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

PERANCANGAN DAN PEMBUATAN WEB BASED CAMERA SERVER BERBASIS MIKROKONTROLER AT89S51 DAN APACHE WEB SERVER

SKRIPSI

Disusun Oleh :

AHMAD ZAID ZAM ZAMI

01.17.115

MARET 2007

[redacted]

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN WEB BASED CAMERA CONTROLER BERBASIS MIKROKONTROLER AT89S51 DAN APACHE WEB SERVER

SKRIPSI

Diajukan Untuk Memenuhi Salah Satu Syarat Memperoleh Gelar Sarjana Teknik
Pada Jurusan Teknik Elektro S-1 Konsentrasi Elektronika

Oleh :

AHMAD ZAID ZAM ZAMI

01.17.115

Dosen Pembimbing

Ir. F. YUDI LIMPRAPTONO, MT.
NIP. Y. 1039500274

Mengetahui
Ketua Jurusan Teknik Elektro S-1

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

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

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNIK INDUSTRI
JURUSAN TEKNIK ELEKTRO

BERITA ACARA UJIAN SKRIPSI

FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa	: Ahmad Zaid Zam Zami
NIM	: 01 17 115
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Elektronika
Judul Skripsi	: Perancangan dan Pembuatan Web Based Camera Controler Berbasis Mikrokontroler AT89S51 dan Apache Web Server.

Dipertahankan dihadapan team penguji Skripsi jenjang Sarjana (S-1) pada :

Hari	: Jumat
Tanggal	: 16 Maret 2007
Dengan Nilai	: A (86,75)

PANITIA UJIAN SKRIPSI

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

ANGGOTA PENGUJI

Penguji I	Penguji II
Ir. Eko Nurcahyo	I Komang Somawirata, ST, MT
NIP. 102 870 0172	NIP. Y. 103 010 0361

ABSTRAKSI

PERANCANGAN DAN PEMBUATAN WEB BASED CAMERA SERVER BERBASIS MIKROKONTROLER AT89S51 DAN APACHE WEB SERVER

(Nama : Ahmad Zaid Zam Zami, NIM : 01. 17. 115, Teknik Elektronika S1)

(Dosen Pembimbing : Ir. F. Yudi Limpraptono, MT.)

Penggunaan jaringan internet sebagai media penyaluran sinyal kontrol dan monitoring telah berkembang pesat, diantaranya untuk pengaturan peralatan di rumah, kontrol robot, peralatan rumah tangga dan mesin produksi di industri. Namun demikian, dalam beberapa aplikasi umpan balik yang diberikan ke oprator atau user hanya status dari objek yang dikontrol. Dalam skripsi ini yang dipaparkan tentang monitoring sebuah ruangan, yaitu dengan memberikan umpan balik berupa gambar video. Komunikasi internet pada sistem monitoring ruangan terdiri atas *client* dan *camera server*. *Software* yang akan digunakan terdiri atas : *Linux, Camserv, PHP (Hypertext Preprocessor)*, dan Bahasa Pemrograman C.

Dengan menggunakan teknologi internet dan web browser maka dengan alat ini suatu ruangan dapat diamati dari jarak jauh dan dengan penerimaan gambar yang dapat langsung ditampilkan melalui browser. Selain itu sudut pengambilan gambar juga dapat diatur melalui web browser, sehingga pengambilan gambar pada ruangan bisa menyeluruh.

KATA PENGANTAR

بسم الله الرحمن الرحيم

AlhamdulillahiRobbil'Alamin, puji syukur hamba panjatkan kehadirat-Mu Ya Allah atas segala nikmat dan karunia-Mu. Sholawat serta salam semoga tercurah kepada Rasulullah SAW., keluarga, sahabat, dan para pengikutnya. Amin. Atas kehendak Allah SWT. sajalah kami dapat menyelesaikan skripsi yang berjudul **"PERENCANAAN DAN PEMBUATAN WEB BASED CAMERA CONTROLER BERBASIS MIKROKONTROLER AT89S51 DAN APACHE WEB SERVER"** ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan studi di jurusan Teknik Elektro S-1 konsentrasi Elektronika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan penyelesaian laporan Skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penyusun menyampaikan terimakasih kepada :

1. Kedua orang tuaku yang telah memberikan do'a dan dukungannya.
2. Bapak Prof. Dr. Ir. Abraham Lomi, MSEE. selaku Rektor ITN Malang.
3. Bapak Ir. F. Yudi Limpraptono, MT. selaku Ketua Jurusan Teknik Elektro S-1 dan sekaligus selaku dosen pembimbing.
4. Teman-temanku, terimakasih atas dukungannya selama ini.
5. Berbagai pihak yang tidak bisa disebutkan satu persatu, yang telah banyak membantu hingga selesainya laporan ini.

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

Harapan penyusun, semoga laporan Skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Maret 2007

Penyusun

Ahmad Zaid Zam Zami

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Salah satu keinginan setiap orang adalah ingin merasakan keamanan baik keamanan diri maupun lingkungan sekitarnya,sehingga orang berpikiran untuk membuat suatu alat yang bisa melihat kondisi keamanan daripada lingkungan disekitarnya tanpa harus dipantau dalam jarak pandang mata dalam arti kita bisa memantau atau melihat keamanan lokasi yang di pantau dalam lokasi atau jarak yang jauh.

Hal tersebut yang mendasari dari pembuatan suatu sistem monitoring ruangan pada skripsi ini. Seperti diketahui banyak sekali sistem keamanan ruangan yang sudah dipakai oleh masyarakat kita. Tetapi, sistem keamanan yang bisa memonitoring ruangan secara real time masih jarang. Dalam sistem ini gambar akan dimunculkan dalam kondisi real time dan bisa dilihat di warnet.

1.2. Tujuan

Tujuan dari skripsi ini adalah Untuk memudahkan monitoring sebuah ruangan yang dapat dipantau pada jarak jauh dengan cara membuat suatu sistem monitoring yang bekerja secara *on-line*, sehingga dapat diakses melalui *Internet*.

1.3. Perumusan Masalah

Masalah yang akan ditangani dari skripsi ini adalah Pembuatan Sistem Monitoring Ruangan Berbasis *Camera server*.

1.4. Batasan Masalah

Masalah yang akan ditangani dari Skripsi ini Pembuatan Sistem *Monitoring* Ruangan menggunakan mikrokontroler AT89S51 dan diintegrasikan ke *server* secara *On-line*, sehingga bisa diakses pada Internet.

Batasan masalah pada skripsi ini antara lain :

1. Komunikasi mikrokontroler dengan PC menggunakan komunikasi serial
2. Perancangan driver motor stepper
3. Pembuatan software pengontrol gerak kamera
4. Tidak membahas webserver
5. Tidak membahas TCP/IP secara detail
6. Tidak membahas catudaya

1.5. Metodologi

Adapun langkah-langkah yang diambil untuk menyelesaikan perubahan teoritis pada perancangan dan pembuatan alat ini adalah sebagai berikut ::

1. Studi literatur, yaitu mengumpulkan data dan bahan-bahan acuan yang dapat digunakan untuk perencanaan dan pembuatan alat.
2. Perencanaan dan pembuatan alat yang digunakan dengan cara pendekatan secara *hardware* dan *software*.
3. Perencanaan blok diagram.
4. Pembuatan alat secara keseluruhan, menyatukan rangkaian dan masing-masing blok diagram untuk mendapatkan rangkaian secara lengkap.
5. Menguji peralatan sebagai unit pemroses.
6. Menyusun naskah skripsi.

1.6. Sistematika Penulisan

Penyusunan laporan skripsi ini secara garis besar terdiri atas 5 bab pembahasan. Sistematika pembahasan tersebut adalah sebagai berikut:

BAB I : Pendahuluan, menguraikan secara singkat latar belakang, tujuan, batasan masalah dan sistematika pembahasan.

BAB 11: Teori penunjang berisi pembahasan secara garis besar tentang *Camera Server*, sistem *mikrokontroller*, komunikasi serial, sistem motor stepper.

BAB III: Perencanaan dan Implementasi, yang membahas tentang perencanaan dan implementasi sistem yang dibangun, meliputi pembuatan perangkat lunak pengontrol putaran motor stepper, perangkat lunak untuk menjalankan fungsi-fungsi *mikrokontroller*, perangkat lunak untuk pengontrol perangkat keras, dan perangkat keras secara keseluruhan.

BAB IV: Analisa dan Pengujian, yang berisi analisa hasil dari alat yang dibuat, kegagalan serta penyebab kegagalan tersebut.

BAB V : Penutup yang berisi kesimpulan yang diambil berdasarkan analisa hal–hal penting, keunikan, kelebihan/kekurangan, serta saran–saran untuk penyempurnaan alat yang dibuat.

BAB II

LANDASAN TEORI

Pada bab ini diuraikan beberapa teori dasar yang mempunyai kaitan dan mendukung tentang sistem monitoring ruangan berbasis *camera server* dan pengontrolan arah gerak kamera dengan motor stepper menggunakan *mikrokontroller.*

2.1. Linux

2.1.1. Pengenalan Linux

Kata "Linux" untuk saat ini sudah tidak asing lagi bagi para pengguna *internet* dan komunitas mahasiswa yang memiliki hobi untuk mencoba software-software baru. Secara teknis dan singkat dapat dikatakan, Linux adalah suatu sistem operasi yang bersifat *multi user* dan *multi tasking*, yang dapat berjalan di berbagai *platform* termasuk *prosesor Intel 386* maupun yang lebih tinggi. Linux dapat berinteroperasi secara baik dengan sistem operasi yang lain, termasuk Apple, Microsoft dan Novell. Nama Linux sendiri diturunkan dari pencipta awalnya, Linus Torvalds, yang sebetulnya mengacu pada suatu kumpulan software lengkap yang bersama-sama dengan kernel menyusun suatu sistem operasi yang lengkap. Lingkungan sistem operasi ini mencakup ratusan program, termasuk *kompiler*, *interpreter*, *editor* dan *utilitas*. Perangkat bantu yang mendukung konektifitas, *ethernet*, SLIP dan PPP dan *interoperabilitas*. Produk perangkat lunak yang handal (*reliable*), termasuk versi pengembangan terakhir. Kelompok pengembang yang tersebar di seluruh dunia yang telah bekerja dan

menjadikan Linux portabel ke suatu platform baru, begitu juga mendukung komunitas pengguna yang memiliki beragam kebutuhan dan juga pengguna dapat turut serta bertindak sebagai tim pengembang sendiri.

2.1.2. Perbedaan Mendasar Linux

Satu hal yang membedakan Linux terhadap sistem operasi lainnya adalah harga. Linux ini lebih murah dan dapat diperbanyak serta didistribusikan kembali tanpa harus membayar *fee* atau *royalti* kepada seseorang. Tetapi ada hal lain yang lebih utama selain pertimbangan harga yaitu mengenai *source code*. *Source code* Linux tersedia bagi semua orang sehingga setiap orang dapat terlibat langsung dalam pengembangannya.Kebebasan ini telah memungkinkan para *vendor* perangkat keras membuat driver ntuk device tertentu tanpa harus mendapatkan lisensi source code yang mahal atau menandatangani *Non Disclosure Agreement* (NDA). Dan itu juga telah menyediakan kemungkinan bagi setiap orang untuk melihat ke dalam suatu sistem operasi yang nyata dan berkualitas komersial. Karena Linux itu tersedia secara bebas di *internet*, berbagai *vendor* telah membuat suatu paket distribusi yang dapat dianggap sebagai versi kemasan Linux. Paket ini termasuk lingkungan Linux lengkap, penagkat lunak untuk instalasi dan mungkin termasuk perangkat lunak khusus dan dukungan khusus.

2.1.3. Kelebihan Linux

Linux disusun berdasarkan standar sistem operasi POSIX yang sebenarnya diturunkan berdasarkan fungsi kerja UNIX.UNIX kompatibel dengan Linux pada level *system call*, ini berarti sebagian besar program yang ditulis untuk UNIX atau

Linux dapat direkompilasi dan dijalankan pada sistem lain dengan perubahan yang minimal. Secara umum dapat dikatakan Linux berjalan lebih cepat dibanding UNIX lain pada hardware yang sama. Dan lagi UNIX memiliki kelemahan yaitu tidak bersifat *free*. MS-DOS memiliki kemiripan dengan Linux yaitu *file* sistem yang bersifat hirarkis. Tetapi MS-DOS hanya dapat dijalankan pada *prosesor x86* dan tidak mendukung *multi user* dan *multi tasking*, serta tidak bersifat *free*. Juga MSDOS tidak memiliki dukungan yang baik agar dapat berinteroperasi dengan sistem operasi lainnya, termasuk tidak tersedianya perangkat lunak *network*, program pengembang dan program utilitas yang ada dalam Linux.

MS Windows menawarkan kemampuan grafis yang ada pada Linux termasuk kemampuan *networking* tetapi tetap memiliki kekurangan yang ada pada MS-DOS. Windows NT yang juga tersedia untuk Digital Alpha selain *prosesor x86*. Namun Windows NT ini masih juga memiliki beberapa kekurangan yang telah ada pada MS-DOS.Waktu untuk menemukan suatu bug dalam suatu sistem operasi ini tak sebanding dengan harga yang harus dibayar. Sistem operasi Apple untuk Macintosh hanya dapat berjalan di sistem Mac. Juga memiliki kekurangan dari sisi

Ketersediaan perangkat bantu pengembang (*development tool*) dan juga kurang dapat secara mudah untuk berintoperasi dengan sistem operasi lainnya. Apple juga telah memungkinkan Linux dapat dijalankan pada Power

- **Kelebihan Linux**

  Di sini akan dijelaskan beberapa kelebihan dari sistem operasi Linux/UNIX dibandingkan dengan dengan sistem operasi yang lain. Dan

berikut ini adalah beberapa fakta dari hal-hal yang menguntungkan dengan menggunakan program dan *file-file* Linux/UNIX :

1. Pada dasarnya semua data tersimpan di dalam harddisk walau ada beberapa kondisi dimana data tersimpan di disket. Linux/UNIX memberikan beberapa proses spesial dimana terminal, printer dan device hardware lainnya dapat diakses seperti kita mengakses *file* yang tersimpan dalam harddisk atau disket.
2. Ketika program dijalankan, program tersebut dijalankan dari harddisk ke dalam RAM dan setelah dijalankan akan dinamakan sebagai proses.
3. Linux/UNIX menyediakan servis untuk membuat, memodifikasi program, proses dan *file*.
4. Linux/UNIX mendukung struktur *file* yang bersifat hirarki.
5. Linux/UNIX adalah salah satu sistem operasi yang termasuk ke dalam kelas sistem operasi yang dapat melakukan *multitasking*. Multitasking sendiri adalah keadaan dimana suatu sistem operasi dapat melakukan banyak kerjaan pada saat yang bersamaan.
6. Selain *multitasking*, Linux/UNIX juga dapat mendukung *multiuser*. Yaitu sistem operasi yang pada saat bersamaan dapat digunakan oleh lebih dari satu user yang masuk ke dalam sistem. Bahkan untuk Linux juga mendukung untuk *multiconsole* dimana pada saat bersamaan di depan komputer langsung tanpa harus melalui jaringan dan memungkinkan lebih dari satu user masuk ke dakam sistem.

2.1.4. Bagian Sistem Operasi Linux

Sistem Operasi Linux/UNIX terdiri dari kernel, program sistem dan beberapa program aplikasi. *Kernel* merupakan inti dari sistem operasi yang mengatur penggunaan *memori*, piranti masukan keluaran, proses-proses, pemakaian *file* pada *file system* dan lain-lain. *Kernel* juga menyediakan sekumpulan layanan yang digunakan untuk mengakses kernel yang disebut *system call*. *System call* ini digunakan untuk mengimplementasikan berbagai layanan yang dibutuhkan oleh sistem operasi. Program sistem dan semua program-program lainnya yang berjalan di atas kernel disebut user mode. Perbedaan mendasar antara program sistem dan program aplikasi adalah program sistem dibutuhkan agar suatu sistem operasi dapat berjalan sedangkan program aplikasi adalah program yang dibutuhkan untuk menjalankan suatu aplikasi tertentu. Contoh : daemon merupakan program sistem dan pengolah kata (*word processor*) merupakan program aplikasi.

2.1.5. Bagian Penting Kernel Linux

Kernel Linux terdiri dari beberapa bagian penting, seperti : manajemen proses, manajemen memori, *hardware device drivers*, *filesystem drivers*, manajemen jaringan dan lain-lain. Namun bagian yang terpenting ialah manajemen proses dan manajemen memori. Manajemen memori menangani daerah pemakaian memori, daerah *swap*, bagian-bagian *kernel* dan untuk *buffer cache*. Manajemen proses menangani pembuatan proses-proses dan penjadwalan proses.

2.2. Video Streaming

Video streaming merupakan bidang yang menarik untuk dijelajahi karena relatif baru dengan biaya yang cukup murah dengan semakin murahnya peralatan elektronik. Aplikasi dari *Video Streaming* salah satunya untuk *memonitoring* atau pemantauan kondisi ruangan, informasi video akan dikirmkan melalui saluran komunikasi, termasuk jaringan, kabel telepon, saluran ISDN atau radio. Informasi video mempunyai *bandwidth* yang lebar (sangat banyak *byte* per detik yang dikirimkan), yang oleh karenanya sangat membutuhkan teknologi kompresi video untuk mengurangi kebutuhan *bandwidth* sebelum dikirimkan melalui saluran komunikasi. Peralatan yang perlu ditambahkan hanya kamera video sederhana. Sekedar gambaran singkat, sebuah kanal video yang baik tanpa dikompresi akan mengambil *bandwidth* sekitar 9 Mbps. Dengan teknik kompresi yang sudah ada pada hari ini, kita dapat menghemat sebuah kanal video sekitar 30 Kbps. Itu berarti sebuah saluran *Internet* yang tidak terlalu cepat sebetulnya dapat digunakan untuk menyalurkan video.

Dengan banyaknya WARNET maupun RT/RW-Net yang menggunakan peralatan wireless *Internet* 2.4 GHz pada kecepatan 11 Mbps, bahkan sebagian mulai bereksperimen dengan kecepatan 54Mbps pada frekuensi 5.8GHz, sebetulnya banwidth yang ada pada infrastruktur jaringan *internet* lokal sebetulnya sudah cukup lebar, bahkan mungkin sangat lebar.

Beberapa hal yang perlu diperhatikan dalam pengiriman video adalah :

- Jika kita menggunakan video hitam-putih akan memakan *bandwidth* lebih kecil daripada jika kita melakukan konferensi menggunakan video berwarna.

- Jika kita menggunakan kecepatan pengiriman kecepatan pengiriman *frame per second* (fps) video yang rendah, akan memakan *bandwidth* yang lebih rendah dibandingkan frame per second (fps) yang tinggi.

Video yang cukup baik biasanya dikirim dengan kecepatan frame per second (fps) sekitar 30 fps. Jika dikirimkan tanpa kompresi, sebuah video dengan 30 fps akan mengambil *bandwidth* kira-kira 9Mbps, amat sangat besar untuk ukuran kanal komunikasi data.

2.3. TCP/IP

TCP/IP merupakan protokol jaringan komputer terbuka dan bisa terhubung dengan berbagai jenis perangkat keras dan lunak. TCP terdiri beberapa layer atau lapisan yang memiliki fungsi tertentu dalam komunikasi data. Setiap fungsi dari layer selain dapat bekerjasama dengan layer pada tingkat lebih rendah atau lebih tinggi, juga bisa berkomunikasi dengan layer sejenis pada remote *host* (*peering*). IP adalah jantung TCP/IP memiki peran sebagai pembawa data yang *independen*. IP dibagi atas kelas *network* A,B, dan C. Sedangkan kelas D untuk keperluan reverse IP yang boleh diabaikan. IP ditulis dalam bilangan desimal dari 0 sampai 255. Data yang mengalir antar layer atau antar *host* dienkapsulasi dan diberi header agar tiap layer bisa memprosesnya. Sebuah *host* tidak tahu alamat IP *gateway* di *network* lain, tetapi data mengalir ke *host* tujuan di *network* lain melalui *gateway network*nya setelah diberi penentuan ruting alamat IP.

TCP/IP adalah salah satu perangkat lunak jaringan komputer (*networking* software) yang terdapat dalam sistem UNIX, dan dipergunakan dalam banyak komunikasi data UNIX dalam *local area network* (LAN) maupun *Internet.*

Layanan dalam TCP/IP yang berbeda dikelompokkan menurut fungsi – fungsinya.*Protokol – protokol transport* mengendalikan pergerakan data antara dua mesin, dan mencakup :

1. TCP (*Transmission Control Protocol*)

   Protokol ini bersifat *connection-based* , artinya kedua mesin pengirim dan penerima tersambung dan berkomunikasi satu satu sama lain sepanjang waktu.

2. UDP (*User Datagram Protokol*)

   Protokol ini bersifat *connectionless* (tanpa koneksi), artinya dikirim tanpa kedua mesin penerima dan pengirim saling berhubungan. Ini seperti mengirim surat lewat kantor pos, surat dikirim oleh pengirim namun ia tidak pernah bisa tahu apakah surat tersebut sampai di tujuan atau tidak.

Sementara itu ada pula protokol – protokol *routing* untuk menangani pengalaman (*addressing*) data dan menentukan jalur terbaik untuk mencapai tujuan. Protokol – protokol tersebut juga bertanggung jawab memecah informasi ukuran besar dan menyusunnya kembali pada tujuan, protokol –protokol tersebut antara lain :

- IP (*Internet Protocol*) menangani transmisi data yang sebenarnya .
- ICMP (*Internet Control Message Protocol*) menangani informasi status untuk IP, seperti error (kesalahan) dan perubahan – perubahan dalam perangkat keras jaringan yang mempengaruhi *routing* (penentuan jalur).

- RIP (*Routing Information Protocol*) dan OSPF (*Open Shortest-Path First*) , yaitu satu dari berbagai protocol yang mempengaruhi metode *routing* terbaik untuk menyampaikan data.

TCP singkatan dari Transfer Control Protocol dan IP singkatan dari *Internet* Protocol. TCP/IP menjadi satu nama karena fungsinya selalu bergandengan satu sama lain dalam komunikasi data. TCP/IP saat ini dipergunakan dalam banyak jaringan komputer lokal (LAN) yang terhubung ke *Internet*, karena memiliki sifat:

1. Merupakan protokol standar yang terbuka, gratis dan dikembangkan terpisah dari perangkat keras komputer tertentu. Karena itu protokol ini banyak didukung oleh vendor perangkat keras, sehingga TCP/IP merupakan pemersatu perangkat keras komputer yang beragam merk begitu juga sebagai pemersatu berbagai perangkat lunak yang beragam merk sehingga walau anda memakai perangkat keras dan perangkat lunak komputer yang berlainan dengan teman anda pada jaringan komputer berbeda, anda dan teman anda dapat bermkomunikasi data melalui *Internet*.
2. Berdiri sendiri dari perangkat keras jaringan apapun. Sifat ini memungkinkan TCP/IP bergabung dengan banyak jaringan komputer. TCP/IP bisa beroperasi melalui sebuah Ethernet, sebuah token ring, sebuah saluran dial-up, sebuah X-25 dan secara vitrual melalui berbagai media fisik transmisi data.
3. Bisa dijadikan alamat umum sehingga tiap perangkat yang memakai TCP/IP akan memiliki sebuah alamat unik dalam sebuah jaringan komputer lokal, atau dalam jaringan kumputer global seperti *Internet*.
4. Protokol ini distandarisasi dengan skala tinggi secara konsisten, dan bisa memberikan servis kepada user-user di dunia.

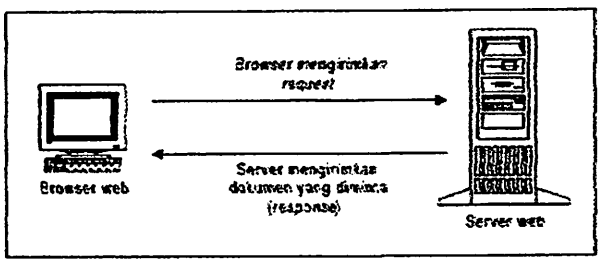
2.4. World Wide Web

Pada awalnya *internet* adalah sebuah proyek yang dimaksudkan untuk menghubungkan para ilmuan dan peneliti di Amerika, namun ini telah tumbuh menjadi media komunikasi global yang dipakai semua orang di muka bumi. Perumbuhan ini membawa beberapa masalah penting mendasar, dianataranya kenyataan bahwa *Internet* tidak diciptakan pada jaman *Gaphical User Interface* (*GUI*) seperti saat ini, *Internet* dimulai pada masa dimana orang masih menggunakan alat – alat akses yang tidak *user – friendly yaitu terminal* berbasis teks serta perintah – perintah *command line* yang panjang – panjang serta sukar diingat, sangat berbeda dengan komputer dewasa ini yang menggunakan klik tombol mouse pada layar grafik berwarna.

Kemudian orang berpikir untuk membuat sesuatu yang lebih baik. Popularitas *Internet* mulai berkembang pesat seperti jamur di musim penghujan setelah standar baru yaitu HTTP dan HTML diperkenalkan kepada masyarakat. HTTP (*Hypertext Trasfer Protocol*) membuat pengaksesan informasi melalui protokol TCP/IP menjadi lebih mudah dari sebelumnya. HTML (*Hypertext Markup Language*) memungkinkan orang menyajikan informasi yang secara visual lebih menarik. Pemunculan HTTP dan HTML kemudian membuat orang mengenal istilah baru dalam *internet* yang sekarang menjadi sangat populer, bahkan sedemikian populernya sehingga sering dianggap identik denga *internet* itu sendiri, yaitu *World Wide Web* (WWW).

Pada prinsipnya *World Wide Web* (singkatan cukup disebut "*web*" saja) bekerja dengan cara menampilkan *file–file* HTML yang berasal dari *web server* pada program *client* khusus, yaitu *web browser*. Program *browser* pada *client*

mengirimkan permintaan (*request*) kepada *web server*, yang kemudian akan dikirimkan oleh *server* dalam bentuk HTML. *File* HTML berisi instruksi – instruksi yang diperlukan untuk membentuk tampilan. Perintah –perintah HTML ini kemudian diterjemahkan oleh *web browser* sehingga isi informasinya dapat ditampilkan secara visual kepada pengguna di layar komputer.



Gambar 2-1

Konsep dasar browser dan web server

2.5. Hypertext Transfer Protocol (HTTP)

Web merupakan terobosan baru sebagai teknologi sistem informasi yang menghubungkan data dari banyak sumber dan layanan yang beragam macamnya di *internet*. Pengguna tinggal mengklikkan tombol mousenya pada link – link *hypertext* yang ada untuk melompat ke dokumen – dokumen di berbagai lokasi si *internet*. Link – linknya sendiri bisa mengacu kepada dokumen *web*., *server* FTP (*File Trasfer Protokol*), e-mail ataupun layanan – layanan lain.

Server dan *web browser* berkomunikasi satu sama lain dengan protokol yang memang dibuat khusus untuk ini, yaitu HTTP. HTTP bertugas menangani permintaan –permintaan (*request*) dari *browser* untuk mengambil dokumen – dokumen *web*.

HTTP bisa dianggap sebagai sistem yang bermodel *client-server*. *Browser web*, sebagai *client*nya, mengirimkan permintaan kepada *web server* untuk menentukan apakah dokumen yang diminta bisa dikirimkan kepada *browser* atau tidak.

HTTP bekerja di atas *Transmission Control Protocol* (TCP) yang menjamin sampainya data di tujuan dalam urutan yang benar. Bila suatu kesalahan terjadi selam proses pengiriman, pihak pengirim akan mendapat pemberitahuan bahwa telah terjadi ketidakberesan. Karenanya *server* dan *client* tidak harus menyediakan mekanisme untuk memeriksa kesalahan transmisi data, yang berarti mempermudah pekerjaan pemrograman. Namun demikian, HTTP tidak memiliki apa yang disebut *session*, seperti halnya FTP, yang menjaga hubungan antara *server* dan *client* secara konsisten. Setiap halaman *web* yang dikirim akan melibatkan satu proses penyambungan antara *client* dan *server*, baru kemudian datanya ditransfer. Setelah data selesai ditransfer, koneksi antara *server* dan *client* akan diputus. Sifatnya ini membuat HTTP sering disebut dengan istilah protokol *hit-and-run*.

Suatu halaman *web* seringkali berisi bebrapa *file* gambar, atau beberapa *file–file* lain. HTTP memaksa *server* untuk menjalin hubungan baru setiap kali hendak mengirim satu buah *file*. Ini tidak menguntungkan dan tidak efisien, mengingat proses hubung-putus-hubung semacam ini menyebabkan beban bagi jaringan.

Standar baru protokol HTTP, yaitu HTTP/1.1 yang baru-baru ini diperkenalkan, dirancang untuk mengatasi masalah di atas. *Web* diarahkan agar mengarah ke pengguna *persistent connection* (sambungan yang terjaga berkesinambungan) secara lebih efisien. Dalam HTTP/1.1, *server* tidak akan

memutuskan hubungan dengan *client* pada akhir pentransferan dokumen. Hubungan tetap terbuka untuk melayani bila saja ada *request* lagi dalam waktu yang singkat. Hubungan baru akan diputuskan bila setelah melewati suatu batas waktu tertentu (yang bisa ditentukan oleh administrator *server*) *client* tidak mengirimkan *request* lagi.

Keuntungan lain dari *persisten connection* adalah penggunaan *pipelining*. *Pipelining* adalah proses pengiriman *request* berikutnya segera setelah *request* sebelumnya dikirimkan tanpa menunggu balasan dari *server* terlebih dahulu. *server*nya tetap harus melayani setiap *request* secara berurutan, namun ini mengurangi waktu tunda antara setiap *request*. Hasilnya, data akan lebih cepat sampai di tujuan.

Standar HTTP/1.1 ini sekarang sudah mulai dimasyarakatkan dan banyak paket perangkat lunak *web server* komersial dan non-komersial yang sudah mendukung standar baru ini. *Browser-web browser* keluaran terbaru umumnya juga sudah mendukung HTTP/1.1.

2.6. Browser dan Web Server

Dalam dunia *web*, perangkat lunak *client*, yaitu *web browser* mempunyai tugas yang sama yaitu menterjemahkan informasi yang diterima dari *web server* dan menampilkannya pada layar komputer pengguna. Oleh karena HTTP memungkinkan *web server* mengirimkan beragam data, seperti teks atau gambar, *browser* harus bisa mengenali berbagai macam data yang akan diterimanya, dan selanjutnya harus tahu cara untuk menampilkannya dengan benar. Teks harus sitampilkan sebagai teks dan gambar harus ditampilkan sebagai gambar.

Umumnya *web browser* menerima data dalam bentuk HTML. *File* HTML sebenarnya adalah *file* teks biasa yang selain berisi informasi yang hendak ditampilkan kepada pengguna, juga mempunyai perintah – perintah untuk mengatur tampilan data tersebut, *Browser* lah yang memiliki kuasa penuh dalam menerjemahkan perintah – perintah tadi. Meskipun sudah dibuat konsessus untuk menstandarkan format dan elemen-elemen HTML, setiap jenis *browser* bisa menterjemahkan *file* HTML yang sama secara berbeda.

Pada awal pertama kalinya protokol-protokol dasar *web* dikembangkan yaitu sekitar awal 1990-an, *web browser* pertama yang diperkenalkan adalah Mosaic yang dibuat oleh *National Center for Supercomputing Applications* (NCSA) di Amerika Serikat. Mosaic dimaksudkan agar menjadi sebuah *interface* grafis yang mudah dipergunakan, yang dengan demikian diharapkan dapat mempercepat perkembangan dan dukunganumum akan *web*. Mosaic langsung dibuat untuk tiga macam *platform* berbeda, yaitu X Window (untuk lingkungan UNIX dan keluarganya), Microsoft Windows dan Macintosh. Mosaic inilah yang lalu dianggap sebagai legenda yang memacu revolusi *web* menjadi sedemikian populernya seperti sekarang ini.

Perkembangan jaman serta semakin populernya lingkungan GUI (*Graphical User Interface*) membuat banyak orang sekarang berlomba-lomba membuat program *browser* yang menarik serta mudah dipakai. *Browser-web browser* modern dilengkapi dengan fasilitas-fasilitas yang mendukung tampilan multimedia berupa *audio* (suara), animasi 3 dimensi, bahkan video. Program *web browser* yang paling terkenal saat ini adalah Netcape Navigator dan Microsoft *Internet* Explorer.

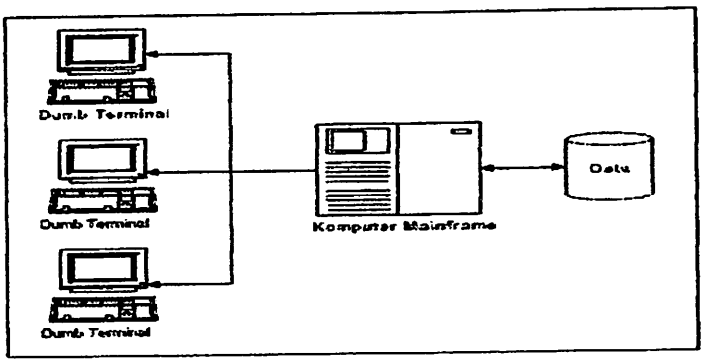
Sementara itu *web server* pada dasarnya adalah perangkat lunak khusus yang bertugas melayani permintaan-permintaan dari *web browser* akan dokumen-dokumen yang tersimpan di dalamnya. Perangkat lunak *web server* sekarang tersedia untuk berbagai macam *platform* dan lingkungan sistem operasi untuk lingkungan UNIX, yang paling populer adalah Apache, Netcape FastTrack, O'Reilly *Website* dan banyak lagi. Sistem operasi jaringan Novell Netware pun memiliki suatu modul *add-on* yang berfungsi sebagai *web server*, yang bisa dijalankan pada saat *startup* jaringan.

Beberapa perangkat lunak *web server* mempunyai *feature* seperti *server side programming, security control* dan lain sebagainya. Meskipun beragam macamnya, secara fungsional semua jenis *web server* adalah sama saja, yaitu berfungsi melayani permintaan-permintaan dari *web browser*.

2.7. Internet Sebagai Sistem Client Server

Istilah *client/server* dewasa ini telah demikian populer. Keuntungan utama dari sistem berbasis *client/server* adalah bahwa perangkat keras dan perangkat lunak bisa ditempatkan di mana saja mereka bisa bekerja secara lebih optimal. Dulu, di jaman komputer *mainframe*, komputer *mainframe*-lah yang menjadi pusat kendali dan mengerjakan semua proses komputasi. Pengguna berinteraksi dengan sistem *mainframe* melalui terminal – terminal yang dibutuhkan secara langsung ke komputer *mainframe*. Terminal-terminal ini tidak punya kemampuan pmrosesan sama sekali, dan oleh karena itu disebut "terminal dungu" (*dumb terminal*). Termianl dungu tidak lebih dari sekedar perpanjangan kabel untuk

keyboard dan layar monitor, dan hanya berfungsi sebagai alat untuk memasukkan dan melihat data saja.



Gambar 2-2

Konsep Sistem Mainframe

Definisi yang banyak dipergunakan untuk menjelaskan sistem berbasis *client/server* adalah "sistem yang memisahkan antara tugas-tugas komputasi antara proses-proses *client* dan *serve* ". Dengan sistem *client/server*, kekuatan pemrosesan bisa disebarkan (didistribusikan) ke banyak mesin *client* dan mesin *server* yang terpisah secara fisik (itu sebabnya disebut *distributed system*).

Misalnya sebuah *web server* yang mengambil informasi dari *database* menampilkan hasilnya pada *client* dengan menggunakan *broser web*. *web server* dan *database* bisa saja ditempatkan pada sutu mesin saja, namun apabiala jumlah *client* yang melakukan akses ke *server* semakin banyak dan melebihi kapasitas mesin *server*, perangkat lunak *database* dan *web server* bisa saja dipisahkan dan ditempatkan di mesin kedua, ketiga atau bahkan lebih. Dengan begitu pemrosesan pada sisi *server* dapat disebarkan ke beberapa mesin, yang memungkinkan

efisiensi komputasi. Begitu pula, dengan cara ini, kapasitas *server* bisa dikembangkan dan ditingkatkan sesuai dengan kebutuhan.

Dari susut pandang lain, sistem berbasis *client/server* juga bisa memanfaatkan *web browser* untuk meringankan kerja *server*. Tugas menampilkan informasi dan menyediakan tampila pengguna (user interface) tidak perlu dilakukan secara langsung oleh *server*, namun diserahkan sepenuhnya kepada *web browser*. Dengan hadirnya teknologi pemrograman *client-side* (yang dijalankan di sisi *client*) seperti Java dan bahasa *client-scripting* seperti Javascript, fungsi – fungsi lain seperti pemeriksaan /validasi input bisa dilakukan oleh *bwoser* sebelum data dikirimkan kepada *server*, menjamin data yang dikirikkan ke *server* tidak keliru. Hal ini mempercepat kerja *server*, karena hanya mengerjakan tugas-tugas yang berguna.

Konsep dasar sistem berbasis *clint-server* adalah *balancing* (penyebaran), yaitu proses yang mencegah suatu prosesor mengalami overload (terbeban lagi) sementara mesin lainnya justru menganggur .

Jadi, setidaknya dalam teori, sistem berbasis *client/server* memberikan keuntungan yang banyak seperti penggunaan resourcesecara lebih efisien, penyimpanan data terpusat, serta lalu lintas di dalam jaringan menjadi lebih rendah (dibandingkan dengan sistem yang seluruhnya terpusat). Satu-satunya kelemahan utama dari sistem berbasis *client/server* adalah manajemen dan perawatan mesin-mesin *client* yang membutuhkan *upgrading* serta proses konfigurasi yang memakan waktu dan tenaga. Semua program *client*, misalnya front-end untuk suatu *database*, harus dipasang satu persatu di setiap komputer

client, dan apabila pada suatu saat program *client* tersebut harus diubah atau dikembangkan, prosesnya harus diulang di setiap komputer *client*.

Bagaimana dengan *Internet*, dalam hal ini *web*? Sebenarnya, secara mudah kita bisa mengatakan bahwa konsep *web* pada awalnya bisa dipandang mirip dengan konsep jaringan dengan *dump terminal*; pengguna *web browser* mengirimkan permintaan ke *web server*, lalu menerima informasi dari *server* berupa dokumen statis yang oleh *browser* hanya perlu ditampilkan saja ke layar. Meskipun kommputer *client* memiliki prosesor sendiri, memori sendiri serta media penyimpanan sendiri, sifat-sifat alami *web browser* sebenarnya memenuhi syarat-syarat sebagai *dumb terminal*. Oleh karana *web browser* mempunyai kemampuan untuk membentuk *user interface* (tampilan pengguna) yang sifatnya grafis, sebagian orang menyebutnya sebagai *"smart dumb terminal"* (terminal dungu yang cerdas, atau tidak terlalu dungu).

Struktur fisik dari *Internet* sendiri juga tidak berbeda jauh dari sistem *mainframe*. *Bandwidth* (lebar jalur) besarnya terbatas. Tidak seperti teknologi lokal (LAN) seperti Ethernet atau Token Ring yang bisa membawa informasi sampai 10 bahkan 100 Mbps, kebanyakan pengguna *Internet* masih terbatas pada sambungan berorde kilobit per detik. Perusahaan-perusahaan besar mungkin sanggup menyewa sambungan kapasitas besar seperti T1 (1,55 Mbps) ke *internet*, namun kebanyakan pengguna yang lebih kecil aatau rumah tangga hanya mempunyai sambungan berkecepatan rendah, sekitar 33 – 56 Kbps. Bahkan teknologi ISDN pun (yang masih saat ini masih tergolong mahal) jarang melampaui 128 Kbps.

Melihat kenyatan bahwa pada mesin-mesin yang menjalankan *browser* pada umumnya adalah komputer-komputer modern yang memiliki potensi pemrosesan dan penyimpanan data yang cukup besar, mesin-mesin *client* seharusnya bisa mengerjakan lebih banyak pekerjaan lagi. Juga, tidak seperti sistem berbasis *client/server* "tradisional", kenyatan bahwa penggunaan *web browser* nyaris tidak membutuhkan perawatan yang mahal, *upgrade* berkala dan konfigurasi yang rumit, membuat *web* (dan *Internet* pada umumnya) menjadi tempat yang menarik untuk mengembangkan sistem berbasis *client/server*.

Dari sisi pengelola jaringan, sistemberbasis *web* mengundang banyak perhatian karena sisrem semacam ini bisa mengatasi banyak kekurangan dari sistem-sitem tradisional. Pada intinya, mengembangkan sistem *client/server* di *web* membawa keuntungan-keuntungan langsung seperti :

1. **Tidak ada masalah distribusi program**

   Pendistribusian berlangsung secara sendirinya, karan setiap salinan dokumen (sebagai satu komponen aplikasi) di-*download* ke mesin *client* setiap saat mesin *client* membutuhkan dan meminta *update* atau salinan yang lebih baru. Tidak perlu lagi seorang administrator jaringan meng-*install* perangkat lunak *client* untuk setiap komputer yang ada di organisasinya, suatu pekerjaan yang melelahkan dan memboroskan waktu.

2. **Efisien**

   Distribusi otomatis dan tidak perlunya instalasi untuk setiap *client* jelas mempermudah perawatan dan *updating* aplikasi. Perubahan-perubahan pada aplikasi bisa dikerjakan secara terpusat dan bisa langsung diterapkan tanpa

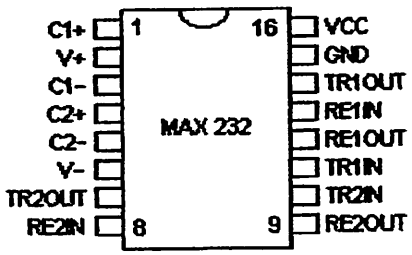
perlu menyesuaikan semua *client*. Bahkan tampilan pada pengguna bisa diubah – ubah secara berkala sisesuaikan dengan waktu dan situasi.

3. **Fleksibel**

   Web browser tersedia untuk hampir semua macam *platform* mesin dan sistem operasi, baik itu mesin Windows berbasis prosesor Intel, mesin UNIX berbasis prosesor RISC, ataupun komputer Macintosh. Fleksibilitas aplikasi *web* lebih terjamin, karena tidak perlu lagi mengembangkan program-program *client* yang berbeda untuk tiap macam *platform*.

2.8. Rangkaian Pengubah Level Tegangan

Kita menggunakan RS232 untuk komunikasi dengan komputer *server* secara serial, untuk itu *mikrokontroller* memerlukan sebuah piranti yang berfungsi sebagai pengubah level tegangan. RS232 menggunakan level/karakteristik elektrik yang berbeda dengan level TTL. RS232 bekerja pada level tegangan +3 s/d +25 Volt untuk *space (logic 0)* dan -3 s/d -25 Volt untuk *mark (logic 1)*. Sedangkan TTL bekerja pada level tegangan -5 s/d +5 Volt. Piranti tambahan yang kita butuhkan adalah IC MAX232. Pada dasarnya IC ini hanya digunakan sebagai pengubah level tegangan ke level *Transistor Transistor Logic* (TTL), tidak berfungsi sebagai pengkodean sinyal yang melewati RS232, dan juga tidak mengonversikan data serial ke paralel.

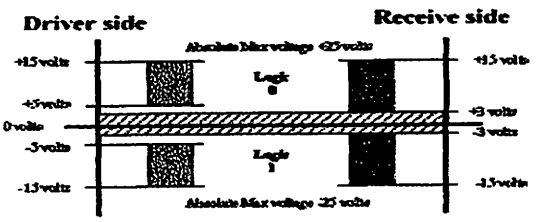


Gambar 2-3

IC MAX232 sebagai pengubah level tegangan

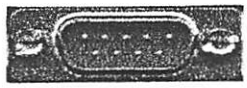
RS232 sebagai komunikasi serial mempunyai 9 pin yang memiliki fungsi masing-masing. Pin yang biasa digunakan adalah pin 2 sebagai *received data*, pin 3 sebagai *transmited data*, dan pin 5 sebagai *ground signal*. Karakteristik elektrik dari RS232 adalah sebagi berikut :

- *Space (logic 0)* mempunyai level tegangan sebesar +3 s/d +25 Volt.
- *Mark (logic 1)* mempunyai level tegangan sebesar -3 s/d -25 Volt.
- Level tegangan antara +3 s/d -3 Volt tidak terdefinisikan.
- Arus yang melalui rangkaian tidak boleh melebihi dari 500 mA., ini dibutuhlan agar sistem yang dibangun bekerja dengan akurat.



Gambar 2-5

Karakteristik Elektrik RS232



Gambar 2-6

RS232 (Sebagai Komunikasi Serial)

Tabel 2-1

Fungsi masing-masing pin RS232

RS232 Pin Assignments (DB9 PC signal set)	
Pin 1	*Received* Line Signal Detector (Data Carrier Detect)
Pin 2	*Received* Data
Pin 3	Transmit Data
Pin 4	Data Terminal Ready
Pin 5	Signal Ground
Pin 6	Data Set Ready
Pin 7	Request To Send
Pin 8	Clear To Send
Pin 9	Ring Indicator

2.9. Motor Stepper

Motor stepper adalah salah satu tipe motor yang sangat populer digunakan sebagai penggerak/pemutar peralatan industri. Prinsip kerja motor stepper ini mirip dengan motor DC, yaitu sama-sama dicatu dengan tegangan DC untuk memperoleh medan magnet. Bila motor DC memiliki magnet tetap pada stator, motor stepper mempunyai magnet tetap pada rotor. Suatu motor stepper biasanya cukup dinyatakan dengan spesifikasi : "berapa phasa ", "berapa derajat perstep", "berapa volt tegangan catu untuk tiap lilitan" dan berapa ampere/miliampere arus yang dibutuhkan untuk tiap lilitan". Walau bagaimanapun motor stepper jauh berbeda dengan motor DC. Motor stepper tidak dapat bergerak dengan sendirinya. Motor stepper bergerak secara per-step sesuai dengan spesifikasinya, dan bergerak dari satu step ke step berikutnya memerlukan waktu. Juga ada perbedaan pada *torque-speed* antara motor stepper dan motor DC. Secara umum motor DC tidak menghasilkan torsi yang besar pada kecepatan rendah, sebaliknya motor stepper dapat menghasilkan torsi yang besar pada kecepatan rendah. Motor stepper juga memiliki karakteristik yang lain yaitu *holding torque*, yang tidak dimiliki oleh motor DC. *Holding torque* memungkinkan motor stepper dapat menahan posisinya ketika tidak berputar. Hal ini sangat berguna untuk aplikasi dimana suatu sistem memerlukan keadaan *start* dan *stop*.

Motor stepper tidak merespon sinyal *clock*, motor stepper mempunyai beberapa lilitan dimana lilitan-lilitan tersebut harus dicatu (tegangan) dahulu dengan suatu urutan tertentu agar dapat berotasi. Membalik urutan pemberian tegangan tersebut akan menyebabkan putaran motor stepper yang berbalik arah. Jika sinyal kontrol tidak terkirim sesuai dengan perintah maka motor stepper tidak

akan berputar secara tepat, mungkin hanya akan bergetar dan tidak bergerak. Untuk mengontrol motor stepper kita menggunakan suatu rangkaian driver yang menangani kebutuhan arus dan tegangan.

Karakteristik dari motor stepper adalah sebagai berikut:

- *Voltage*

  Tiap motor stepper mempunyai tegangan rata-rata yang biasanya tertulis pada tiap unitnya atau tercantum pada datasheet masing-masing motor stepper. Tegangan rata-rata ini harus diperhatikan dengan seksama karena bila melebihi dari tegangan rata-rata ini akan menimbulkan panas yang terlalu besar pada motor stepper yang menyebabkan kinerja putarannya tidak maksimal atau bahkan motor stepper akan rusak dengan sendirinya.

- *Resistance*

  Resistance-per-winding adalah karakteristik yang lain dari motor stepper. *Resistance* ini akan menentukan arus yang mengalir, selain itu juga akan mempengaruhi torsi dan kecepatan maksimum dari motor stepper.

- *Degrees per step*

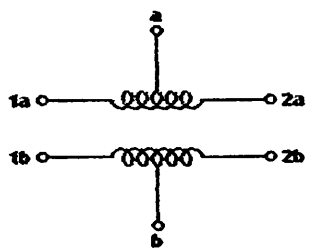
  Derajat per step adalah faktor terpenting dalam pemilihan motor stepper sesuai dengan aplikasinya. Tiap-tiap motor stepper mempunyai spesifikasi masing-masing, antara lain: 0.72^0 per step, 1.8^0 per step, 3.6^0 per step, 7.5^0 per step, 15^0/step, dan bahkan ada yang 90^0 per step. Dalam pengoperasiannya kita dapat menggunakan 2 prinsip yaitu *full-step* atau *half-step*. Dengan *full-step* berarti motor stepper aan berputar sesuai dengan spesifikasi derajat per stepnya, sedangkan *half-step* berarti motor

stepper akan berputar setengah derajat/step dari spesifikasi motor stepper tersebut.

Ada beberapa macam motor stepper. Hal ini dibedakan menjadi dua kategori besar yaitu: *permanent magnet* dan *variable reluctance*. Tipe *permanent magnet* terbagi menjadi dua motor stepper yaitu *unipolar stepper motor* dan *bipolar stepper motor*.

Unipolar stepper motor sangat mudah untuk dikontrol. Dengan rangkaian counter '*-n*' yang sedehana dapat mengontrol putaran dari motor stepper dengan urutan yang benar. *Unipolar stepper motor* mempunyai karakteristik khusus yaitu berupa lilitan *center-tapped* dan 1 lilitan sebagai *common*. Lilitan *common* akan mencatu tegangan pada *center-tapped* dan sebagai ground adalah rangkaian *driver* nya.

Unipolar stepper motor dapat dikenali dengan mengetahui adanya lilitan *center-tapped*. Jumlah phase dari motor stepper adalah dua kali dari jumlah *coil* nya. Biasanya pada *unipolar stepper motor* terdapat dua buah *coil*.

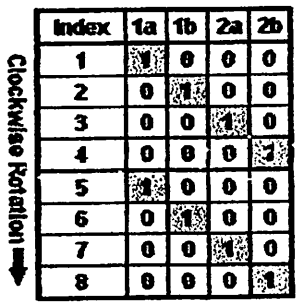


Gambar 2-7

Susunan coil unipolar motor stepper

Tabel 2-2

Pola 1-phase putaran unipolar motor stepper



Index	1a	1b	2a	2b
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

Pada prinsipnya ada dua macam cara kerja *unipolar* stepper motor, yaitu *full-step* dan *half-step*. Terlihat pada tabel 2.3 dan tabel 2.2.

Tabel 2-3

Pemberian Tegangan untuk Bekerja FullStep

*FULL*STEP								
	Tegangan yang diberikan pada lilitan							
	searah jarum jam				melawan jarum jam			
	L3	L2	L1	L0	L3	L2	L1	L0
1	1	0	0	0	0	0	0	1
2	0	1	0	0	0	0	1	0
3	0	0	1	0	0	1	0	0
4	0	0	0	1	1	0	0	0

Pada *full*step, suatu titik pada sebuah kutub magnet di rotor akan kembali mendapat tarikan medan magnet stator pada lilitan yang sama setelah step ke 4. berikutnya dapat diberikan lagi mulai dari step 1. Setiap step, rotor bergerak searah atau berlawanan dengan jarum jam sebesar spesifikasi derajat per step dari mtor stepper. Setiap step hanya menarik sebuah kutub saja. Tegangan '1' adalah menunjukkan logika dalam level *Transistor Transistor Logic* (TTL). Besar tegangan sesungguhnya diatur dengan spesifikasi stepper motor yang dipakai.

Tabel 2-4

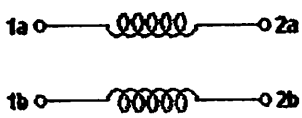
Pemberian Tegangan untuk Bekerja Half Step

	HALF STEP							
	Tegangan yang diberikan pada lilitan							
	Arah putar searah jarum jam				Arah putar melawan jarum jam			
	L3	L2	L1	L0	L3	L2	L1	L0
1	1	0	0	0	0	0	0	1
2	1	1	0	0	0	0	1	1
3	0	1	0	0	0	0	1	0
4	0	1	1	0	0	1	1	0
5	0	0	1	0	0	1	0	0
6	0	0	1	1	1	1	0	0

7	0	0	0	1	1	0	0	0
8	1	0	0	1	1	0	0	1

Untuk half step, setiap kutub magnet pada rotor akan kembali mendapatkan tarikan dari medan magnet lilitan yang sama setelah step ke 8. berikutnya kembali mulai step 1. Setiap step posisi rotor berubah sebesar setengah derajat dari spesifikasi derajat per step motor stepper.

Berbeda dengan *unipolar stepper motor*, *bipolar stepper motor* sangat sulit dalam pengontrolannya. Motor stepper jenis ini memerlukan rangkaian *driver* yang kompleks. Keuntungan *bipolar* motor stepper adalah ukurannya yang besar dan dapat menghasilkan torsi yang besar daripada *unipolar* motor stepper. *Bipolar* motor stepper di desain dengan coil yang terpisah yang akan di catu dari dua arah (polaritas harus dibalik selama pencatuan). *Bipolar* motor stepper menggunakan logika yang sama seperti *unipolar* motor stepper yaitu hanya '0' dan '1' untuk merespon coilnya.



Gambar 2-8

Susunan coil bipolar motor stepper

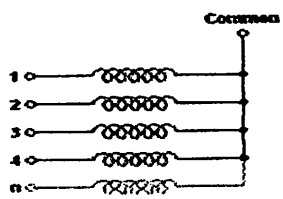
Tabel 2-5

Pola phase bipolar motor stepper

Index	1a	1b	2a	2b
1	+	–	–	–
2	–	+	–	–
3	–	–	+	–
4	–	–	–	+
5	+	–	–	–
6	–	+	–	–
7	–	–	+	–
8	–	–	–	+

Clockwise Rotation ⬇

Variable reluctance stepper motor, juga sering disebut sebagai *hybrid motor*. Motor stepper jenis ini sangat mudah sekali untuk dikontrol jika dibandingkan dengan jenis motor steper yang lain. Rangkaian driver untuk mencatu tiap-tiap lilitannya sangatlah sederhana. Prinsip kerjanya adalah driver mencatu tiap lilitan secara bergantian. Jika kita putar dengan tangan, kelihatan motor stepper ini seperti motor DC, berputar sangat bebas dan tidak terasa adanya step. Motor stepper jenis ini tidak memiliki magnet permanen serperti pada *unipolar* dan *bipolar* motor stepper.



Gambar 2-9

Susunan coil variable reluctance motor stepper

Tabel 2-6

Pola Phase variable reluctance stepper motor

Clockwise Rotation

Index	1	2	3	4
1	1	0	0	0
2	0	1	0	0
3	0	0	1	0
4	0	0	0	1
5	1	0	0	0
6	0	1	0	0
7	0	0	1	0
8	0	0	0	1

2.10. Driver Motor Stepper

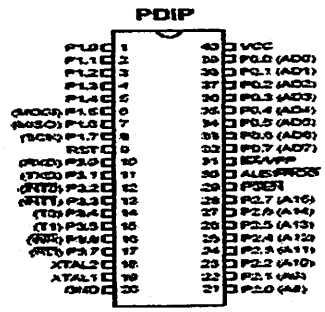
Secara teoritis, sebuah motor stepper dapat digerakkan langsung oleh *mikrokontroller*. Dalam kenyataannya, arus dan tegangan yang dikeluarkan oleh *mikrokontroller* terlalu kecil untuk menggerakkan sebuah motor stepper. Gerbang-gerbang *Transistor Transistor Logic* (TTL) *mikrokontroller* hanya mampu mengeluarkan arus dalam orde *mili-ampere* dan tegangan antara 2 sampai 2,5 Volt. Sementara itu untuk menggerakkan motor stepper diperlukan arus yang lebih besar (dalam orde ampere) dan tegangan berkisar 5 sampai 24 Volt.

Untuk mengatasi masalah tersebut, diperlukan sebuah piranti tambahan yang memenuhi kebutuhan arus dan tegangan yang cukup besar. Rangkaian driver motor stepper merupakan rangkaian "*open collector*", dimana output rangkaian ini terhubung dengan *ground* untuk mencatu lilitan-lilitan motor stepper.

2.11. MIKROKONTROLLER AT89S51 SEBAGAI ANTARMUKA

AT89S51 adalah *mikrokontroller* keluaran Atmel dengan 4K *byte* Flash PEROM *(Programmable and Erasable Read Only Memory)*, AT89S51 merupakan memori dengan teknologi *nonvolatile memory*, isi memori tersebut dapat diisi ulang ataupun dihapus berkali-kali.

Memori ini biasa digunakan untuk menyimpan instruksi (perintah) berstandar MCS-51 code sehingga memungkinkan *mikrokontroller* ini untuk bekerja dalam mode *single chip operation* (mode operasi keping tunggal) yang tidak memerlukan *external memory* (memori luar) unruk menyimpan source code tersebut.



Gambar 2-10

Konfigurasi Pin Mikrokontroller AT89S51

2.11.1. Deskripsi Mikrokontroler AT89S51

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

  Port 0 dapat berfungsi sebagai I/O biasa, *low order multiplex addres/data* ataupun menerima kode bye pada saat *Flash Programming*. Pada saat

sebagai I/O biasa port ini dapat memberikan *output sink* ke delapan buah *Transistor Transistor Logic* (TTL) input atau dapat diubah sebagai input dengan memberikan logika 1 pada port tersebut.

- Port 1, yaitu pin p1.0...p1.7

  Port 1 berfungsi sebagai I/O biasa atau menerima *low order address bytes* selama pada saat *Flash Programming*. Port ini mempunyai internal pull updan berfungsi sebagai input dengan memberikan logika 1. Sebagai output port ini dapat memberikan *output sink* keempat buah input TTL. Fasilitas khusus dari port 1 ini adalah adanya *In-System Programming*, yaitu port 1.5 sebagai MOSI, port 1.6 sebagai MISO, port 1.7 sebagai SCK.

- Port 2, yaitu mulai pin p2.0...p2.7

  Port 2 berfungsi sebagai I\O biasa atau *high order address*, pada saat mengakses memori secara 16 bit (Movx @DPTR). Pada saat mengakses memori secara 8 bit (Mov @Rn), port ini akan mengeluarkan sisi dari *Special Function Register*. Port ini mempunyai pull up dan berfungsi sebagai input dengan memberikan logika 1. Sebagai output, port ini dapat memberikan output sink keempat buah input TTL.

- Pin 3.0, sebagai RXD *(Port Serial Input)*.
- Pin 3.1, sebagai TXD *(Port Seial Output)*.
- Pin 3.2, sebagai INT0 *(Port External Interupt 0)*.
- Pin 3.3, sebagai INT1 *(Port External Interupt 1)*.
- Pin 3.4, sebagai T0 *(Port External Timer 0)*.
- Pin 3.5, sebagai T1 *(Port External Timer 1)*.

- Pin 3.6, sebagai WR *(External Data Memory Write Strobe).*
- Pin 3.7, sebagai RD *(External Data Memory Read Strobe).*
- Pin 9, sebagai RST

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

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

  Pin ini berfungsi pada saat mengeksekusi program yang terletak pada memori eksteranal. PSEN akan aktif dua kali setiap cycle.
- Pin 31, Sebagai EA/VPP

  Pada kondisi low, pin ini akan berfungsi sebagai EA yaitu *mikrokontroller* akan menjalankan program yang ada pada memori eksternal setelah sistem di reset. Jika berkondisi high, pin ini akan berfungsi untuk menjalankan program yang ada pada memori internal. Pada saat Flash Programming pin ini akan mendapat tegangan 12 Volt (VPP).
- Pin 19, sebagai XTALL1 *(Input Oscillator).*
- Pin 18, sebagai XTALL2 *(Output Oscillator).*

2.11.2. Struktur Memori

AT89S51 mempunyai stuktur memori yang terdiri atas :

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

AT89S51 mempunyai struktur memori yang terpisah antara RAM Internal dan Flash PEROM nya. RAM *Internal* dialamati oleh *RAM Address Register* (*Register* Alamat RAM) sedangkan *Flash PEROM* yang menyimpan perintah-perintah MCS-51 dialamti oleh *Program Adderss Register* (Register Alamat Program). Dengan adanya struktur memori yang terpisah tersebut, walaupun RAM Internal dan Flash PEROM mempunyai alamat yang sama, yaitu alamat 00, namun secara fisiknya kedua memori tidak saling berhubungan.

2.11.3. Antarmuka Serial

Port serial pada AT89S51 bersifat dupleks-penuh atau *fullduplex*, artinya port serial bisa menerima dan mengirim secara bersamaan. Selain itu juga memiliki penyangga penerima, artinya port serial mulai bisa menerima *byte* yang kedua sebelum *byte* pertama dibaca oleh register penerima (jika sampai *byte* yang kedua selesai diterima sedangkan *byte* pertama belum juga dibaca, maka salah

satu *byte* akan hilang). Penerimaan dan pengiriman data port serial melalui register SBUF. Penulisan ke SBUF berarti mengisi register pengiriman SBUF sedangkan pembacaan dari SBUF berarti membaca register penerimaan SBUF yang memang terpisah secara fisik (secara perangkat lunak namanya menjadi satu yaitu SBUF).

Port serial pada AT89S51 bisa digunakan dalam 4 mode kerja yang berbeda. Dari 4 mode tersebut, 1 mode diantaranya bekerja secara sinkron dan 3 lainnya bekerja secara asinkron. Keempat mode kerja tersebut adalah :

- Mode 0 : Mode ini bekerja secara sinkron, data serial dikirim dan diterima melalui kaki P3.0 (RxD), sedangkan kaki P3.1 (TxD) dipakai untuk menyalurkan detak pendorong data serial yang dibangkitkan AT89S51. Data dikirim/diterima 8 bit sekaligus, dimulai dari bit yang bobotnya paling kecil atau LSB (bit 0) dan diakhiri dengan bit yang bobotnya paling besar atau MSB (bit 7). Kecepatan pengiriman data (*baudrate*) adalah 1/12 frekuensi kristal yang digunakan.
- Mode 1 : Pada mode ini tetap yaitu, data dikirim dan diterima melalui kaki P3.0 (RxD), secara asinkron (juga mode 2 dan 3). Pada Mode 1 data dikirim/diterima 10 bit sekaligus, diawali dengan 1 bit *start*, disusul dengan 8 bit data yang dimulai dari bit yang bobotnya paling kecil (bit 0), diakhiri dengan 1 bit *stop*. Pada AT89S51 yang berfungsi sebagai penerima bit *stop* adalah RB8 dalam register SCON. Kecepatan pengiriman data (*baud rate*) bisa diatur sesuai dengan keperluan. Mode

inilah (mode 2 dan 3) yang umum dikenal sebagai UART (*Universal Asynchronous Receiver/Transmiter*).

- Mode 2 : Data dikirim/diterima 11 bit sekaligus, diawali dengan 1 bit *start*, disusul 8 bit data yang dimulai dari bit yang bobotnya paling kecil (bit 0), kemudian bit ke 9 yang bisa diatur lebih lanjut, diakhiri dengan 1 bit *stop*. Pada AT89S51 yang berfungsi sebagai penerima, bit 9 ditampung pada bit RB8 dalam register SCON, sedangkan bit *stop* diabaikan tidak ditampung. Kecepatan pengiriman data (*baud rate*) bisa dipilih antara 1/32 atau 1/64 frekuensi kristal yang digunakan.
- Mode 3 : mode ini sama dengan mode 2, hanya saja kecepatan pengiriman data (*baud rate*) bisa diatur sesuai dengan keperluan, seperti halnya Mode 1.

Pada mode asinkron (mode 1, mode 2, mode 3), port AT89S51 bekerja secara *fullduplex*.

2.11.3.1. Register Kontrol Port Serial

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

SCON - Serial Port Control Register

	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
	SM0	SM1	SM2	REN	TB8	RB8	TI	R1
RESET	0	0	0	0	0	0	0	0

Gambar 2-11

Register SCON

Keterangan:

- SM0

  Serial port mode bit 0, bit pengubah mode serial.

- SM1

  Serial port mode bit 1, bit pengatur mode serial.

- SM2

  Serial port mode bit 2, bit untuk mengaktifkan komunikasi multiprosessor pada kondisi set.

- REN

  Receive Enable, bit untuk mengaktifkan penerimaan data dari port serial pada kondisi set.

- TB8

  Transmit bit 8, bit ke-9 yang akan dikirim pada mode 2 atau mode 3.

- RB8

  Receive bit 8, bit ke-9 yang diterima pada mode 2 atau mode 3. Pada mode 1 bit ini berfungsi sebagai *stop* bit.

- TI

  Transmit Interupt Flag, bit yang akan di set pada akhir pengiriman karakter.

- RI

  Receive Interupt Flag, bit yang akan diset pada akhir penerimaan karakter.

Tabel 2-7

Register Kontol Port Serial

SM0	SM1	MODE	Keterangan	Baur Rate
0	0	0	Register Geser	Tetap (fosc/12)
0	1	1	UART 8-bit	Bisa diubah-ubah(dengan Timer)
1	0	2	UART 9-bit	Tetap(fosc/64 atau fosc/32)
1	1	3	UART 9-bit	Bisa diubah-ubah(dengan timer)

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

- Pada mode 1, RB8 dipakai untuk menampung bit *stop* yang diterima, dengan demikian apabila RB8 bernilai '1' maka data diterima dengan benar, sebaliknya apabila RB8 = '0' berarti terjadi kesalahan *frame* (*framing error*). Kalau bit SM2(bit 5) bernilai '1' pada mode 1, jika terjadi kesalahan frame, RI tidak akan menjadi '1' (aktif) meskipun SBUF sudah berisi data dari port serial (bit *stop* diterima dengan benar).

  Bit ke 9 ini bisa dipakai sebagai bit paritas, hanya saja bit paritas yang dikirim harus ditentukan sendiri dengan program dan diletakkan pada TB8 dan bit paritas yang diterima pada RB8 dipakai untuk menentukan integritas data secara program pula. Tidak seperti dalam UART standart. Semuanya dikerjakan oleh perangakat keras dalam IC UART.
- Bit T1 (bit 1) merupakan sinyal yang setara dengan sinyal *Transmiter Holding Register Empty* (THRE) yang umum dijumpai pada UART standard. Setelah port seial selesai mengirim data yang tersimpan dalam SBUF, bit T1 akan bernilai '1' dengan sendirinya, kemudian bit ini harus di-nol-kan dengan program agar bisa dipakai untuk memantau keadaan SBUF dalam pengiriman data berikutnya.
- Bit RI (bit0) merupakan sinyal yang setara dengan sinyal RDA (*Receiver Data Available*) yang umum dijumpai pada UART standart. Setelah SBUF menerima data dari port serial, bit RI akan bernilai '1' dengan sendirinya, bit ini harus di-nol-kan dengan program agar bisa dipakai untuk memantau keadaan SBUF dalam penerimaan data berikutnya.

2.11.3.2. Pengaturan Baud Rate Port Serial

Baud rate dari port serial dapat diatur pada mode 1 dan mode 3, namun pada Mode 0 dan Mode 2 *baud rate* tersebut mempunyai kecepatan yang permanen yaitu untuk mode 0 adalah 1/12 frekwensi osilator dan mode 2 adalah 1/64 frekwensi osilator.

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

Tabel 2-8

Rumus Penghitungan Baudrate pada Komunikasi Serial

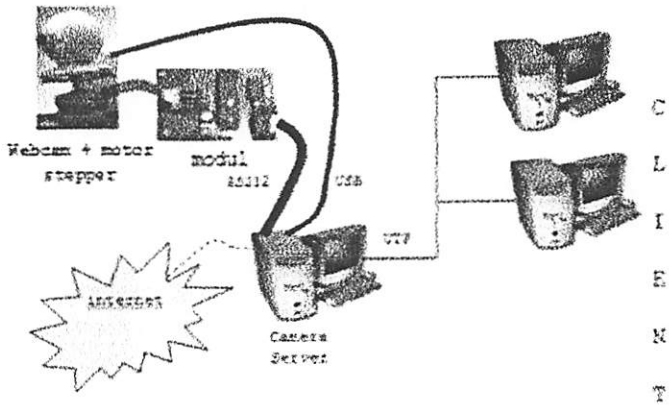
Mode	Baud Rate	
0	1/12 f osilator	
1	SMOD = 0 $Baudrate = \frac{fosc}{12x[256-TH1]x32}$	SMOD = 1 $Baudrate = \frac{fosc}{12x[256-TH1]x16}$
2	1/32 f	1/32 f
3	$Baudrate = \frac{fosc}{12x[256-TH1]x32}$	$Baudrate = \frac{fosc}{12x[256-TH1]x16}$

BAB III

PERENCANAAN DAN PEMBUATAN ALAT

Penggunaan jaringan *internet* sebagai media penyaluran sinyal k*on*trol dan m*on*itoring telah berkembang pesat, diantaranya untuk pengaturan peralatan di rumah, k*on*trol robot, peralatan rumah tangga dan mesin produksi di Industri. Namun demikian, dalam beberapa aplikasi umpan balik yang diberikan ke operator atau user hanya status dari obyek yang dik*on*trol. Dalam buku ini yang dipaparkan tentang m*on*itoring sebuah ruangan, yaitu dengan memberikan umpan balik berupa gambar video. Komunikasi *internet* pada sistem m*on*itoring ruangan terdiri dari *Client* dan *Camera server* .*Software* yang akan digunakan terdiri dari : Linux, *Camera server*, PHP (*Hypertext Preprocessor*), Bahasa Pemrograman C.

Pada sistem ini, kami menggunakan *mikrokontroller* AT89S51 untuk meng*on*trol putaran motor stepper. Motor stepper ini kami gunakan untuk mengatur arah putaran kamera secara horiz*on*tal (0° - 180°). Peng*on*trol ini terdiri dari beberapa blok rangkaian yang saling terkait satu sama lain, sehingga dapat dihasilkan sudut ysng akurat. Blok rangkaian yang digunakan adalah rangkaian *mikrokontroler AT89S51* yang dalam sistem ini digunakan sebagai *interface* antara PC dengan *driver* motor stepper.



Gambar 3-1

Blok Diagram Sistem

3.1. Perencanaan dan Pembuatan Perangkat Keras

3.1.1. Perangkat Keras Webcam

Pada skripsi ini digunakan *webcam* produk dari *Genius* dengan type *VideoCAM Express*. Kamera ini menggunakan CMOS untuk sensornya. Ini merupakan teknologi paling canggih dari *Genius*. Desain kamera ini memungkinkan untuk berputar 360 derajat, untuk pengambilan gambarnya (*snapshot*) dapat dilakukan dengan menekan bagian atas dari kamera. Adapun spesifikasi dari kamera tersebut adalah sebagai berikut :

- *Image pixel* yang dihasilkan adalah 300,000 pixel
- Resolusinya adalah : 160x120, 176x144, 320x240, 352x288
- *Video preview* dan *recording* mencapai 30 fps
- Terdapat *button* untuk pengambilan gambar (*snapshot*)
- Hasil *record* berformat data *MPEG* dan *AVI*

- Memungkinkan untuk berotasi 360 derajat
- Pengaturan fokus kamera terdapat pada sisi depan kamera, metode yang digunakan adalah *pan focus*
- Pengaturan pencahayaan putih (*white balance*) secara otomatis
- *Scan mode* secara *progressive* (setahap demi setahap)

3.1.2. Perencanaan dan Pembuatan Rangkaian Pengubah Level Tegangan

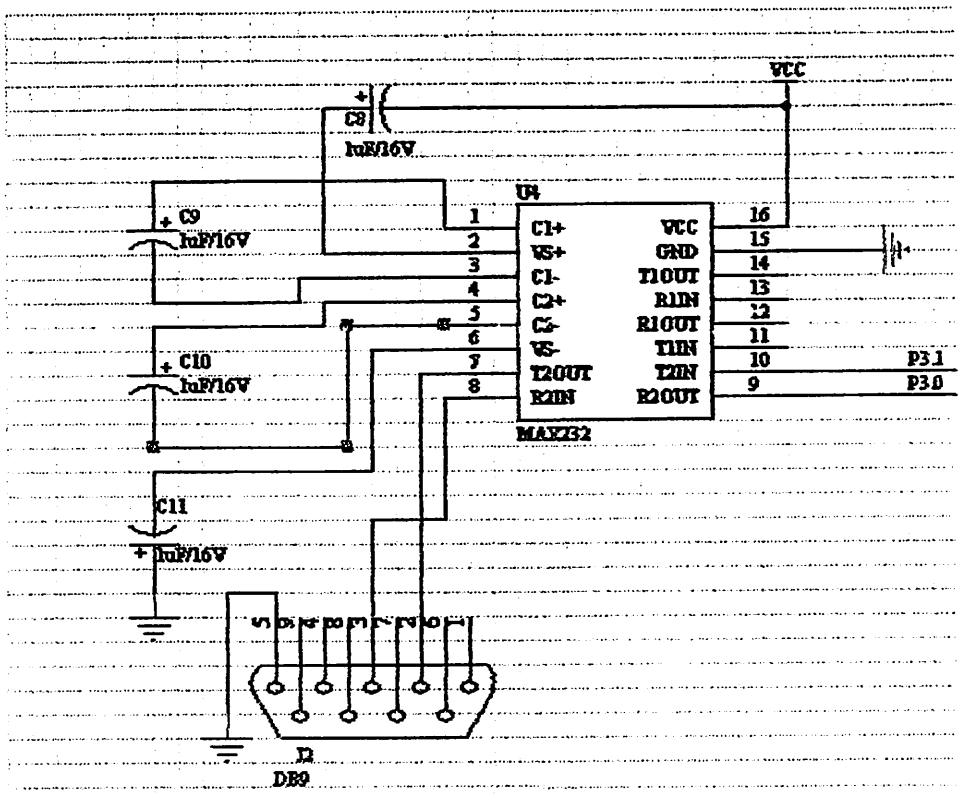
Untuk pembuatan skripsi ini kami menggunakan IC MAX232 sebagai pengubah level tegangan. IC MAX232 mempunyai 2 *receivers* yang berfungsi sebagai pengubah level tegangan dari level RS232 ke level *Transistor Transistor Logic* (TTL) dan mempunyai 2 *drivers* yang berfungsi mengubah level tegangan dari level TTL ke level RS232. Pasangan *driver*/receiver ini digunakan untuk *TX* dan *RX*, sedangkan pasangan yang lainnya digunakan untuk *CTS* dan *RTS*.

Tabel 3-1

Konfigurasi Pin IC MAX232

Nbr	Name	Purpose	Signal Voltage
1	C1+	+ connector for capacitor C1	capacitor should stand at least 16V
2	V+	output of voltage pump	+10V
3	C1-	+ connector for capacitor C1	capacitor should stand at least 16V
4	C2+	+ connector for capacitor C2	capacitor should stand at least 16V
5	C2-	- connector for capacitor C2	capacitor should stand at least 16V
6	V-	output of voltage pump / inverter	-10V
7	$T2_{out}$	Driver 2 output	RS-232
8	$R2_{in}$	Receiver 2 input	RS-232
9	$R2_{out}$	Receiver 2 output	TTL
10	$T2_{in}$	Driver 2 input	TTL
11	$T1_{in}$	Driver 1 input	TTL
12	$R1_{out}$	Receiver 1 output	TTL
13	$R1_{in}$	Receiver 1 input	RS-232
14	$T1_{out}$	Driver 1 output	RS-232
15	GND	Ground	0V
16	Vcc	Power supply	+5V

Dalam pembuatan rangkaian, IC MAX232 memerlukan beberapa kapasitor. Kami menggunakan kapasitor sebesar 1 µF dengan tegangan 16 Volt pada beberapa kaki pin. IC ini memerlukan input +5 Volt.



Gambar 3-2

Rangkaian Pengubah Level Tegangan

Ada 4 kapasitor yang digunakan dalam rangkaian ini yaitu pada pin 1 dengan pin 3, pin 4 dengan pin 5, pin 2 dengan pin 16. Untuk pin 6, karena bertegangan -10 Volt maka terhubung dengan kaki kapasitor kutub negatif sedangkan *Ground* terhubung ke kaki positif kapasitor. Koneksi antara IC MAX232 dengan RS232 terhubung melalui pin 14 *(driver 1 output)* yaitu sebagai *Tx (transmitter)* dengan DB9 pin2 *(received data)* dan pin 13 *(receiver 1 input)* sebagai *Rx (receiver)* dengan DB9 pin 3 *(transmitted data)*. Sedangkan pin 9 dan pin 10 menuju ke *mikrokontroller*.

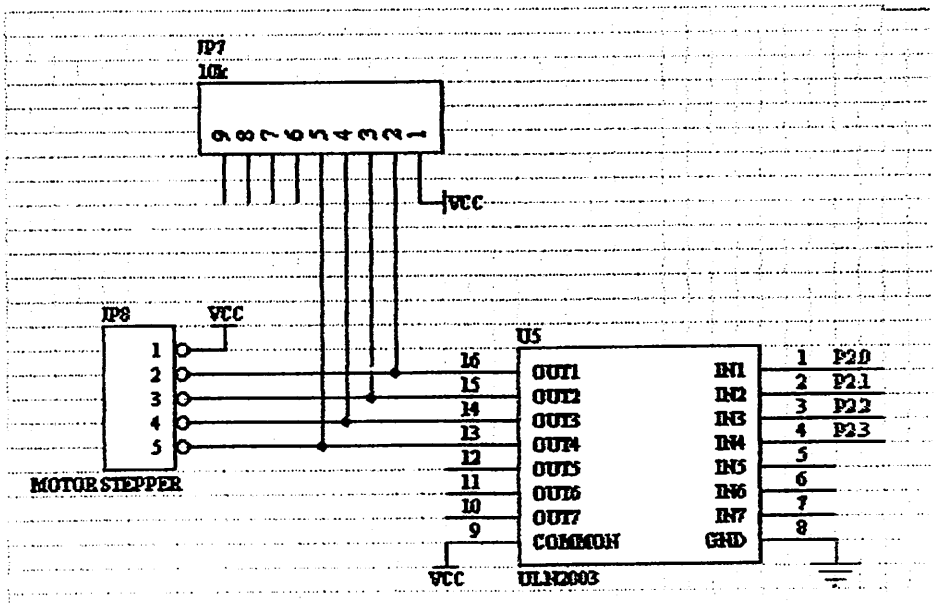
3.1.3. Perencanaan dan Pembuatan Driver Motor Stepper

Pada skripsi kali ini kami menggunakan motor stepper dengan type *unipolar*. Dibanding dengan jenis motor stepper yang lain motor stepper type *unipolar* sangat mudah dalam pengontrolannya. Rangkaian *driver* sebagai pengontrol putaran motor stepper jenis *unipolar* sangat mudah mekanismenya dibandingkan dengan rangkaian *driver* untuk motor stepper jenis yang lain.

Secara teoritis, sebuah motor stepper dapat digerakkan langsung oleh *mikrokontroller*. Dalam kenyataannya, arus dan tegangan yang dikeluarkan oleh *mikrokontroller* terlalu kecil untuk menggerakkan sebuah motor stepper. Gerbang-gerbang *Transistor Transistor Logic* (TTL) *mikrokontroller* hanya mampu mengeluarkan arus dalam orde mili-ampere dan tegangan antara 2 sampai 2,5 Volt. Sementara itu untuk menggerakkan motor stepper diperlukan arus yang lebih besar (dalam orde ampere) dan tegangan berkisar 5 sampai 24 Volt.

Untuk mengatasi masalah tersebut, kami menggunakan sebuah piranti tambahan yang memenuhi kebutuhan arus dan tegangan yang cukup besar. Rangkaian *driver* motor stepper menggunakan IC ULN2003 yang didalamnya terdapat transistor-transistor dengan susunan darlington yang merupakan rangkaian "*open collector*", dimana output rangkaian ini terhubung dengan *ground* untuk mencatu lilitan-lilitan motor stepper. Rangkaian ini pada dasarnya hanya merupakan rangkaian switching arus yang mengaliri lilitan pada motor stepper. Urutan pemberian data pada motor stepper ini dapat mengontrol arah putaran dari motor stepper ini. Penambahan kecepatan pada motor stepper dapat dilakukan dengan cara meningkatkan frekuensi pemberian data pada rangkaian *switching* arus.

Rangkaian k*on*trol ini nantinya terhubung langsung dengan lilitan pada motor, rangkaian power supplai, dan rangkaian yang pada akhirnya menentukan kapan lilitan yang diinginkan dalam k*on*disi *off* atau *on*. Kami menggunakan P2.0, P2.1, P2.2, P2.3 pada *mikrokontroller* sebagai output yang terhubung pada rangkaian *driver motor* . Ada dua parameter yang harus didefinisikan bila kita bekerja dengan stepper motor, yaitu: arah putaran dan kecepatan putaran. Arah putaran dapat diatur dengan memberikan tegangan pada lililtan-lilitan dengan urutan-urutan tertentu. Kecepatan putaran dapat ditentukan dengan mengatur dan menghitung perbandingan waktu yang dibutuhkan/diberikan untuk mencatu tegangan pada lilitan-lilitan dan selang waktu antara tegangan step yang diberikan pada satu lilitan dengan lilitan lain yang berurutan.



Gambar 3-3

Rangkaian Driver Motor

Perhitungan nilai resistor adalah sebagai berikut :

$$R = \frac{Vcc - Vol}{Iol}$$

$$R = \frac{5 - 0{,}45}{1{,}6} = 2843{,}5\ \Omega$$

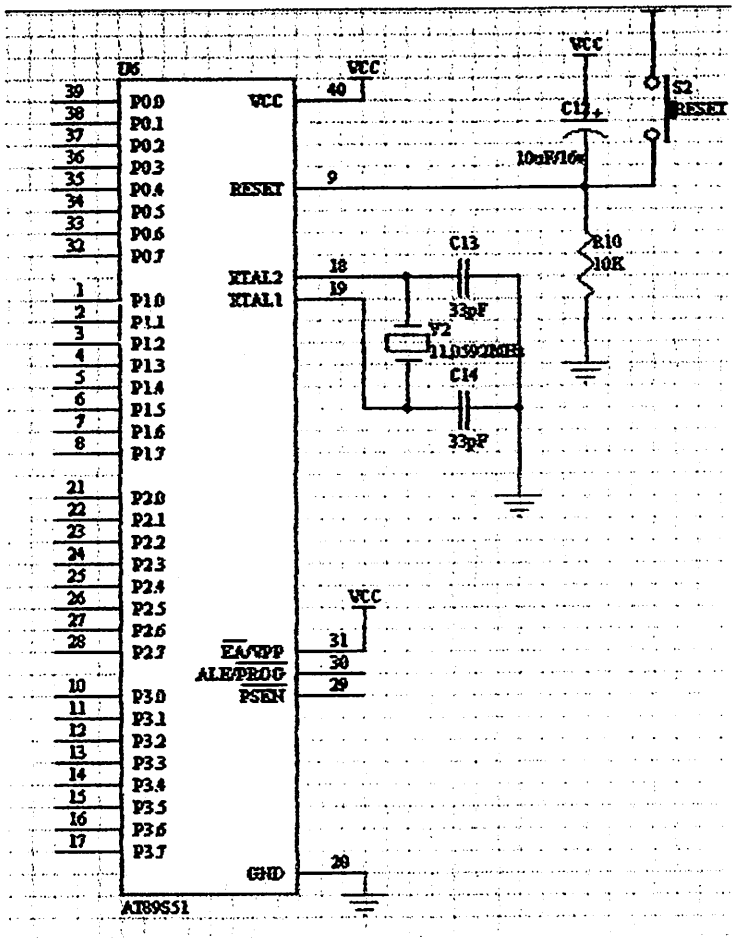
Dalam perancangan nilai resistor yang digunakan 10 KΩ, sehingga arus yang mengatur ketika logika rendah dapat diketahui yaitu :

$$Iol = \frac{5\,V - 0{,}45\,V}{10\,K} = 0.455\ mA$$

Jadi dengan nilai resistor sebesar 10 KΩ, maka arus yang terserap ketika keluaran rendah tidak lebih besar dari batas arus Iol yaitu 1,6 mA (berdasarkan datasheet).

3.1.4. Perencanaan dan Pembuatan Antarmuka dengan Mikrokontroler AT89S51

Dalam skripsi ini digunakan AT89S51 sebagai *interface* antara komputer *server* dengan memanfaatkan pin *Tx*, *Rx* yang ada pada AT89S51 sebagai komunikasi *serial*, maka AT89S51 fungsinya akan sama dengan PPI8255 tetapi pada PPI8255 digunakan komunikasi *paralel* sedangkan AT89S51 menggunakan komunikasi *serial*.



Gambar 3-4

Sistem Minimum AT89S51

3.2. Perencanaan dan Pembuatan Perangkat Lunak

3.2.1. Pembuatan Camera Server

Penanganan untuk *video streaming* akan dikerjakan oleh sebuah paket linux debian yaitu *camserv*. *Camserv* merupakan sebuah paket yang digunakan untuk menangani transfer video streaming secara real time dari kamera lewat jaringan *internet*. Sedangkan aplikasi *browser* yang bisa digunakan banyak sekali seperti : Internet Explorer, Nescape Navigator, Konqueror, Mozilla.

Untuk melakukan koneksi, *client* harus mengetahui IP *Address* komputer *server*. Selain itu, komputer *client* harus mengetahui pada nomor *port* yang mana komputer *server* akan mendengarkan permintaan koneksi, karena *server* diset untuk mendengarkan permintaan koneksi dari *client* pada suatu nomor *port* tertentu. Pada *Camserv*, *port* yang digunakan yaitu *port* 9192.

Perangkat keras yang dibutuhkan untuk mendukung *camera server* adalah sebuah *webcam*. *Webcam* yang kami gunakan adalah poduk dari *genius*. Sebenarnya *webcam* ini tidak bisa beroperasi dengan sistem operasi linux, *webcam* hanya beroperasi pada sistem operasi Windows. Sehingga diperlukan sebuah *driver* agar *webcam* dapat beroperasi pada Linux. *Driver* yang dibutuhkan adalah OV511. OV511 adalah *driver* untuk *webcam* yang menggunakan seri *chip* OmniVision OV5xx. Kebanyakan interface antara computer dengan *webcam* yang mendukung *driver* ini menggunakan *USB port*. Driver ini dapat support terhadap *video streaming* dan penangkapan gambar (*capture*) secara *monochrome* ataupun berwarna dengan menngunakan *Video4Linux* (V4L) API. Kebanyakan aplikasi dari V4L *compatible* terhadap performance dari *driver* ini. Penginstalannya adalah sebagai berikut :

```
# apt-get install ov511
```

Sehingga dengan *driver* ini *webcam* tersebut akan dapat beroperasi pada sistem operasi Linux. *Camserv* merupakan suatu paket untuk menangani video streaming. Paket ini disediakan oleh distro linux Debian secara gratis. Kami mengambil paket ini dari *http://debian.indika.net.id*. Di mirror ini tersedia

berbagai paket yang support dengan Linux Debian. Penginstalannya adalah sebagai berikut :

```
# apt-get install camserv
```

dengan perintah tersebut paket *camserv* secara otomatis akan terinstal pada komputer, dan secara sekaligus komputer kita akan menjadi *camera server*. Dimana *camera server* akan melakukan broadcast lewat jaringan IP. Untuk pengecekan apakah paket tersebut sudah terinstall atau belum, maka kita lakukan perintah :

```
# dpkg -l | grep camsrv
```

Maka akan muncul :

```
ii    camserv    0.5.1-4 Stream Live Out on to Web
```

ii merupakan tanda bahwa paket tersebut sudah terinstall pada komputer. Setelah paket tersebut terinstall maka kita harus mengaktifkannya. Perintah untuk mengaktifkan adalah sebagai berikut :

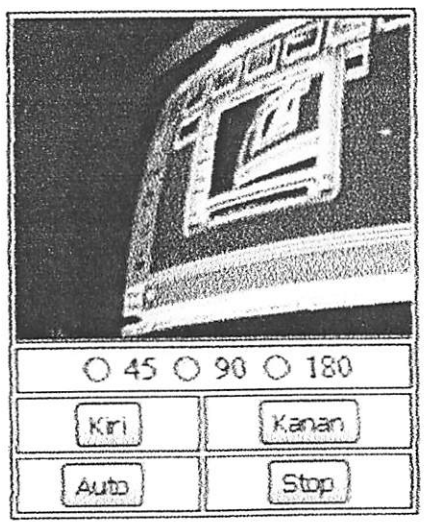
```
# /etc/init.d/camserv start
Starting camserv: camserv
```

kemudian untuk mengkonfigurasi *camserv*, file konfigurasinya terdapat pada *directory /etc/camserv/camserv.cfg*. Didalam file konfigurasi ini terdapat berbagai macam setting untuk mengkonfigurasi *camserv*, antara lain panjang dan lebar tampilan pada *web browser*, mode yang digunakan (PAL, NTSC, dll), pencahayaan, warna, dan lain-lain.

Selain paket tersebut dibutuhkan juga paket pendukung yaitu *devfsd* , paket ini digunkan untuk memanajemen *Linux Device File System*. Selain itu paket lain yang dibutuhkan adalah *apache* yang digunakan untuk akses localhost. Setelah semua itu terinstall maka kita harus mengecek , apakah device untuk video sudah ada pada *directory /dev/video0* atau */dev/video1*, apabila masih belum ada, maka kita harus membuat device itu untuk menangani inputan dari *webcam* yaitu dengan perintah sebagai berikut:

```
# mknod /dev/video c 81 o
# ls video0 –la
# ln –s /dev/video /dev/video0
```

Kemudian untuk melihat hasil dari *camserv* adalah pada *web browser*. Pada kolom *address* kita ketikkan http://localhost:9192, dimana 9192 adalah *port* untuk *camserv*.



Gambar 3-5

Tampilan Pada Web Browser

3.2.2. Perangkat Lunak Untuk Pengontrolan Derajat Putar Motor Stepper (Pada PC Server)

Untuk membuat integrasi antara *web* dengan mikrokontroller diperlukan suatu jembatan untuk mengirimkan data yang diinginkan oleh *client*. Jembatan disini kita menggunakan *software* pemrograman PHP. Pada dasarnya program PHP disini mempunyai 2 tugas yaitu :

1. Membuka port serial pada server sekaligus melakukan inisialisasi port.
2. Mengirimkan data untuk putar kanan atau kiri ke mikrokontroler.

Untuk dapat berkomunikasi antara *mikrokontroller* AT89S51 dengan komputer diperlukan setting beberapa parameter antara lain *baudrate* dan *port* yang dipakai. Untuk itu pertama kali yang kita lakukan adalah penginisialisasian *port serial* yang akan kita gunakan. Diantaranya adalah membuka *port serial*, menuliskan data ke *port serial* dan kemudian mengirimkannya, membaca data pada *port serial*, dan menutup *port serial*. Listing program yang digunakan sebagai berikut :

- Membuka *serial port*

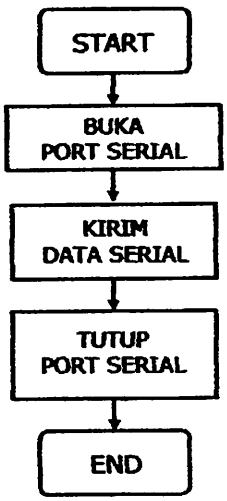
  Pada Linux Ubuntu *serial port* terdapat pada *directory /dev/ttyS0*. Untuk dapat mengaksesnya kita harus *login* sebagai *super user*.

```
<?
`mode com1: BAUD=9600 PARITY=N data=8 stop=1 FlowControl=N`;
$fp = fopen("COM1:", "w+");
 if (!$fp) {
   echo "Port not opened.";
  }
?>
```

- Menutup *port serial*

```
<?
 fclose ($fp);
 ?>
```

Secara garis besar *flowchart* untuk program php terdapat pada gambar 3.8.

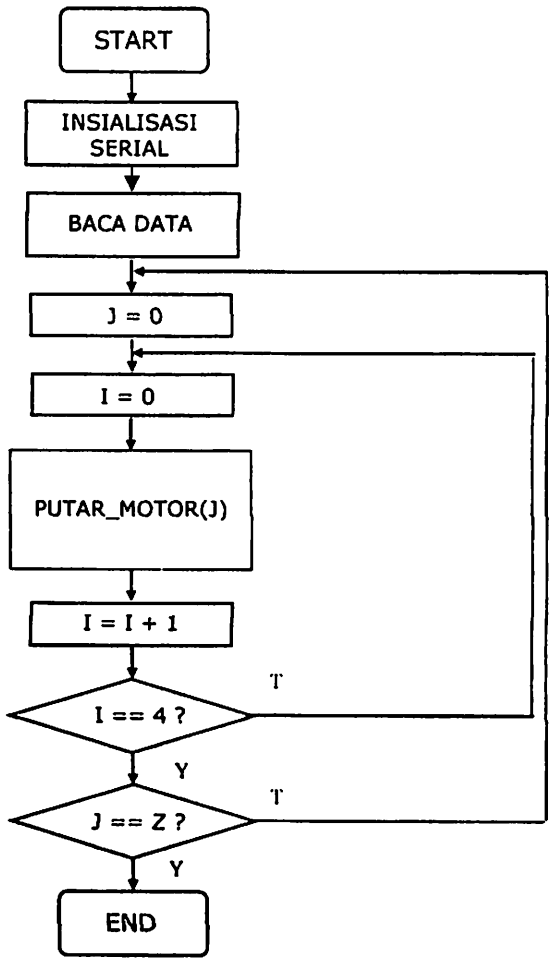


Gambar 3-6

Flowchart program php

3.2.3. Perangkat Lunak Untuk Pengontrolan Motor Stepper (pada Mikrokontroler)

Perangkat lunak ini digunakan untuk pengontrolan arah putaran kamera dalam pengambilan gambar. Motor stepper yang digunakan adalah motor stepper jenis *unipolar* dengan spesifikasi $1{,}8^0$ per step. Adapun *flowchart*nya terdapat pada gambar 3.9.



Gambar 3-7

Flowchart Program Mikrokontroller

3.2.3.1. Penginisialisasian Port Serial Pada Mikrokontroler

Pada *mikrokontroler* penginisialisasian *port serial* juga harus dilakukan. *Register* yang digunakan adalah Init_Serial. Dimana 8 bit data *asinkron* yang terdiri atas 10 bit, yaitu 1 bit start, 8 bit data, dan 1 bit stop. Dan pada mode ini kita dapat mengatur *baudrate*. Pada mode ini fungsi-fungsi *alternatif* dari

penerimaan data *serial* dan P3.1 berfungsi sebagai pin untuk pengiriman data *serial*. Pengiriman data dilakukan dengan menuliskan data yang akan dikirim ke *register* SBUF. Data *serial* akan digeser keluar diawali dengan bit *start*, kemudian data dari bit yang berbobot terendah (LSB) hingga bit berbobot tertinggi (MSB) dan diakhiri dengan bit *stop*.

Penerimaan data dilakukan oleh *mikrokontroller* dengan mendeteksi adanya perubahan k*on*disi dari logika high ke logika *low* pada kaki RxD. Perubahan k*on*disi tersebut merupakan bit start. Data yang masuk pada *serial* pertama kali akan ditampung oleh bit RI (*Receive Interupt Flag*), bit ini akan set pada akhir pengiriman karakter. Selanjutnya, data *serial* akan digeser masuk kedalam SBUF.

Pada sistem ini kami tidak menggunakan bit TI (*Transmit Interupt Flag*) yaitu bit yang akan diset pada saat akhir pengiriman karakter karena kami tidak memerlukan perintah balikan dari PC *server*. PC hanya mengirimkan perintah berupa karakter ke *mikrokontroller* untuk pengontrolan motor stepper.

3.2.3.2. Pengaturan Baudrate Pada Komunikasi Serial

Dalam komunikasi *serial* ini mode *serial* yang digunakan adalah mode 1, dimana dengan mode *serial* ini kita bisa mengatur *baud rate*nya. *Baud rate* yang digunakan dalam komunikasi *serial* ini adalah 9600 bps. Penentuan *baud rate* sangat bergantung pada crystal yang kita gunakan, *crystal* ini adalah sebagai oscillator. *Crystal* yang digunakan adalah 11,0592 MHz. Pada mode ini pengaturan *baud rate* menggunakan *Timer* 1. Penghitungannya adalah sebagai berikut :

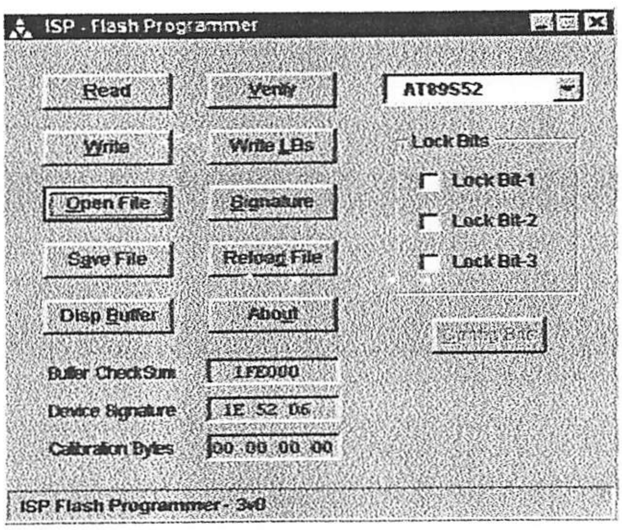
$$9600 = \frac{fosc}{12\, x[256 - TH\ 1]x32} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad 1$$

$$9600 = \frac{11{,}0592\ MHz}{12\, x[256 - TH\ 1]x32} \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad 2$$

Dengan frekwensi oscillator sebesar 11,0592 MHz, TH1 (*Timer 1*) adalah 253 atau dalam format heksa 0FDH.

3.2.3.3. Perangkat Lunak Downloader Pada Mikrokontroler AT89S51

Perangkat lunak yang digunakan untuk men-*mendownload*-kan program dari komputer ke *mikrokontroller* adalah *ISP Flash Programmer – 3v0.*



Gambar 3-8

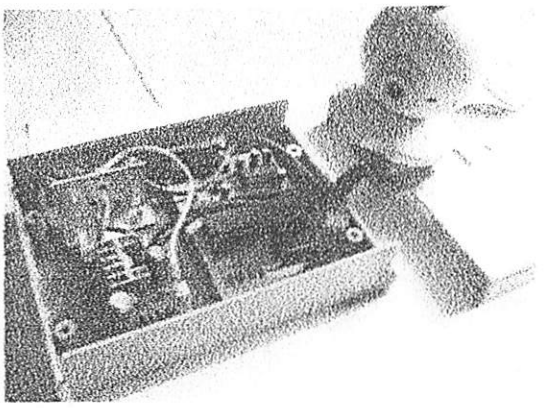
Tampilan Menu Utama ISP Flash Programmer

Keterangan :

- *Read* (baca) dan *Write* (tulis) file Hexa ke *mikrokontroller*
- *Read Signature*, untuk mengunci bit
- *Clear* dan *Fill*, untuk mengosongkan dan mengisi kembali memori pada *buffer*
- *Verify*, untuk mem-verifikasi memori pada *buffer*
- *Reload*, Untuk mengisi kembali *mikrokontroller* dengan program yang terakhir kali di-*download*-kan
- *Display Buffer*, digunakan untuk pengecekan dan tampilan isi pada *buffer*
- *Auto Detection of Hardware*, untuk mendeteksi perangkat keras sistem minimum AT89S51

3.3. PERANGKAT KERAS SISTEM MONITORING RUANGAN

Perangkat keras untuk system monitoring ruangan tediri atas : *webcam*, *Mikrokontroller* AT89S51 sebagai *interface* dengan computer *server*, power supply, motor stepper. Sistem yang telah dibuat terlihat pada gambar 3.11.



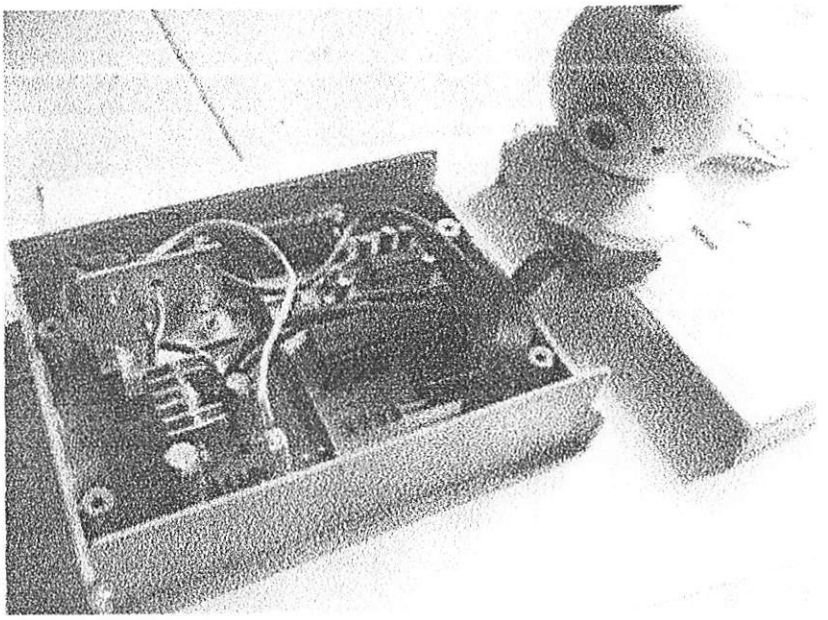
Gambar 3-9

Gambar Perangkat Keras yang sudah dibuat

BAB IV
PENGUJIAN ALAT

4.1. Pengujian *Hardware*

Secara umum, pengujian ini bertujuan untuk mengetahui apakah alat dapat bekerja sesuai dengan spesifikasi perencanaan yang telah ditetapkan. Alat yang dirancang dan dibuat seperti pada gambar berikut ini:



Gambar 4-1

Rangkaian Keseluruhan

4.1.1. Pengujian Motor Stepper

Motor stepper berfungsi sebagai pemutar arah kamera. Pengujian motor stepper ini berfungsi untuk mendapatkan nilai resistansi motor dan menentukan kaki common pada motor stepper serta untuk mengetahui apakah motor dapat berputar. Kaki common ini nantinya dihubungkan ke sumber tegangan.

Pengukuran resistansi motor stepper dapat dilakukan dengan menghubungkan salah satu kaki, dengan kaki lainnya pada motor stepper dengan ohm-meter secara bergantian dan berurutan. Untuk menandakan adanya common adalah harga nilai resistansi common ½ dari nilai resistansi kaki-kaki lainnya. Berikut adalah hasil pengukuran dari nilai resistansi motor stepper.

Tabel 4-1

Hasil Pengukuran Resistansi Motor Stepper

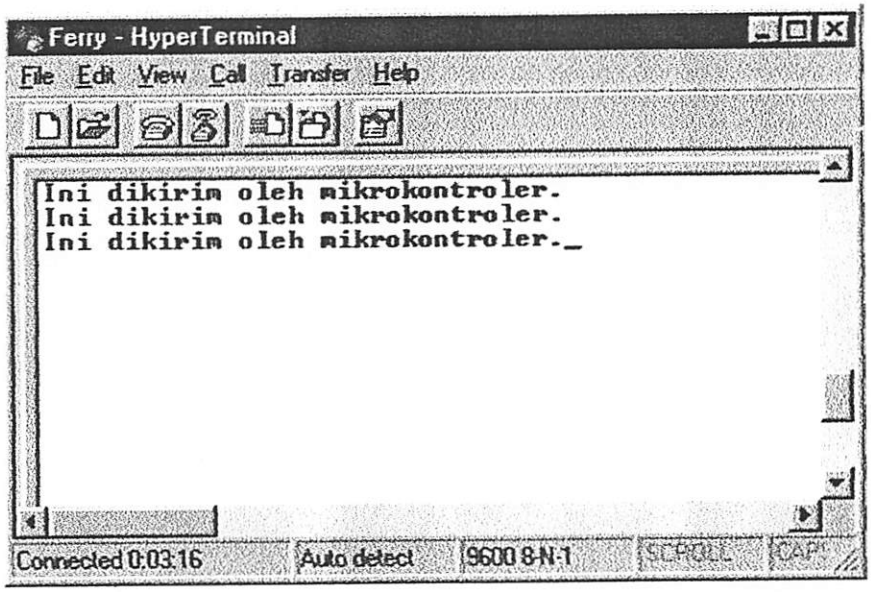
R(Ω)	C	A1	A2	A3	A4
C	-	77	80	78	80
A1	77	-	160	160	160
A2	80	160	-	160	160
A3	78	160	160	-	160
A4	80	160	160	160	-

Untuk mengetahui apakah motor dapat berputar dan sekaligus untuk menentukan urutan putaran motor maka dilakukan dengan cara menghubungkan kaki common pada motor stepper dengan sumber tegangan DC (+) dan kaki-kaki yang lain

dihubungkan dengan ground sumber tegangan DC (-) secara bergantian sampai mendapatkan urutan kaki dengan arah perputaran motor searah.

4.1.2. Pengujian Komunikasi Data Serial PC dan Mikrokontroller

Untuk pengujian komunikasi data serial PC dan Mikrokontroller menggunakan fasilitas yang telah disediakan oleh windows yaitu ***Hyper Terminal***. Dimana data yang terkirim dari mikro dapat diterima oleh PC, hasil tampilan dari pengujian alat dapat dilihat pada gambar berikut:



Gambar 4-2

Tampilan Hasil Pengujian Mikrokontroler dengan PC

Bahasa pemrograman untuk pengujian komunikasi data serial adalah sebagai berikut :

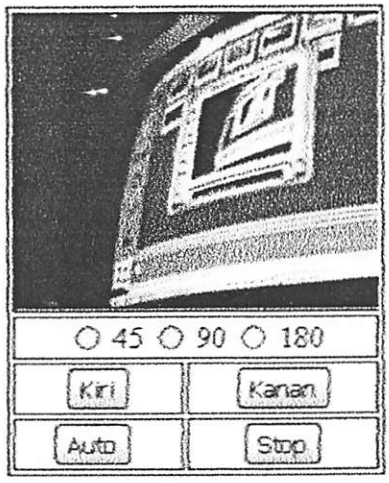
```
#include <reg51.h>
#include <stdio.h>

void main()
{
while(1)
{
printf("Ini dikirim oleh mikrokontroler\n");
}
}
```

4.1.3. Pengujian Webcam

Webcam disini berfungsi untuk menangkap gambar. Pengujian *webcam* disini dimaksudkan untuk mengetahui apakah *webcam* berfungsi dengan baik atau tidak, selain itu juga untuk mengetahui kualitas penerimaan gambarnya.

Untuk pengujian *webcam* ini digunakan bahasa pemrograman *PHP* serta *HTML* serta *browser* untuk mengetahui apakah gambar bisa ditampilkan di *browser*.



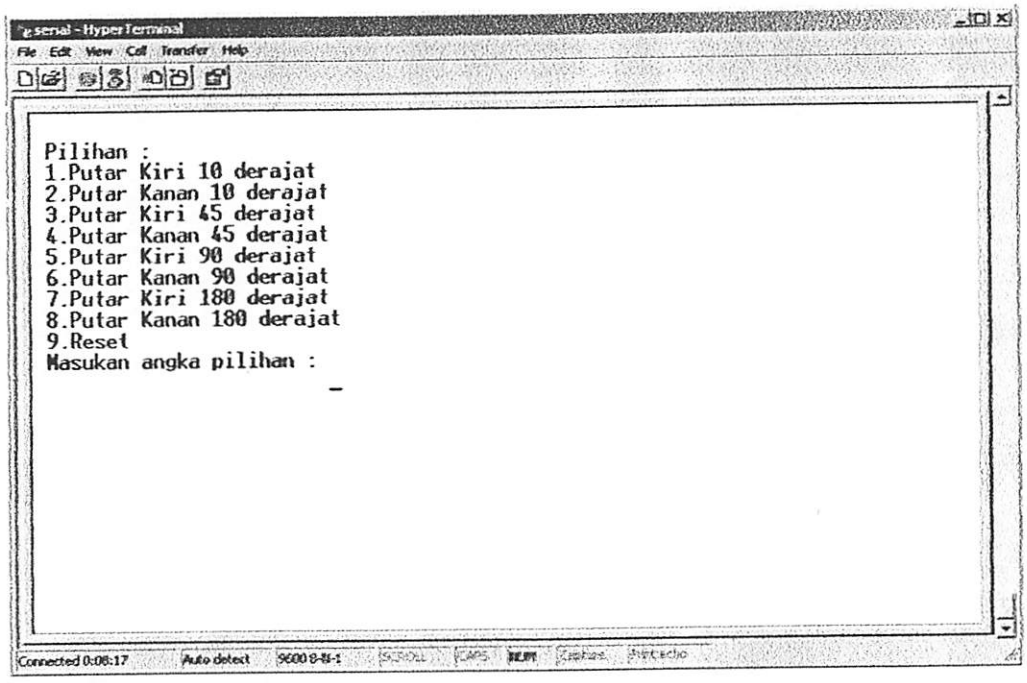
Gambar 4-2

Tampilan Gambar Hasil Perekaman Gambar dari Webcam

4.2. Pengujian Software

4.2.1. Pengujian Program C

Pengujian program C ini dilakukan pada saat mengkompile program C, yang merupakan kumpulan baris-baris perintah dan telah disimpan dengan *extention* .c. Program ini dapat ditulis menggunakan *software text editor* seperti *Notepad* atau *editor* DOS. Program inilah yang akan di*download*kan ke mikrokontroller. Berikut adalah tampilan hasil pemrograman C yang telah didownloadkan ke mikrokontroler dan dihubungkan ke PC :



Gambar 4-3

Hasil Pengujian Program C

4.2.2. Pengujian Program PHP

Program PHP disini berfungsi sebagai jembatan komunikasi antara mikrokontroler dan browser. Sehingga mikrokontroler dapat dikontrol dari web browser. Pengujian yang dilakukan adalah dengan memberikan inputan sesuai dengan pilihan yang tersedia pada software yang telah didownloadkan pada mikrokontroler menggunakan bahasa pemrograman PHP yang diakses melalui browser. Bahasa pemrogramannya adalah sebagai berikut :

```
`mode com1: BAUD=9600 PARITY=N data=8 stop=1 FlowControl=N`;
$fp = fopen("COM1:", "w+");
fputs ($fp, chr(56) );
fputs ($fp, chr(57) );
fclose ($fp);
```

BAB V

PENUTUP

5.1. KESIMPULAN

Berdasarkan hasil analisa yang dilakukan, sistem yang telah dibangun ini masih jauh dari sempurna. Dari hasil pengujian dan pengukuran dapat disimpulkan bahwa :

1. Pengontrolan mikrokontroler dapat diintegrasikan melalui internet dengan menggunakan web browser, sehingga dapat dikembangkan untuk model alat komunikasi jarak jauh.
2. Untuk pengontrolan motor stepper, dibutuhkan rangkaian driver yang akan mencatu lilitan-lilitan motor stepper. Hal yang perlu diperhatikan adalah kecepatan putaran dan arah putaran.
3. Dalam pengiriman data perintah putaran motor stepper dari client, data yang dikirimkan adalah karakter ASCII.
4. Pada komunikasi serial, penginisialisasian baudrate dilakukan pada PC server dan pada mikrokontroller. Kedua inisialisasi ini harus sama. Pada sistem ini digunakan baudrate sebesar 9600 bps.

5.2. SARAN

Sistem monitoring ruangan ini dapat dikembangkan ke arah *motion* atau *image tracking*. Selain itu sistem ini bukan hanya dapat dimonitoring lewat internet tetapi juga dapat dikembangkan melalui Hand Phone. Untuk putaran kamera diharapkan dapat berputar secara horisontal dan vertikal sehingga memungkinkan untuk memonitoring segala sudut ruangan.

DAFTAR PUSTAKA

1. Paulus Andi Nalwan, *Panduan Praktis Teknik Antar Muka dan Pemrograman Mikrokontroller AT89S51, 2003.*
2. Onno W. Purbo, *Konferensi Video Melalui Internet,* Andi Yogyakarta, 2002
3. ____, *"Modul Pelatihan Pengenalan Linux"*, http://ftp.ui.edu/bebas/v01/TimPandu/linux-dasar-single.pdf
4. ____, *"Camserv Package"*, http://www.debian .org/
5. ____, *"Camserv Package"* http://mirror.eepis-its.edu/
6. ____, *"RS-232"* , http://www.lookrs232.com/
7. ____, *"How to work Stepper Motor"*, http://eio.com/jasstep.htm
8. ____,*"SerialProgramming"*, http://en.wikibooks.org/
9. ____, *"AT89S51 datasheet"*, http://www.atmel.com/
10. ____,*"ISP Flash Programming "* http://www.kmitl.ac.th/~kswichit/ISP-Pgm3v0/ISP-Pgm3v0.html

INSTITUT TEKNOLOGI NASIONAL
JL. BENDUNGAN SIGURA-GURA 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Ahmad Zaid Zam Zami
Nim : 0117115
Masa Bimbingan : 15 Desember 2006 s/d 15 Juni 2007
Judul Skripsi : Perancangan dan Pembuatan Web Based Camera Controller Berbasis Mikrokontroler AT89S51 dan Apache Web Server

NO	Tanggal	Uraian	Paraf Pembimbing
1.			
2.			
3.			
4.			
5.			
6.			

Malang,
Dosen Pembimbing

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

Form. S-4a

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

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

NAMA : AHMAD.. ZAID ZAM ZAMS
N I M : 02.17.115
Perbaikan meliputi :

1) Gambar lengkap !
2) Lampiran.
3) Flow Chart.

Malang,

( )

INSTITUT TEKNOLOGI NASIONAL
JL. BENDUNGAN SIGURA-GURA 2
MALANG

LEMBAR PERBAIKAN SKRIPSI

Nama	: Ahmad Zaid Zam Zami
Nim	: 0117115
Jurusan	: Teknik Elektro S1
Konsentrasi	: *Teknik Elektronika*
Hari / Tanggal Ujian Skripsi	: Jumat / 16 Maret 2007

Materi Perbaikan	Paraf
Gambar lengkap *Lampiran* Flowchart	

Diperiksa / Disetujui

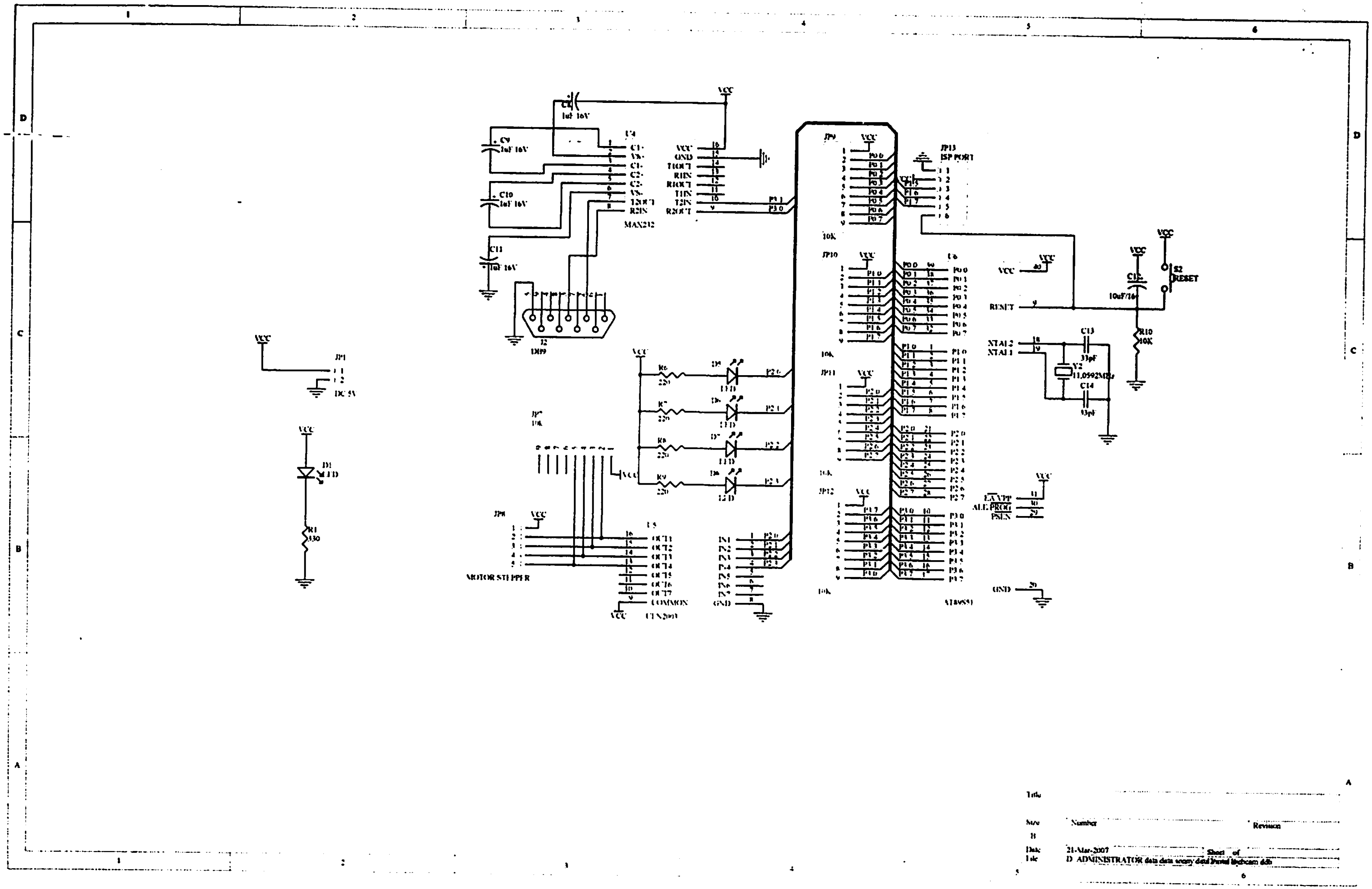
PENGUJI II

I Komang Somawirasta ST, MT
NIP.P. 1030100361

Mengetahui :

Dosen Pembimbing

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


VCC
U4
C1+
VS+
C1-
C2+
C2-
VS-
T2OUT
R2IN
MAX232
TIOUT
RIIN
RIOUT
TIIN
T2IN
R2OUT
GND
C8 1uF 16V
C9 1uF 16V
C10 1uF 16V
C11 1uF 16V
J2
DB9
JP1
DC 5V
D1 LED
R1 330
JP13 ISP PORT
JP9
JP10
JP11
JP12
10K
U6
AT89S51
RESET
XTAL2
XTAL1
C12 10uF/16v
S2 RESET
R10 10K
C13 33pF
C14 33pF
Y2 11.0592MHz
EA/VPP
ALE/PROG
PSEN
GND
R6 220
R7 220
R8 220
R9 220
D5 LED
D6 LED
D7 LED
D8 LED
JP8
MOTOR STEPPER
U5
OUT1
OUT2
OUT3
OUT4
OUT5
OUT6
OUT7
COMMON
ULN2003
IN1
IN2
IN3
IN4
IN5
IN6
IN7
Title
Size
Number
Revision
B
Date
21-Mar-2007
Sheet of
File
D:\ADMINISTRATOR\data data\...ddb

LAMPIRAN

LISTING PROGRAM

1. Program C

```
#include <at89x51.h>
void Init_Serial(char baud)
{
 TMOD = 0x20;
 PCON = 0x0;
 SM0 = 0;
 SM1 = 1;
 REN = 1;
 TH1 = baud;
 TL1 = baud;
 TR1 = 1;
}

void KirimChar(unsigned char datanya)
{
 SBUF = datanya; // kirim melalui port serial
 while(!TI); // tunggu sampai semua bit terkirim
 TI = 0; // timer dinolkan
}

void KirimTeks(char *teks) // Fungsi untuk mengirim teks ke
port serial pada mode 1
{
 char i=0;
 while(teks[i]!=0)
 {
 KirimChar(teks[i]);
 i++;
 }
}

char TerimaChar() // Fungsi data terima karakter
{
 while(! RI);
 RI = 0; // RI dinolkan kembali
 return(SBUF);
}

char TerimaDataPort() // Fungsi data terima karakter
{
 while(! RI);
 RI = 0; // RI dinolkan kembali
 SBUF = P0;
 return(SBUF);
}

void tundalms()
```

```
 {
  int i;
  for(i=0;i<100;i++);
 }
  void tunda(int n)
 {
  int i;
  for (i=0; i<n; i++)
  tundalms();
 }

 void PutarKanan(unsigned char i)
 {
  while (i > 0)
  {
   P2 = 0x01;
   tunda(100);
   P2 = 0x02;
   tunda(100);
   P2 = 0x04;
   tunda(100);
   P2 = 0x08;
   tunda(100);
   i--;
  }
 }

 void PutarKiri(unsigned char i)
 {
  while (i > 0)
  {
   P2 = 0x08;
   tunda(100);
   P2 = 0x04;
   tunda(100);
   P2 = 0x02;
   tunda(100);
   P2 = 0x01;
   tunda(100);
   i--;
  }
 }

 void Reset()
 {
  P0 = 0xFF;
  P1 = 0xFF;
  P2 = 0xFF;
  P3 = 0xFF;
 }

 void main()
 {
 char tombol;
 Init_Serial(0xFD);
 while(1)
 {
       KirimTeks("\r Pilihan : \n");
```

```
            KirimTeks("\r 1.Putar Kiri 10 derajat\n");
            KirimTeks("\r 2.Putar Kanan 10 derajat\n");
            KirimTeks("\r 3.Putar Kiri 45 derajat\n");
            KirimTeks("\r 4.Putar Kanan 45 derajat\n");
            KirimTeks("\r 5.Putar Kiri 90 derajat\n");
            KirimTeks("\r 6.Putar Kanan 90 derajat\n");
            KirimTeks("\r 7.Putar Kiri 180 derajat\n");
            KirimTeks("\r 8.Putar Kanan 180 derajat\n");
            KirimTeks("\r 9.Reset \n");
            KirimTeks("\r Masukan angka pilihan : \n");
            tombol=TerimaChar();
            KirimTeks("\r Pilihan Anda : \f ");
            KirimChar(tombol);
            KirimChar('\n');
            switch(tombol)
            {
              case '1': PutarKiri(1);break;
              case '2': PutarKanan(1);break;
              case '3': PutarKiri(6);break;
              case '4': PutarKanan(6);break;
              case '5': PutarKiri(13);break;
              case '6': PutarKanan(13);break;
              case '7': PutarKiri(25);break;
              case '8': PutarKanan(25);break;
              case '9': Reset();break;
              default : break;
            }
      }
      }
```

2. Program PHP

```
<form name="putaran" method="get" action="serial.php">
  <script type="text/javascript">
  function pilihan(){
  if(document.putaran.opt[2].selected){
  document.getElementById("kaki").style.visibility = "visible";
   document.getElementById("kepala").style.visibility = "hidden";
  }else{
   document.getElementById("kaki").style.visibility = "hidden";
   document.getElementById("kepala").style.visibility = "visible";
  }

  }
  </script>

  <table width="29%"  border="0" cellspacing="0" cellpadding="0">

    <tr align="left">
      <td width="30%">
        <?
        if($_GET['opt'] == "manual"){
        $sel = "selected";
        }
        ?>
        <select name="opt" id="select" onChange="pilihan()">
        <option selected>--Pilih--</option>
```

```
        <option value="auto">Auto</option>
        <option value="manual" <?=$sel?>>Manual</option>
      </select></td>
      <td width="70%">    <div id="kepala"
style="visibility:hidden"><input name="L2" type="submit" id="L23"
value="Proses">
        <input type="hidden" name="set" value="1">
      <input name="stop" type="submit" value="stop">
</div></td>
    </tr>
  </table>
    <?
        if($_GET['opt'] == "manual"){
       ?>
       <div id="kaki" style="visibility:visible">
       <?
        }else{
        ?>
       <div id="kaki" style="visibility:hidden">
       <? } ?>
  <table width="29%"  border="0" cellspacing="0" cellpadding="0">
  <tr align="left">
    <td colspan="2"> </td>
    <td width="40%"> </td>
  </tr>
  <tr align="left">
    <td width="29%"><input name="deg" type="radio" value="0">
      0        </td>
    <td width="31%"><input name="deg" type="radio" value="90">
90</td>
    <td><input name="deg" type="radio" value="180">
180</td>
  </tr>
  <tr align="left">
    <td colspan="3"> </td>
    </tr>
  <tr align="left">
    <td colspan="3">
        <?
         if(!$_GET["stepki"]){ $stepki  =3; $diski = "disabled";}
         if(!$_GET["stepka"]){ $stepka  =1;}
        if($_GET["stepka"] == 1){
        $va = "45";
        $stepka =2;
        $stepki  =3;
        }elseif($_GET["stepka"] == 2){
        $va = "90";
        $stepka  =3;
        $stepki  =2;
        }elseif($_GET["stepka"] == 3){
        $va = 135;
        $stepka  =4;
        $stepki  =1;
        }elseif($_GET["stepka"] == 4){
        $va = 180;
        $stepka  =4;
        $stepki  =1;
        }
```

```
          if($_GET["stepki"] == 3){
          $vi = 45;
          $stepki  =3;
          }
         //-> BUKA FILE KIRI
       $fkiri = fopen("kiri.txt", "w+");
         fputs($fkiri,$vi);
         fclose ($fkiri);
         //--> BUKA FILE KANAN
       $fkanan = fopen("kanan.txt", "w+");
         fputs($fkanan,$va);
         fclose ($fkanan);
          ?>
          <input name="L" type="submit" id="L3" value="Kiri"
<?=$diski?>>
          <input name="R" type="submit" id="R" value="Kanan">
          <input type="hidden" name="stepki" value="<?=$stepki?>">
          <input type="hidden" name="stepka" value="<?=$stepka?>">
          </td>
    </tr>
</table>
   </div>
</form>
<?php
// HOW TO USE PHP TO WRITE TO YOUR SERIAL PORT: TWO METHODS
//$serproxy=true;
$com1=true;
if ($serproxy) {
   // Use this code in conjunction with SERPROXY.EXE
   // (http://www.lspace.nildram.co.uk/freeware.html)
   // which converts a Serial stream to a TCP/IP stream
   $fp = fsockopen ("localhost", 5331, $errno, $errstr, 30);
   if (!$fp) {
       echo "$errstr ($errno)";
   } else {
       $e = chr(56);
       $string  = $e . "A" . $e . "H300";
       $string .= $e . "V100" . $e . "XL1SATO";
       $string .= $e . "Q1" . $e . "Z";
       echo $string;
       fputs ($fp, $string );
       fclose ($fp);
   }
} elseif ($com1) {
   // Use this code to write directly to the COM1 serial port
   // First, you want to set the mode of the port. You need to set
   // it only once; it will remain the same until you reboot.
   // Note: the backticks on the following line will execute the
   // DOS 'mode' command from within PHP
   `mode com1: BAUD=9600 PARITY=N data=8 stop=1 FlowControl=N`;
   $fp = fopen("COM1:", "w+");
   if (!$fp) {
       echo "Uh-oh. Port not opened.";
   } else {
   if($_GET['opt'] == "auto"){
   if($_GET["set"] == "1"){
   if($_GET["stop"]){exit();}
```

```
     fputs ($fp, chr(55) );
    if($_GET["stop"]){exit();}
       sleep(9);
       fputs ($fp, chr(56) );
       fputs ($fp, chr(57) );
       sleep(9);
     print '<meta http-equiv="refresh"
 content="0;URL=http://localhost/serial.php?opt=auto&L2=Proses&set=
 2">';
             if($_GET["stop"]){
             sleep(9);
       fputs ($fp, chr(57) );
       }
       }
   if($_GET["set"] == "2"){
       fputs ($fp, chr(55) );
       if($_GET["stop"]){
       fputs ($fp, chr(57) );
       }
       sleep(9);
       fputs ($fp, chr(56) );
       fputs ($fp, chr(57) );
       sleep(9);
            if($_GET["stop"]){
       fputs ($fp, chr(57) );
       }
       print '<meta http-equiv="refresh"
content="0;URL=http://localhost/serial.php?opt=auto&L2=Proses&set=
1">';
       }
   }elseif($_GET['opt'] == "manual"){
   if($_GET['L']){
   $arah = chr(52);
   }else{
   $arah = chr(51);
   }
    $e = $arah;
      fputs ($fp, $e );
      fputs ($fp, chr(57) );
   }


       fclose ($fp);
   }
}
     ?>
```

Features

- Compatible with MCS®-51 Products
- 4K Bytes of In-System Programmable (ISP) Flash Memory
  - Endurance: 1000 Write/Erase Cycles
- 4.0V to 5.5V Operating Range
- Fully Static Operation: 0 Hz to 33 MHz
- Three-level Program Memory Lock
- 128 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Two 16-bit Timer/Counters
- Six Interrupt Sources
- Full Duplex UART Serial Channel
- Low-power Idle and Power-down Modes
- Interrupt Recovery from Power-down Mode
- Watchdog Timer
- Dual Data Pointer
- Power-off Flag
- Fast Programming Time
- Flexible ISP Programming (Byte and Page Mode)
- Green (Pb/Halide-free) Packaging Option

Description

e AT89S51 is a low-power, high-performance CMOS 8-bit microcontroller with 4K es of In-System Programmable Flash memory. The device is manufactured using nel'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 onolithic chip, the Atmel AT89S51 is a powerful microcontroller which provides a ly-flexible and cost-effective solution to many embedded control applications.

e AT89S51 provides the following standard features: 4K bytes of Flash, 128 bytes AM, 32 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a -vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, clock 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 rrupt system to continue functioning. The Power-down mode saves the RAM con- s but freezes the oscillator, disabling all other chip functions until the next external rrupt or hardware reset.

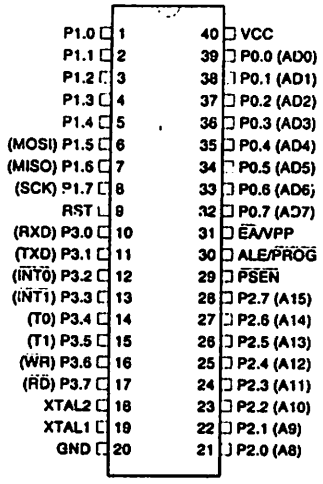
8-bit Microcontroller with 4K Bytes In-System Programmable Flash

AT89S51

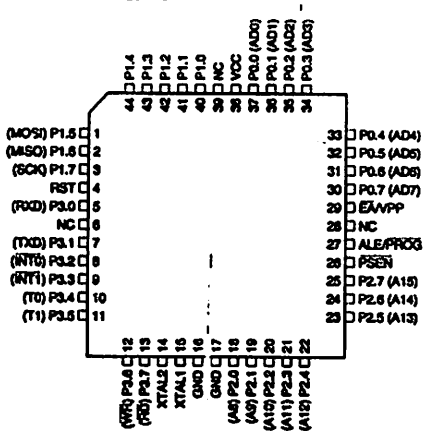
2487C–MICRO–03/05

2. Pin Configurations

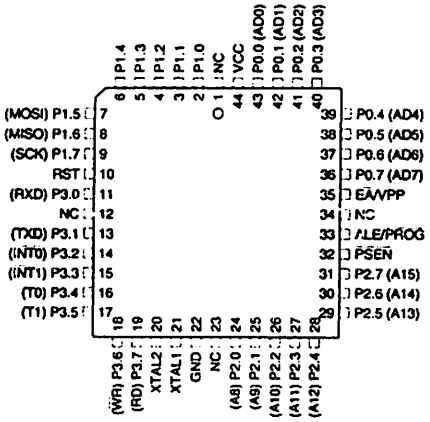
2.1 40-lead PDIP



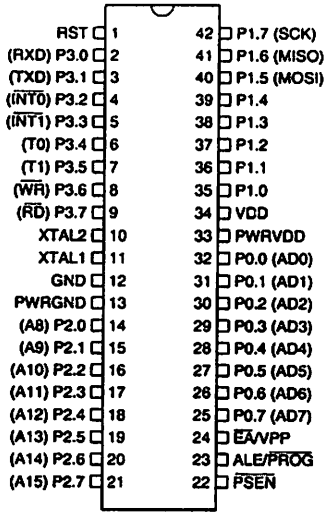
2.2 44-lead TQFP



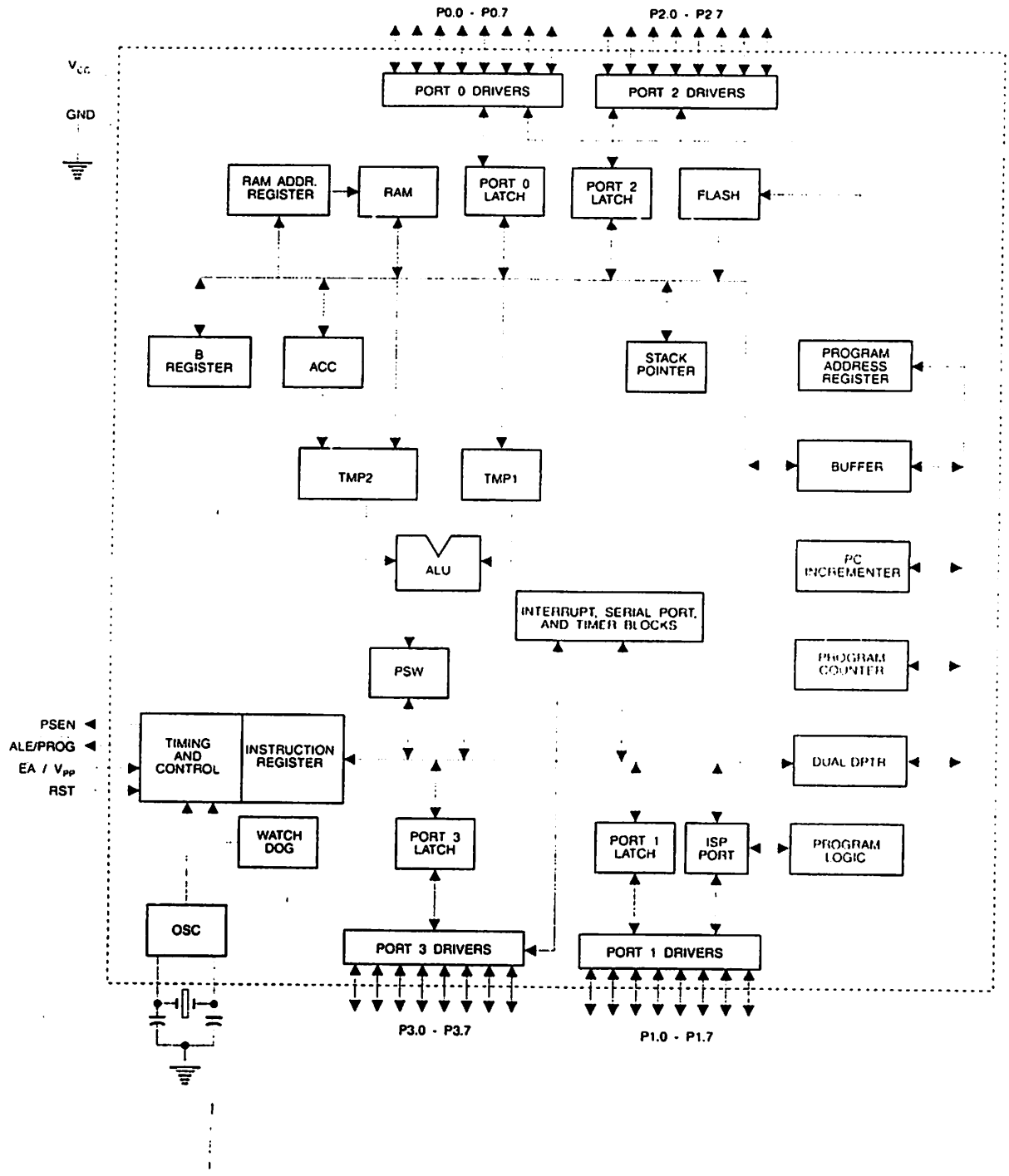
2.3 44-lead PLCC



2.4 42-lead PDIP



Block Diagram



4. Pin Description

4.1 VCC

Supply voltage (all packages except 42-PDIP).

4.2 GND

Ground (all packages except 42-PDIP; for 42-PDIP GND connects only the logic core and the embedded program memory).

4.3 VDD

Supply voltage for the 42-PDIP which connects only the logic core and the embedded program memory.

4.4 PWRVDD

Supply voltage for the 42-PDIP which connects only the I/O Pad Drivers. The application board **MUST** connect both VDD and PWRVDD to the board supply voltage.

4.5 PWRGND

Ground for the 42-PDIP which connects only the I/O Pad Drivers. PWRGND and GND are weakly connected through the common silicon substrate, but not through any metal link. The application board **MUST** connect both GND and PWRGND to the board ground.

4.6 Port 0

Port 0 is an 8-bit open drain bi-directional 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.**

4.7 Port 1

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

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)

.8 Port 2

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

Port 2 emits the high-order address byte during fetches from external program memory and during accesses to external data memory that use 16-bit addresses (MOVX @ DPTR). In this application, Port 2 uses strong internal pull-ups when emitting 1s. During accesses to external data memory that use 8-bit addresses (MOVX @ RI), Port 2 emits the contents of the P2 Special Function Register.

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

9 Port 3

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

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

Port 3 also serves the functions of various special features of the 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	$\overline{INT0}$ (external interrupt 0)
P3.3	$\overline{INT1}$ (external interrupt 1)
P3.4	T0 (timer 0 external input)
P3.5	T1 (timer 1 external input)
P3.6	$\overline{WR}$ (external data memory write strobe)
P3.7	$\overline{RD}$ (external data memory read strobe)

0 RST

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

1 ALE/$\overline{PROG}$

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

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

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

.12 $\overline{\text{PSEN}}$

Program Store Enable ($\overline{\text{PSEN}}$) is the read strobe to external program memory.

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

13 $\overline{\text{EA}}$/VPP

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

$\overline{\text{EA}}$ 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.

14 XTAL1

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

15 XTAL2

Output from the inverting oscillator amplifier

Special Function Registers

A map of the on-chip memory area called the Special Function Register (SFR) space is shown in Table 5-1.

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 5-1. 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

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

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

Table 5-2. 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 5-3. 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

Memory Organization

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

1 Program 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.

2 Data 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.

Watchdog Timer (One-time Enabled with Reset-out)

The WDT is intended as a recovery method in situations where the CPU may be subjected to software upsets. The WDT consists of a 14-bit counter and the Watchdog Timer Reset (WDTRST) SFR. The WDT is defaulted to disable from exiting reset. To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, it will increment every machine cycle while the oscillator is running. The WDT timeout period is dependent on the external clock frequency. There is no way to disable the WDT except through reset (either hardware reset or WDT overflow reset). When WDT overflows, it will drive an output RESET HIGH pulse at the RST pin.

Using 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.

.2 WDT During Power-down and Idle

In Power-down mode the oscillator stops, which means the WDT also stops. While in Power-down mode, the user does not need to service the WDT. There are two methods of exiting Power-down mode: by a hardware reset or via a level-activated external interrupt, which is enabled prior to entering Power-down mode. When Power-down is exited with hardware reset, servicing the WDT should occur as it normally does whenever the 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.

UART

The UART in the AT89S51 operates the same way as the UART in the AT89C51. For further information on the UART operation, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC[illegible].PDF

Timer 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, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC[illegible].PDF

). Interrupts

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 10-1.

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 10-1 shows that bit positions IE.6 and IE.5 are 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 10-1. Interrupt Enable (IE) Register

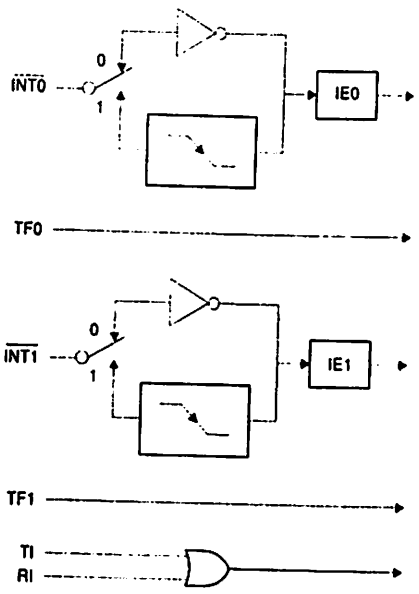
(MSB)							(LSB)
EA	–	–	ES	ET1	EX1	ET0	EX0

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.		

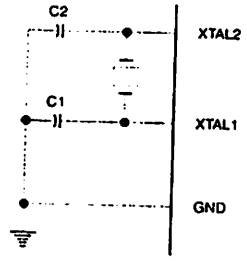
Figure 10-1. Interrupt Sources



Oscillator Characteristics

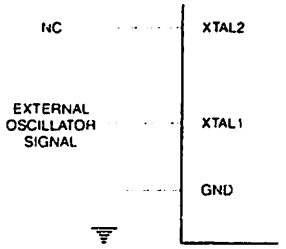
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 11-1. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 11-2. There are no requirements on the duty cycle of the external clock signal, since the input to the internal clocking circuitry is through a divide-by-two flip-flop, but minimum and maximum voltage high and low time specifications must be observed.

Figure 11-1. Oscillator Connections



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

Figure 11-2. External Clock Drive Configuration



2. Idle Mode

In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special 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.

3. Power-down Mode

In the Power-down mode, the oscillator is stopped, and the instruction that invokes Power-down is the last instruction executed. The on-chip RAM and Special Function Registers retain their values until the Power-down mode is terminated. Exit from Power-down mode can be initiated either by a hardware reset or by activation of an enabled external interrupt ($\overline{INT0}$ 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 13-1. 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

4. Program Memory Lock Bits

The AT89S51 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in Table 14-1.

Table 14-1. 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.

5. Programming the Flash – Parallel 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 Modes table (Table 17-1) and Figure 17-1 and Figure 17-2. 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/$\overline{Busy}$: The progress of byte programming can also be monitored by the RDY/$\overline{BSY}$ output signal. P3.0 is pulled low after ALE goes high during programming to indicate $\overline{BUSY}$. P3.0 is pulled high again when programming is done to indicate READY.

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

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

(000H) = 1EH indicates manufactured by Atmel
(100H) = 51H indicates AT89S51
(200H) = 06H

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

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

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

Programming the Flash – Serial Mode

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

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

Either an external system clock can be supplied at pin XTAL1 or a crystal needs to be connected across pins XTAL1 and XTAL2. The maximum serial clock (SCK) frequency should be less than 1/16 of the crystal frequency. With a 33 MHz oscillator clock, the maximum SCK frequency is 2 MHz.

Serial Programming Algorithm

To program and verify the AT89S51 in the serial programming mode, the following sequence is recommended:

1. Power-up sequence:
   a. Apply power between VCC and GND pins.
   b. 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):

1. Set XTAL1 to "L" (if a crystal is not used).
2. Set RST to "L".
3. Turn V_{CC} power off.

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

6.2 Serial Programming Instruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in the "Serial Programming Instruction Set" on page 20.

7. Programming Interface – Parallel 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.

Most major worldwide programming vendors offer worldwide support for the Atmel AT89 microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Table 17-1. Flash Programming Modes

Mode	V_{CC}	RST	$\overline{\text{PSEN}}$	ALE/ $\overline{\text{PROG}}$	$\overline{\text{EA}}$/ V_{PP}	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.3-0 Address	P1.7-0 Address
Write Code Data	5V	H	L	~	12V	L	H	H	H	H	D_{IN}	A11-8	A7-0
Read Code Data	5V	H	L	H	H	L	L	L	H	H	D_{OUT}	A11-8	A7-0
Write Lock Bit 1	5V	H	L	~	12V	H	H	H	H	H	X	X	X
Write Lock Bit 2	5V	H	L	~	12V	H	H	H	L	L	X	X	X
Write Lock Bit 3	5V	H	L	~	12V	H	L	H	H	L	X	X	X
Read Lock Bits 1, 2, 3	5V	H	L	H	H	H	H	L	H	L	P0.2, P0.3, P0.4	X	X
Chip Erase	5V	H	L	~	12V	H	L	H	L	L	X	X	X
Read Atmel ID	5V	H	L	H	H	L	L	L	L	L	1EH	0000	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	51H	0001	00H
Read Device ID	5V	H	L	H	H	L	L	L	L	L	06H	0010	00H

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

Figure 17-1. Programming the Flash Memory (Parallel Mode)

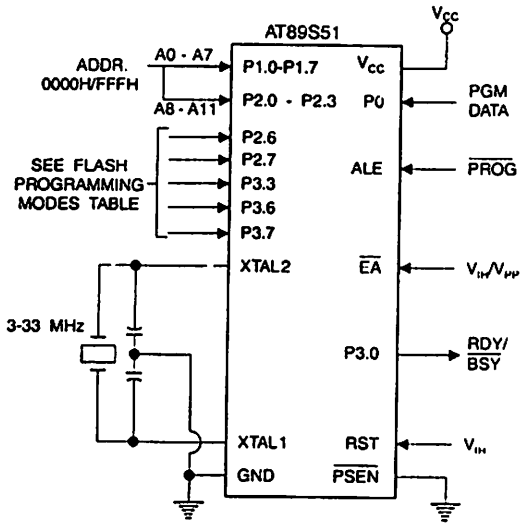
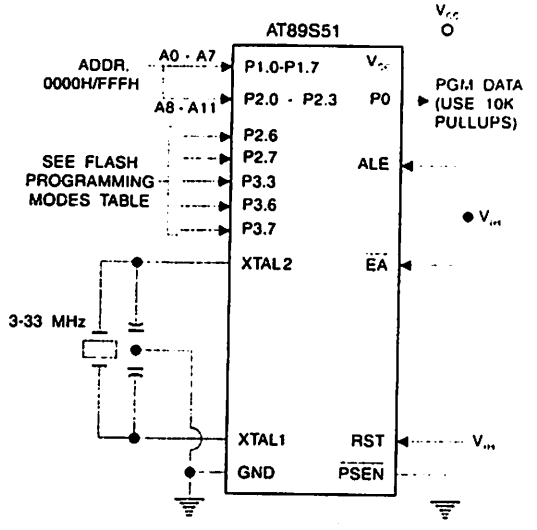


Figure 17-2. Verifying the Flash Memory (Parallel Mode)

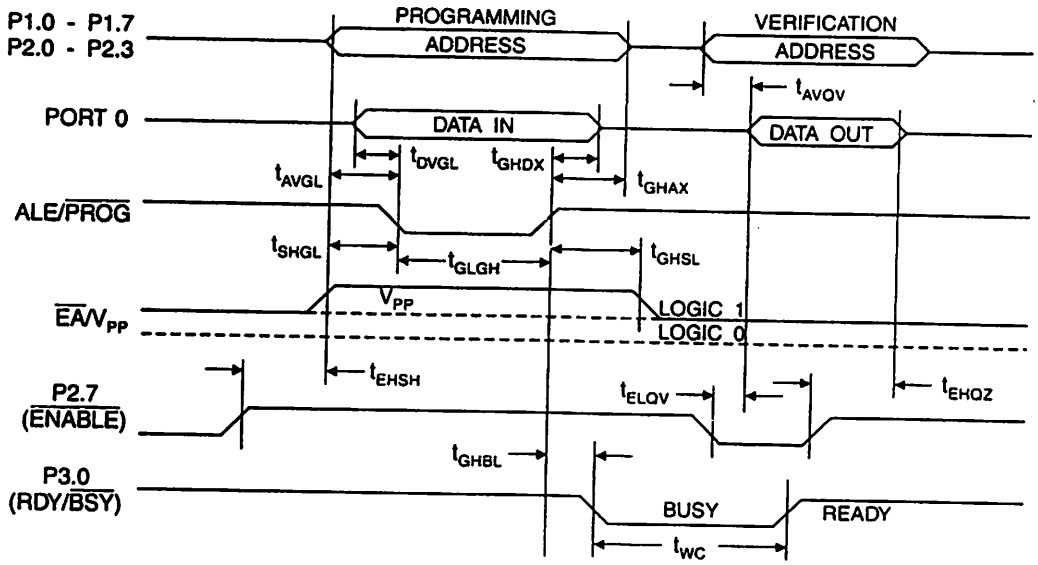


8. Flash Programming and Verification Characteristics (Parallel Mode)

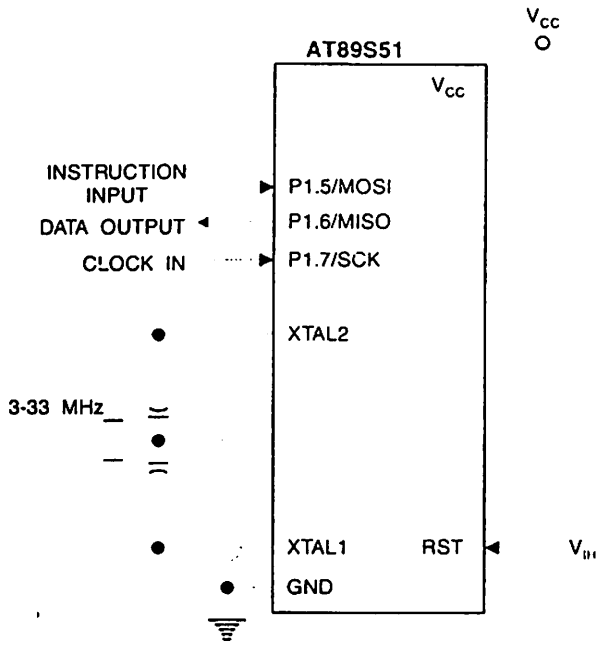
$_A$ = 20°C to 30°C, V_{CC} = 4.5 to 5.5V

Symbol	Parameter	Min	Max	Units
V_{PP}	Programming Supply Voltage	11.5	12.5	V
I_{PP}	Programming Supply Current		10	mA
'CC	V_{CC} Supply Current		30	mA
$1/t_{CLCL}$	Oscillator Frequency	3	33	MHz
AVGL	Address Setup to $\overline{PROG}$ Low	48 t_{CLCL}		
GHAX	Address Hold After $\overline{PROG}$	48 t_{CLCL}		
DVGL	Data Setup to $\overline{PROG}$ Low	48 t_{CLCL}		
GHDX	Data Hold After $\overline{PROG}$	48 t_{CLCL}		
EHSH	P2.7 ($\overline{ENABLE}$) High to V_{PP}	48 t_{CLCL}		
SHGL	V_{PP} Setup to $\overline{PROG}$ Low	10		µs
GHSL	V_{PP} Hold After $\overline{PROG}$	10		µs
GLGH	$\overline{PROG}$ Width	0.2	1	µs
AVQV	Address to Data Valid		48t_{CLCL}	
ELQV	$\overline{ENABLE}$ Low to Data Valid		48t_{CLCL}	
EHQZ	Data Float After $\overline{ENABLE}$	0	48t_{CLCL}	
GHBL	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
WC	Byte Write Cycle Time		50	µs

ure 18-1. Flash Programming and Verification Waveforms – Parallel Mode

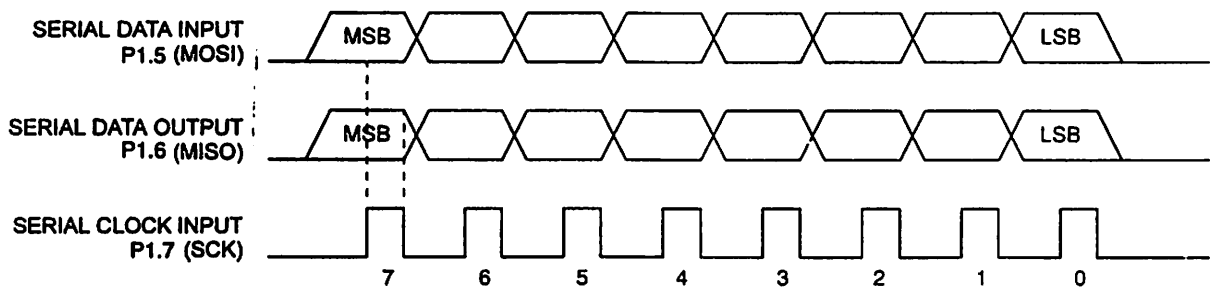


gure 18-2. Flash Memory Serial Downloading



). Flash Programming and Verification Waveforms – Serial Mode

ıure 19-1. Serial Programming Waveforms



Serial Programming Instruction Set

truction	Instruction Format				Operation
	Byte 1	Byte 2	Byte 3	Byte 4	
gramming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output on MISO)	Enable Serial Programming while RST is high
ɔ Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
ıd Program Memory :e 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
e Program Memory e 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
ɔ Lock Bits[1]	1010 1100	1110 00 B1 B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (1).
d 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")
d Signature Bytes	0010 1000	xxxx A11 A10 A9 A8	A7 xxx xxx0	Signature Byte	Read Signature Byte
d Program Memory ɟe 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)
ɜ Program Memory ɟe 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)

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

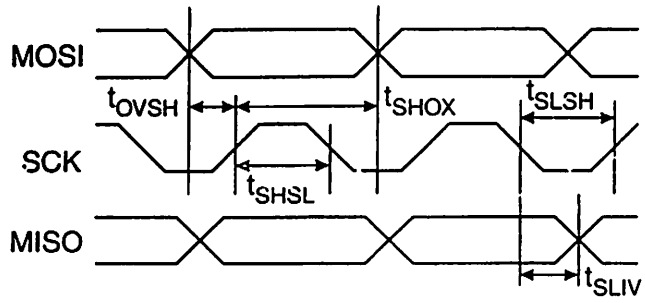
   } Each of the lock bit modes need to be activated sequentially before Mode 4 can be executed.

Reset signal is high, SCK should be low for at least 64 system clocks before it goes high to clock in the enable data . No pulsing of Reset signal is necessary. SCK should be no faster than 1/16 of the system clock at XTAL1.

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

Serial Programming Characteristics

e 21-1. Serial Programming Timing



21-1. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

›ol	Parameter	Min	Typ	Max	Units
;L	Oscillator Frequency	3		33	MHz
	Oscillator Period	30			ns
	SCK Pulse Width High	8 t_{CLCL}			ns
	SCK Pulse Width Low	8 t_{CLCL}			ns
	MOSI Setup to SCK High	t_{CLCL}			ns
	MOSI Hold after SCK High	2 t_{CLCL}			ns
	SCK Low to MISO Valid	10	16	32	ns
:	Chip Erase Instruction Cycle Time			500	ms
	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	µs

Absolute Maximum Ratings*

ıting Temperature	-55°C to +125°C
ge Temperature	-65°C to +150°C
ge on Any Pin lespect to Ground	-1.0V to +7.0V
ıum Operating Voltage	6.6V
utput Current	15.0 mA

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

DC Characteristics

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

mbol	Parameter	Condition	Min	Max	Units
	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ± 10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-300	µA
	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
ST	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. Under steady state (non-transient) conditions, I_{OL} must be externally limited as follows:
   Maximum I_{OL} per port pin: 10 mA
   Maximum I_{OL} per 8-bit port:
   Port 0: 26 mA Ports 1, 2, 3: 15 mA
   Maximum total I_{OL} for all output pins: 71 mA
   If I_{OL} exceeds the test condition, V_{OL} may exceed the related specification. Pins are not guaranteed to sink current greater than the listed test conditions.
2. Minimum V_{CC} for Power-down is 2V.

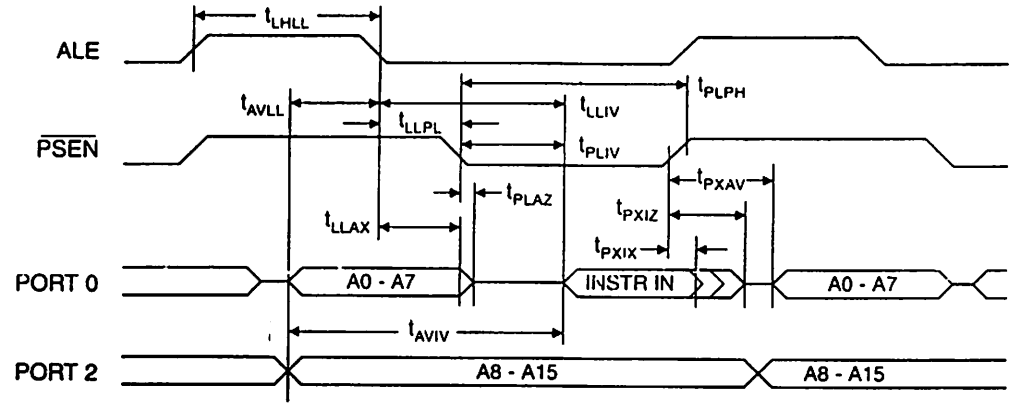
AC Characteristics

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

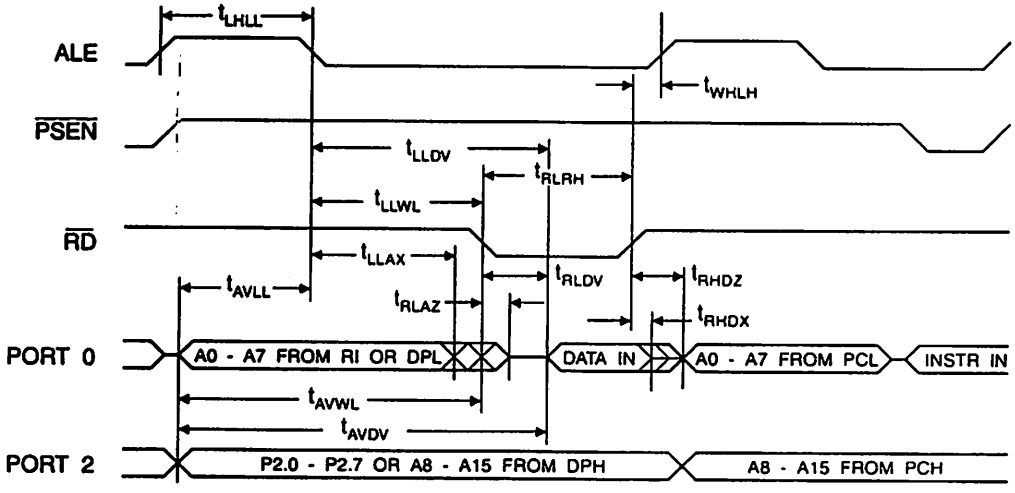
External Program and Data Memory Characteristics

ıbol	Parameter	12 MHz Oscillator		Variable Oscillator		Units
		Min	Max	Min	Max	
CL	Oscillator Frequency			0	33	MHz
	ALE Pulse Width	127		$2t_{CLCL}$-40		ns
	Address Valid to ALE Low	43		t_{CLCL}-25		ns
	Address Hold After ALE Low	48		t_{CLCL}-25		ns
	ALE Low to Valid Instruction In		233		$4t_{CLCL}$-65	ns
	ALE Low to $\overline{PSEN}$ Low	43		t_{CLCL}-25		ns
	$\overline{PSEN}$ Pulse Width	205		$3t_{CLCL}$-45		ns
	$\overline{PSEN}$ Low to Valid Instruction In		145		$3t_{CLCL}$-60	ns
	Input Instruction Hold After $\overline{PSEN}$	0		0		ns
	Input Instruction Float After $\overline{PSEN}$		59		t_{CLCL}-25	ns
	$\overline{PSEN}$ to Address Valid	75		t_{CLCL}-8		ns
	Address to Valid Instruction In		312		$5t_{CLCL}$-80	ns
	$\overline{PSEN}$ Low to Address Float		10		10	ns
	$\overline{RD}$ Pulse Width	400		$6t_{CLCL}$-100		ns
	$\overline{WR}$ Pulse Width	400		$6t_{CLCL}$-100		ns
	$\overline{RD}$ Low to Valid Data In		252		$5t_{CLCL}$-90	ns
	Data Hold After $\overline{RD}$	0		0		ns
	Data Float After $\overline{RD}$		97		$2t_{CLCL}$-28	ns
	ALE Low to Valid Data In		517		$8t_{CLCL}$-150	ns
	Address to Valid Data In		585		$9t_{CLCL}$-165	ns
	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	200	300	$3t_{CLCL}$-50	$3t_{CLCL}$+50	ns
	Address to $\overline{RD}$ or $\overline{WR}$ Low	203		$4t_{CLCL}$-75		ns
	Data Valid to $\overline{WR}$ Transition	23		t_{CLCL}-30		ns
	Data Valid to $\overline{WR}$ High	433		$7t_{CLCL}$-130		ns
	Data Hold After $\overline{WR}$	33		t_{CLCL}-25		ns
	$\overline{RD}$ Low to Address Float		0		0	ns
	$\overline{RD}$ or $\overline{WR}$ High to ALE High	43	123	t_{CLCL}-25	t_{CLCL}+25	ns

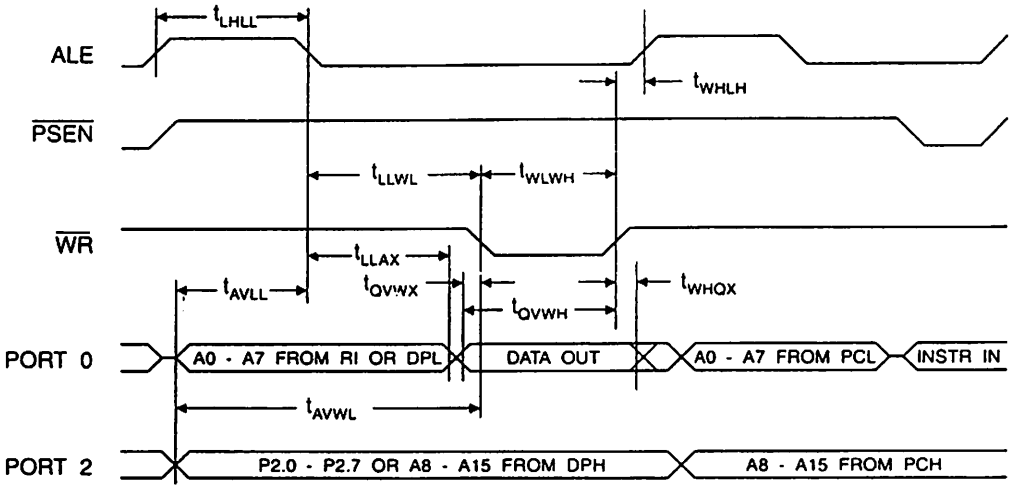
External Program Memory Read Cycle



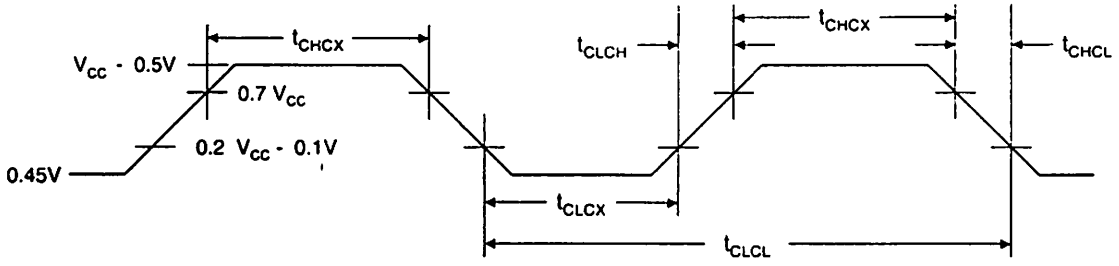
External Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

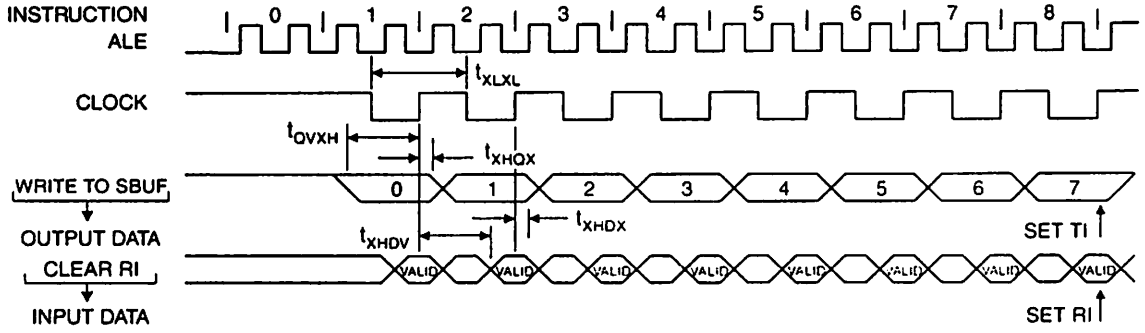
Symbol	Parameter	Min	Max	Units
1/t_{CLCL}	Oscillator Frequency	0	33	MHz
t_{CLCL}	Clock Period	30		ns
t_{CHCX}	High Time	12		ns
t_{CLCX}	Low Time	12		ns
t_{CLCH}	Rise Time		5	ns
t_{CHCL}	Fall Time		5	ns

Serial Port Timing: Shift Register Mode Test Conditions

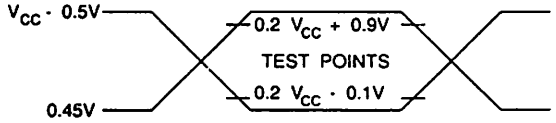
values in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

mbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
KL	Serial Port Clock Cycle Time	1.0		12 t_{CLCL}		µs
XH	Output Data Setup to Clock Rising Edge	700		10 t_{CLCL}-133		ns
QX	Output Data Hold After Clock Rising Edge	50		2 t_{CLCL}-80		ns
DX	Input Data Hold After Clock Rising Edge	0		0		ns
DV	Clock Rising Edge to Input Data Valid		700		10 t_{CLCL}-133	ns

Shift Register Mode Timing Waveforms

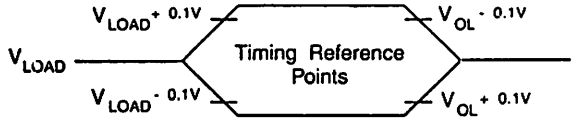


AC Testing Input/Output Waveforms[1]



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.

Float 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.

. Ordering Information

.1 Standard Package

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC AT89S51-24JC AT89S51-24PC AT89S51-24SC	44A 44J 40P6 42PS6	Commercial (0°C to 70°C)
		AT89S51-24AI AT89S51-24JI AT89S51-24PI AT89S51-24SI	44A 44J 40P6 42PS6	Industrial (-40°C to 85°C)
33	4.5V to 5.5V	AT89S51-33AC AT89S51-33JC AT89S51-33PC AT89S51-33SC	44A 44J 40P6 42PS6	Commercial (0°C to 70°C)

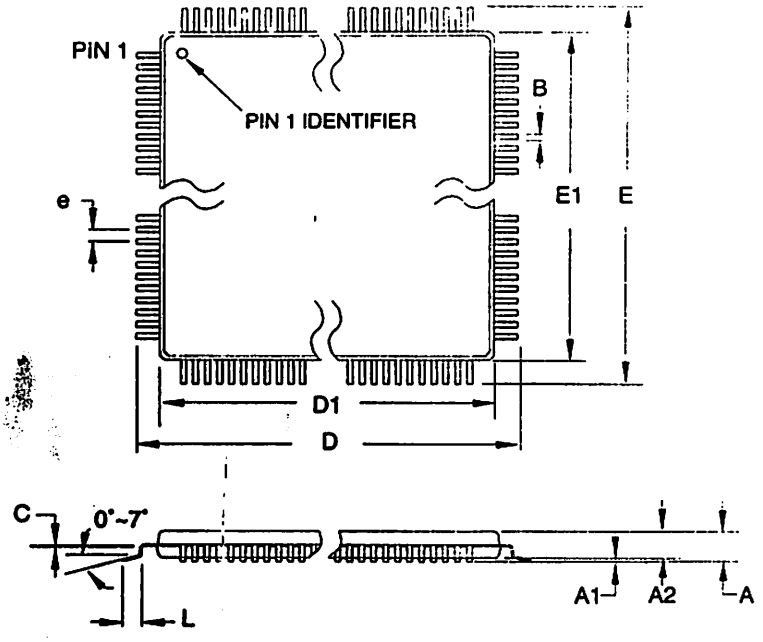
2 Green Package Option (Pb/Halide-free)

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AU AT89S51-24JU AT89S51-24PU	44A 44J 40P6	Industrial (-40°C to 85°C)

	Package Type
A	44-lead, Thin Plastic Gull Wing Quad Flatpack (TQFP)
J	44-lead, Plastic J-leaded Chip Carrier (PLCC)
P6	40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
PS6	42-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)

Packaging Information

44A – TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

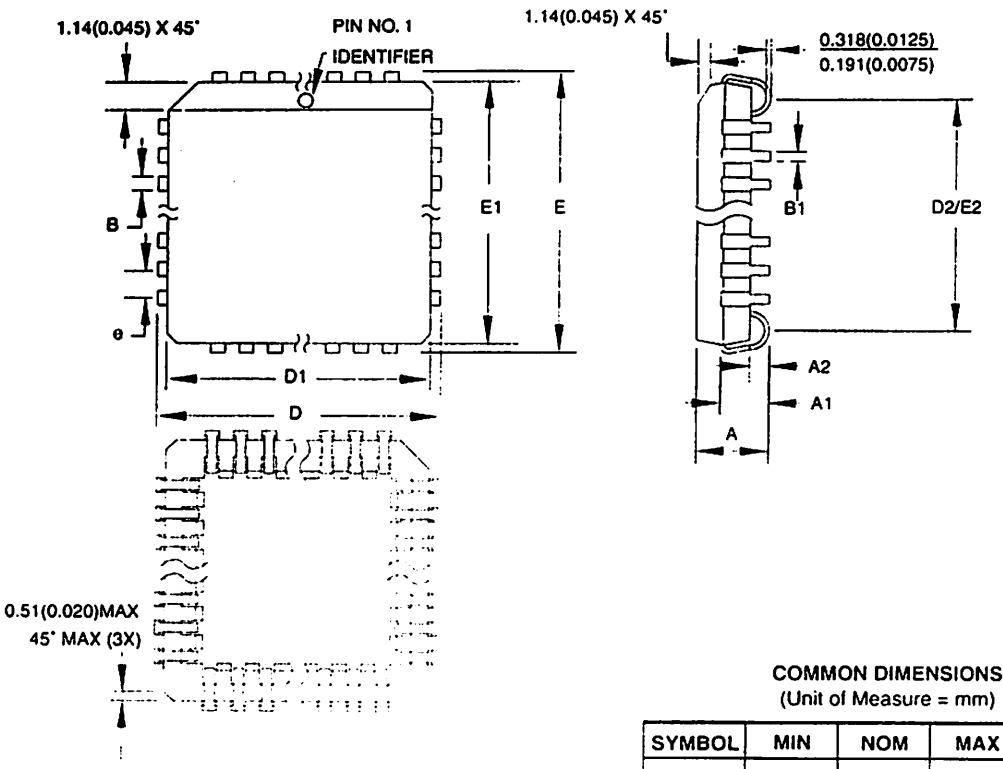
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

Notes:
1. This package conforms to JEDEC reference MS-026, Variation ACB.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is 0.25 mm per side. Dimensions D1 and E1 are maximum plastic body size dimensions including mold mismatch.
3. Lead coplanarity is 0.10 mm maximum.

10/5/2001

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	44A, 44-lead, 10 x 10 mm Body Size, 1.0 mm Body Thickness, 0.8 mm Lead Pitch, Thin Profile Plastic Quad Flat Package (TQFP)	44A	B

44J – PLCC



COMMON DIMENSIONS
(Unit of Measure = mm)

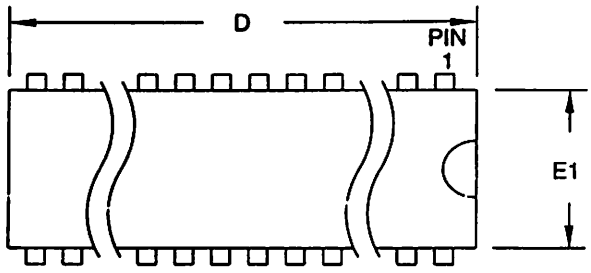
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

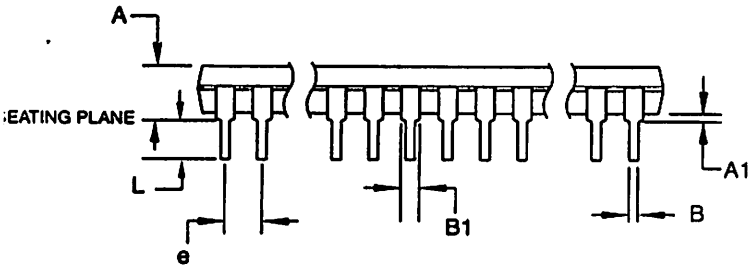
Notes:
1. This package conforms to JEDEC reference MS-018, Variation AC.
2. Dimensions D1 and E1 do not include mold protrusion. Allowable protrusion is .010"(0.254 mm) per side. Dimension D1 and E1 include mold mismatch and are measured at the extreme material condition at the upper or lower parting line.
3. Lead coplanarity is 0.004" (0.102 mm) maximum.

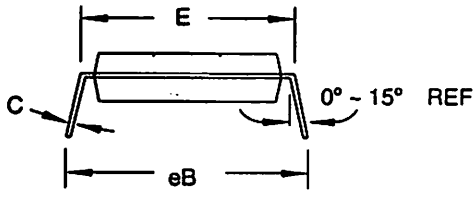
10/04/01

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**44J**, 44-lead, Plastic J-leaded Chip Carrier (PLCC)	44J	B

40P6 – PDIP







COMMON DIMENSIONS
(Unit of Measure = mm)

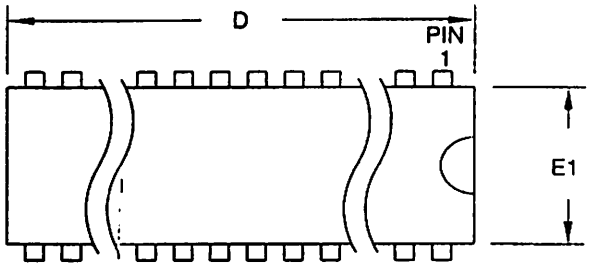
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

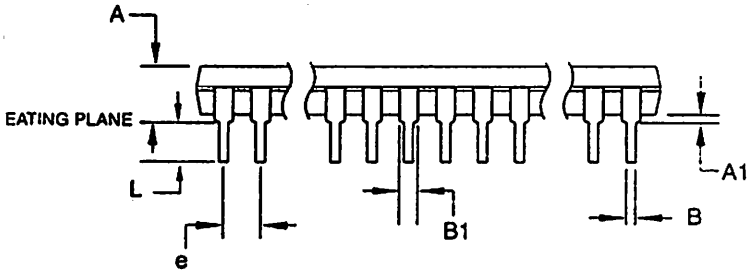
Notes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

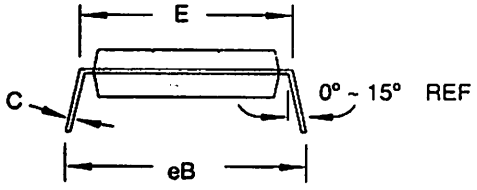
09/28/01

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

42PS6 – PDIP







COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.83	
A1	0.51	–	–	
D	36.70	–	36.96	Note 2
E	15.24	–	15.88	
E1	13.46	–	13.97	Note 2
B	0.38	–	0.56	
B1	0.76	–	1.27	
L	3.05	–	3.43	
C	0.20	–	0.30	
eB	–	–	18.55	
e	1.78 TYP			

Notes:
1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion. Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

11/6/03

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**42PS6**, 42-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	42PS6	A

ıel Corporation

:25 Orchard Parkway
ın Jose, CA 95131, USA
el: 1(408) 441-0311
ıx: 1(408) 487-2600

gional Headquarters

ope
mel Sarl
oute des Arsenaux 41
ıse Postale 80
H-1705 Fribourg
vitzerland
l: (41) 26-426-5555
x: (41) 26-426-5500

om 1219
inachem Golden Plaza
Mody Road Tsimshatsui
st Kowloon
ng Kong
l: (852) 2721-9778
x: (852) 2722-1369

n
Tonetsu Shinkawa Bldg.
24-8 Shinkawa
uo-ku, Tokyo 104-0033
oan
l: (81) 3-3523-3551
x: (81) 3-3523-7581

Atmel Operations

Memory
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

Microcontrollers
2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 436-4314

La Chantrerie
BP 70602
44306 Nantes Cedex 3, France
Tel: (33) 2-40-18-18-18
Fax: (33) 2-40-18-19-60

ASIC/ASSP/Smart Cards
Zone Industrielle
13106 Rousset Cedex, France
Tel: (33) 4-42-53-60-00
Fax: (33) 4-42-53-60-01

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Scottish Enterprise Technology Park
Maxwell Building
East Kilbride G75 0QR, Scotland
Tel: (44) 1355-803-000
Fax: (44) 1355-242-743

RF/Automotive
Theresienstrasse 2
Postfach 3535
74025 Heilbronn, Germany
Tel: (49) 71-31-67-0
Fax: (49) 71-31-67-2340

1150 East Cheyenne Mtn. Blvd.
Colorado Springs, CO 80906, USA
Tel: 1(719) 576-3300
Fax: 1(719) 540-1759

Biometrics/Imaging/Hi-Rel MPU/
High Speed Converters/RF Datacom
Avenue de Rochepleine
BP 123
38521 Saint-Egreve Cedex, France
Tel: (33) 4-76-58-30-00
Fax: (33) 4-76-58-34-80

Literature Requests
www.atmel.com/literature

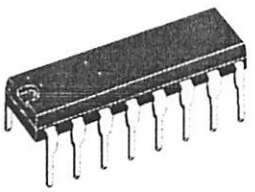
Printed on recycled paper.

2487C–MICRO–03/05 /xM

ULN2001A-ULN2002A
ULN2003A-ULN2004A

SEVEN DARLINGTON ARRAYS

- *SEVEN DARLINGTONS PER PACKAGE*
- OUTPUT CURRENT 500mA PER DRIVER (600mA PEAK)
- OUTPUT VOLTAGE 50V
- INTEGRATED SUPPRESSION DIODES FOR INDUCTIVE LOADS
- OUTPUTS CAN BE PARALLELED FOR *HIGHER CURRENT*
- TTL/CMOS/PMOS/DTL COMPATIBLE INPUTS
- INPUTS PINNED OPPOSITE OUTPUTS TO SIMPLIFY LAYOUT



DIP16

ORDERING NUMBERS: ULN2001A/2A/3A/4A

SO16

ORDERING NUMBERS: ULN2001D/2D/3D/4D

DESCRIPTION

The ULN2001A, ULN2002A, ULN2003 and ULN2004A are high voltage, high current darlington arrays each containing seven open collector darlington pairs with common emitters. Each channel rated at 500mA and can withstand peak currents of 600mA. Suppression diodes are included for inductive load driving and the inputs are pinned opposite the outputs to simplify board layout.

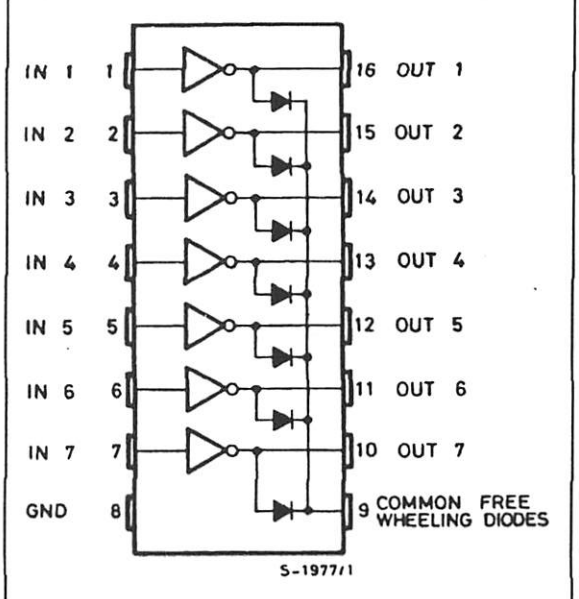
The four versions interface to all common logic families :

ULN2001A	General Purpose, DTL, TTL, PMOS, CMOS
ULN2002A	14-25V PMOS
ULN2003A	5V TTL, CMOS
ULN2004A	6–15V CMOS, PMOS

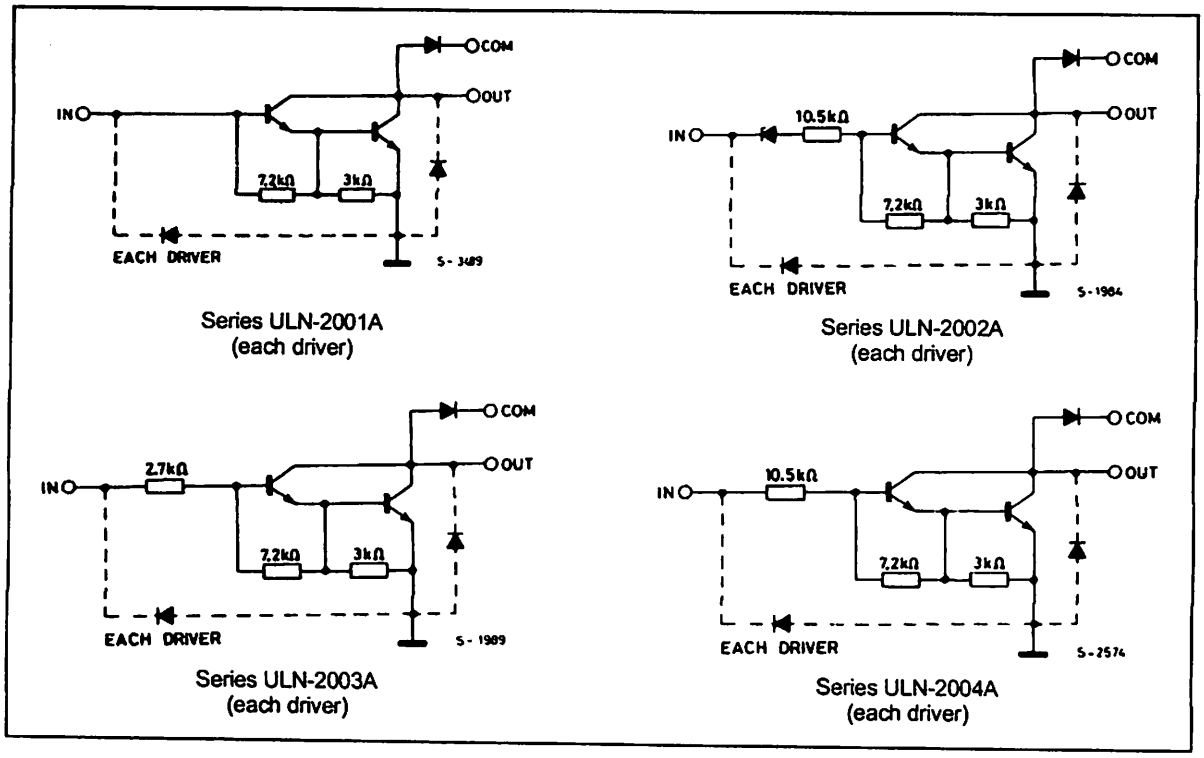
These versatile devices are useful for driving a wide range of loads including solenoids, relays DC motors, LED displays filament lamps, thermal printheads and high power buffers.

The ULN2001A/2002A/2003A and 2004A are supplied in 16 pin plastic DIP packages with a copper leadframe to reduce thermal resistance. They are available also in small outline package (SO-16) as ULN2001D/2002D/2003D/2004D.

PIN CONNECTION



SCHEMATIC DIAGRAM



ABSOLUTE MAXIMUM RATINGS

Symbol	Parameter	Value	Unit
V_o	Output Voltage	50	V
V_{in}	Input Voltage (for ULN2002A/D - 2003A/D - 2004A/D)	30	V
I_c	Continuous Collector Current	500	mA
I_b	Continuous Base Current	25	mA
T_{amb}	Operating Ambient Temperature Range	– 20 to 85	°C
T_{stg}	Storage Temperature Range	– 55 to 150	°C
T_j	Junction Temperature	150	°C

THERMAL DATA

Symbol	Parameter		DIP16	SO16	Unit
$R_{th\ j\text{-}amb}$	Thermal Resistance Junction-ambient	Max.	70	120	°C/W

ELECTRICAL CHARACTERISTICS (T_{amb} = 25°C unless otherwise specified)

Symbol	Parameter	Test Conditions	Min.	Typ.	Max.	Unit	Fig.
I_{CEX}	Output Leakage Current	V_{CE} = 50V			50	µA	1a
		T_{amb} = 70°C, V_{CE} = 50V			100	µA	1a
		T_{amb} = 70°C for ULN2002A V_{CE} = 50V, V_i = 6V			500	µA	1b
		for ULN2004A V_{CE} = 50V, V_i = 1V			500	µA	1b
$V_{CE(sat)}$	Collector-emitter Saturation Voltage	I_C = 100mA, I_B = 250µA		0.9	1.1	V	2
		I_C = 200 mA, I_B = 350µA		1.1	1.3	V	2
		I_C = 350mA, I_B = 500µA		1.3	1.6	V	2
$I_{i(on)}$	Input Current	for ULN2002A, V_i = 17V		0.82	1.25	mA	3
		for ULN2003A, V_i = 3.85V		0.93	1.35	mA	3
		for ULN2004A, V_i = 5V		0.35	0.5	mA	3
		V_i = 12V		1	1.45	mA	3
$I_{i(off)}$	Input Current	T_{amb} = 70°C, I_C = 500µA	50	65		µA	4
$V_{i(on)}$	Input Voltage	V_{CE} = 2V				V	5
		for ULN2002A I_C = 300mA			13		
		for ULN2003A I_C = 200mA			2.4		
		I_C = 250mA			2.7		
		I_C = 300mA			3		
		for ULN2004A I_C = 125mA			5		
		I_C = 200mA			6		
		I_C = 275mA			7		
		I_C = 350mA			8		
h_{FE}	DC Forward Current Gain	for ULN2001A V_{CE} = 2V, I_C = 350mA	1000				2
C_i	Input Capacitance			15	25	pF	
t_{PLH}	Turn-on Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	
t_{PHL}	Turn-off Delay Time	0.5 V_i to 0.5 V_o		0.25	1	µs	
I_R	Clamp Diode Leakage Current	V_R = 50V			50	µA	6
		T_{amb} = 70°C, V_R = 50V			100	µA	6
V_F	Clamp Diode Forward Voltage	I_F = 350mA		1.7	2	V	7

TEST CIRCUITS

Figure 1a.

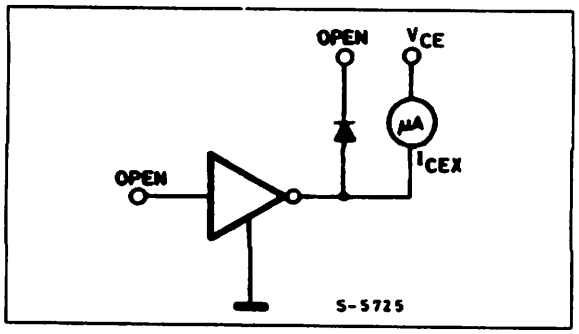


Figure 1b.

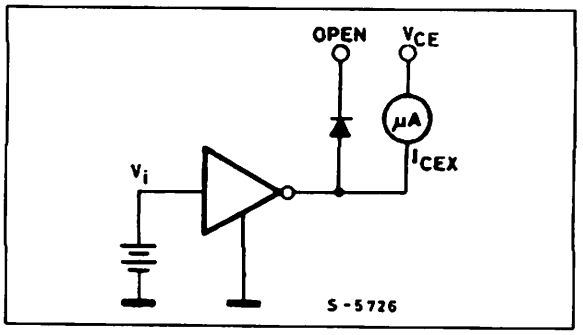


Figure 2.

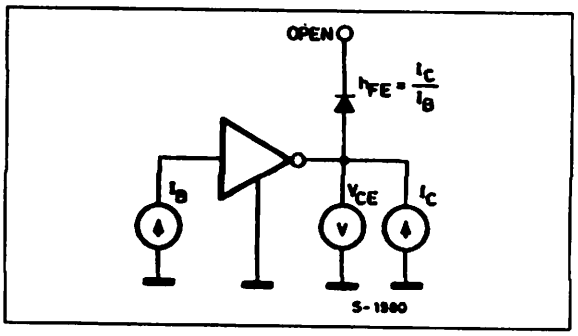


Figure 3.

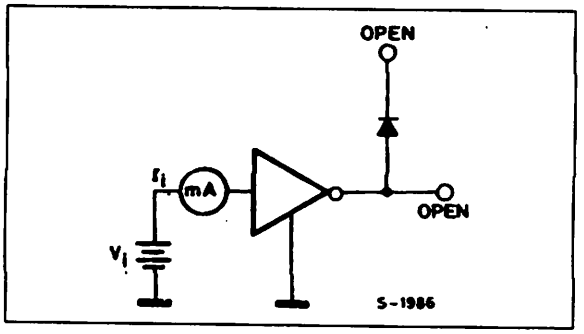


Figure 4.

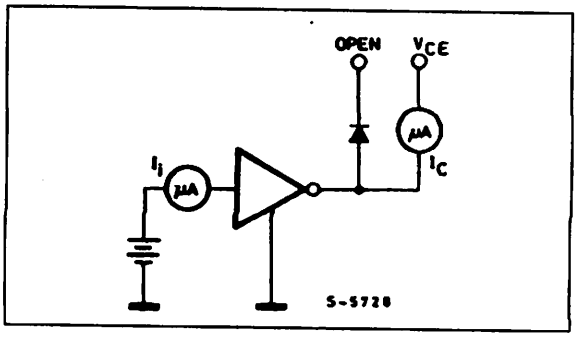


Figure 5.

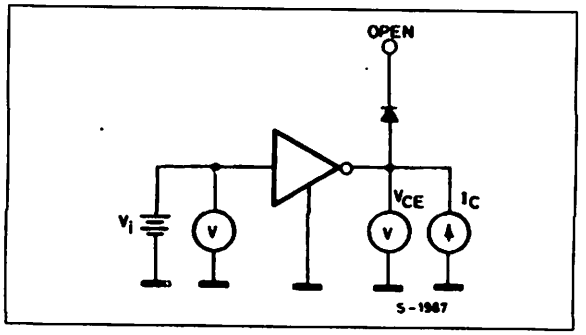


Figure 6.

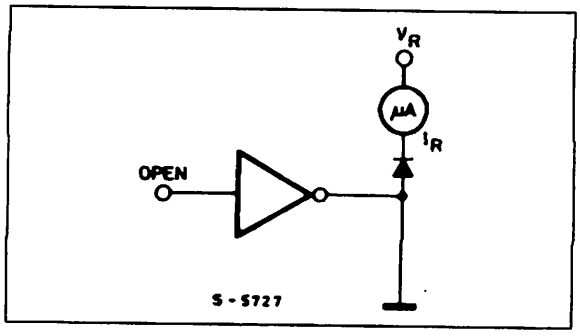


Figure 7.

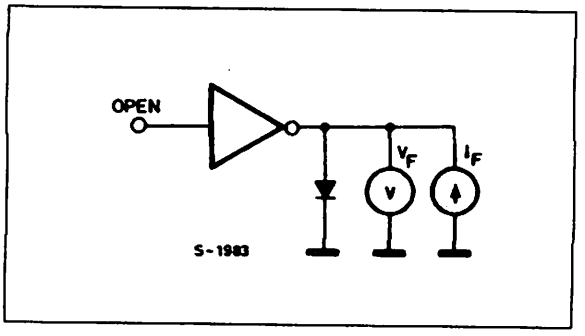


Figure 8: Collector Current versus Input Current

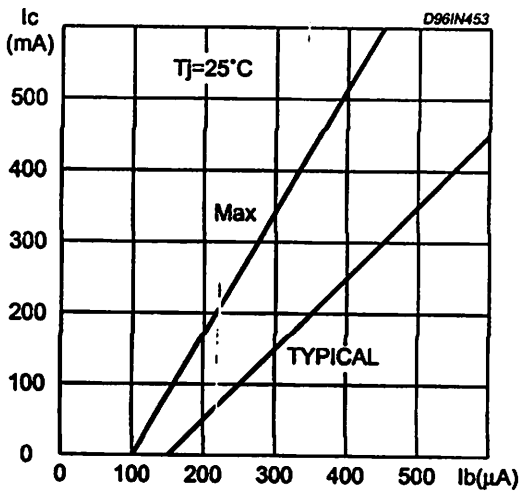


Figure 9: Collector Current versus Saturation Voltage

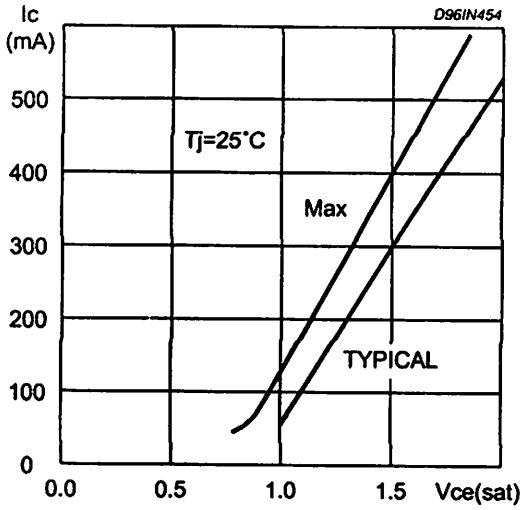


Figure 10: Peak Collector Current versus Duty Cycle

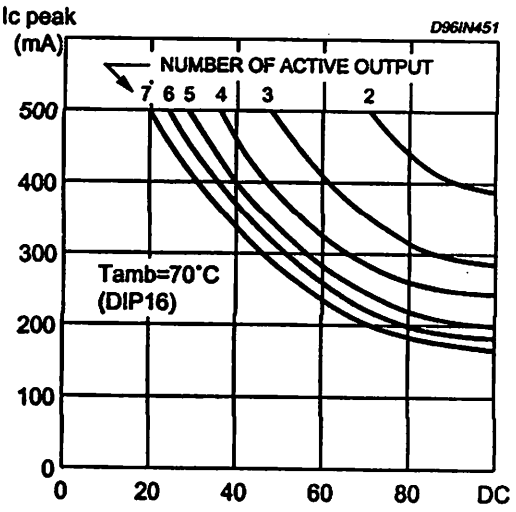
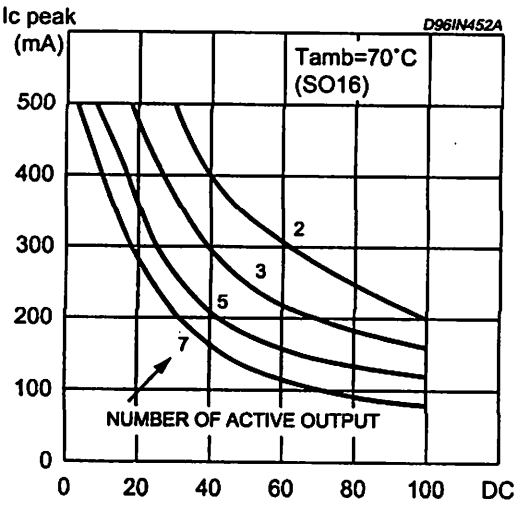
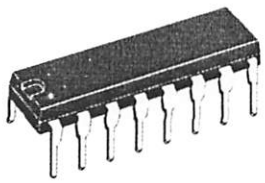


Figure 11: Peak Collector Current versus Duty Cycle

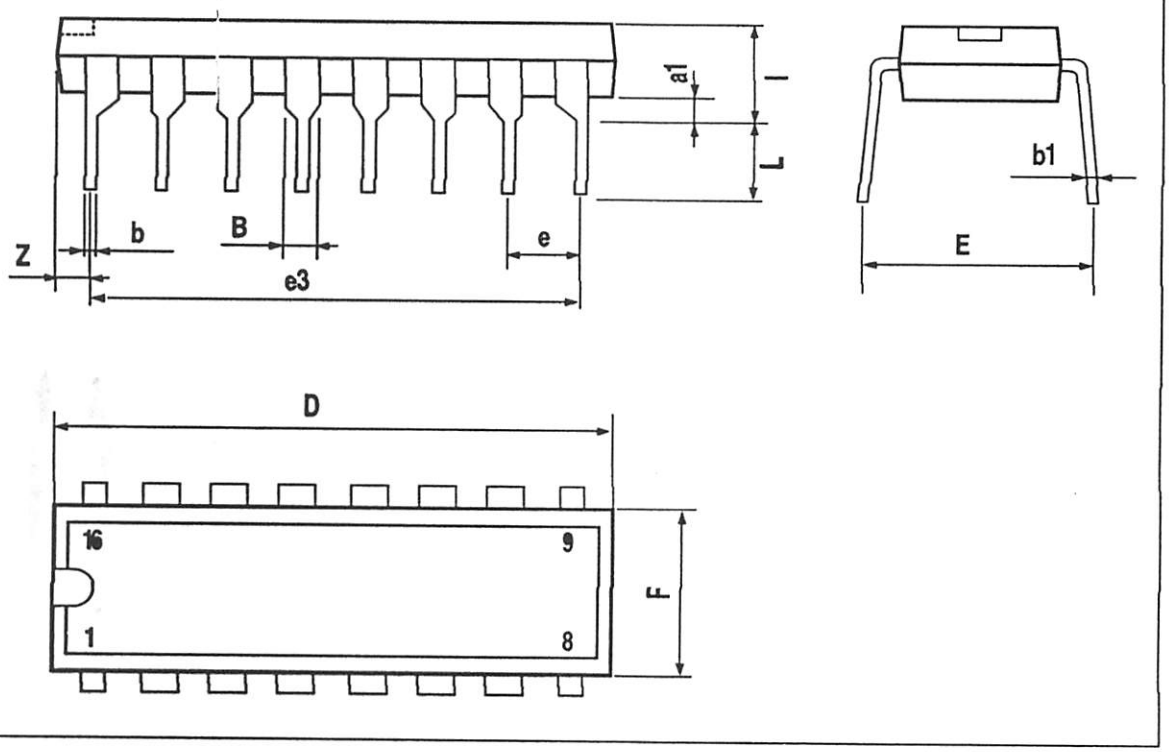


DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
a1	0.51			0.020		
B	0.77		1.65	0.030		0.065
b		0.5			0.020	
b1		0.25			0.010	
D			20			0.787
E		8.5			0.335	
e		2.54			0.100	
e3		17.78			0.700	
F			7.1			0.280
I			5.1			0.201
L		3.3			0.130	
Z			1.27			0.050

OUTLINE AND MECHANICAL DATA

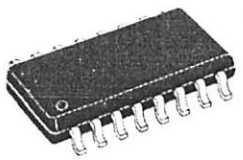


DIP16



DIM.	mm			inch		
	MIN.	TYP.	MAX.	MIN.	TYP.	MAX.
A			1.75			0.069
a1	0.1		0.25	0.004		0.009
a2			1.6			0.063
b	0.35		0.46	0.014		0.018
b1	0.19		0.25	0.007		0.010
C		0.5			0.020	
c1	45° (typ.)					
D (1)	9.8		10	0.386		0.394
E	5.8		6.2	0.228		0.244
e		1.27			0.050	
e3		8.89			0.350	
F (1)	3.8		4	0.150		0.157
G	4.6		5.3	0.181		0.209
L	0.4		1.27	0.016		0.050
M			0.62			0.024
S	8°(max.)					

OUTLINE AND MECHANICAL DATA



SO16 Narrow

(1) D and F do not include mold flash or protrusions. Mold flash or potrusions shall not exceed 0.15mm (.006inch).