer to the I²C-bus characteristics. In the write mode a data transfer is minated by sending either a stop condition or the start condition of the next data transfer.

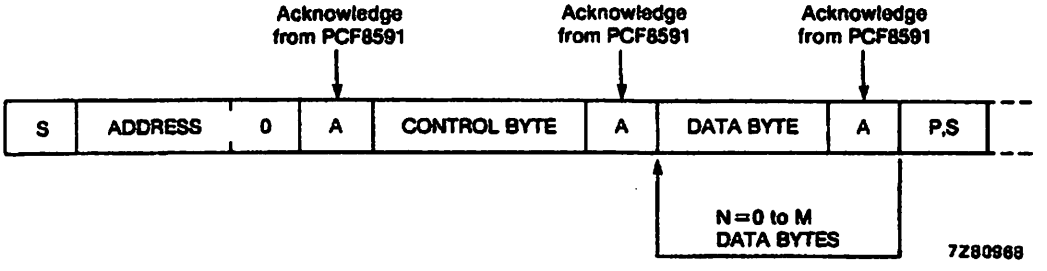


Fig.15 Bus protocol for write mode, D/A conversion.

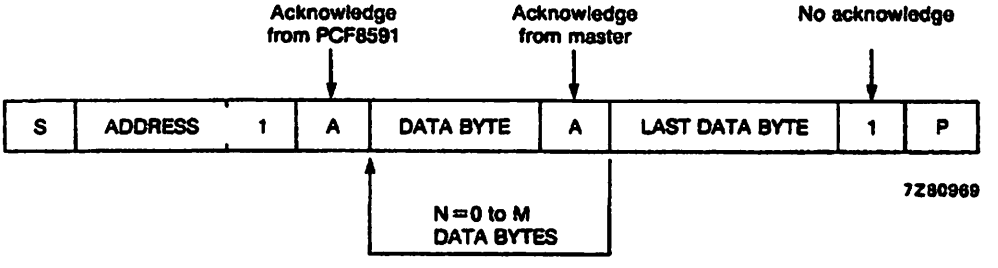


Fig.16 Bus protocol for read mode, A/D conversion.

LIMITING VALUES

accordance with the Absolute Maximum Rating System (IEC 134).

SYMBOL	PARAMETER	MIN.	MAX.	UNIT
DD	supply voltage (pin 16)	–0.5	+8.0	V
	input voltage (any input)	–0.5	V_{DD} + 0.5	V
	DC input current	–	±10	mA
	DC output current	–	±20	mA
D, Iss	V_{DD} or V_{SS} current	–	±50	mA
ot	total power dissipation per package	–	300	mW
D	power dissipation per output	–	100	mW
amb	operating ambient temperature	–40	+85	°C
tg	storage temperature	–65	+150	°C

HANDLING

uts and outputs are protected against electrostatic discharge in normal handling. However, to be totally safe, it is sirable to take precautions appropriate to handling MOS devices. Advice can be found in Data Handbook IC12 under *andling MOS Devices"*.

DC CHARACTERISTICS

$_D$ = 2.5 V to 6 V; V_{SS} = 0 V; T_{amb} = –40 °C to +85 °C unless otherwise specified.

YMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
ıpply						
)D	supply voltage (operating)		2.5	–	6.0	V
D	supply current					
	standby	$V_I = V_{SS}$ or V_{DD}; no load	–	1	15	µA
	operating, AOUT off	f_{SCL} = 100 kHz	–	125	250	µA
	operating, AOUT active	f_{SCL} = 100 kHz	–	0.45	1.0	mA
›OR	Power-on reset level	note 1	0.8	–	2.0	V
gital inputs/output: SCL, SDA, A0, A1, A2						
L	LOW level input voltage		0	–	$0.3 \times V_{DD}$	V
H	HIGH level input voltage		$0.7 \times V_{DD}$	–	V_{DD}	V
	leakage current					
	A0, A1, A2	$V_I = V_{SS}$ to V_{DD}	–250	–	+250	nA
	SCL, SDA	$V_I = V_{SS}$ to V_{DD}	–1	–	+1	µA
	input capacitance		–	–	5	pF
L	LOW level SDA output current	V_{OL} = 0.4 V	3.0	–	–	mA
ıference voltage inputs						
ŒF	reference voltage	$V_{REF} > V_{AGND}$; note 2	V_{SS} + 1.6	–	V_{DD}	V
ıGND	analog ground voltage	$V_{REF} > V_{AGND}$; note 2	V_{SS}	–	V_{DD} – 0.8	V
	input leakage current		–250	–	+250	nA
ŒF	input resistance	pins V_{REF} and AGND	–	100	–	kΩ
ıcillator: OSC, EXT						
	input leakage current		–	–	250	nA
SC	oscillator frequency		0.75	–	1.25	MHz

tes

The power on reset circuit resets the I²C-bus logic when V_{DD} is less than V_{POR}.

A further extension of the range is possible, if the following conditions are fulfilled:

$$\frac{V_{REF}+V_{AGND}}{2} \geq 0.8V, V_{DD} - \frac{V_{REF}+V_{AGND}}{2} \geq 0.4V$$

D/A CHARACTERISTICS

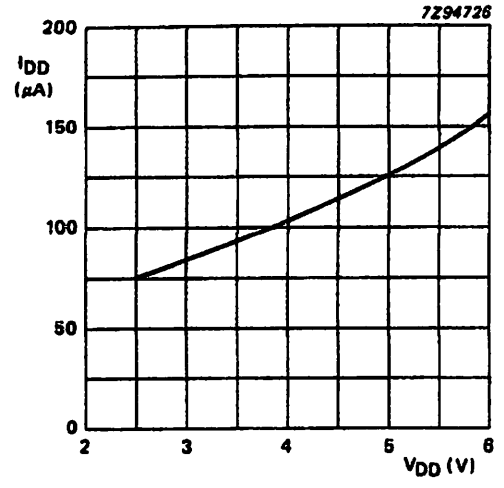
$_D$ = 5.0 V; V_{SS} = 0 V; V_{REF} = 5.0 V; V_{AGND} = 0 V; R_L = 10 kΩ; C_L = 100 pF; T_{amb} = –40 °C to +85 °C unless otherwise ecified.

YMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
nalog output						
OA	output voltage	no resistive load	V_{SS}	–	V_{DD}	V
		R_L = 10 kΩ	V_{SS}	–	0.9 × V_{DD}	V
O	output leakage current	AOUT disabled	–	–	250	nA
ccuracy						
S_e	offset error	T_{amb} = 25 °C	–	–	50	mV
	linearity error		–	–	±1.5	LSB
E	gain error	no resistive load	–	–	1	%
AC	settling time	to ½ LSB full scale step	–	–	90	µs
AC	conversion rate		–	–	11.1	kHz
NRR	supply noise rejection ratio	f = 100 Hz; V_{DDN} = 0.1 × V_{PP}	–	40	–	dB

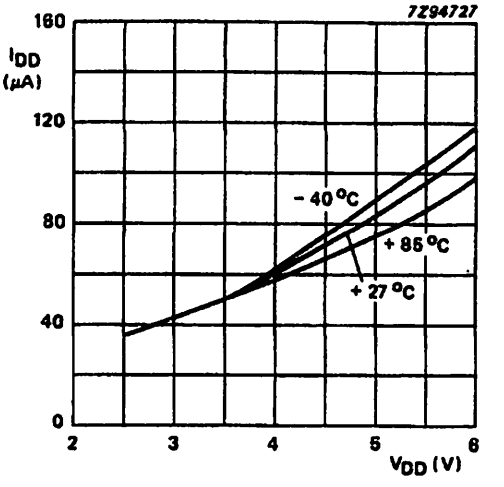
A/D CHARACTERISTICS

$_D$ = 5.0 V; V_{SS} = 0 V; V_{REF} = 5.0 V; V_{AGND} = 0 V; R_S = 10 kΩ; T_{amb} = –40 °C to +85 °C unless otherwise specified.

YMBOL	PARAMETER	CONDITIONS	MIN.	TYP.	MAX.	UNIT
nalog inputs						
A	analog input voltage		V_{SS}	–	V_{DD}	V
A	analog input leakage current		–	–	100	nA
A	analog input capacitance		–	10	–	pF
D	differential input capacitance		–	10	–	pF
S	single-ended voltage	measuring range	V_{AGND}	–	V_{REF}	V
D	differential voltage	measuring range; $V_{FS} = V_{REF} - V_{AGND}$	$\frac{-V_{FS}}{2}$	–	$\frac{+V_{FS}}{2}$	V
ccuracy						
S_e	offset error	T_{amb} = 25 °C	–	–	20	mV
	linearity error		–	–	±1.5	LSB
E	gain error		–	–	1	%
S_e	small-signal gain error	ΔV_i = 16 LSB	–	–	5	%
MRR	common-mode rejection ratio		–	60	–	dB
NRR	supply noise rejection ratio	f = 100 Hz; V_{DDN} = 0.1 × V_{PP}	–	40	–	dB
OC	conversion time		–	–	90	µs
OC	sampling/conversion rate		–	–	11.1	kHz

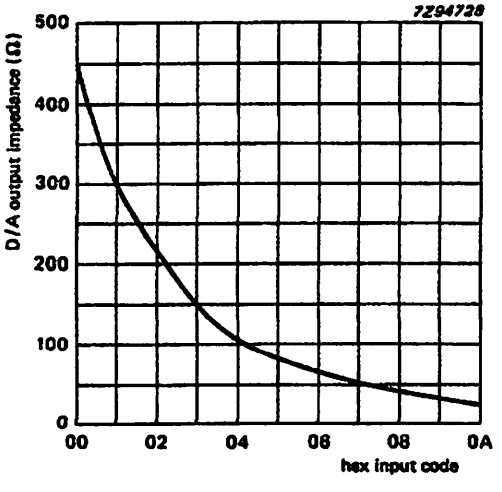


(a) Internal oscillator; T_{amb} = +27 °C.

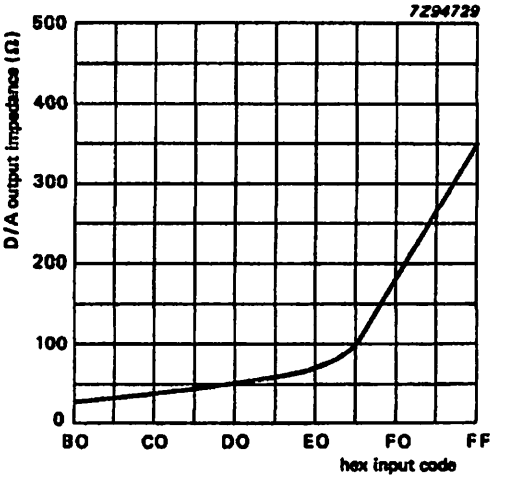


(b) External oscillator.

Fig.17 Operating supply current as a function of supply voltage (analog output disabled).



(a) Output impedance near negative power rail; T_{amb} = +27 °C.



(b) Output impedance near positive power rail; T_{amb} = +27 °C.

The x-axis represents the hex input-code equivalent of the output voltage.

Fig.18 Output impedance of analog output buffer (near power rails).

AC CHARACTERISTICS

timing values are valid within the operating supply voltage and ambient temperature range and reference to V_{IL} and $_I$ with an input voltage swing of V_{SS} to V_{DD}.

SYMBOL	PARAMETER	MIN.	TYP.	MAX.	UNIT
C-bus timing (see Fig.19; note 1)					
CL	SCL clock frequency	–	–	100	kHz
P	tolerable spike width on bus	–	–	100	ns
UF	bus free time	4.7	–	–	µs
U;STA	START condition set-up time	4.7	–	–	µs
D;STA	START condition hold time	4.0	–	–	µs
OW	SCL LOW time	4.7	–	–	µs
IGH	SCL HIGH time	4.0	–	–	µs
	SCL and SDA rise time	–	–	1.0	µs
	SCL and SDA fall time	–	–	0.3	µs
U;DAT	data set-up time	250	–	–	ns
D;DAT	data hold time	0	–	–	ns
D;DAT	SCL LOW-to-data out valid	–	–	3.4	µs
U;STO	STOP condition set-up time	4.0	–	–	µs

te

A detailed description of the I²C-bus specification, with applications, is given in brochure *"The I²C-bus and how to use it"*. This brochure may be ordered using the code 9398 393 40011.

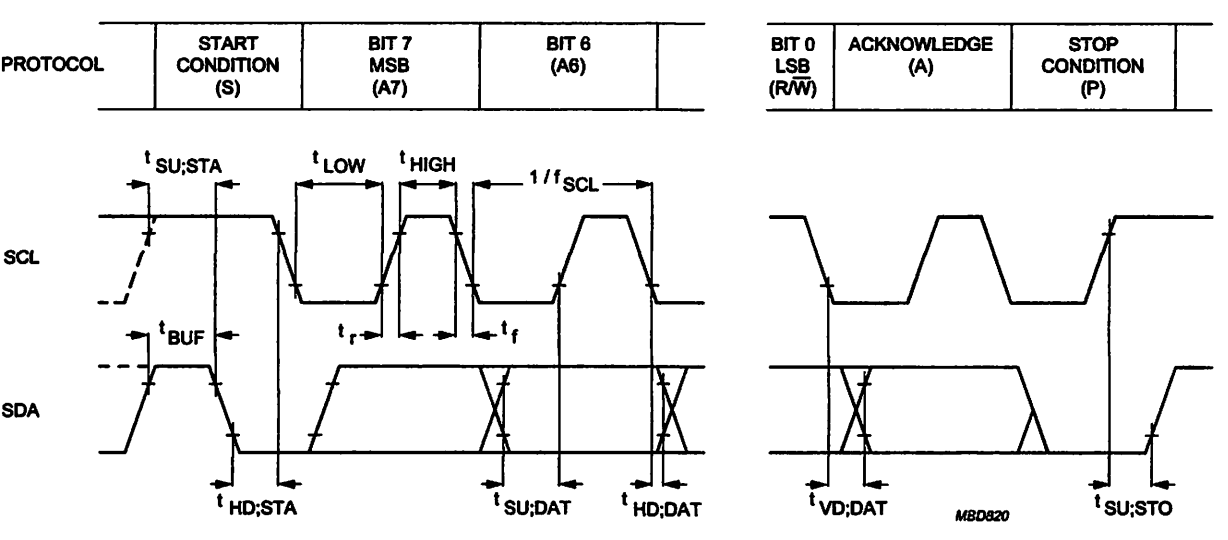


Fig.19 I²C-bus timing diagram; rise and fall times refer to V_{IL} and V_{IH}.

APPLICATION INFORMATION

uts must be connected to V_{SS} or V_{DD} when not in use. Analog inputs may also be connected to AGND or V_{REF}.

order to prevent excessive ground and supply noise and to minimize cross-talk of the digital to analog signal paths the er has to design the printed-circuit board layout very carefully. Supply lines common to a PCF8591 device and noisy ital circuits and ground loops should be avoided. Decoupling capacitors (>10 μF) are recommended for power supply d reference voltage inputs.

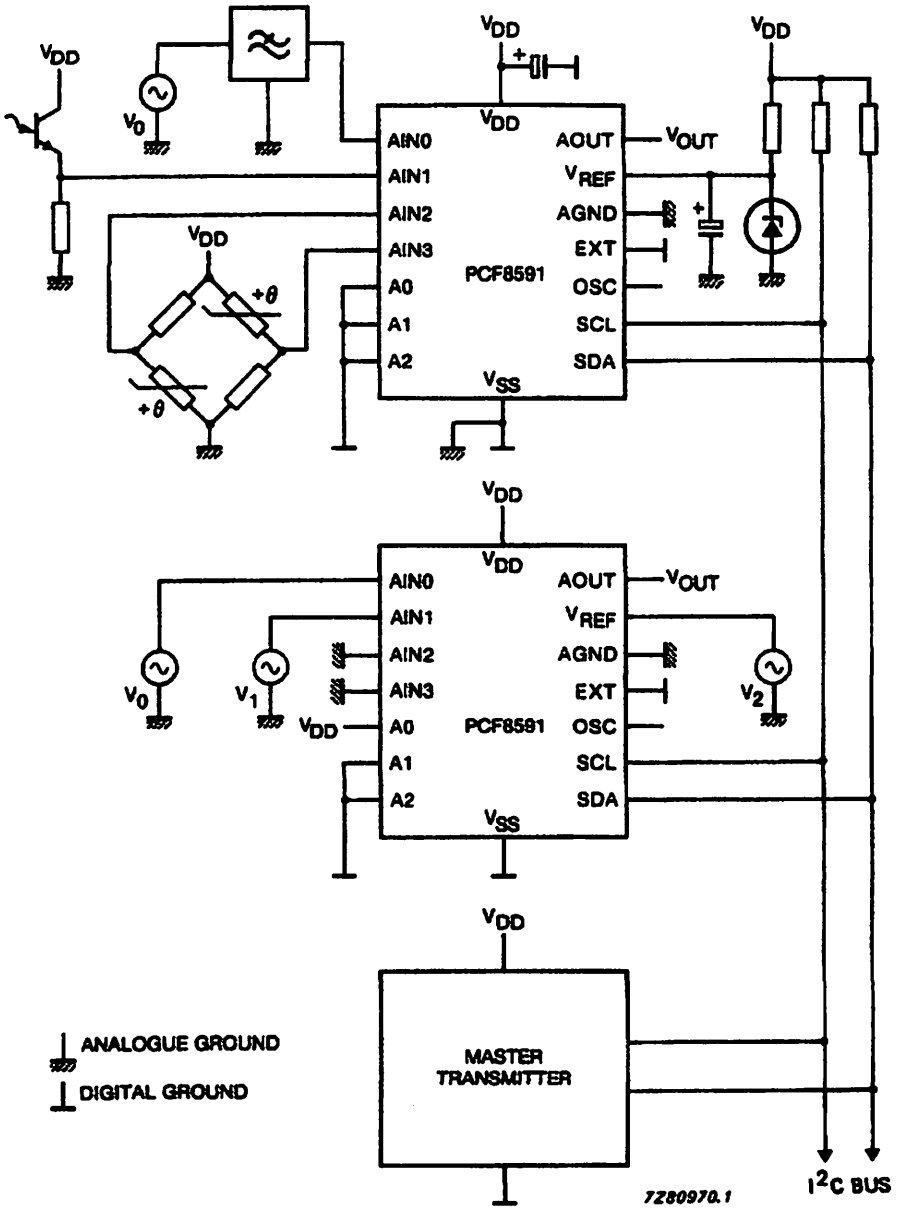
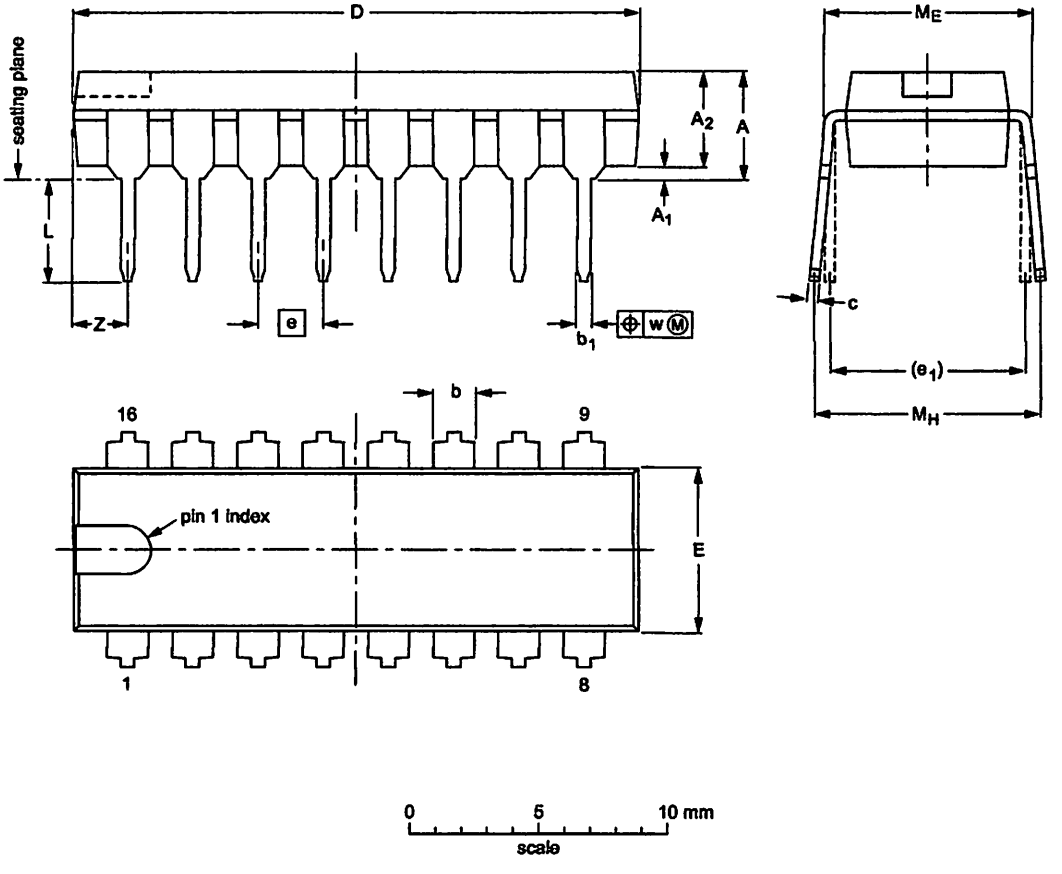


Fig.20 Application diagram.

PACKAGE OUTLINES

P16: plastic dual in-line package; 16 leads (300 mil); long body **SOT38-1**



DIMENSIONS (inch dimensions are derived from the original mm dimensions)

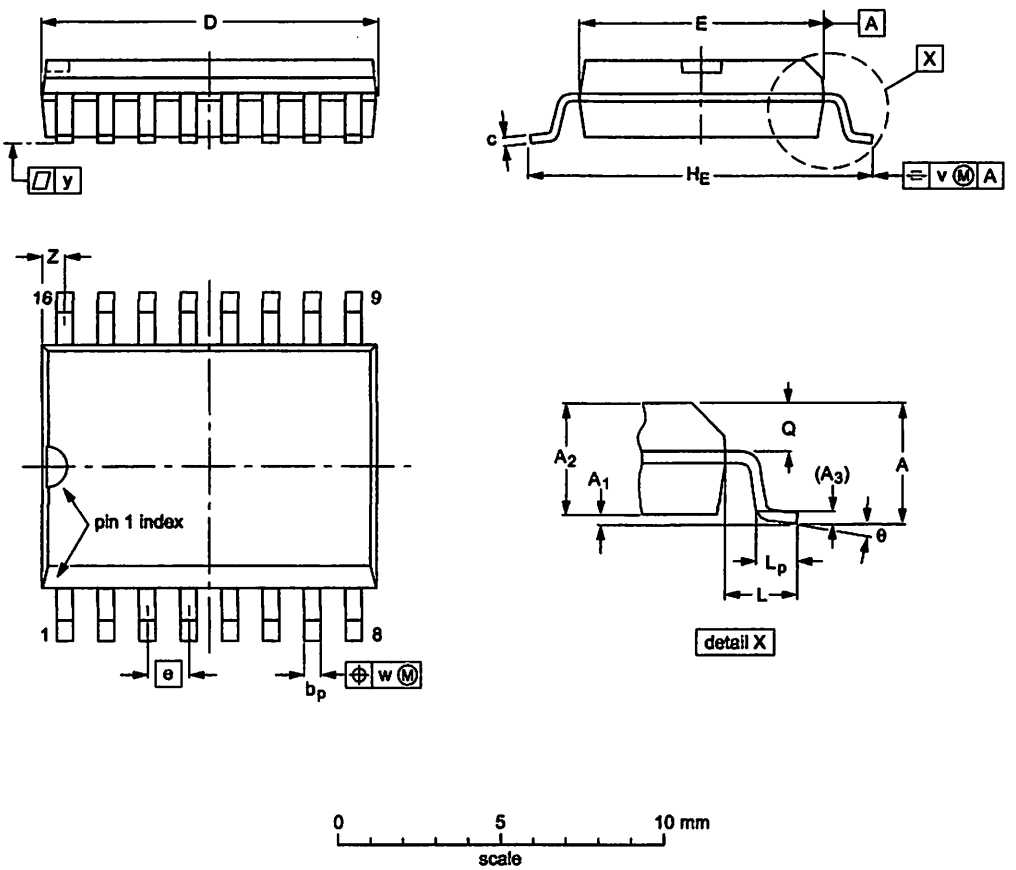
UNIT	A max.	A_1 min.	A_2 max.	b	b_1	c	$D^{(1)}$	$E^{(1)}$	e	e_1	L	M_E	M_H	w	$Z^{(1)}$ max.
mm	4.7	0.51	3.7	1.40 1.14	0.53 0.38	0.32 0.23	21.8 21.4	6.48 6.20	2.54	7.62	3.9 3.4	8.25 7.80	9.5 8.3	0.254	2.2
inches	0.19	0.020	0.15	0.055 0.045	0.021 0.015	0.013 0.009	0.86 0.84	0.26 0.24	0.10	0.30	0.15 0.13	0.32 0.31	0.37 0.33	0.01	0.087

Note

1. Plastic or metal protrusions of 0.25 mm maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT38-1	050G09	MO-001AE				~~92-10-02~~ 95-01-19

)16: plastic small outline package; 16 leads; body width 7.5 mm **SOT162-1**



)IMENSIONS (Inch dimensions are derived from the original mm dimensions)

UNIT	A max.	A_1	A_2	A_3	b_p	c	$D^{(1)}$	$E^{(1)}$	e	H_E	L	L_p	Q	v	w	y	$Z^{(1)}$	θ
mm	2.65	0.30 0.10	2.45 2.25	0.25	0.49 0.36	0.32 0.23	10.5 10.1	7.6 7.4	1.27	10.65 10.00	1.4	1.1 0.4	1.1 1.0	0.25	0.25	0.1	0.9 0.4	8° 0°
inches	0.10	0.012 0.004	0.096 0.089	0.01	0.019 0.014	0.013 0.009	0.41 0.40	0.30 0.29	0.050	0.419 0.394	0.055	0.043 0.016	0.043 0.039	0.01	0.01	0.004	0.035 0.016	

Note

1. Plastic or metal protrusions of 0.15 mm maximum per side are not included.

OUTLINE VERSION	REFERENCES				EUROPEAN PROJECTION	ISSUE DATE
	IEC	JEDEC	EIAJ			
SOT162-1	075E03	MS-013AA				~~95-01-24~~ 97-05-22

SOLDERING

.1 Introduction

ere is no soldering method that is ideal for all IC :kages. Wave soldering is often preferred when ough-hole and surface mounted components are mixed one printed-circuit board. However, wave soldering is : always suitable for surface mounted ICs, or for nted-circuits with high population densities. In these ations reflow soldering is often used.

is text gives a very brief insight to a complex technology. nore in-depth account of soldering ICs can be found in · *"Data Handbook IC26; Integrated Circuit Packages"* der code 9398 652 90011).

.2 DIP

2.1 SOLDERING BY DIPPING OR BY WAVE

e maximum permissible temperature of the solder is) °C; solder at this temperature must not be in contact h the joint for more than 5 seconds. The total contact e of successive solder waves must not exceed econds.

e device may be mounted up to the seating plane, but temperature of the plastic body must not exceed the ecified maximum storage temperature ($T_{stg\ max}$). If the nted-circuit board has been pre-heated, forced cooling y be necessary immediately after soldering to keep the nperature within the permissible limit.

2.2 REPAIRING SOLDERED JOINTS

ply a low voltage soldering iron (less than 24 V) to the d(s) of the package, below the seating plane or not re than 2 mm above it. If the temperature of the dering iron bit is less than 300 °C it may remain in ntact for up to 10 seconds. If the bit temperature is ween 300 and 400 °C, contact may be up to 5 seconds.

3 SO

3.1 REFLOW SOLDERING

flow soldering techniques are suitable for all SO ckages.

flow soldering requires solder paste (a suspension of solder particles, flux and binding agent) to be applied he printed-circuit board by screen printing, stencilling or ssure-syringe dispensing before package placement.

Several techniques exist for reflowing; for example, thermal conduction by heated belt. Dwell times vary between 50 and 300 seconds depending on heating method. Typical reflow temperatures range from 215 to 250 °C.

Preheating is necessary to dry the paste and evaporate the binding agent. Preheating duration: 45 minutes at 45 °C.

17.3.2 WAVE SOLDERING

Wave soldering techniques can be used for all SO packages if the following conditions are observed:

- A double-wave (a turbulent wave with high upward pressure followed by a smooth laminar wave) soldering technique should be used.
- The longitudinal axis of the package footprint must be parallel to the solder flow.
- The package footprint must incorporate solder thieves at the downstream end.

During placement and before soldering, the package must be fixed with a droplet of adhesive. The adhesive can be applied by screen printing, pin transfer or syringe dispensing. The package can be soldered after the adhesive is cured.

Maximum permissible solder temperature is 260 °C, and maximum duration of package immersion in solder is 10 seconds, if cooled to less than 150 °C within 6 seconds. Typical dwell time is 4 seconds at 250 °C.

A mildly-activated flux will eliminate the need for removal of corrosive residues in most applications.

17.3.3 REPAIRING SOLDERED JOINTS

Fix the component by first soldering two diagonally-opposite end leads. Use only a low voltage soldering iron (less than 24 V) applied to the flat part of the lead. Contact time must be limited to 10 seconds at up to 300 °C. When using a dedicated tool, all other leads can be soldered in one operation within 2 to 5 seconds between 270 and 320 °C.

DEFINITIONS

ata sheet status	
bjective specification	This data sheet contains target or goal specifications for product development.
reliminary specification	This data sheet contains preliminary data; supplementary data may be published later.
roduct specification	This data sheet contains final product specifications.

imiting values

miting values given are in accordance with the Absolute Maximum Rating System (IEC 134). Stress above one or ore of the limiting values may cause permanent damage to the device. These are stress ratings only and operation the device at these or at any other conditions above those given in the Characteristics sections of the specification not implied. Exposure to limiting values for extended periods may affect device reliability.

pplication information

here application information is given, it is advisory and does not form part of the specification.

LIFE SUPPORT APPLICATIONS

ese products are not designed for use in life support appliances, devices, or systems where malfunction of these oducts can reasonably be expected to result in personal injury. Philips customers using or selling these products for e in such applications do so at their own risk and agree to fully indemnify Philips for any damages resulting from such proper use or sale.

PURCHASE OF PHILIPS I²C COMPONENTS

Purchase of Philips I²C components conveys a license under the Philips' I²C patent to use the components in the I²C system provided the system conforms to the I²C specification defined by Philips. This specification can be ordered using the code 9398 393 40011.

NOTES

NOTES

Philips Semiconductors – a worldwide company

rgentina: see South America

ustralia: 34 Waterloo Road, NORTH RYDE, NSW 2113,
el. +61 2 9805 4455, Fax. +61 2 9805 4466

ustria: Computerstr. 6, A-1101 WIEN, P.O. Box 213, Tel. +43 160 1010,
ax. +43 160 101 1210

elarus: Hotel Minsk Business Center, Bld. 3, r. 1211, Volodarski Str. 6,
20050 MINSK, Tel. +375 172 200 733, Fax. +375 172 200 773

elgium: see The Netherlands

razil: see South America

ulgaria: Philips Bulgaria Ltd., Energoproject, 15th floor,
1 James Bourchier Blvd., 1407 SOFIA,
el. +359 2 689 211, Fax. +359 2 689 102

anada: PHILIPS SEMICONDUCTORS/COMPONENTS,
el. +1 800 234 7381

hina/Hong Kong: 501 Hong Kong Industrial Technology Centre,
2 Tat Chee Avenue, Kowloon Tong, HONG KONG,
el. +852 2319 7888, Fax. +852 2319 7700

olombia: see South America

zech Republic: see Austria

enmark: Prags Boulevard 80, PB 1919, DK-2300 COPENHAGEN S,
el. +45 32 88 2636, Fax. +45 31 57 0044

inland: Sinikalliontie 3, FIN-02630 ESPOO,
el. +358 9 615800, Fax. +358 9 61580920

rance: 51 Rue Carnot, BP317, 92156 SURESNES Cedex,
el. +33 1 40 99 6161, Fax. +33 1 40 99 6427

ermany: Hammerbrookstraße 69, D-20097 HAMBURG,
el. +49 40 23 53 60, Fax. +49 40 23 536 300

reece: No. 15, 25th March Street, GR 17778 TAVROS/ATHENS,
el. +30 1 4894 339/239, Fax. +30 1 4814 240

ungary: see Austria

ndia: Philips INDIA Ltd, Band Box Building, 2nd floor,
54-D, Dr. Annie Besant Road, Worli, MUMBAI 400 025,
el. +91 22 493 8541, Fax. +91 22 493 0966

ndonesia: *PT Philips Development Corporation, Semiconductors Division,*
edung Philips, Jl. Buncit Raya Kav.99-100, JAKARTA 12510,
el. +62 21 794 0040 ext. 2501, Fax. +62 21 794 0080

eland: Newstead, Clonskeagh, DUBLIN 14,
el. +353 1 7640 000, Fax. +353 1 7640 200

srael: RAPAC Electronics, 7 Kehilat Saloniki St, PO Box 18053,
EL AVIV 61180, Tel. +972 3 645 0444, Fax. +972 3 649 1007

aly: PHILIPS SEMICONDUCTORS, Piazza IV Novembre 3,
0124 MILANO, Tel. +39 2 6752 2531, Fax. +39 2 6752 2557

apan: Philips Bldg 13-37, Kohnan 2-chome, Minato-ku,
OKYO 108-8507, Tel. +81 3 3740 5130, Fax. +81 3 3740 5077

orea: Philips House, 260-199 Itaewon-dong, Yongsan-ku, SEOUL,
el. +82 2 709 1412, Fax. +82 2 709 1415

lalaysia: No. 76 Jalan Universiti, 46200 PETALING JAYA, SELANGOR,
el. +60 3 750 5214, Fax. +60 3 757 4880

lexico: 5900 Gateway East, Suite 200, EL PASO, TEXAS 79905,
el. +9-5 800 234 7381

Middle East: see Italy

Netherlands: Postbus 90050, 5600 PB EINDHOVEN, Bldg. VB,
Tel. +31 40 27 82785, Fax. +31 40 27 88399

New Zealand: 2 Wagener Place, C.P.O. Box 1041, AUCKLAND,
Tel. +64 9 849 4160, Fax. +64 9 849 7811

Norway: Box 1, Manglerud 0612, OSLO,
Tel. +47 22 74 8000, Fax. +47 22 74 8341

Pakistan: see Singapore

Philippines: Philips Semiconductors Philippines Inc.,
106 Valero St. Salcedo Village, P.O. Box 2108 MCC, MAKATI,
Metro MANILA, Tel. +63 2 816 6380, Fax. +63 2 817 3474

Poland: Ul. Lukiska 10, PL 04-123 WARSZAWA,
Tel. +48 22 612 2831, Fax. +48 22 612 2327

Portugal: see Spain

Romania: see Italy

Russia: Philips Russia, Ul. Usatcheva 35A, 119048 MOSCOW,
Tel. +7 095 755 6918, Fax. +7 095 755 6919

Singapore: Lorong 1, Toa Payoh, SINGAPORE 319762,
Tel. +65 350 2538, Fax. +65 251 6500

Slovakia: see Austria

Slovenia: see Italy

South Africa: S.A. PHILIPS Pty Ltd., 195-215 Main Road Martindale,
2092 JOHANNESBURG, P.O. Box 7430 Johannesburg 2000,
Tel. +27 11 470 5911, Fax. +27 11 470 5494

South America: Al. Vicente Pinzon, 173, 6th floor,
04547-130 SÃO PAULO, SP, Brazil,
Tel. +55 11 821 2333, Fax. +55 11 821 2382

Spain: Balmes 22, 08007 BARCELONA,
Tel. +34 93 301 6312, Fax. +34 93 301 4107

Sweden: Kottbygatan 7, Akalla, S-16485 STOCKHOLM,
Tel. +46 8 5985 2000, Fax. +46 8 5985 2745

Switzerland: Allmendstrasse 140, CH-8027 ZÜRICH,
Tel. +41 1 488 2741 Fax. +41 1 488 3263

Taiwan: Philips Semiconductors, 6F, No. 96, Chien Kuo N. Rd., Sec. 1,
TAIPEI, Taiwan Tel. +886 2 2134 2865, Fax. +886 2 2134 2874

Thailand: PHILIPS ELECTRONICS (THAILAND) Ltd.,
209/2 Sanpavuth-Bangna Road Prakanong, BANGKOK 10260,
Tel. +66 2 745 4090, Fax. +66 2 398 0793

Turkey: Talatpasa Cad. No. 5, 80640 GÜLTEPE/ISTANBUL,
Tel. +90 212 279 2770, Fax. +90 212 282 6707

Ukraine: PHILIPS UKRAINE, 4 Patrice Lumumba str., Building B, Floor 7,
252042 KIEV, Tel. +380 44 264 2776, Fax. +380 44 268 0461

United Kingdom: Philips Semiconductors Ltd., 276 Bath Road, Hayes,
MIDDLESEX UB3 5BX, Tel. +44 181 730 5000, Fax. +44 181 754 8421

United States: 811 East Arques Avenue, SUNNYVALE, CA 94088-3409,
Tel. +1 800 234 7381

Uruguay: see South America

Vietnam: see Singapore

Yugoslavia: PHILIPS, Trg N. Pasica 5/v, 11000 BEOGRAD,
Tel. +381 11 625 344, Fax.+381 11 635 777

or all other countries apply to: Philips Semiconductors,
nternational Marketing & Sales Communications, Building BE-p, P.O. Box 218,
600 MD EINDHOVEN, The Netherlands, Fax. +31 40 27 24825

Internet: http://www.semiconductors.philips.com

Printed in The Netherlands 415106/1200/04/pp28 Date of release: 1998 Jul 02 Document order number: 9397 750 04058

Let's make things better.

Philips Semiconductors

PHILIPS

ures

patible with MCS-51® Products
ytes of In-System Programmable (ISP) Flash Memory
Endurance: 1000 Write/Erase Cycles
to 5.5V Operating Range
Static Operation: 0 Hz to 33 MHz
e-level Program Memory Lock
8-bit Internal RAM
ogrammable I/O Lines
e 16-bit Timer/Counters
Interrupt Sources
Duplex UART Serial Channel
power Idle and Power-down Modes
rupt Recovery from Power-down Mode
hdog Timer
Data Pointer
r-off Flag
Programming Time
ble ISP Programming (Byte and Page Mode)

cription

'89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K of in-system programmable Flash memory. The device is manufactured using high-density nonvolatile memory technology and is compatible with the industdard 80C51 instruction set and pinout. The on-chip Flash allows the program y to be reprogrammed in-system or by a conventional nonvolatile memory proer. By combining a versatile 8-bit CPU with in-system programmable Flash on olithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a flexible and cost-effective solution to many embedded control applications.

'89S52 provides the following standard features: 8K bytes of Flash, 256 bytes l, 32 I/O lines, Watchdog timer, two data pointers, three 16-bit timer/counters, a tor two-level interrupt architecture, a full duplex serial port, on-chip oscillator, ck circuitry. In addition, the AT89S52 is designed with static logic for operation o zero frequency and supports two software selectable power saving modes. e Mode stops the CPU while allowing the RAM, timer/counters, serial port, and t system to continue functioning. The Power-down mode saves the RAM conut freezes the oscillator, disabling all other chip functions until the next interrupt ware reset.

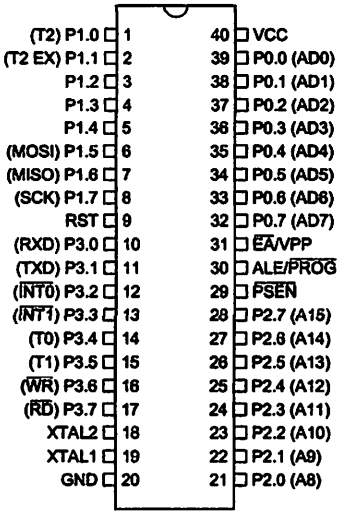
8-bit Microcontroller with 8K Bytes In-System Programmable Flash

AT89S52

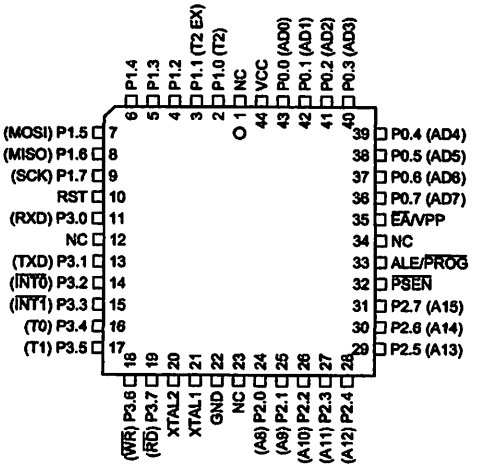
1919B–MICRO–11/03

Configurations

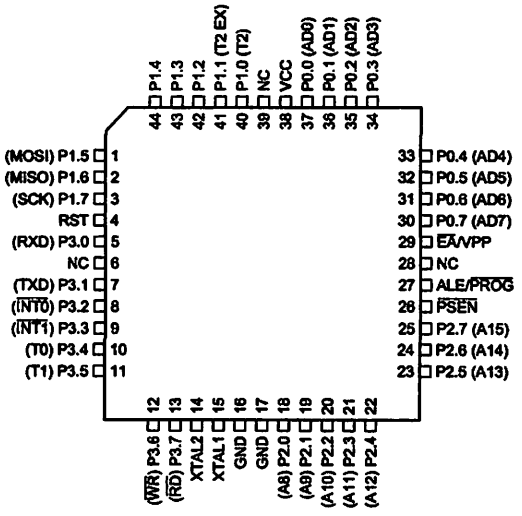
PDIP



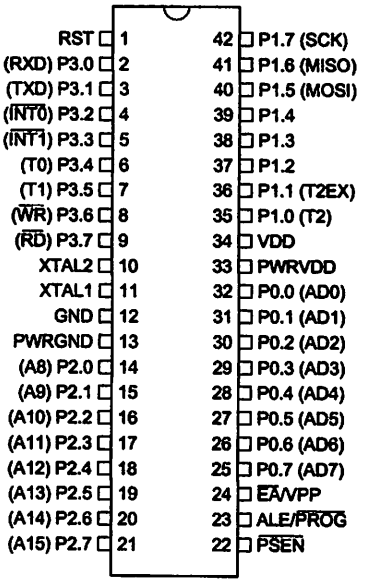
PLCC



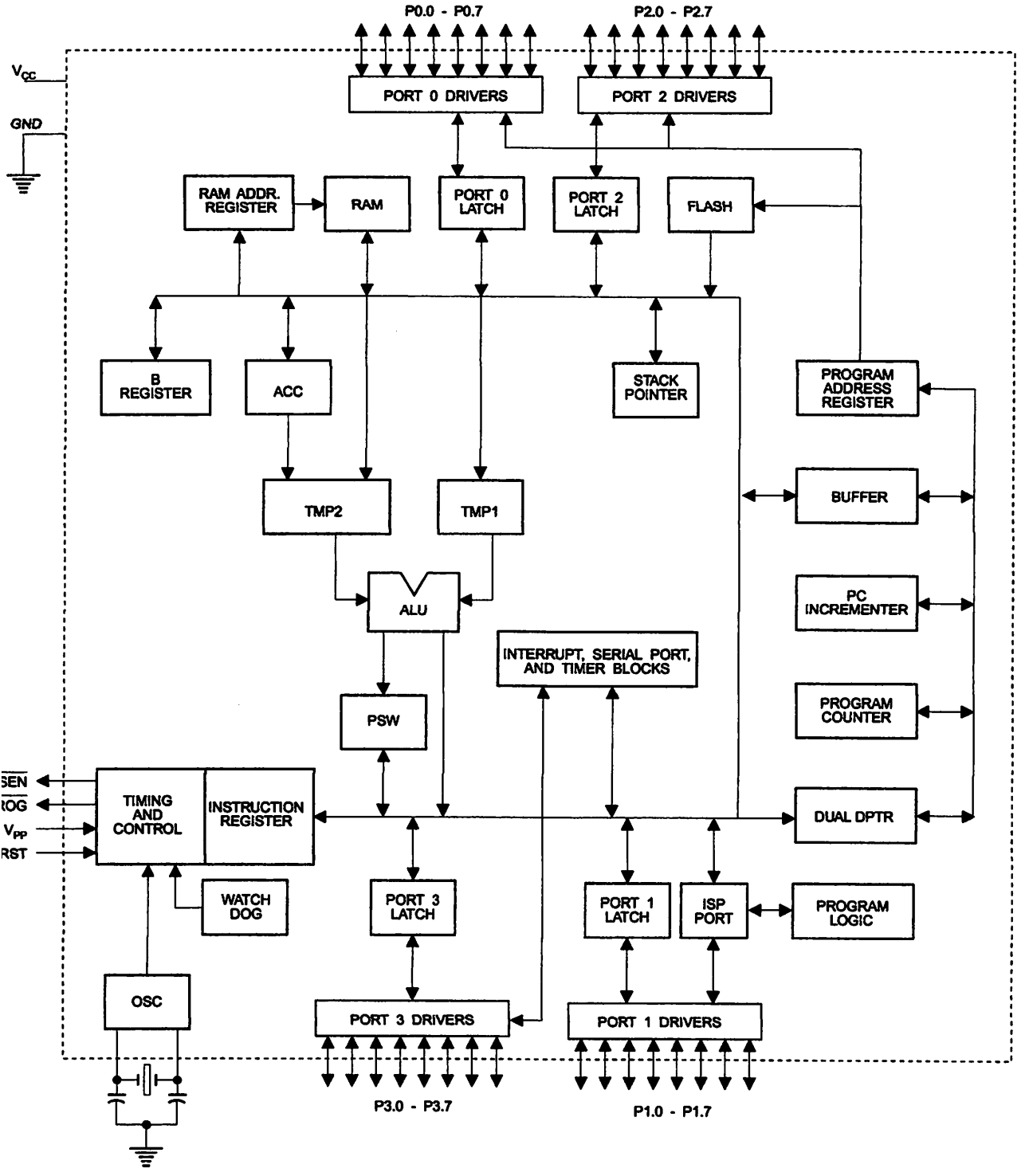
TQFP



PDIP



k Diagram


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

Description

Supply voltage.

Ground.

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

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

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

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

In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table.

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

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

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

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

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

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

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

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

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

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

$\overline{ROG}$

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

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

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

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

When the AT89S52 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory.

P

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

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming.

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

Output from the inverting oscillator amplifier.

cial Function isters

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

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

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

Timer 2 Registers: Control and status bits are contained in registers T2CON (shown in Table 2) and T2MOD (shown in Table 6) 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.

1. AT89S52 SFR Map and Reset Values

								0FFH
B 00000000								0F7H
								0EFH
ACC 00000000								0E7H
								0DFH
PSW 00000000								0D7H
T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000			0CFH
								0C7H
IP XX000000								0BFH
P3 11111111								0B7H
IE 0X000000								0AFH
P2 11111111		AUXR1 XXXXXXX0				WDTRST XXXXXXXX		0A7H
SCON 00000000	SBUF XXXXXXXX							9FH
P1 11111111								97H
TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0		8FH
P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

2. T2CON – Timer/Counter 2 Control Register

2CON Address = 0C8H — Reset Value = 0000 0000B

it Addressable

it

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

ol	Function
	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
	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).
	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.
	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.
2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
	Timer or counter select for Timer 2. C/$\overline{T2}$ = 0 for timer function. C/$\overline{T2}$ = 1 for external event counter (falling edge triggered).
$\overline{2}$	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.

3. AUXR: Auxiliary Register

Address = 8EH — Reset Value = XXX00XX0B

Not Bit Addressable

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

Reserved for future expansion

LE 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

TO Disable/Enable Reset out

DISRTO	
0	Reset pin is driven High after WDT times out
1	Reset pin is input only

LE Disable/Enable WDT in IDLE mode

WDIDLE	
0	WDT continues to count in IDLE mode
1	WDT halts counting in IDLE mode

)ata Pointer Registers: To facilitate accessing both internal and external data memory, two banks of 16-bit Data r Registers are provided: DP0 at SFR address locations 82H-83H and DP1 at 84H-85H. Bit DPS = 0 in SFR AUXR1 ; DP0 and DPS = 1 selects DP1. The user should **ALWAYS** initialize the DPS bit to the appropriate value before iing the respective Data Pointer Register.

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 an be set and rest under software control and is not affected by reset.

4. AUXR1: Auxiliary Register 1

:1 Address = A2H — Reset Value = XXXXXXX0B

Not Bit Addressable

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

Reserved for future expansion

Data Pointer Register Select

DPS	
0	Selects DPTR Registers DP0L, DP0H
1	Selects DPTR Registers DP1L, DP1H

ıory Organization

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

ram Memory

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

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

Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

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

:hdog Timer
)-time Enabled
Reset-out)

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

g the WDT

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

During Power-n and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S52 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

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

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the 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.

T

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".

ır 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".

ır 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 5. 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 5. 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)

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

ıre Mode

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

reload (Up or Down ter)

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 6). 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.

1. Timer in Capture Mode

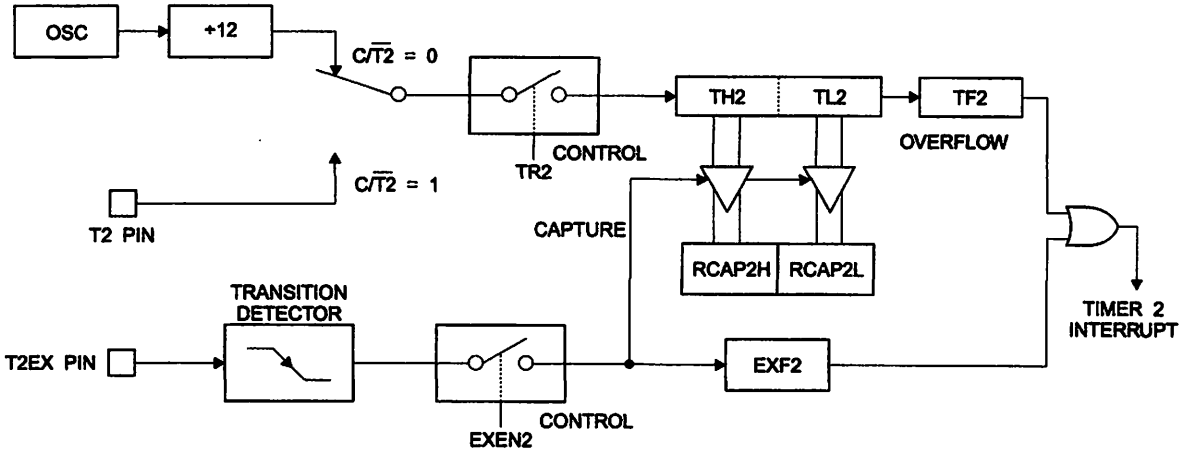


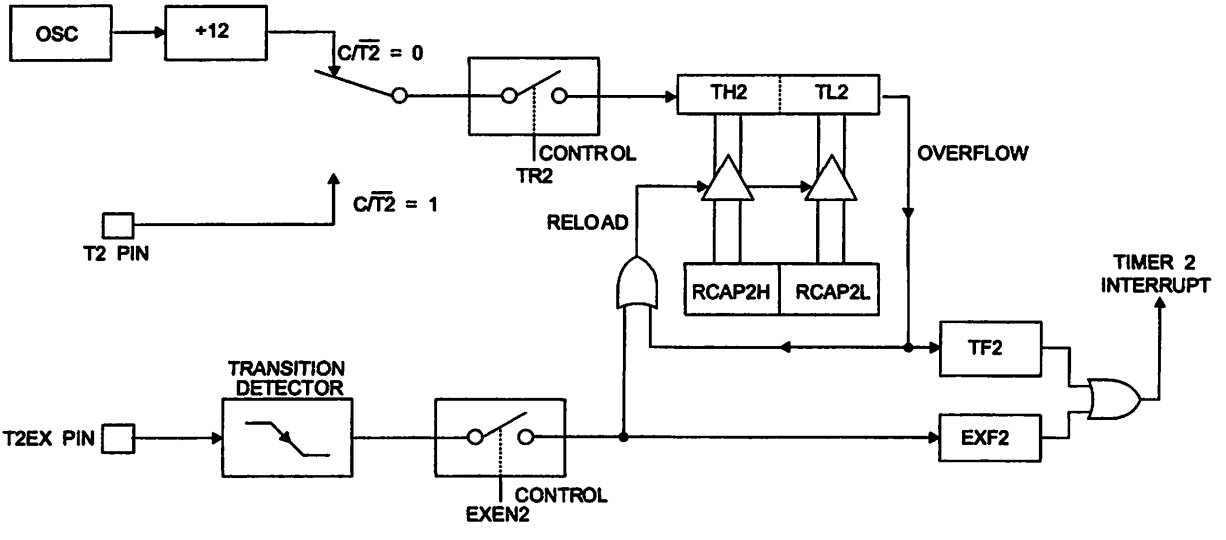
Figure 2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H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

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

2. *Timer 2 Auto Reload Mode (DCEN = 0)*



;. T2MOD – Timer 2 Mode Control Register

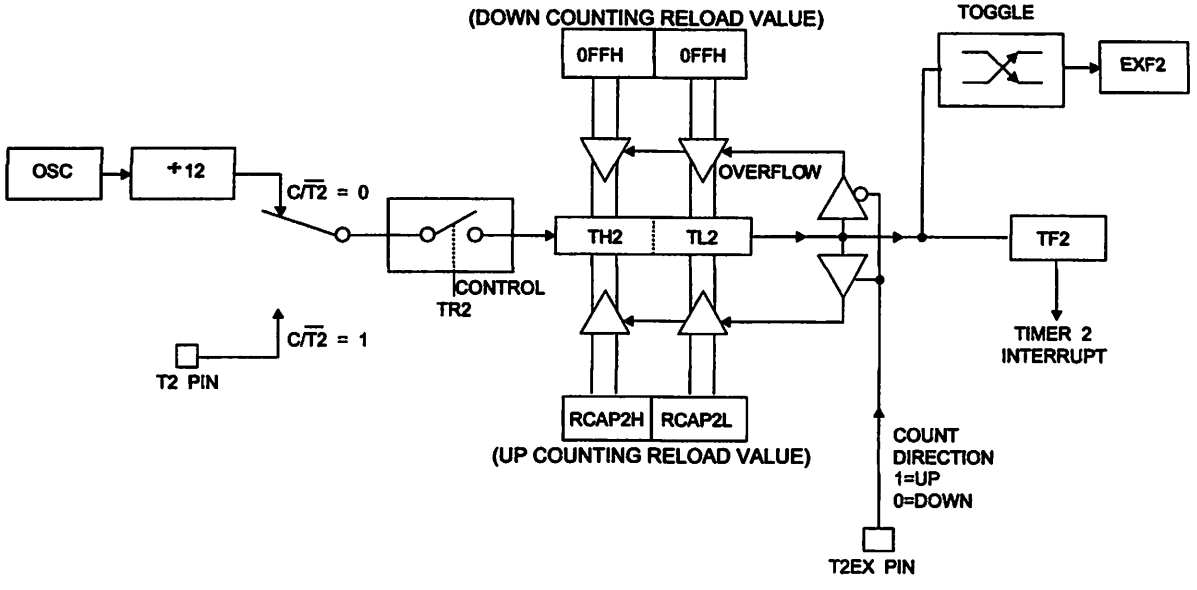
?MOD Address = 0C9H Reset Value = XXXX XX00B

;t Bit Addressable

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

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

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



d Rate Generator

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

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

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

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

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

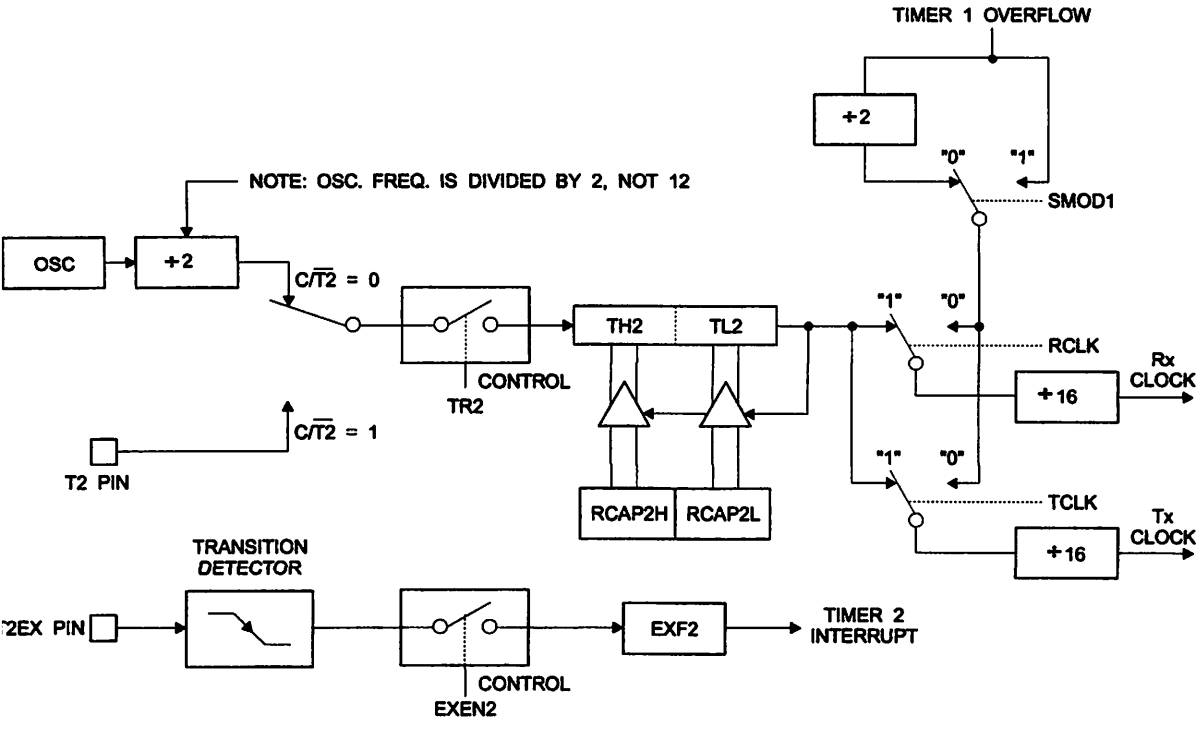
$$\frac{\text{Modes 1 and 3}}{\text{Baud Rate}} = \frac{\text{Oscillator Frequency}}{32 \times [65536\text{-RCAP2H,RCAP2L})]}$$

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

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

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

4. Timer 2 in Baud Rate Generator Mode



ammable
Out

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

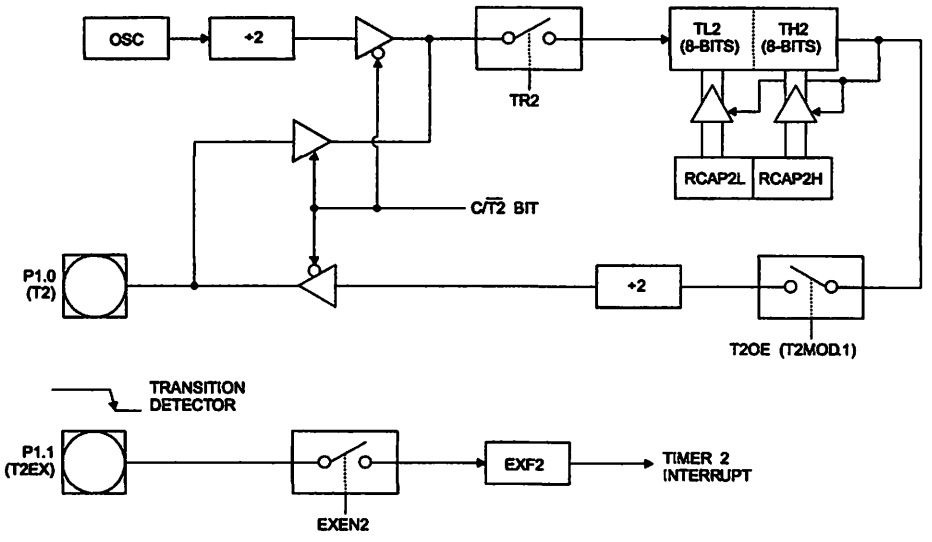
To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

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

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

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

5. Timer 2 in Clock-Out Mode



rupts

The AT89S52 has a total of six interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 6.

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

Note that Table 5 shows that bit position IE.6 is unimplemented. User software should not write a 1 to this bit position, since it may be used in future AT89 products.

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

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

7. Interrupt Enable (IE) Register

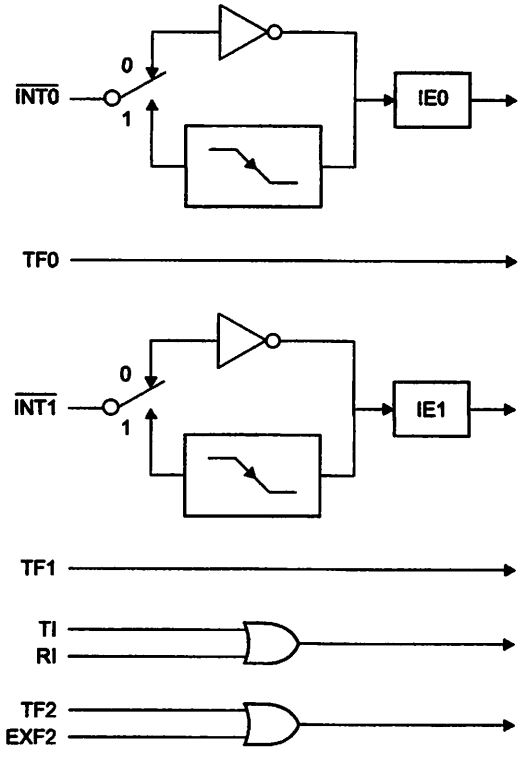
(MSB)							(LSB)
EA	–	ET2	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

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

6. Interrupt Sources



llator acteristics

XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 7. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 8. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Mode

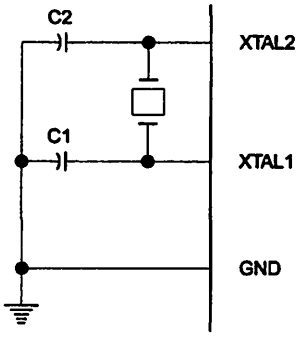
In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware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

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

ɔr-down Mode

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

Figure 7. Oscillator Connections



Note: 1. C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 8. External Clock Drive Configuration

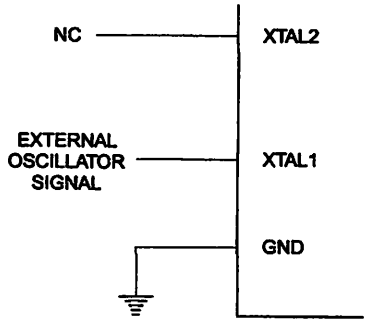


Table 8. Status of External Pins During Idle and Power-down Modes

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

ram Memory Bits

The AT89S52 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in the following table.

Table 9. Lock Bit Protection Modes

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

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

ramming the
ı – Parallel Mode

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

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

Programming Algorithm: Before programming the AT89S52, the address, data, and control signals should be set up according to the Flash programming mode table and Figures 13 and 14. To program the AT89S52, take the following steps:

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

$\overline{Data}$ Polling: The 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 AT89S52
(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.

ramming the
ı – Serial Mode

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

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

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

Programming
ithm

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

1. Power-up sequence:

   Apply power between VCC and GND pins.

   Set RST pin to "H".

   If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.
2. Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less than the CPU clock at XTAL1 divided by 16.
3. The Code array is programmed one byte at a time in either the Byte or Page mode. The write cycle is self-timed and typically takes less than 0.5 ms at 5V.
4. Any memory location can be verified by using the Read instruction 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.

Programming ction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 11.

amming ace – Parallel

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

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

. Flash Programming Modes

	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
de Data	5V	H	L	(2)	12V	L	H	H	H	H	D_{IN}	A12-8	A7-0
de Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A12-8	A7-0
k Bit 1	5V	H	L	(3)	12V	H	H	H	H	H	X	X	X
k Bit 2	5V	H	L	(3)	12V	H	H	H	L	L	X	X	X
k Bit 3	5V	H	L	(3)	12V	H	L	H	H	L	X	X	X
k Bits	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
e	5V	H	L	(1)	12V	H	L	H	L	L	X	X	X
el ID	5V	H	L	H	H	L	L	L	L	L	1EH	X 0000	00H
ice ID	5V	H	L	H	H	L	L	L	L	L	52H	X 0001	00H
ice ID	5V	H	L	H	H	L	L	L	L	L	06H	X 0010	00H

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

Figure 9. Programming the Flash Memory (Parallel Mode)

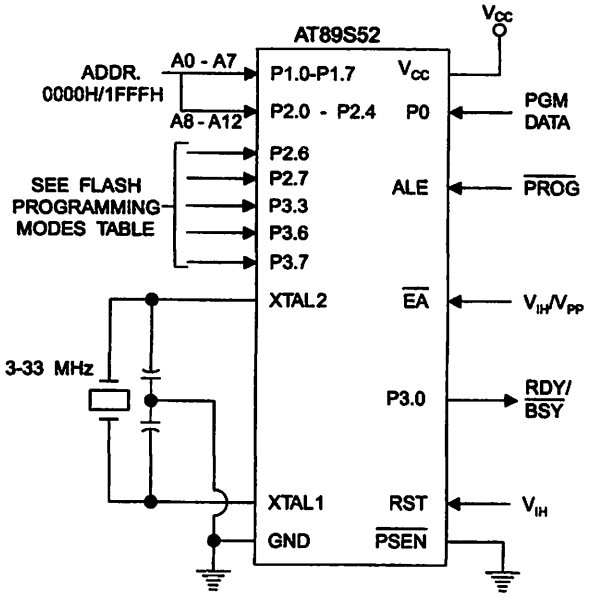
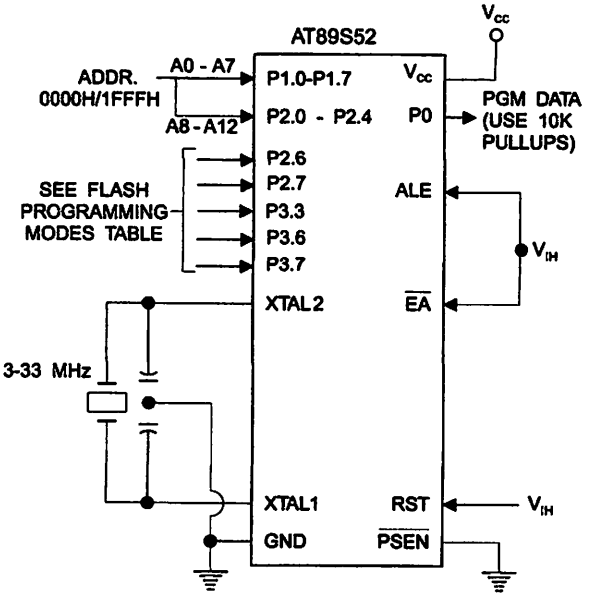


Figure 10. Verifying the Flash Memory (Parallel Mode)



ı Programming and Verification Characteristics (Parallel Mode)

°C to 30°C, V_{CC} = 4.5 to 5.5V

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

11. Flash Programming and Verification Waveforms – Parallel Mode

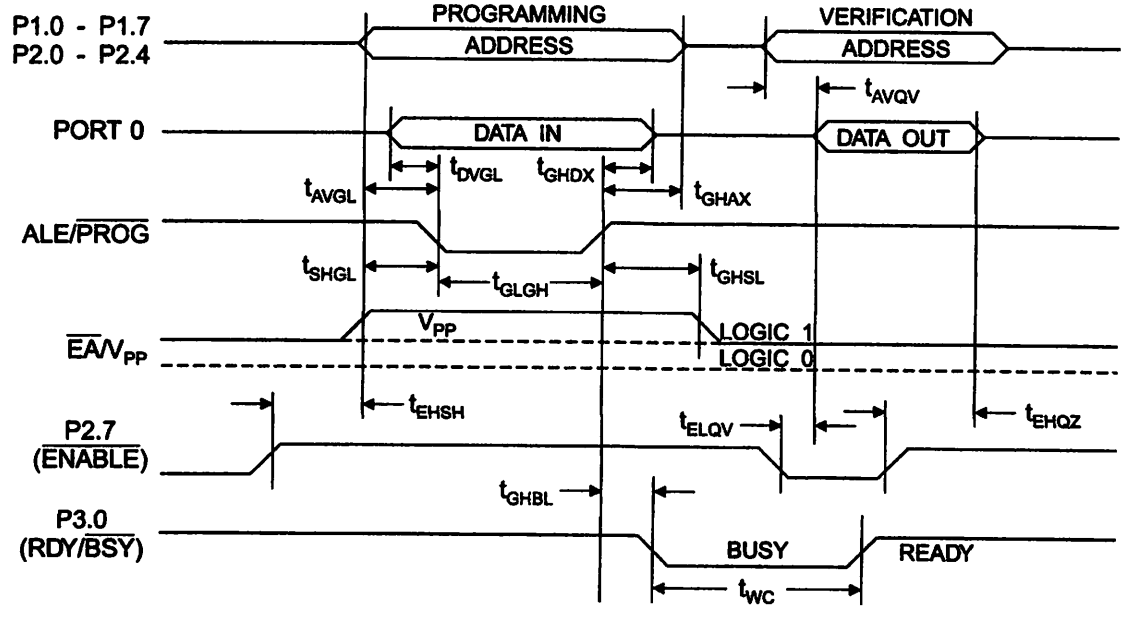
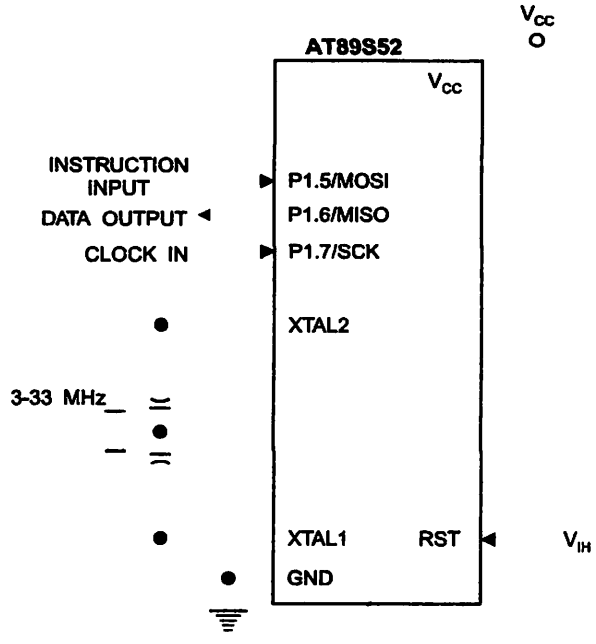
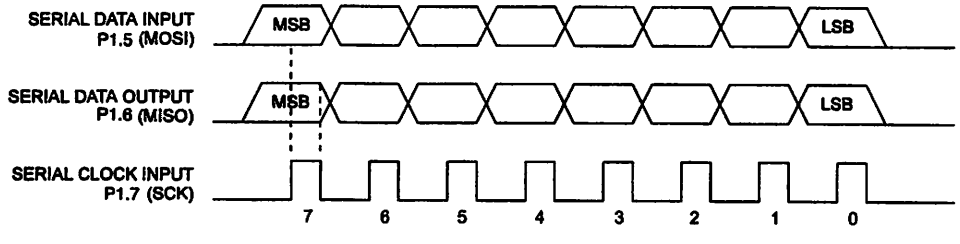


Figure 12. Flash Memory Serial Downloading



Programming
erification
forms – Serial

Figure 13. Serial Programming Waveforms



1. Serial Programming Instruction Set

ction	Instruction Format Byte 1	Byte 2	Byte 3	Byte 4	Operation
mming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output on MISO)	Enable Serial Programming while RST is high
rase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
Program Memory 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
Program Memory 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
ock Bits[1]	1010 1100	1110 00 B1 B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (1).
ock Bits	0010 0100	xxxx xxxx	xxxx xxxx	xxx LB3 LB2 LB1 xx	Read back current status of the lock bits (a programmed lock bit reads back as a "1")
ignature Bytes	0010 1000	xxx A12 A11 A10 A9 A8	A7 xxx xxx0	Signature Byte	Read Signature Byte
Program Memory 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)
Program Memory 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)

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

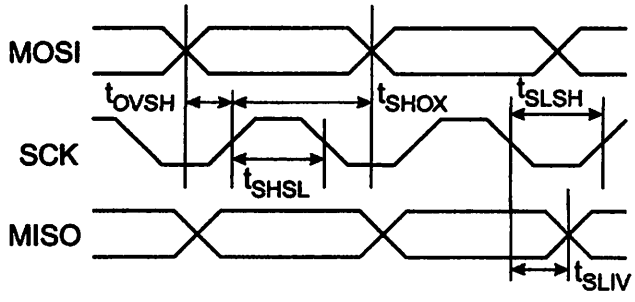
   } Each of the lock bit modes needs to be activated sequentially before Mode 4 can be executed.

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

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

Programming Characteristics

14. Serial Programming Timing



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

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

lute Maximum Ratings*

ng Temperature	-55°C to +125°C
ı Temperature	-65°C to +150°C
en Any Pin spect to Ground	-1.0V to +7.0V
ım Operating Voltage	6.6V
put 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.

haracteristics

ıes 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.

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

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

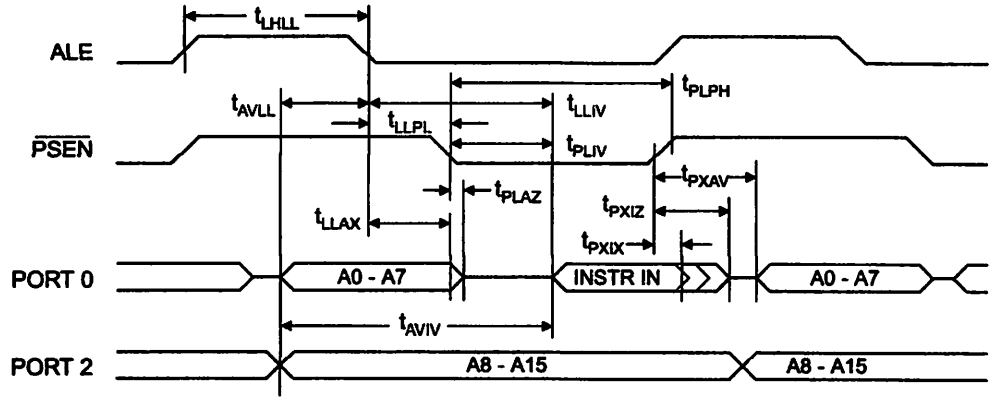
haracteristics

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

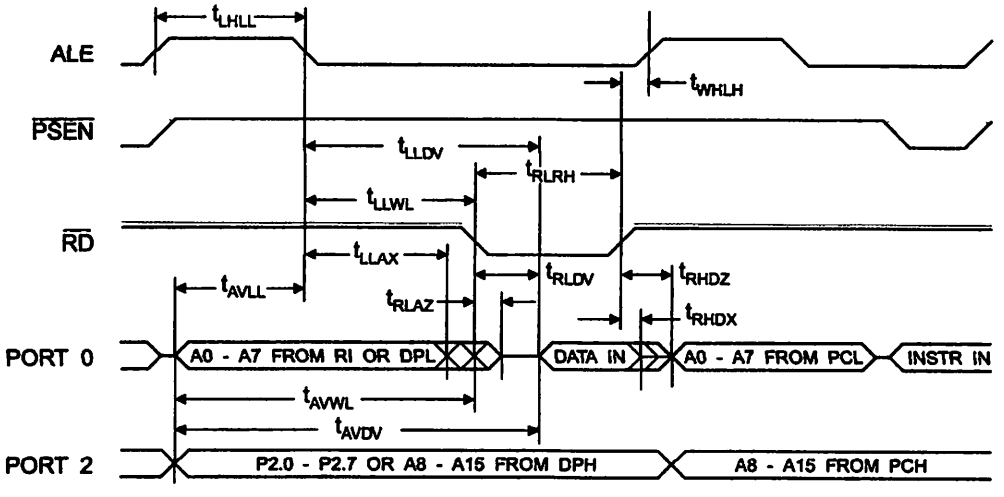
nal Program and Data Memory Characteristics

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

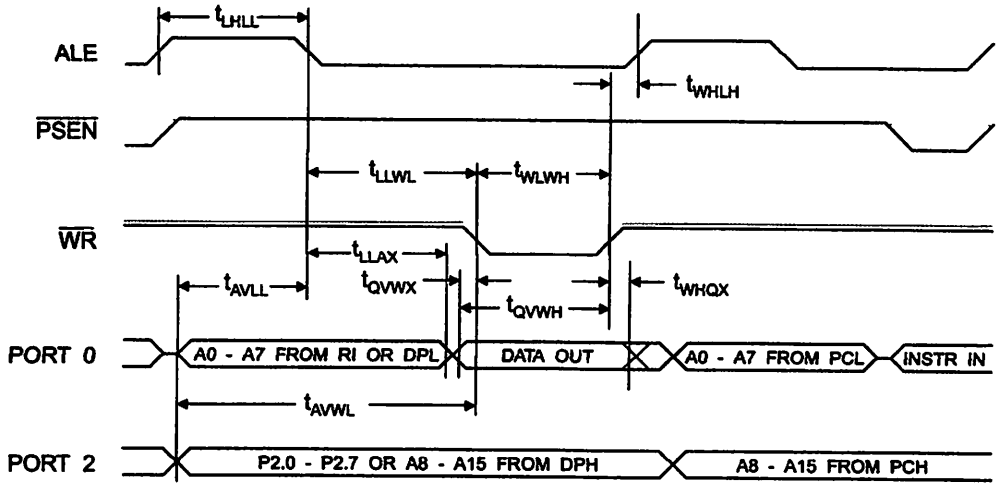
rnal Program Memory Read Cycle



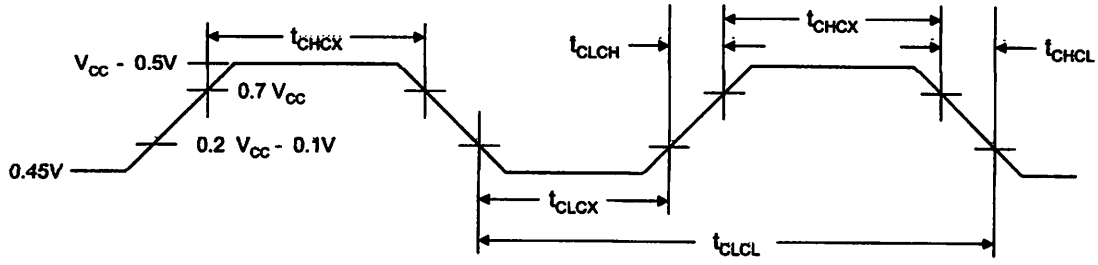
rnal Data Memory Read Cycle



nal Data Memory Write Cycle



nal Clock Drive Waveforms



nal Clock Drive

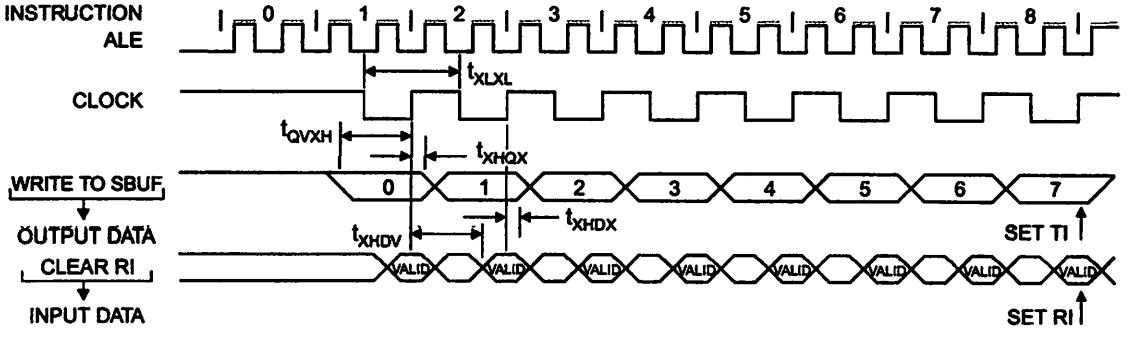
	Parameter	Min	Max	Units
	Oscillator Frequency	0	33	MHz
	Clock Period	30		ns
	High Time	12		ns
	Low Time	12		ns
	Rise Time		5	ns
	Fall Time		5	ns

ıl Port Timing: Shift Register Mode Test Conditions

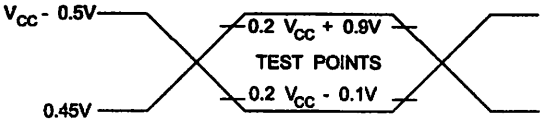
lues in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF:

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

Register Mode Timing Waveforms

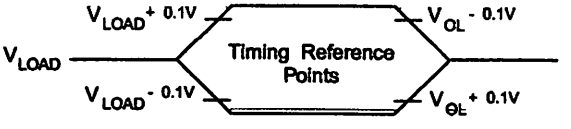


esting Input/Output Waveforms(1)



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

: Waveforms(1)



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

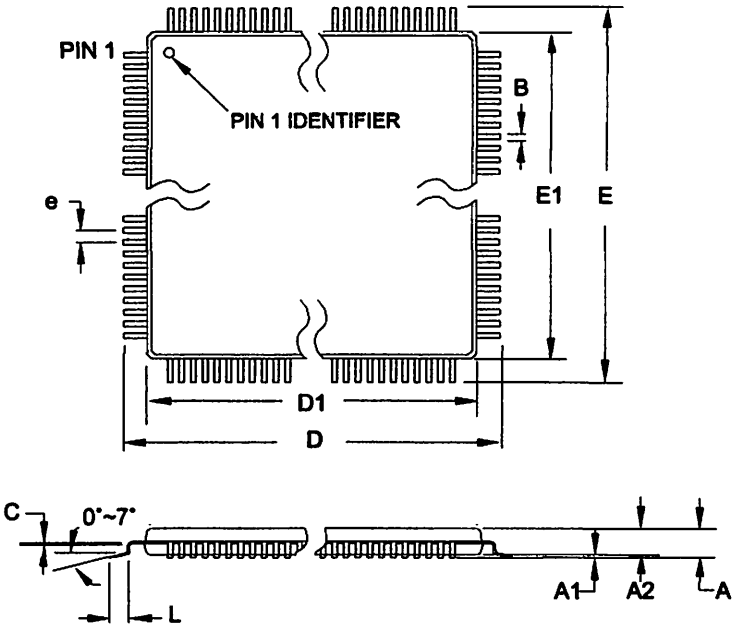
ring Information

d)	Power Supply	Ordering Code	Package	Operation Range
	4.0V to 5.5V	AT89S52-24AC AT89S52-24JC AT89S52-24PC AT89S52-24SC	44A 44J 40P6 42PS6	Commercial (0° C to 70° C)
		AT89S52-24AI AT89S52-24JI AT89S52-24PI AT89S52-24SI	44A 44J 40P6 42PS6	Industrial (-40° C to 85° C)
	4.5V to 5.5V	AT89S52-33AC AT89S52-33JC AT89S52-33PC AT89S52-33SC	44A 44J 40P6 42PS6	Commercial (0° C to 70° C)

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

aging Information

TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

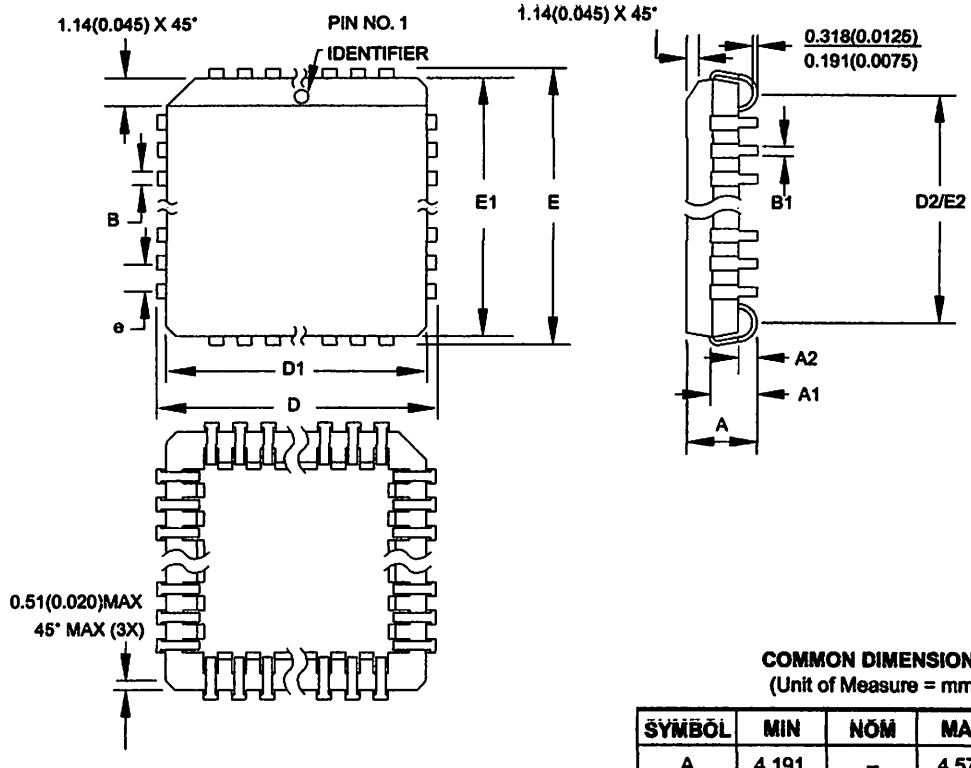
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

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

10/5/2001

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

PLCC



COMMON DIMENSIONS
(Unit of Measure = mm)

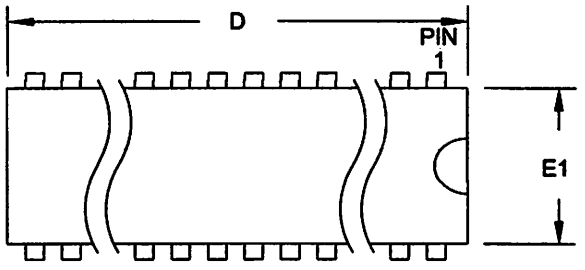
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

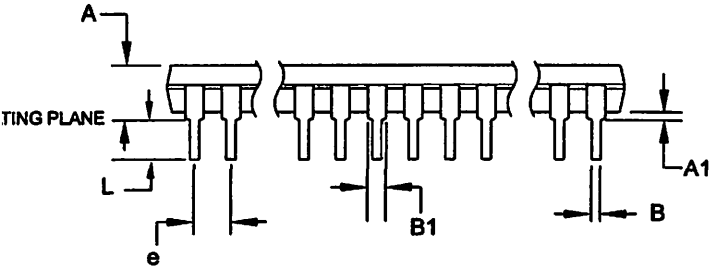
lotes:
1. This package conforms to JEDEC reference MS-018, Variation AC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is .010"(0.254 mm) per side. Dimension D1 and E1 include mold mismatch and are measured at the extreme material condition at the upper or lower parting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

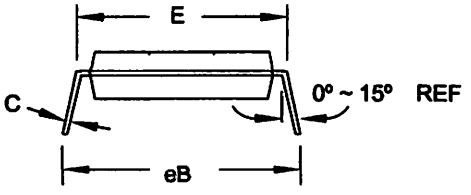
10/04/01

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

– PDIP







COMMON DIMENSIONS
(Unit of Measure = mm)

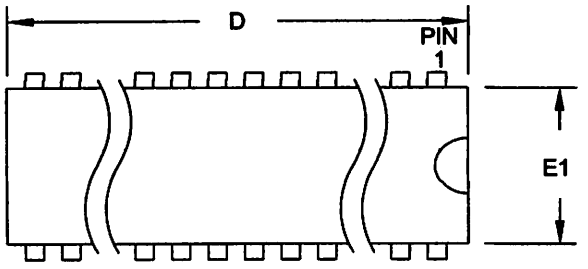
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

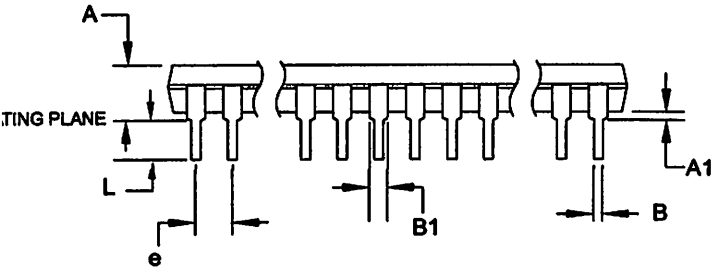
otes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

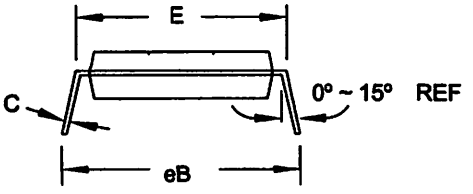
09/28/01

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

– PDIP







COMMON DIMENSIONS
(Unit of Measure = mm)

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

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

11/6/03

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

əl Corporation

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

onal Headquarters

ıe
el Sarl
ıte des Arsenaux 41
e Postale 80
1705 Fribourg
tzerland
(41) 26-426-5555
: (41) 26-426-5500

ım 1219
ıachem Golden Plaza
Mody Road Tsimshatsui
t Kowloon
g Kong
(852) 2721-9778
: (852) 2722-1369

Tonetsu Shinkawa Bldg.
I-8 Shinkawa
o-ku, Tokyo 104-0033
ən
(81) 3-3523-3551
: (81) 3-3523-7581

Atmel Operations

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

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

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

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

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

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

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

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

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

1919B–MICRO–11/03