PENGENDALIAN PINTU GERBANG BERBASIS MIKROKONTROLLER AT 89S51 YANG DIAKTIFKAN DENGAN HP VIA MISCALL

TUGAS AKHIR

Disusun Oleh:
Nama : Handik Prastyawan
Nim : 0752003

PROGRAM STUDI TEKNIK LISTRIK D-III
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

AGUSTUS 2010

LEMBAR PERSETUJUAN

PENGENDALIAN PINTU GERBANG BERBASIS MIKROKONTROLLER AT 89S51 YANG DIAKTIFKAN DENGAN HP VIA MISCALL

TUGAS AKHIR

Disusun dan Diajukan Untuk Melengkapi dan
Memenuhi Syarat-syarat Guna Mencapai Gelar Diploma Tiga

Disusun Oleh:

Nama : Handik Prastyawan
Nim : 0752003

Diperiksa dan Disetujui

Ketua Program Studi	**Dosen Pembimbing**
Teknik Listrik DIII	
Ir. Taufik Hidayat, MT	**Bambang Prio Hartono,ST,MT**
NIP. Y. 1018700151	**NIP. Y. 1028400082**

PROGRAM STUDI TEKNIK LISTRIK D III
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2010

KATA PENGANTAR

Segala puji bagi ALLAH SWT, yang telah memberi *Rahmat* serta *Hidayah* srhingga penulis dapat menyelesaikan penyusunan tugas akhir dengan judul: **"PERANCANGAN DAN PEMBUATAN SISTEM KENDALI PAGAR RUMAH BERBASIS MIKROKONTROLLER AT89S51 YANG DIAKTIFKAN DENGAN MISSCAL HANDPHONE"**

Tugas akhir ini disusun sebagai salah satu syarat kelulusan program D-3 energi listrik di lingkungan Institut Teknologi Nasional Malang. Kelancaran dalam penulisan tugas akhir ini tak lepas bantuan dari semua pihak. Dalam kesempatan ini penulis mengucapkan banyak terimakasih dan penghargaan yang setinggi tingginya kepada:

1. Bapak Dr. Ir. Abraham Lomi,MSEE. Selaku Rektor ITN Malang.
2. Bapak Ir.Sidik Noertjahjono, MT. Selaku Dekan Fakultas Teknologi Industri ITN Malang.
3. Bapak Ir.H. Taufik Hidayat, MT. Selaku Ketua Jurusan Teknik Elektro D-III ITN Malang.
4. Bapak Bambang Prio Hartono,ST,MT. Selaku Dosen Pembimbing.
5. Rekan – rekan dan semua pihak yang telah membantu terselesaikanya Laporan Tugas Akhir ini.

Laporan Tugas Akhir ini tentu masih jauh dari kesempurnaan mengingat keterbatasan pengetahuan dan kemampuan yang penulis miliki. Untuk itu segala saran dan kritik yang sifatnya membangun, penulis hargai demi kebaikan dimasa yang akan dating. Semoga dengan terselesaikannya Laporan Tugas Akhir ini bias bermanfaat bagi penulis khususnya dan bagi pembaca pada umumnya.

Malang, 11 Agustus 2010

Penulis

DAFTAR ISI

BAB III PERANCANGAN DAN PEMBUATAN ALAT

BAB VI PENGUJIAN ALAT

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1
PENDAHULUAN

1.1 Latar Belakang

Perkembangan jaman atau teknologi elektronika semakin moderen menuntut kita untuk lebih tanggap dan jeli memahaminya. Adanya berbagai permasalahan yang dikerjakan manusia atau secara manual, terutama kalau kita melihat orang membuka atau menutup pintu gerbang dengan cara manual (menarik / mendorong) ada juga yang sudah menggunakan tombol buka / tutup, memunculkan ide – ide baru kita untuk memecahkan masalah dan menciptakan suatu sistem perangkat keras atau perangkat lunak elektronik yang lebih handal.

Salah satu perngkat yang dimaksut antara lain adalah pengendali pintu garbang dengan Hp via misscal yanglebih efisien dan memudahkan kita atau pemilik rumah untuk mengendalikan (Membuka dan Menutup) pintu gerbang dari jarak yang cukup jauh, terutama bagi orang yang sibuk dan tidak memiliki seorang pembantu.

Berangkat dari hal ini maka kami terdorong untuk membuat alat secara elektronik yang dapat membantu memecahkan permasalahan tersebut diatas, yaitu dengan membuat pengendali pintu garbang berbasis Mikro controler AT89S51 yang diaktifkan dengan Hp via miscall.

1.2 Rumusan Masalah

Dalam pembuatan tugas akhir ini dibuat sebuah alat untuk mengendalikan pintu gerbang sekaligus mengunci / membuka pintu dan menyalakan / mematikan lampu rumah dengan Hp via Miss Call,sehingga memudahkan pemilik untuk mengendalikan rumah. Adapun permasalahan desain dan pembuatan alat ini dapat dirumuskan sebagai berikut:

1. Merancang dan membuat pengendali pintu gerbang yang di program menggunakan mikrokontroler AT89S51.
2. Bagaimana memanfaatkan mikrokontroler AT89S51 untuk interface Handphone dalam mengendalikan rumah.
3. Bagaimana menciptakan pengontrolan yang baik menggunakan Handphone

1.3 Tujuan penulisan

Tujuan pembuatan tugas akhir ini adalah untuk merencanakan sebuah alat yang dapat difungsikan untuk mengendalikan rumah dengan menggunakan Hp via Miss Call, dimana di dalamnya terdapat system yang dapat mengontrol rumah kita melalui Miss Call yang dikirim dari Handphone kita ke Handphone pada alat control rumah dengan memanfaatkan teknologi Hp.

1.4 Batasan masalah

Dalam laporan tugas akhir ini kami membatasi masalah sebagai berikut:

1. Hanya menggunakan Hp tipe Siemens C45/M35
2. Membahas single chip AT89S51.

3. Membahas komunikasi antara Hp dengan mikrokontroler AT89S51.
4. Tidak membahas power Suppy.
5. Tidak membahas frekuensi Hp.
6. Tidak membahas mekanik pada alat.

1.5 Metodologi

Dalam penulisan Tugas Akhir ini penulis menggunakan metode – metode yang sering di gunakan. Adapun metode tersebut adalah sebagai berikut :

1. Studi Literatur

   Yaitu: penelitian yang dilakukan dengan jalan mengumpulkan data dari buku maupun download di internet, yang akan dijadikan acuan dalam pembahasan masalah tugas akhir ini.

2. Tahap Perencanaan
   - Perencanaan tiap-tiap blok diagram dengan komponen yang sesuai.
   - Perencanaan sistem secara keseluruhan.
3. Tahap Pembuatan
   - Penggabungan tiap-tiap blok menjadi satu kesatuan rangkaian.
   - Pembuatan perangkat keras.
4. Tahap Pengujian Alat.

   Pengujian dilakukan untuk mengetahui unjuk kerja alat.
5. Menarik Kesimpulan.

1.6 Sistematika Pembahasan

Pada penulisan skripsi ini ditulis sedimikian rupa sehingga diperoleh hubungan yang jelas antara bagian yang satu dengan bagian yang lainnya.

Sistematikanya sebagai berikut:

BABI. PENDAHULUAN

Berisi latar belakang, permasalahan, tujuan, batasan masalah, dan sistematika penulisan.

BABII. TEORI PENUNJANG

Meliputi dasar teori yang yang mendukung perancangan alat.

BABIII. PERENCANAAN DAN PEMBUATAN ALAT

Meliputi penjelasan mengenai perancangan dan pembuatan alat.

BABIV. PENGUJIAN ALAT

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

BABV. PENUTUP

Berisi kesimpulan dan saran.

BAB II

TEORI DASAR

2.1 Mikrokontroler AT89S51

Mikrokontroller, sesuai dengan namanya adalah suatu alat atau komponen pengontrol atau pengendali yang berukuran kecil (mikro). Mikrokontroller lebih unggul dibandingkan dengan mikroprosessor, hal ini disebabkan karena :

- Tersedia I/O

  I/O dalam mikrikontroller sudah tersedia, bahkan untuk AT89S51 ada 32 jalur I/O, sementara pada mikroprosessor dibutuhkan IC tambahan untuk menangani I/O.
- Memori internal

  Memori merupakan media untuk menyimpan program dan data sehingga mutlak harus ada. Mikroprosessor belum memiliki memori internal sehingga memerlukan IC memori internal.

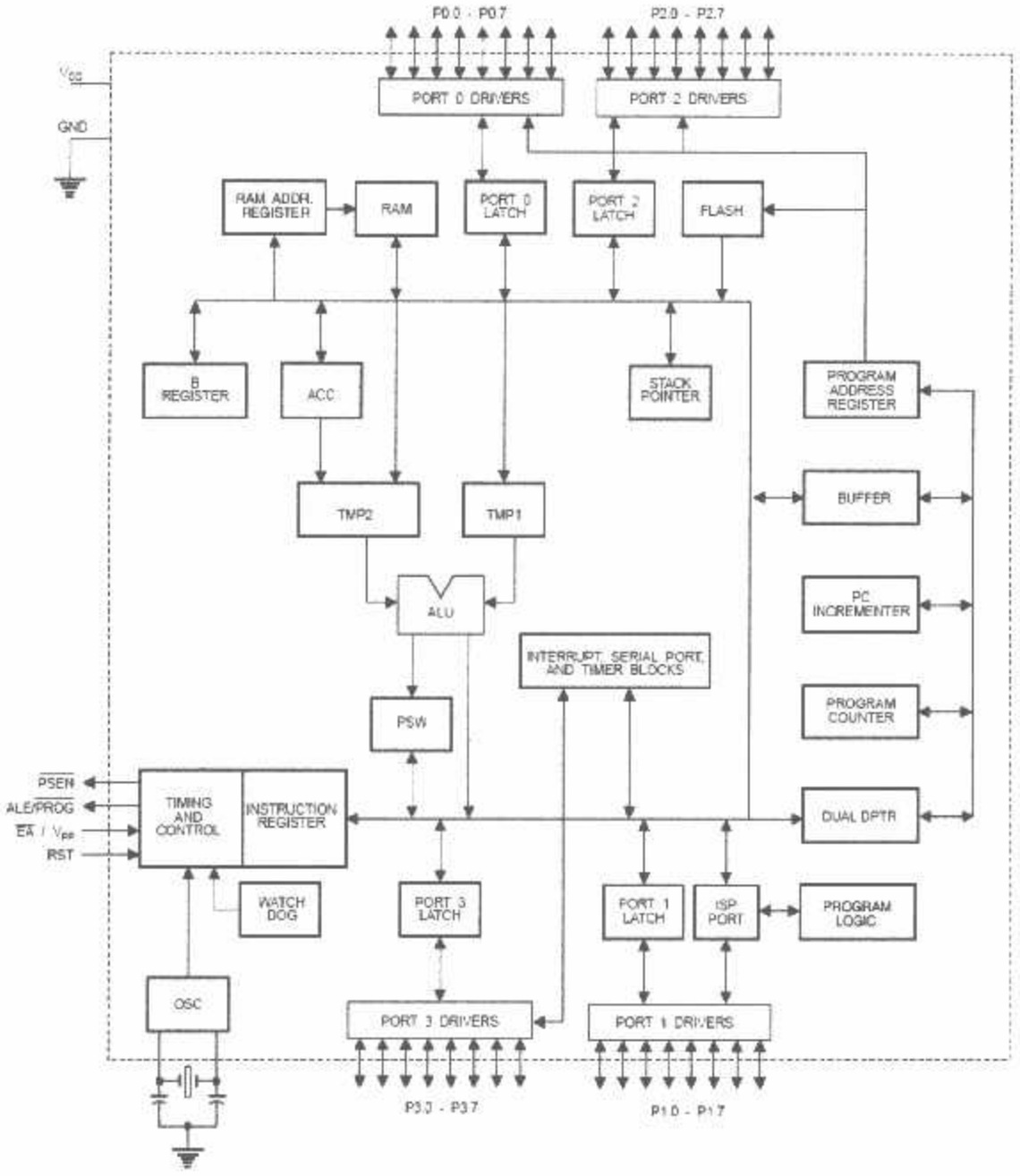
  (Pemrograman Mikrokontroller AT89S51)

2.1.1 Fitur-fitur AT89S51

Mikrokontroller AT89S51 merupakan produk ATMEL yang memilikii fitur-fitur sebagai berikut :

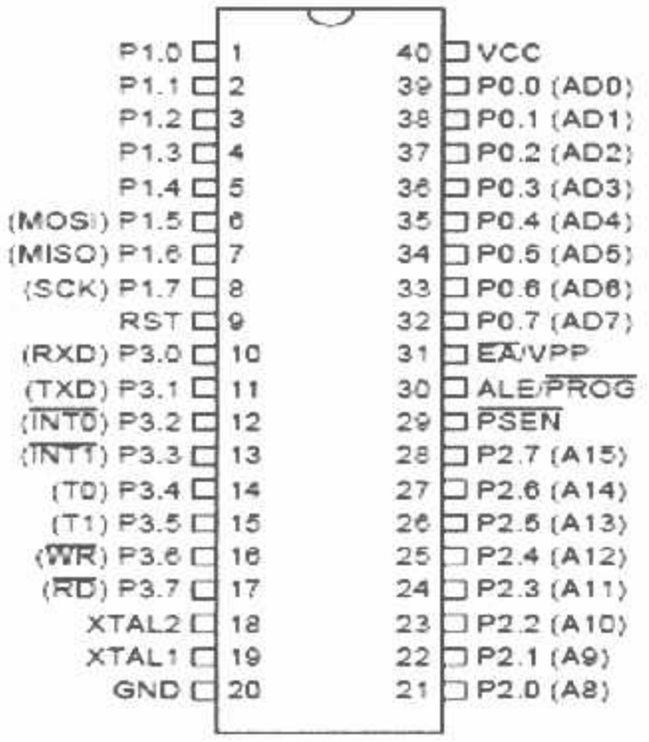
1. Kompatibel dengan MCS-51
2. 4 kbyte memori program yang dapat ditulis hingga 1000 kali
3. 0 kecepatan clock -33 MHz
4. 128 byte memori RAM internal
5. 32 jalur input – output (4 buah port parallel I/O)
6. 2 timer/counter 16 bit
7. 2 data pointer
8. 6 interup (2 timer, 2 counter, 1 serial, 1 reset)
9. ISP (In System Programmable) Flash memori

10. Port serial full-duplex



Gambar 2.1
Blok Diagram AT89S51 (www.atmel.com)

2.1.2 Konfigurasi Pin AT89S51



Gambar 2.2
Konfigurasi Pin IC AT89S51 (www.atmel.com)

Mikrokontroller AT89S51 memiliki pin berjumlah 40 dan umumnya dikemas dalam DIP (*Dual Line Package*). Masing-masing pin mikrokontroller AT89S51 mempunyai kegunaan sebagai berikut :

a. Port 1

   Merupakan salah satu port yang berfungsi sebagai *general purpose* I/O dengan lebar 8 bit. Sedangkan untuk fungsi lainnya , port 1 tidak memiliki.

b. RST

Pin ini berfungsi sebagai input untuk melakukan reset terhadap mikro dan jika RST bernilai high selama minimal 2 *machine cycle*, maka nilai internal register akan kembali seperti awal mula bekerja.

c. Port 3

Merupakan port yang terdiri dari 8 bit masukan dan keluaran. Di samping berfungsi sebagai masukan dan keluaran, port 3 juga mempunyai fungsi khusus lainnya.

Tabel 2.1 Fungsi Khusus Port 3
Sumber: www.atmel.com

Port/Pin	Fungsi Alternatif
P3.0	RXD (port serial input)
P3.1	TXD (port serial output)
P3.2	INT0 (interupsi eksternal 0)
P3.3	INT1 (interupsi eksternal 1)
P3.4	T0 (input eksternal timer 0)
P3.5	T1 (input eksternal timer 1)
P3.6	WR (write strobe memori data eksternal)
P3.7	WR (read strobe memori program eksternal)

d. XTAL 1 dan XTAL 2

Merupakan pin inputan untuk kristal osilator

e. GND

Pada kaki berfungsi sebagai pentanahan (ground)

f. Port 2

Merupakan salah satu port yang berfungsi sebagai *general purfose* I/O dengan lebar 8 bit. Fungsi lainnya adalah sebagai *high byte addres bus* (pada penggunaan memori eksternal).

g. PSEN

PSEN (*program Store Enable*) adalah pulsa pengaktif untuk membaca program memori luar.

h. ALE

Berfungsi untuk *demultiplexer* pada saat port 0 bekerja sebagai *mulatiplexed address/data bus* (pengaksesan memori eksternal). Pada paruh pertama memory cycle, pin ALE mengeluarkan signal latch yang menahan alamat ke eksternal register. Pada paruh kedua memory cycle, port 0 akan digunakan sebagai data bus. Jadi fungsi utama dari ALE adalah memberikan signal ke IC latch (bisa 74HCT573) agar menahan/ menyimpan address dari port 0 yang menuju memori eksternal (address 0-7) dan selanjutnya memori eksternal akan mengeluarkan data yang melalui port 0 juga.

i. EA

EA (*External Access*) harus dihubungkan dengan ground jika menggunakan program memori luar. Jika menggunakan program memori internal maka EA dihubungkan dengan VCC. Dalam keadaan ini mikrokontroller bekerja secara *single chip*.

j. Port 0

Merupakan salah satu port yang berfungsi sebagai *general purpose* I/O (dapat digunakan sebagai masukan dan juga sebagai keluaran) dengan lebar 8 bit. Fungsi lainnya adalah sebagai *multiplexed address/data bus* (pada saat mengakses memori eksternal).

k. VCC

Pada kaki ini berfungsi sebagai tempat sumber tegangan sebesar +5 Volt.

2.1.3 SFR (Special Function Register)

SFR adalah suatu alamat pada memori RAM internal yang memiliki fungsi khusus. Berikut ini penjelasan secara singkat SFR yang dimiliki AT89S51 :

1. Akumulator

   ACC atau akumulator digunakan sebagai register utama dalam proses aritmatik dan penyimpan data sementara. Dalam instruksi pemrograman akumulator dituliskan sebagai A.

2. Register B

   Register B digunakan selama operasi perkalian dan pembagian. Untuk iinstruksi lain dapat diperlakukan sebagai scratch-pad.

3. Stack Pointer

   Register SP (Stack Pointer) merupakan register penunjuk alamat dari stack. Pada operasi PUSH dan POP serta CALL dan Ret maka nilainya akan berubah sesuai dengan alamat stack saat itu.

4. Data Pointer

   Register DPTR (Data Pointer) merupakan register 16 bit yang digunakan sebagai penyimpan alamat data. Terdiri dari DPH sebagai penyimpan high byte dan DPL sebagai penyimpan low byte.

5. Port 0, Port 1, Port 2 dan Port 3

   P0, P1, P2 dan P3 merupakan latches yang digunakan untuk menyimpan data yang akan ditulis dari/ke masing-masing port.

6. Serial Data Buffer

   SBUF (*Serial Data Buffer*) sebenarnya terdiri dari dua registeryang terpisah, yaitu register penyangga pengirim (*transmit buffer*) dan penyangga penerima (*receive buffer*).

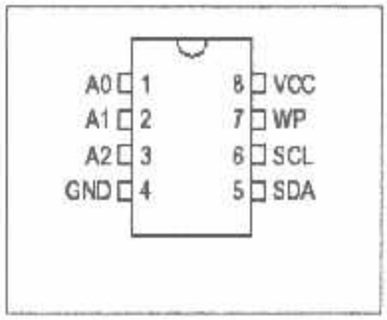
7. Kontrol Register

   Register-register IP, IE, TMOD, SCON, TCON, dan PCON berisi bit-bit control dan status untuk system interufsi, timer, counter dan port serial.

2.2 EEPROM AT24C16

Dalam perencanaaan dan pembuatan tugas akhir ini, mikrokontroller dihubungkan dengan memori data external guna keperluan khusus yang memerlukan penyimpanan data yang tetap jika catu daya pada rangkaian dimatikan.

AT24C16 yang berkapasitas 16 byte ini menempati alamat C00H – DFFFH. Keluarga AT34C16 menonjolkan suatu komunakasi serial dan mendukung suatu bit-directional 2-write bus protokol transmisi. Gambar dari EEPROM AT24C16 diperlihatkan pada gambar 2.3 dibawah ini



Gambar 2.3 EEPROM 24C16
Sumber : Data Sheet EEPROM 24C16

Fungsi dari kaki – kaki sebagai berikut:

1. **A0** sampai dengan A2 adalah untuk masukan.
2. **GND** dihubungkan dengan ground.
3. **SDA** Adalah suatu pin Bi-Directional digunakan perpindahan alamat dan data ke dalam dan keluar dari alat itu. Pin SDA adalah suatu saluran keluaran terbuka dan dapat wire-Ored dengan saluran terbuka lain atau keluaran pengumpulan terbuka. SDA bus memerlukan suatu pullup resistor ke Vcc
4. **SCL** adalah pin masukan clock, ini digunakan untuk mensinkronkan data, memindahkan dari alat itu.
5. **WP** adalah pin untuk melindungi dari penulisan ulang. Jika pin WP diikat ke Vcc yang sebagian atas separuh array menjadi melindungi dari penulisan

ulang (yang dibaca saja). Kapan WP diikat ke GND atau yang ditinggalkan tergantung read / write normal operasi diijinkan kepada alat itu.

6. **VCC** dihubungkan dengan power supply.

Ketika pembacan alamat akan dilakukan maka alamat rendah (low Byte) tersimpan di port 0. Sedangkan alamat tertinggi (high order) tersimpan di port 2. Sinyal ALE kemudian diberikan ke addres latch berisi addres low yang dikehendaki. Selanjutnya sinyal RCT DS 1287A / WR di berikan ke EEPROM dank ode bit dapat dibaca oleh mikrokontroller AT89S51.

2.3 Kabel data Siemen C35 / 45

Data kabel adalah merupakan perangkat keras yang merupakan sarana penghubung antara ponsel dan mikrokontroller agar bisa berkomunikasi. Seperti kabel printer, penghubung printer dengan computer sebagai sarana komunikasi antar dua perangkat keras. Dengan kata lain, apa yang bisa dilakukan oleh data kabel tergantung dari softwarenya.

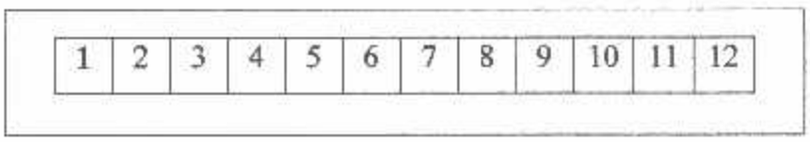
Keindahan menggunakan data kabel adalah fungsinya yang tak terbatas karena setiap langkah digerakkan oleh perangkat lunak (software). Data kabel hanya sebagai penterjemah antara ponsel dengan mikrokontroller. Mulai dari koneksi ke internet atau fax sampai dengan mengorganisasikan isi dari SIM card anda ataupun fitur ponsel. Termasuk juga untuk mengganti logo dan nada dering pada ponsel yang memiliki fitur tersebut, upgrade versi, ganti bahasa, dan lain sebagainya.

Macam data kabel yaitu: FBUS, MBUS, dan gabungan FBUS / MBUS. FBUS memerlukan dua input / output untuk mengirim dan menerima data dimana MBUS hanya memerlukan satu input / output saja. Hanya dengan hanya dengan kabel MBUS anda dapat mengakses data penting di dalam ponsel untuk menservis ponsel, upgrate software, dan lain – lain. FBUS biasanya digunakan untuk memasukkan logo dan ringtone di ponsel serta untuk kebutuhan mobilitas seperti konek ke internet dan faxing.

Kecepatan data ketika untuk kirim maupun terima biasanya 9,6 K sampai dengan 14,4 K.

2.4 Interface Pada Hanphone dengan Mikrokontroller

Handphone saat ini bukanlah alat yang mewah, hamper semua dapat memilikinya. Logo maupun ringtone – nya dapat diganti sesuai dengan keinginan user. Dunia komunikasi saat ini sudah semakin canggih, alat komunikasi sekarang tidak sekedar untuk berkonunikasi saja namun telah menjadi alat bantu bagi para pengguna. Semakin banyak merek ataupun jenis handphone sering menimbulkan masalah pada interface antara mikrokontroller dan hanphone tidak mempunyai standart, tidak seperti protocol komonikasi RS232 yang sudah distandartkan dalam IEEE. Oleh karena itu, agar sebuah mikrokontroller dengan hanphone maka perlu diketahui koneksi pada setiap jenis hanphone. Contoh konektor pada handphone SIEMENS C35 / C45 dapat dilihat pada gambar berikut:



Gambar 2.4 Konektor Siements C35 / C45
Sumber : Data Sheet Cable Data For Siemens C35 / C45

Untuk pengguna dari masing – masing pin pada konektor diatas dapat dilihat pada table berikut:

Tabel 2.2 Pin out Konektor Siemens C35 / C45
Sumber : Data Sheet Cable Data For Siemens C35 / C45

NAMA	FUNGSI	In / out
GND	Ground	
Self Service	Recognition/control battery carge	In/out
LOAD	Charging Voltage	In

BATTERY	Battery	Out
DATA OUT	Data Sent	Out
DATA IN	Data Received	In
Z_CLK	Recognition/control accessories	
Z_DATA	Recognition/control accessories	
MICG	Ground for microphone	In
MIC	Mikrophone input	In
AUD	Loudspeaker	Out
AUGND	Ground for eksternal speaker	

2.5 Relay

Dalam dunia elektronika ,relay dikenal sebagai komponen yang dapat mengimplementasikan logika switching.Sebelum tahun 70an,relay merupakan "otak"dari rangkaian pengendali.Baru setelah itu muncul PLC yang mulai menggantikan posisi relay. Relay yang paling sederhana ialah relay elektromekanis yang memberikan pergerakan mekanis saat mendapatkan energi listrik.secara sederhana relay elektromekanis ini didenifisikan sebagai berikut:

• Alat y ang menggunakan gaya elektromagnetik untuk menutup (atau membuka)

kontak saklar. Saklar yang digerakkan (secara mekanis) oleh daya /energi listrik.

2.5.1 Prinsip kerja dan simbol Relay

Relay terdiri dari coil dan contact. perhatikan gambar 2.6,coil adalah gulungan kawat yang mendapat arus listrik,sedang contact adalah sejenis saklar yang pergerakan nya tergantung dari dari ada tidak nya arus listrik di coil.

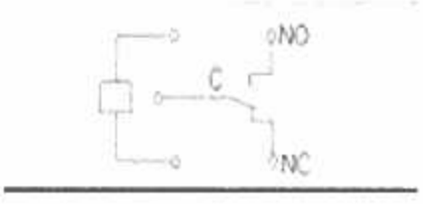
Contact ada 2 jenis:

a. Normally Open (kondisi awal sebelum di aktifkan open).
b. Normally Closed (kondisi awal sebelum di aktifkan close).

Secara sederhana berikut ini prinsip kerja dari relay : ketika coil mendapat energi listrik (energized),akan timbul gaya electromagnet yang akan menarik armature yang perpegas,dan contak akan menutup

Bagian titik kontak di bagi menjadi 2 bagian yaitu bagian kontak utama dan kontak Bantu yaitu

1. Bagian kontak utama gunanya untuk menghubungkan dan memutuskan arus listrik bagian yang menuju beban/pemakai.
2. Bagian kontak Bantu gunanya untuk menghubungkan dan memutusknan hubungan listrik ke bagian yang menuju bagian pengendali.
   Kontak Bantu mempunyai 2 kontak yaitu kontak hubunh (NC) dan kontak putus (NO) menandakan masing-masing kontak dan gulungan spool.



Gambar 2.5 symbol relay
Sumber : Direct industri 2005

2.5.2 Jenis-jenis Relay

Seperti saklar ,relay juga dibedakan berdasarkan pole dan throw yang di miliki nya .

Berikut definisi pole dan throw :

. pole : banyak nya contact yang dimiliki oleh relay

. Throw : banyaknya kondisi (state) yang mungkin dimiliki contact

Berikut ini penggolongan relay berdasarkan jumlah pole dan throw :

. SPST (Single Pole Single Throw)

. DPST (Doudle Pole Single Throw)

. SPDT (Single Pole Double Throw)

. DPDT (Double Pole Double Throw)

. 3PDT (Three Pole Double Throw)

. 4PDT (Four Pole Double Throw)

2.6 LCD

Tampilan Kristal Cair (bahasa_Inggris: *Liquid Crystal Display*) juga dikenal sebagai *LCD* adalah suatu jenis media tampilan yang menggunakan kristal cair sebagai penampil utama. LCD sudah digunakan di berbagai bidang misalnya dalam alat-alat elektronik seperti televisi, kalkulator ataupun layar komputer. Kini LCD mendominasi jenis tampilan untuk komputer desktop maupun notebook karena membutuhkan daya listrik yang rendah, bentuknya tipis, mengeluarkan sedikit panas dan beresoulusi tinggi.

Pada LCD berwarna semacam monitor terdapat banyak sekali titik_cahaya (piksel) yang terdiri dari satu buah kristal cair sebagai sebuah titik cahaya. Walau disebut sebagai titik cahaya, namun kristal cair ini tidak memancarkan cahaya sendiri. Sumber cahaya di dalam sebuah perangkat LCD adalah lampu neon berwarna putih di bagian belakang susunan kristal cair tadi.

Titik cahaya yang jumlahnya puluhan ribu bahkan jutaan inilah yang membentuk tampilan citra. Kutub kristal cair yang dilewati arus listrik akan berubah karena pengaruh polarisasi medan magnetik yang timbul dan oleh karenanya akan hanya membiarkan beberapa warna diteruskan sedangkan warna lainnya tersaring.

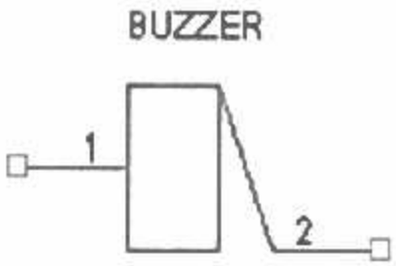
2.7 Lampu Pijar

Lampu pijar adalah sumber cahaya buatan yang dihasilkan melalui penyaluran arus listrik melalui filamen yang kemudian memanas dan menghasilkan foton. Kaca yang menyelubungi filamen panas tersebut menghalangi oksigen di udara dari berhubungan dengannya sehingga filamen tidak akan langsung rusak akibat teroksidasi. Suhu warna cahaya tampak yang dipanjarkan oleh lampu pijar adalah di antara 2700K sampai dengan 3300K.

Salah satu kelebihan lampu pijar adalah dapat dihasilkannya lampu pijar dalam berbagai besar voltase, dari puluhan hingga ratusan volt, namun karena energi listrik yang diperlukan lampu pijar untuk menghasilkan cahaya yang terang lebih besar dibandingkan dengan sumber cahaya buatan lainnya, maka secara bertahap lampu pijar mulai digantikan lampu pendar, LED, dan lain-lain.

2.8 Buzzer

Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Pada dasarnya prinsip kerja buzzer hampir sama dengan loud speaker, jadi buzzer juga terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet, kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara. Buzzer biasa digunakan sebagai indikator bahwa proses telah selesai atau terjadi suatu kesalahan pada sebuah alat (alarm).



Gambar 2.6 Simbol Buzzer
(Sumber. Direct Industry, 2005)

2.9 Limit Switch

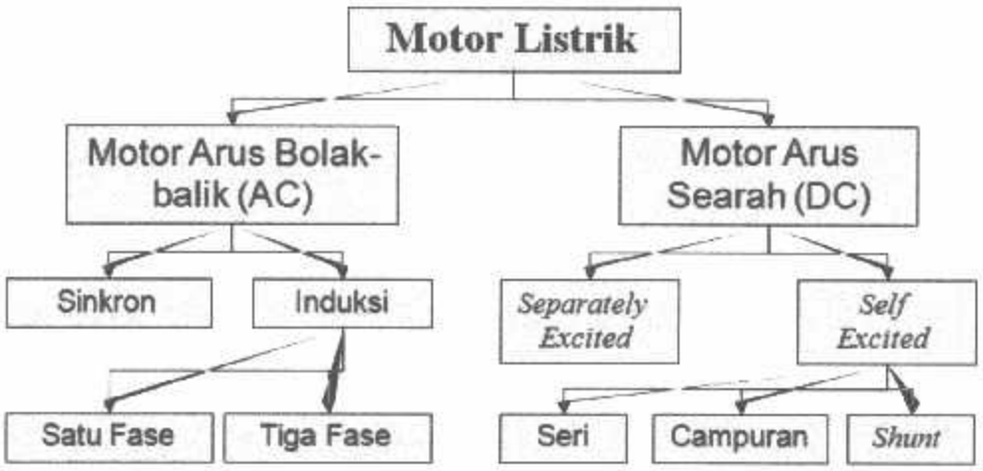
Limit Switch adalah sensor peraba yang bersifat mekanis dan mendeteksi sesuatu setelah terjadi kontak fisik. Penggunaan sensor ini biasanya digunakan untuk membatasi gerakan maksimum sebuah mekanik. Contohnya pada penggerak lengan di mana limit switch akan aktif dan memberikan masukan pada CPU untuk menghentikan gerak motor di saat lengan sudah ditarik maksimum.

Sensor ini juga seringkali digunakan untuk sensor cadangan bilamana sensor yang lain tidak berfungsi. Contohnya pada bagian pinggir dari sebuah robot, pada saat.sensor infrared gagal berfungsi untuk mendeteksi adanya halangan, maka limit switch akan mendeteksi dan memerintahkan motor untuk berhenti saat terjadi kontak fisik.

2.10 Jenis Motor Listrik

Bagian ini menjelaskan tentang dua jenis utama motor listrik: DC dan motor AC.

Motor tersebut dikategorikan berdasarkan pasokan *input*, konstruksi, dan mekanisme operasi, dan dijelaskan lebih lanjut dibawah ini.



Gambar 2.7 Klasifikasi jenis motor listrik

(Sumber. Direct Industry, 2005)

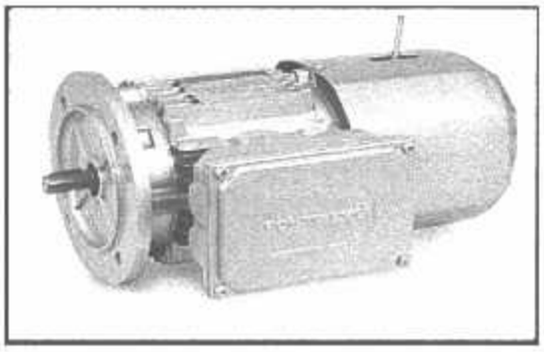
2.11 Motor DC

Motor arus searah, sebagaimana namanya, menggunakan arus langsung yang tidak langsung/*direct-unidirectional*. Motor DC digunakan pada penggunaan khusus dimana diperlukan penyalaan *torque* yang tinggi atau percepatan yang tetap untuk kisaran kecepatan yang luas.

Gambar 2.7 memperlihatkan sebuah motor DC yang memiliki tiga komponen utama:

- ***Kutub medan.*** Secara sederhada digambarkan bahwa interaksi dua kutub magnet akan menyebabkan perputaran pada motor DC. Motor DC memiliki kutub medan yang stasioner dan dinamo yang menggerakan *bearing* pada ruang diantara kutub medan. Motor DC sederhana memiliki dua kutub medan: kutub utara dan kutub selatan. Garis magnetik energi membesar melintasi bukaan diantara kutub-kutub dari utara ke selatan. Untuk motor yang lebih besar atau lebih komplek terdapat satu atau lebih elektromagnet. Elektromagnet menerima listrik dari sumber daya dari luar sebagai penyedia struktur medan.

- ***Dinamo.*** Bila arus masuk menuju dinamo, maka arus ini akan menjadi elektromagnet. Dinamo yang berbentuk silinder, dihubungkan ke as penggerak untuk menggerakan beban. Untuk kasus motor DC yang kecil, dinamo berputar dalam medan magnet yang dibentuk oleh kutub-kutub, sampai kutub utara dan selatan magnet berganti lokasi. Jika hal ini terjadi, arusnya berbalik untuk merubah kutub-kutub utara dan selatan dinamo.
- ***Commutator.*** Komponen ini terutama ditemukan dalam motor DC. Kegunaannya adalah untuk membalikan arah arus listrik dalam dinamo. *Commutator* juga membantu dalam transmisi arus antara dinamo dan sumber daya.



Gambar 2.8 Sebuah motor DC
(Sumber. Direct Industry, 2005)

Keuntungan utama motor DC adalah sebagai pengendali kecepatan, yang tidak mempengaruhi kualitas pasokan daya. Motor ini dapat dikendalikan dengan mengatur:

- Tegangan dinamo – meningkatkan tegangan dinamo akan meningkatkan kecepatan
- Arus medan – menurunkan arus medan akan meningkatkan kecepatan.

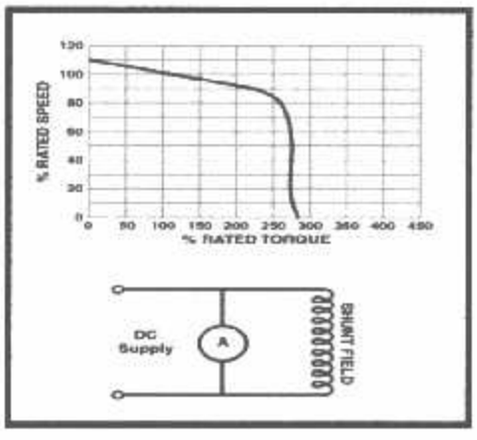
Motor DC tersedia dalam banyak ukuran, namun penggunaannya pada umumnya dibatasi untuk beberapa penggunaan berkecepatan rendah, penggunaan daya rendah hingga sedang seperti peralatan mesin dan *rolling mills*, sebab sering terjadi masalah dengan perubahan arah arus listrik mekanis pada ukuran yang lebih besar. Juga, motor tersebut dibatasi hanya untuk penggunaan di area yang bersih dan tidak berbahaya sebab resiko percikan api pada sikatnya. Motor DC juga relatif mahal dibanding motor AC.

2.11.1 Motor DC sumber daya terpisah/ *Separately Excited*

Jika arus medan dipasok dari sumber terpisah maka disebut motor DC sumber daya terpisah / *separately excited.*

2.11.2 Motor DC sumber daya sendiri/ *Self Excited*: motor *shunt*

Pada motor *shunt*, gulungan medan (medan *shunt*) disambungkan secara paralel dengan gulungan dinamo (A) seperti diperlihatkan dalam gambar 4. Oleh karena itu total arus dalam jalur merupakan penjumlahan arus medan dan arus dinamo.



Gambar 2.9: Karakteristik Motor DC *Shunt*
(Sumber: Rodwell International Corporation, 1999)

Berikut tentang kecepatan motor *shunt* (E.T.E., 1997):

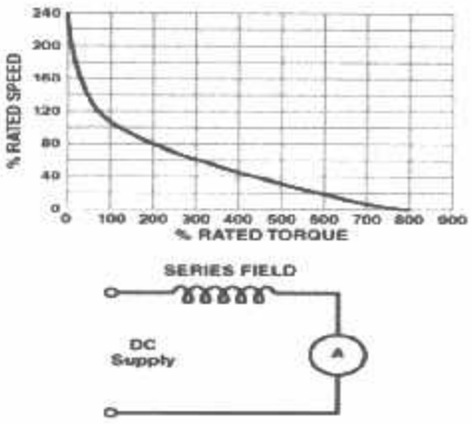
- Kecepatan pada prakteknya konstan tidak tergantung pada beban (hingga *torque* tertentu setelah kecepatannya berkurang, lihat Gambar 2.8) dan oleh karena itu cocok untuk penggunaan komersial dengan beban awal yang rendah, seperti peralatan mesin.
- Kecepatan dapat dikendalikan dengan cara memasang tahanan dalam susunan seri dengan dinamo (kecepatan berkurang) atau dengan memasang tahanan pada arus medan (kecepatan bertambah).

2.11.3 Motor DC daya sendiri: motor seri

Dalam motor seri, gulungan medan (medan *shunt*) dihubungkan secara seri dengan gulungan dinamo (A) seperti ditunjukkan dalam gambar 5. Oleh karena itu, arus medan sama dengan arus dinamo. Berikut tentang kecepatan motor seri (Rodwell International Corporation, 1997; L.M. Photonics Ltd, 2002):

- Kecepatan dibatasi pada 5000 RPM
- Harus dihindarkan menjalankan motor seri tanpa ada beban sebab motor akan mempercepat tanpa terkendali.

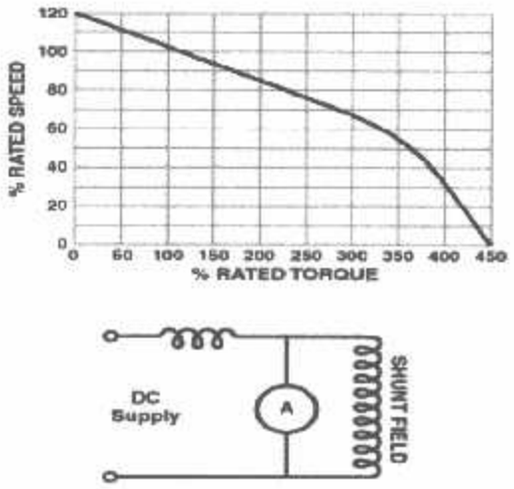
Motor-motor seri cocok untuk penggunaan yang memerlukan *torque* penyalaan awal yang tinggi, seperti derek dan alat pengangkat *hoist* (lihat Gambar 2.10).



Gambar 2.10 Karakteristik Motor Seri DC
(Rodwell International Corporation, 1999)

2.11.4 Motor DC Kompon/Gabungan

Motor Kompon DC merupakan gabungan motor seri dan *shunt*. Pada motor kompon, gulungan medan (medan *shunt*) dihubungkan secara paralel dan seri dengan gulungan dinamo
(A) seperti yang ditunjukkan dalam gambar 2.10. Sehingga, motor kompon memiliki *torque* penyalaan awal yang bagus dan kecepatan yang stabil. Makin tinggi persentase penggabungan (yakni persentase gulungan medan yang dihubungkan secara seri), makin tinggi pula *torque* penyalaan awal yang dapat ditangani oleh motor ini. Contoh, penggabungan 40-50% menjadikan motor ini cocok untuk alat pengangkat *hoist* dan derek, sedangkan motor kompon yang standar (12%) tidak cocok (myElectrical, 2005).



Gambar 2.11 Karakteristik Motor Kompon DC

(Rodwell International Corporation, 1999)

2.13 Pengkajian Motor Listrik

Bagian ini menjelaskan tentang bagaimana mengkaji kinerja alat dalam menentukan kapasiatas motor dalam menarik suatu beban. Diperoleh sebuah permasalahan sebaga berikut:

Motor untuk pintu pagar dengan berat pintu 250 kg dan ditarik dengan panjang 5 meter dalam waktu 10 detik. Hitung daya motor listrik dalam kW dan HP (House Power, 1HP = 746 Watt)

Penyelesaian

Daya mekanik

P = W . t Watt

Keterangan

P = daya mekanik dalam Watt (W)

W = tenaga mekanik dalam Joule

F = Gaya (Force) dalam Newton (N)

t = waktu kerja dalam detik

F = 9,8 . m

= 9,8 x 250 = 2450 N

W = F . d

= 2450 x 5

= 12250 J

Daya Motor

P = W/t

P = 12.250 / 10

= 1225 Watt

= 1.225 kW

P = 1.225 / 746

= 1,64 HP

Gear box 1/10

P = P x Gear box

= 1,64 x o,1

= 0,164

Jadi untuk menarik sebuah pintu pagar seberat 250 kg membutuhkan motor listrik dengan daya 1.225 kW, 1,64 HP dan membutuhkan Gear box 1/10 = 0,164 rpm.

BAB III

PERANCANGAN DAN OEMBUATAN ALAT

3.1 Pendahuluan

Dalam bab ini akan dibahas mengenai perancangan dan pembuatan perangkat keras (Hadware), beserta perangkat lunak (Sofware) pada alat yang akan dibuat yakni: PERANCANGAN DAN PEMBUATAN SISTEM KENDALI PAGAR RUMAH BERBASIS MIKROKONTROLLER AT89S51 YANG DIAKTIFKAN DENGAN MISSCALHANDPHONE. Sedangkan perangkat keras (Hardware) yang diperlukan selain chip AT89S51 antara lain meliputi:

» IC EEPROM 24C16

» Ponsel (Hand Phone) SIEMENS C35/45

» Kabel Data Handphone SIEMENS C35/45

» Pengontrol Relay (Driver)

» Keypad

» LCD

» Saklar Pembatas (Limit Switch)

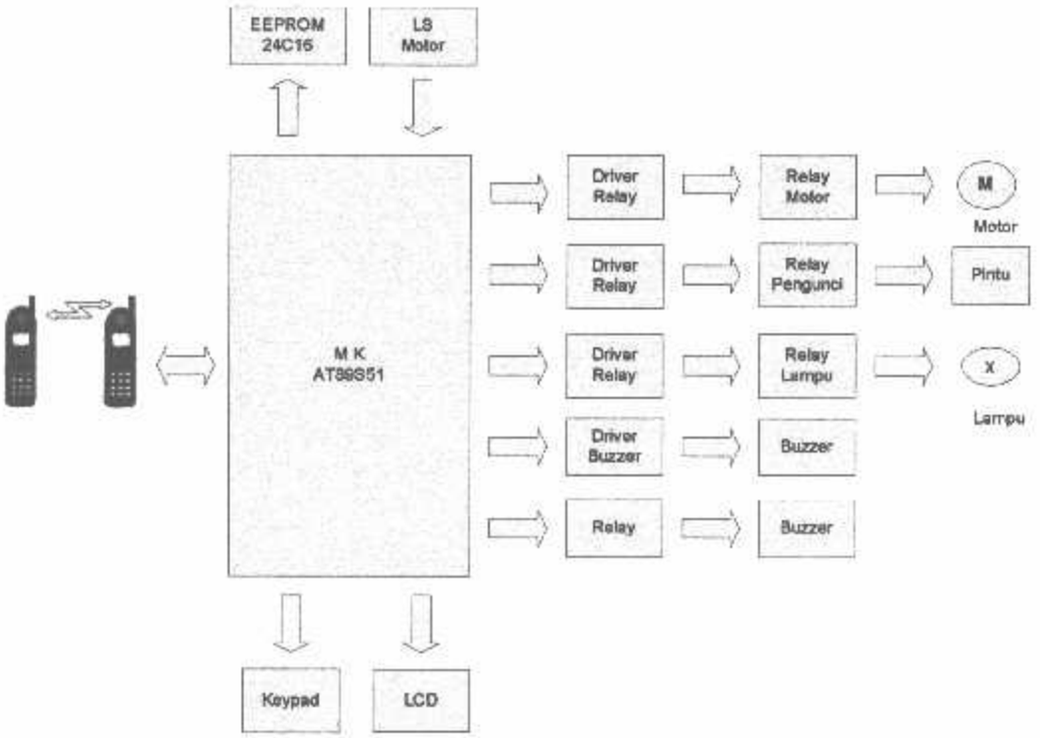
» Motor Gearbox DC

» Relay Pengunci

» Lampu

3.2 Perancangan Perangkat Keras (Hardware)

Secara garis besar, prinsip kerja dari alat pengontrol keadaan rumah dengan handphone via miss call ini dapat digambarkan seperti yang terlihat melalui blok diagram dibawah ini:



Gambar 3.1 Digram blok rangkaian

Adapun fungsi dari masing – masing blok adalah sebagai berikut :

- EEPROM 24C16 : Berfungsi untuk menyimpan daftar nomer telepon yang dapat mengaktifkan alat control ini.
- Mikrokontroller AT 89C52 : Berfungsi sebagai kontrol utama pada system.
- Keypad : Berfungsi untuk memilih bagian mana yang akan dikontrol dan sebagai alat untuk memasukkan nomer telpon yang dapat mengaktifkan alat alat ini.
- LCD : Berfungsi sebagai tampilan dari system

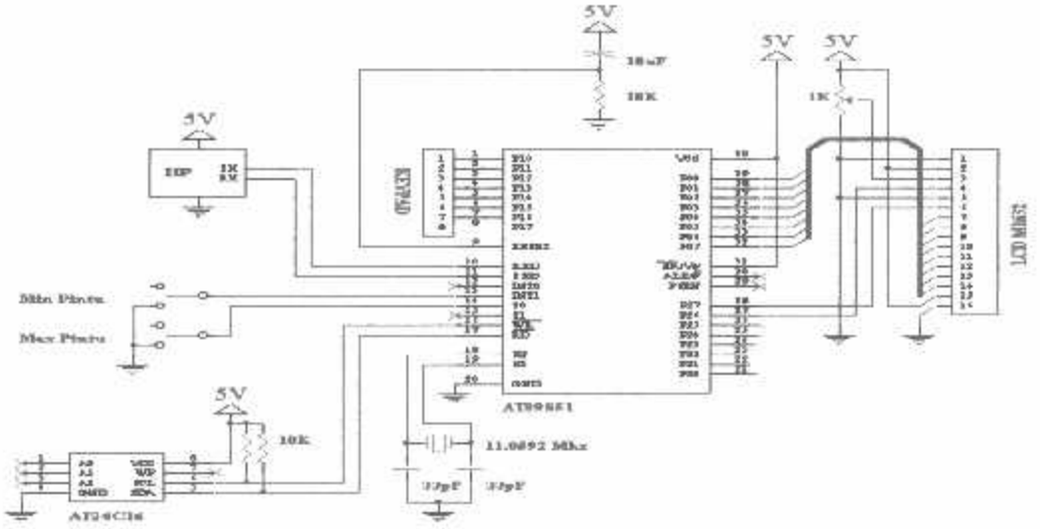
- Relay : Berfungsi untuk saklar on / off sekaligus membalik putaran motor, Selain itu juga berfungsi untuk saklar lampu dan pengunci.
- Motor : Berfungsi untuk membuka dan menutup pintu pagar sekaligus Sebagai kunci pagar.
- Lampu : Berfungsi sebagai penerangan dalam rumah.
- Limit switch : Berfungsi sebagai pembatas dari putaran motor ketika sedang membuka atau menutup pintu.
- Buzzer : Berfungsi sebagai pengaman dari pintu, ketika dibuka paksa maka Buzzer akan bunyi.

Dari blok diagram diatas dapat dijelaskan prinsip kerja secara umum dari sistem adalah sebagai berikut ;

Jika handphone pada alat mendapat telpon dari luar maka otomatis nomer telpon yang menghubungi tadi akan diketahui melalui layar pada handphone. Oleh mikrokontroller, nomer trersebut akan dicocokan dengan daftar nomer telpon yang ada di database. Apabila nomer tersebut cocok maka mikrokontroller akan memerintahkan driver untuk mengaktifkan relay. Sebelumnya harus diset terlebih dahulu bagian mana yang akan diaktifkan oleh mikrokontroller. Relay akan bekerja sesuai dengan perintah yang diberikan mikrokontroller seperti membuka / menutup pintu, mengunci / membuka kunci pintu, maupun menyalakan dan mematikan lampu rumah. Sedangkan jika nomer telpon yang menghubungi tadi tidak cocok mikrokontroller tidak akan bekerja.

3.2.1 Perancangan Mikrokontroller AT89S51

AT89S51 adalah chip berdaya rendah, perfoma tinggi 8 – bit CMOS dengan 256 x 8 EEPROM. Device dibuat dengan menggunakan teknologi Atmel's high density nonfolatile memoriy dan compatible dengan standar industri. AT89S51 diklasifikasikan dalam embedded Microcontroller yang artinya dapat deprogram ulang.



Gambar 3.2 Rangkaian Mikrokontroller AT89S51
dengan Komponen Lain

Fungsi kaki – kaki Mikrokontroller AT89S51 adalah sebagai berikut:

- Pin 1 sampai 8 digunakan sebagai output ke LCD.
- Pin 9 digunakan sebagai reset.
- Pin 10, dan 11 dihubungkan dengan kabel data data Hp.
- Pin 15, dan 16 dihubungkan dengan EEPROM 24C16.
- Pin 18 dan 19 sebagai oscillator.
- Pin 20 sebagai ground.
- Pin 21 sampai 28 dihubungkan dengan driver relay dan limit switch.

- Pin 31 dihubungkan power supplay.
- Pin 32 sampai 39 digunakan sebagai keypad.

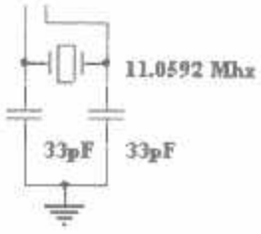
3.2.2 Rangkaian Clock

Mikrokontroller AT 89S51 ini memiliki rangkain internal clock generator yang berfungsi sebagai sumber clock, tetapi masih diperlukan rangkaian tambahan untuk membangkitkan clock yang diperlukan.

Rangkain ini terdiri dari dua buah kapasitor danb dua buah kristal. dengan ketentuan sebagai berikut:

- 20pf - 40pf untuk besarnya nilai kristal.
- 30pf – 50pf untuk besarnya nilai capasitor.

Gambar dari rangkain clock adalah sebagai berikut:

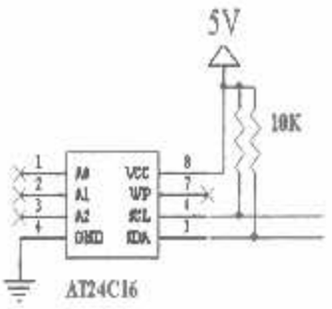


Gambar 3.3 Rangkain Clok AT89S51

3.2.3 Perancangan EEPROM 24C16

Dalam perencanaan dan pembuatan ini, mikrokontroller dihubungkan dengan memori data external guna keperluan khusus yang memerlukan penyimpanan data yang tetap jika catu daya pada rangkain dimatikan.

EEPROM 24C16 yang berkapasitas 8 Kbyte ini merupakan memory dengan kounikasi serial dan mendukung suatu bi-directional 2-wire bus protokol transmisi. Gambar ini rangkain EEPROM 24C16 diperlihatkan pada gambar 3.3 Mikrokontroller AT89S51 dihubungkan melalui pin 5 dan 6.



Gambar 3.4 Rangkain Memory EEPROM 24C16

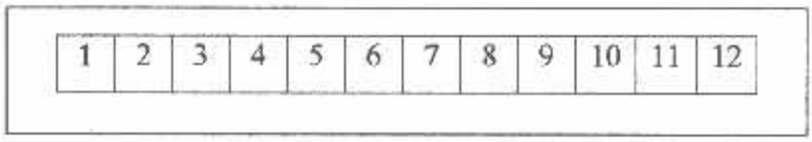
Fungsi dari kaki – kaki EEPROM 24C16 adalah sebagai berikut:

- Pin 1 sampai 3 dan 7 dihubungkan dengan ground.
- Pin 5 dan 6 dihubungkan dengan mikrontroller AT89S51.
- Power supply dihubungkan ke pin 5.

3.2.4 Perencanaan Kabel Data SIEMENS C35 / C45

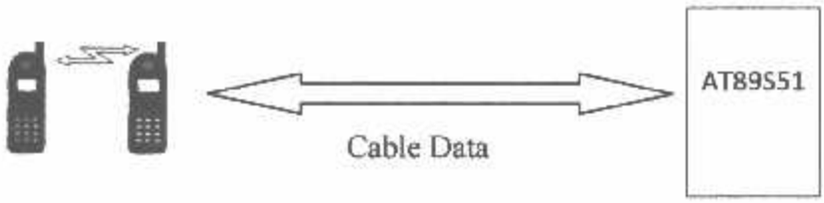
Agar mikrokontroller dapat mencocokkan nomer telpon yang masuk maka diperlukan suatu alat yang dapat mentransfer data nomer telpon tersebut dari handphone ke mikrokontroller. Untuk itu digunakan kabel data yang disertakan dari handphone yang digunakan pada alat ini.

Contoh konektor pada kabel data handphone SIEMENS C35/45 dapat dilihat pada gambar berikut:



Gambar 3.5 Konektor Kabel Data Siemens C35/C45

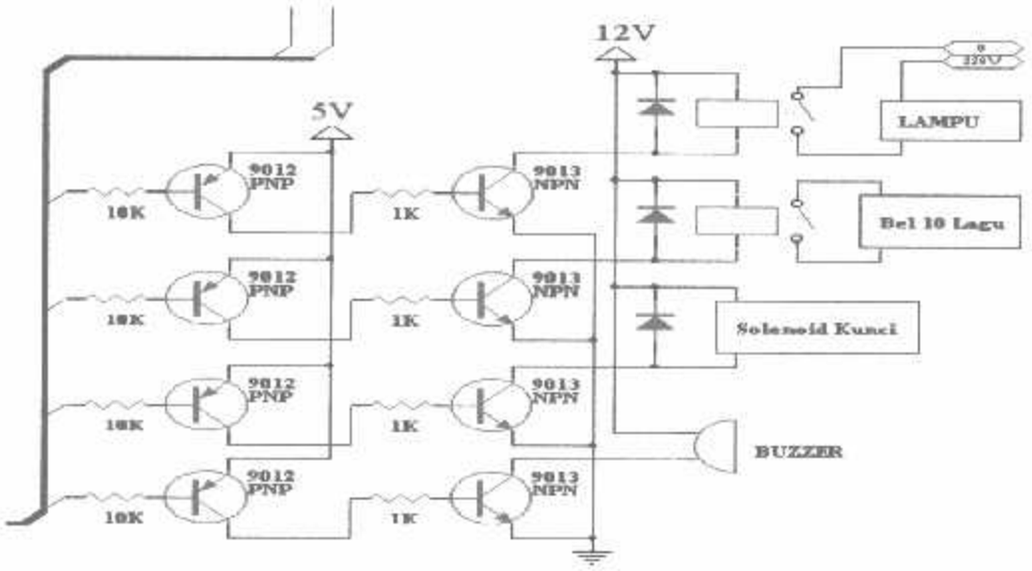
Untuk meng-inter-face-kan Handphone Siemens C35 / C45 dengan mikrokontroller dapat digunakan skematik rangkaian pada gambar berikut:



Gambar 3.6 Blok Diagram
Interface Siemens C35/C45 Dengan Mikrokontroller

3.2.5 Perancangan Rangkaian Pengontrol Relay (Driver)

pada tugas akhir ini digunakan 8 buah pengontrol relay. relay ini berfungsi sebagai penggerak relay. Relay akan aktif bila port aktif berlogika 1. Transistor yang digunakan adalah type 9012 dan 9013 yang berfungsi sebagai switch. Jika ada sinyal dari mikrokontroller maka transistor akan saturasi sehingga relay menjadi aktif. Relay disini menggunakan catu daya sebesar 12 volt.



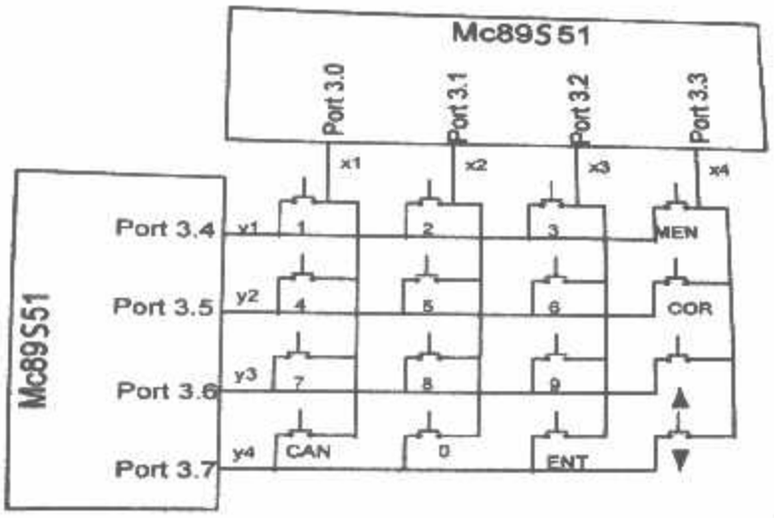
Gambar 3.7 Rangkaian pengontrol relay

3.2.6 Perancangan Keypad

Pada alat ini digunakan keypad matrik 4 x 4 atau sebanyak 12 tombol,adapun masing - masing kegunaan tombol tersebut adalah sebagai berikut:

1. Tombol angka "0 -9" digunakan untuk memasukan nomer telepon yang dapat digunakan.
2. Tombol "COR" digunakan untuk mengoreksi data.
3. Tombol " MENU" digunakan untuk memilih bagian mana yang datanya akan di edit.
4. Tombol "CAN" digunakan untuk menghapus data atau kenbali ke menu sebelumnya.
5. Tombol " ENT" digunakan untuk memasukkan nomer telpon yang dapat dipergunakan.

Keypad matriks 4 x 4 yang digunakan disini menggunakan sistem scaning kolom dalam melakukan pengenalan tombolnya. Sistem scanning kolom ini dikerjakan oleh oleh mikrokontroller dalam rountime program guna menyederhanakan lagi IC decoder dalam rangkaian ini.

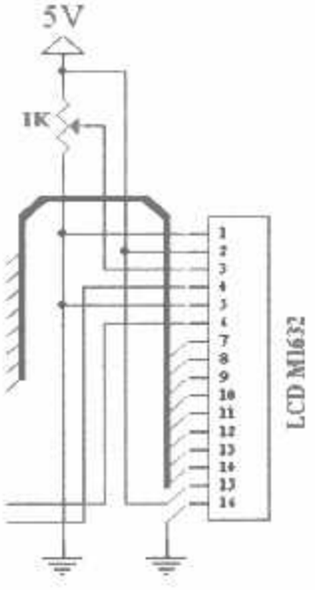


Gambar 3.8 Rangkian Keypad

3.2.7 Perencanaan LCD

Untuk menampilkan nomer telpon yang dapat mengontrol juga menampilkan menu sekaligus menampilkan kontrol yang sedang aktif, digunakan LCD ini dihubungkan dengan bus data mikrokontroller. Dalam hal ini LCD menempati alamat A000H - A0001H.

Untuk Rangkain dari LCD dapat dilihat pada gambar di bawah ini.



Gambar 3.9 Rangkaian LCD
Sumber Data Sheet LCD M1632

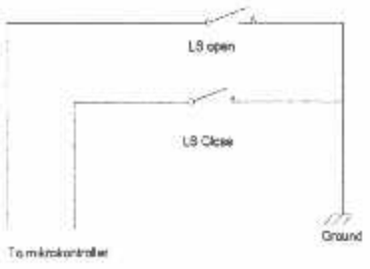
Dengan bantuan perangkat lunak yang dibuat dapat ditampilkan karakter yang diinginkan pada layer tampilan, yaitu dengan mengendalikan sinyal pada E,R/W dan RS. Untuk penggunaan dari masing - masing pin pada LCD dapat dilihat pada tabel berikut:

Tabel 3.1 Fungsi dari Kaki kaki LCD
Sumber : Data Sheet LCD TM162ABC

Nama Pin	Jumblah	I/O	Tujuan	Fungsi
Dim - Dip	8	I/Q	Mikrokontroller	Bus data berisi data yang akan ditampilkan
E (enable)	1	Input	Mikrokomtroller	Sinyal yang mengaktifkan data tulis dan baca
R/W	1	Input	Mikrokomtroller	Sinyal pilih baca dan tulis (0=tulis, 1 =baca)
RS	1	Input	Mikrokomtroller	Sinyal pilih register 0 = intruksi register tulis, 1 = data register.
Vee	1	-	Power Supply	Penyetelan kontras pada tampilan LCD
Vcc	1		Power Supply	Catu daya positif (+5v)
Vss			Power Supply	Terminal ground

3.2.8 Perencanaan Saklar Pembatas (Limit Switch)

Untuk mengatur gerak membuka dan menutup pintu maka digunakan saklar pembatas (Limit Switch). Limit Switch tersebut merupakan sinyal input yang kemudian diolah oleh mikrokontroller untuk menjalankan motor. Pada rangkaian kontrol ini limit switch berfungsi menghentikan, dan memutar balik putaran motor yang digunakan untuk simulasi pintu pagar.



Gambar 3.10 Limit Switch

3.2.9 Perencanaan Relay

Relay adalah suatu perangkat switch (saklar) yang dioperasikan oleh gaya elektromagnetik yang dihasilkan oleh kumparan yang berada didalamnya. Alat ini dalam tugas akhir ini berfungsi sebagai saklar untuk menjalankan motor, juga berfungsi sebagai pengunci pintu.

Dalam tugas akhir ini relay yang digunakan yaitu relay dua induk empat anak (DPDT).

3.2 Perencanaan Perangkat Lunak (software)

Adapun langkah - langkah pembuatan daripada program ini adalah sebagai berikut:

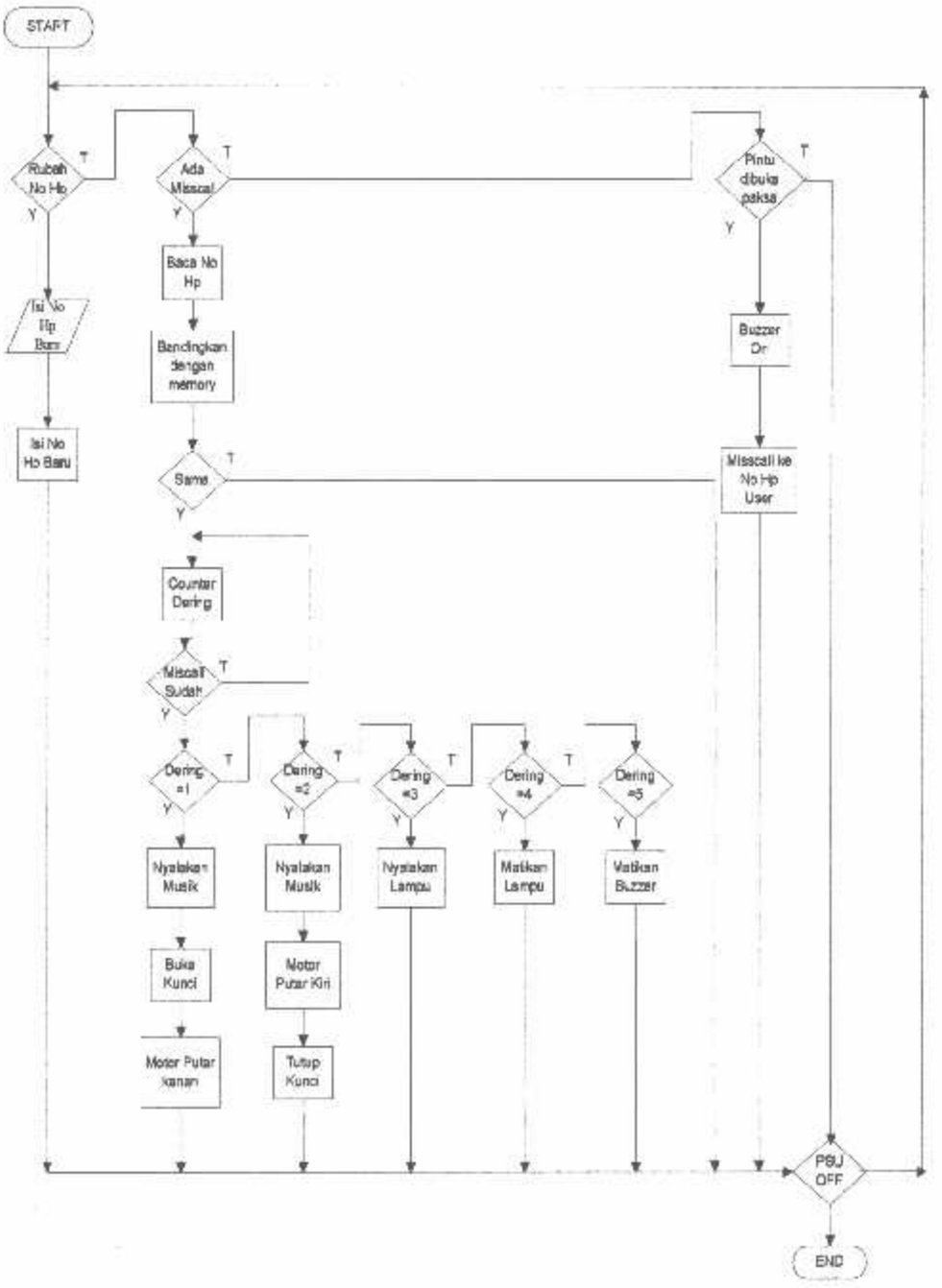
- Membuat diagram alir (Flowchart) dari program yang akan dibuat.
- Mengubah diagram aliran tersebut kedalam bahasa pemograman.
- Mengkomplisasikan program yang telah dibuat kedalam memori sampai menghasilkan program yang paling sesuai.
- Kemudian memasukan program yang telah selesai, dan sistem akan bekerja dengan baik jika perancangan perangkat lunak (software) sesuai dengan perangkat keras (hardware) yang mendukung.

Sedangkan urutan kerja dari keseluruhan sistem adalah sebagai berikut :

- Mikrokontroller di nyalakan.
- Kirim comand ke hp dengan AT+Clip = 1 yang berfungsi ketika ada telepon masuk maka Hp dapat memberikan data No Hp ke mikrokontroller.
- Tulis nama, nim dan fakultas.
- Setelah itu Program menunggu 3 kemungkinan yang dapat diakses yaitu: (Rubah no hp .Ada telpon masuk. Pintu di buka pakasa.) jika tidak ada akan memunculkan nama nim dan fakultas.
- Jika kita memilih rubah No Hp, maka kita diberikesempatan mengisi No Hp baru melalui keypad dan disimpan pada memory external.
- Jika ada pintu dibuka paksa maka buzzer akan On dan Memiscall Hp user dengan cara mengambil No Hp yang terdapat pada memory, dan mengirimkan ke Hp melalui comand ATD 08 .... ; dan beberapa detik kemudian di stop

dengan comand ATH.

- Jika ada telp masuk maka Hp akan mengirimkan ring berserta no Hp dan mikrokontroller akan membaca ring tersebut dan membandingkan dengan memori yang disimpan pada memori eksternal.Jika benar,mikrokontroller akan melakukan aktion setelah ring habis(tidak ada ring dalam 3 detik maka dianggap habis.) Kemudian mikrokontroller menghitung jumblah ring.
- Jika ring =1 maka, mikrokontroller mengintruksikan relay untuk bekerja kemudian membunyikan bel. setelah itu membuka kunci pintu pagar dan membuka pagar.
- Jika ring =2 maka, mikrokontroller mengintruksikan relay untuk bekerja kemudian membunyikan bel. setelah itu menutup pintu pagar lalu mengunci nya.
- Jika ring = 3 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian menyalakan lampu.
- Jika ring = 4 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian mematikan lampu.
- Jika ring = 5 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian mematikan buzzer.



Gambar 3 .11 Flowcart rangkaian

BAB IV
PENGUJIAN ALAT

4.1 Pendahuluan

Dalam bab ini membahas pengujian dan analisa sistem yang telah dirancang. Pengujian dilakukan dengan tujuan untuk mengetahui kerja sistem. Pengujian dilakukan dengan melakukan pengukuran tiap blok sistem dan mengamati apakah blok - blok sistem tersebut bekerja sesuai dengan yang diharapkan. Jika terjadi penyimpangan, maka penyebab dari penyimpangan yang terjadi di analisa.

Pengujian juga dilakukan berdasarkan pada masing - masing komponen pendukung dari sistem ini dan dilanjutkan dengan pengujian sistem secara keseluruhan .

4.2 Pengujian Rangkaian Pengontrol Relay (Driver) Motor

Pada pengujian ini rangkaian pengontrol relay (driver) motor ini, motor berfungsi membuka dan menutup pintu pagar seperti yang telah dibahas pada waktu perancangan, maka tahap berikutnya pengujian rangkaian sebagai berikut:

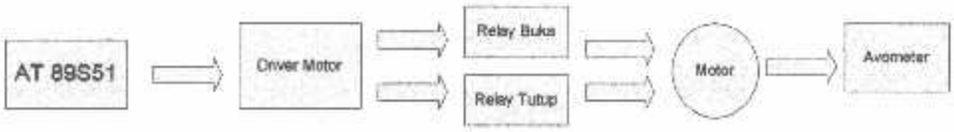
1. Tujuan :

Tujuan dari pengukuran rangkaian ini adalah untuk mengetahui respon perputaran motor akibat data logic yang diberikan pada masing - masing driver dalam suatu motor.

2. Alat dan bahan :

- Avometer
- Sumber tegangan 12 volt
- Blok lengkap pengujian rangkaian driver motor.

3. Blok pengujian rangkain driver motor :



Gambar 4.1 Blok pengujian driver motor

4. Data hasil pengujian:

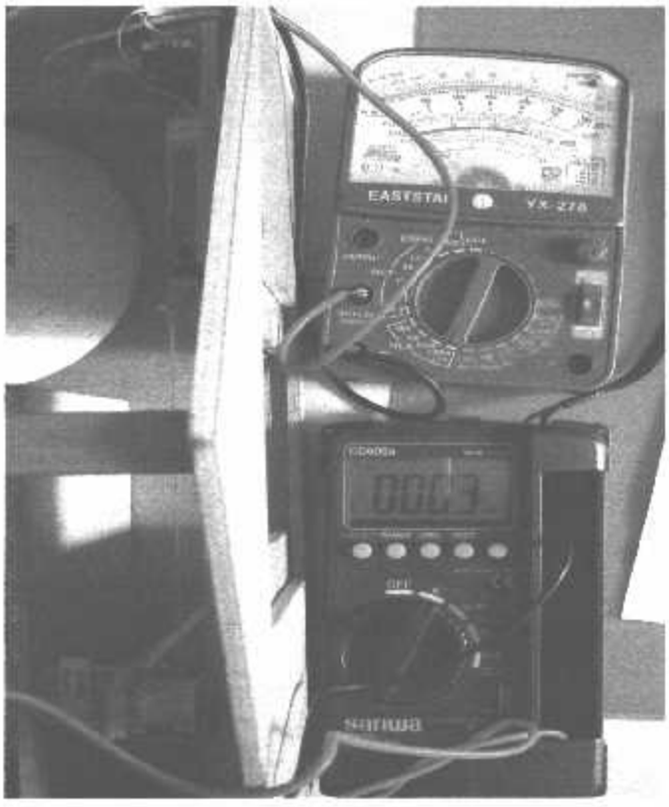
Tabel 4.1 Data Hasil Pengukuran Driver Motor

No	V in (Logic) "A"	V in (Logic) "B"	Vmeter (Vmotor)	KETERANGAN
1	"0"	"0"	0 volt	Motor diam
2	"0"	"1"	0,3 volt	Motor putar kanan
3	"1"	"0"	0,3 volt	Motor putar kiri
4	"1"	"1"	0 volt	Motor diam

5. Analisa :

Tegangan output atau data output yang dikeluarkan oleh IC AT89S51 terhadap driver, bila kedua input driver memiliki data logic yang sama maka motor tidak berbutar, tegangan pada motor nol Volt. Data yang sama tersebut akan membuat kedua transistor yang sejenis dan diletakkan sedimikian rupa akan mati keduanya dan sumber motor (ground keduanya) atau sumber motor (positif keduanya), sehingga motor tidak memiliki beda potensial pada ujung sumbernya.

Dan bila kedua input driver memiliki data logic yang tidak sama maka motor akan berputar, tegangan pada motor 12 Volt. Perbedaan data logic yang tidak sama tersebut akan menimbulkan beda potensial dan mengakibatkan motor akan berputar ke kiri atau kekanan .



Gambar 4.2 Pengujian driver motor

4.3 Pengujian Driver Relay Selenoid dan lampu

Pada pengujian ini basis driver akan diberi data logic "1". Pengujian ini tidak jauh beda dengan saat pengujian driver motor. Langkah pertama yang dilakukan adalah :

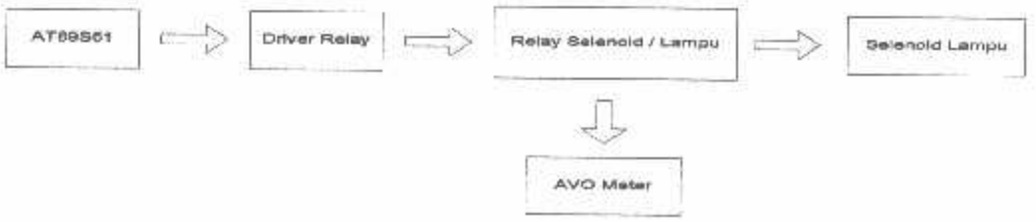
1. Tujuan :

Pengujian driver ini bertujuan untuk mengetahui kinerja dan kecepatan reaksi relay akibat perubahan data logic yang diberikan padanya.

2. Alat dan bahan :

- Avometer analog
- Sumber tegangan 12 V
- Blok pengujian rangkaian driver relay

3. Blok pengujian rangkain driver relay :



Gambar 4.3 Blok Pengujian Rangkaian Driver Relay

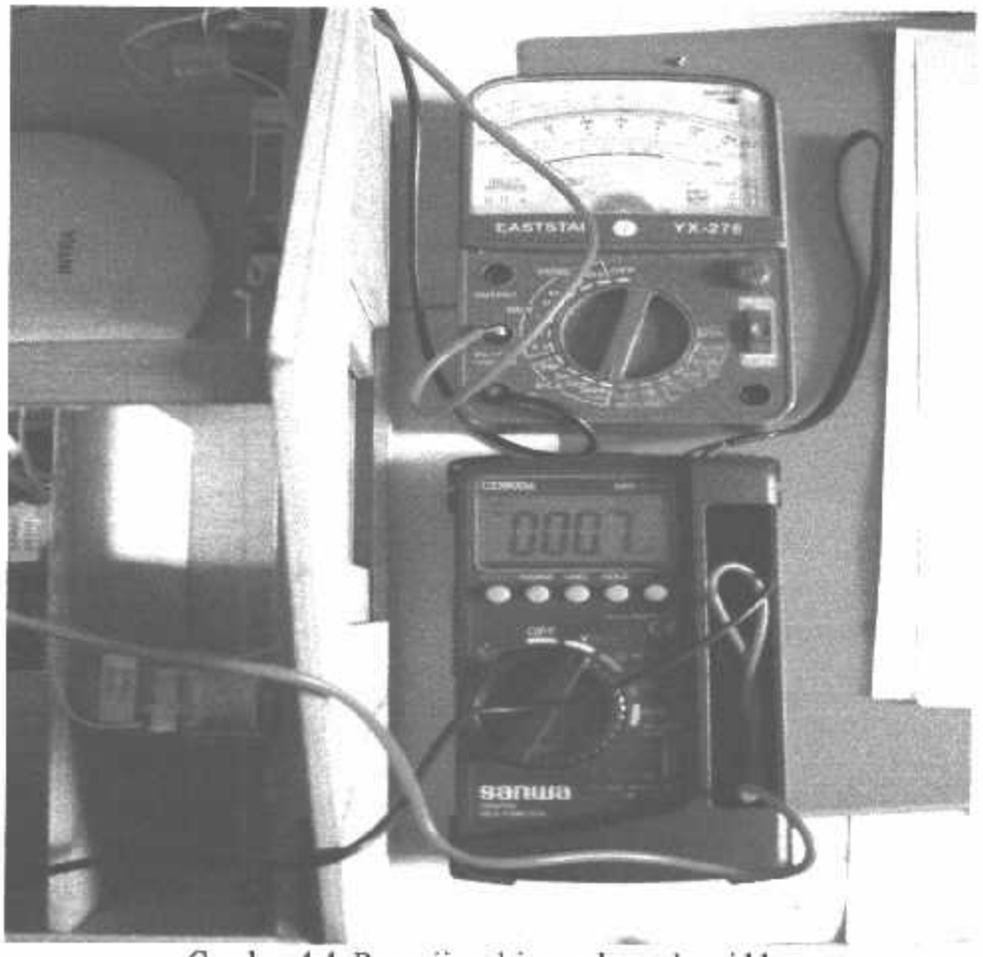
4. Data hasil pengujian :

Tabel 4.2 Data hasil pengujian Driver Relay

NO	Data Logic	Vrelay (Volt)	Kondisi Relay	Keterangan Lampu
1	"0"	0	Off	Padam
2	"1"	0,7	On	Menyala
3	"1"	0	Off	Padam

5. Analisa :

Pada rangkaian ini dapat dianalisa saat data logic "1" diberikan pada driver, mengakibatkan transistor bekerja dan relay mendapat sumber tegangan, sehingga lampu akan menyala dan sebaliknya bila diberi logic "0" / don't care "x" lampu padam.



Gambar 4.4 Pengujian driver relay solenoid lampu

4.4 Pengujian Rangkaian Tampilan LCD

1. Tujuan

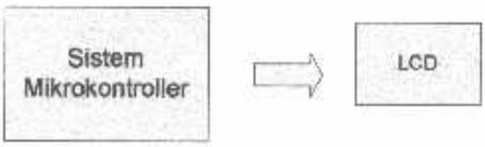
Untuk mengetahui kemampuan rangkain tampilan yang sudah dibuat apakah dapat mendukung sistem yang direncanakan untuk menampilkan data pada LCD.

2. Peralatan yang dibutuhkan

- Power Suplay 5 volt
- Sistem Mikrokontroller dan LCD M1632

3. Prosedur Pengujian

- Menyusun rangkain
- Menjalankan program untuk menampilkan tulisan "Nama Nim Fakultas dan Nomer Hp yang menghubungi" ke LCD programnya terlampir
- Mengamati keluaran pada LCD programnya terlampir.
- Mengamati keluaran pada LCD



Gambar 4.5 Diagram Blok Pengujian Rangkain LCD

4. Analisa

Dari hasil pengujian dapat dilihat bahwa rangkain tamnpilan dapat bekerja dengan baik.

Gambar 4.6 Pengujian LCD

4.5 Pengujian Motor Dc

1. Tujuan

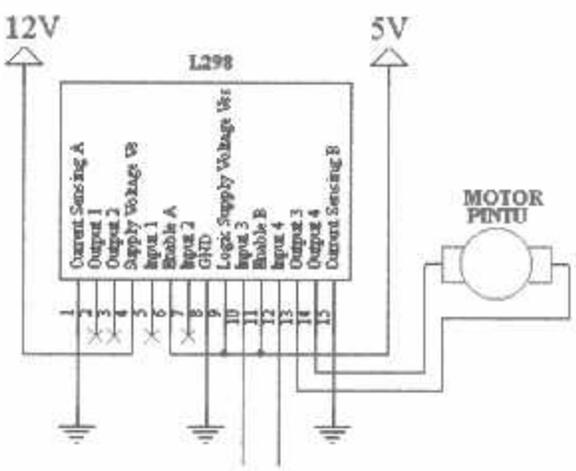
Pengujian motor DC ini adalah apakah motor yang akan digunakan dapat berjalan sesuai dengan yang dikehendaki atau tidak.

2. Peralatan yang digunakan

- Minimum system
- Power suplay
- Avometer digital

3. Prosedur pengujian

- Hubungkan rangkain dengan power supplay
- Hubungkan rangkain Motor DC dan Driver seperti gambar berikut



Gambar 4.7 pengujian rangkaian motor Dc

Rangkain Motor DC yang telah diuji adalah apabila Vin berlogika High (266,6 v) dan transistor akan bekerja sebagai saklar off maka saklar Relay akan memberikan tegangan pada rangkaian Motor Dc sesuai dengan tegangan yang diberikan sehingga motor Dc akan berputar kanan atau kiri.

Tabel 4.3 Data Hasil Pengujian Motor DC

No	INPUT	ARAH PUTAR
1	00	Motor Diam
2	01	Putar Kanan
3	10	Putar Kiri

Gamabar 4.8 Pengujian motor DC

4.6 Pengujian Rangkian Input Keypad

1. Tujuan

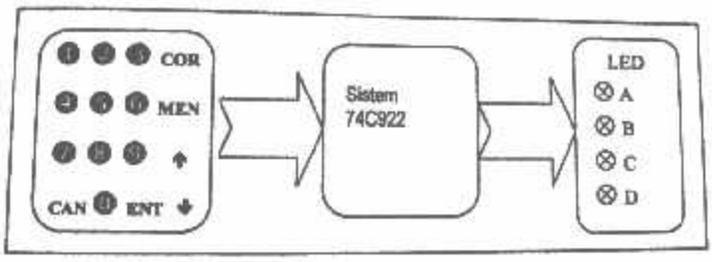
Pengujian ini bertujuan untuk mengetahui konfigurasi logika keluaran dari unit papan tombol pada saat tombol ditekan. Dalam pengujian ini keluaran yang diamati adalah proses *scaning* yang terjadi pada lajur baris dan kolom. Lajur baris merupakan bagian output sedangkan lajur kolom merupakan bagian input. Untuk mengetahui kebenaran rangkain keypad yang telah dibuat maka keluaran dari rangkain keypad ini akan ditampilkan ke port 3 MCU 89S51.

2. Peralatan yang digunakan

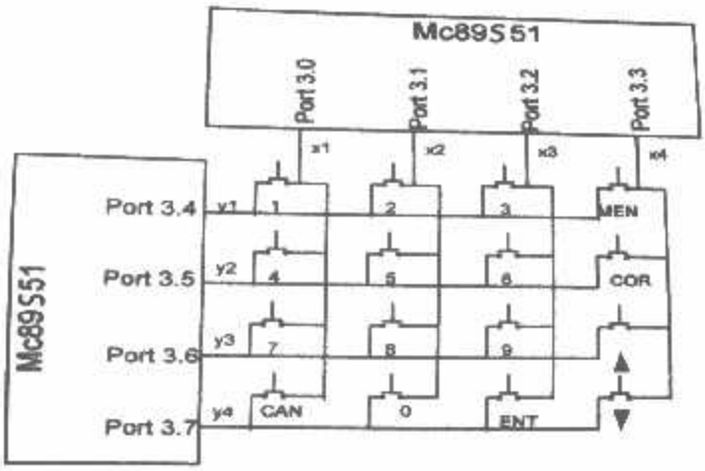
   - Keypad
   - Minimum System
   - LED peraga
   - Catu daya 5 VDC

3. Pelaksanaan pengujian

   - Merangkai alat
   - Menekan tombol – tombol pada *keypad* dan mengamati serta mencatat keluaran yang ditampilkan LED pereaga.



Gambar 4.9 Blok Diagram Pengujian *keypad*



Gambar 4.10 rangkaian *keypad*

4. Hasil dan Analisa Pengujian

Data dikirim melalui *keypad* matrik baris dan kolom. Persambungan antara baris dan kolom akan menghasilkan keluaran data tertentu. Data keluaran IC 74C922 terdiri atas dua bagian, yaitu bagian data dan statu. Bagian data menghasilkan data dari kombinasi hubungan antara baris dan kolom sedangkan bagian status akan mengidentifikasikan adanya hubungan antara baris dan kolom yang *valid*. Keluaran masing – masing bagian adalah aktif tinggi.

Tabel 4-4

Hasil pengujian pengkode *keypad*

No	Baris	Kolom	LED				DA
			A	B	C	D	
1	Y0	X0	0	0	0	0	1
2	Y0	X1	0	0	0	1	1
3	Y0	X2	0	0	1	0	1
4	Y0	X3	0	0	1	1	1
5	Y1	X0	0	1	0	0	1
6	Y1	X1	0	1	0	1	1
7	Y1	X2	0	1	1	0	1
8	Y1	X3	0	1	1	1	1
9	Y2	X0	1	0	0	0	1
10	Y2	X1	1	0	0	1	1
11	Y2	X2	1	0	1	0	1
12	Y2	X3	1	0	1	1	1
13	Y3	X0	1	1	0	0	1
14	Y3	X1	1	1	0	1	1
15	Y3	X2	1	1	1	0	1
16	Y3	X3	1	1	1	1	1

Tabel 4.5 menunujukan hasil pengujian *keypad* dimana setiap penekanan tombol menyebabkan LED peraga menampilkan data *biner* sesuai dengan tombol yang ditekan. Hal ini sesuai dengan perancangan yang telah dibuat.

4.7 Pengujian Fungsi Limit Switch

1. Tujuan

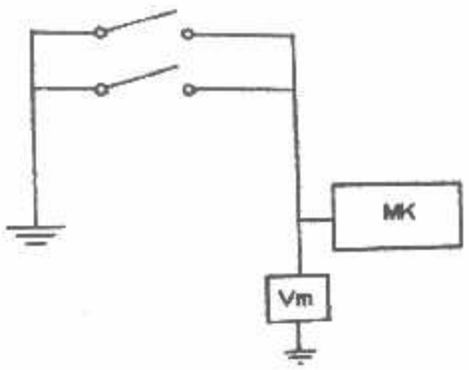
Tujuan dari pengujian ini adalah untuk mengetahui seberapa besar fungsi perencanaan *limit switch* dalm pengamanan proses kerja dari motor dc, dalam hal ini yang kita lihat pengaruhnya terhadap motor dc yang digunakan pada system ini. Pada system buka tutup pintu geser menggunakan 2 buah *limit switch.* Disini *limit switch* bersifat *normali open* dan terletak pada sisi kanan dan kiri pintu adapun cara pengujiannya dengan menjalankan alat setelah itu kita amati fungsi dari tiap – tiap *limit switch* yaitu dengan menekan satu persatu *limit switch.*

2. Peralatan yang digunakan

- Rangkain *Limit Switch* yang akan diuji.
- 2 buah *Limit Switch.*
- Catu Daya 5V
- Multimeter

3. Prosedur Pengujian

- Merangkai rangkain *Limit Switch* seperti pada gambar dibawah ini :



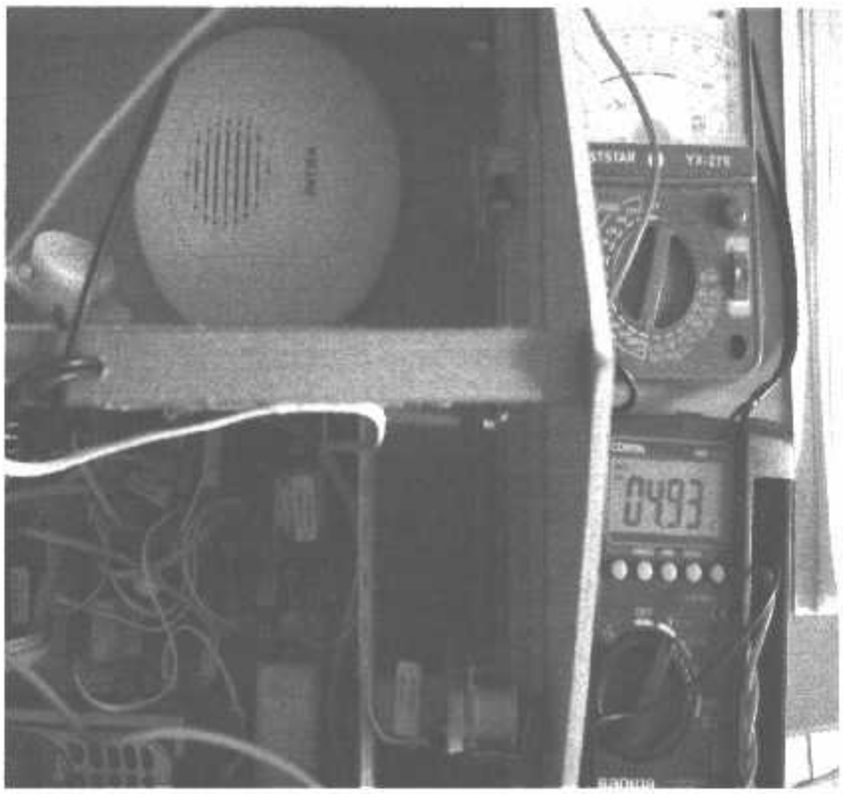
Gambar 4.11 Pengujian Rangkain *Limit Switch*

- Mengamati hasil pengujian *Limit Switch.*

Dan hasil pengujian tersebut diperoleh data seperti pada table berikut :

Tabel 4.5 Data Hasil Pengujian *Limit Switch*

Kondisi	V Limit	V Limit Ukur	Logic
Terbuka	0 volt	0,25 volt	0
Tertutup	5 volt	04,93 vot	1



Gambar 4.12 Pengujian Limit switch

4.8 Pengujian Driver Buzzer

1. Tujuan

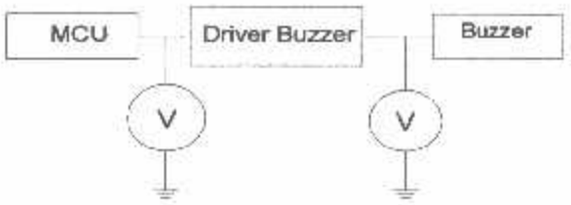
Pada pengujian driver buzzer ini bertujuan untuk mengetahui buzzer dapat bekerja sesuai dengan perintah atau tidak.

2. Peralatan yang digunakan

   - Rangkain driver buzzer.
   - 2 buah Voltmeter.
   - Catu daya 5V.

3. Prosedur Pengujian

   - Merangkai rangkaian *Limit Switch* seperti pada gambar dibawah ini :



Gambar 4.13 Diagram Blok Driver Buzzer

   - Mengamati hasil pengujian driver busser

Untuk pengujian driver buzzer ini kita menggunakan range 0,4 – 0,6 untuk logic "0" dan 11,3 – 12,0 untuk logic "1".

Hasil pengujian rangkain busser berikut pada table dibawah ini :

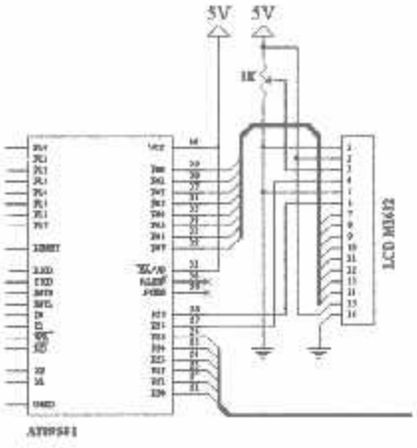
Tabel4.6 Pengujian Rangkain Busser

Logic	**V Logic**	**V Driver**	**V Busser**	**Ket**
0	00,11	015,6	01,62	ON
1	04,92	13,21	13,32	OFF

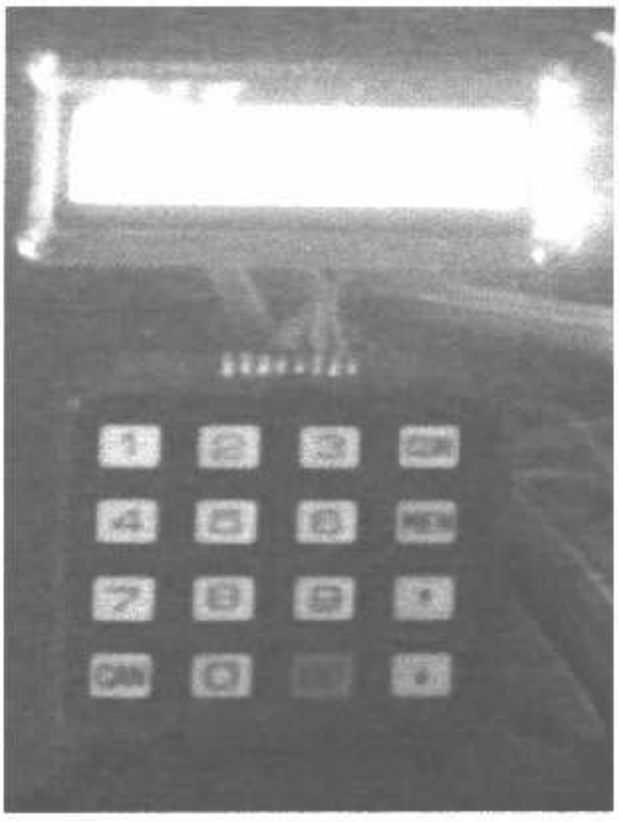
Gambar 4.14 Pengujian driver Busser

4.9 Pengujian Rangkaian LCD (Liquid Cyristal Display)

Untuk mengetahui LCD dapat berfungsi dengan baik maka diperlukan pengujian .Pada pengujian ini MCU ini di program untuk mengontrol LCD agar dapat menampilkan karakter " ITN MALANG " pada baris 1 dan menampilkan karakter " ENERGI LSTRIK DIII" Pada baris 2. Adapun hubungan LCD dengan MCU adalah sebagai berikut.



Gambar 4.15 Gambar rangkaian LCD dengan MCU



Gambar 4.16 pengujian LCD

4.10 Pengujian Hp

1. Tujuan

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

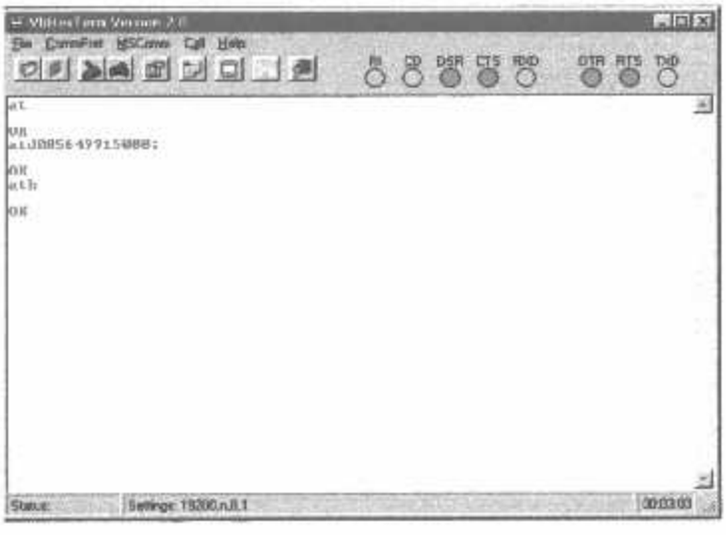
2. Peralatan yang digunakan

- Kabel data RS232
- Hp SIEMENT C45
- Sofware serial tester
- PC

3. Prosedur pengujian

- Hubungkan telepon seluler dengan computer menggunakan kabel data Serial.
- Menjalankan program serial tester
- Melakukan setting port serial pada program serial tester
- Mengetik intruksi +CLIP dan ATH

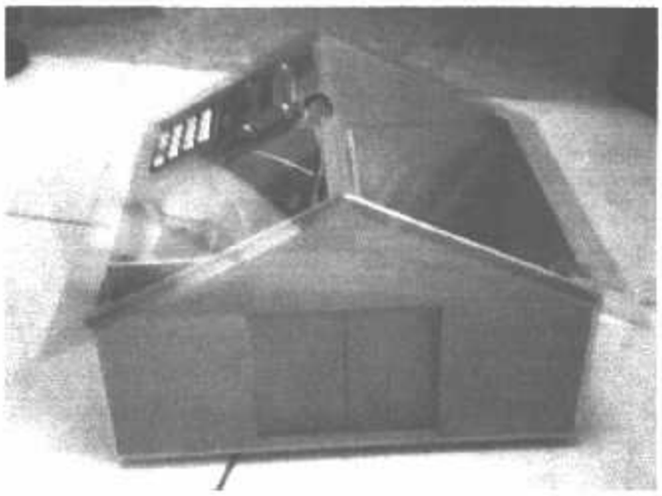
Gambar 4.17 Pengujian Hp Ketika ada Telpon Masuk



Gambar 4.18 Pengujian Miscall Balik Hp

4.11 Pengujian alat keseluruhan

Pengujian terhadap keseluruhan sistem dilakukan pada simulator berupa pintu pagar yang telah dilengkapi dengan sebuah motor girboks dc, pintu rumah yang dilengkapi dengan sebuah solenoid sebagai kunci pintu dan sebuah bola lampu.



Gambar 4.19 Pengujian alat secara keseluruhan

4.11.1 Prosedur pengujian

- Memastikan rangkain pengontrol telah terpasang semua dengan sempurna beserta koponen pendukung lainya.
- Memberi catu daya pada rangkain.
- Memastikan tampilan pada LCD selesai.
- Menseting nomer telpon yang dapat mengontrol sistem.
- Menghubungi handphone pada sistem.
- Mencoba menghubungi hanphone sistem dengan menggunakan nomer telepon lain yang nomernya belum diseting pada sistem.

- Memastikan sistem tidak bekerja bila dihubungi dengan menggunakan nomer telepon lain yang nomernya belum diseting pada sistem.
- Jika ring =1 maka, mikrokontroller mengintruksikan relay untuk bekerja kemudian membunyikan bel. setelah itu membuka kunci pintu pagar dan membuka pagar.
- Jika ring =2 maka, mikrokontroller mengintruksikan relay untuk bekerja kemudian membunyikan bel. setelah itu menutup pintu pagar lalu mengunci nya.
- Jika ring = 3 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian menyalakan lampu.
- Jika ring = 4 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian mematikan lampu.
- Jika ring = 5 maka mikrokontroller mengintruksikan relay untuk bekerja kemudian mematikan buzzer.

4.11.2 Analisis

Dari hasil pengujian yang telah dilakukan dapat disimpulkan sebagai berikut :

- Rangkaian telah berfungsi dengan baik.
- Perangkat lunak yang dibuat telah berfungsi sesuai dengan yang telah direncanakan.
- Sistem pengontrolan dalam pengenalan nomer telpon telah berfungsi dengan baik.

4.11.3 Spesifikasi Alat

Pada simulator pintu pagar ini mempunyai spesifikasi sebagai berikut :

- Tegangan input AC 220-2240V 50/60Hz 15W
- Tegangan Output Motor DC 12 V

BAB V
PENUTUP

Setelah melaksanakan beberapa perencanaan, pembuatan dan pengujian terhadap alat control pintu pagar menggunakan mikrokontroller AT 89C51 yang diaktifkan dengan miscal handphone, maka dapat disimpulkan juga saan yang dianggap perlu sehingga dalam pembuatan berikutnya dapat diperoleh hasil yang lebih baik.

5.1 Kesimpulan

1. Dengn menggunakan metode pengenalan nomer telepon yang masuk, sistem dapat memproteksi dari orang yang tidak berhak mengaktifkan system.
2. Pada pengujian alat tidak ada kendala pergeseran frekuensi yang telah di pakai sebagai pengendali kontrol.
3. Time out untuk membuka pintu pagar adalah 05:29,24 Detik.
4. Time out untuk menutup pintun pagar adalah 05:35,11 Detik.
5. Time out untuk menghidupkan lampu adalah 05:47,07 Detik.
6. Time out untuk mematikan lampu adalah 05:53,09 Deetik.
7. Time out untuk mematikan Buzzer adalah 05:36,06 Detik.
8. Dan waktu yang dibutuhkan untuk membuka maupun menutup pintu pagar adalah 04:19,31 detik.
9. Perencanaan dan pembuatan *hardware* alat pengendali Pintu gerbang dengan Hp via *Misscal* didapatkan ternyata 100% sesuai dengan rancangan.

5.2 Saran

1. Handphone yang digunakan sebaiknya tidak hanya merk Siemens tetapi juga merk lain seperti Nokia, Samsung atau yang lain.
2. Telpon yang digunakan pada alat kontrol juga seharusnya dapat diganti dengan telepon rumah biasa.
3. Untuk laporan apabila pintu dibuka paksa setidaknya dapat berupa sms atau suara jadi tidak hanya miscall balik saja.
4. Ada baiknya terdapat saklar untuk mengubah menjadi mode manual dengan password apabila pemilik rumah lupa membawa handphone.

DAFTAR PUSTAKA

Atmel, 1999, **Data Sheet Microcontroller AT89S51.**

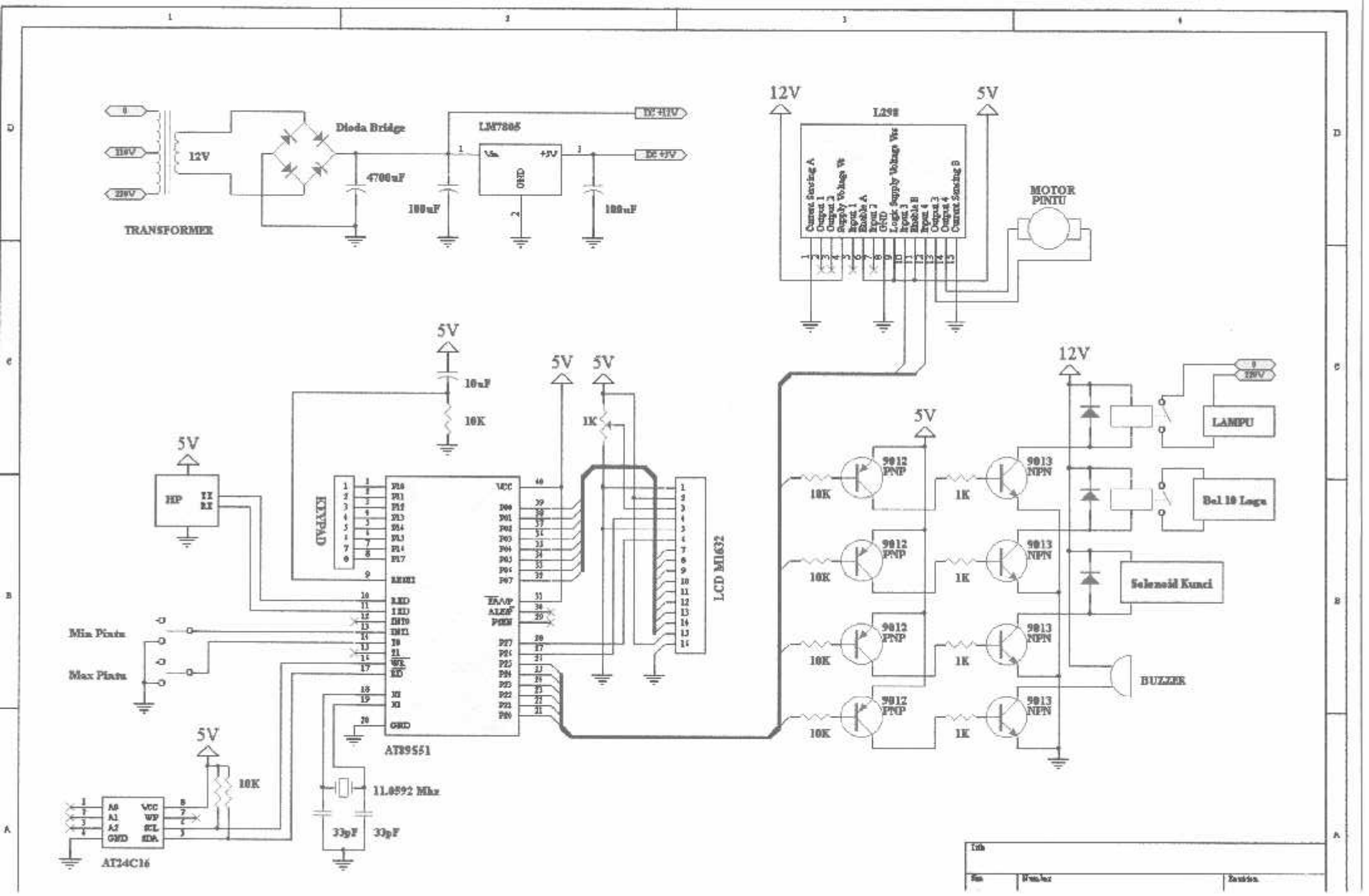
Integred Silicon Solution, Inc, 2000, **Data Sheet EEPROM 24C16.**

Maxim, 2001, **Data Sheet MAX 232.**

Wasito S. 1994, **Vedemekum Elektronika,** PT. Gramedia Pustaka Utama, Jakarta.

Mavino, Hanapi Gunawan, 1979, **Prinsip – prinsip Elektronika,** Penerbit Erlangga Jakarta.

LAMPIRAN


TRANSFORMER
12V
Dioda Bridge
4700uF
LM7805
100uF
100uF
L298
MOTOR PINTU
10uF
10K
1K
KEYPAD
LCD M1632
AT89S51
Min Pinta
Max Pinta
11.0592 Mhz
33pF
33pF
AT24C16
9012 PNP
9013 NPN
LAMPU
Bel 10 Lagu
Selenoid Kunci
BUZZER

LISTNING PROGRAM

```
org   00h
ljmp  init

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

;
Lmtt   Bit P2.0      ; limit tutup
Lmbk   Bit P2.1      ; limit buka
Bell   Bit P2.3      ; bell 10 lagu
Pntt   Bit P2.4      ; pintu tutup
Pnbk   Bit P2.5      ; pintu buka
Rest   Bit P2.6
Enbl   Bit P2.7

Lmpu   Bit P3.3      ; lampu
Kcpn   Bit P3.4      ; kunci pintu
Buzr   Bit P3.5      ; buzzer
ISDA   Bit P3.6      ; I2C data
ISCL   Bit P3.7      ; I2C clock
```

```
Sttp   Bit 20h.0        ; status pintu
Stmc   Bit 20h.1        ; status misscall

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                 ;/

        Char    Equ 50h
        Tmo0    Equ 51h
        Tmo1    Equ 52h
        Tmo2    Equ 53h
        Tmo3    Equ 54h
        Cntr    Equ 55h
        Comd    Equ 56h                 ; command
        Dly0    Equ 57h
        Dly1    Equ 58h
        Dly2    Equ 59h
        Dly3    Equ 5Ah
;
init:   lcall   lcd_in
        mov     DPTR,#tpinls
        lcall   line1
        mov     Char,#16
        lcall   tulis
        mov     DPTR,#tphnph
        lcall   line2
        mov     Char,#16
        lcall   tulis
```

```
        mov   Dly3,#10
        lcall delay3
        lcall srl_in                ; inisialisasi serial
        lcall noecho                ; komunikasi no-echo
        lcall callpr                ; komunikasi calling party
        lcall rstcmd                ; reset command
        mov   Comd,#0               ; action 0
        clr   Stmc                  ; reset status misscall
cekdpn: jb    Lmtt,ckdpn            ;\
        clr   Sttp          ; | cek kondisi
        ljmp  mulai                 ; | pintu
ckdpn:  setb  Sttp                  ;/
;
mulai:  mov   DPTR,#tpnama
        lcall line1
        mov   Char,#16
        lcall tulis
        mov   DPTR,#tpnims
        lcall line2
        mov   Char,#16
        lcall tulis
        lcall delay2
        mov   DPTR,#tpjurs
        lcall line1
        mov   Char,#16
        lcall tulis
        mov   DPTR,#tpuniv
```

```
        lcall  line2
        mov    Char,#16
        lcall  tulis
        lcall  delay2
    ;
    loop0: mov   DPTR,#tptgms
        lcall  line1
        mov    Char,#16
        lcall  tulis
        mov    DPTR,#tpstrp
        lcall  line2
        mov    Char,#16
        lcall  tulis
    ;
    loop1: lcall  tgresp
        djnz   Dly0,loop1
    ;
    loop2: djnz   Tmo2,loop3
        djnz   Tmo3,loop3
        ljmp   action
    loop3: lcall  tgresp
        jb     Sttp,loop4
        jnb    Lmtt,loop4
        jb     Stmc,loop4
        setb   Stmc
        clr    Buzr
        lcall  msscll
```

```
loop4:  ljmp   loop1
;
tgresp: cjne   R7,#0FFh,grsp0          ; tunggu respon
        ljmp   tgrsp1
tgrsp0: lcall  ceksrl
tgrsp1: ret
;
isinhp: lcall  lcdclr
        mov    R1,#1
isnhp0: cjne   R1,#1,isnhp1
        mov    DPTR,#tpnhp0
        lcall  line1
        mov    Char,#16
        lcall  tulis
        mov    A,#0A0h          ;\
        lcall  adrtx            ; | baca memory
        mov    A,#000           ; | no HP 0
        lcall  dtatx            ;/
        ljmp   isnhp4
isnhp1: cjne   R1,#2,isnhp2
        mov    DPTR,#tpnhp1
        lcall  line1
        mov    Char,#16
        lcall  tulis
        mov    A,#0A0h          ;\
        lcall  adrtx            ; | baca memory
        mov    A,#050           ; | no HP 1
```

```
        lcall  dtatx                ;/
        ljmp   isnhp4
isnhp2: cjne   R1,#3,isnhp3
        mov    DPTR,#tpnhp2
        lcall  line1
        mov    Char,#16
        lcall  tulis
        mov    A,#0A0h              ;\
        lcall  adrtx                ; | baca memory
        mov    A,#100               ; | no HP 2
        lcall  dtatx                ;/
        ljmp   isnhp4
isnhp3: cjne   R1,#4,isnhp4
        mov    DPTR,#tpnhp3
        lcall  line1
        mov    Char,#16
        lcall  tulis
        mov    A,#0A0h              ;\
        lcall  adrtx                ; | baca memory
        mov    A,#160               ; | no HP 3
        lcall  dtatx                ;/
isnhp4: lcall  rdmnhp
        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
isnhp5: lcall  scnkpd
        cjne   R0,#11,isnhp6
        mov    SP,#07h
        ljmp   loop0
isrhp6: cjne   R0,#12,isnhp7
        ljmp   isnhpB
isnhp7: cjne   R0,#15,isnhp9
        dec    R1
        cjne   R1,#0,isnhp8
        mov    R1,#1
isnhp8: ljmp   isnhp0
```

```
isnhp9: cjne   R0,#16,isnhp5
        inc    R1
        cjne   R1,#5,isnhpA
        mov    R1,#4
isnhpA: ljmp   isnhp0
;
isnhpB: mov    DPTR,#kosong
        lcall  line2
        mov    Char,#16
        lcall  tulis
        lcall  rsnohp
        lcall  tg_lps
        mov    DPTR,#angka
        mov    P0,#0C1h
        lcall  w_ins
        mov    P0,#'+'
        lcall  w_chr
        mov    P0,#'6'
        lcall  w_chr
        mov    P0,#'2'
        lcall  w_chr
        lcall  tginhp
        mov    Nhp0,R0
        mov    A,R0
        lcall  wr_chr
        lcall  tg_lps
        lcall  tginhp
```

```
mov   Nhp1,R0
mov   A,R0
lcall wr_chr
lcall tg_lps
lcall tginhp
mov   Nhp2,R0
mov   A,R0
lcall wr_chr
lcall tg_lps
lcall tginhp
mov   Nhp3,R0
mov   A,R0
lcall wr_chr
lcall tg_lps
lcall tginhp
mov   Nhp4,R0
mov   A,R0
lcall wr_chr
lcall tg_lps
lcall tginhp
mov   Nhp5,R0
mov   A,R0
lcall wr_chr
lcall tg_lps
lcall tginhp
mov   Nhp6,R0
mov   A,R0
```

```
lcall  wr_chr
lcall  tg_lps
lcall  tginhp
mov    Nhp7,R0
mov    A,R0
lcall  wr_chr
lcall  tg_lps
lcall  tginhp
mov    Nhp8,R0
mov    A,R0
lcall  wr_chr
lcall  tg_lps
lcall  tginhp
mov    Nhp9,R0
mov    A,R0
lcall  wr_chr
lcall  tg_lps
lcall  tginhp
mov    NhpA,R0
mov    A,R0
lcall  wr_chr
lcall  tg_lps
lcall  tginhp
mov    NhpB,R0
mov    A,R0
lcall  wr_chr
lcall  tg_lps
```

```
isnhpC: cjne   R1,#1,isnhpD
        mov    A,#0A0h          ; AT24C16 write address
        lcall  adrtx
        mov    A,#000           ; address memory
        lcall  dtatx
        ljmp   isnhpG
isnhpD: cjne   R1,#2,isnhpE
        mov    A,#0A0h          ; AT24C16 write address
        lcall  adrtx
        mov    A,#050           ; address memory
        lcall  dtatx
        ljmp   isnhpG
isnhpE: cjne   R1,#3,isnhpF
        mov    A,#0A0h          ; AT24C16 write address
        lcall  adrtx
        mov    A,#100           ; address memory
        lcall  dtatx
        ljmp   isnhpG
isnhpF: cjne   R1,#4,isnhpG
        mov    A,#0A0h          ; AT24C16 write address
        lcall  adrtx
        mov    A,#160           ; address memory
        lcall  dtatx
isnhpG: lcall  wrmnhp
        ljmp   isnhp0
;
tginhp: lcall  scnkpd
```

```
        lcall   delay0
tinhp0: cjne    R0,#16,tinhp1
        ljmp    tginhp
tinhp1: cjne    R0,#15,tinhp2
        ljmp    tginhp
tinhp2: cjne    R0,#14,tinhp3
        ljmp    tginhp
tinhp3: cjne    R0,#13,tinhp4
        ljmp    tginhp
tinhp4: cjne    R0,#12,tinhp5
        ljmp    isnhpC
tinhp5: cjne    R0,#11,tinhp6
        mov     SP,#07h
        ljmp    loop0
tinhp6: cjne    R0,#10,tinhp7
        ljmp    tginhp
tinhp7: ret
;
rsnohp: mov     Nhp0,#00Fh
        mov     Nhp1,#00Fh
        mov     Nhp2,#00Fh
        mov     Nhp3,#00Fh
        mov     Nhp4,#00Fh
        mov     Nhp5,#00Fh
        mov     Nhp6,#00Fh
        mov     Nhp7,#00Fh
        mov     Nhp8,#00Fh
```

```
        mov   Nhp9,#00Fh
        mov   NhpA,#00Fh
        mov   NhpB,#00Fh
        ret
;
cknhpn: cjne  A,#00Fh,cnhpn          ; cek angka no hp
        mov   A,#16                  ; if F then chr ' '
cnhpn:  ret
;
wrmnhp: mov   A,Nhp0                 ; data memory
        lcall dtatx
        mov   A,Nhp1                 ; data memory
        lcall dtatx
        mov   A,Nhp2                 ; data memory
        lcall dtatx
        mov   A,Nhp3                 ; data memory
        lcall dtatx
        mov   A,Nhp4                 ; data memory
        lcall dtatx
        mov   A,Nhp5                 ; data memory
        lcall dtatx
        mov   A,Nhp6                 ; data memory
        lcall dtatx
        mov   A,Nhp7                 ; data memory
        lcall dtatx
        mov   A,Nhp8                 ; data memory
        lcall dtatx
```

```
        mov   A,Nhp9            ; data memory
        lcall dtatx
        mov   A,NhpA            ; data memory
        lcall dtatx
        mov   A,NhpB            ; data memory
        lcall dtatx
        lcall i2cstp            ; i2c stop
        lcall delay0
        lcall delay0
        lcall delay0
        ret
;
rdmnhp: mov   A,#0A1h           ; AT24C16 read address
        lcall adrtx
        lcall dtarx             ; terima data
        mov   Nhp0,A
        lcall givack            ; beri ack
        lcall dtarx             ; terima data
        mov   Nhp1,A
        lcall givack            ; beri ack
        lcall dtarx             ; terima data
        mov   Nhp2,A
        lcall givack            ; beri ack
        lcall dtarx             ; terima data
        mov   Nhp3,A
        lcall givack            ; beri ack
        lcall dtarx             ; terima data
```

```
mov   Nhp4,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   Nhp5,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   Nhp6,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   Nhp7,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   Nhp8,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   Nhp9,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   NhpA,A
lcall givack          ; beri ack
lcall dtarx           ; terima data
mov   NhpB,A
lcall i2cstp          ; i2c stop
lcall delay0
lcall delay0
lcall delay0
ret
```

```
;
adrtx: lcall  i2cstr              ; send address
       lcall  putbit
       ret
;
dtatx: lcall  putbit              ; send data
       ret
;
dtarx: lcall  getbit              ; get data
       ret
;
putbit: mov   R6,#8
putbt: RLC    A
       mov    ISDA,C
       setb   ISCL
       clr    ISCL
       djnz   R6,putbt
       setb   ISDA
       lcall  getack
       ret
;
getbit: mov   R6,#8
getbt: setb   ISCL
       mov    C,ISDA
       RLC    A
       clr    ISCL
       djnz   R6,getbt
```

```
        setb   ISDA
        ret
;
getack: setb   ISDA             ; tunggu ack
        setb   ISCL             ;\
ackbit: mov    C,ISDA                   ; | D=1, C=1
        jc     ackbit           ; | tunggu D=0, C=0
        clr    ISCL             ;/
        ret
;
givack: clr    ISDA             ; kirim ack
        setb   ISCL             ;\
        clr    ISCL             ; | D=0, C=1, C=0, D=1
        setb   ISDA             ;/
        ret
;
i2cstr: setb   ISCL             ; i2c start
        setb   ISDA             ;\
        clr    ISDA             ; | C=1, D=1, D=0, C=0
        clr    ISCL             ;/
        ret
;
i2cstp: clr    ISDA             ; i2c stop
        setb   ISCL             ;\ D=0, C=1, D=1, C=0
        setb   ISDA             ;/
        clr    ISCL
        lcall  delay0
```

```
        ret
    ;
    srl_in: mov   Dly3,#1                    ;\
        lcall  delay3                        ; |
        mov    TMOD,#21h                     ; |
        mov    TH1,#0FDh                     ; |
        mov    SCON,#50h                     ; |
        mov    A,#80h                        ; | set baudrate
        orl    87h,A                         ; | 19200bps
        setb   TR1                           ; |
        setb   ES                  ; |
        setb   EA                  ; |
        ret                                  ;/
    ;
    kr_ins: clr    A                         ;\
            movc   A,@A+DPTR                 ; |
            lcall  kr_srl                    ; |
            inc    DPTR                      ; | kirim instruksi ke hp
            djnz   Char,kr_ins               ; | sebagai awal pengiriman data
        mov    A,#0Dh                        ; |
            lcall  kr_srl                    ; |
        ret                                  ;/
    ;
    kr_srl: clr    ES                        ;\
        mov    SBUF,A                        ; |
        jnb    TI,$                          ; |
        clr    TI                  ; | kirim serial ke hp
```

```
        setb   ES                ;|
        lcall  delay0
        ret                      ;/
;
bc_fbk: lcall  rstcmd            ;\
bc_fb0: cjne   R7,#0FFh,bc_fb1       ;| tunggu feedback
        ljmp   bc_fb0            ;| data dari hp
bc_fb1: ret                      ;/
;
bc_srl: lcall  rstcmd            ;\
        mov    Tmo0,#10          ;|
        mov    Tmo1,#0           ;| baca serial
bc_sr0: cjne   R7,#0FFh,bc_sr1       ;| tunggu data bukan FF
        djnz   Tmo1,bc_sr0       ;| sebelum time out
        djnz   Tmo0,bc_sr0       ;|
bc_sr1: ret                      ;/
;
rstcmd: mov    R7,#0FFh
        ret
;
noecho: mov    Dly3,#1               ;\
        lcall  delay3            ;|
        mov    DPTR,#smseco      ;| kirim
        mov    Char,#4           ;| instruksi
        lcall  kr_ins            ;| no-echo ke hp
        lcall  bc_fbk            ;|
        ret                      ;/
```

```
;
callpr: mov   Dly3,#1                ;\
        lcall delay3                 ; |
        mov   DPTR,#clprty           ; | kirim
        mov   Char,#9                ; | instruksi
        lcall kr_ins                 ; | calling party ke hp
        lcall bc_fbk                 ; |
        ret                          ;/
;
msscll: mov   DPTR,#tpmscl
        lcall line1
        mov   Char,#16
        lcall tulis
        mov   DPTR,#tpplwt
        lcall line2
        mov   Char,#16
        lcall tulis
;
        mov   A,#0A0h                ;\
        lcall adrtx                  ; | baca memory
        mov   A,#000                 ; | no HP 0
        lcall dtatx                  ;/
        lcall rdmnhp
        mov   A,Nhp0
        cjne  A,#00Fh,mscll0
        ljmp  mscll1
mscll0: lcall dialmc
```

```
;
mscll1: mov   A,#0A0h                ;\
        lcall adrtx                  ; | baca memory
        mov   A,#050                 ; | no HP 1
        lcall dtatx                  ;/
        lcall rdmnhp
        mov   A,Nhp0
        cjne  A,#00Fh,mscll2
        ljmp  mscll3
mscll2: lcall dialmc
;
mscll3: mov   A,#0A0h                ;\
        lcall adrtx                  ; | baca memory
        mov   A,#100                 ; | no HP 2
        lcall dtatx                  ;/
        lcall rdmnhp
        mov   A,Nhp0
        cjne  A,#00Fh,mscll4
        ljmp  mscll5
mscll4: lcall dialmc
;
mscll5: mov   A,#0A0h                ;\
        lcall adrtx                  ; | baca memory
        mov   A,#160                 ; | no HP 3
        lcall dtatx                  ;/
        lcall rdmnhp
        mov   A,Nhp0
```

```
        cjne   A,#00Fh,mscll6
        ljmp   mscll7
mscll6: lcall  dialmc
;
mscll7: mov    SP,#07h
        ljmp   loop0
;
dialmc: mov    DPTR,#angka      ;\
        mov    A,#'A'           ; |
        lcall  kr_srl           ; |
        mov    A,#'T'           ; |
        lcall  kr_srl           ; |
        mov    A,#'D'           ; |
        lcall  kr_srl           ; |
        mov    A,#'0'           ; |
        lcall  kr_srl           ; |
        mov    A,Nhp0           ; |
        movc   A,@A+DPTR        ; |
        lcall  kr_srl           ; |
        mov    A,Nhp1           ; |
        movc   A,@A+DPTR        ; |
        lcall  kr_srl           ; |
        mov    A,Nhp2           ; |
        movc   A,@A+DPTR        ; |
        lcall  kr_srl           ; |
        mov    A,Nhp3           ; |
        movc   A,@A+DPTR        ; |
```

```
        lcall   kr_srl          ;|
        mov     A,Nhp4          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,Nhp5          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,Nhp6          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,Nhp7          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,Nhp8          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,Nhp9          ;|
        movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,NhpA          ;|
        cjne    A,#00Fh,bypss0  ;|
        ljmp    bypss2          ;|
bypss0: movc    A,@A+DPTR       ;|
        lcall   kr_srl          ;|
        mov     A,NhpB          ;|
        cjne    A,#00Fh,bypss1  ;|
        ljmp    bypss2          ;|
```

```
bypss1: movc   A,@A+DPTR          ;|
        lcall  kr_srl             ;|
bypss2: mov    A,#';'             ;|
        lcall  kr_srl             ;|
        mov    A,#0Dh             ;|
        lcall  kr_srl             ;/
        mov    Dly3,#60
        lcall  delay3
reject: mov    A,#'A'             ;\
        lcall  kr_srl             ;|
        mov    A,#'T'             ;|
        lcall  kr_srl             ;| reject
        mov    A,#'H'             ;| misscall
        lcall  kr_srl             ;|
        mov    A,#0Dh             ;|
        lcall  kr_srl             ;/
        mov    Dly3,#10
        lcall  delay3
        ret
;
ceksrl: mov    Cntr,#30           ;\
cksrl0: lcall  bc_srl             ;| baca serial
        cjne   R7,#'6',cksrl1     ;| sampai character '_'
        ljmp   cksrl2             ;|
cksrl1: djnz   Cntr,cksrl0        ;|
        ljmp   gksido             ;/
cksrl2: lcall  bc_srl             ;\
```

```
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,#2              ;|
        lcall  delay3               ;/
;
tl_dta: lcall  line1
        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            ;/
;
        mov    A,Ch00
        lcall  chrdec
        mov    Ch00,A
        mov    A,Ch01
        lcall  chrdec
        mov    Ch01,A
        mov    A,Ch02
        lcall  chrdec
        mov    Ch02,A
        mov    A,Ch03
        lcall  chrdec
        mov    Ch03,A
```

```
        mov   A,Ch04
        lcall chrdec
        mov   Ch04,A
        mov   A,Ch05
        lcall chrdec
        mov   Ch05,A
        mov   A,Ch06
        lcall chrdec
        mov   Ch06,A
        mov   A,Ch07
        lcall chrdec
        mov   Ch07,A
        mov   A,Ch08
        lcall chrdec
        mov   Ch08,A
        mov   A,Ch09
        lcall chrdec
        mov   Ch09,A
        mov   A,Ch0A
        lcall chrdec
        mov   Ch0A,A
        mov   A,Ch0B
        lcall chrdec
        mov   Ch0B,A
;
cknohp: mov   R1,#1
        mov   Cntr,#4
```

```
        mov    A,#0A0h          ;\
        lcall  adrtx            ; | baca memory
        mov    A,#000           ; | no HP 0
        lcall  dtatx            ;/
cknhp0: lcall  rdmnhp
        mov    A,Ch00
        mov    B,Nhp0
        clr    C
        subb   A,B
        jnz    cknhp1
        mov    A,Ch01
        mov    B,Nhp1
        clr    C
        subb   A,B
        jnz    cknhp1
        mov    A,Ch02
        mov    B,Nhp2
        clr    C
        subb   A,B
        jnz    cknhp1
        mov    A,Ch03
        mov    B,Nhp3
        clr    C
        subb   A,B
        jnz    cknhp1
        mov    A,Ch04
        mov    B,Nhp4
```

```
clr   C
subb  A,B
jnz   cknhp1
mov   A,Ch05
mov   B,Nhp5
clr   C
subb  A,B
jnz   cknhp1
mov   A,Ch06
mov   B,Nhp6
clr   C
subb  A,B
jnz   cknhp1
mov   A,Ch07
mov   B,Nhp7
clr   C
subb  A,B
jnz   cknhp1
mov   A,Ch08
mov   B,Nhp8
clr   C
subb  A,B
jnz   cknhp1
mov   A,Ch09
mov   B,Nhp9
clr   C
subb  A,B
```

```
        jnz   cknhp1
        inc   Comd
        ljmp  cknhp6
;
cknhp1: inc   R1
        cjne  R1,#2,cknhp2
        mov   A,#0A0h          ;\
        lcall adrtx            ; | baca memory
        mov   A,#050           ; | no HP 1
        lcall dtatx            ;/
        ljmp  cknhp4
cknhp2: cjne  R1,#3,cknhp3
        mov   A,#0A0h          ;\
        lcall adrtx            ; | baca memory
        mov   A,#100           ; | no HP 2
        lcall dtatx            ;/
        ljmp  cknhp4
cknhp3: cjne  R1,#4,cknhp4
        mov   A,#0A0h          ;\
        lcall adrtx            ; | baca memory
        mov   A,#160           ; | no HP 3
        lcall dtatx            ;/
cknhp4: djnz  Cntr,cknhp5
        ljmp  cknhp6
cknhp5: ljmp  cknhp0
;
cknhp6: mov   DPTR,#angka
```

```
        lcall   line2
        mov     A,Comd
        lcall   nilai
;
        mov     Dly3,#10        ;\
        lcall   delay3                  ; |
gksido: lcall   rstcmd                  ; |
        mov     Tmo2,#0                 ; | set time out
        mov     Tmo3,#10                ; |
        mov     SP,#07h                 ; |
        ljmp    loop0                   ;/
;
chrdec: mov     B,#030h                         ; character to decimal
        clr     C
        subb    A,B
        ret
;
action: mov     A,Comd
        cjne    A,#0,actio0
        ljmp    actio6
actio0: cjne    A,#1,actio1
        clr     Bell
        mov     Dly3,#2
        lcall   delay3
        setb    Bell
        mov     Dly3,#1
        lcall   delay3
```

```
        lcall   pntbka
        ljmp    actio5
actio1: cjne    A,#2,actio2
        clr     Bell
        mov     Dly3,#2
        lcall   delay3
        setb    Bell
        mov     Dly3,#1
        lcall   delay3
        lcall   pnttip
        ljmp    actio5
actio2: cjne    A,#3,actio3
        clr     Lmpu
        ljmp    actio5
actio3: cjne    A,#4,actio4
        setb    Lmpu
        ljmp    actio5
actio4: cjne    A,#5,actio5
        setb    Buzr
actio5: mov     Comd,#0
        mov     SP,#07h
actio6: ljmp    loop0
;
pntbka: clr     Kcpn                    ; pintu buka
        mov     Dly3,#1
        lcall   delay3
pnbka:  clr     Pnbk
```

```
        setb   Pntt
        lcall  delay0
        lcall  delay0
        setb   Pnbk
        setb   Pntt
        lcall  delay0
        jb     Lmbk,pnbka
        setb   Pnbk
        setb   Pntt
        setb   Kcpn
        setb   Sttp
        ret
;
pntttp: setb   Pnbk                    ; pintu tutup
        clr    Pntt
        lcall  delay0
        lcall  delay0
        setb   Pnbk
        setb   Pntt
        lcall  delay0
        jb     Lmtt,pntttp
        setb   Pnbk
        setb   Pntt
        clr    Sttp
        clr    Stmc
        ret
;
```

```
nilai: mov   B,#100
       div   AB
       lcall wr_chr
       mov   A,B
       mov   B,#10
       div   AB
       lcall wr_chr
       mov   A,B
       lcall wr_chr
       ret
;
line1: mov   P0,#080h
       lcall w_ins
       ret
;
line2: mov   P0,#0C0h
       lcall w_ins
       ret
;
tulis: clr   A
       lcall wr_chr
       inc   DPTR
       djnz  Char,tulis
       ret
;
wr_chr: movc  A,@A+DPTR
       mov   P0,A
```

```
        lcall  w_chr
        ret
;
w_ins:  clr    Enbl
        clr    Rest
        setb   Enbl
        clr    Enbl
        lcall  delay0
        ret
;
w_chr:  clr    Enbl
        setb   Rest
        setb   Enbl
        clr    Enbl
        lcall  delay0
        ret
;
lcd_in: 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
;
lcdclr: mov   P0,#01h                ; Display Clear
        lcall w_ins
        lcall delay0
        lcall delay0
        lcall delay0
        ret
;
scnkpd: mov   R0,#10
        lcall delay0
col1:   mov   P1,#11111110b
        mov   A,P1
c1b1:   cjne  A,#11101110b,c1b2
        mov   R0,#1
c1b2:   cjne  A,#11011110b,c1b3
        mov   R0,#2
c1b3:   cjne  A,#10111110b,c1b4
        mov   R0,#3
c1b4:   cjne  A,#01111110b,col2
        mov   R0,#13
;
col2:   mov   P1,#11111101b
        mov   A,P1
c2b1:   cjne  A,#11101101b,c2b2
```

```
        mov   R0,#4
c2b2:   cjne  A,#11011101b,c2b3
        mov   R0,#5
c2b3:   cjne  A,#10111101b,c2b4
        mov   R0,#6
c2b4:   cjne  A,#01111101b,col3
        mov   R0,#14
;
col3:   mov   P1,#11111011b
        mov   A,P1
c3b1:   cjne  A,#11101011b,c3b2
        mov   R0,#7
c3b2:   cjne  A,#11011011b,c3b3
        mov   R0,#8
c3b3:   cjne  A,#10111011b,c3b4
        mov   R0,#9
c3b4:   cjne  A,#01111011b,col4
        mov   R0,#15
;
col4:   mov   P1,#11110111b
        mov   A,P1
c4b1:   cjne  A,#11100111b,c4b2
        mov   R0,#11
c4b2:   cjne  A,#11010111b,c4b3
        mov   R0,#0
c4b3:   cjne  A,#10110111b,c4b4
        mov   R0,#12
```

```
c4b4:  cjne   A,#01110111b,back
       mov    R0,#16
back:  ret
;
tg_tkn: lcall  scnkpd
       lcall  delay0
tg_tk0: cjne   R0,#16,tg_tk1
       ljmp   tg_tkn
tg_tk1: cjne   R0,#15,tg_tk2
       ljmp   tg_tkn
tg_tk2: cjne   R0,#14,tg_tk3
       ljmp   tg_tkn
tg_tk3: cjne   R0,#13,tg_tk4
       ljmp   tg_tkn
tg_tk4: cjne   R0,#12,tg_tk5
       ljmp   tg_tkn
tg_tk5: cjne   R0,#11,tg_tk6
       ljmp   tg_tkn
tg_tk6: cjne   R0,#10,tg_tk7
       ljmp   tg_tkn
tg_tk7: ret
;
tg_lps: lcall  scnkpd
       lcall  delay0
       cjne   R0,#10,tg_lps
       ret
;
```

```
delay0: djnz   Dly0,delay0
        ret
;
delay1: lcall  scnkpd
        cjne   R0,#13,dely10
        ljmp   isinhp
dely10: djnz   Dly1,delay1
        ret
;
delay2: mov    Dly2,#20
dely20: lcall  delay1
        djnz   Dly2,dely20
        ret
;
delay3: djnz   Dly0,delay3
        djnz   Dly1,delay3
        djnz   Dly3,delay3
        ret
;
tpnama: DB     '  Handik P  '
tpnims: DB     ' NIM: 0752003 '
tpjurs: DB     '  T. Elektro  '
tpuniv: DB     '  ITN Malang  '
tpinls: DB     ' Inisialisasi '
tphnph: DB     '  Hand Phone  '
tptgms: DB     'Tunggu Miss Call'
tpstrp: DB     ' ---------- '
```

```
tpnhp0: DB    ' Nomor HP 1 '
tpnhp1: DB    ' Nomor HP 2 '
tpnhp2: DB    ' Nomor HP 3 '
tpnhp3: DB    ' Nomor HP 4 '
tpmscl: DB    ' Miss Call '
tpplwt: DB    ' Please wait... '
smseco: DB    'ATE0'
clprty: DB    'AT+CLIP=1'
angka:  DB    '0123456789    '
kosong: DB    '              '
;
      end
```

eatures

- Compatible with MCS-51® Products
- K Bytes of In-System Programmable (ISP) Flash Memory
  - Endurance: 1000 Write/Erase Cycles
- .0V to 5.5V Operating Range
- ully Static Operation: 0 Hz to 33 MHz
- hree-level Program Memory Lock
- 28 x 8-bit Internal RAM
- 2 Programmable I/O Lines
- wo 16-bit Timer/Counters
- ix Interrupt Sources
- ull Duplex UART Serial Channel
- ow-power Idle and Power-down Modes
- nterrupt Recovery from Power-down Mode
- Watchdog Timer
- ual Data Pointer
- ower-off Flag
- ast Programming Time
- lexible ISP Programming (Byte and Page Mode)

scription

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

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

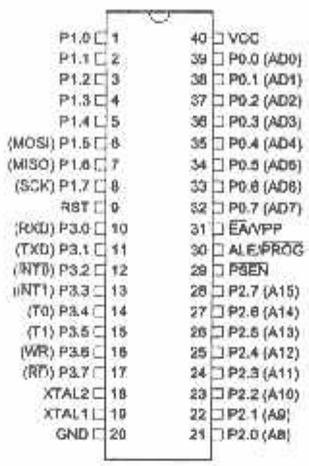
8-bit Microcontroller with 4K Bytes In-System Programmable Flash

AT89S51

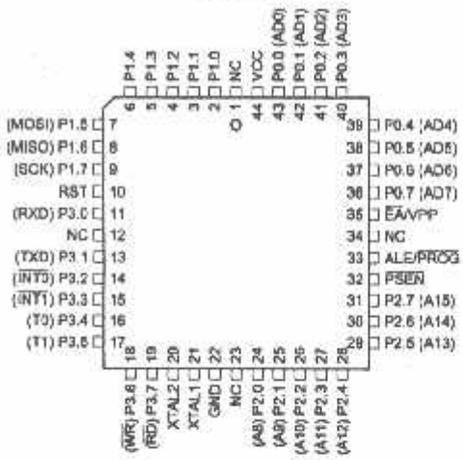
Rev. 2487A–10/01

n Configurations

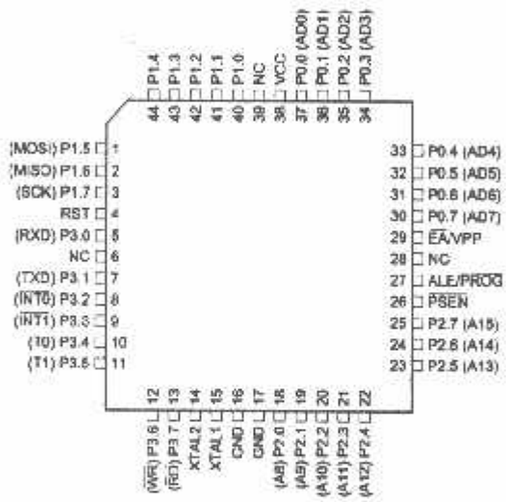
PDIP



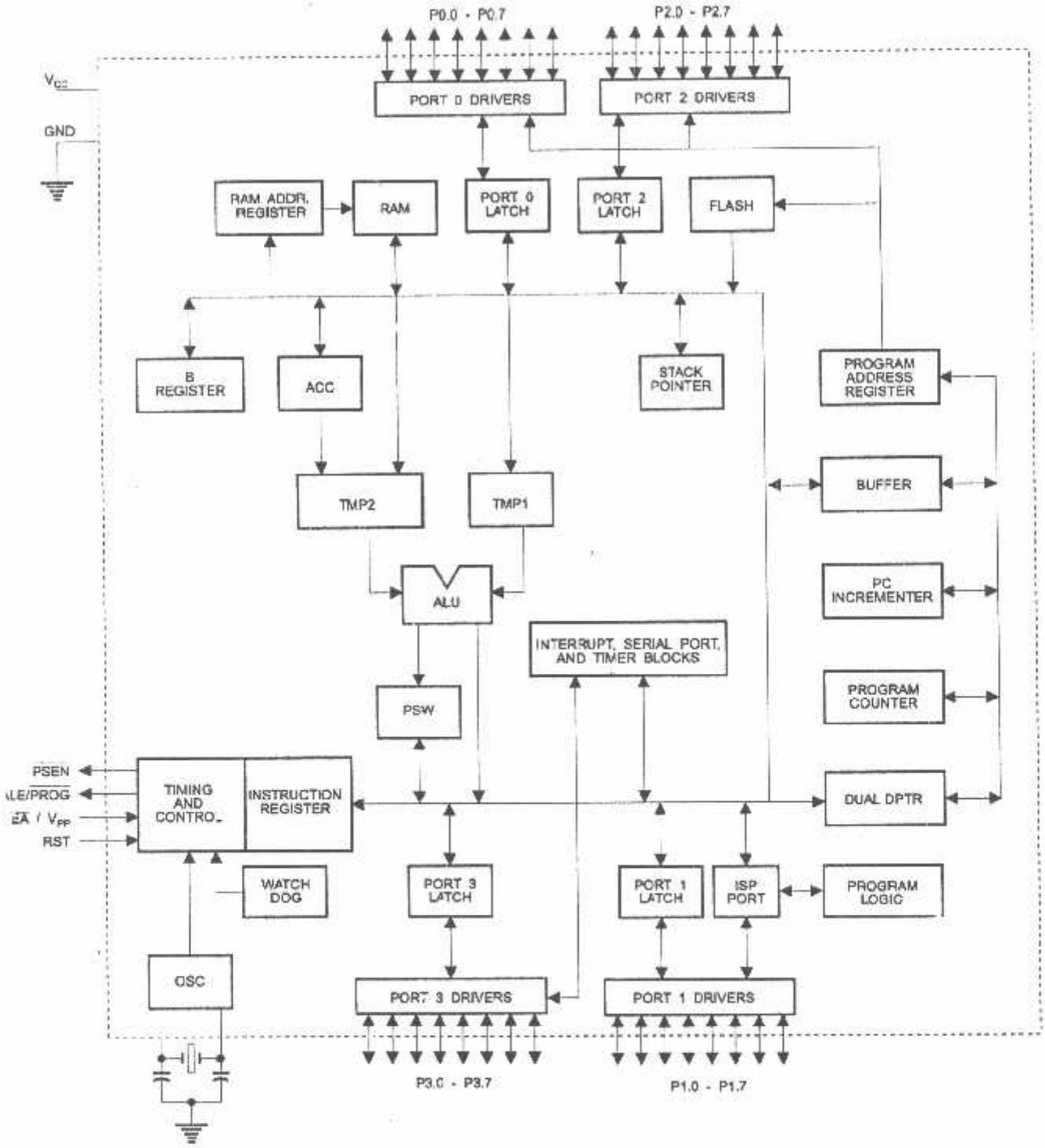
PLCC



TQFP



ock 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
PSEN
ALE/PROG
EA / V_{PP}
RST
TIMING AND CONTROL
INSTRUCTION REGISTER
DUAL DPTR
WATCH DOG
PORT 3 LATCH
PORT 1 LATCH
ISP PORT
PROGRAM LOGIC
OSC
PORT 3 DRIVERS
PORT 1 DRIVERS
P3.0 - P3.7
P1.0 - P1.7

n Description

:C Supply voltage.

ID Ground.

rt 0 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.**

rt 1 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.

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

Port Pin	Alternate Functions
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

rt 2 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.

rt 3 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 AT89S51, as shown in the following table.

Port Pin	Alternate Functions
P3.0	RXD (serial input port)
P3.1	TXD (serial output port)
P3.2	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)

ST

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.

LE/PROG

Address Latch Enable (ALE) is an output pulse for latching the low byte of the address during accesses to external memory. This pin is also the program pulse input (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.

SEN

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

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

A/VPP

External Access Enable. 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, EA will be internally latched on reset.

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.

TAL1

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

TAL2

Output from the inverting oscillator amplifier

Special Function Registers

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

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

Table 1. AT89S51 SFR Map and Reset Values

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

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.

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

Table 2. AUXR: Auxiliary Register

AUXR Address = 8EH Reset Value = XXX00XX0B

Not Bit Addressable

	–	–	–	WDIDLE	DISRTO	–	–	DISALE
Bit	7	6	5	4	3	2	1	0

– Reserved for future expansion

DISALE Disable/Enable ALE

DISALE Operating Mode

0 ALE is emitted at a constant rate of 1/6 the oscillator frequency

1 ALE is active only during a MOVX or MOVC instruction

DISRTO Disable/Enable Reset out

DISRTO

0 Reset pin is driven High after WDT times out

1 Reset pin is input only

WDIDLE Disable/Enable WDT in IDLE mode

WDIDLE

0 WDT continues to count in IDLE mode

1 WDT halts counting in IDLE mode

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

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

Table 3. AUXR1: Auxiliary Register 1

AUXR1

Address = A2H

Reset Value = XXXXXXX0B

Not Bit Addressable

	–	–	–	–	–	–	–	DPS
Bit	7	6	5	4	3	2	1	0

– Reserved for future expansion

DPS Data Pointer Register Select

DPS	
0	Selects DPTR Registers DP0L, DP0H
1	Selects DPTR Registers DP1L, DP1H

emory rganization

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.

ogram Memory

If the $\overline{EA}$ pin is connected to GND, all program fetches are directed to external memory.

On the AT89S51, if $\overline{EA}$ is connected to V_{CC}, program fetches to addresses 0000H through FFFH are directed to internal memory and fetches to addresses 1000H through FFFFH are directed to external memory.

ita Memory

The AT89S51 implements 128 bytes of on-chip RAM. The 128 bytes are accessible via direct and indirect addressing modes. Stack operations are examples of indirect addressing, so the 128 bytes of data RAM are available as stack space.

atchdog mer ne-time abled with set-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.

sing 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.

DT During ›wer-down d Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt, which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the AT89S51 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

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

Before going into the IDLE mode, the WDIDLE bit in SFR AUXR is used to determine whether the WDT continues to count if enabled. The WDT keeps counting during IDLE (WDIDLE bit = 0) as the default state. To prevent the WDT from resetting the AT89S51 while in IDLE mode, the user should always set up a timer that will periodically exit IDLE, service the WDT, and reenter IDLE mode.

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

ART

The UART in the AT89S51 operates the same way as the UART in the AT89C51. 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'.

mer 0 and 1

Timer 0 and Timer 1 in the AT89S51 operate the same way as Timer 0 and Timer 1 in the AT89C51. For further information on the timers' operation, refer to the ATMEL Web site (http://www.atmel.com). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

terrupts

The AT89S51 has a total of five interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), two timer interrupts (Timers 0 and 1), and the serial port interrupt. These interrupts are all shown in Figure 1.

Each of these interrupt sources can be individually enabled or disabled by setting or clearing a bit in Special Function Register IE. IE also contains a global disable bit, EA, which disables all interrupts at once.

Note that Table 4 shows that bit position IE.6 is unimplemented. In the AT89S51, bit position IE.5 is also unimplemented. User software should not write 1s to these bit positions, since they may be used in future AT89 products.

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle

Table 4. Interrupt Enable (IE) Register

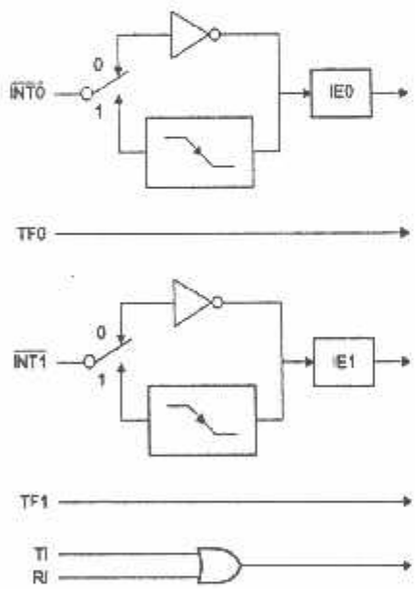
(MSB)							(LSB)
EA	–	–	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

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

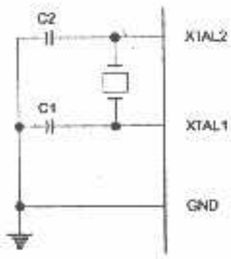
Figure 1. Interrupt Sources



:illator iracteristics

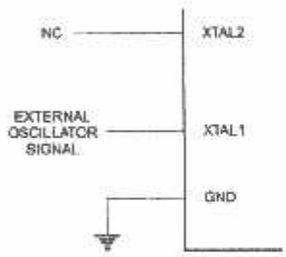
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 3. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 2. Oscillator Connections



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

Figure 3. External Clock Drive Configuration



; Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special function registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware reset.

Note that when idle mode is terminated by a hardware reset, the device normally resumes program execution from where it left off, up to two machine cycles before the internal reset algorithm takes control. On-chip hardware inhibits access to internal RAM in this event, but access to the port pins is not inhibited. To eliminate the possibility of an unexpected write to a port pin when idle mode is terminated by a reset, the instruction following the one that invokes idle mode should not write to a port pin or to external memory.

ver-down de

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by activation of an enabled external interrupt into $\overline{INT0}$ or $\overline{INT1}$. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Table 5. 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

›gram mory Lock s

The AT89S51 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 6. 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.

›gramming Flash – ·allel Mode

The AT89S51 is shipped with the on-chip Flash memory array ready to be programmed. The programming interface needs a high-voltage (12-volt) program enable signal and is compatible with conventional third-party Flash or EPROM programmers.

The AT89S51 code memory array is programmed byte-by-byte.

Programming Algorithm: Before programming the AT89S51, the address, data, and control signals should be set up according to the Flash programming mode table and Figures 13 and 14. To program the AT89S51, take the following steps:

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise $\overline{EA}/V_{PP}$ to 12V.
5. Pulse ALE/$\overline{PROG}$ once to program a byte in the Flash array or the lock bits. The byte-write cycle is self-timed and typically takes no more than 50 µs. Repeat steps 1 through 5, changing the address and data for the entire array or until the end of the object file is reached.

$\overline{Data}$ Polling: The AT89S51 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/Busy: The progress of byte programming can also be monitored by the RDY/$\overline{BSY}$ output signal. P3.0 is pulled low after ALE goes high during programming to indicate $\overline{BUSY}$. P3.0 is pulled high again when programming is done to indicate READY.

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

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

(000H) = 1EH indicates manufactured by Atmel
(100H) = 51H indicates 89S51
(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.

ɔgramming
› Flash –
rial 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.

ial
gramming
orithm

To program and verify the AT89S51 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 that returns the content at the selected address at serial output MISO/P1.6.
5. At the end of a programming session, RST can be set low to commence normal device operation.

Power-off sequence (if needed):

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

Set RST to "L".

Turn V_{CC} power off.

Data Polling: The 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.

al gramming ruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 8 on page 18.

gramming rface – allel Mode

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

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

e 7. Flash Programming Modes

e	V_{CC}	RST	PSEN	ALE/ PROG	EA/ V_{PP}	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.3-0 Address	P1.7-0 Address
e Code Data	5V	H	L	(2)	12V	L	H	H	H	H	D_{IN}	A11-8	A7-0
d Code Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A11-8	A7-0
e Lock Bit 1	5V	H	L	(3)	12V	H	H	H	H	H	X	X	X
n Lock Bit 2	5V	H	L	(3)	12V	H	H	H	L	L	X	X	X
e Lock Bit 3	5V	H	L	(3)	12V	H	L	H	H	L	X	X	X
d Lock Bits 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Erase	5V	H	L	(1)	12V	H	L	H	L	L	X	X	X
d Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	0000	00H
d Device ID	5V	H	L	H	H	L	L	L	L	L	51H	0001	00H
d Device ID	5V	H	L	H	H	L	L	L	L	L	06H	0010	00H

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

Figure 4. Programming the Flash Memory (Parallel Mode)

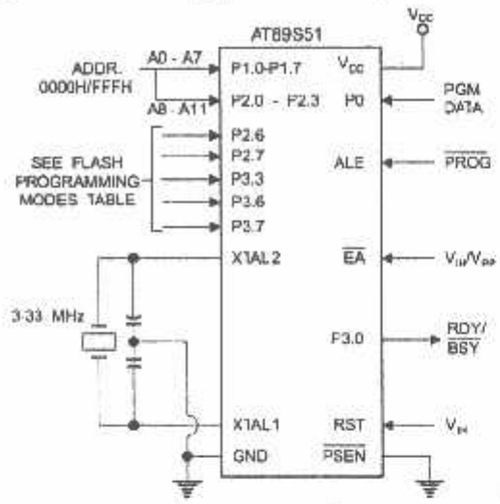
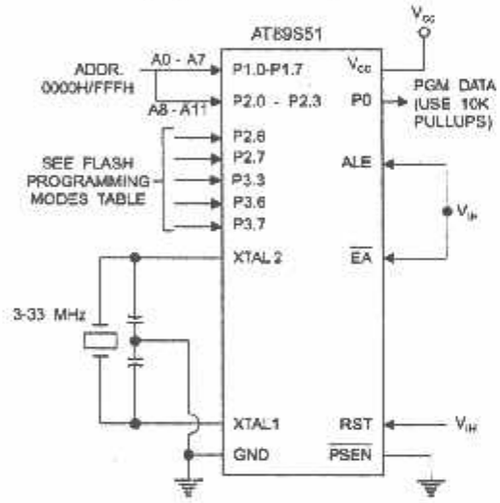
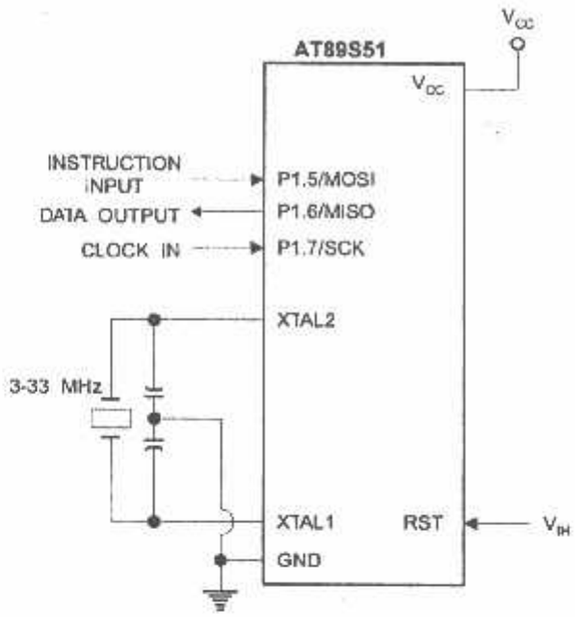


Figure 5. Verifying the Flash Memory (Parallel Mode)

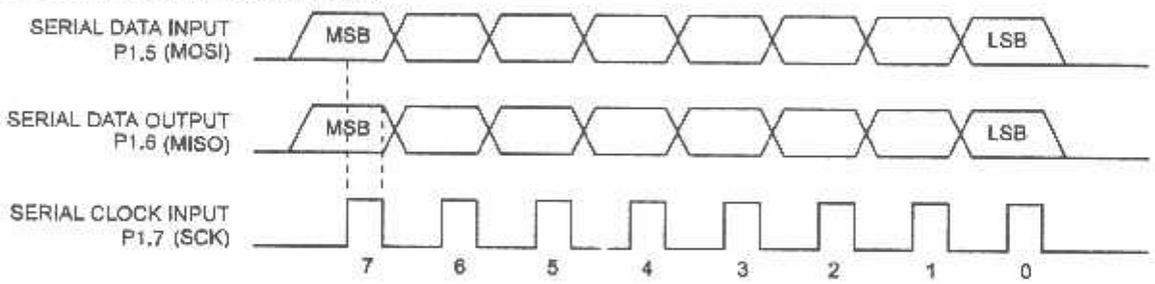


ıre 7. Flash Memory Serial Downloading



ısh Programming and Verification Waveforms – Serial Mode

ıre 8. Serial Programming Waveforms

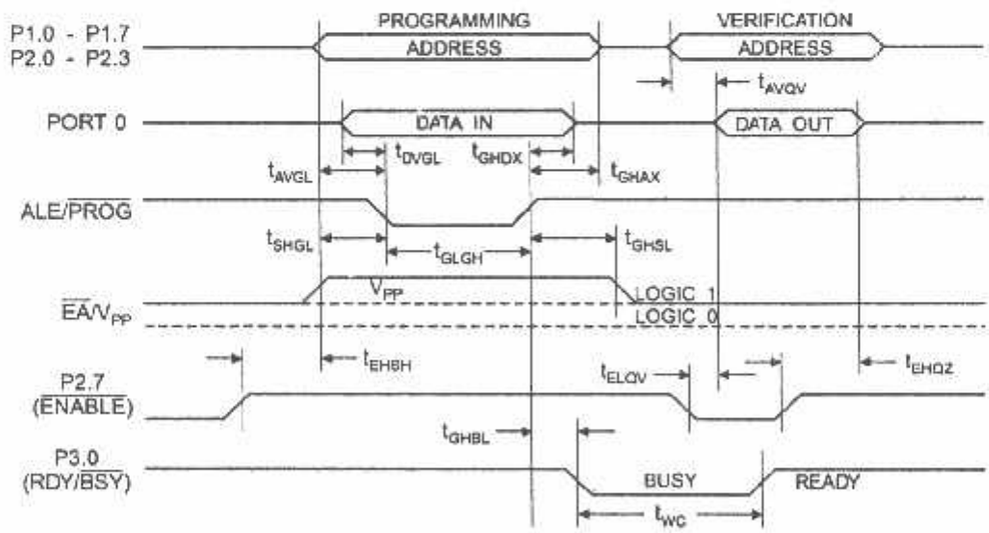


sh Programming and Verification Characteristics (Parallel Mode)

20°C to 30°C, V_{CC} = 4.5 to 5.5V

ıbol	Parameter	Min	Max	Units
	Programming Supply Voltage	11.5	12.5	V
	Programming Supply Current		10	mA
	V_{CC} Supply Current		30	mA
:LCL	Oscillator Frequency	3	33	MHz
iL	Address Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
ıX	Address Hold After $\overline{PROG}$	$48t_{CLCL}$		
iL	Data Setup to $\overline{PROG}$ Low	$48t_{CLCL}$		
ıX	Data Hold After $\overline{PROG}$	$48t_{CLCL}$		
iH	P2.7 ($\overline{ENABLE}$) High to V_{PP}	$48t_{CLCL}$		
iL	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
iL	V_{PP} Hold After $\overline{PROG}$	10		µs
iH	$\overline{PROG}$ Width	0.2	1	µs
ıV	Address to Data Valid		$48t_{CLCL}$	
ıV	$\overline{ENABLE}$ Low to Data Valid		$48t_{CLCL}$	
ıZ	Data Float After $\overline{ENABLE}$	0	$48t_{CLCL}$	
iL	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
	Byte Write Cycle Time		50	µs

ıre 6. Flash Programming and Verification Waveforms – Parallel Mode



e 8. Serial Programming Instruction Set

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

s:
1. The signature bytes are not readable in Lock Bit Modes 3 and 4.
2. B1 = 0, B2 = 0 → Mode 1, no lock protection
   B1 = 0, B2 = 1 → Mode 2, lock bit 1 activated
   B1 = 1, B2 = 0 → Mode 3, lock bit 2 activated
   B1 = 1, B1 = 1 → Mode 4, lock bit 3 activated

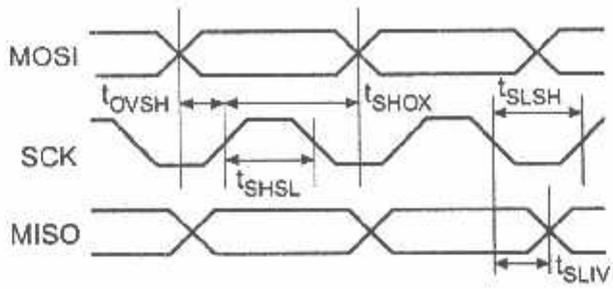
   } Each of the lock bits needs to be activated sequentially before Mode 4 can be executed.

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.

·rial Programming Characteristics

gure 9. Serial Programming Timing



ble 9. Serial Programming Characteristics, T_A = -40° C to 85° C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

ymbol	Parameter	Min	Typ	Max	Units
/t_{CLCL}	Oscillator Frequency	0		33	MHz
CLCL	Oscillator Period	30			ns
SHSL	SCK Pulse Width High	8 t_{CLCL}			ns
SLSH	SCK Pulse Width Low	8 t_{CLCL}			ns
OVSH	MOSI Setup to SCK High	t_{CLCL}			ns
SHOX	MOSI Hold after SCK High	2 t_{CLCL}			ns
SLIV	SCK Low to MISO Valid	10	16	32	ns
ERASE	Chip Erase Instruction Cycle Time			500	ms
SWC	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	µs

;olute Maximum Ratings*

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

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

Characteristics

'alues 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.

ibol	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 Vo tage (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 Vo tage (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_N < V_{CC}		±10	µA
'T	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[2]	V_{CC} = 5.5V		50	µA

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

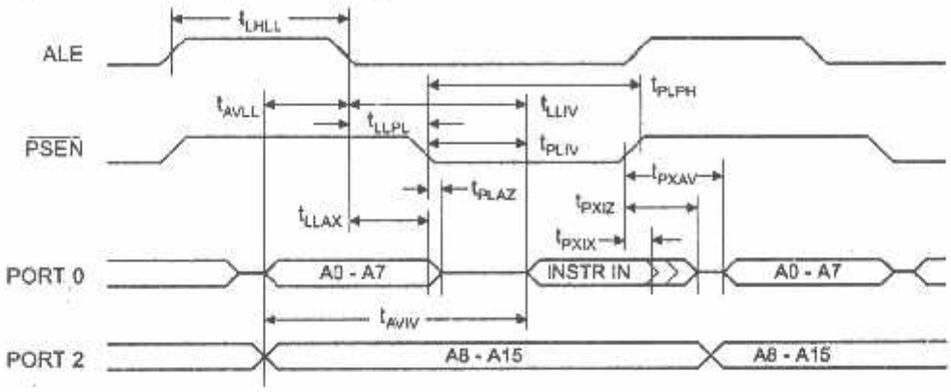
Characteristics

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

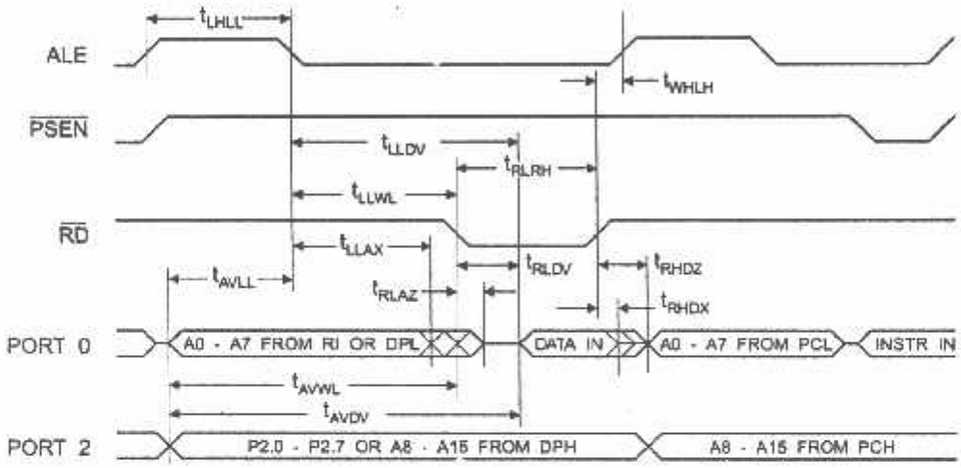
ernal Program and Data Memory Characteristics

bol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
$1/t_{CLCL}$	Oscillator Frequency			0	33	MHz
t_{LHLL}	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
t_{AVLL}	Address Valid to ALE Low	43		t_{CLCL}-25		ns
t_{LLAX}	Address Hold After ALE Low	48		t_{CLCL}-25		ns
t_{LLIV}	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-25		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-45		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-60	ns
t_{PXIX}	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
t_{PXIZ}	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-25	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
t_{AVIV}	Address to Valid Instruction In		312		$5t_{CLCL}$-80	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		10		10	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
t_{RHDX}	Data Hold After $\overline{RD}$	0		0		ns
t_{RHDZ}	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
t_{LLDV}	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
t_{AVDV}	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-30		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-130		ns
t_{WHQX}	Data Hold After $\overline{WR}$	33		t_{CLCL}-25		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		0		0	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-25	t_{CLCL}+25	ns

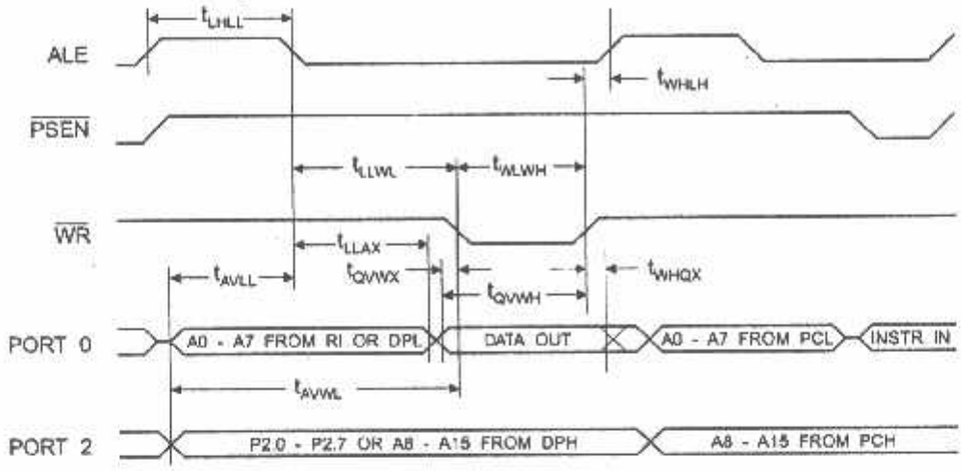
ernal Program Memory Read Cycle



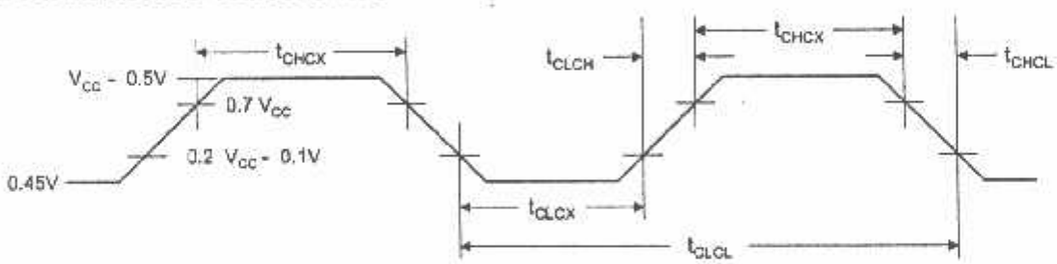
ernal Data Memory Read Cycle



ernal Data Memory Write Cycle



ernal Clock Drive Waveforms



ernal Clock Drive

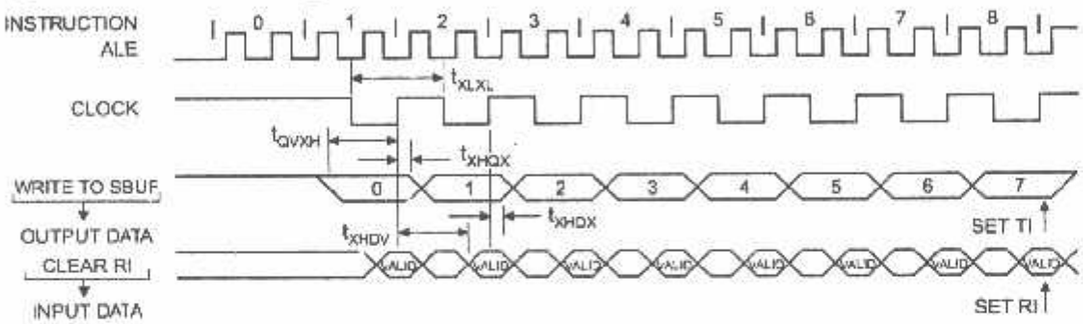
ıbol	Parameter	Min	Max	Units
CL	Oscillator Frequency	0	33	MHz
	Clock Period	30		ns
X	High Time	12		ns
X	Low Time	12		ns
H	Rise Time		5	ns
L	Fall Time		5	ns

al Port Timing: Shift Register Mode Test Conditions

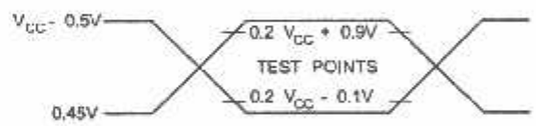
alues in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

bol	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

t Register Mode Timing Waveforms

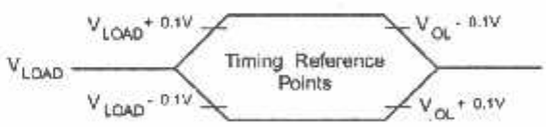


Testing Input/Output Waveforms[1]



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

at 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.

lering Information

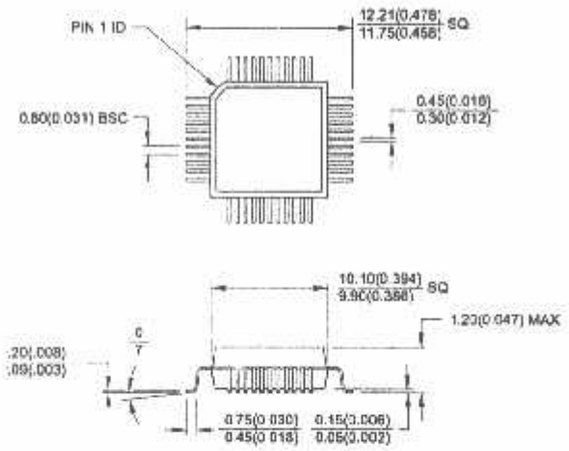
ɔeed /Hz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC AT89S51-24JC AT89S51-24PC	44A 44J 40P6	Commercial (0° C to 70° C)
		AT89S51-24AI AT89S51-24JI AT89S51-24PI	44A 44J 40P6	Industrial (-40° C to 85° C)
33	4.5V to 5.5V	AT89S51-33AC AT89S51-33JC AT89S51-33PC	44A 44J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

= Preliminary Availability

	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)

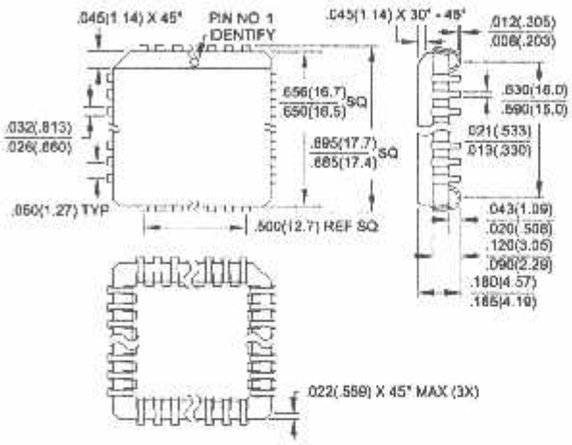
:kaging Information

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

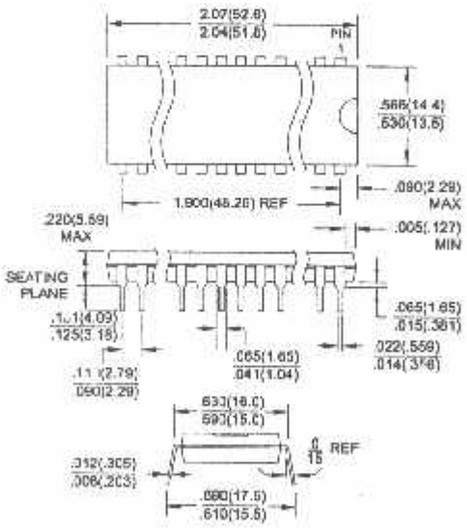


Controlling dimension: millimeters

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



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



nel Headquarters

,orate Headquarters
325 Orchard Parkway
an Jose, CA 95131
EL (408) 441-0311
AX (408) 487-2600

'ope
tmel SarL
oute des Arsenaux 41
asa Postale 80
H-1705 Fribourg
witzerland
EL (41) 26-426-5555
AX (41) 26-426-5500

'a
tmel Asia, Ltd.
oom 1219
hinachem Golden Plaza
7 Modv Road Tsimhatsui
ast Kowloon
long Kong
EL (852) 2721-9778
AX (852) 2722-1369

'an
tmel Japan K.K.
F, Tonetsu Shinkawa Bldg.
-24-8 Shinkawa
huo-ku, Tokyo 104-0033
apan
EL (81) 3-3523-3551
AX (81) 3-3523-7581

Atmel Product Operations

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

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

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

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

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

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

e-mail
literature@atmel.com

Web Site
http://www.atmel.com

2487A–10/01/xM

Turbo IC, Inc.

24C16

CMOS I²C 2-WIRE BUS
16K ELECTRICALLY ERASABLE PROGRAMMABLE ROM
2K X 8 BIT EEPROM

TURES :

tended Power Supply Voltage
- Single Vcc for Read and Programming (Vcc = 2.7 V to 5.5 V)

w Power (Isb = 2µa @ 5.5 V)

Bus, 2-Wire Serial Interface

pport Byte Write and Page Write (16 Bytes)

tomatic Page write Operation (maximum 10 ms)
- Internal Control Timer
- Internal Data Latches for 16 Bytes

rdware Data Protection by Write Protect Pin

gh Reliability CMOS Technology with EEPROM Cell
- Endurance : 1,000,000 Cycles
- Data Retention : 100 Years

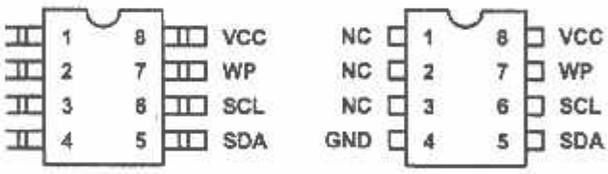
DESCRIPTION:

The Turbo IC 24C16 is a serial 16K EEPROM fabricated with Turbo's proprietary, high reliability, high performance CMOS technology. It's 16K of memory is organized as 2,048 x 8 bits. The memory is configured as 128 pages with each page containing 16 bytes. This device offers significant advantages in low power and low voltage applications.

The Turbo IC 24C16 uses the I²C addressing protocol and 2-wire serial interface which includes a bidirectional serial data bus synchronized by a clock. It offers a flexible byte write and a faster 16-byte page write. The data in the upper half of memory can be protected by a write protect pin.

The Turbo IC 24C16 is assembled in either a 8-pin PDIP or 8-pin SOIC package. Pin #1, #2, and #3 are not connected (NC). Pin #4 is the ground (Vss). Pin #5 is the serial data (SDA) pin used for bidirectional transfer of data. Pin #6 is the serial clock (SCL) input pin. Pin #7 is the write protect (WP) input pin, and Pin #8 is the power supply (Vcc) pin.

DESCRIPTION



8 pin SOIC 8 pin PDIP

All data is serially transmitted in bytes (8 bits) on the SDA bus. To access the Turbo IC 24C16 (slave) for a read or write operation, the controller (master) issues a start condition by pulling SDA from high to low while SCL is high. The master then issues the device address byte which consists of 1010 (B10) (B9) (B8) (R/$\overline{W}$). The most significant bits (1010) are a device type code signifying an EEPROM device. The B[10:8] bits are the 3 most significant bits of the memory address. The read/write bit determines whether to do a read or write operation. After each byte is transmitted, the receiver has to provide an acknowledge by pulling the SDA bus low on the ninth clock cycle. The acknowledge is a handshake signal to the transmitter indicating a successful data transmission.

)ESCRIPTION

TE PROTECT (WP)

the write protect input is connected to Vcc, pper half of memory (400-7FFH) is protected ist write operations. For normal write operathe write protect pin should be grounded. this pin is left unconnected, WP is interd as zero.

SERIAL DATA (SDA)

SDA is a bidirectional pin used to transfer data in and out of the Turbo IC 24C16. The pin is an open-drain output. A pullup resistor must be connected from SDA to Vcc.

SERIAL CLOCK (SCL)

The SCL input synchronizes the data on the SDA bus. It is used in conjunction with SDA to define the start and stop conditions. It is also used in conjunction with SDA to transfer data to and from the Turbo IC 24C16.

:CRIPTION (Continued):

a write operation, the master issues a start condition, a ce address byte, a memory address byte, and then up to ata bytes. The Turbo IC 24C16 acknowledges after each transmission. To terminate the transmission, the master es a stop condition by pulling SDA from low to high while is high.

For a read operation, the master issues a start condition and a device address byte. The Turbo IC 24C16 acknowledges, and then transmits a data byte, which is accessed from the EEPROM memory. The master acknowledges, indicating that it requires more data bytes. The Turbo IC 24C16 transmits more data bytes, with the memory address counter automatically incrementing for each data byte, until the master does not acknowledge, indicating that it is terminating the transmission. The master then issues a stop condition.

'ICE OPERATION:

IRECTIONAL BUS PROTOCOL:

Turbo IC 24C16 follows the I²C bus protocol. The protolefines any device that sends data onto the SDA bus as ansmitter, and the receiving device as a receiver. The ce controlling the transfer is the master and the device g controlled is the slave. The master always initiates the transfers, and provides the clock for both transmit and ive operations. The Turbo IC 24C16 acts as a slave de- in all applications. Either the master or the slave can control of the SDA bus, depending on the requirement e protocol.

RT/STOP CONDITION AND DATA TRANSITIONS:

e SCL clock is high, a high to low transition on the SDA s recognized as a START condition which precedes any or write operation. While SCL clock is high, a low to transition on the SDA bus is recognized as a STOP con- n which terminates the communication and places the o IC 24C16 into standby mode. All other data transitions ie SDA bus must occur while SCL clock is 'ow to ensure er operation.

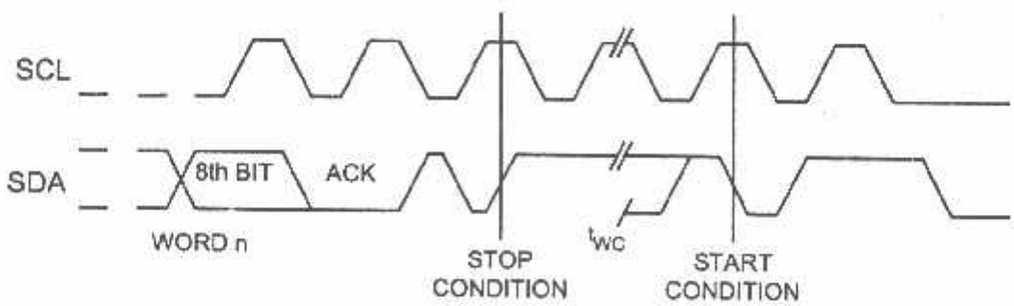
ACKNOWLEDGE:

All data is serially transmitted in bytes (8 bits) on the SDA bus. The acknowledge protocol is used as a handshake signal to indicate successful transmission of a byte of data. The bus transmitter, either the master or the slave (Turbo IC 24C16), releases the bus after sending a byte of data on the SDA bus. The receiver pulls the SDA bus low during the ninth clock cycle to acknowledge the successful transmission of a byte of data. If the SDA is not pulled low during the ninth clock cycle, the Turbo IC 24C16 terminates the data transmission and goes into standby mode.

For the write operation, the Turbo IC 24C16 acknowledges after the device address byte, acknowledges after the memory address byte, and acknowledges after each subsequent data byte.

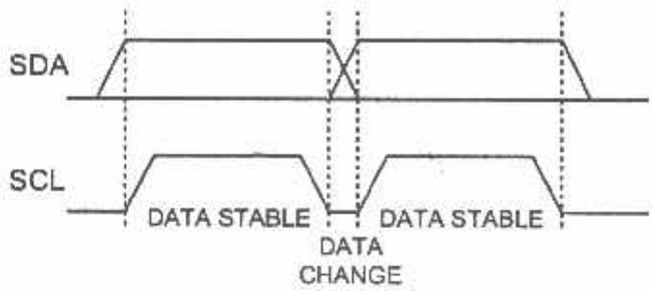
For the read operation, the Turbo IC 24C16 acknowledges after the device address byte. Then the Turbo IC 24C16 transmits each subsequent data byte, and the master acknowledges after each data byte transfer, indicating that it requires more data bytes. The Turbo IC 24C16 monitors the SDA bus for the acknowledge. To terminate the transmission, the master does not acknowledge, and then sends a stop condition.

te Cycle Timing

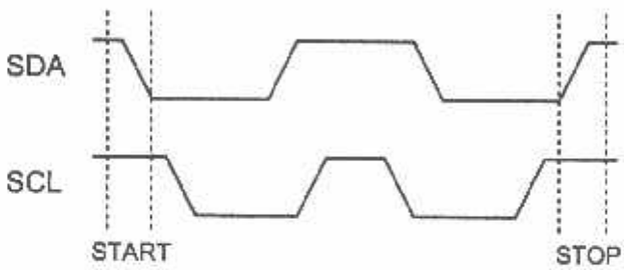


The write cycle time t_{wc} is the time from a valid stop condition of a write sequence to the end of the internal clear / write cycle.

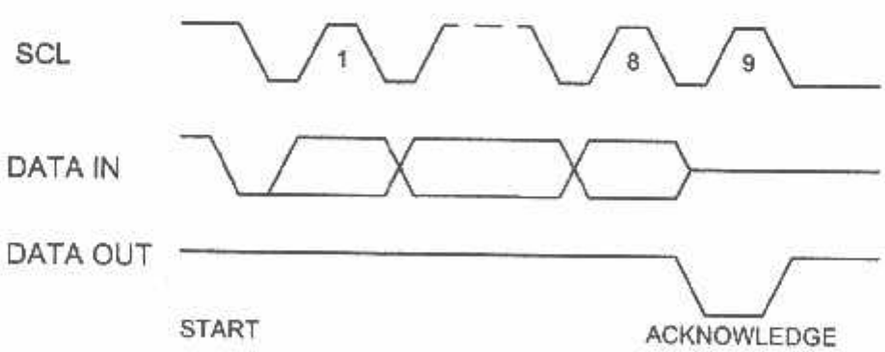
ıta Valid



ırt and Stop Definition



ıtput Acknowledge



ICE ADDRESSING:
wing the start condition, the master will issue a device ess byte consisting of 1010 (B10) (B9) (B8) (R/$\overline{W}$) to ss the selected Turbo IC 24C16 for a read or write op- on. The B[10:8] bits are the 3 most significant bits of the ory address. The (R/$\overline{W}$) bit is a high (1) for read and low ir write.

A INPUT DURING WRITE OPERATION:
ng the write operation, the Turbo IC 24C16 latches the bus signal on the rising edge of the SCL clock.

A OUTPUT DURING READ OPERATION:
ng the read operation, the Turbo IC 24C16 serially shifts data onto the SDA bus on the falling edge of the SCL .

MORY ADDRESSING:
memory address is sent by the master in the form of 2 s. Memory address bits B[10:8], are included in the de- address byte. The remaining memory address bits B[7:0] ncluded in the second byte. The memory address byte only be sent as part of a write operation.

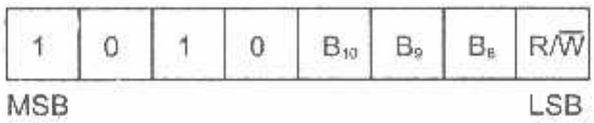
E WRITE OPERATION:
master initiates the byte write operation by issuing a condition, followed by the device address byte 1010) (B9) (B8) 0, followed by the memory address byte, wed by one data byte, followed by an acknowledge, then p condition. After each byte transfer, the Turbo IC 24C16 owledges the successful data transmission by pulling SDA bus low. The stop condition starts the internal ROM write cycle, and all inputs are disabled until the pletion of the write cycle. If the WP pin is high (1) and the mory address is within the upper half (400-7FFH) of mory, then the stop condition does not start the internal cycle and the Turbo IC 24C16 is immediately ready for ext command.

E WRITE OPERATION:
master initiates the page write operation by issuing a condition, followed by the device address byte 1010) (B9) (B8) 0, followed by the memory address byte, wed by up to 16 data bytes, followed by an acknowledge, a stop condition. After each byte transfer, the Turbo IC 16 acknowledges the successful data transmission by ng SDA low. After each data byte transfer, the memory ess counter is automatically incremented by one. The condition starts the internal EEPROM write cycle only if top condition occurs in the clock cycle immediately fol- ig the acknowledge (10th clock cycle). All inputs are dis- d until the completion of the write cycle. If the WP pin is (1) and the memory address is within the upper half (400-7FFH) of memory, then the stop condition does not start the internal write cycle, and the Turbo IC 24C16 is immediately ready for the next command.

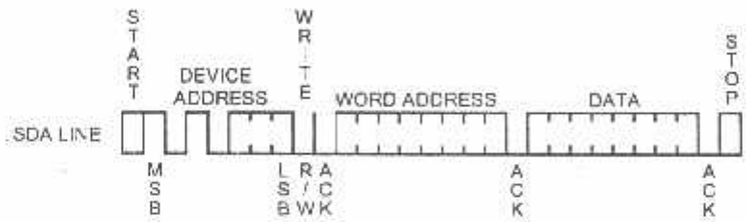
POLLING ACKNOWLEDGE:
During the internal write cycle of a write operation in the Turbo IC 24C16, the completion of the write cycle can be detected by polling acknowledge. The master starts acknowledge polling by issuing a start condition, then followed by the device address byte 1010 (B10) (B9) (B8) 0. If the internal write cycle is finished, the Turbo IC 24C16 acknowledges by pulling the SDA bus low. If the internal write cycle is still ongoing, the Turbo IC 24C16 does not acknowledge because it's inputs are disabled. Therefore, the device will not respond to any command. By using polling acknowledge, the system delay for write operations can be reduced. Otherwise, the system needs to wait for the maximum internal write cycle time, tWC, given in the spec.

POWER ON RESET:
The Turbo IC 24C16 has a Power On Reset circuit (POR) to prevent data corruption and accidental write operations during power up. On power up, the internal reset signal is on and the Turbo IC 24C16 will not respond to any command until the VCC voltage has reached the POR threshold value.

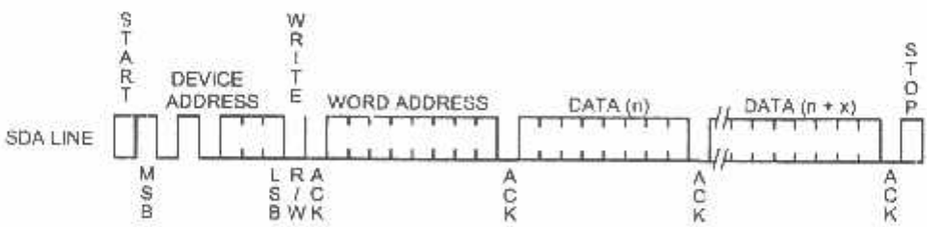
evice Address



te Write



ge Write



RRENT ADDRESS READ:

internal memory address counter of the Turbo IC 24C16 tains the last memory address accessed during the pre- is read or write operation, incremented by one. To start current address read operation, the master issues a start dition, followed by the device address byte 1010 (B10)) (B8) 1. The Turbo IC 24C16 responds with an acknowl- e by pulling the SDA bus low, and then serially shifts out data byte accessed from memory at the location corre- nding to the memory address counter. The master does acknowledge, then sends a stop condition to terminate read operation. It is noted that the memory address nter is incremented by one after the data byte is shifted

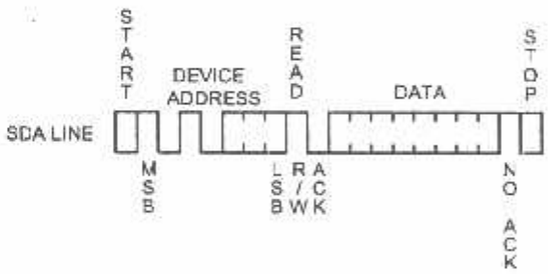
NDOM ADDRESS READ:

master starts with a dummy write operation (one with no a bytes) to load the internal memory address counter by .Issuing a start condition, followed by the device address 1010 (B10) (B9) (B8) 0, followed by the memory ad- ss bytes. Following the acknowledge from the Turbo IC 216, the master starts the current read operation by issu- n start condition, followed by the device address byte 0 (B10) (B9) (B8) 1. The Turbo IC 24C16 responds with an acknowledge by pulling the SDA bus low, and then serially shifts out the data byte accessed from memory at the location corresponding to the memory address counter. The master does not acknowledge, then sends a stop condition to terminate the read operation. It is noted that the memory address counter is incremented by one after the data byte is shifted out.

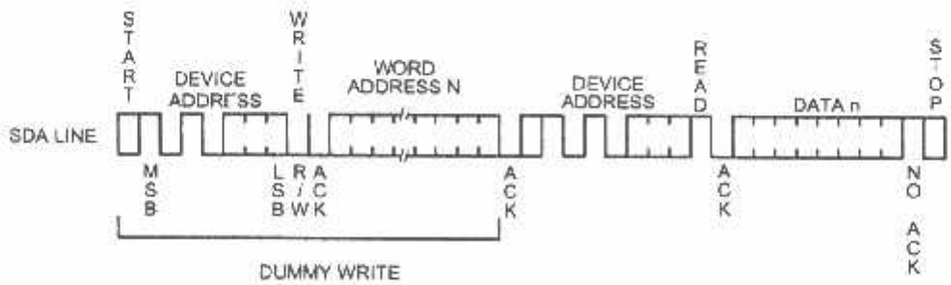
SEQUENTIAL READ:

The sequential read is initiated by either a current address read or random address read. After the Turbo IC 24C16 serially shifts out the first data byte, the master acknowledges by pulling the SDA bus low, indicating that it requires additional data bytes. After the data byte is shifted out, the Turbo IC 24C16 increments the memory address counter by one. Then the Turbo IC 24C16 shifts out the next data byte. The sequential reads continues for as long as the master keeps acknowledging. When the memory address counter is at the last memory location, the counter will 'roll-over' when incremented by one to the first location in memory (address zero). The master terminates the sequential read operation by not acknowledging, then sends a stop condition.

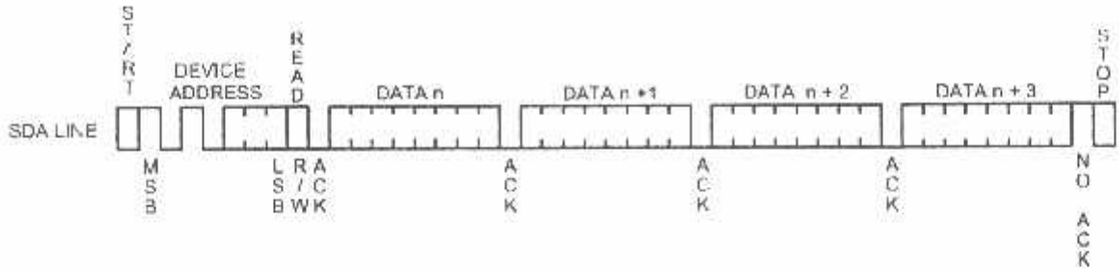
rrent Address Read



ndom Read



quential Read



SOLUTE MAXIMUM RATINGS

MPERATURE
orage: -65° C to 150° C
der Bias: -55° C to 125° C

L INPUT OR OUTPUT VOLTAGES
h respect to Vss +6 V to -0.3 V

Absolute Maximum Ratings" may cause permanent damage to the de-e. This is a stress rating only and functional operation of the device at ise or any other conditions above those indicated in the operation sec-n of this specification is not implied. Exposure to absolute maximum ing conditions for extended periods may affect device reliability.

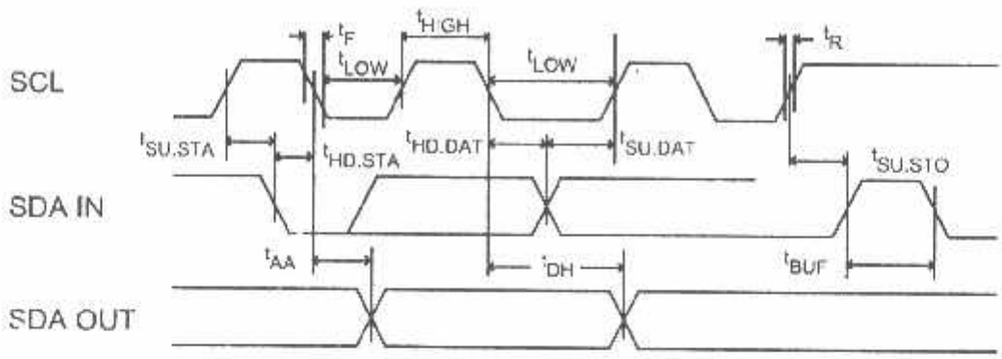
RECOMMENDED OPERATING CONDITIONS

Temperature Range:	Commercial:	0° C to 70° C
Vcc Supply Voltage:	2.7 to 5.5 Volts	
Endurance:	1,000,000 Cycles/Byte (Typical)	
Data Retention:	100 Years	

C. CHARACTERISTICS

Symbol	Parameter	Condition	Min	Max	Units
I_{cc1}	Active Vcc Current	READ at 100 KHZ	0.4	1.0	mA
I_{cc2}	Active Vcc Current	WRITE at 100 KHZ		3.0	mA
I_{sb}	Standby Current	Vcc = 2.7 v		0.5	uA
		Vcc = 5.5 v		2.0	uA
I_{li}	Input Leakage Current	Vin=Vcc Max		3	uA
I_{lo}	Output Leakage Current			3	uA
V_{il}	Input Low Voltage		-1.0	0.8	V
V_{ih}	Input High Voltage		Vccx0.7	Vcc+0.5	V
V_{ol2}	Output Low	Vcc=3.0v Iol=2.1 mA		0.4	V
V_{ol1}	Output Low	Vcc=2.7v Iol=-0.15 mA		0.25	V

: Timing



. CHARACTERISTICS

ymbol	Parameter	2.7 volt		5.5 volt		
		Min	Max	Min	Max	Units
SCL	SCL Clock Frequency		100		400	kHZ
T	Noise Suppression Time (1)		100		100	ns
LOW	Clock Low Period	4.7		1.2		us
HIGH	Clock High Period	4.0		0.6		us
AA	SCL Low to SDA Data Out	0.1	4.5	0.1	0.9	us
BUF	Bus Free to New Start (1)	4.7		1.2		us
HD.STA	Start Hold Time	4.0		0.6		us
SU.STA	Start Set-up Time	4.7		0.6		us
HD.DAT	Data-in Hold Time	0		0		us
SU.DAT	Data-in Set-up Time	200		100		ns
R	SCL and SDA Rise Time (1)	1.0		0.3		us
F	SCL and SDA Fall Time (1)	300		300		ns
SU.STO	Stop Set-up Time	4.7		0.6		us
DH	Data-out Hold Time	100		50		ns
WC	Write Cycle Time	10		10		ms

: 1 This parameter is characterized and not 100% tested.

Part Numbers & Order Information

TU24C16CS3

- 2K X 8 Serial EEPROM
- Revision C
- Package: P -PDIP, S -SOIC
- Voltage: 3 - 2.7V to 5.5V, 2 - 2.2V to 5.5V

Turbo IC, Inc. 2365 Paragon Drive, Suite I, San Jose, CA 95131 Phone: 408-392-0208 Fax: 408-392-0207
See us at www.turbo-ic.com

Rev. 5.0 -11/27/02

Manual Reference

AT Command Set
(GSM 07.07, GSM 07.05)

for SIEMENS Mobile Phone

S25

and Derivatives

Table of Contents

1. Software Interface

1.1. Overview of the Supported AT Command Set

Page	Commands 07.07	Function	E10	S10	S10 act	Rel aun ch	C25	S25
6	AT+CGMI	Issue manufacturer ID code	✔	✔	✔	✔	✔	✔
6	AT+CGMM	Issue model ID code	✔	✔	✔	✔	✔	✔
6	AT+CGMR	Output the GSM telephone version	✔	✔	✔	✔	✔	✔
7	AT+CGSN	Output the serial number (IMEI)	✔	✔	✔	✔	✔	✔
7	AT+GSN	Output the serial number (IMEI)	✔	✔	✔	✔	✔	✔
7	AT+CHUP	Terminate call	✔	✔	✔	✔	✔	✔
7	AT+CEER	Query the reason for disconnection of last call	✔	✔	✔	✔	✔	✔
8	AT+CREG	Power status	✔	✔	✔	✔	✔	✔
8	AT+COPS	Commands concerning selection of network operator	✔	✔	✔	✔	✔	✔
9	AT+CLCK	Switch blocking on and off	✔	✔	✔	✔	✔	✔
9	AT+CPWD	Change password to a block	✔	✔	✔	✔	✔	✔
10	AT+CLIP	Display telephone number of calling party	✔	✔	✔	✔	✔	✔
10	AT+CCFC	Call forwarding	✔	✔	✔	✔	✔	✔
11	AT+CHLD	Call hold and multiparty	✔	✔	✔	✔	✔	✔
11	AT+CPAS	Query the telephone status	✔	✔	✔	✔	✔	✔
12	AT+CPIN	Enter PIN and query block	✔	✔	✔	✔	✔	✔
12	AT+CBC	Battery charge	✔	✔	✔	✔	✔	✔
13	AT+CSQ	Output signal quality	✔	✔	✔	✔	✔	✔
13	AT+CPBS	Select a telephone book	✔	✔	✔	✔	✔	✔
14	AT+CPBR	Read a telephone-book entry	✔	✔	✔	✔	✔	✔
14	AT+CPBW	Write a telephone-book entry	✔	✔	✔	✔	✔	✔
15	AT+CMEE	Expanded error messages according to GSM 07.07	✔	✔	✔	✔	✔	✔
16	AT+VTS	Send a DTMF tone	✔	✔	✔	✔	✔	✔
17	AT+VTD	Set duration of a DTMF tone	✔	✔	✔	✔	✔	✔
17	AT+WS46	Select wireless network	✔	✔	✔	✔	✔	✔
17	AT+CSCS	Select TE character set	✔	✔	✔	✔	✔	✔
18	AT+CAOC	Advice of charge	✔	✔	✔	✔	✔	✔
18	AT+CSSN	Supplementary service notifications	✔	✔	✔	✔	✔	✔
19	AT+CRSM	Restricted SIM access			✔	✔	✔	✔
19	AT+CIMI	Output of IMSI			✔	✔	✔	✔
20	AT+CACM	Accumulated call meter				✔	✔	✔
20	AT+CAMM	Accumulated call meter maximum				✔	✔	✔
21	AT+CLCC	List Current Calls						✔
22	AT+CCLK	Clock						✔
22	AT+COPN	Read operator names						✔

Page	Commands 07.05	Function	E10	S10	S10 act	Rel aun ch	C25	S25
23	AT+CSMS	Selection of message service	✔	✔	✔	✔	✔	✔
24	AT+CPMS	Selection of SMS memory	✔	✔	✔	✔	✔	✔
24	AT+CMGF	SMS format	✔	✔	✔	✔	✔	✔
25	AT+CSCA	Address of the SMS service center	✔	✔	✔	✔	✔	✔
25	AT+CNMI	Display new incoming SMS	✔	✔	✔	✔	✔	✔
27	AT+CNMA	Acknowledgment of a short message directly output	✔	✔	✔	✔	✔	✔
27	AT+CMGL	List SMS	✔	✔	✔	✔	✔	✔
28	AT+CMGR	Read in an SMS	✔	✔	✔	✔	✔	✔
28	AT+CMGS	Send an SMS	✔	✔	✔	✔	✔	✔
28	AT+CMSS	Send an SMS from the SMS memory	✔	✔	✔	✔	✔	✔
29	AT+CMGW	Write an SMS to the SMS memory	✔	✔	✔	✔	✔	✔
29	AT+CMGD	Delete an SMS in the SMS memory	✔	✔	✔	✔	✔	✔
29	AT+CSCB	Select cell broadcast messages			✔	✔	✔	✔
29	AT+CMGC	Send an SMS command			✔	✔	✔	✔

1.2. AT Command Set

Remote-control operation of the GSM mobile telephone runs via a serial interface, where AT+C commands according to GSM 07.07 and GSM 07.05 as well as several manufacturer-specific AT commands are available. These commands are described in more detail on the following pages.

The commands are entered by way of the operating functions of the respective base unit. This converts the operating functions to AT commands so that the mobile phone can execute the required action. The following should be noted:

The modem guideline V.25ter applies to the sequence of the interface commands. According to this guideline, commands should begin with the character string **"AT"** and end with **"<CR>" (= 0x0D).** The input of a command is acknowledged by the display of "OK" or "ERROR". **A command currently in process is interrupted by each additional character entered**. This means that you should not enter the next command until you have received the acknowledgment; otherwise the current command is interrupted.

The commands supported are listed in the following tables:

1.2.1. Hayes-Standard Commands

The Hayes-standard commands correspond to the commands of AT Hayes-compatible modems.

Command	Function
A/	Repeat last command
AT...	Prefix for all other commands
ATA	Accept call
ATD<str>;	Dial the dialing string <str> with the voice utility Valid dial modifiers: "T" (tone dialing), "P" (pulse dialing) is ignored. The character ";" is important, for this tells the phone that the call should be set up with the voice utility. Otherwise an attempt is made to set up a data call, which the phone immediately acknowledges with "ERROR".
ATD><n>;	Dial the telephone number from the current telephone book location number <n> The telephone book is selected with the command at+cpbs
ATD><mem> <n>;	Dial the telephone number from the telephone book <mem> location number <n>
ATDL	Dial last telephone number
ATE0	Deactivate command echo
ATE1	Activate command echo
ATH[0]	Separate connection
ATQ0	Display acknowledgments
ATQ1	Suppress acknowledgments
ATV0	Output acknowledgments as numbers

Command	Function
ATV1	Output acknowledgments as text
AT&F[0]	Reset to stored profile
AT&V	Display active and stored profiles
ATZ	Set to default configuration
AT+GCAP	Output the capabilities list

1.2.2. Acknowledgments for Normal Data Communication

Response	Numeric	Meaning
OK	0	Command executed, no errors
RING	2	Ring detected
NO CARRIER	3	Link not established or disconnected
ERROR	4	Invalid command or command line too long
NO DIALTONE	6	No dial tone, dialing impossible, wrong mode
BUSY	7	Remote station busy

1.3. AT Commands and Responses According to GSM 07.07 and GSM 07.05

According to GSM, it is possible to execute an AT command in various forms.

Test command	AT+CXXX=?	The telephone responds by sending the list of parameters and value ranges; these can be set using the affiliated Write command or by means of internal processes.
Read command	AT+CXXX?	This command tells you the current value setting of the parameter(s).
Write command	AT+CXXX=<...>	This command is used to set parameters that can be set.
Execute command	AT+CXXX	The Execute command reads non-settable parameters which are influenced by internal processes in the telephone.

1.3.1. AT Cellular Commands According to GSM 07.07

AT+CGMI	Issue manufacturer ID code
Test command AT+CGMI=?	Response OK
Execute command AT+CGMI	Response <manufacturer> Parameter <manufacturer> Name of manufacturer (SIEMENS) **Important**: There is a leading output prefix +CGMI in models before the S25.

AT+CGMM	Issue model ID code
Test command AT+CGMM=?	Response OK
Execute command AT+CGMM	Response <model> Parameter <model> Name of telephone (MOBILE) **Important**: There is a leading output prefix +CGMM in models before the S25.

AT+CGMR	Output the GSM telephone version
Test command AT+CGMR=?	Response OK
Execute command AT+CGMR	Response <revision> Parameter <revision> Version of the telephone software **Important**: There is a leading output prefix +CGMR in models before the S25.

AT+CGSN	Output the serial number (IMEI)
Test command AT+CGSN=?	Response OK
Execute command AT+CGSN	Response <sn> Parameter <sn> IMEI of the telephone **Important:** There is a leading output prefix +CGMI in models before the S25.

AT+GSN	Output the serial number (IMEI)
Test command AT+GSN=?	Response OK
Execute command AT+GSN	Response +GSN: <sn> Parameter <sn> IMEI of the telephone **Important:** The output prefix +GSN may be missing in future versions.

AT+CHUP	Terminate call
Test command AT+CHUP=?	Response OK
Execute command AT+CHUP	Response OK/ERROR
	Description: All active calls and all calls on hold are terminated.

AT+CEER	Query the reason for disconnection of last call
Test command AT+CEER=?	Response OK
Execute command AT+CEER	Response +CEER: <report> Parameter <report> Disconnection reason reported as number

AT+CREG	Power status
Test command AT+CREG=?	Response +CREG: (list of supported <n>s) OK/ERROR/+CME ERROR
	Parameter <n> 0 Suppresses the unexpected network-status messages 1 Displays the unexpected network-status messagesOK/ERROR/+CME ERROR
Read command AT+CREG?	Response +CREG: <n>,<stat>[,<lac>,<ci>] OK/ERROR/+CME ERROR
	Parameter <n> See Test command <stat> 0 Not checked in, not seeking 1 Checked in 2 Not checked in, but seeking a network 3 Check-in denied by network 4 Unknown 5 Registered, roaming <lac> Hexadocimal 2-byte string type of location area code <ci> Hexadecimal 2-byte string type of cell ID
Write command AT+CREG=<n>	Parameter <n> See Test command Response OK/ERROR/+CME ERROR
	Unexpected message +CREG: <stat>

AT+COPS	Commands concerning selection of network operator
Test command AT+COPS=?	Response +COPS: [list of supported (<stat>,long alphanumeric <oper>,,numeric <oper>)s][,,(list of supported <mode>s),(list of supported <format>s)] OK/ERROR/+CME ERROR Parameter <stat> 0 Unknown 1 Useful network operator 2 Used network operator 3 Prohibited network operator <oper> Operator in the format according to <mode>
Read command AT+COPS?	Response +COPS: <mode>[,<format>,<oper] OK/ERROR/+CME ERROR Parameter <mode> 0 Automatic mode 1 Manual selection of network operator 3 Setting of format 4 Automatic, manual selected <format> 0 Long alphanumeric 2 Numeric <oper> <oper> Network operator
Write command AT+COPS=<mode>[,<format>[,<oper]]	Parameter <mode> See Read command <format> See Read command If <mode> = 1, <format> can only = 2 <oper> In numeric form only Response OK/ERROR/+CME ERROR

AT+CLCK	Switch blocking on and off **Revision to GSM 07.07 according to CR TDOC ETSI/SMG4 187/96**
Test command AT+CLCK=?	Response +CLCK: (list of supported <fac>s) OK/ERROR/+CME ERROR Parameter <fac> "CS" Keyboard lock "PS" Phone locked to SIM (device code) "SC" SIM card (PIN) "FD" FDN lock "AO" BAOC (bar all outgoing calls) "OI" BOIC (bar outgoing international calls) "OX" BOIC-exHC (bar outgoing international calls except to home country) "AI" BAIC (bar all incoming calls) "IR" BIC-Roam (bar incoming calls when roaming outside the home country) "AB" All Barring services "AG" All outgoing barring services "AC" All incoming barring services
Write command AT+CLCK=<fac>, <mode>[,<passwd>[,<class>]]	Parameter <fac> See Test command <mode> 0 Cancels block 1 Activates block 2 Queries block status <passwd> Password <class> 1 Voice 2 Data 4 Fax 7 All classes (default value) Response If <mode>=2 and command is successful +CLCK: <status>[,<class1>[<CR><LF> +CLCK: <status>, class2....]] Parameter <status> 0 On 1 Off OK/ERROR/+CME ERROR

AT+CPWD	Change password to a block
Test command AT+CPWD=?	Response +CPWD: list of supported (<fac>, <pwdlength>)s OK/ERROR/+CME ERROR Parameter <fac> "P2" PIN2 otherwise See Test command for AT+CLCK command, without "FD" <pwdlength> Password length
Write command AT+CPWD=<fac>, <oldpwd>, <newpwd>	Parameter <fac> See Test command for AT+CLCK command <oldpwd>, <newpwd> Old and new password Response OK/ERROR/+CME ERROR

AT+CLIP	Display telephone number of calling party
Test command AT+CLIP=?	Response +CLIP: (list of supported <n>s) OK/ERROR/+CME ERROR Parameter <n> 0 Suppresses the unexpected messages 1 Displays the unexpected messages
Read command AT+CLIP?	Response +CLIP: <n>, <m> OK/ERROR/+CME ERROR Parameter <n> See Test command <m> 0 CLIP not booked 1 CLIP booked 2 Unknown
Write command AT+CLIP=<n>	Parameter <n> See Read command Response OK/ERROR/+CME ERROR
	Unexpected message +CLIP: <num>,<type> Telephone number of caller

AT+CCFC	Call forwarding
Test command AT+CCFC=?	Response +CCFC: (list of supported <reas>s) OK/ERROR/+CME ERROR Parameter <reas> 0 Always 1 If busy 2 If no answer 3 If not available 4 All reasons (0-3) 5 All conditional reasons (1-3)
Write command AT+CCFC=<reas>, <mode>[, <num>[, <type>[,<class> [,,,<time>]]]]	Parameter <reas> See Test command <mode> 0 Deactivate 1 Activate 2 Query 3 Install 4 Delete <num> Telephone number <type> Type of telephone number <class> 1 Voice 2 Data 4 Fax 7 All classes <time> 1-30 Time, rounded to a multiple of five seconds Response If <mode>=2 and command is successful +CCFC: <status>, <class1>[, <num>, <type>[,,,<time>]][<CR><LF>+CCFC: ....] OK/ERROR/+CME ERROR Parameter <status> 0 Not active 1 Active

AT+CHLD	Call hold and multiparty		
Test command AT+CHLD=?	Response [+CHLD: (list of supported <n>s)] OK/ERROR/+CME ERROR		
Write command AT+CHLD=[<n>]	Parameter <n>	0	Terminates all held calls or sets UDUB (**U**ser **D**etermined **U**ser **B**usy) for a waiting call
		1	Terminates all active calls (if there are any) and accepts the other call (waiting call or held call)
		1X	Terminates call number X (X= 1-7)
		2	Puts all active calls on hold (if there are any) and accepts the other call (waiting call or held call) as active
		2X	Puts all active calls except call X (X= 1-7) on hold
		3	Connects the call put on hold to the active call
	For terminating		Terminating all calls except waiting calls is done with "AT+CHUP"
	Note:		Command scope depends on the SIM clearing and/or on the network support
	Response OK/ERROR/+CME ERROR		

AT+CPAS	Query the telephone status		
Test command AT+CPAS=?	Response +CPAS: (list of supported <pas>s) OK/ERROR/+CME ERROR		
	Parameter <pas>	0	Ready
		3	Incoming call (phone is ringing)
		4	Call is active
Execute command AT+CPAS	Response +CPAS: <pas> OK/ERROR/+CME ERROR		
	Parameter <pas>		See Test command
	OK/ERROR/+CME ERROR		

AT+CPIN	Enter PIN and query block
Test command AT+CPIN=?	Response OK
Read command AT+CPIN?	Response +CPIN: <code> OK/ERROR/+CME ERROR Parameter <code> READY — No further input necessary SIM PIN — SIM PIN input necessary SIM PUK — SIM PUK input necessary PH-SIM PIN — Device-code (theft protection) input necessary PH-SIM PUK — Device-code PUK (theft protection) input necessary SIM PIN2 — PIN2, e.g. for editing the FDN book; only possible if previous command was acknowledged with +CME ERROR:17 SIM PUK2 — Only possible if previous command was acknowledged with error +CME ERROR:18 The required error message can (must) be provoked by an attempted Write command.
Write command AT+CPIN=<pin>[, <new pin>]	Parameter <pin> — Password for appropriate block; if the block is a PUK, then a <new pin> is necessary. <new pin> — New password for the block Response OK/ERROR/+CME ERROR

AT+CBC	Battery charge
Test command AT+CBC=?	Response +CBC: (list of supported <bcs>s),(list of supported <bcl>s) OK/ERROR/+CME ERROR Parameter <bcs> 0 ME is supplied from battery 1 ME has battery but is not supplied from there 2 ME has no battery connected 3 Error <bcl> 0 Battery is flat, but no more actions possible 1-100 charge in per cent
Execute command AT+CBC	Response +CBC: <bcs>,<bcl>

AT+CSQ	Output signal quality
Test command AT+CSQ=?	Response +CSQ: (list of supported <rssi>s), list of supported <ber>) OK/ERROR/+CME ERROR Parameter <rssi> Reception level: 0 -113 dBm or less 1 -111 dBm 2-30 -109 to -53 dBm 31 -51 dBm or more 99 Unknown <ber> Bit error rate: 0-7 Like RXQUAL values from Table GSM 05.08 in Section 8.2.4 99 Unknown
Execute command AT+CSQ	Response +CSQ: <rssi>, <ber> OK/ERROR/+CME ERROR Parameter <rssi> See Test command <ber> See Test command

AT+CPBS	Select a telephone book
Test command AT+CPBS=?	Response +CPBS: (list of supported <sto>s) OK/ERROR/+CME ERROR Parameter <sto> "FD" SIM fix-dialing phonebook "SM" SIM phonebook "ME" ME phonebook "DC" ME Dialled Calls List "ON" SIM (or ME) own numbers (MSISDNs) list "LD" SIM last-dialling phonebook "MC" ME missed (unanswered received) calls list "RC" ME received calls list ***For description of telephone-book features, see** *Appendix A* **Note**: "DC" and "LD" are never both available.
Read command AT+CPBS?	Response +CPBS: <sto> OK/ERROR/+CME ERROR Parameter <sto> See Test command
Write command AT+CPBS=<sto>	Parameter <sto> See Test command Response OK/ERROR/+CME ERROR

AT+CPBR	Read a telephone-book entry
Test command AT+CPBR=?	Response +CPBR: (list of supported <index>s), <nlength>, <tlength> OK/ERROR/+CME ERROR Parameter <index> Location number <nlength> Max. length of telephone number <tlength> Max. length of text corresponding to the number
Write command AT+CPBR=<index1>[,<index2>]	Response +CPBR: <index1>, <nummer>, <typ>, <text>[<CR><LF> +CPBR: ..... +CPBR: <index2>, <nummer>, <typ>, <text>] OK/ERROR/+CME ERROR Parameter <index1> Location number where the read of the entry starts <index2> Location number where the read of the entry ends <nummer> Telephone number <typ> Type of number <text> Text corresponding to the telephone number **NOTE:** In models before the S25, empty phonebook records are reported as follows: **+CPBR: <index1>,empty** In S25ff, those empty entries don't produce any output.

AT+CPBW	Write a telephone-book entry
Test command AT+CPBW=?	Response +CPBW: (list of supported <index>s), <nlength>, <tlength> OK/ERROR/+CME ERROR Parameter <index> Location number <nlength> Max. length of telephone number <tlength> Max. length of text corresponding to the number
Write command AT+CPBW=[<index>],[<nummer>,[<typ>,[<text>]]]	Parameter <index> Location number at which the entry is written <nummer> Telephone number <typ> Type of number <text> Text corresponding to the telephone number Response OK/ERROR/+CME ERROR Note: The following characters in <text> must be entered via the escape sequence: GSM char. Seq. Seq.(hex) Note \ \5C 5C 35 43 (backslash) " \22 5C 32 32 (string delimiter) BSP \08 5C 30 38 (backspace) NULL \00 5C 30 30 (GSM null) '0' (GSM null) may cause problems on application level when using the function strlen() and should thus be represented by an escape sequence when necessary

AT+CMEE	Expandec error messages according to GSM 07.07
Test command AT+CMEE=?	Response +CMEE: (list of supported <n>s) Parameter <n> 0 Suppresses the expanded error format 1 Expanded error messages as number 2 Expanded error messages as text
Read command AT+CMEE?	Response +CMEE: <n> Parameter <n> See Read command
Write command AT+CMEE=<n>	Parameter <n> See Read command Response OK/ERROR/+CME ERROR
	Description: **The following CME errors are possible:** 0 PHONE FAILURE 1 NO CONNECTION TO PHONE 2 PH-TA LINK RESERVED 3 OPERATION NOT ALLOWED 4 OPERATION NOT SUPPORT 5 PH-SIM PIN REQUIRED 10 SIM NOT INSERTED 11 SIM PIN REQUIRED 12 SIM PUK REQUIRED 13 SIM FAILURE 14 SIM BUSY 15 SIM WRONG 16 INCORRECT PASSWORD 17 SIM PIN2 REQUIRED 18 SIM PUK2 REQUIRED 20 MEMORY FULL 21 INVALID INDEX 22 NOT FOUND 23 MEMORY FAILURE 24 TEXT TOO LONG 25 INV CHAR IN TEXT 26 DIAL STRING TOO LONG 27 INV CHAR IN DIAL 30 NO NETWORK SERVICE 31 NETWORK TIMEOUT 100 UNKNOWN 512 CALL BARRED BY BLACKLIST 513 PHONE LINK RESERVED 514 INVALID DIAL STRING 515 PHONE BUSY 550 PH-SIM PUK REQUIRED 551 NTF-SIM PIN REQUIRED

AT+VTD	Set duration of a DTMF tone
Test command AT+VTD=?	Response +VTD: (list of supported <duration>s) OK/ERROR/+CME ERROR Parameter <duration> 1-255 Duration of tone in (duration/10) seconds
Read command AT+VTD?	Response +VTD: <duration> OK/ERROR/+CME ERROR
Write command AT+VTD= <duration>	Parameter <duration> See Test command Response OK/ERROR **Important**: There is a leading output prefix +VTD in models before the S25.

AT+WS46	Select wireless network
Test command AT+WS46=?	Response +WS46: (list of supported <n>s) OK
Read command AT+WS46?	Response +WS46: <n> OK/ERROR/+CME ERROR Parameter <n> Integer; WDS side stack 12 GSM digital cellular
Write command AT+WS46=[<n>]	Response OK/ERROR/+CME ERROR **Important**: There is a leading output prefix +WS46 in models before the S25.

AT+CSCS	Select TE character set
Test command AT+CSCS=?	Response +CSCS: (list of supported <chset>s) OK
Read command AT+CSCS?	Response +CSCS: <chset> OK/ERROR/+CME ERROR Parameter <chset> String; determines which TE character set is used
Write command AT+CSCS= [<chset>]	Response OK/ERROR/+CME ERROR

AT+CAOC	Advice of charge
Test command AT+CAOC=?	Response OK
Execute command AT+CAOC	Response +CAOC: <ccm> OK/ERROR/+CME ERROR Parameter <ccm> Updated hexadecimal call meter, measured in home units; coding analogous to ACMmax on the SIM

AT+CSSN	Supplementary service notifications **Revision according to GSM 07.07 Version 5.0.0**
Test command AT+CSSN=?	Response +CSSN: (list of supported <n>s), (list of supported <m>s) Parameter <n> 0 Suppresses the +CSSI messages 1 Activates the +CSSI messages <m> 0 Suppresses the +CSSU messages 1 Activates the +CSSU messages For supported +CSSI/+CSSU messages, see also **Fehler! Verweisquelle konnte nicht gefunden werden.**
Read command AT+CSSN?	Response +CSSN: <n>,<m> Parameter <n> See Test command <m> See Test command
Write command AT+CSSN=<n>[,<m>]	Parameter <n> See Read command <m> See Read command
	Unexpected message +CSSI: <code1> +CSSU: <code2> Parameter <code1> Intermediate result code 3 Waiting call is pending <code2> Unsolicited result code 5 Held call was terminated

AT+CACM	Accumulated call meter
Test command AT+CACM=?	Response OK
Read command AT+CACM?	Response +CACM: <acm> OK/ERROR/+CME ERROR Parameter <acm> Accumulated call meter in hexadecimal format, measured in home units; coding analogous to ACMmax on the SIM
Write command AT+CACM=[<passwd>]	Response OK/ERROR/+CME ERROR Parameter <passwd> String type; usually PIN2

AT+CAMM	Accumulated call meter maximum
Test command AT+CAMM=?	Response OK
Read command AT+CAMM?	Response +CAMM: <acmmax> OK/ERROR/+CME ERROR Parameter <acmmax> Accumulated call meter maximum in hexadecimal format, measured in home units; coding analogous to ACMmax on the SIM
Write command AT+CAMM=[<acmmax>[,<passwd>]]	Response OK/ERROR/+CME ERROR Parameter <acmmax> (see Read command) <passwd> String type; usually PIN2

AT+CLCC	List Current Calls
Test command AT+CLCC=?	Response OK
Execute command AT+CLCC	Response [+CLCC: <id1>,<dir>,<stat>,<mode>,<mpty>, <number>,<type>] [<CR><LF>+CLCC: <id2>,<dir>,<stat>,<mode>,<mpty>, <number>,<type> [...]]] OK/ERROR/+CME ERROR Parameter <idx>: integer type; call identification number as described in GSM 02.30 [19] subclause 4.5.5.1; this number can be used in +CHLD command operations <dir>: 0 mobile originated (MO) call 1 mobile terminated (MT) call <stat> (state of the call): 0 active 1 held 2 dialing (MO call) 3 alerting (MO call) 4 incoming (MT call) 5 waiting (MT call) <mode> (bearer/teleservice): 0 voice 1 data 2 fax 3 voice followed by data, voice mode 4 alternating voice/data, voice mode 5 alternating voice/fax, voice mode 6 voice followed by data, data mode 7 alternating voice/data, data mode 8 alternating voice/fax, fax mode 9 unknown <mpty>: 0 call is not one of multiparty (conference) call parties 1 call is one of multiparty (conference) call parties <number>: string type phone number in format specified by <type> <type>: type of address octet in integer format

AT+CCLK	Clock
Test command AT+CCLK=?	Response OK
Write command AT+CCLK=<time>	Response OK/ERROR/+CME ERROR Parameter <time> see Test command

AT+COPN	Read operator names
Test command AT+COPN=?	Response OK
Execute command AT+COPN	Response +COPN:numeric <oper>,long alphanumeric <oper><CR><LF> +COPN:..... OK/ERROR/+CME ERROR Parameter <oper> Network operator in numeric and alphanumeric notation

1.3.2. AT Commands According to GSM 07.05 for SMS

The GSM 07.05 commands are used for operating the SMS functions of the GSM mobile phone. The GSM module MOBILE supports the SMS PDU mode.

AT+CSMS	Selection of message service **Revision according to GSM 07.05 Version 5.0.0**
Test command AT+CSMS=?	Response +CSMS: (list of supported <service>s) Parameter <service> 0 GSM 3.40 and 3.41 1 GSM 3.40 and 3.41 and compatibility of the AT command syntax for phase 2+ *(**NOTE**: Deactivating the phase 2+ compatibility is only possible if the direct output of short messages +CNMI=2,2 or +CNMI=2,3 is not activated. If necessary, the latter should be deactivated first).*
Read command AT+CSMS?	Response +CSMS: <service>,<mt>,<mo>,<bm> Parameter <service> 0 GSM 3.40 and 3.41 <mt> Mobile terminated messages 1 Type supported <mo> Mobile originated messages 1 Type supported <bm> Broadcast type messages 0 Type not supported
Write command AT+CSMS=<service>	Parameter <service> 0 GSM 3.40 and 3.41 Response +CSMS: <mt>,<mo>,<bm> OK/ERROR/+CMS ERROR

AT+CPMS	Selection of SMS memory **Revision according to GSM 07.05 Version 4.7.0**
Test command AT+CPMS=?	Response +CPMS: (list of supported <mem1>s),(list of supported <mem2>s) ,(list of supported <mem3>s) Parameter <mem1> Memory from which messages are read and deleted "SM" SIM-messages memory <mem2> Memory to which messages are written and sent "SM" SIM-messages memory <mem3> Memory in which received messages are stored, if forwarding to the PC is not set ("+CNMI") "SM" SIM-messages memory
Read command AT+CPMS?	Response +CPMS: <mem1>,<used1>,<total1>,<mem2>,<used2>,<total2>,<mem3>,<used3>,<total3> Parameter <memx> Memory from which messages are read and deleted <usedx> Number of messages currently in <memx> <totalx> Number of storable messages in <memx>
Write command AT+CPMS=<mem1>[,<mem2>[,<mem3>]]	Parameter <mem1> See Test command <mem2> See Test command <mem3> See Test command Response +CPMS: <used1>,<total1>,<used2>,<total3>,<used3>,<total3> OK/ERROR/+CMS ERROR

AT+CMGF	SMS format
Test command AT+CMGF=?	Response +CMGF: (list of supported <mode>s) Parameter <mode>: 0 PDU mode
Read command AT+CMGF?	Response +CMGF: <mode> Parameter <mode>: 0 PDU mode
Write command AT+CMGF=[<mode>]	Parameter <mode>: 0 PDU mode Response OK/ERROR

AT+CSCA	Address of the SMS service center
Test command AT+CSCA=?	Response OK
Read command AT+CSCA?	Response +CSCA: <sca>,<tosca> Parameter <sca> Service-center address in string format <tosca> Service-center address format
Write command AT+CSCA=<sca>[,<tosca>]	Parameter <sca> Service-center address in string format <tosca> Service-center address format Response OK/ERROR

AT+CNMI	Display new incoming SMS **Revision according to GSM 07.05 Version 4.7.0**
Test command AT+CNMI=?	Response +CNMI: (list of supported <mode>s),(list of supported <mt>s),(list of supported <bm>s),(list of supported <ds>s),(list of supported <bfr>s) Parameter <mode> 0 Buffers unexpected messages (but is equivalent to rejecting; see <bfr>) 2 Buffers unexpected messages if serial interface is occupied, otherwise they are output <mt> 0 Suppresses unexpected messages for incoming short messages 1 Unexpected messages of a received short message (SMS-DELIVER) that is stored on a chip card are output in the form **+CMTI: <mem>,<index>** 2 Unexpected messages of a received short message (SMS-DELIVER) (except class 2 and the message "Waiting Indication Group: *store message*") are output in the form **+CMT: [<alpha>],<length><CR><LF><pdu>** (<alpha> is not supported) Class 2 and the message "Waiting Indication Group: *store message*" are output as <mt>=1 3 Unexpected messages of a received short message (SMS-DELIVER) class 3 are output as <mt>=2. Messages with other data coding schemes are output as <mt>=1. (***NOTE**: <mt>=2 and <mt>=3 are not possible unless the Phase 2+ compatibility has been activated by means of +CSMS=1*)

	<bm>	0	Suppresses unexpected messages for incoming cell broadcast messages
		2	Outputs unexpected messages for cell broadcast messages in the form **+CBM: <length><CR><LF><pdu>**
	<ds>	0	Suppresses unexpected messages for incoming SMS status reports
		2	Outputs unexpected messages for SMS status reports in the form **+CDS: <length><CR><LF><pdu>**
	<bfr>	1	Buffered unexpected messages are rejected when switching from <mode> 0 to <mode> 2.
	<mem>		See +CPMS
	<index>		Index of the record on the chip card
	<alpha>		alphanumeric representation of the sender address
	<length>		Length of <pdu>
	<pdu>		See +CMGL
Read command AT+CNMI?	Response +CNMI: <mode>,<mt>,<bm>,<ds>,<bfr> Parameter <mode> See Test command <mt> See Test command <bm> See Test command <ds> See Test command <bfr> See Test command		
Write command AT+CNMI=[<mode>[,<mt>[,<bm>[,<ds>[,<bfr>]]]]]	Parameter <mode> See Test command <mt> See Test command <bm> See Test command <ds> See Test command <bfr> See Test command Response OK/ERROR/+CMS ERROR		
	Unexpected message +CMTI: <mem>,<index>		Indication that new message has arrived
	+CMT: ,<length><CR><LF><pdu>		Direct output of the short message
	+CDS: <length><CR><LF><pdu>		Direct output of the status report
	+CBM: <length><CR><LF><pdu>		Direct output of the cell broadcast message

AT+CNMA	Acknowledgment of a short message directly output (without storing on the chip card) **Revision according to GSM 07.05 Version 5.0.0** ***(NOTE**: This command is not possible unless the Phase 2+ compatibility has been activated by means of +CSMS=1)*
Test command AT+CNMA=?	Response +CNMA: (list of supported <n>s) Parameter <n> 0 Mode of functioning analogous to GSM 07.05 text mode
Write command AT+CNMA[=<n>]	Parameter <n> See Test command Response OK/ERROR/+CMS ERROR: <err>

AT+CMGL	List SMS **Revision according to GSM 07.05 Version 4.7.0**
Test command AT+CMGL=?	Response +CMGL: (list of supported <stat>s) Parameter <stat> 0 "REC UNREAD": received unread messages (default) 1 "REC READ": received read messages 2 "STO UNSENT": stored unsent messages 3 "STO SENT": stored sent messages 4 "ALL": all messages
Write command AT+CMGL[=<stat>]	Parameter <stat> See Test command Response **If PDU mode (+CMGF=0) and command are successful:** +CMGL: <index>,<stat>,[<alpha>],<length><CR><LF><pdu> [<CR><LF>+CMGL: <index>,<stat>,[alpha],<length><CR><LF><pdu> [...]] <pdu> The PDU begins with the service-center address (according to GSM04.11), followed by the TPDU according to GSM03.40 in hexadecimal format **otherwise:** +CMS ERROR: <err>

AT+CMGR	Read in an SMS **Revision according to GSM 07.05 Version 4.7.0**
Test command AT+CMGR=?	Response OK
Write command AT+CMGR=<index>	Parameter <index> Index of message in selected memory <mem1> Response **If PDU mode (+CMGF=0) and command are successful:** +CMGR: <stat>,[<alpha>],<length><CR><LF><pdu> <pdu> Siehe "AT+CMGL" **otherwise:** +CMS ERROR: <err>

AT+CMGS	Send an SMS
Test command AT+CMGS=?	Response OK
Write command **If PDU mode (+CMGF=0)** +CMGS=<length><CR>***PDU is given*** <ctrl-Z/ESC>	Parameter <length> Length of PDU <pdu> See "AT+CMGL" <mr> Message reference Response **If sending is successful:** +CMGS: <mr> **If sending is not successful:** +CMS ERROR: <err>

AT+CMSS	Send an SMS from the SMS memory
Test command AT+CMSS=?	Response OK
Write command +CMSS=<index>[,<da>[,<toda>]]	Parameter <index> Index of message in selected memory <mem1> <da> Destination address in string format <toda> Format of destination address <mr> Message reference Response **If sending is successful:** +CMSS: <mr> **If sending is not successful:** +CMS ERROR: <err>

AT+CMGW	Write an SMS to the SMS memory
Test command AT+CMGW=?	Response OK
Write command **If PDU mode (+CMGF=0)** AT+CMGW=<length>[,<stat>]<CR>***PDU is given*** <ctrl-Z/ESC>	Parameter <length> Length of PDU <stat> See command +CMGL <pdu> See "AT+CMGL" <index> Index of message in selected memory <mem1> Response +CMGW: <index> +CMS ERROR: <err>

AT+CMGD	Delete an SMS in the SMS memory
Test command At+CMGD=?	Response OK
Write command AT+CMGD=<index>	Parameter <index> Index of message in the selected memory <mem1> Response OK/ERROR/+CMS ERROR

AT+CSCB	Select cell broadcast messages
Test command AT+CSCB=?	Response +CSCB: (list of supported <mode>s) Parameter <mode> 0 Accepts messages that are defined in <mids> and <dcss> 1 Does not accept messages that are defined in <mids> and <dcss>
Read command AT+CSCB? Write command AT+CSCB=[<mode>[,<mids>[,<dcss>]]]	Response +CSCB: <mode>,<mids>,<dcss> Parameter <mode> See Test command <mids> String type; combinations of CBM message Ids <dcss> String type; combinations of CBM data coding schemes

AT+CMGC	Send an SMS command
Test command AT+CMGC=?	Response OK
Write command **If PDU mode (+CMGF=0)** +CMGC=<length><CR>***PDU is given*** <ctrl-Z/ESC>	Parameter <length> Length of PDU <pdu> See "AT+CMGL" <mr> Message reference Response **If sending is successful:** +CMGC: <mr> **If sending is not successful:** +CMS ERROR: <err>

Appendix A

Features of the Telephone-Book Memory

Name	Description	Category / Access	Write	Delete completely
FD	Fix-dialing number (SIM fix-dialing telephone book)	GSM 07.07 / +CPBS	Allowed (PIN2 required)	
SM	Abbreviate dialing number (SIM telephone book)	GSM 07.07 / +CPBS	Allowed (device code required if FDN replacement is active)	
DC (MD)	Mobile last dialing number (last number redial memory; only if "LD" is not available)	GSM 07.07 / +CPBS	Not allowed	
ON (OW)	Own Numbers (SIM own telephone numbers)	GSM 07.07 (Siemens) / +CPBS (historical)	Allowed	
LD	SIM last dialing number (last number redial memory on SIM)	GSM 07.07 / +CPBS	Not allowed	
ME	Mobile-equipment telephone book (ME dialing numbers)	GSM 07.07 / +CPBS	Allowed (device code required if FDN replacement is active)	
MC (MS)	Missed dialing numbers (unanswered calls)	GSM 07.07 (Siemens) / +CPBS	Not allowed	
RC (CD)	Callback dialing numbers (answered calls)	GSM 07.07 (Siemens) / +CPBS	Not allowed	

Writing to the FDN Phonebook / FDN Replacement

Writing to the fix-dialing number phonebook is protected by PIN2.
A Write sequence (to e.g. record 5) runs as follows:

```
AT+CMEE=2                    //Activate expanded error message
OK
```

```
AT+CPBS=?                          // Listing of available telephone books
+CPBS: "FD","SM","LD"
OK

AT+CPBS="FD"                       // Selection of the FDN telephone book
OK

AT+CPBW=5,1234,,"test"             // A Write to record 5 is attempted...
+CME ERROR: SIM PIN2 REQUIRED      // ... PIN2 is required for this purpose

AT+CPIN?                           // Query of the PIN status...
+CPIN: SIM PIN2                    // ... PIN2 is to be entered

AT+CPIN=12345678                   // Input of PIN2
OK

AT+CPBW=5,1234,,"test"             // A Write to record 5 is attempted...
OK                                 // PIN2 remains active as long as you use the commands
                                   // RCCL3_CMD_CPIN, RCCL3_CMD_CPBS,
                                   // RCCL3_CMD_CPBR, RCCL3_CMD_CPBW,
                                   // RCCL3_CMD_SPIC.
                                   // If you use other commands or if none of the
                                   // above commands are executed within five
                                   // minutes, the validity of PIN2 is voided.

AT+CPBW=6,5678,,"new test"         // A Write to record 6 is attempted...
OK

...
```

In addition, if there is no FDN phonebook available on the SIM, it is possible to activate a feature which activates FDN-like behavior for the "SM" and "ME" phonebooks (FDN replacement). (Currently this feature can only be activated via the MMI block/device block/excluding telephone book.)
In this case, the Write to the "SM" and "ME" phonebooks is ensured by the device code (PH-SIM PIN and PH-SIM PUK, respectively).
The sequence for entering the device code is analogous to the above example.

Lebar Asistensi Bimbingan Skripsi

Nama : Handik Prastyawan

Nim : 0752003

Masa Bimbingan :

Judul Skripsi : Pengendalian Pintu Gerbang Berbasis Mikro Controler AT89S51 Yang Diaktifkan Dengan Hp Via Miscall

No	Tanggal	Uraian	Paraf Pembimbing
1	22/2/10	Bab I Pendahuluan	
2	15/4/10	Bab II Teori Penunjang	
3	29/4/10	Bab III Perencanaan Dan Pembuatan Alat	
4	21/6/10	Bab IV Pengujian Alat (disempurnakan).	
5	10/8/10	Bab v Penutup	

Malang, 2010

Dosen Pembimbing

Bambang Prio Hartono,ST,MT

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 5530 5 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN TUGAS AKHIR
FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa	: Handik Prastyawan
Nim	: 0752003
Program Studi	: Teknik Listrik D-III
Judul Tugas Akhir	: **Pengendalian Pintu Gerbang Berbasis Mikrokontroller At 89s51 Yang Diaktifkan Dengan Hp Via Miscall**

Dipertahankan di hadapan Tim Penguji Tugas Akhir jenjang Program Diploma Tiga (D-III)

Pada Hari	: Kamis
Tanggal	: 19 Agustus 2010
Dengan Nilai	: 84 (A+)

PANITIA UJIAN TUGAS AKHIR

KETUA,

Ir. Taufik Hidayat,MT
NIP.Y 1018700151

SEKRETARIS

Bambang Prio Hartono,ST,MT
NIP. Y. 1028400082

ANGGOTA PENGUJI

Penguji I

Ir. Taufik Hidayat,MT
NIP.Y. 1018700151

Penguji II

Ir. Choirul Saleh,MT
NIP.Y. 1018800190

PENGENDALIAN PINTU GERBANG BERBASIS MIKROKONTROLLER AT 89S51 YANG DIAKTIFKAN DENGAN HP VIA MISCALL

TUGAS AKHIR

Disusun Oleh:
Nama : Handik Prastyawan
Nim : 0752003

PROGRAM STUDI TEKNIK LISTRIK D-III
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG

AGUSTUS 2010