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

RANCANG BANGUN SISTEM LAYANAN SPBU PRABAYAR MENGGUNAKAN TEKNOLOGI RFID

SKRIPSI

Disusun Oleh :
Mahardian Mahendradhata
NIM. 02.17.019

MARET 2007

LEMBAR PERSETUJUAN

RANCANG BANGUN SISTEM LAYANAN SPBU PRABAYAR MENGGUNAKAN TEKNOLOGI RFID

SKRIPSI

Disusun Dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh Gelar Sarjana Teknik Elektronika Strata Satu (S-1)

Disusun Oleh :

MAHARDIAN MAHENDRADHATA
NIM : 02.17.019

Diperiksa dan Disetujui

Dosen Pembimbing

Joseph Dedy Irawan, ST, MT
NIP. 132 315 178

Mengetahui

Ketua Jurusan Teknik Elektro S-1

Ir. F. Yudi Limpraptono, MT
NIP. Y. 103 950 0274

KONSENTRASI TEKNIK ELEKTRONIKA

JURUSAN TEKNIK ELEKTRO S-1

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2007

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNIK INDUSTRI
JURUSAN TEKNIK ELEKTRO

BERITA ACARA UJIAN SKRIPSI

FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa	: Mahardian Mahendradhata
NIM	: 02.17.019
Jurusan	: Teknik Elektro S1
Konsentrasi	: Teknik Elektronika
Judul Skripsi	: Rancang Bangun Sistem Layanan SPBU Prabayar Menggunakan Teknologi RFID

Dipertahankan dihadapan team penguji Skripsi jenjang Sarjana (S1) pada :

Hari	: Jum'at
Tanggal	: 16 Maret 2007
Dengan Nilai	: A (83,25)

PANITIA UJIAN SKRIPSI

KETUA

Ir. Mochtar Asroni, MSME
NIP.Y 1018100036

SEKRETARIS

Ir Yudi Limpraptono, MT
NIP.Y 1039500274

ANGGOTA PENGUJI

PENGUJI I

Ir Widodo Pudji M, MT
NIP.Y 1028700171

PENGUJI II

DR. Cahyo Chrysdian, Msc
NIP. 1030400412

ABSTRAK

Mahardian Mahendradhata, 2007. *Rancang Bangun Sistem Layanan SPBU Prabayar Menggunakan Teknologi RFID* Skripsi, Konsentrasi Teknik Elektronika, Jurusan Teknik Elektro, Fakultas Teknik Industri, Institut Teknologi Nasional, Malang. Pembimbing: Joseph Dedy Irawan ST.MT.

Kata kunci: RFID (*Radio Frequency Identification*), Mikrokontroller AT 89S52, transaksi, pelanggan, *hardware, software, database* dan *simulasi*.

Meningkatnya kebutuhan masyarakat akan jasa dan fasilitas umum mendorong suatu perusahaan semakin meningkatkan pelayanannya. Salah satunya adalah peningkatan pelayanan pembayaran bagi pelanggan. Perlu kita ketahui bahwa kebanyakan system yang digunakan pada SPBU di sini ini dilakukan secara manual atau konvensional sehingga terkadang pelyanan terhadap konsumen kurang maksimal, apalagi kalau terjadi antrian panjang akan menunda waktu dan juga pasti akan merepotkan. Disamping itu pembayaran dengan menggunakan uang riil pada beberapa transaksi seringkali menyebabkan salah hitung atau kesulitan pengembalian.

Dengan alasan tersebut, Untuk itu perlu merancang bangun suatu perangkat dan pemrograman yang memberi solusi untuk memudahkan pelayanan pada suatu SPBU dengan menggunakan system prabayar yang diidentifikasikan dengan teknologi RFID. Disamping aman teknologi RFID menawarkan kenyamanan dan juga efisiensi bagi pengguna layanan.layanan ini akan dikontrol oleh peralatan yang berbasis mikrokontroller dan metode prabayar yang system penyimpanan datanya menggunakan sebuah *database* yang dijalankan oleh *software Delphi*.

Alat ini dapat dikembangkan dengan melakukan riset nyata pada SPBU atau pada simulasi SPBU agar lebih presisi sebaiknya digunakan sensor mengenai aliran air, pembesaran tendon agar debit air lebih banyak dan juga mekatronika yang baik sebagai penunjang.

KATA PENGANTAR

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

"RANCANG BANGUN SISTEM LAYANAN SPBU PRABAYAR MENGGUNAKAN TEKNOLOGI RFID"

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

Atas terselesaikannya Skripsi ini, penulis mengucapkan terima kasih kepada :

- Bapak Prof. Dr. Ir. Abraham Lomi, MSEE. selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. Mochtar Asroni, MSME selaku Dekan Fakultas Teknologi Industri Institut Teknologi Nasional Malang
- Bapak Ir. F.Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S1 Konsentrasi Teknik Elektronika.
- Bapak Joseph Dedy Irawan, ST, MT selaku Dosen Pembimbing yang telah memberikan bimbingan, pengarahan, serta ilmu-ilmu yang sangat berharga schingga skripsi ini dapat terselesaikan.

- Bapak Ir. Yusuf Ismail Nahkoda, MT. selaku Sekretaris Jurusan Teknik Elektro S1 Konsentrasi Teknik Elektronika.

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

Akhir kata, penulis mohan maaf kepada semua pihak bilamana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua.

Malang, 2007

Penulis

DAFTAR ISI

Halaman

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1 LATAR BELAKANG

Pada era globalisasi keamanan informasi merupakan hal yang sangat penting. Sebuah sistem keamanan informasi harus memperhatikan tiga hal yaitu keamanan, autentifikasi, dan integritas. Untuk mencapai tiga hal tersebut maka dibutuhkan sebuah sistem yang dapat melakukan identifikasi terhadap pengguna yang akan mengakses suatu informasi.

Radio Frequency Identification (RFID) adalah teknologi *wireless* yang ringkas yang berpotensi sangat besar untuk kemajuan perniagaan (*commerce*). RFID menggunakan chip yang dapat dideteksi pada range beberapa meter oleh pembaca RFID. RFID sebagai barcode generasi berikutnya dapat digunakan untuk otomatisasi inventory control.

Dalam beberapa tahun terakhir ini teknologi identifikasi berbasis frekuensi radio (*Radio Frequency Identification*) berkembang dengan pesat. Hal ini diakibatkan oleh beberapa hal, salah satu di antaranya kebutuhan yang besar dari aplikasi untuk konsumen dengan menggunakan teknologi ini.

Bagaimana kalau teknologi tersebut diaplikasikan untuk merancang bangun SPBU dengan menggunakan system prabayar. Perlu kita ketahui bahwa kebanyakan system yang digunakan pada SPBU di negara ini dilakukan secara manual sehingga terkadang pelyanan terhadap konsumen kurang maksimal, apalagi kalau terjadi antrian panjang akan menunda waktu dan juga pasti akan

merepotkan petugas, bahkan keterbatasan karyawan juga akan menghambat efisiensi pelayanan, misalnya ada sebagian SPBU yang tidak memberi layanan sehari 24 jam penuh.

Untuk itu perlu merancang bangun suatu perangkat dan pemrograman yang memberi solusi untuk memudahkan pelayanan pada suatu SPBU dengan menggunakan system prabayar yang diidentifikasikan dengan teknologi RFID. Disamping aman teknologi RFID menawarkan kenyamanan dan juga efisiensi bagi pengguna layanan.

1.2 RUMUSAN MASALAH

Dengan mengacu pada permasalahan di atas, maka skripsi ini dapat dirumuskan permasalahanya sebagai berikut :

1. Bagaimana cara untuk melakukan komuniksi antara Mikrokontroller dengan RFID Reader.
2. Bagaimana membuat program pada mikrokontroller untuk membuat simulator dan menampilkan ke dalam display.
3. Bagaimana membuat Data Base pelanggan.

1.3 TUJUAN

Tujuan penyusunan skripsi ini adalah mencoba merencanakan dan membuat system yang mampu memudahkan memudahkan pelayanan pada suatu SPBU dengan menggunakan system prabayar yang diidentifikasikan dengan teknologi RFID. dengan menggunakan mikrokontroller sebagai pengolah display

dan simulator, PC sebagai penyimpan hasil pengolahah data dengan menggunakan *Data Base*.

Sehubungan dengan hal itu, maka tugas akhir ini diberi judul

RANCANG BANGUN SISTEM LAYANAN SPBU PRABAYAR DENGAN TEKNOLOGI RFID

1.4 BATASAN MASALAH

Dengan mengacu pada permaalahan yang telah dirumuskan, maka hal-hal yang berkaitan dengan alat yang akan dibuat, diberi batasan berikut :

1. Menggunakan system yang berbasiskan mikrokontroler AT89S52.
2. Pembahasan mengenai software *Delphi*.
3. Menggunakan RS 232 sebagai komunikasi serial.
4. Berlaku pada SPBU tertentu.
5. Alat yang dibuat berupa model atau simulasi.
6. Tidak membahas catu daya yang digunakan.

1.5 METODOLOGI

Adapun metodologi yang digunakan dalam perencanaan pembuatan alat ini adalah:

1. **Study Literatur**

   Dengan mencari referensi – referensi yang berhubungan dengan perencanaan dan pembuatan alat terutama referensi yang berhubungan dengan teknologi RFID serta mikrokontroller AT89S52.

2. **Field Research**

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

3. **Perancangan dan Pembuatan alat.**

4. **Penyusunan laporan tugas akhir**

1.6 SISTEMATIKA

Adapun sistematika penulisan yang digunakan dalam perencanaan dan pembuatan alat ini adalah:

BAB I PENDAHULUAN

Berisi latar belakang, tujuan, permasalah, batasan masalah, metodologi, serta sistematika penulisan.

BAB II LANDASAN TEORI

Berisi tentang teori – teori yang menunjang dalam penulisan, perancangan ,dan pembuatan skripsi.

BAB III PERENCANAAN DAN PEMBUATAN ALAT

Berisi tentang perencanaan dan pembuatan alat baik perangkat keras maupun perangkat lunak dan cara kerja blok diagram.

BAB IV PENGUJIAN ALAT

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

BAB V PENUTUP

Berisi kesimpulan dan saran – saran.

BAB II

LANDASAN TEORI

Landasan teori sangat membantu untuk dapat memahami suatu sistem. Selain dari pada itu dapat juga dijadikan sebagai bahan acuan di dalam merencanakan suatu system. Dengan pertimbangan hal-hal tersebut, maka landasan teori merupakan bagian yang harus dipahami untuk pembahasan selanjutnya. Pengetahuan yang mendukung perencanaan dan realisasi alat meliputi;

2.1. Radio Frequency Identification (RFID)

Penggunaan gelombang radio untuk pengiriman data telah banyak digunakan untuk berbagai jenis aplikasi. Salah satu penggunaan gelombang radio ini adalah pada *radio frequency identification* (RFID). RFID ini merupakan sebuah sistem yang mampu mengirimkan identitas secara otomatis dengan menggunakan media gelombang radio. RFID ini sendiri merupakan *enabling technology*, yang berarti teknologi ini tidak dapat berdiri sendiri untuk dapat memberikan manfaat bagi perusahaan, tetapi perusahaan dapat membangun aplikasi yang menggunakan RFID ini untuk mendapatkan manfaat.

Auto-ID dengan sistem Radio Frekuensi Identification (RFID) disebut juga sebagai " *Smart-label*" dapat diprediksi akan menggantikan kedudukan *optical barcode* pada pelabelan barang.

RFID transponder atau *tag* berisi identitas obyek barang. Data ini terdiri dari Merk barang, jenis, model dan nomor serial yang unik, atau data yang diperlukan untuk memberikan abstraksi keterangan barang. *Tag RFID* mempunyai beberapa keuntungan jika dibandingkan dengan sistem *optical barcode*. Data yang ada pada *tag* dapat dibaca secara otomatis

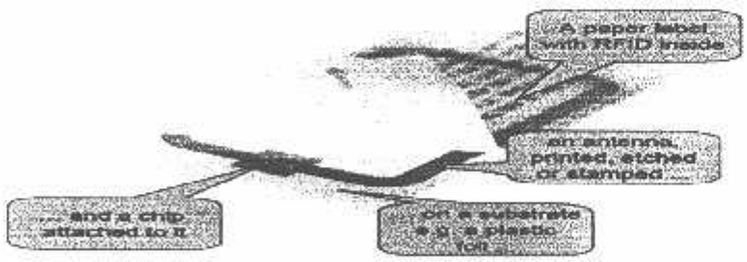
2.1.1. Komponen *RFID Tag*

Radio Frequncy ID terdiri dari tiga komponen sebagai berikut .

- *RFID Tag* atau *tranponder*, yang menampung identifikasi data obyek.
- *RFID tag reader* atau *transciver* yang berfungsi untuk membaca dan menulis data *tag*.
- *Server database* menyimpan kumpulan *record* isi dari *tag*.

2.1.1.1. *Tag*

Tag (kartu/label) secara fisik ditempelkan pada barang. *Tag* tersusun dari *microchip* yang berfungsi untuk menyimpan dan komputasi, yang disatukan dengan lilitan antena yang berfungsi untuk komunikasi. Pada Gambar 2.1 terlihat bagian bagian *tag RFID*.



Gambar 2.1. Bagian bagian *Tag RFID*
Sumber : *RFID – its Applications and Benefits*, Philips, 2004

Menurut klasifikasi *tag* dibedakan menjadi tiga yaitu : aktif, semi-pasif dan pasif.

- *Tag* aktif mempunyai sumber tenaga seperti baterai dan dapat dilakukan komunikasi untuk dibaca dan ditulis.
- *Tag* semi-pasif mempunyai baterai tetapi hanya dapat merespon transmisi yang datang (*incoming transmissions).*
- *Tag* pasif menerima tenaga dari *reader*, antena yang akan menjadi sumber tenaga dengan memanfaatkan medan magnet yang ditimbulkan dari pembaca (*reader*).

Pada Tabel 1. terlihat klasifikasi *tag.*

Tabel 2-1. Klasifikasi *Tag*
sumber : *passive tags www.Atmel RFID*

	Pasif	Semi-pasif	Aktif
Sumber daya	Pasif	Baterai	Baterai
Transmitter	Pasif	Pasif	Aktif
Jangkauan Maksimal	10 meter	100 meter	1000 meter

2.1.1.1.1. *Inductive Coupled RFID Tag dan Capacitive Coupled RFID Tag*

a) *Inductive Coupled RFID tag* terdiri dari dua jenis yaitu yang aktif (yang menggunakan baterai) dan pasif tidak menggunakan baterai, yang tidak menggunakan baterai hanya dapat dibaca saja, sedangkan yang menggunakan baterai dapat dibaca dan ditulis, baik untuk menambah data maupun menghapusnya.

Bahan yang digunakan untuk membuat *inductive coupled RFID tag* terdiri dari bahan bahan sebagai berikut[3] :

- *Silicon Microprocessor*

  Silicon microprocessor adalah sebuah *chip* yang terletak dalam sebuah *tag* yang berfungsi sebagai penyimpan data.

- *Metal Coil*

  Meta coil merupakan komponen yang terbuat dari kawat alumunium yang berfungsi sebagai antena dan beroprasi pada frekuensi 13,56 MHz. Jika sebuah *tag* masuk ke dalam jangkauan *reader* maka antena ini akan mengirimkan data yang ada pada *tag* kepada *reader* terdekat.

- *Encapsulating Material*

  Encapsulating Matrial adalah bahan yang membungkus *tag* yang terbuat dari bahan kaca.

b) *Capacitive Coupled RFID Tag*

Secara struktur *Capasitive Coupled RFID tag* tidak berubah, tetapi bahan dan ukurannya berubah dengan maksud untuk menekan biaya prosuksi. *Capacitive coupled RFID tag* terdiri dari bahan sebagai berikut [3] :

- *Silicon Microprocessor*

  Ukuran dalam *capacitive RFID silicon microprocessor*-nya ($3mm^2$) lebih kecil daripada ukuran *inductive RFID*. *Microprocessor* ini dapat menyimpan data sebesar 96-bit, yang mampu menyimpan jutaan angka sekaligus sebagai kode.

- *Conductive Carbon Ink*

  Conductive Carbon Ink adalah tinta karbon yang memiliki sifat dapat menghantarkan listrik. Tinta inilah yang berfungsi sebagai antena.. Tinta ini dicetakkan di atas sebuah kertas yang melapisi *silicon microprocessor*.

- Kertas

  Kertas inilah yang melapisi *microprocessor* dan tempat tinta dicetakkan. Penggunaan bahan kertas menjadikan label jenis ini mudah dihancurkan bila sudah tidak dipakai.

2.1.1.2. *Tag Reader*

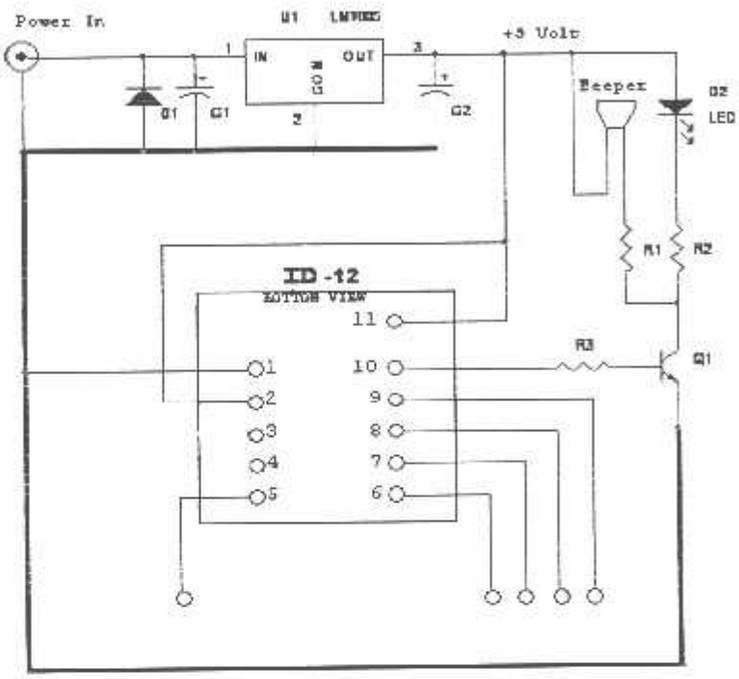
Tag reader berfungsi untuk membaca data yang ada pada *tag* melewati RF *interface*. Untuk menambah fungsi *reader* dilengkapi dengan *internal storage*,

```
      else if length(StrLtr2)=2 then
      StrLtr2:='0'+StrLtr2+'00'
      else if length(StrLtr2)=3 then
      StrLtr2:=StrLtr2+'00';
    end;

    if length(StrRp2)=3 then
    StrRp2:='0000'+StrRp2
    else if length(StrRp2)=4 then
    StrRp2:='000'+StrRp2
    else if length(StrRp2)=5 then
    StrRp2:='00'+StrRp2
    else if length(StrRp2)=6 then
    StrRp2:='0'+StrRp2 ;

    if StrToInt(lstr1)<=StrToInt(StrRp1) then
    begin
      strkirimdata:='y'+StrRp2+'r'+StrLtr2+'l';
      Edit4.Text:=Edit4.Text+' '+strkirimdata;

      for  lInti:=1  to length(strkirimdata)  do
      begin
        kirimdata(strkirimdata[lInti]);
      end;
      Edit1.Text:='';
      Intj4:=1;
      Intj3:=0;

    end
    else
    begin
      strkirimdata:='t';
      Edit4.Text:=Edit4.Text+' '+strkirimdata;
      kirimdata(strkirimdata);
      FormCreate(Sender);
      Edit1.Text:='';

    end;

  end;
end;

if Intj2 = 0 then
begin
ADOQuerytblSPBU.Close;
ADOQuerytblSPBU.SQL.Clear;
ADOQuerytblSPBU.SQL.Add('select * from tblSPBU where norfid='+QuotedStr(StrRfid));
ADOQuerytblSPBU.Open;


if
(ADOQuerytblSPBU.FieldByName('norfid').AsString=StrRfid)and(ADOQuerytblSPBU.FieldByN
ame('paswd').AsString=StrPasswd)and(StrRfid<>'') then
begin

StrRp1:=IntToStr(ADOQuerytblSPBU.FieldByName('account').AsInteger);
  StrLtr1:=FloatToStr(RpToLtr(StrToInt(StrRp1)));

Edit2.Text:=StrRp1;
Edit3.Text:=StrLtr1;
      strkirimdata:='b'+StrRp1+'r'+StrLtr1+'l';
      Edit4.Text:=strkirimdata;
```

```
      for  lInti:=1  to length(strkirimdata)  do
       begin
                  kirimdata(strkirimdata[lInti]);
       end;


     Edit1.Text:='';
    Intj3:=1;
    Intj2:=1;
end
else if
(ADOQuerytblSPBU.FieldByName('norfid').AsString=StrRfid)and(ADOQuerytblSPBU.FieldByN
ame('paswd').AsString<>StrPasswd)and(StrRfid<>'') then
begin
kirimdata('s');
Edit4.Text:=Edit4.Text+'s';
FormCreate(Sender);
Edit1.Text:='';
//Intj1:=0;
end
else if
(ADOQuerytblSPBU.FieldByName('norfid').AsString<>StrRfid)and(ADOQuerytblSPBU.FieldBy
Name('paswd').AsString<>StrPasswd)and(StrRfid<>'') then
begin
kirimdata('s');
Edit4.Text:=Edit4.Text+'s';
FormCreate(Sender);
Edit1.Text:='';
//Intj1:=0;
end;

end;


end;

end.
```

Advanced Digital Reader Technology
----Better by Design

Circuit Diagram for the ID-12



COMPONENT LIST
R1 = 100R
R2 = 1K
R3 = 1K
C1 = 100uF 16V
C2 = 100uF 10V
Beeper = 2.7-3.5KHz 100R
D1 = 1N4001
D2 = GREEN LED
U1 = LM7805
Q1 = UTC8050 (NPN)
ID2 = ID Innovations ID2

* Please Note the ID2 has an internal tuning capacitor of 1.5nF and this makes the total tuning capacity = 2.5nF

The 3.1Khz Beeper Logic is centered for most Beepers in range 2.7-3.5Khz

DATA FORMATS

Output Data Structure – ASCII

STX (02h)	DATA (10 ASCII)	CHECK SUM (2 ASCII)	CR	LF	ETX (03h)

[The 1byte (2 ASCII characters) Check sum is the "Exclusive OR" of the 5 hex bytes (10 ASCII) Data characters.]

Output Data Structure – Wiegand26

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26
P	E	E	E	E	E	E	E	E	E	E	E	E	O	O	O	O	O	O	O	O	O	O	O	O	P
Even parity (E)													Odd parity (O)												

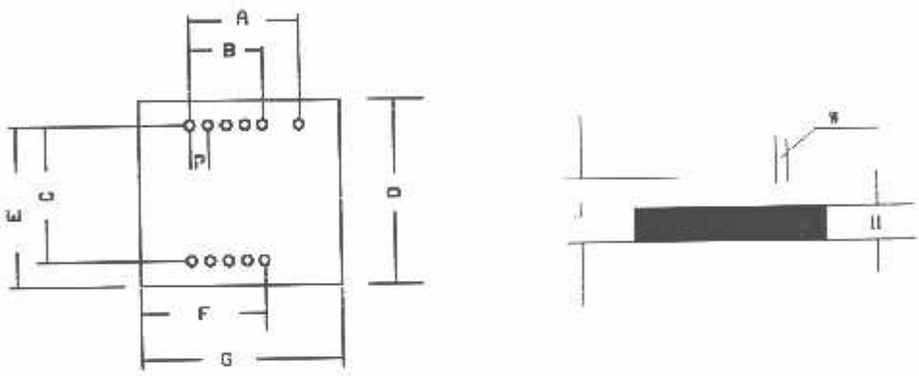
P = Parity start bit and stop bit

Output Data Magnetic ABA Track2

10 Leading Zeros	SS	Data	ES	LCR	10 Ending Zeros

[SS is the Start Character of 11010, ES is the end character of 11111. LRC is the Longitudinal Redundancy Check.]

Dimensions (Top View) (mm)



	ID-0/ID-2			ID-10/ID-12			ID-15/ID-20		
	Nom.	Min.	Max.	Nom.	Min.	Max.	Nom.	Min.	Max
A	12.0	11.6	12.4	12.0	11.6	12.4	12.0	11.6	12.4
B	8.0	7.6	8.4	8.0	7.6	8.4	8.0	7.6	8.4
C	15.0	14.6	15.4	15.0	14.6	15.4	15.0	14.6	15.4
D	20.5	20.0	21.5	25.3	24.9	25.9	40.3	40.0	41.0
E	18.5	18.0	19.2	20.3	19.8	20.9	27.8	27.5	28.5
F	14.0	13.0	14.8	16.3	15.8	16.9	22.2	21.9	23.1
G	22.0	21.6	22.4	26.4	26.1	27.1	38.5	38.2	39.2
P	2.0	1.8	2.2	2.0	1.8	2.2	2.0	1.8	2.2
H	5.92	5.85	6.6	6.0	5.8	6.6	6.8	6.7	7.0
J	9.85	9.0	10.5	9.9	9.40	10.5	9.85	9.4	10.6
W	0.66	0.62	0.67	0.66	0.62	0.67	0.66	0.62	0.67

Note – measurements do not include any burring of edges.

NOTICE - Innovated Devices reserve the right to change these specifications without prior notice.

Advanced Digital Reader Technology

----Better by Design

eatures

- Compatible with MCS®-51 Products
- 8K 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
- 256 x 8-bit Internal RAM
- 32 Programmable I/O Lines
- Three 16-bit Timer/Counters
- Eight 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

ıe AT89S52 is a low-power, high-performance CMOS 8-bit microcontroller with 8K /tes of in-system programmable Flash memory. The device is manufactured using ımel's high-density nonvolatile memory technology and is compatible with the indus-y-standard 80C51 instruction set and pinout. The on-chip Flash allows the program emory to be reprogrammed in-system or by a conventional nonvolatile memory pro-ammer. By combining a versatile 8-bit CPU with in-system programmable Flash on monolithic chip, the Atmel AT89S52 is a powerful microcontroller which provides a ghly-flexible and cost-effective solution to many embedded control applications.

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

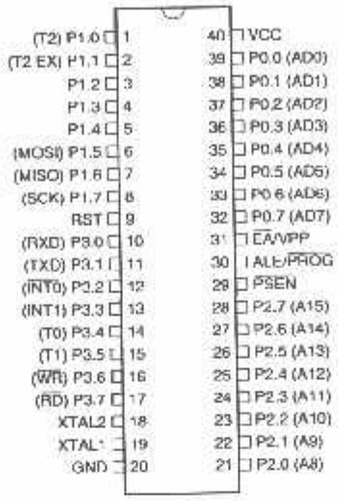
8-bit Microcontroller with 8K Bytes In-System Programmable Flash

AT89S52

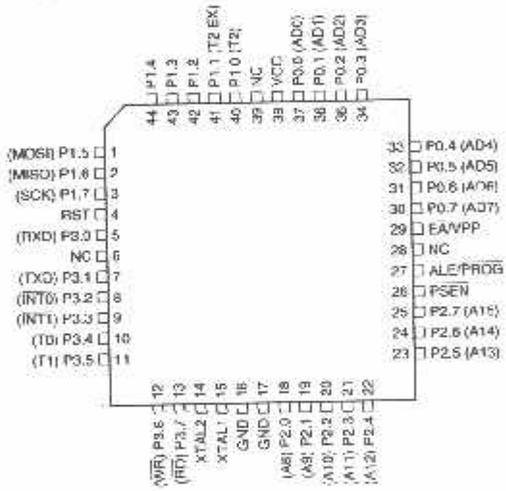
1919C–MICRO 3/05

. Pin Configurations

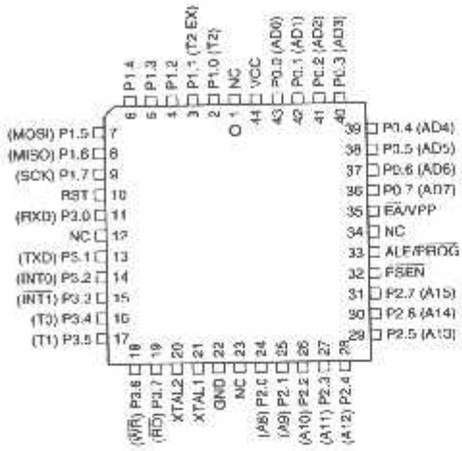
.1 40-lead PDIP



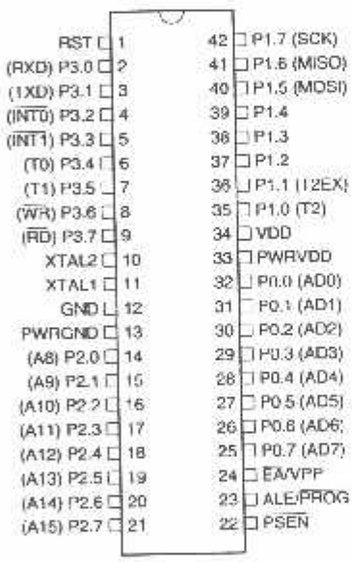
!.2 44-lead TQFP



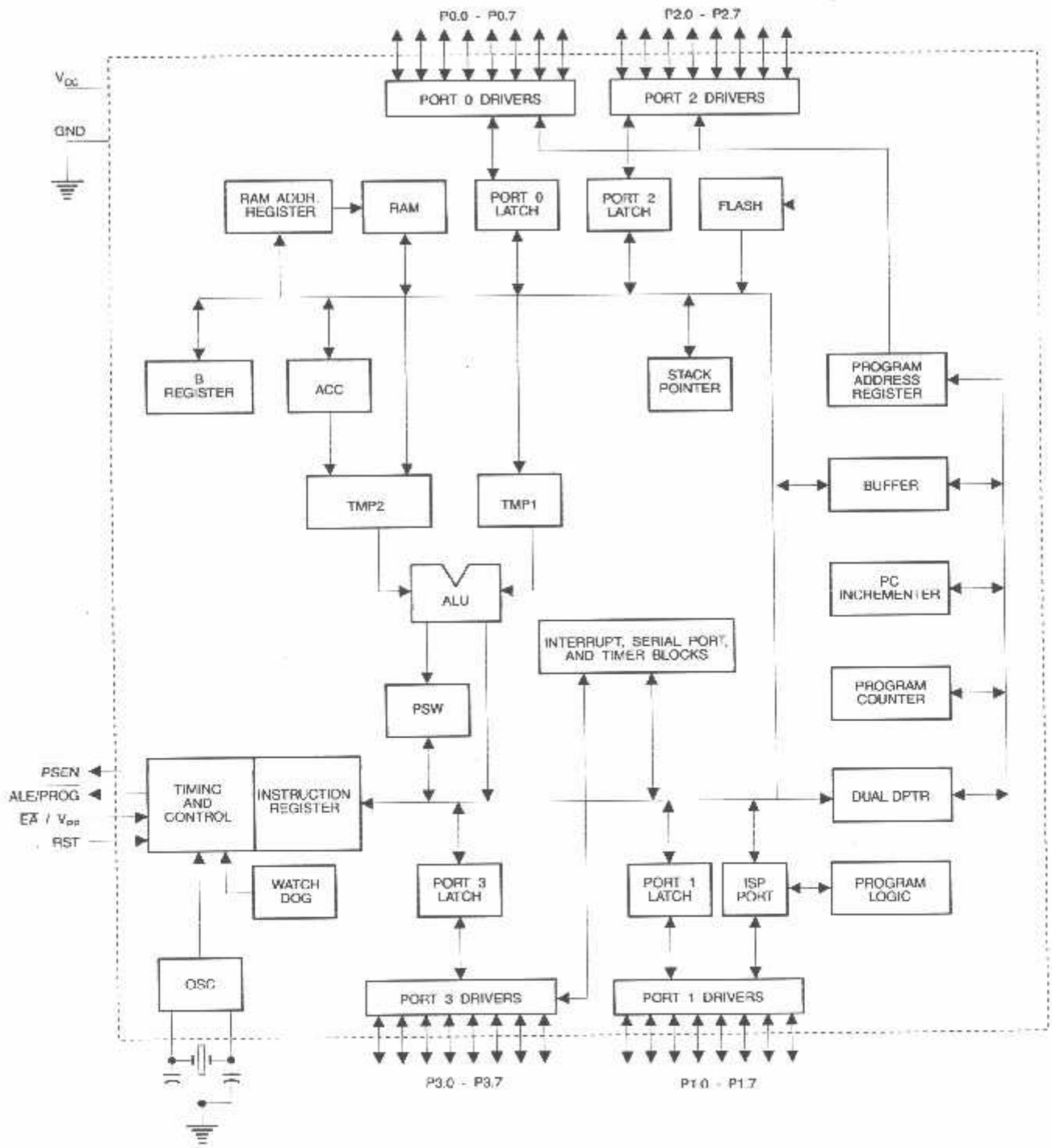
2.3 44-lead PLCC



2.4 42-lead PDIP



Block Diagram



. Pin Description

.1 VCC

Supply voltage.

.2 GND

Ground.

.3 Port 0

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

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

Port 0 also receives the code bytes during Flash programming and outputs the code bytes during program verification. **External pull-ups are required during program verification.**

.4 Port 1

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

In addition, P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively, as shown in the following table.

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

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.5	MOSI (used for In-System Programming)
P1.6	MISO (used for In-System Programming)
P1.7	SCK (used for In-System Programming)

4.5 Port 2

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

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

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

6 Port 3

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

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

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

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

.7 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.

.8 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.

9 PSEN

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

When the AT89S52 is executing code from external program memory, $\overline{PSEN}$ is activated twice each machine cycle, except that two $\overline{PSEN}$ activations are skipped during each access to external data memory.

10 EA/VPP

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

$\overline{EA}$ should be strapped to V_{CC} for internal program executions.

This pin also receives the 12-volt programming enable voltage (V_{PP}) during Flash programming.

11 XTAL1

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

12 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.

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

Timer 2 Registers: Control and status bits are contained in registers T2CON (shown in Table 5-2) and T2MOD (shown in Table 10-2) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture/Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

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

Table 5-1. AT89S52 SFR Map and Reset Values

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

able 5-2. T2CON – Timer/Counter 2 Control Register

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

Bit Addressable

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

Symbol	Function
TF2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
EXF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
RCLK	Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflow to be used for the receive clock.
TCLK	Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
EXEN2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
TR2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
C/$\overline{T2}$	Timer or counter select for Timer 2. C/$\overline{T2}$ = 0 for timer function. C/$\overline{T2}$ = 1 for external event counter (falling edge triggered).
CP/$\overline{RL2}$	Capture/Reload select. CP/$\overline{RL2}$ = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1. CP/$\overline{RL2}$ = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

able 5-3. 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

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

ower 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 . It can be set and rest under software control and is not affected by reset.

able 5-4. 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 AT89S52, if $\overline{EA}$ is connected to V_{CC}, program fetches to addresses 0000H through 1FFFH are directed to internal memory and fetches to addresses 2000H through FFFFH are to external memory.

2 Data Memory

The AT89S52 implements 256 bytes of on-chip RAM. The upper 128 bytes occupy a parallel address space to the Special Function Registers. This means that the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

When an instruction accesses an internal location above address 7FH, the address mode used in the instruction specifies whether the CPU accesses the upper 128 bytes of RAM or the SFR space. Instructions which use direct addressing access the SFR space.

For example, the following direct addressing instruction accesses the SFR at location 0A0H (which is P2).

```
MOV 0A0H, #data
```

Instructions that use indirect addressing access the upper 128 bytes of RAM. For example, the following indirect addressing instruction, where R0 contains 0A0H, accesses the data byte at address 0A0H, rather than P2 (whose address is 0A0H).

```
MOV @R0, #data
```

Note that stack operations are examples of indirect addressing, so the upper 128 bytes of data RAM are available as stack space.

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.

1 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 AT89S52 is reset. Exiting Power-down with an interrupt is significantly different. The interrupt is held low long enough for the oscillator to stabilize. When the interrupt is brought high, the interrupt is serviced. To prevent the WDT from resetting the device while the interrupt pin is held low, the WDT is not started until the interrupt is pulled high. It is suggested that the WDT be reset during the interrupt service for the interrupt used to exit Power-down mode.

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

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

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

. UART

The UART in the AT89S52 operates the same way as the UART in the AT89C51 and AT89C52. For further information on the UART operation, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC4316.PDF

. Timer 0 and 1

Timer 0 and Timer 1 in the AT89S52 operate the same way as Timer 0 and Timer 1 in the AT89C51 and AT89C52. For further information on the timers' operation, please click on the document link below:

http://www.atmel.com/dyn/resources/prod_documents/DOC4316.PDF

0. Timer 2

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/$\overline{T2}$ in the SFR T2CON (shown in Table 5-2). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 10-1. Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

Table 10-1. Timer 2 Operating Modes

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

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

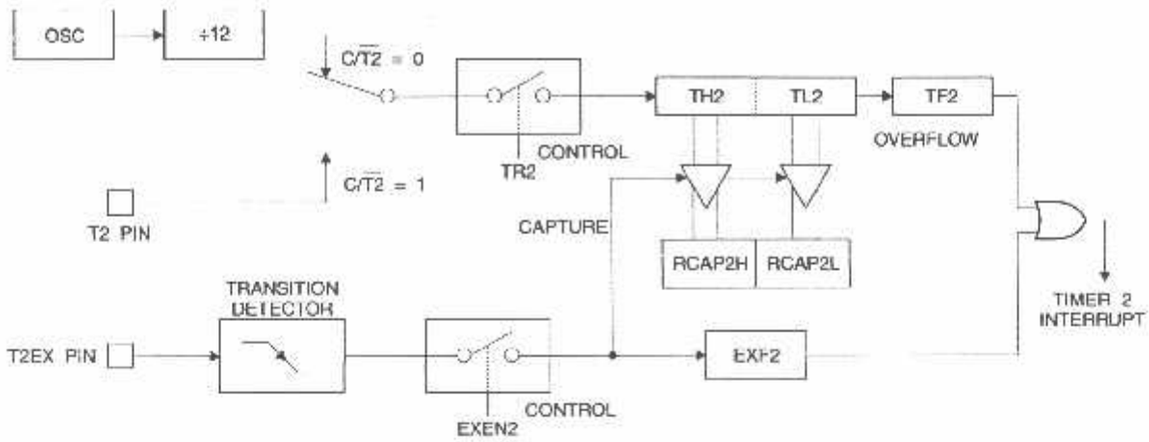
0.1 Capture Mode

In the capture mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 is a 16-bit timer or counter which upon overflow sets bit TF2 in T2CON. This bit can then be used to generate an interrupt. If EXEN2 = 1, Timer 2 performs the same operation, but a 1-to-0 transition at external input T2EX also causes the current value in TH2 and TL2 to be captured into RCAP2H and RCAP2L, respectively. In addition, the transition at T2EX causes bit EXF2 in T2CON to be set. The EXF2 bit, like TF2, can generate an interrupt. The capture mode is illustrated in Figure 10-1.

0.2 Auto-reload (Up or Down Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 10-2). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.

igure 10-1. Timer in Capture Mode



able 10-2. T2MOD – Timer 2 Mode Control Register

T2MOD Address = 0C9H — Reset Value = XXXX XX00B

Not Bit Addressable

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

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

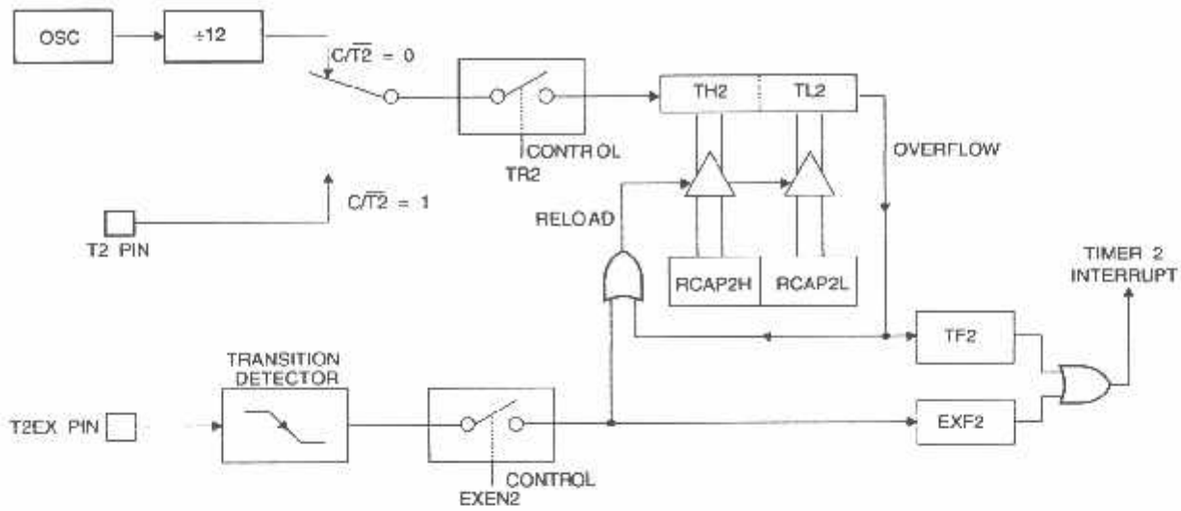
Figure 10-2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H and RCAP2L. The values in Timer in Capture ModeRCAP2H and RCAP2L are preset by software. If EXEN2 = 1, a 16-bit reload can be triggered either by an overflow or by a 1-to-0 transition at external input T2EX. This transition also sets the EXF2 bit. Both the TF2 and EXF2 bits can generate an interrupt if enabled.

Setting the DCEN bit enables Timer 2 to count up or down, as shown in Figure 10-2. In this mode, the T2EX pin controls the direction of the count. A logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit. This overflow also causes the 16-bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively.

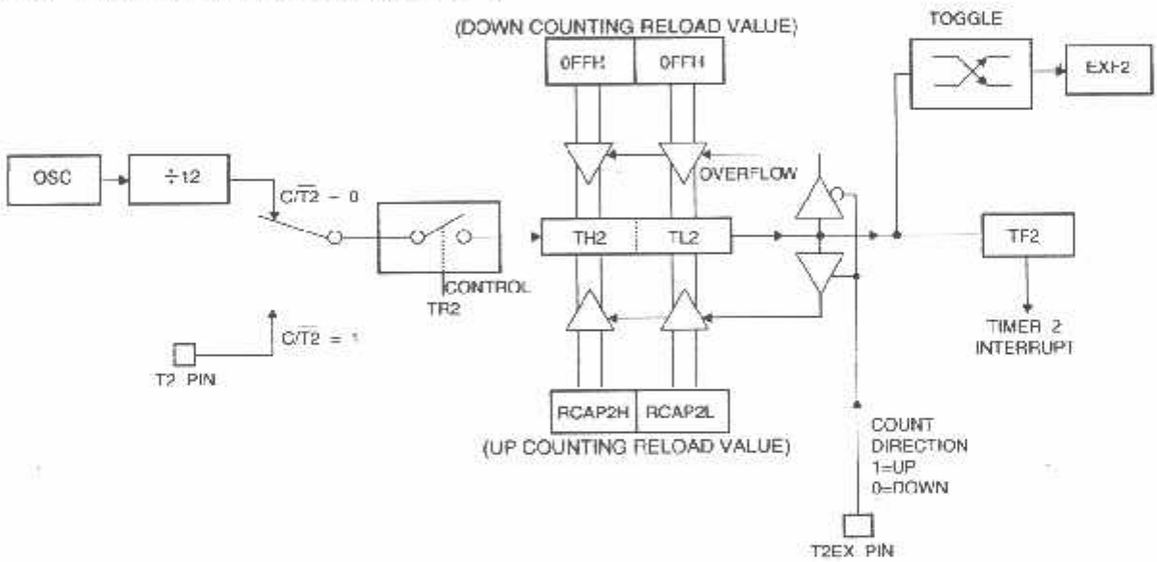
A logic 0 at T2EX makes Timer 2 count down. The timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers.

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

igure 10-2. Timer 2 Auto Reload Mode (DCEN = 0)



igure 10-3. Timer 2 Auto Reload Mode (DCEN = 1)



1. Baud Rate Generator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 5-2). Note that the baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 11-1.

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

The baud rates in Modes 1 and 3 are determined by Timer 2's overflow rate according to the following equation.

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

The Timer can be configured for either timer or counter operation. In most applications, it is configured for timer operation ($C/\overline{T2} = 0$). The timer operation is different for Timer 2 when it is used as a baud rate generator. Normally, as a timer, it increments every machine cycle (at 1/12 the oscillator frequency). As a baud rate generator, however, it increments every state time (at 1/2 the oscillator frequency). The baud rate formula is given below.

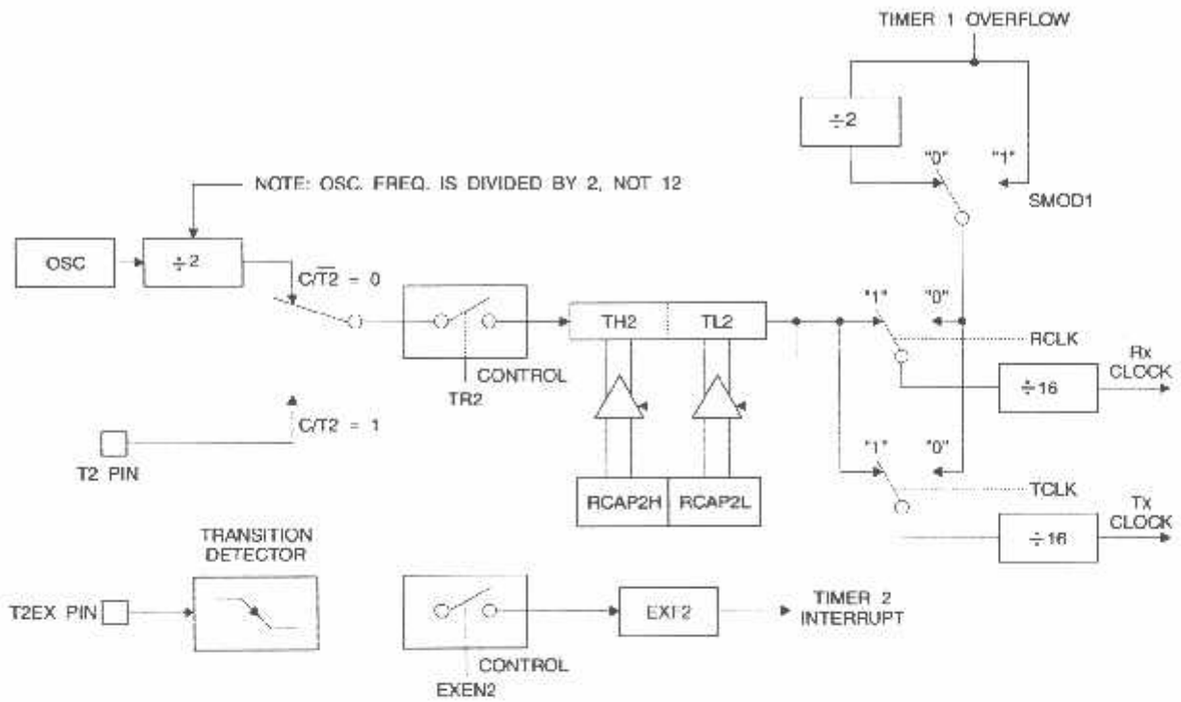
$$\frac{\text{Modes 1 and 3}}{\text{Baud Rate}} = \frac{\text{Oscillator Frequency}}{32 \times [65536\text{-RCAP2H,RCAP2L})]}$$

where (RCAP2H, RCAP2L) is the content of RCAP2H and RCAP2L taken as a 16-bit unsigned integer.

Timer 2 as a baud rate generator is shown in Figure 11-1. This figure is valid only if RCLK or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2 and will not generate an interrupt. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus, when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt.

Note that when Timer 2 is running (TR2 = 1) as a timer in the baud rate generator mode, TH2 or TL2 should not be read from or written to. Under these conditions, the Timer is incremented every state time, and the results of a read or write may not be accurate. The RCAP2 registers may be read but should not be written to, because a write might overlap a reload and cause write and/or reload errors. The timer should be turned off (clear TR2) before accessing the Timer 2 or RCAP2 registers.

gure 11-1. Timer 2 In Baud Rate Generator Mode



2. Programmable Clock Out

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 12-1. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz (for a 16-MHz operating frequency).

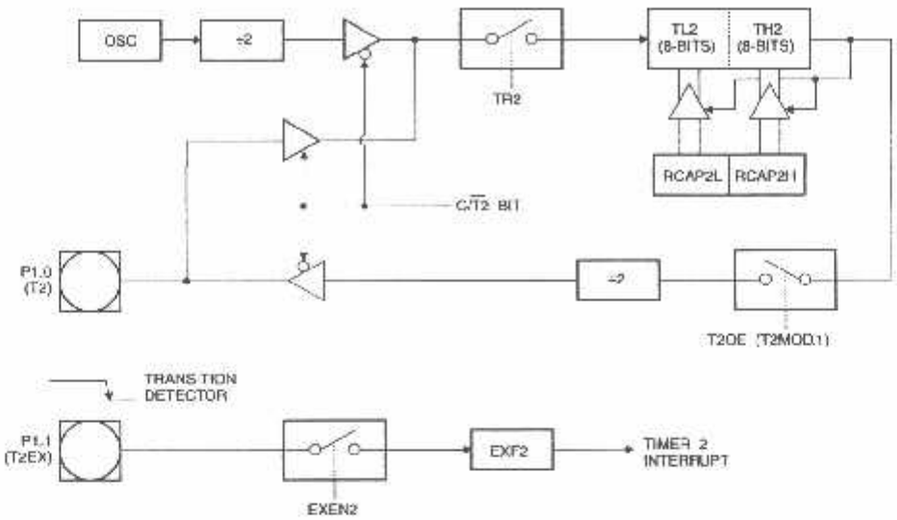
To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

The clock-out frequency depends on the oscillator frequency and the reload value of Timer 2 capture registers (RCAP2H, RCAP2L), as shown in the following equation.

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

In the clock-out mode, Timer 2 roll-overs will not generate an interrupt. This behavior is similar to when Timer 2 is used as a baud-rate generator. It is possible to use Timer 2 as a baud-rate generator and a clock generator simultaneously. Note, however, that the baud-rate and clock-out frequencies cannot be determined independently from one another since they both use RCAP2H and RCAP2L.

gure 12-1. Timer 2 in Clock-Out Mode



3. Interrupts

The AT89S52 has a total of six interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 13-1.

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 13-1 shows that bit position IE.6 is unimplemented. User software should not write a 1 to this bit position, since it may be used in future AT89 products.

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

The Timer 0 and Timer 1 flags, TF0 and TF1, are set at S5P2 of the cycle in which the timers overflow. The values are then polled by the circuitry in the next cycle. However, the Timer 2 flag, TF2, is set at S2P2 and is polled in the same cycle in which the timer overflows.

ible 13-1. Interrupt Enable (IE) Register

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

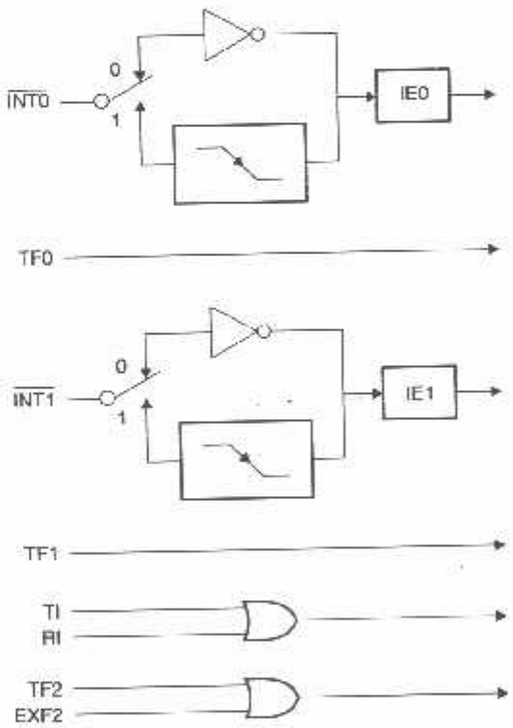
Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
-	IE.6	Reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	Serial Port interrupt enable bit.
ET1	IE.3	Timer 1 interrupt enable bit.
EX1	IE.2	External interrupt 1 enable bit.
ET0	IE.1	Timer 0 interrupt enable bit.
EX0	IE.0	External interrupt 0 enable bit.

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

gure 13-1. Interrupt Sources



4. 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 16-1. 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 16-2. 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.

5. Idle Mode

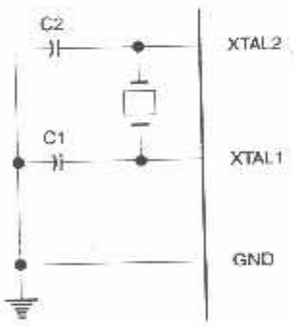
In idle mode, the CPU puts itself to sleep while all the on-chip peripherals remain active. The mode is invoked by software. The content of the on-chip RAM and all the special functions registers remain unchanged during this mode. The idle mode can be terminated by any enabled interrupt or by a hardware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

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

6. 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 an enabled external interrupt. Reset redefines the SFRs but does not change the on-chip RAM. The reset should not be activated before V_{CC} is restored to its normal operating level and must be held active long enough to allow the oscillator to restart and stabilize.

Figure 16-1. Oscillator Connections



Note: 1. C1, C2 = 30 pF ± 10 pF for Crystals
= 40 pF ± 10 pF for Ceramic Resonators

Figure 16-2. External Clock Drive Configuration

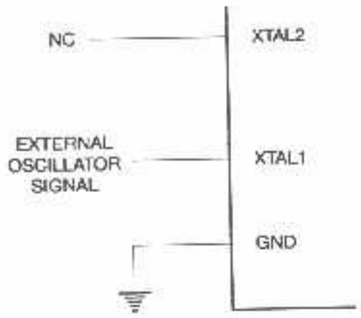


Table 16-1. Status of External Pins During Idle and Power-down Modes

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

7. Program Memory Lock Bits

The AT89S52 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in Table 17-1.

Table 17-1. 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.

8. Programming the Flash – Parallel Mode

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

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

Programming Algorithm: Before programming the AT89S52, the address, data, and control signals should be set up according to the "Flash Programming Modes" (Table 22-1) and Figure 22-1 and Figure 22-2. To program the AT89S52, take the following steps:

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

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

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

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

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

(000H) = 1EH indicates manufactured by Atmel
(100H) = 52H indicates AT89S52
(200H) = 06H

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

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

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

9. 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.

0. Serial Programming Algorithm

To program and verify the AT89S52 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 which returns the content at the selected address at serial output MISO/P1.6.
5. At the end of a programming session, RST can be set low to commence normal device operation.

Power-off sequence (if needed):

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.

1. Serial Programming Instruction Set

The Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 24-1.

2. 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 support for the Atmel AT89 microcontroller series. Please contact your local programming vendor for the appropriate software revision.

ıble 22-1. Flash Programming Modes

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

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

Figure 22-1. Programming the Flash Memory (Parallel Mode)

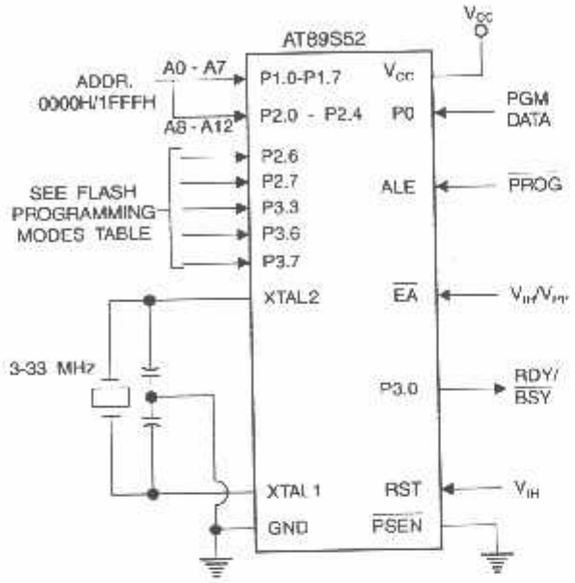
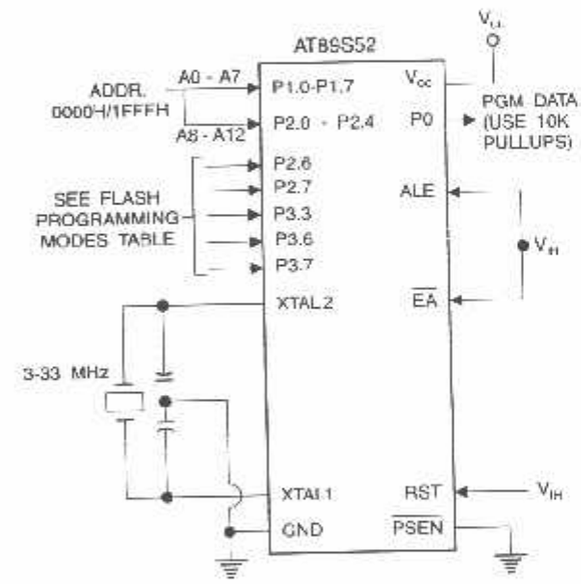


Figure 22-2. Verifying the Flash Memory (Parallel Mode)



3. Flash Programming and Verification Characteristics (Parallel Mode)

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

Symbol	Parameter	Min	Max	Units
I_{PP}	Programming Supply Voltage	11.5	12.5	V
PP	Programming Supply Current		10	mA
CC	V_{CC} Supply Current		30	mA
$/t_{CLCL}$	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		48 t_{CLCL}	
ELQV	$\overline{ENABLE}$ Low to Data Valid		48 t_{CLCL}	
EHQZ	Data Float After $\overline{ENABLE}$	0	48 t_{CLCL}	
GHBL	$\overline{PROG}$ High to $\overline{BUSY}$ Low		1.0	µs
WC	Byte Write Cycle Time		50	µs

Figure 23-1. Flash Programming and Verification Waveforms – Parallel Mode

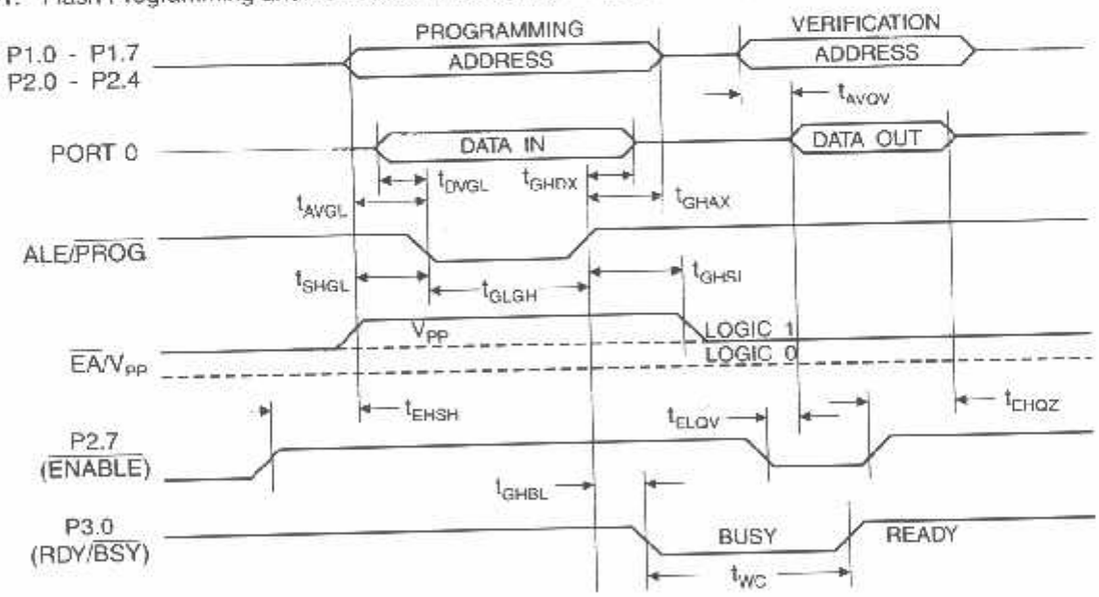
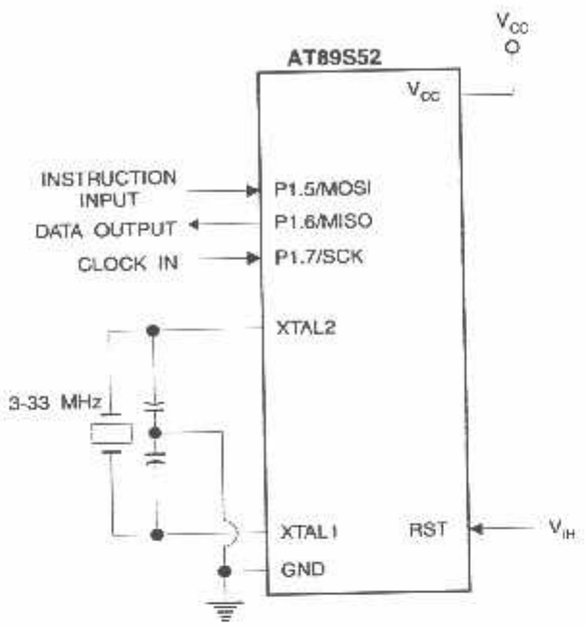
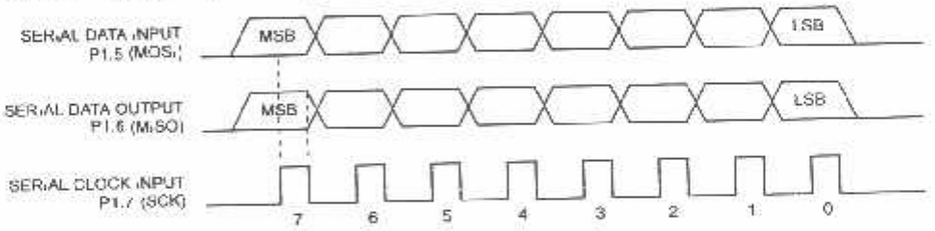


Figure 23-2. Flash Memory Serial Downloading



4. Flash Programming and Verification Waveforms – Serial Mode

Figure 24-1. Serial Programming Waveforms



ble 24-1. Serial Programming Instruction Set

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

Note: 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

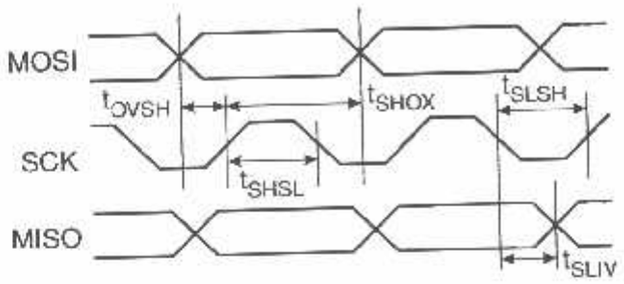
Programming the Lock bits requires a 12V programming voltage. Once programmed, the Lock bits can only be erased using the Chip Erase operation.

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

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

5. Serial Programming Characteristics

gure 25-1. Serial Programming Timing



Serial Programming Characteristics, T_A = -10°C to +85°C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
	Oscillator Frequency	3		33	MHz
t_{CLCL}	Oscillator Period	30			ns
t_{SHSL}	SCK Pulse Width High	$8\ t_{CLCL}$			ns
t_{SLSH}	SCK Pulse Width Low	$8\ t_{CLCL}$			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	$2\ t_{CLCL}$			ns
t_{SLIV}	SCK Low to MISO Valid	10	16	32	ns
t_{ERASE}	Chip Erase Instruction Cycle Time			500	ms
t_{SWC}	Serial Byte Write Cycle Time			$64\ t_{CLCL} + 400$	µs

6. Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on Any Pin with Respect to Ground	-1.0V to +7.0V
Maximum Operating Voltage	6.6V
DC Output Current	15.0 mA

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

7. DC Characteristics

The 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.

Symbol	Parameter	Condition	Min	Max	Units
V_{IL}	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
V_{IL1}	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
V_{IH}	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
V_{IH1}	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
V_{OL}	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
V_{OL1}	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
V_{OH}	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
V_{OH1}	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
I_{IL}	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ± 10%		-300	µA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
R_{RST}	Reset Pulldown Resistor		50	300	KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
I_{CC}	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

Notes: 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.

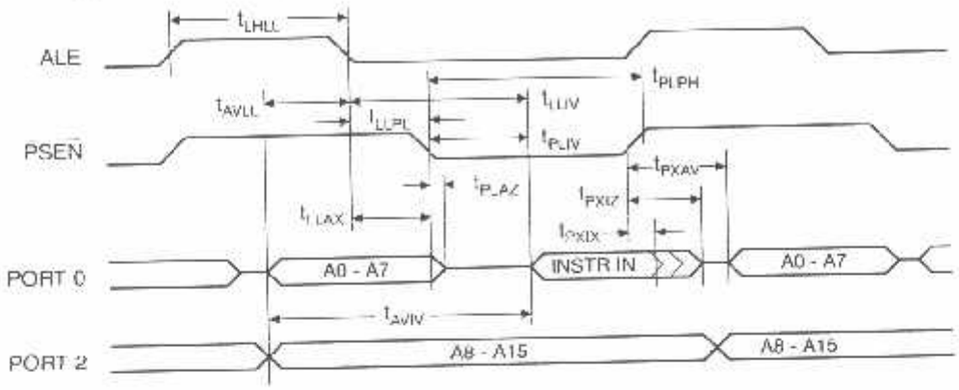
3. AC Characteristics

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

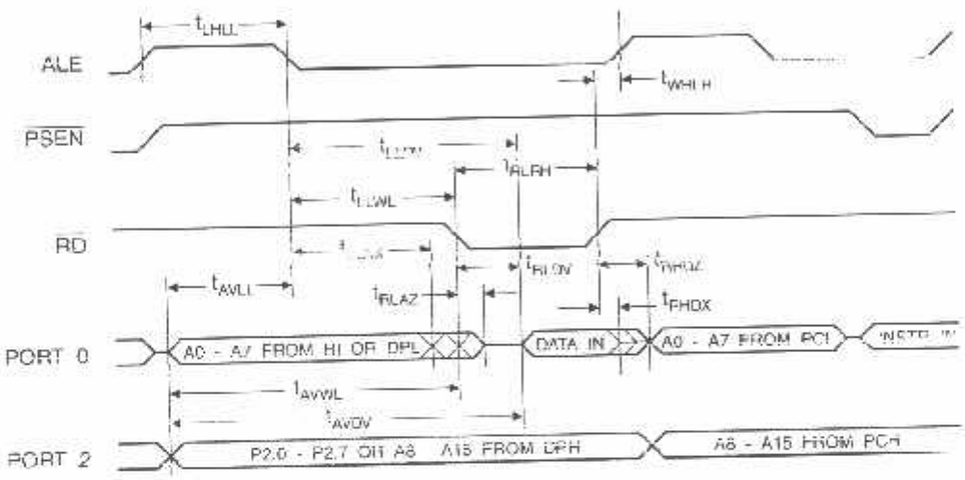
3.1 External Program and Data Memory Characteristics

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

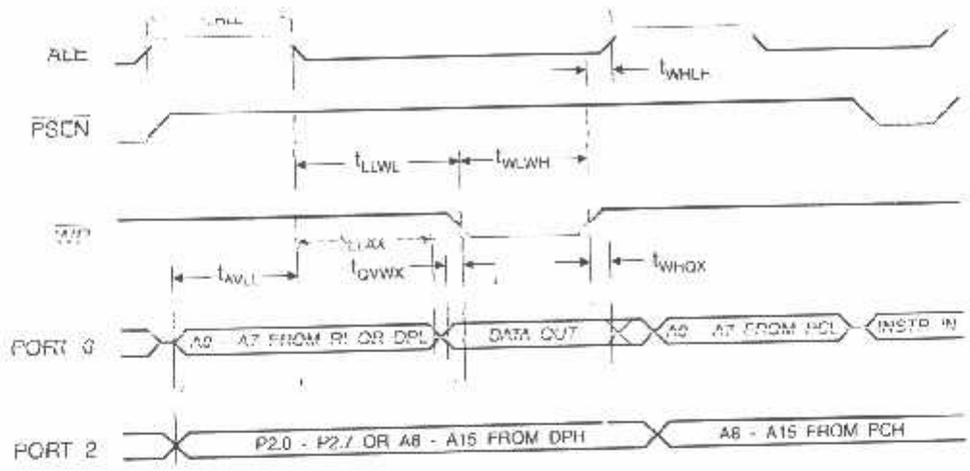
9. External Program Memory Read Cycle



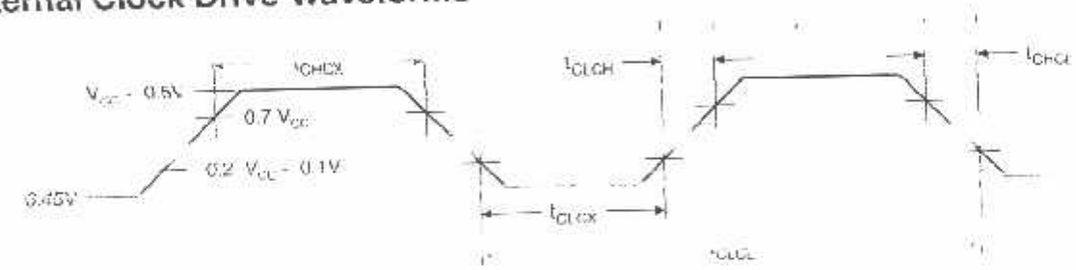
0. External Data Memory Read Cycle



1. External Data Memory Write Cycle



2. External Clock Drive Waveforms



3. External Clock Drive

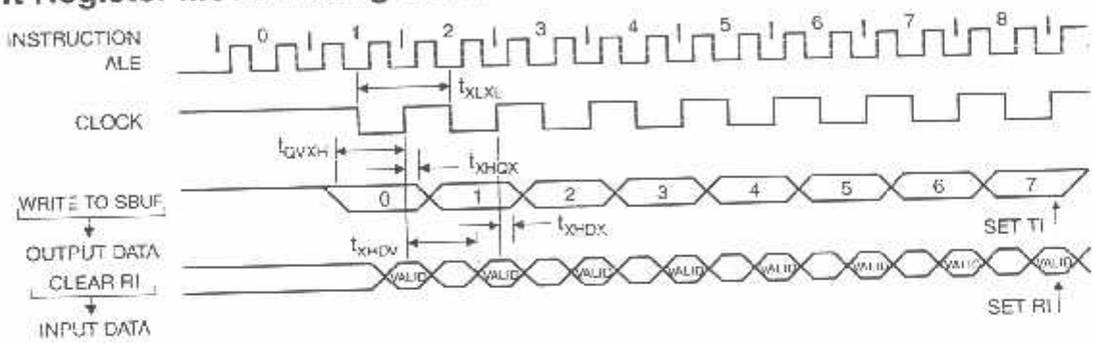
Symbol	Parameter	Min	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0	33	MHz
t_{CLCL}	Clock Period	30		ns
t_{CHCX}	High Time	12		ns
t_{CLCX}	Low Time	12		ns
t_{CLCH}	Rise Time		5	ns
t_{CHCL}	Fall Time		5	ns

4. Serial Port Timing: Shift Register Mode Test Conditions

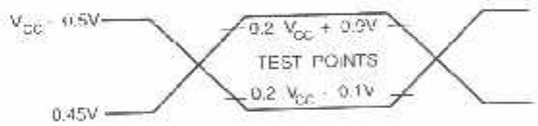
ıe values in this table are valid for V_{CC} = 4.0V to 5.5V and Load C...

Symbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
XLXL	Serial Port Clock Cycle Time	1.0		12 t_{CLCL}		µs
QVXH	Output Data Setup to Clock Rising Edge	700		10 t_{CLCL}-133		ns
XHQX	Output Data Hold After Clock Rising Edge	50		2 t_{CLCL}-80		ns
XHDX	Input Data Hold After Clock Rising Edge	0		0		ns
XHDV	Clock Rising Edge to Input Data Valid		700		10 t_{CLCL}-133	ns

5. Shift Register Mode Timing Waveforms

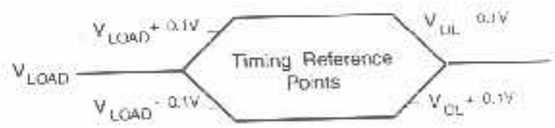


6. AC Testing Input/Output Waveforms[1]



ıte: 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.

7. Float Waveforms[1]



ıte: 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.

8. Ordering Information

8.1 Standard Package

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S52-24AC	44A	Commercial (0°C to 70°C)
		AT89S52-24JC	44J	
		AT89S52-24PC	40P6	
		AT89S52-24SC	42PS6	
		AT89S52-24AI	44A	Industrial (-40°C to 85°C)
		AT89S52-24JI	44J	
		AT89S52-24PI	40P6	
		AT89S52-24SI	42PS6	
33	4.5V to 5.5V	AT89S52-33AC	44A	Commercial (0°C to 70°C)
		AT89S52-33JC	44J	
		AT89S52-33PC	40P6	
		AT89S52-33SC	42PS6	

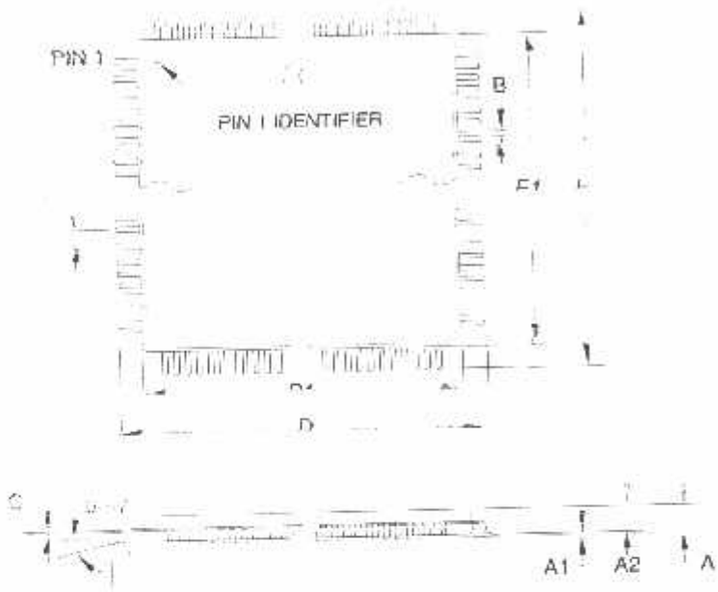
8.2 Green Package Option (Pb/Halide-free)

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S52-24AU	44A	Industrial (-40°C to 85°C)
		AT89S52-24JU	44J	
		AT89S52-24PU	40P6	

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

9. Packaging Information

9.1 44A – TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

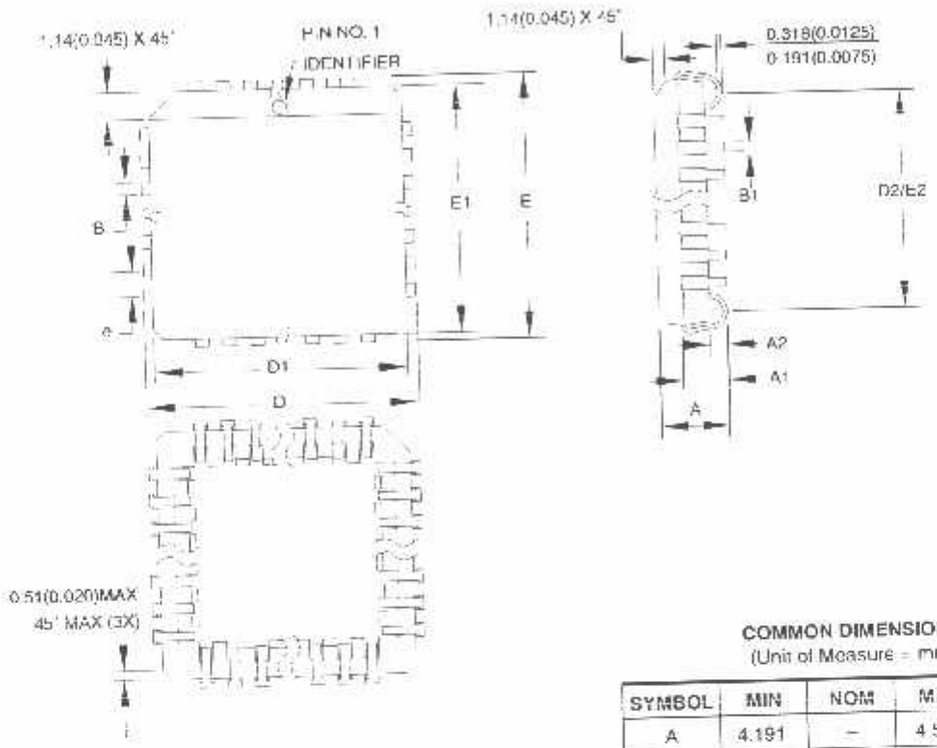
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e		0.80 TYP		

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)

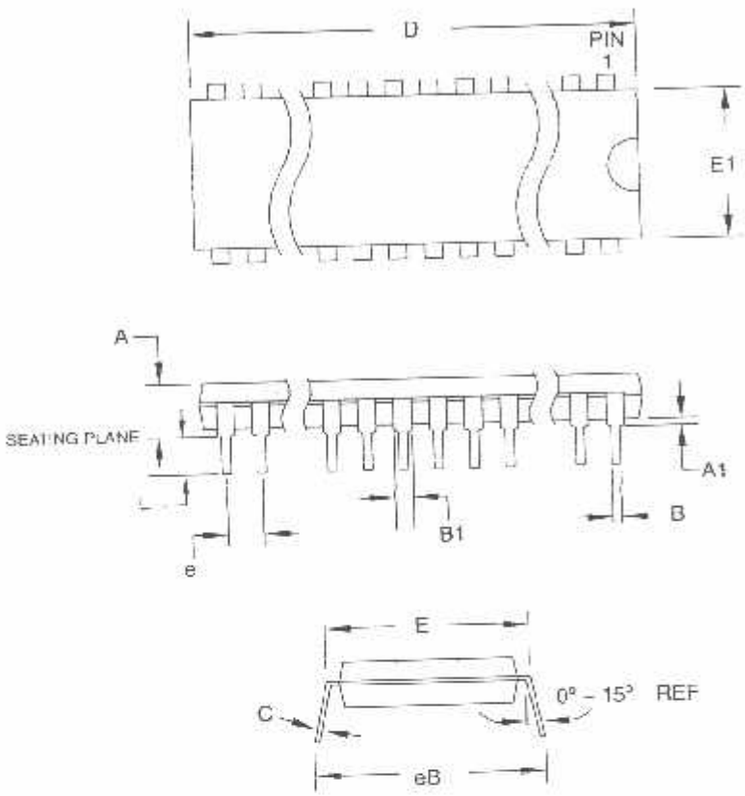
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.333	–	0.533	
e	1.270 TYP			

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

1.3 40P6 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

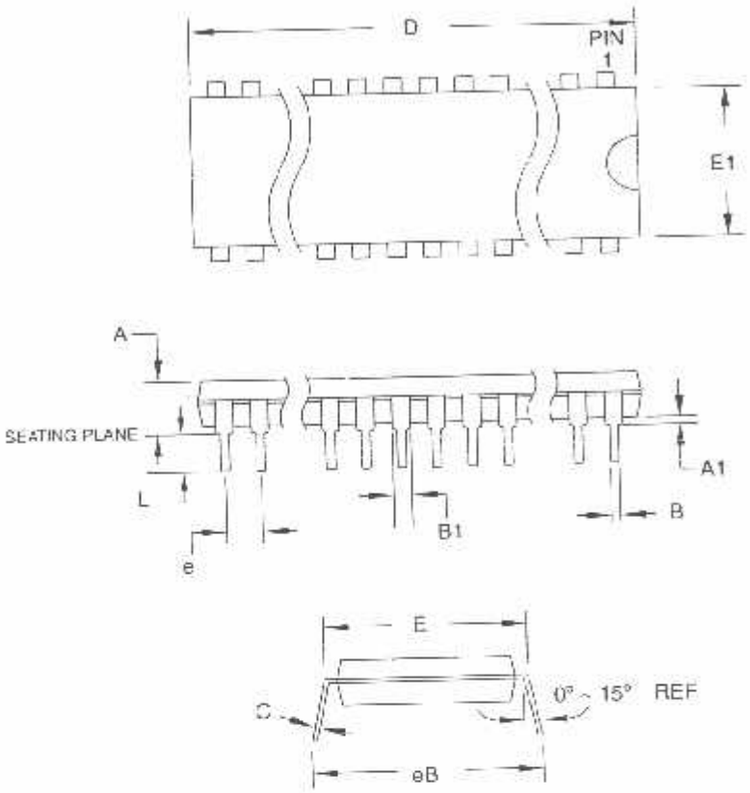
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

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

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

1.4 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

tmel Corporation

2325 Orchard Parkway
San Jose, CA 95131, USA
Tel: 1(408) 441-0311
Fax: 1(408) 487-2600

tegional Headquarters

urope
Atmel Sarl
Route des Arsenaux 41
Case Postale 80
CH-1705 Fribourg
Switzerland
Tel: (41) 26-426-5555
Fax: (41) 26-426-5500

sia
Room 1219
Chinachem Golden Plaza
77 Mody Road Tsimshatsui
East Kowloon
Hong Kong
Tel: (852) 2721-9778
Fax: (852) 2722-1369

apan
9F, Tonetsu Shinkawa Bldg.
1-24-8 Shinkawa
Chuo-ku, Tokyo 104-0033
Japan
Tel: (81) 3-3523-3551
Fax: (81) 3-3523-7581

Atmel 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

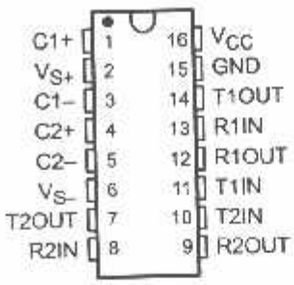
Printed on recycled paper.

1913C–MICRO–3/05 xM

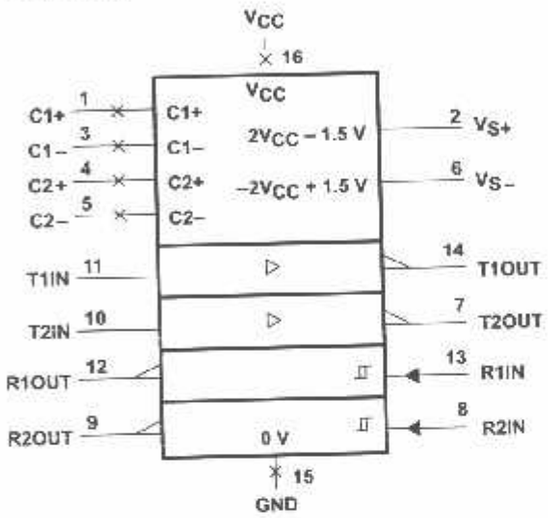
SLLS047G – FEBRUARY 1989 – REVISED AUGUST 1998

- Operates With Single 5-V Power Supply
- LinBiCMOS™ Process Technology
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current . . . 8 mA Typical
- Meets or Exceeds TIA/EIA-232-F and ITU Recommendation V.28
- Designed to be Interchangeable With Maxim MAX232
- Applications
  - TIA/EIA-232-F
  - Battery-Powered Systems
  - Terminals
  - Modems
  - Computers
- ESD Protection Exceeds 2000 V Per MIL-STD-883, Method 3015
- Package Options Include Plastic Small-Outline (D, DW) Packages and Standard Plastic (N) DIPs

D, DW, OR N PACKAGE
(TOP VIEW)



logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12.

description

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

The MAX232 is characterized for operation from 0°C to 70°C. The MAX232I is characterized for operation from –40°C to 85°C.

AVAILABLE OPTIONS

T_A	PACKAGED DEVICES		
	SMALL OUTLINE (D)	SMALL OUTLINE (DW)	PLASTIC DIP (N)
0°C to 70°C	MAX232D‡	MAX232DW‡	MAX232N
–40°C to 85°C	MAX232ID‡	MAX232IDW‡	MAX232IN

‡ This device is available taped and reeled by adding an R to the part number (i.e., MAX232DR).

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

Input supply voltage range, V_{CC} (see Note 1) . . . . . −0.3 V to 6 V
Positive output supply voltage range, V_{S+} . . . . . V_{CC} −0.3 V to 15 V
Negative output supply voltage range, V_{S-} . . . . . −0.3 V to −15 V
Input voltage range, V_I: Driver . . . . . −0.3 V to V_{CC} + 0.3 V
Receiver . . . . . ±30 V
Output voltage range, V_O: T1OUT, T2OUT . . . . . V_{S-} −0.3 V to V_{S+} + 0.3 V
R1OUT, R2OUT . . . . . −0.3 V to V_{CC} + 0.3 V
Short-circuit duration: T1OUT, T2OUT . . . . . Unlimited
Package thermal impedance, θ_{JA} (see Note 2): D package . . . . . 113°C/W
DW package . . . . . 105°C/W
N package . . . . . 78°C/W
Storage temperature range, T_{stg} . . . . . −65°C to 150°C
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds . . . . . 260°C

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

NOTE 1: All voltage values are with respect to network ground terminal.
2. The package thermal impedance is calculated in accordance with JESD 51, except for through-hole packages, which use a trace length of zero.

recommended operating conditions

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

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (unless otherwise noted)

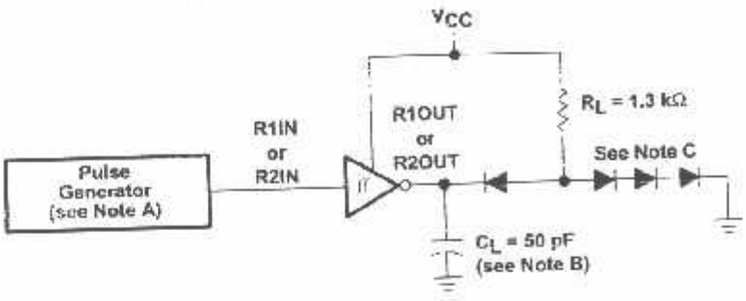
PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND		5	7		V
		R1OUT, R2OUT	I_{OH} = –1 mA		3.5			
V_{OL}	Low-level output voltage‡	T1OUT, T2OUT	R_L = 3 kΩ to GND			–7	–5	V
		R1OUT, R2OUT	I_{OL} = 3.2 mA				0.4	
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V,	T_A = 25°C		1.7	2.4	V
V_{IT-}	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V	T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V		0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5,	T_A = 25°C	3	5	7	kΩ
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0,	V_O = ±2 V	300			Ω
I_{OS}	Short-circuit output current§	T1OUT, T2OUT	V_{CC} = 5.5 V,	V_O = 0		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0				±200	µA
I_{CC}	Supply current		V_{CC} = 5.5 V,	All outputs open,		8	10	mA

† All typical values are at V_{CC} = 5 V, T_A = 25°C.
‡ The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
§ Not more than one output should be shorted at a time.

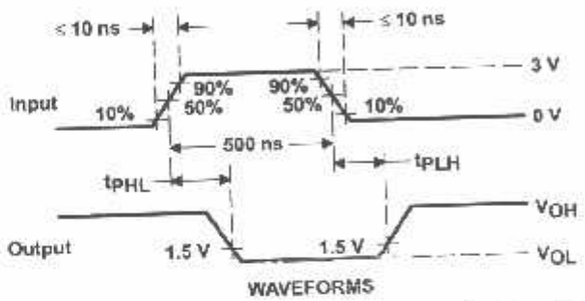
switching characteristics, V_{CC} = 5 V, T_A = 25°C

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	See Figure 1		500		ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	See Figure 1		500		ns
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, See Figure 2			30	V/µs
SR(tr)	Driver transition region slew rate	See Figure 2		3		V/µs

PARAMETER MEASUREMENT INFORMATION



TEST CIRCUIT

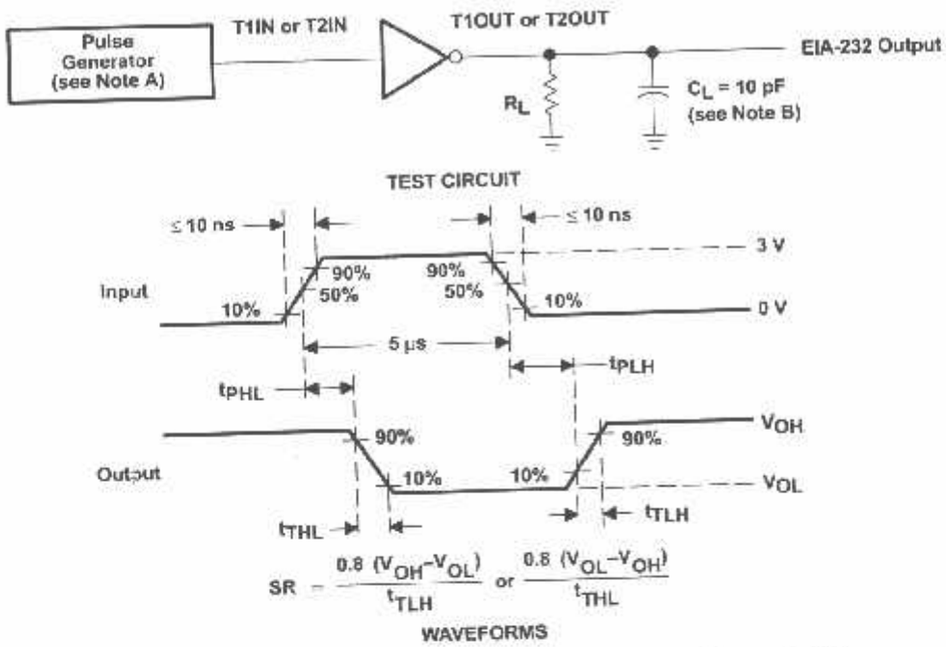


WAVEFORMS

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

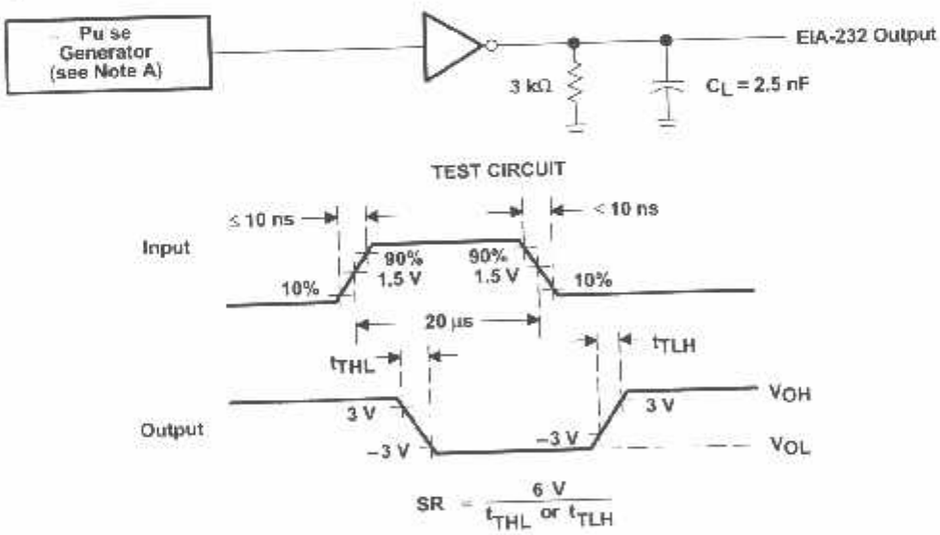
Figure 3. Receiver Test Circuit and Waveforms for t_{PHL} and t_{PLH} Measurements

PARAMETER MEASUREMENT INFORMATION



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

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



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

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

APPLICATION INFORMATION

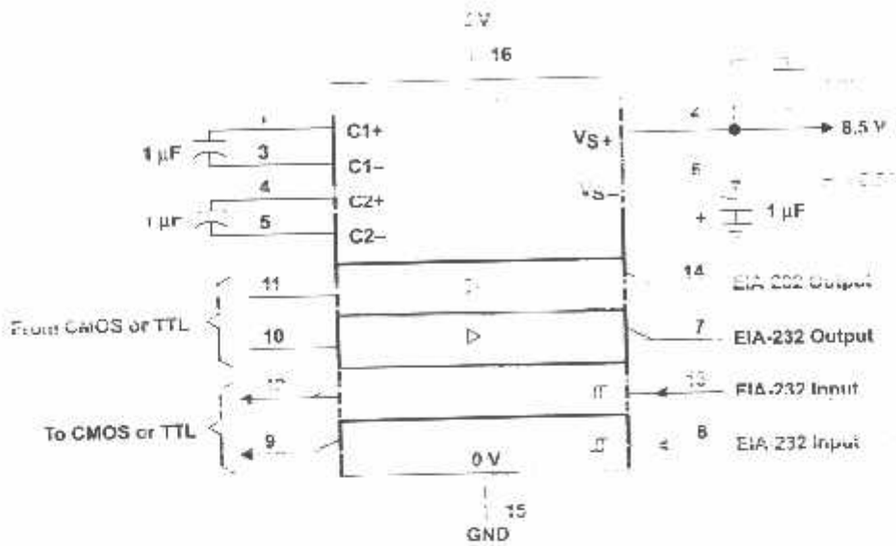
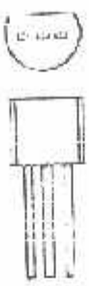


Figure 4. Typical Operating Circuit

TO—92



1.EMITTER

2.BASE

3.COLLECTOR 1 2 3

FEATURES

Power dissipation

P_{CM} : 0.4 W (Tamb=25℃)

Collector current

I_{CM} : 0.1 A

Collector-base voltage

$V_{(BR)CBO}$: 50 V

ELECTRICAL CHARACTERISTICS (Tamb=25℃ unless otherwise specified)

Parameter	Symbol	Test conditions	MIN	TYP	MAX	UNIT
Collector-base breakdown voltage	$V(BR)_{CBO}$	Ic= 100 μA, I_E=0	50			V
Collector-emitter breakdown voltage	$V(BR)_{CEO}$	Ic= 0.1 mA, I_B=0	45			V
Emitter-base breakdown voltage	$V(BR)_{EBC}$	I_E= 100 μA, I_C=0	5			V
Collector cut-off current	I_{CBO}	V_{CB}=50 V, I_E=0			0.1	μA
Collector cut-off current	I_{CEO}	V_{CE}=35 V, I_B=0			0.1	μA
Emitter cut-off current	I_{EBO}	V_{EB}= 3 V, I_C=0			0.1	μA
DC current gain(note)	$H_{FE(1)}$	V_{CE}= 5 V, I_C= 1mA	60		1000	
Collector-emitter saturation voltage	V_{CE}(sat)	I_C= 100mA, I_B= 5 mA			0.3	V
Base-emitter saturation voltage	V_{BE}(sat)	I_C= 100 mA, I_B= 5mA			1	V
Transition frequency	f_T	V_{CE}= 5 V, I_C= 10mA f =30MHz	150			MHz

CLASSIFICATION OF $H_{FE(1)}$

Rank	A	B	C	D
Range	60-150	100-300	200-600	400-1000

Wing Shing Computer Components Co., (H.K.)Ltd. Tel:(852)2341 9276 Fax:(852)2797 8153
Homepage http://www.wingshing.com E-mail wsceltd@hkstar.com

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised April 2000

DM74LS164
8-Bit Serial In/Parallel Out Shift Register

General Description

These 8-bit shift registers feature gated serial inputs and an asynchronous clear. A low logic level at either input inhibits entry of the new data, and resets the first flip-flop to the low level at the next clock pulse, thus providing complete control over incoming data. A high logic level on either input enables the other input, which will then determine the state of the first flip-flop. Data at the serial inputs may be changed while the clock is HIGH or LOW, but only information meeting the setup and hold time requirements will be entered. Clocking occurs on the LOW-to-HIGH level transition of the clock input. All inputs are diode-clamped to minimize transmission-line effects.

Features

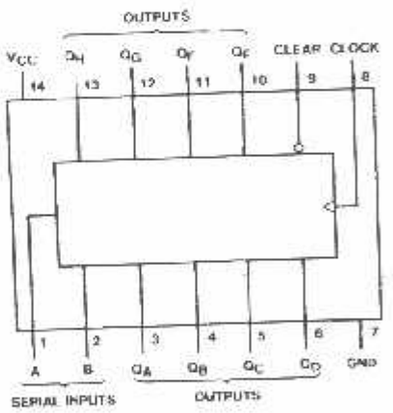
- Gated (enable/disable) serial inputs
- Fully buffered clock and serial inputs
- Asynchronous clear
- Typical clock frequency 36 MHz
- Typical power dissipation 80 mW

Ordering Code:

Order Number	Package Number	Package Description
DM74LS164M	M14A	14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
DM74LS164N	N14A	14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide

Devices also available in Tape and Reel. Specify by appending the suffix letter "X" to the ordering code.

Connection Diagram

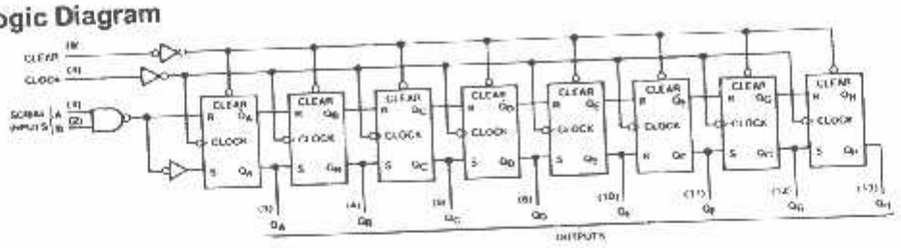


Function Table

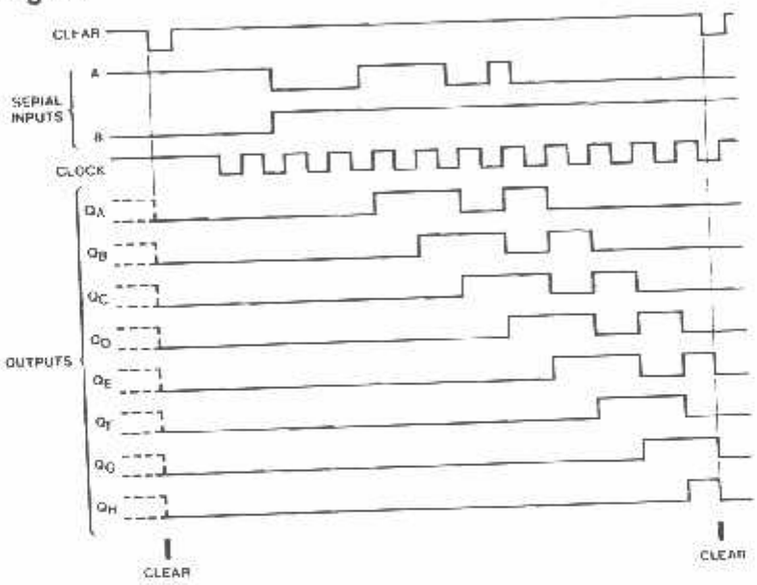
Inputs				Outputs			
Clear	Clock	A	B	Q_A	Q_B	...	Q_H
L	X	X	X	L	L	...	L
H	L	X	X	Q_{A0}	Q_{B0}	...	Q_{H0}
H	↑	H	H	H	Q_{An}	...	Q_{Gn}
H	↑	L	X	L	Q_{An}	...	Q_{Gn}
H	↑	X	L	L	Q_{An}	...	Q_{Gn}

H = HIGH Level (steady state)
L = LOW Level (steady state)
X = Don't Care (any input, including transitions)
↑ = Transition from LOW-to-HIGH level
Q_{A0}, Q_{B0}, Q_{H0} = The level of Q_A, Q_B, or Q_H, respectively, before the indicated steady-state input conditions were established.
Q_{An}, Q_{Gn} = The level of Q_A or Q_G before the most recent ↑ transition of the clock; indicates a one-bit shift.

Logic Diagram



Timing Diagram



Absolute Maximum Ratings(Note 1)

Supply Voltage	7V
Input Voltage	7V
Operating Free Air Temperature Range	0°C to +70°C
Storage Temperature Range	−65°C to +150°C

Note 1: The "Absolute Maximum Ratings" are those values beyond which the safety of the device cannot be guaranteed. The device should not be operated at these limits. The parametric values defined in the Electrical Characteristics tables are not guaranteed at the absolute maximum ratings. The "Recommended Operating Conditions" tables will define the conditions for actual device operation.

Recommended Operating Conditions

Symbol	Parameter		Min	Nom	Max	Units
V_{CC}	Supply Voltage		4.75	5	5.25	V
V_{IH}	HIGH Level Input Voltage		2			V
V_{IL}	LOW Level Input Voltage				0.8	V
I_{OH}	HIGH Level Output Current				−0.4	mA
I_{OL}	LOW Level Output Current				8	mA
f_{CLK}	Clock Frequency (Note 2)		0		25	MHz
t_W	Pulse Width (Note 2)	Clock	20			ns
		Clear	20			
t_{SU}	Data Setup Time (Note 2)		17			ns
t_H	Data Hold Time (Note 2)		5			ns
t_{REL}	Clear Release Time (Note 2)		30			ns
T_A	Free Air Operating Temperature		0		70	°C

Note 2: T_A = 25°C and V_{CC} = 5V.

Electrical Characteristics

over recommended operating free air temperature range (unless otherwise noted)

Symbol	Parameter	Conditions	Min	Typ (Note 3)	Max	Units
V_I	Input Clamp Voltage	V_{CC} = Min, I_I = −18 mA			−1.5	V
V_{OH}	HIGH Level Output Voltage	V_{CC} = Min, I_{OH} = Max, V_{IL} = Max, V_{IH} = Min	2.7	3.4		V
V_{OL}	LOW Level Output Voltage	V_{CC} = Min, I_{OL} = Max, V_{IL} = Max, V_{IH} = Min		0.35	0.5	V
		I_{OL} = 4 mA, V_{CC} = Min		0.25	0.4	
I_I	Input Current @ Max Input Voltage	V_{CC} = Max, V_I = 7V			0.1	mA
I_{IH}	HIGH Level Input Current	V_{CC} = Max, V_I = 2.7V			20	µA
I_{IL}	LOW Level Input Current	V_{CC} = Max, V_I = 0.4V			0.4	mA
I_{OS}	Short Circuit Output Current	V_{CC} = Max (Note 4)	−20		−100	mA
I_{CC}	Supply Current	V_{CC} = Max (Note 5)		16	27	mA

Note 3: All typicals are at V_{CC} = 5V, T_A = 25°C.

Note 4: Not more than one output should be shorted at a time, and the duration should not exceed one second.

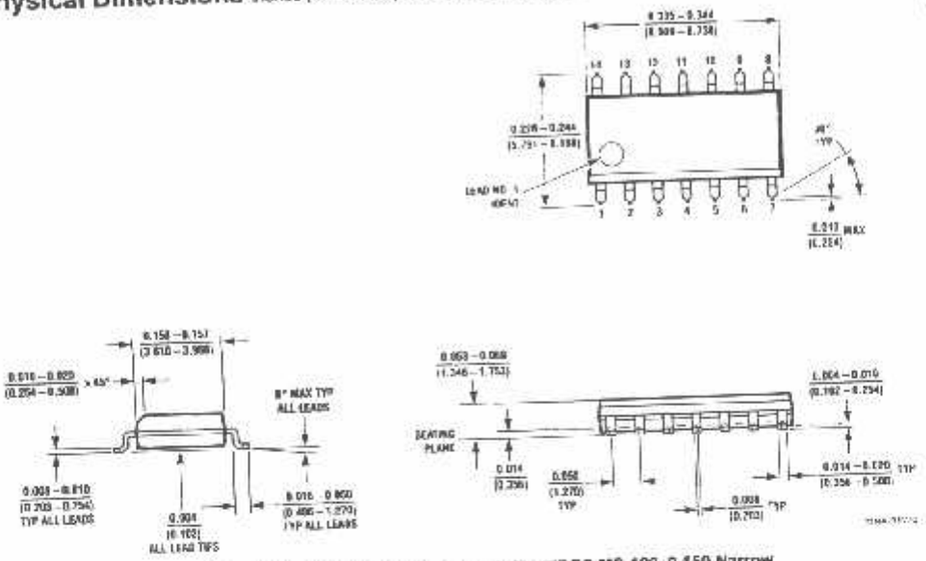
Note 5: I_{CC} is measured with all outputs OPEN, the SERIAL input grounded, the CLOCK input at 2.4V, and a momentary ground, then 4.5V, applied to the CLEAR input.

Switching Characteristics

at V_{CC} = 5V and T_A = 25°C

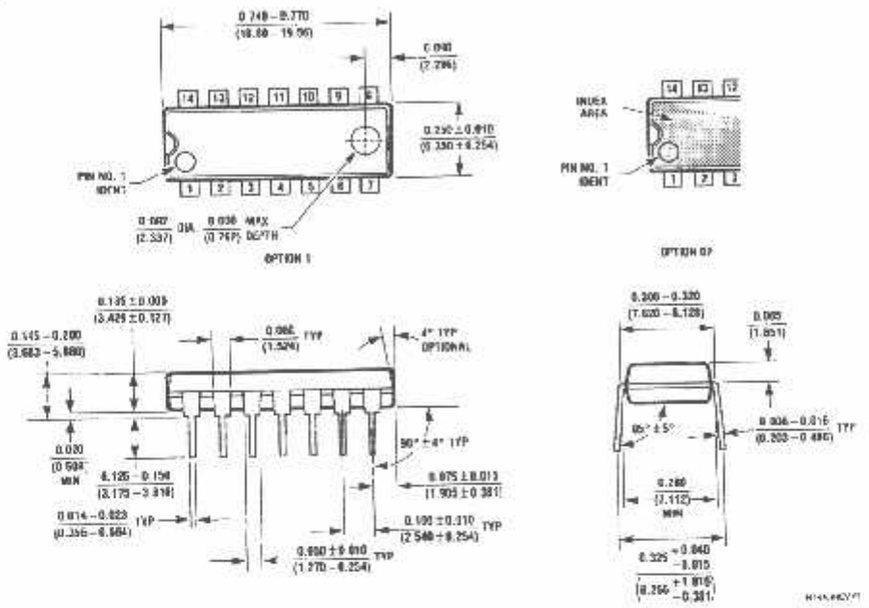
Symbol	Parameter	From (Input) To (Output)	R_L = 2 kΩ				Units
			C_L = 15 pF		C_L = 50 pF		
			Min	Max	Min	Max	
f_{MAX}	Maximum Clock Frequency		25				MHz
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Clock to Output		27		30	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clock to Output		32		40	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clear to Output		36		45	ns

Physical Dimensions inches (millimeters) unless otherwise noted



14-Lead Small Outline Integrated Circuit (SOIC), JEDEC MS-120, 0.150 Narrow
Package Number M14A

Physical Dimensions inches (millimeters) unless otherwise noted (Continued)



14-Lead Plastic Dual-In-Line Package (PDIP), JEDEC MS-001, 0.300 Wide
Package Number N14A

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

Formulir Perbaikan Ujian Skripsi

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

NAMA : Mahardika
N I M : 02.17.019
Perbaikan meliputi :

(1) Ubah rangkaian [illegible] ke rangkaian yg [illegible] & jelaskan mekanismenya kerja nya + pengaruhnya [illegible] volt tdk ada pembacaan [illegible] & [illegible] tabel ([illegible] ada pembahasan [illegible]).

(2) Jelaskan cara [illegible] menguji IC yg [illegible] IC [illegible] operasional (alamat memori bg?).

(3) Jelaskan mekanisme KFID kerja nya.

Malang

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNIK INDUSTRI
JURUSAN TEKNIK ELEKTRO

LEMBAR PERBAIKAN SKRIPSI

Nama Mahasiswa	: Mahardian Mahendradhata
NIM	: 02.17.019
Jurusan	: Teknik Elektro S1
Konsentrasi	: Teknik Elektronika
Hari / Tanggal	: Selasa / 27 Maret 2007

No	Materi Perbaikan	Paraf
1.	Gambar Rangkaian Keypad + Penjelasan Prinsip kerja + Pengujian	
2.	Rangkaian Mikrokontroller + Penjelasanya	
3.	Penjelasan Mekanisme RFID	

Telah Diperiksa /Disetujui:

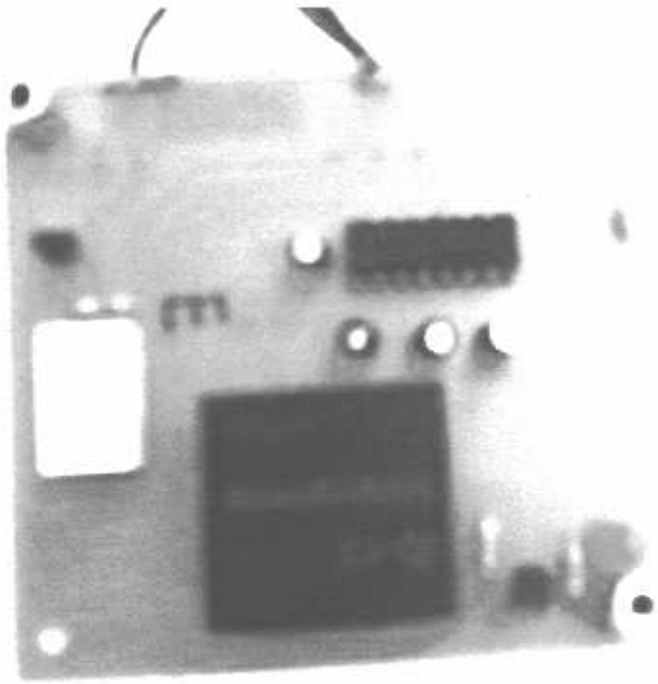
PENGUJI II

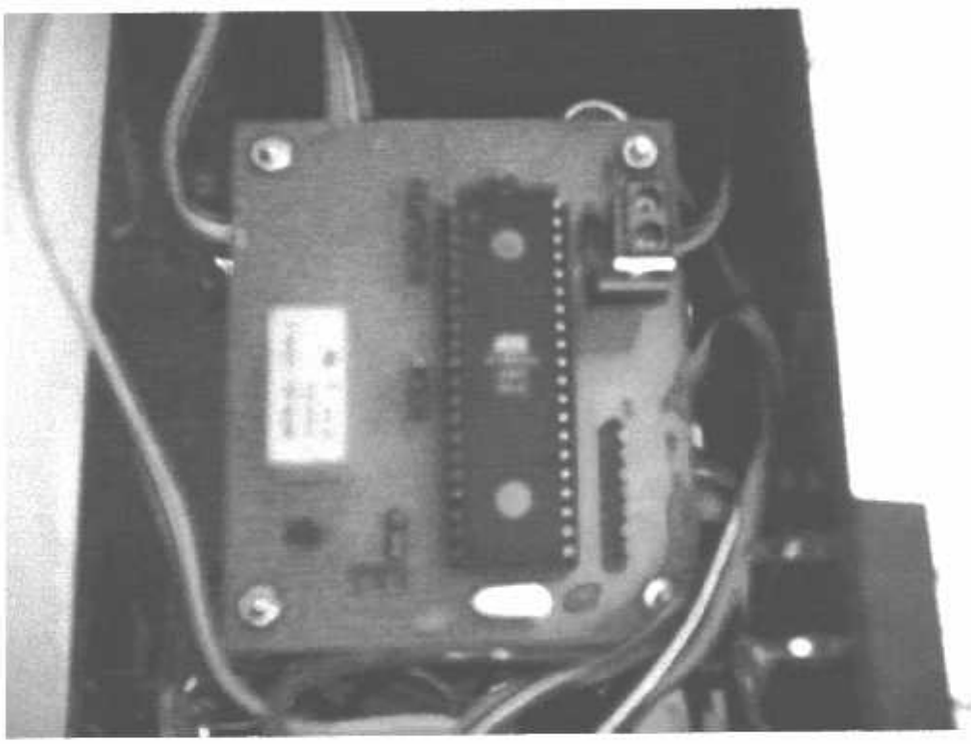
DR. Cahyo Chrysdian, Msc
NIP. 1030400412

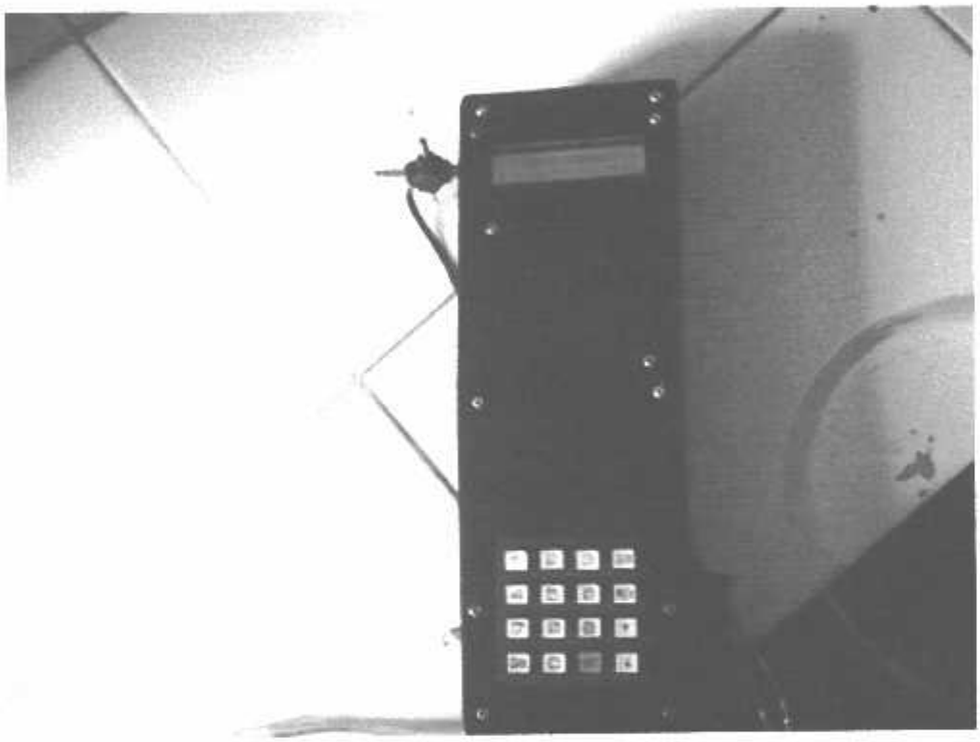
Mengetahui,

Dosen pembimbing

Joseph Dedy Irawan, ST, MT.
NIP. 132 315 178

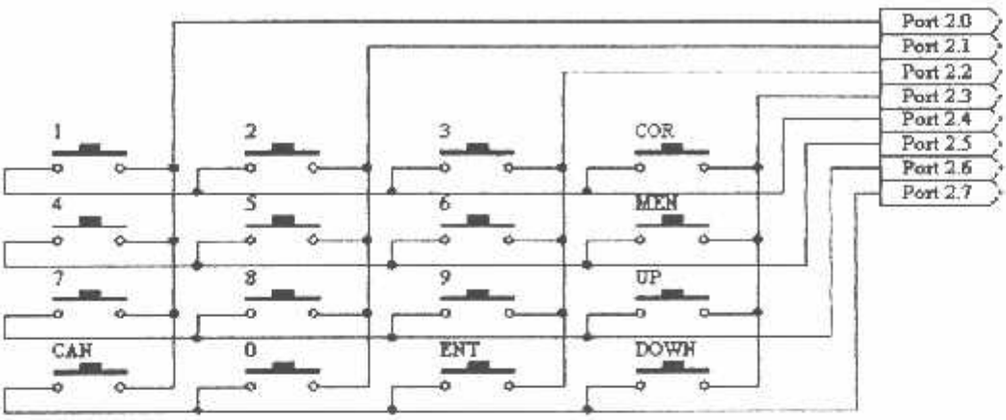






LEMBAR PERBAIKAN

1. Prinsip Kerja Keypad



Gambar Rangkaian Keypad

Proses scanning keypad pada mikrokontroller AT89S52

Membuat listing program untuk scanning tombol (lihat di BAB IV):

```
# include <at89x51>
unsigned char temp;
char Tombolnya()
```

Scanning pada baris pertama :

- P2=0xFE (seluruh port 2 dikasih logika dengan program FE h)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	1	1	1	1	1	0

- Switch ((P2>>4)&0x0F) : (seluruh port 2 digeser sebanyak 4x)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	1	0	1	1	1	1

Port 2.4 berlogika 0 jadi pada baris pertama pada rangkaian diatas aktif dan siap untuk dilakukan proses scanning dan sebelumnya di ANDkan dengan 0Fh

- Case 0x0E

```
return '1' ;
break  ;
```

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	1	0

Pada port 2.0 berlogika 0 maka akan terjadi pengaktifan pada kolom pertama dan terjadi pertemuan antara baris pertama (port 2.4) dengan kolom pertama (port 2.0), sehingga identitas tombol yang dikeluarkan '1'.

- Case 0x0D

```
return '2' ;
break  ;
```

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	0	1

Pada port 2.1 berlogika 0 maka akan terjadi pengaktifan pada kolom kedua dan terjadi pertemuan antara baris pertama (port 2.4) dengan kolom kedua (port 2.1), sehingga identitas tombol yang dikeluarkan '2'.

- Case 0x0B

```
return '3' ;
break  ;
```

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	0	1	1

Pada port 2.2 berlogika 0 maka akan terjadi pengaktifan pada kolom ketiga dan terjadi pertemuan antara baris pertama (port 2.4) dengan kolom ketiga (port 2.2), sehingga identitas tombol yang dikeluarkan '3'.

- Case 0x07

```
return 'R' ;
break  ;
```

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	0	1	1	1

Pada port 2.3 berlogika 0 maka akan terjadi pengaktifan pada kolom keempat dan terjadi pertemuan antara baris pertama (port 2.4) dengan kolom keempat (port 2.3), sehingga identitas tombol yang dikeluarkan 'R'.

Scanning pada baris kedua :

- P2=0xFD (seluruh port 2 dikasih logika dengan program FD h)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	1	1	1	1	0	1

- Switch ((P2>>4)&0x0F) : (seluruh port 2 digeser sebanyak 4x)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	0	1	1	1	1	1

Port 2.5 berlogika 0 jadi pada baris kedua pada rangkaian diatas aktif dan siap untuk dilakukan proses scanning dan sebelumnya di ANDkan dengan 0Fh.

➢ Case 0x0E
  return '4' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	1	0

Pada port 2.0 berlogika 0 maka akan terjadi pengaktifan pada kolom pertama dan terjadi pertemuan antara baris kedua (port 2.5) dengan kolom pertama (port 2.0), sehingga identitas tombol yang dikeluarkan '4'.

- Case 0x0D
  return '5' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	0	1

Pada port 2.1 berlogika 0 maka akan terjadi pengaktifan pada kolom kedua dan terjadi pertemuan antara baris kedua (port 2.5) dengan kolom kedua (port 2.1), sehingga identitas tombol yang dikeluarkan '5'.

- Case 0x0B
  return '6' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	0	1	1

Pada port 2.2 berlogika 0 maka akan terjadi pengaktifan pada kolom ketiga dan terjadi pertemuan antara baris kedua (port 2.5) dengan kolom ketiga (port 2.2), sehingga identitas tombol yang dikeluarkan '6'.

- Case 0x07
  return 'M' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	0	1	1	1

Pada port 2.3 berlogika 0 maka akan terjadi pengaktifan pada kolom keempat dan terjadi pertemuan antara baris kedua (port 2.5) dengan kolom keempat (port 2.3), sehingga identitas tombol yang dikeluarkan 'M'.

Scanning pada baris ketiga :

- P2=0xFB (seluruh port 2 dikasih logika dengan program FD h)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	1	1	1	0	1	1

- Switch ((P2>>4)&0x0F) : (seluruh port 2 digeser sebanyak 4x)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	0	1	1	1	1	1	1

Port 2.6 berlogika 0 jadi pada baris ketiga pada rangkaian diatas aktif dan siap untuk dilakukan proses scanning dan sebelumnya di ANDkan dengan 0Fh.

- Case 0x0E
  return '7' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	1	0

Pada port 2.0 berlogika 0 maka akan terjadi pengaktifan pada kolom pertama dan terjadi pertemuan antara baris ketiga (port 2.6) dengan kolom pertama (port 2.0), sehingga identitas tombol yang dikeluarkan '7'.

- Case 0x0D
  return '8' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	0	1

Pada port 2.1 berlogika 0 maka akan terjadi pengaktifan pada kolom kedua dan terjadi pertemuan antara baris ketiga (port 2.6) dengan kolom kedua (port 2.1), sehingga identitas tombol yang dikeluarkan '8'.

- Case 0x0B
  return '9' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	0	1	1

Pada port 2.2 berlogika 0 maka akan terjadi pengaktifan pada kolom ketiga dan terjadi pertemuan antara baris ketiga (port 2.6) dengan kolom ketiga (port 2.2), sehingga identitas tombol yang dikeluarkan '6'.

- Case 0x07
  return 'U' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	0	1	1	1

Pada port 2.3 berlogika 0 maka akan terjadi pengaktifan pada kolom keempat dan terjadi pertemuan antara baris ketiga (port 2.6) dengan kolom keempat (port 2.3), sehingga identitas tombol yang dikeluarkan 'U'.

Scanning pada baris keempat :

- P2=0xF7 (seluruh port 2 dikasih logika dengan program FD h)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
1	1	1	1	0	1	1	1

- Switch ((P2>>4)&0x0F) : (seluruh port 2 digeser sebanyak 4x)

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	1	1	1	1	1	1	1

Port 2.7 berlogika 0 jadi pada baris keempat pada rangkaian diatas aktif dan siap untuk dilakukan proses scanning dan sebelumnya di ANDkan dengan 0Fh.

➢ Case 0x0E
return 'N' ;
break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	1	0

Pada port 2.0 berlogika 0 maka akan terjadi pengaktifan pada kolom pertama dan terjadi pertemuan antara baris keempat (port 2.7) dengan kolom pertama (port 2.0), sehingga identitas tombol yang dikeluarkan 'N'.

➢ Case 0x0D
return '0' ;
break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	1	0	1

Pada port 2.1 berlogika 0 maka akan terjadi pengaktifan pada kolom kedua dan terjadi pertemuan antara baris keempat (port 2.7) dengan kolom kedua (port 2.1), sehingga identitas tombol yang dikeluarkan '0'.

➢ Case 0x0B
return 'E' ;
break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	1	0	1	1

Pada port 2.2 berlogika 0 maka akan terjadi pengaktifan pada kolom ketiga dan terjadi pertemuan antara baris keempat (port 2.7) dengan kolom ketiga (port 2.2), sehingga identitas tombol yang dikeluarkan 'E'.

- Case 0x07
  return '.' ;
  break ;

Port 2.7	Port 2.6	Port 2.5	Port 2.4	Port 2.3	Port 2.2	Port 2.1	Port 2.0
0	0	0	0	0	1	1	1

Pada port 2.3 berlogika 0 maka akan terjadi pengaktifan pada kolom keempat dan terjadi pertemuan antara baris keempat (port 2.7) dengan kolom keempat (port 2.3), sehingga identitas tombol yang dikeluarkan '.'.

- Void main()

```
While(1)
Temp=tombolnya();
P1=temp;
```

Mengeluarkan hasil scanning pada Port 1.

Tabel Pengujian Keypad

Input Keypad	Output							
	P2.0	P2.1	P2.2	P2.3	P2.4	P2.5	P2.6	P2.7
	Teg	Teg	Teg	Teg	Teg	Teg	Teg	Teg
1	0.62	4.70	4.72	4.71	0.52	4.58	4.62	4.72
Tanpa Penekanan	4.72	4.63	4.82	4.70	4.72	4.82	4.81	4.72
2	4.76	0.60	4.52	4.72	0.59	4.60	4.74	4.72
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
3	4.64	4.61	0.61	4.72	0.65	4.64	4.72	4.78
Tanpa Penekanan	4.72	4.63	4.82	4.70	4.82	4.78	4.81	4.82
4	0.72	4.72	4.55	4.69	4.61	0.62	4.75	4.77
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
5	4.64	0.70	4.69	4.75	4.68	0.68	4.70	4.69
Tanpa Penekanan	4.72	4.63	4.72	4.80	4.68	4.62	4.81	4.72
6	4.70	4.68	0.69	4.69	4.68	0.68	4.71	4.69
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
7	0.69	4.71	4.69	4.69	4.72	4.71	0.63	4.69
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
8	4.69	0.70	4.70	4.62	4.64	4.63	0.70	4.68
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
9	4.68	4.70	0.70	4.69	4.69	4.68	0.70	4.71
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.80	4.79	4.81	4.78
0	4.71	0.69	4.71	4.70	4.69	4.62	4.70	0.71
Tanpa Penekanan	4.72	4.63	4.78	4.80	4.82	4.78	4.81	4.82
COR	4.71	4.62	4.69	0.70	0.66	4.72	4.65	4.69
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82
MEN	4.68	4.70	4.65	0.65	4.69	0.71	4.66	4.70
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.78	4.82	4.81	4.78
UP	4.71	4.73	4.70	0.73	4.72	4.71	0.71	4.71
Tanpa Penekanan	4.72	4.63	4.70	4.80	4.82	4.72	4.70	4.82
DOWN	4.72	4.67	4.62	0.61	4.71	4.61	4.70	0.70
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.70	4.82	4.77	4.70
CAN	0.70	4.64	4.68	4.62	4.70	4.72	4.64	0.71
Tanpa Penekanan	4.72	4.63	4.76	4.80	4.78	4.82	4.81	4.82
ENT	4.66	4.71	0.65	4.71	4.71	4.72	4.65	0.70
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82

Analisa

Dari tabel pengujian diatas, dapat dilihat bahwa setiap kali penekanan tombol yang berbeda, output tegangan di port 2 juga berbeda pula. Tegangan output pada port 2 yang mendekati 5 volt dianggap sebagai logika 'HIGH', dan jika mendekati 0 volt dianggap sebagai logika 'LOW'.

2. Prinsip kerja mikrokontroller adalah sebagai berikut:

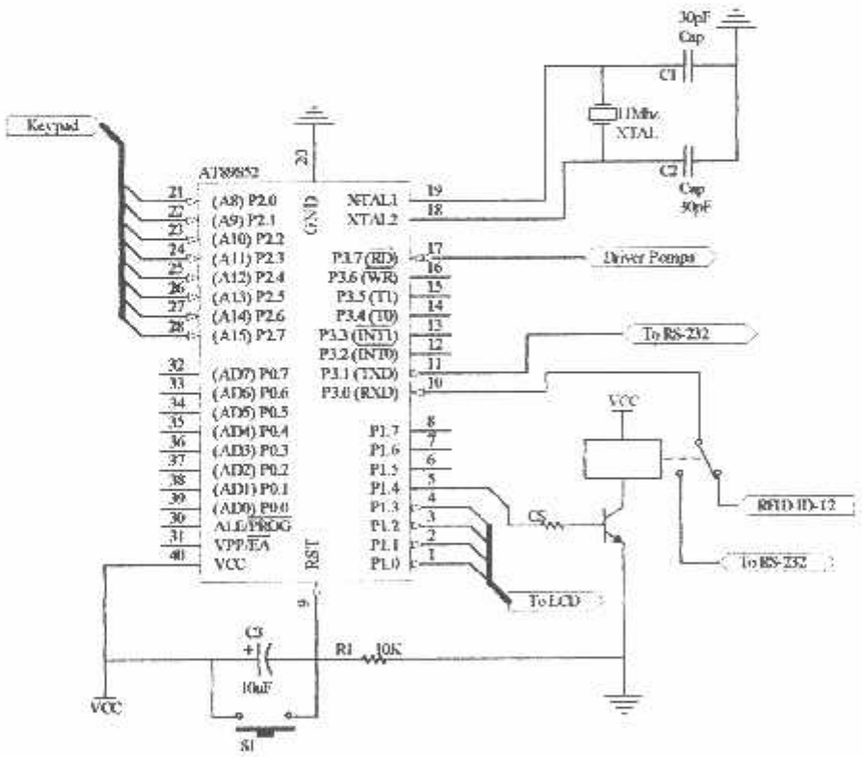
1. Berdasarkan nilai yang berada pada register *program counter*, mikrokonteroller mengambil data pada ROM dengan address sebagaimana nilai yang tertera dalam *program counter*. Selanjutnya program counter ditambah nilainya dengan 1(*increment*) secara otomatis. Data yang diambil tersebut adalah urutan instruksi program pengendali mikrokontroller yang sebelumnya telah dibuat oleh pemakai.
2. Instruksi tersebut diolah dan dijalankan. Proses pengerjaan bergantung pada jenis instruksi: bias membaca, mengubah nilai pada register, RAM, isi port, atau melakukan pembacaan dan dilanjutkan dengan pengubahan data.
3. *Program counter* telah berubah nilainya (baik karena penambahan otomatis sebagaimana pada langkah 1 diatas dan pengubahan pada langkah 2 diatas). Selanjutnya yang dilakukan mikrokontroller adalah mengulang kembali siklus ini pada langkah 1.demikian seterusnya sampai powerdimatikan.

Dari pengertian diatas dapat disimpulkan bahwa dasarnya unjuk kerja mikrokontroller sangatlah bergantung pada urutan instruksi yang dijalankanya, yaitu program yang ditulis dim ROM.

Dengan membuat program yang bermacam-macam, maka tentunya mikrokontroller dapat mengerjakan proses yang bermacam-macam pula. Fasilitas yang ada misalkan timer/counter, port I/O, serial port, ADC dapat dimanfaatkan program untuk menghasilkan proses yang diinginkan.

Penulisan program mikrokontroller pada umumnya menggunakan bahasa assembly ataupun memakai C++ yang kemudian dengan bantuan computer, dan disalin ke dalam ROM mikrokontroller.

Mikrokontroller AT89S52 adalah suatu chip IC yang terdiri dari 40 pin, dalam perencanaan alat ini mikrokontroller digunakan sebagai pusat pengendali kerja dari alat yang dibuat tersebut, disinilah tersimpan program-program (*software*) perintah serta alamat yang akan dituju program. Agar mikrokontroller dapat dipergunakan dengan baik dalam perancangan alat ini diperlukan beberapa elemen penting seperti *Vcc, Ground*, rangkaian *Reset*, rangkaian *clock* dan juga Port I/O yang digunakan



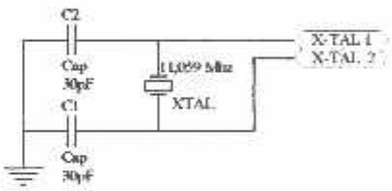
Gambar Rancangan pemakaian *port-port* mikrokontroller AT89S52

Rangkaian *Clock dan Reset* Minimum Sistem

Rangkaian yang mendukung mikrokontroller ada dua, yaitu rangkaian *clock* dan *reset.*

- *Clock* (X-TAL 1 dan X-TAL 2 pada pin 18 ,19)

Kecepatan proses yang dilakukan oleh mikrokontroller ditenyukan ole sumber *clock* (pewaktuan) yang mengendalikan mikrokontroller tersebut. System yang dirancang ini akan menggunakan *oscillator* internal yang sudah tersedia dalam chip mikrokontroller. Dalam perencanaan system ini frekuensi oscillator yang digunakan sebesar 11,059 MHz dan kapasitor penstabil sebesar 30pF. Berikut adalh gambar unruk rangkaian *clock.*



Gambar Rangkaian *Clock*

Berdasarkan rangkaian diatas maka didapatkan :

$$\text{Clock} = T = \frac{1}{f}$$

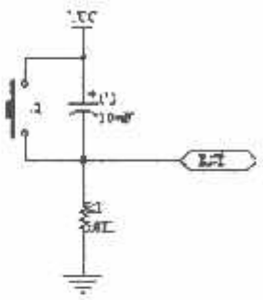
$$= \frac{1}{11{,}059}$$

$$= 0{,}09\ \mu s$$

- Rangkaian *Reset* (pin 9)

Untuk m*reset* mikrokontroller AT 89S52, maka RST (pin 9) diberi logika tinggi selama sekurangnya dua siklus mesin (*24 periode oscillator*). Untuk

membangkitkan sinyal *reset* dibutuhkan rangkaian *reset* yang diharapkan akan mempunyai kemampuan *power ON Reset*, Reset terjadi saat *Power* diaktifkan seperti yang ditunjukkan gambar di bawah ini.



Gambar Rangakaian *Reset*

Pada rangkaian *reset* diatas didapatkan Fo dari persamaan berikut :

Jika diketahui nilai R = 10 KΩ, dan C = 10 μF

$$Fo = \frac{1}{1{,}1.RC}$$

$$Fo = \frac{1}{1{,}1.10^{3}.10^{-6}}$$

$$Fo = 9{,}09 \text{ Hz}$$

Perangkat Lunak Mikrokontroller AT89S52

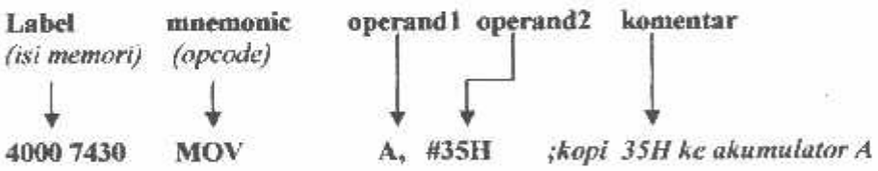
Perangkat Lunak yang digunakan untuk minimum sistem AT89S52 ini adalah menggunakan bahasa pemrograman bahasa C. Program yang ditulis dengan pemrograman bahasa C Label; Kode *Mnemonic* dan lain sebagainya, pada umumnya dinamakan sebagai Program Sumber (*Source Code*) yang belum bisa

diterima oleh prosesor untuk dijalankan sebagai program, tetapi harus diterjemahkan dulu menjadi bahasa mesin dalam bentuk kode hexa atau biner.

Pada skripsi ini Program Sumber diterjemahkan atau di*compile* menggunakan program *Keile51* bila *compile* tidak ada tanda *compile error* maka program benar, untuk mensimulasikan program mengunakan program TS Emulator 8051 V.1.1. Hasil *compile* berektensi "*Hex*", "*Obj*", "*Lst*", dan "*Bin*".

Pengalamatan

Data bisa berada diberbagai tempat yang berlainan, dengan demikian dikenal beberapa cara untuk menyebut data ('Addressing Mode'), antara lain sebagai berikut: Mode pengalamatan menjelaskan bagaimana operand dioperasikan. Berikut penjelasan dari berbagai mode pengalamatan. Bentuk program assembly yang umum ialah sebagai berikut :



1. Pengalamatan Tak Langsung

Operand pengalamatan tak langsung menunjuk ke sebuah register yang berisi lokasi alamat memori yang akan digunakan dalam operasi. untuk melaksanakan engalamatan tak langsung digunakan simbol @.

contoh :

ADD A,@R0 ;Tambahkan isi RAM yang lokasinya ditunjukan olehregister R0 ke akumulator

DEC @R1 ;Kurangi satu isi RAM yang alamatnya ditunjukan oleh register R1

MOVX @DPT,A ;Pidahkan isi dari akumulator ke momori luar yang lokasinya ditunjukan oleh data pointer (DPTR)

2. Pengalamatan Langsung

Pengalamatan langsung dilakukan dengan memberikan nilai ke suatu register secara langsug. untuk melaksanakan hal tersebut digunakan tanda #.

conttoh:

MOV A,#01H ;Isi akumuator dengan bilangan 01H

MOV DPTR,19AH ;Isi register DPTR dengan bilangan 19AH

3. Pengalamatan Bit

Pengalamatan Bit adalah penunjukan alamat lokasi bit .Untuk melaksanakan pengalamatan bit dilakukan dengan tanda titik (.).

Contoh;

SETB TR1 ; Set TR1 (logika 1 pada TR1)

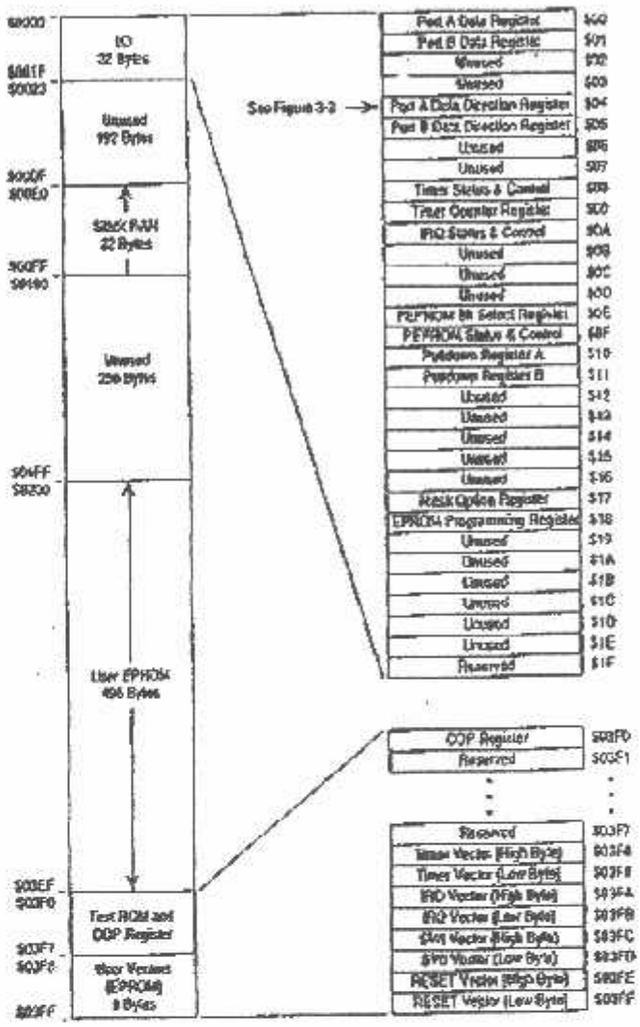
SETB P1.0 ; Logika 1 pada P1.0

PetaMemori

Karena terdapat ribuan dan bahkan lebih lokasi memori dalam suatu sistem mikrokontroler, menjadi penting untuk memiliki cara yang enak untuk menangani alamat masing-masing data dalam memori. Suatu peta memori adalah penggambaran yang mewakili semua spasi dalam memori mikrokontroler. Gambar di bawah adalah peta memori umum yang menggambarkan memori dalamMCS51.

Empat digit heksadesimal yang terletak pada bagian kiri dari gambar di bawah adalah alamat yang dimulai pada $0000 di atas dan terus bertambahsampai $03F di bagian bawah. Alamat $0000 berhubungan dengan awal lokasi memori sedangkan alamat $03FF berhubungan dengan lokasi memori akhir. Sedangkan ketrangan dalam kotak menunjukkan macam tipe dari memori dan isinya (RAM, EPROM, register I/O, dan sebagainya). Beberapa daerah, seperti register I/O, perlu dijelaskan lebih detail karena penting untuk mengetahui nama dari setiap lokasi. Setiap lokasi memori sebanyak 1024 ini memiliki delapan bit data seperti pada gambar di bawah

Lokasi memori 256 pertama ($0000-$00FF) dapat diakses oleh komputer dengan cara khusus yang sebut dengan mode pengalamatan langsung (direct addressing mode). Register I/O on-chip dan 32 byte RAM terletak dalam area $0000-$00FF. Dalam peta memori pada gambar di bawah terlihat konfigurasi penempatan pada area ini yang dipaparkan dalam kotak yang terletak di sebelah kanan.


$0000
I/O
32 Bytes
$001F
$0020
Unused
192 Bytes
$00DF
$00E0
256 x 8 RAM
32 Bytes
$00FF
$0100
Unused
256 Bytes
$04FF
$0500
User EPROM
496 Bytes
$03EF
$03F0
Test ROM and
COP Register
$03F7
$03F8
User Vectors
(EPROM)
8 Bytes
$03FF
See Figure 3-2
Port A Data Register $00
Port B Data Register $01
Unused $02
Unused $03
Port A Data Direction Register $04
Port B Data Direction Register $05
Unused $06
Unused $07
Timer Status & Control $08
Timer Counter Register $09
IRQ Status & Control $0A
Unused $0B
Unused $0C
Unused $0D
PEPROM Bit Select Register $0E
PEPROM Status & Control $0F
Pulldown Register A $10
Pulldown Register B $11
Unused $12
Unused $13
Unused $14
Unused $15
Unused $16
Mask Option Register $17
EPROM Programming Register $18
Unused $19
Unused $1A
Unused $1B
Unused $1C
Unused $1D
Unused $1E
Reserved $1F
COP Register $03FD
Reserved $03FE
Reserved $03F7
Timer Vector (High Byte) $03F8
Timer Vector (Low Byte) $03F9
IRQ Vector (High Byte) $03FA
IRQ Vector (Low Byte) $03FB
SWI Vector (High Byte) $03FC
SWI Vector (Low Byte) $03FD
RESET Vector (High Byte) $03FE
RESET Vector (Low Byte) $03FF

Pengalamatan memori

variable	Address	Size
serial_buffer	E0h	30
ID_buffer	FEh	9
data_lenght	R4	1
jumlahring	R5	1
serial_current	R6	1
mode	R7	1
digit	R8	1
timer_kedip	R9,R10	2
tim1	R11,R12	2
tim_tunggu_PC	R13,R14	2
pernahring	R2.0	1/8
pernahstd	R2.1	1/8
ceksambung	R2.2	1/8

Register file — **Data Address Space**

R0	$0000
R1	$0001
R2	$0002
.....	.....
R29	$001D
R30	$001E
R31	$001F

I/O Register

$00	$0020
$01	$0021
$02	$0022
.....	.....
$3D	$005D
$3E	$005E
$3F	$005F

Internal SRAM

$0060
$0061
.....
$025E
$025F

Penjelasan :

Catatan : pada pemrograman bahasa C pengalamatan diatur sendiri oleh compilernya.

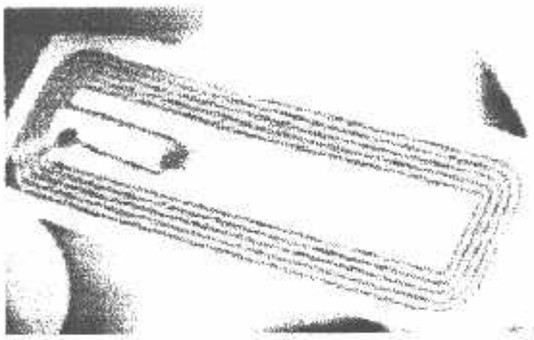
- Serial_buffer : variabel yang digunakan pada program untuk membuffer data serial yang akan dikirim disini dialamatkan pada address E0h sebanyak 30 byte tempat pemesanan di internal SRAM.
- ID_buffer : variabel yang digunakan pada program untuk membuffer data RFID yang akan dikirim disini dialamatkan pada address FEh sebanyak 9 byte tempat pemesanan di internal SRAM.
- Data_lenght : variabel yang digunakan pada program untuk memprediksi lebar data disini dialamatkan pada address R4 sebanyak 1 byte tempat pemesanan Register File.
- Jumlahring : variabel yang digunakan pada program untuk memprediksi banyaknya looping data disini dialamatkan pada address R5 sebanyak 1 byte tempat pemesanan Register File.
- Serial_current : variabel yang digunakan pada program untuk memprediksi aliran data serial disini dialamatkan pada address R6 sebanyak 1 byte tempat pemesanan Register File.
- Mode : variabel yang digunakan pada program untuk pengaturan mode data disini dialamatkan pada address R7 sebanyak 1 byte tempat pemesanan Register File.
- Digit : variabel yang digunakan pada program untuk memprediksi digit data disini dialamatkan pada address R8 sebanyak 1 byte tempat pemesanan Register File.
- Timer_kedip : variabel yang digunakan pada program untuk pewaktuan data disini dialamatkan pada address R9,R10 sebanyak 2 byte tempat pemesanan Register File.
- Tim1: variabel yang digunakan pada program untuk pewaktuan data disini dialamatkan pada address R11,R12 sebanyak 2 byte tempat pemesanan Register File.

- Tim_tunggu_PC: variabel yang digunakan pada program untuk pewaktuan data disini dialamatkan pada address R13,R14 sebanyak 2 byte tempat pemesanan Register File.
- Pernahring : variabel yang digunakan pada program untuk memprediksi banyaknya looping data disini dialamatkan pada address R2.0 sebanyak 1 bit dari 1 byte tempat pemesanan Register File. .
- Pernahstd : variabel yang digunakan pada program untuk memprediksi pengiriman data disini dialamatkan pada address R2.1 sebanyak 1 bit dari 1 byte tempat pemesanan Register File.
- Ceksambung : variabel yang digunakan pada program untuk memprediksi pengecekan data disini dialamatkan pada address R2.2 sebanyak 1 bit dari 1 byte tempat pemesanan Register File.

3. Mekanisme RFID

- **Prinsip kerja Reader danTag/transponder**

Suatu transponder secara induktif yang tergabungkan terdiri atas suatu data elektronik di dalam suatu microchip yang pada umumnya tunggal dan suatu coil area besar yang berfungsi sebagai suatu antena.



Gambar. RFID tag dengan silikon chip dan antena eksternal.

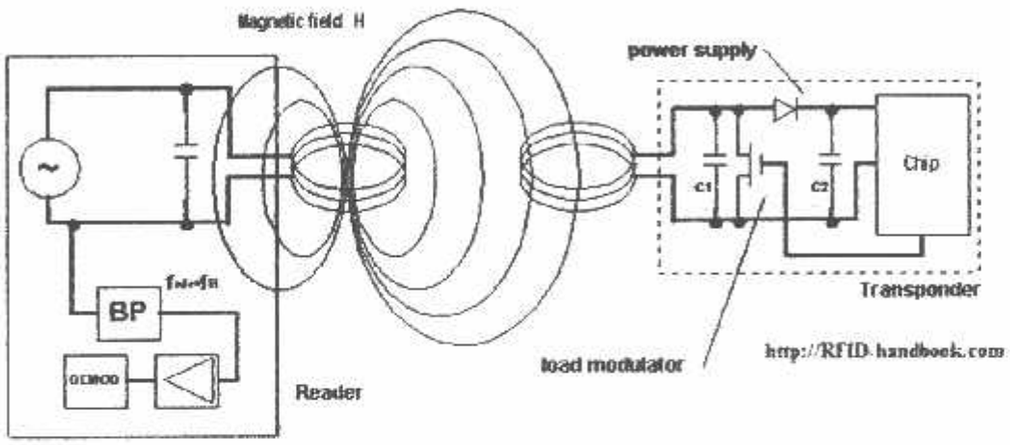
Secara induktif transponders dioperasikan dengan pasif. yaitu bahwa semua energi yang diperlukan untuk operasi microchip harus disajikan oleh pembaca Reader. coil antena pembaca menghasilkan suatu bidang elektro magnet frekwensi tinggi, yang menembus panampang-lintang area coil dan area di sekitar coil itu. Sebab panjang gelombang cakupan frekwensi menggunakan *low frekuensi* (125 kHz - 135 kHz)

Suatu bidang elektro magnet yang dipancarkan menembus coil antena transponder, yang mana saat terinduksi, suatu tegangan dihasilkan coil antena transponder. Tegangan ini berfungsi sebagai power untuk pengaktifan data di dalam microchip.

Suatu kapasitor C yang dihubungkan paralel dengan coil antena pembaca berkombinasi dengan induktans coil antena untuk membentuk suatu rangkaian resonansi paralel, dengan suatu frekwensi resonan yang bersesuaian dengan frekwensi transmisi

pembaca yang dihasilkan di dalam coil antena pembaca akan meningkatkan rangkaian resonan yang parallel tersebut, yang dapat digunakan untuk menghasilkan kekuatan bidang elektro magnet.

Coil Antena transponder dan kapasitor untuk membentuk suatu rangkaian resonan dan mengatur kesesuaian kepada frekwensi transmisi pembaca . Tegangan di transponder coil akan meningkatkan rangkaian frekuensi resonan pararel pada tag.



Gambar . Komunikasi antara reader dan transponder (tag)

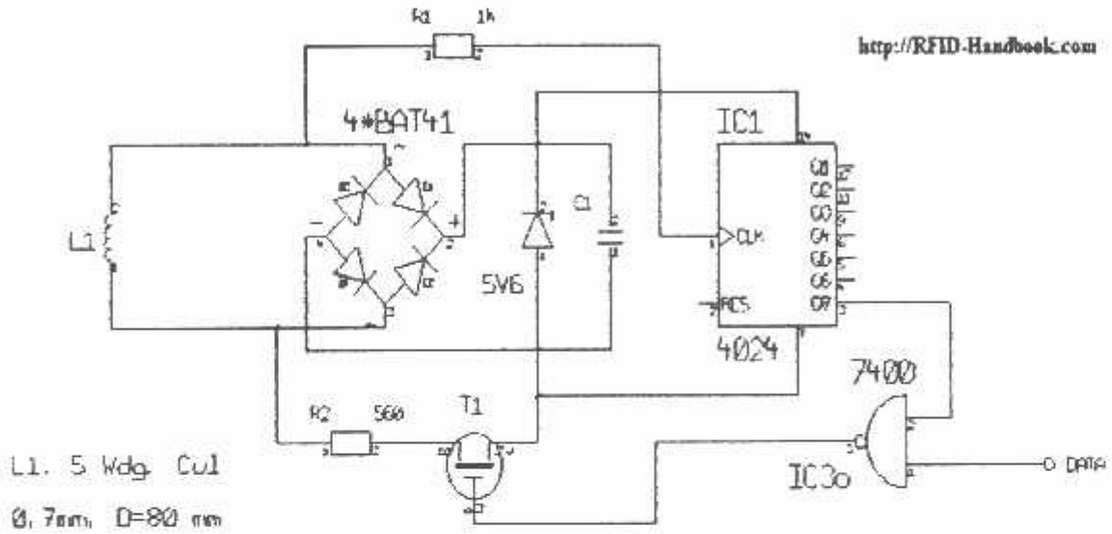
- **Pembacaan**

Jika suatu resonan transponder (yaitu. self-resonant frekwensi transponder bersesuaian dengan frekwensi transmisi pembaca) berada di dalam bidang magnetis antena pembaca akan mendapatkan energi seri dari medan magnet itu. Konsumsi energi terjadi ketika adanya penurunan-voltase di dalam antenna pembaca

Switch transponder sekali-kali melakukan pembalasan beban di antena transponder's dan itu mempengaruhi perubahan tegangan di antena pembaca juga mempunyai efek dari suatu modulasi amplitudo tegangan oleh antena transponder. Pada

saat menswitch sekali-kali resistor beban dikendalikan oleh data, kemudian data ini dapat ditransfer dari transponder kepada pembaca Perpindahan Data jenis ini disebut modulasi beban.

Untuk mengambil kembali data di dalam pembaca, pengaturan voltase di penyearah antena pembaca digunakan demodulasi untuk membalikkan modulasi sinyal.



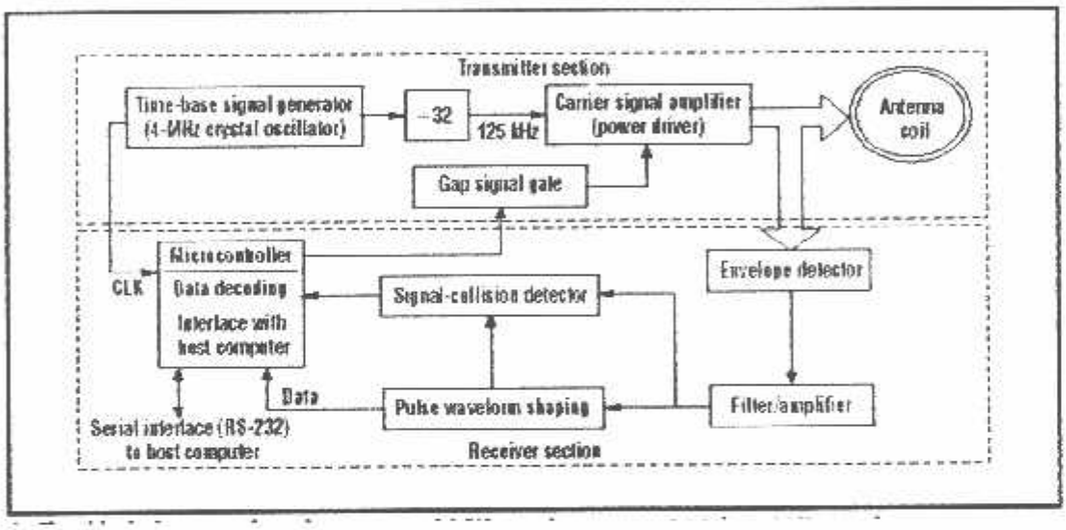
Gambar supply dalam transponder.

Gambar di atas: Jika resistor beban di dalam transponder diswitch on/off pada suatu frekwensi dasar tinggi fH, kemudian dua garis spektrum diciptakan jauh ± fH di sekitar frekwensi transmisi pembaca, dan ini dapat dengan mudah dideteksi (fH harus kurang dari f READER). Di dalam istilah teknologi radio frekwensi dasar yang baru dapat disebut subcarrier. Perpindahan data dengan FSK atau PSK modulasi subcarrier pada saat data mengalir. akan menghadirkan suatu modulasi amplitudo subcarrier.

- **Modulasi yang digunakan**

Pembaca terdiri dari suatu pemancaran dan suatu penerima bagian. Bagian Pemancaran meliputi suatu generator freknensi carrier, gerbang gap sinyal, dan suatu rangkaian antena. Bagian penerima meliputi suatu detektor puncak gelombang, suatu amplifier/filter sinyal, suatu detektor signal-collision, dan suatu microcontroller untuk pengolahan data.

Pembaca juga berkomunikasi dengan suatu komputer eksternal. Suatu diagram blok dasar RFID pembaca yang ditunjukkan



Gambar Blok diagram model RFID Reader dengan sinyal FSK 125 kHz

125-Khz sinyal carrier dihasilkan dari] 4-MHz timebase sinyal generator dari suatu osilator kristal. Sinyal 125-Khz dikuatkan ke dalam NOR gate dan two-stage menggerakkan amplifier sinyal carrier 125-Khz kemudian diteruskan ke dalam rangkaian antena yang dibentuk oleh L (162 mH) dan C (0.01 mF), rangkaian L Dan C membentuk suatu series-resonant dengan suatu 125-kHz frekuensi resonans., sinyal

carrier (125 kHz) disaring untuk dikirim ke coil antena. Rangkaian tersebut menyediakan suatu impedansi minimum di frekuensi resonans. Ini mengakibatkan memaksimalkan arus antena dan oleh karena itu, intensitas medan magnet dimaksimalkan di frekwensi yang resonan.

sinyal Gap mengendalikan 125-kHz rangkaian pengarah antena sepanjang isyarat gap " tinggi maka tidak ada RF sinyal di coil antena selama periode gap ini.

- **Pengiriman Data**

Saat ini model alat identifikasi sangatlah bermacam – macam ada yang berupa kartu dengan lubang, barcode, RFID, dll. RFID (RF Identification) merupakan suatu alat untuk identifikasi yang biasanya ditempelkan pada barang atau dibuat menjadi kartu. Di dalam artikel ini akan dibahas mengenai cara membaca format data yang dikeluarkan oleh RFID reader dengan format output ASCII.

RFID reader mempunyai banyak sekali tipe, antara lain: ID-10, ID-19, EM-13, dll. biasanya RFID reader ini memiliki dua bentuk output serial yaitu: ASCII dan Wiegand 26-bit. Yang paling banyak digunakan adalah output dengan format ASCII, karena output ini sangat mudah untuk dihubungkan pada mikrokontroler atau PC menggunakan komunikasi serial UART.

Format Data ASCII

Data kode yang dikirim dari tag RFID ke pembaca yaitu sebanyak 128 bit (16 byte ASCII). berikut format data ASCII yang dikirimkan :

02	10 data karakter ASCII	checksum	CR	LF	03

Gb1. Format data ASCII

Checksum merupakan hasil EXOR (Exclusive OR) dari 5 biner data byte. Untuk lebih jelasnya tentang cara pembacaan format ASCII, lihat contoh berikut.

Misalnya data output serial (dalam hexadesimal) yang kita tangkap adalah sebagai berikut:

02	30	34	36	32	30	31	44	37	36	43	44	43	0D	0A	03

Langkah pertama adalah merubah semua nilai data diatas menjadi karakter ASCII. Misalnya 30H menjadi karakter "0", 34H menjadi karakter "4", dst. Langkah kedua adalah menyusun data – data tersebut ke dalam Format Data ASCII seperti gb 1. Kemudian ambil 10 data karakter ASCII. Dalam contoh ini berarti data tersebut adalah:

30	34	36	32	30	31	44	37	36	43	Data Heksa
		6	2	0	1	D	7	6	C	Data ASCII

Untuk data dengan warna biru merupakan data untuk jenis – jenis kartu dan tidak digunakan dalam proses konversi, yang akan dipakai disini adalah data yang ke 3 s/d 10. Hasil konversi dari data heksa ke dalam data ASCII adalah "6201D76C". Gabungkan karakter data ASCII menjadi bilangan Hexadesimal, kemudian konversikan bilangan hexadesimal tsb ke dalam desimal. Hasilnya sebagai berikut: 6201D76C H menjadi 1644287852 (ini merupakan nomor kartu sebenarnya yang tertera pada badan kartu tsb). Cara ini hanya berlaku pada kartu yang tidak dienkripsi.

```
lude <at89x51.h>
lude "Pending.c"
lude "lcdku.c"
lude "myser1.c"

ine cs                          P3_2//1_4
ine tekan    P3_3
ine pompa    P3_7


gned char j[16],i,trfid,k[20],a,b1,b2,b3,b4,b5,b6,b7,b8,tandap;
gned char liter[7],trl;
signed int atarget,r;
gned long Ndata,atarget,r;
t;

ine MatrikX1        P2_4
ine MatrikX2        P2_5
ine MatrikX3        P2_6
ine MatrikX4        P2_7
ine MatrikY1        P2_0
ine MatrikY2        P2_1
ine MatrikY3        P2_2
ine MatrikY4        P2_3

 Tombolnya()

    MatrikX1        = 0                 ;
    MatrikX2        = 1                 ;
    MatrikX3        = 1                 ;
    MatrikX4        = 1                 ;
    switch ( P2 & 0x0F )
    {
    case 0x0E:
            while(MatrikY1==0){;}
            return '1';
            break;
    case 0x0D:
            while(MatrikY2==0){;}
            return '4';
            break;
    case 0x0B:
            while(MatrikY3==0){;}
            return '7';
            break;
    case 0x07:
            while(MatrikY4==0){;}
            return 'N';
            break;
    }
    MatrikX1        = 1                 ;
    MatrikX2        = 0                 ;
    MatrikX3        = 1                 ;
    MatrikX4        = 1                 ;
    switch ( P2 & 0x0F )
    {
    case 0x0E:
            while(MatrikY1==0){;}
            return '2';
            break;
    case 0x0D:
            while(MatrikY2==0){;}
            return '5';
            break;
    case 0x0B:
            while(MatrikY3==0){;}
            return '8';
            break;
    case 0x07:
            while(MatrikY4==0){;}
            return '0';
            break;
    }
    MatrikX1        = 1                 ;
    MatrikX2        = 1                 ;
    MatrikX3        = 0                 ;
    MatrikX4        = 1                 ;
    switch ( P2 & 0x0F )
    {
    case 0x0E:
            while(MatrikY1==0){;}
            return '3';
            break;
    case 0x0D:
            while(MatrikY2==0){;}
            return '6';
            break;
    case 0x0B:
            while(MatrikY3==0){;}
            return '9';
            break;
    case 0x07:
            while(MatrikY4--0){;}
            return 'E';
```

```
        break;
    }
    MatrikX1        = 1                     ;
    MatrikX2        = 1                     ;
    MatrikX3        = 1                     ;
    MatrikX4        = 0                     ;
    switch ( P2 & 0x0F )
    {
    case 0x0E:
            while(MatrikY1==0){;}
            return 'R';
            break;
    case 0x0D:
            while(MatrikY2==0){;}
            return 'M';
            break;
    case 0x0B:
            while(MatrikY3==0){;}
            return 'U';
            break;
    case 0x07:
            while(MatrikY4==0){;}
return 'D';
turn '.';
            break;
    default:
            return 'z';
    //      break;
    }


 key()

Tombolnya();
e (asu=='z') asu=Tombolnya();

 rkey()

Tombolnya();
e (asu=='z') asu=Tombolnya();
rn asu;

 SerialInterrupt (void) interrupt 4 using 1


      while(!RI){;}
      RI                = 0     ;
      if (SBUF==2) {trfid=1;i=0;}
      else if(SBUF==3) {trfid=2;t=0;}
      else ;
if (trfid==1)

[i]=SBUF;
++;


oid cetakrfid()

=3;
ile(i<11)

asu=j[i];
dataout();
i++;
 }
 t=1;
 //pos(2,1);

nsigned char kon(unsigned char n)

if (n <= 0x3a) return(n-0x30);
else return(n-0x37);
/*else if(n=='A') return(0xa);
else if(n=='B') return(0xb);
else if(n=='C') return(0xc);
else if(n=='D') return(0xd);
else if(n=='E') return(0xe);
else if(n=='F') return(0xf);
else return(0);*/

oid konrfid()


urjannah=0x10000000*kon(j[3])+0x1000000*kon(j[4])+0x100000*kon(j[5])+0x10000*kon(j[6])+0x1000*kon(j[7])+0x
*kon(j[8])+0x10*kon(j[9])+kon(j[10]);
1=kon(j[3]);b2=kon(j[4]);b3=kon(j[5]);b4=kon(j[6]);b5=kon(j[7]);b6=kon(j[8]);
7=kon(j[9]);b8=kon(j[10]);
data=0x10000000*b1+0x1000000*b2+0x100000*b3+0x10000*b4+0x1000*b5+0x100*b6+0x10*b7+b8;

/Nurjannah=2657107;
su=Ndata/1000000000+0x30;k[0]=asu;
ataout();
su=(Ndata/100000000)%10+0x30;k[1]=asu;
```

```
taout();
u=(Ndata/10000000)%10+0x30;k[2]=asu;
taout();
u=(Ndata/1000000)%10+0x30;k[3]=asu;
taout();
u=(Ndata/100000)%10+0x30;k[4]=asu;
taout();
u=(Ndata/10000)%10+0x30;k[5]=asu;
taout();
u=(Ndata/1000)%10+0x30;k[6]=asu;
taout();
u=(Ndata/100)%10+0x30;k[7]=asu;
taout();
u=(Ndata/10)%10+0x30;k[8]=asu;
taout();
u=Ndata%10+0x30;k[9]=asu;
taout();
1;
pos(2,1);


d konharga()

on(liter[0]);b2=kon(liter[1]);b3=kon(liter[2]);b4=kon(liter[3]);b5=kon(liter[4]);
on(liter[5]);b7=kon(liter[6]);
00000*b1+100000*b2+10000*b3+1000*b4+100*b5+10*b6+b7;

d cetakpass()

0;
hile(a!='E')
{
key();
a=asu;
k[i]=asu;i++;
asu='*';
dataout();
}

d kirimkom()

;
ile(i<11)

su=k[i];
utChar(asu);
++;

i=0;

ile(asu!='E')
{
//dataout();
delay(100);
asu=k[i];PutChar(asu);i++;
}

d tampilr()

(2,1);
=atarget/1000000+48;dataout();
=((atarget/100000)%10)+48;dataout();
=((atarget/10000)%10)+48;dataout();
=((atarget/1000)%10)+48;dataout();
=((atarget/100)%10)+48;dataout();
=((atarget/10)%10)+48;dataout();
=(atarget%10)+48;dataout();
=' ';dataout();
ta=atarget*100/4500;
=Ndata/10000+48;dataout();
=((Ndata/1000)%10)+48;dataout();
=((Ndata/100)%10)+48;dataout();
='.';dataout();
=((Ndata/10)%10)+48;dataout();
=(Ndata%10)+48;dataout();

d phitung()

rget=0;a=0;
le((atarget<=r)&&(a!='E'))

le((tekan==1)&&(a!='E')) {pompa=0;a=Tombolnya();}
pa=1;delay(2);atarget++;tampilr();

rget--;tampilr();
pa=0;
a=='E'){PutChar('i');
        asu=atarget/1000000+48;PutChar(asu);
        asu=((atarget/100000)%10)+48;PutChar(asu);
        asu=((atarget/10000)%10)+48;PutChar(asu);
        asu=((atarget/1000)%10)+48;PutChar(asu);
        asu=((atarget/100)%10)+48;PutChar(asu);
        asu=((atarget/10)%10)+48;PutChar(asu);
        asu=(atarget%10)+48;PutChar(asu);
```

```
      //PutChar('r');
     }
Putchar('j');

------------
rogram Utama
-----------------------------------------------------
main ()
Begin of Main
tser(0xfd);
tlcd();
=0;
0;
i;
mpa=0;
fid=0;tekan=1;
r=1000;
    while(1)//* Pengulangan Loop tanpa henti
    {//* Begin of While
    //busek();
tak(1,1,"   RFID SPBU   ");
f (t==0)

sek();cetak(1,1,"ID: ");konrfid();cetak(2,1,"Pass: ");
takpass();kirimkom();EA=0;
=1;
GetChar();
(a=='s') {busek();cetak(1,1," Salah Masukan  ");}
se if(a=='b')

sek();
=0;cetak(1,1,"Rp.");i=0;
ile(asu!='r') { asu=GetChar();//harga[i]=asu;
                if(asu=='r') ;
                else {dataout();i++;}
              }
tak(2,1,"Lt:");i=0;
ile(asu!='l') { asu=GetChar();liter[i]=asu;
                if(asu=='l') ;
                else {dataout();i++;}
              }
busek();
ile(a!='E') {a=rkey();}
sek();
tak(1,1,"1:Mode Rupiah");
tak(2,1,"2:Mode Liter");
rkey();i=0; busek();
if(a=='1') trl='r';
else if(a=='2') trl='l';
while(a!='E'){ a=rkey();liter[i]=asu;
               if(asu=='E') liter[i]=trl;
               else {dataout();i++;}
             }

             i=0;
while(liter[i]!=trl) { PutChar(liter[i]);i++;
                     }
PutChar(trl);
a=GetChar();
if(a=='t') {busek();cetak(1,1,"Tidak mencukupi");}
else if(a=='y') {busek();i=0;//cetak(1,1,"siap isi");
                 while(asu!='l'){asu=GetChar();liter[i]=asu;
                                 if(asu=='l') ;
                                 else if(asu=='r') {asu=' ';dataout();i++;}
                                 else {dataout();i++;}
                                 if(i==11) {asu='.';dataout();}
                                 }
                 konharga();
                 phitung();
                 }
}
delay(2000);

cs=0;EA=1;busek();


}
    }//* End of While
* End of Main
```

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, DB, ADODB, Menus, ExtCtrls, StdCtrls, DBCtrls, Grids, DBGrids,
  CPort;

type
  TForm1 = class(TForm)
    Notebook1: TNotebook;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    close1: TMenuItem;
    ADOConnection1: TADOConnection;
    ADOQuery1: TADOQuery;
    ADOConnection2: TADOConnection;
    ADOQuery2: TADOQuery;
    lbcipta: TLabel;
    button: TImage;
    GroupBox1: TGroupBox;
    Label1: TLabel;
    edbt: TEdit;
    buton: TButton;
    lbpasscipta: TLabel;
    Label2: TLabel;
    GroupBox2: TGroupBox;
    Label3: TLabel;
    Label4: TLabel;
    Label5: TLabel;
    Label6: TLabel;
    EdtInsertNama: TEdit;
    EdtInsertAlamat: TEdit;
    EdtInsertNoTelp: TEdit;
    EdtInsertAcc: TEdit;
    BtnInsert: TButton;
    BtnCancel: TButton;
    Data1: TMenuItem;
    Insert1: TMenuItem;
    ADOConnection3: TADOConnection;
    ADOQuerytblSPBU: TADOQuery;
    DataSource1: TDataSource;
    DBGrid1: TDBGrid;
    DBNavigator1: TDBNavigator;
    Button1: TButton;
    ComPort1: TComPort;
    Label7: TLabel;
    EdtInsertPassword: TEdit;
    Label8: TLabel;
    EdtInsertRFID: TEdit;
    Edit1: TEdit;
    Edit2: TEdit;
    Edit3: TEdit;
    Edit4: TEdit;
    Edit5: TEdit;
    BtnCek: TButton;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;

    procedure close1Click(Sender: TObject);
```

```
    procedure butonClick(Sender: TObject);
    procedure buttonClick(Sender: TObject);
    procedure FormCreate(Sender: TObject);
    procedure Button1Click(Sender: TObject);
    procedure BtnInsertClick(Sender: TObject);
    procedure BtnCancelClick(Sender: TObject);
    procedure Insert1Click(Sender: TObject);
    procedure ComPort1RxChar(Sender: TObject; Count: Integer);
        function RpToLtr(lIntRp: Integer): real;
    function LtrToRp(lRealLtr:Real):real;
    procedure kirimdata(lstrdata:string);

  private
    { Private declarations }
  public

    { Public declarations }
  end;

var
  Form1: TForm1;
  realltr:real;
  Intj1,Intj2,Intj3,Intj4:integer;
  StrRfid,StrPasswd,StrRp1,StrLtr1,StrRp2,StrLtr2,strkirimdata,StrRp3:string;

implementation

uses Unit5, Unit3;

{$R *.dfm}

procedure TForm1.kirimdata(lstrdata:string);
begin
ComPort1.WriteStr(lstrdata);
sleep(1000);
end;


function TForm1.RpToLtr(lIntRp: Integer): real;
Begin
Result:=lIntRp/4500;
end;

function TForm1.LtrToRp(lRealLtr:Real):real;
begin

Result:= lRealLtr*4500;
end;



procedure TForm1.close1Click(Sender: TObject);
begin
close;
end;

procedure TForm1.butonClick(Sender: TObject);
begin
DM5.g;
end;
```

```
procedure TForm1.buttonClick(Sender: TObject);
begin
DM3.a;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
Notebook1.ActivePage:='utama';
StrRfid:='';
StrPasswd:='';
StrRp1:='';
StrLtr1:='';
StrRp2:='';
StrLtr2:='';
StrRp3:='';
Intj1:=0;
Intj2:=0;
Intj3:=0;
Intj4:=0;
strkirimdata:='';
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
Notebook1.ActivePage:='insert';
end;

procedure TForm1.BtnInsertClick(Sender: TObject);
begin
ADOQuerytblSPBU.Close;
ADOQuerytblSPBU.SQL.Clear;
ADOQuerytblSPBU.SQL.Add('insert into tblSPBU values(:a,:b,:c,:d,:e,:f)');
ADOQuerytblSPBU.Parameters.ParamByName('a').Value:=EdtInsertRFID.Text;
ADOQuerytblSPBU.Parameters.ParamByName('b').Value:=EdtInsertPassword.Text;
ADOQuerytblSPBU.Parameters.ParamByName('c').Value:=EdtInsertNama.Text;
ADOQuerytblSPBU.Parameters.ParamByName('d').Value:=EdtInsertAlamat.Text;
ADOQuerytblSPBU.Parameters.ParamByName('e').Value:=EdtInsertNoTelp.Text;
ADOQuerytblSPBU.Parameters.ParamByName('f').Value:=EdtInsertAcc.Text;
ADOQuerytblSPBU.ExecSQL;

ADOQuerytblSPBU.Close;
ADOQuerytblSPBU.SQL.Clear;
ADOQuerytblSPBU.SQL.Add('select * from tblSPBU');
ADOQuerytblSPBU.Open;

Notebook1.ActivePage:='tblSPBU';
end;

procedure TForm1.BtnCancelClick(Sender: TObject);
begin
Notebook1.ActivePage:='utama';
end;

procedure TForm1.Insert1Click(Sender: TObject);
begin
Notebook1.ActivePage:='tblSPBU';
ADOQuerytblSPBU.Close;
ADOQuerytblSPBU.SQL.Clear;
ADOQuerytblSPBU.SQL.Add('select * from tblSPBU');
ADOQuerytblSPBU.Open;
end;
```

```
procedure TForm1.ComPort1RxChar(Sender: TObject; Count: Integer);
var lstrdtserial,lstrkirimdata,lstrdtserial2,lStr,lstr1:string;
lInti,lIntlengLtr,lInt:integer;
begin
ComPort1.ReadStr(lstrdtserial,Count);
Edit1.Text:=Edit1.Text+lstrdtserial;
//Edit5.Text:=Edit5.Text+IntToStr(Count);

lstrdtserial2:=Edit1.Text;
if (Length(lstrdtserial2)>=11) and (Length(lstrdtserial2)<=16)
and(lstrdtserial2[Length(lstrdtserial2)]='E')and(Intj1=0) then
begin
Intj1:=1;

  for  lInti:=1  to 10  do
  begin
    StrRfid:=StrRfid+lstrdtserial2[lInti];
  end;
    for lInti :=11  to Length(lstrdtserial2)-1  do
  begin
     StrPasswd:=StrPasswd+lstrdtserial2[lInti];
  end;

end;

if Intj4 = 1 then
begin
  if lstrdtserial2='j' then
  begin
    lInt:=StrToInt(StrRp1)-StrToInt(StrRp3);

    ADOQuerytblSPBU.Close;
    ADOQuerytblSPBU.SQL.Clear;
    ADOQuerytblSPBU.SQL.Add('update tblSPBU set account='+IntToStr(lInt)+' where
norfid='+QuotedStr(StrRfid));
    ADOQuerytblSPBU.ExecSQL;

    ADOQuerytblSPBU.Close;
    ADOQuerytblSPBU.SQL.Clear;
    ADOQuerytblSPBU.SQL.Add('select * from tblSPBU');
    ADOQuerytblSPBU.Open;
    Edit1.Text:='';
    FormCreate(Sender);
  end;

  if lstrdtserial2[1]='i' then
  begin
    StrRp3:='';
    for lInti := 2  to length(lstrdtserial2)  do
    begin
      StrRp3:=StrRp3+lstrdtserial2[lInti];
    end;

    lInt:=StrToInt(StrRp1)-StrToInt(StrRp3);



    ADOQuerytblSPBU.Close;
    ADOQuerytblSPBU.SQL.Clear;
    ADOQuerytblSPBU.SQL.Add('update tblSPBU set account='+IntToStr(lInt)+' where
norfid='+QuotedStr(StrRfid));
    ADOQuerytblSPBU.ExecSQL;
```

```
    ADOQuerytblSPBU.Close;
    ADOQuerytblSPBU.SQL.Clear;
    ADOQuerytblSPBU.SQL.Add('select * from tblSPBU');
    ADOQuerytblSPBU.Open;

    Edit1.Text:='';
    FormCreate(Sender);
  end;
end;

if Intj3 = 1 then
begin
  lIntlengLtr:=0;
  if lstrdtserial2[length(lstrdtserial2)]='r' then
  begin
    for lInti := 1  to length(lstrdtserial2)-1  do
    begin
      StrRp2:=StrRp2+lstrdtserial2[lInti];
    end;
    StrLtr2:=FloatToStr(RpToLtr(StrToInt(StrRp2)));
    //Edit4.Text:=StrLtr2;
    lStr:=StrLtr2;
    lstr1:=StrRp2;
    StrRp3:=StrRp2;

    StrLtr2:='';
    for lInti := 1  to length(lStr)  do
    begin
      if lStr[lInti] <>',' then
      begin
        StrLtr2:=StrLtr2+lStr[lInti];
      end
      else
      begin
        lIntlengLtr:=lInti-1;
      end;
    end;

    if lIntlengLtr=1 then
    StrLtr2:='00'+StrLtr2
    else if lIntlengLtr=2 then
    StrLtr2:='0'+StrLtr2
    else if lIntlengLtr=0 then
    begin
      if length(StrLtr2)=1 then
      StrLtr2:='00'+StrLtr2+'00'
      else if length(StrLtr2)=2 then
      StrLtr2:='0'+StrLtr2+'00'
      else if length(StrLtr2)=3 then
      StrLtr2:=StrLtr2+'00';
    end;

    if length(StrRp2)=3 then
    StrRp2:='0000'+StrRp2
    else if length(StrRp2)=4 then
    StrRp2:='000'+StrRp2
    else if length(StrRp2)=5 then
    StrRp2:='00'+StrRp2
    else if length(StrRp2)=6 then
    StrRp2:='0'+StrRp2 ;
```

```
  if StrToInt(lstr1)<=StrToInt(StrRp1) then
  begin
    strkirimdata:='y'+StrRp2+'r'+StrLtr2+'l';
    Edit4.Text:=Edit4.Text+' '+strkirimdata;

    for  lInti:=1  to length(strkirimdata)  do
    begin
      kirimdata(strkirimdata[lInti]);
    end;
    Edit1.Text:='';
    Intj4:=1;
    Intj3:=0;

  end
  else
  begin
    strkirimdata:='t';
    Edit4.Text:=Edit4.Text+' '+strkirimdata;
    kirimdata(strkirimdata);
    FormCreate(Sender);
    Edit1.Text:='';

  end;
end;

if lstrdtserial2[length(lstrdtserial2)]='l' then
begin
  for lInti := 1  to length(lstrdtserial2)-1  do
  begin
    StrLtr2:=StrLtr2+lstrdtserial2[lInti];
  end;
  for lInti := 1  to length(StrLtr2)  do
  begin
    if StrLtr2[lInti] ='.' then
    begin
    StrLtr2[lInti]:=',';
    lIntlengLtr:=lInti-1;
    end;
    StrLtr2[lInti]:=StrLtr2[lInti];
  end;
  StrRp2:=FloatToStr(LtrToRp(StrToFloat(StrLtr2)));
  lStr:=StrLtr2;
  lstr1:=StrRp2;
  StrRp3:=StrRp2;

  StrLtr2:='';
  for lInti := 1  to length(lStr)  do
  begin
    if lStr[lInti] <>',' then
    begin
    StrLtr2:=StrLtr2+lStr[lInti];
    end;
  end;

  if lIntlengLtr=1 then
  StrLtr2:='00'+StrLtr2
  else if lIntlengLtr=2 then
  StrLtr2:='0'+StrLtr2
  else if lIntlengLtr=0 then
  begin
    if length(StrLtr2)=1 then
    StrLtr2:='00'+StrLtr2+'00'
```

Tabel 2-2 Fungsi Pin ID-12 Reader (ASCII/RS232)
Sumber : Data Sheet ID-12

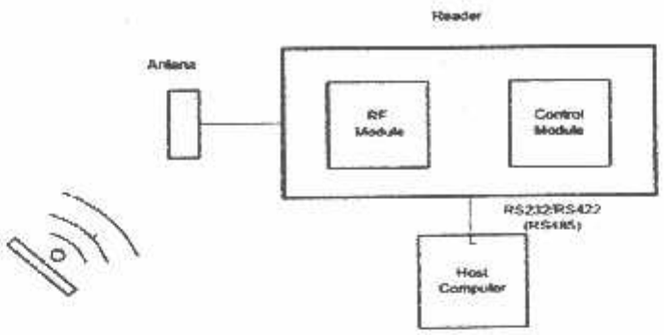
Pin1	Ground 0V	Zero volts and Tuning Capacitor Ground
Pin2	Reset Bar	Strap to +5V
Pin3	Antenna	NC
Pin4	Antenna	NC
Pin5	Strap to Ground	
Pin6	future	
Pin7	+/-	Format selection
Pin8	CMOS	Serial ASCII
Pin9	TTL Data	Serial ASCII inverted
Pin10	BEEP/LED	2.7KHz Logic
Pin11	+4.6 through +5.5V	Supply DC volts

2.1.1.3. *Server Database*

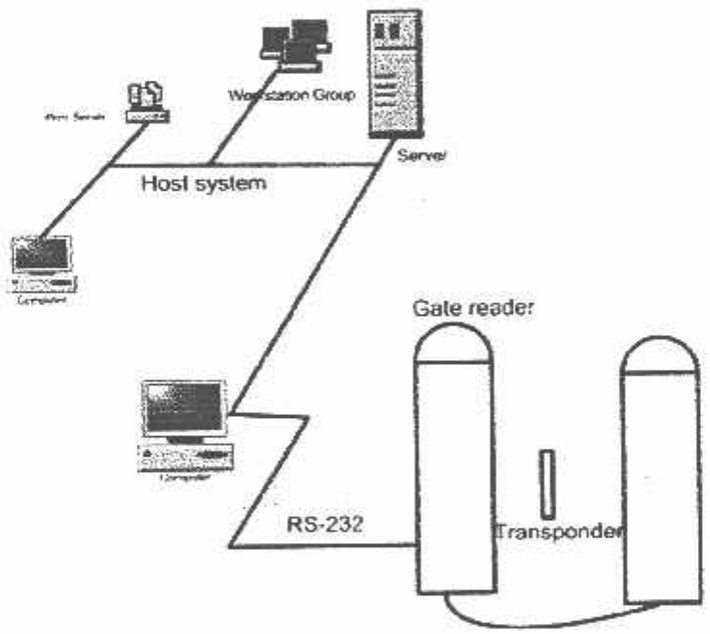
Untuk menyimpan data yang ada pada *tag* digunakan *server database*.

2.1.2. Cara Kerja *RFID*

Telah dijelaskan bahwa *tag* ada yang memiliki sumber listrik sendiri dan ada yang tidak. Cara kerja untuk *tag* yang tidak memiliki energi antenalah yang mengambil tenaga dari *reader* akan memodulasi medan magnet untuk berkomunikasi mengirim data ke *reader*. Data yang diterima *reader* akan diteruskan menuju *host* komputer atau *server database*. Data yang masuk pada *host* komputer akan diolah sesuai dengan program aplikasi yang ada di komputer.



Gambar 2.3 Bagan Rangkaian *Reader*
Sumber : *RFID – its Applications and Benefits*, Philips, 2004



Gambar 2.4 Bagan Rangkaian *RFID*
Sumber : *RFID – its Applications and Benefits*, Philips, 2004

Reader yang digunakan oleh *RFID* memiliki bagian antena yang berfungsi untuk menyalurkan frekuensi. RF *module* yang mengatur frekuensi dan *control module* memproses data. Pada Gambar 1.2. dan Gambar 1.3. terlihat bagan rangkaian *reader* dan bagan rangkaian sistem *RFID*.

2.1.2.1. Frekuensi *RFID*

RFID beroperasi pada frekuensi *Industrial-Scientific-Medical (ISM) band*, bebas untuk daya yang rendah dan sistem jarak pendek. *Band* ini ditentukan oleh *International Telecommunication Union (ITU)[6].*

Tabel 2-3 Daftar frekuensi yang digunakan pada *smart label*
sumber : *passive tags Atmel www RFID*

Frekuensi	Klasifikasi
5.8 GHz *Europe toll standard*	*Very High Frequency*
2.45 GHz 900 MHz *US toll standard*	
13.56 MHz	*High Frequency*
Smart cards	
Smart labels	
125 134 kHz *LF/passive tags* *Livestock. auto anti-theif*	*Low Frequency*

2.1.2.2. Pembacaan Format *RFID*

RFID reader mempunyai banyak sekali tipe, antara lain: ID-10, ID-19, EM-13, dll. biasanya RFID reader ini memiliki dua bentuk output serial yaitu: ASCII dan Wiegand 26-bit. Yang paling banyak digunakan adalah output dengan format ASCII, karena output ini sangat mudah untuk dihubungkan pada mikrokontroler atau PC menggunakan komunikasi serial UART.

Format Data ASCII

Output yang memiliki format ASCII memiliki struktur sebagai berikut:

02	10 data karakter ASCII	checksum	CR	LF	03

Gambar 2.5 Format Data *ASCII*
Sumber : Data Sheet ID-12

Checksum merupakan hasil EXOR (Exclusive OR) dari 5 biner data byte. Untuk lebih jelasnya tentang cara pembacaan format ASCII, lihat contoh berikut. Misalnya data output serial (dalam hexadesimal) yang kita tangkap adalah sebagai berikut:

02	30	34	36	32	30	31	44	37	36	43	44	43	0D	0A	03

Langkah pertama adalah merubah semua nilai data diatas menjadi karakter ASCII. Misalnya 30H menjadi karakter "0", 34H menjadi karakter "4", dst. Langkah kedua adalah menyusun data – data tersebut ke dalam Format Data ASCII seperti gb 1. Kemudian ambil 10 data karakter ASCII. Dalam contoh ini berarti data tersebut adalah:

30	34	36	32	30	31	44	37	36	43	Data Heksa
		6	2	0	1	D	7	6	C	Data ASCII

Untuk data dengan warna biru merupakan data untuk jenis – jenis kartu dan tidak digunakan dalam proses konversi, yang akan dipakai disini adalah data yang ke 3 s/d 10. Hasil konversi dari data heksa ke dalam data ASCII adalah "6201D76C". Gabungkan karakter data ASCII menjadi bilangan Hexadesimal, kemudian konversikan bilangan hexadesimal tsb ke dalam desimal. Hasilnya sebagai berikut: 6201D76C H menjadi 1644287852 (ini merupakan nomor kartu sebenarnya yang tertera pada badan kartu tsb). Cara ini hanya berlaku pada kartu yang tidak dienkripsi

2.2. Sistem Mikrokontroller AT89S52

Mikrokrontroller berbeda dengan mikroprosessor karena selain memiliki CPU juga dilengkapi dengan memori dan input-input yang merupakan kelengkapan sistem dalam mikrokomputer dalam keping tunggal *(Single Chip Mikrokomputer)* yang dapat berdiri sendiri.

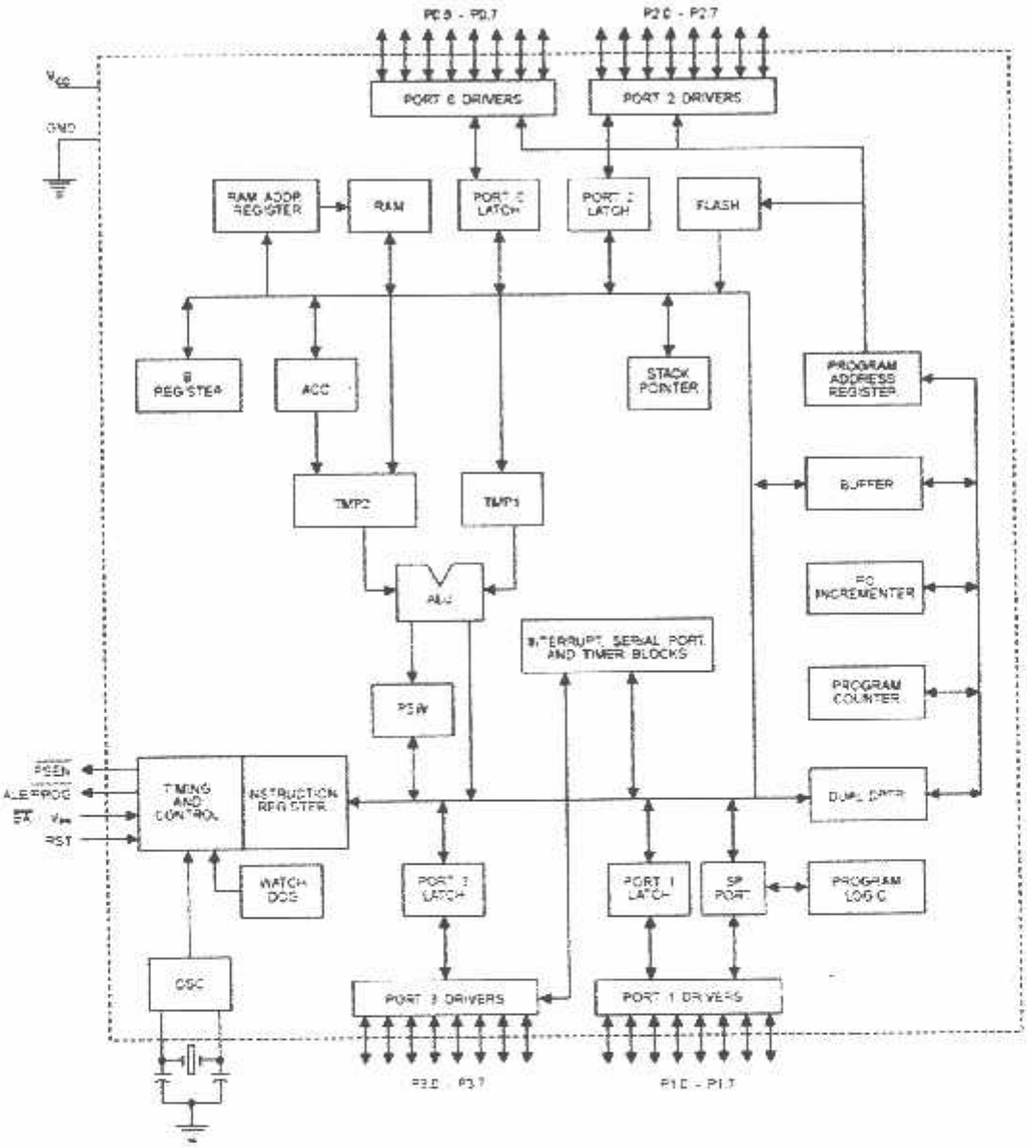
2.2.1. Pendahuluan

Mikrokontroller AT 89S52 merupakan salah satu anggota keluarga dari MCS®51, yaitu suatu komponen produksi ATMEL yang berorientasi kontrol *(microcontroller)*. Intel mengklarifikasikan dalam kelompok *embedded microcontroller*, yang artinya adalah mikrokontroller yang dapat diprogram ulang *(reprogrammable)*. Di dalam *chip* mikrokontroller AT 89S52 ini sudah tersedia berbagai macam peralatan pendukung mikroprosessor seperti RAM, *serial port*, *bus-bus* data dan lainnya yang membuat pemakai *chip* ini dapat menekan penambahan komponen pendukung. Spesifikasi perangkat keras dari mikrokontroller AT 89S52 adalah sebagai berikut:

- CPU *(Central Processing Unit)* dengan lebar data 8 *bit*.
- Prosessor *Boolean* untuk operasi logika 1 *bit*.
- Pembangkit *clock internal*.
- 3 buah *timer/counter* 16 *bit*.
- 2 buah saluran interupsi eksternal.
- Jalur I/O dua arah *(bidirectional)* 32 buah.
- Memori program terpisah dari memori data.

- Memori data internal 256 *byte*.
- Alamat memori program eksternal 64 *Kilobyte*.
- Alamat memori data eksternal 64 *Kilobyte*.
- Memori program internal sebesar 8 *Kilobyte*.

Adapun blok diagram dari Mikrokontroller adalah sebagai berikut :

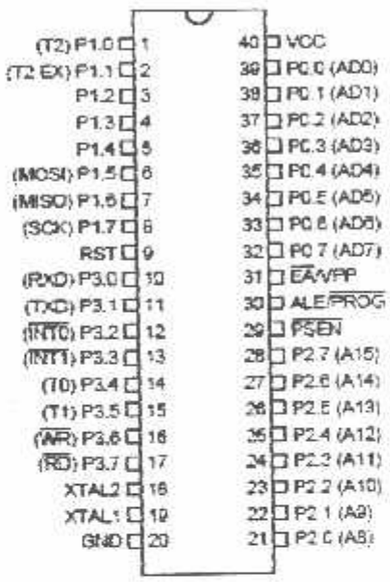


Gambar 2.6 Diagram Blok Mikrokontroller AT 89S52

Sumber: Atmel AT 89S52 *Data Sheet*

2.2.2. Konfigurasi Pin-Pin Mikrokontroller AT 89S52

Mikrokontroller AT 89S52 terdiri dari 40 pin dengan konfigurasi sebagai berikut :



Gambar 2.7 Konfigurasi PIN Mikrokontroller AT 89S52

Sumber: Atmel AT 89S52 *Data Sheet*

Berikut ini adalah penjelasan dari masing-masing pin mikrokontroller AT 89S52:

- Pin 1 sampai 8

  Port 1 : merupakan 8-*bit* saluran masukan atau keluaran dua arah. Setiap saluran mampu melayani 4 masukan.

- Pin 9

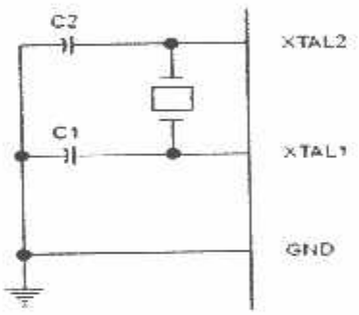
  RST : merupakan masukan *reset*. Logika *high* yang akan membuat mikrokontroller AT 89S52 menjalankan rutin *reset*.

- Pin 10 sampai 17

Port 3 : *port* 3 terdiri dari 8 saluran masukan atau keluaran dua arah. Setiap salurannya mampu melayani 4 masukan. Selain sebagai *port* masukan atau keluaran, *port* 3 juga mempunyai fungsi-fungsi khusus yang dimiliki oleh keluarga MCS®51, fungsi tersebut dapat dilihat dalam Tabel 2-7.

- Pin 18 sampai 19

  X_1 ($XTAL_1$) dan X_2 ($XTAL_2$) : jika dikonfigurasikan bersama sebuah kristal akan membentuk rangkaian *oscilator on-chip* pada mikrokontroller (Atmel AT 89S52 *Data Sheet*).



Gambar 2.8 *Oscilator External* Mikrokontroller AT 89S52

Sumber: Atmel AT 89S52 *Data Sheet*

- Pin 20 sampai 27

  Port 2 : *port* 2 terdiri 8 saluran masukan atau keluaran dua arah. Setiap salurannya mampu melayani 4 masukan. *Port* 2 mengeluarkan alamat bagian tinggi (A8-A15), selama pengambilan instruksi dari memori program eksternal dan pengambilan data dari memori data eksternal yang menggunakan mode pengalamatan 16-*bit* (dengan perintah MOVX @DPTR).

- Pin 29

  $\overline{PSEN}$: *Program Store Enable* merupakan sinyal baca yang mengeksekusi memori program eksternal.

- Pin 30

  ALE/$\overline{PROG}$: *Adress Latch Enable* merupakan pulsa yang berfungsi menahan alamat rendah (A0-A7) pada *port* 0, selama dilakukan proses baca atau tulis memori eksternal. Pin ini juga berfungsi sebagai masukan pulsa program ($\overline{PROG}$), selama dilakukan pemrograman pada EEPROM eksternal.

- Pin 31

  $\overline{EA}$/VPP : *External Access* $\overline{EA}$ dihubungkan dengan V_{SS} untuk memungkinkan pengambilan instruksi pada memori program eksternal yang berlokasi 0000_H sampai $FFFF_H$. Jika diinginkan menggunakan memori program eksternal, maka $\overline{EA}$ dihubungkan *ground*.

- Pin 32 sampai 39

  Port 0 : *port* 0 terdiri dari 8 saluran masukan atau keluaran dua arah. Setiap saluran mampu melayani 8 masukan. *Port* 0 merupakan saluran alamat bagian rendah (A0-A7), yang dimultipleks dengan bus data (D0-D7), yang digunakan pada saat mengakses memori data eksternal dan memori program eksternal.

- Pin 40

  V_{CC} : merupakan masukan catu daya 5 volt, dengan toleransi kurang lebih 10%.

Keluarga MCS®51 yang diproduksi Intel mempunyai konfigurasi yang berbeda-beda sesuai dengan jenisnya. Masing-masing jenis saling kompatibel serta mempunyai kelebihan tersendiri. Misalnya mikrokontroller 8051. Tabel 2-4 memperlihatkan sebagian dari keluarga MCS®51.

Tabel 2-4 Keluarga MCS®51

Sumber : Bereksperimen dengan Mikrokontroller 8031,

Tipe	Tipe tanpa EPROM	Tipe ber-EPROM	Kapasitas ROM	Kapasitas RAM	Port I/O	Pewaktu
8051	8031	-	4K	128	4	2
8051AH	8031AH	8751H	4K	128	4	2
8052AH	8032AH	8752BH	8K	256	4	3
80C51BH	80C31BH	87C51	4K	128	4	2
80C52	80C32	-	8K	256	4	3
83C51FA	80C51BH	8751FA	8K	256	4	3
83C51FB	80C51FB	87C51FB	16K	256	4	3
83C152	80C152	-	8K	256	5	3
89S52	-	89S52	8K	256	4	3

2.2.3. Organisasi Memori

Organisasi memori pada mikrokontroller AT 89S52 dapat dibagi menjadi dua bagian besar yaitu memori program dan memori data. Pembagian tersebut

didasarkan atas fungsi dari penyimpanan data maupun program. Memori program digunakan untuk menyimpan instruksi-instruksi yang akan dijalankan oleh mikrokontroller, sedangkan memori data digunakan sebagai tempat penyimpanan data yang sedang diolah mikrokontroller.

Program mikrokontroller disimpan dalam memori program berupa ROM. Mikrokontroller AT 89S52 dilengkapi dengan ROM internal namun untuk program yang besar digunakan ROM eksternal yang terpisah dari mikrokontroller. Untuk dapat menggunakan memori program eksternal ini penyemat / $\overline{EA}$ dihubungkan dengan penyemat Vss (logika 0).

Memori program mikrokontroller menggunakan alamat 16 bit mulai 0000_H-$FFFF_H$, sehingga kapasitas penyimpanan program maksimal adalah 2^{16} *byte* atau 64 K*byte*. Sinyal yang digunakan untuk membaca memori program eksternal adalah sinyal $\overline{PSEN}$ *(Program Store Enable)*.

Selain memori program mikrokontroller AT 89S52 juga memiliki memori data internal berkapasitas 256 *byte* dan mampu mengakses memori data eksternal sebesar 64 K*byte*. Semua memori data internal dapat dialamati dengan pengalamatan langsung atau tidak langsung. Ciri dari pengalamatan langsung adalah *operand* berisi alamat data yang diolah. Sedangkan ciri dari pengalamatan tidak langsung adalah *operand* alamat *register* yang berisi alamat data yang akan diolah. Untuk membaca data digunakan sinyal / $\overline{RD}$, sedangkan untuk menulis data digunakan sinyal / $\overline{WR}$.

2.2.4. *Register* Fungsi Khusus

Register fungsi khusus *(Special Function Register)* terletak pada 128 *byte* bagian atas memori data internal dan berisi *register-register* untuk pelayanan *latch port, timer, program status words, control peripheral* dan sebagainya. Alamat *register* fungsi khusus ditunjukkan pada Tabel 2-5.

Register-register ini hanya dapat diakses dengan pengalamatan langsung. Enam belas alamat pada *register* fungsi khusus dapat dialamati per*bit* maupun per-*byte* dan terletak pada alamat 80_H-FF_H. Secara perangkat keras, *register* fungsi khusus ini dibedakan dengan memori data internal.

Tabel 2-5 Nama dan Alamat *Register* pada *Register* Fungsi Khusus

Sumber : Bereksperimen dengan Mikrokontroller 8031

Simbol	Nama *Register*	Nilai pada saat *reset*	Alamat
Acc	*Accumulator*	00_H	$0E0_H$
B	*Register* B	00_H	$0F0_H$
PSW	*Program Status Word*	00_H	$0D0_H$
SP	*Stack Pointer*	07_H	81_H
DPTR	Data *Pointer* 2 *bytes*		
DPL	*Low bytes*	0000_H	82_H
DPH	*High bytes*	0000_H	83_H
P0	*Port* 0	FF_H	80_H
P1	*Port* 1	FF_H	90_H
P2	*Port* 2	FF_H	$0A0_H$
P3	*Port* 3	FF_H	$0B0_H$

IP	*Interrupt Priority control*	$XXX00000_B$	$0B8_H$
IE	*Interrupt Enable Control*	$0XX00000_B$	$0A8_H$
TMOD	*Timer/Counter Mode Control*	00_H	89_H
TCON	*Timer/Counter Control*	00_H	88_H
TH0	*Timer/Counter 0 high byte*	00_H	$8C_H$
TL0	*Timer/Counter 0 low byte*	00_H	$8A_H$
TH1	*Timer/Counter 1 high byte*	00_H	$8D_H$
TL1	*Timer/Counter 1 low byte*	00_H	$8B_H$
SCON	*Serial Control*	00_H	98_H
SBUF	*Serial Data Buffer*	Independen	99_H
PCON	*Power Control*	HMOS $0XXXXXXX_B$	87_H
		CHMOS $0XXX0000_B$	

Beberapa macam *register* fungsi khusus yang sering digunakan, dijelaskan sebagai berikut :

- *Accumulator* (ACC) merupakan *register* untuk penambahan dan pengurangan. Perintah *Mnemonic* untuk mengakses akumulator disederhanakan sebagai A.
- *Register* B merupakan *register* khusus yang berfungsi melayani operasi perkalian dan pembagian.
- *Program Status Word* (PSW) terdiri dari beberapa *bit* status yang menggambarkan kejadian di akumulator sebelumnya. Yaitu *carry bit*,

auxiliary carry, dua *bit* pemilih bank, bendera *overflow*, *parity bit*, dan dua bendera yang dapat didefenisikan sendiri oleh pemakai.

- *Stack Pointer* (SP) merupakan *register* 8 *bit* yang dapat diletakkan di alamat manapun pada RAM internal. Isi *register* ini ditambah sebelum data disimpan, selama instruksi *PUSH* dan *CALL*. Pada saat *reset*, *register* SP diinisialisasi pada alamat 07_H, sehingga *stack* akan dimulai pada lokasi 08_H.
- *Data Pointer* (DPTR) terdiri dari dua *register*, yaitu *byte* tinggi (Data *Pointer High*, DPH) dan *byte* rendah (*Data Pointer Low*, DPL) yang berfungsi untuk pengalamatan alamat 16 *bit*.
- *Port* 0 sampai *port* 3 merupakan *register* yang berfungsi untuk membaca dan mengeluarkan data pada *port* 0, 1, 2, 3. Masing-masing *register* ini dapat dialamati per-*byte* maupun per-*bit*.
- *Serial Data Buffer* (SBUF) merupakan data dua *register* yang terpisah, *register buffer* pengirim dan sebuah *register buffer* penerima. Meletakkan data pada SBUF berarti meletakkan pada *buffer* pengirim yang akan mengirimkan data melalui transmisi *serial*. Membaca data SBUF berarti menerima data dari *buffer* penerima.
- *Control Register* terdiri dari *register* yang mempunyai fungsi kontrol. Untuk mengontrol sistem interupsi, terdapat dua *register* khusus, yaitu *register* IP *(Interrupt Priority)* dan *register* IE *(Interrupt Enable)*. Untuk mengontrol pelayanan *timer/counter* terdapat *register* khusus, yaitu

register TCON *(Timer/Counter Control)* serta untuk pelayanan *port serial* menggunakan *register* SCON *(Serial Port Control)*.

2.2.5. *Port* Masukan dan Keluaran

Mikrokontroller AT 89S52 mempunyai 4 *port* dan masing-masing *port* terdiri dari 8 saluran *bit*. Ke empat *port* ini bersifat *bidirectional* yaitu dapat digunakan sebagai masukan dan keluaran.

Port 0 digunakan sebagai saluran data yang dimultipleks dengan saluran alamat rendah untuk mengakses memori eksternal, baik memori program maupun memori data. *Port* 2 mengeluarkan bagian alamat tinggi untuk mode pengalamatan 16 *bit*. *Port* 1 dan 3 berfungsi sebagai saluran masukan dan keluaran multi fungsi. Jika dibutuhkan *port* 3 mempunyai fungsi khusus seperti ditunjukkan pada Tabel 2-6.

Tabel 2-6 Fungsi Khusus *Port* 3
Sumber: Bereksperimen dengan Mikrokontroller 8031

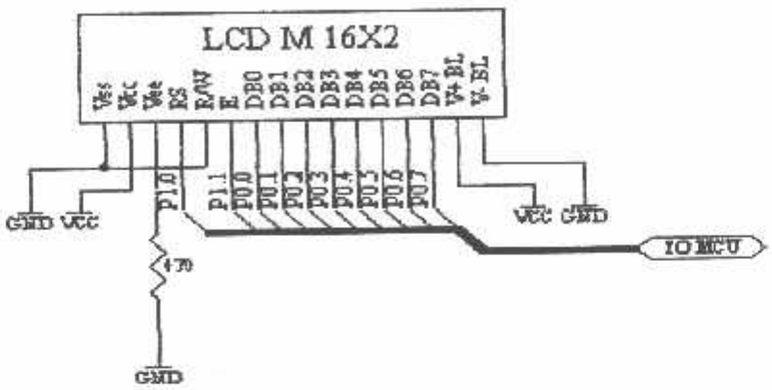
Nama Penyemat	Fungsi Khusus
Port 3.0	RxD (*port* masukan *serial*)
Port 3.1	TxD (*port* keluaran *serial*)
Port 3.2	/INT0 (masukan interupsi eksternal 0)
Port 3.3	/INT1 (masukan interupsi eksternal 1)
Port 3.4	T0 (masukan pewaktu eksternal 0)
Port 3.5	T1 (masukan pewaktu eksternal 1)
Port 3.6	/WR (sinyal tulis memori data eksternal)

Port 3.7	/RD (sinyal baca memori data eksternal)

2.2.6. Sistem Interupsi

Mikrokontroller AT 89S52 mempunyai dua sumber interupsi eksternal dan sumber interupsi internal yang dapat diprogram agar sensitif terhadap perubahan level atau transisi. Interupsi *timer* aktif saat *register timer* yang bersangkutan mengalami *rollover*, interupsi *serial* akan aktif pada saat mikrokontroller mengirim atau menerima data. Setiap sumber interupsi dapat diaktifkan atau dimatikan melalui perangkat lunak. Interupsi yang mempunyai tingkatan prioritas lebih tinggi tidak dapat diinterupsi oleh yang lebih rendah. Meskipun demikian melalui perangkat lunak dapat diubah, yaitu dalam *register interrupt priority* (IP).

2.3. LCD 16x2 (M1632)



Gambar 2.9 Rangkaian LCD
Sumber : *LCD Modul User Manual*

LCD (*Liquid Crystal Display*) adalah komponen *display* yang tidak memancar (*nonemissive*), sehingga tidak menghasilkan sumber cahaya seperti CRT (*Cathode Ray Tube*), dan berdaya sangat rendah (lebih rendah dari LED) yaitu dalam hitungan mikrowatt (LED dalam hitungan miliwatt). LCD menahan atau membiarkan cahaya yang dipantulkan dari sunber cahaya luar dan cahaya yang berasal dari belakang atau samping yang melewatinya. LCD dikontrol oleh ROM/RAM generator karakter dan RAM data display. Semua fungsi display dikontrol dengan instruksi dan LCD dapat dengan mudah diinterfacekan dengan MPU (*Mikroprosessor Unit*).

Karakteristik dari LCD dot-matriks adalah sebagai berikut:

- 16X2 karakter dengan 5X7 dot matriks+kursor
- ROM generator karakter dengan 8 tipe karakter (untuk program write)
- 80X8 bit RAM data display
- Dapat diinterfacekan dengan 4 atau 8 bit MPU

- RAM data dan RAM generator karakter dapat dibaca dari MPU
- +5V single power supply
- Power-on reset
- Range temperature operasi 0-60°C
- Beberapa fungsi instruksi:

  Display clear, Cursor home, Display ON/OFF, Cursor ON/OFF, Display charackter blink, Cursor Shift dan Display shift.

LCD disini dapat menampilkan karakter yang ada pada ROM generator karakter, yang sudah berisi 192 jenis karakter, dengan cara memberikan kode karakter untuk tiap-tiap karakter yang diinginkan pada pada bus data dengan menggunakan sinyal kontrol.

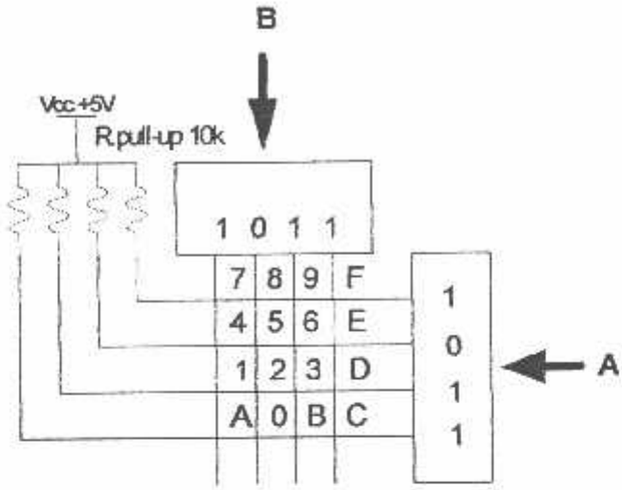
Tabel 2-7 Pin LCD
Sumber : *LCD Module User Manual*

NO	Simbol	Level	Function
1	Vss	-	0V (GND)
2	Vcc	-	5V
3	Vee	-	LCD Drive
4	RS	H/L	H: Data Input L: Instruktion In
5	R/W	H/L	H: Read L: Write
6	E	1/0	Enable Signal
7	DB0	H/L	Data Bus
8	DB1	H/L	Data Bus

9	DB2	H/L	Data Bus
10	DB3	H/L	Data Bus
11	DB4	H/L	Data Bus
12	DB5	H/L	Data Bus
13	DB6	H/L	Data Bus
14	DB7	H/L	Data Bus
15	V+ BL	-	Vcc
16	V- BL	-	GND

2.4. Matriks Keypad

Keypad disini digunakan untuk memasukkan data acuan, proses *scanning* matrik keypad pada dasarnya *mendecoder* penekanan suatu tombol dengan *konfigurasi matrik*. Diumpamakan Port B dioperasikan sebagai *Output scanning* bagian kolom. *Scanning* dilakukan secara berurutan dari kolom paling kiri sampai kolom paling kanan, Kolom yang aktif akan berada pada kondisi low. Untuk mengetahui ada tidaknya tombol ditekan, Maka harus dilakukan pembacaan terhadap port (Diumpamakan port A) yang dioperasikan sebagai input dari setiap baris pada kolom yang sedang aktif. Jika tidak ada tombol ditekan semua kondisi baris akan high karena karena *dipull-up oleh resistor pull-up ke Vcc*.Jika salah satu baris tombol ditekan maka tombol tersebut terletak pada kolom yang sedang aktif, Kondisi baris yang terbaca pada port A adalah low. Hal ini dapat dijelaskan pada gambar berikut :



Gambar 2.10 Proses scanning Keypad matriks 4X4
Sumber : Perencanaan

Diumpamakan pin Y1-Y4 sebagai *row* dan pin X1-X4 sebagai *colom*. yaitu apabila salah satu pin misalnya Y1 terhubung dengan X1 maka data outnya akan 0 sedangkan untuk Y1 dengan X2 data outnya akan 1. Diumpamakan tombol 5 ditekan, Maka pada proses kerjanya port B dikirim data 0111. Dalam keadaan ini dilakukan pembacaan pada port A dan hasil yang diperoleh adalah 1111. ini berarti tidak ada tombol yang ditekan pada kolom paling kiri. Selanjutnya pada port B dikirim data 1011 dan dilakukan pembacaan dari port A. Hasil diperoleh dari pembacaan tersebut yaitu 1011. Ini berarti ada tombol yang ditekan pada kolom kedua baris kedua. Melalui *software* yang dibuat dapat diketahui kode dari tombol yang ditekan. Untuk menghindari pembacaan yang salah karena adanya dua tombol atau atau lebih yang ditekan bersamaan, maka proses scanning dilakukan terhadap seluruh tombol *keypad* dan penghitungan jumlah tombol yang ditekan.

Tabel 2 – 8 Tabel kebenaran keypad 3 x 4
Sumber : Perencanaan

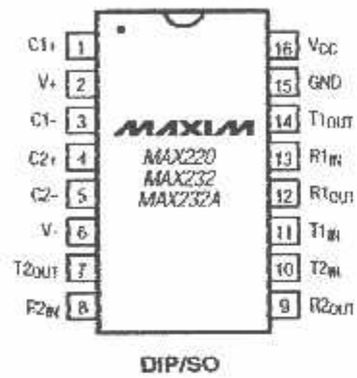
Posisi Swicth	D	C	B	A	Desimal
Y1,X1	0	0	0	0	0
Y1,X2	0	0	0	1	1
Y1,X3	0	0	1	0	2
Y1,X4	0	0	1	1	3
Y2,X1	0	1	0	0	4
Y2,X2	0	1	0	1	5
Y2,X3	0	1	1	0	6
Y2,X4	0	1	1	1	7
Y3,X1	1	0	0	0	8
Y3,X2	1	0	0	1	9
Y3,X3	1	0	1	0	*
Y3,X4	1	0	1	1	#

2.5. IC MAX 232

Saluran data pada *port* seri *PC* menggunakan standard RS232, dimana logic 0 (*low*) dinyatakan sebagai tegangan antara +3 *Volt* sampai +10 *Volt* dan logic 1 (*high*) dinyatakan sebagai tegangan antara -3 *Volt* sampai -10 *Volt*. Level tegangan ini tidak sesuai dengan level tegangan yang dipakai pada *port* seri AT89C51 yang menggunakan standard *TTL (Transistor Transistor Logic)*, yaitu level tegangan baku dalam rangkaian – rangkaian digital.

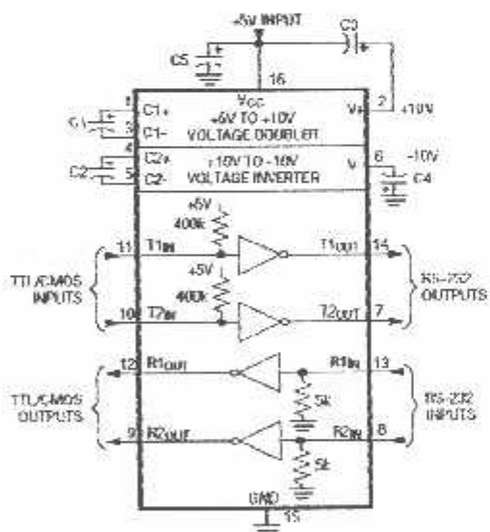
Dalam standar *TTL* logic 0 (*low*) dinyatakan sebagai tegangan antara 0 *Volt* sampai 0.8 *Volt*, dan logic 1 (*high*) dinyatakan sebagai tegangan antara 3.5 *Volt* sampai 5 *Volt*. Karena perbedaan tegangan tersebut, agar *port* seri PC tidak

merusak *port* seri AT89C51 antara keduanya dipasangkan IC MAX232 sebagai penyesuai tegangan.



Gambar 2.11. Konfigurasi Pin IC MAX232
Sumber : *Maxim Data Sheet*

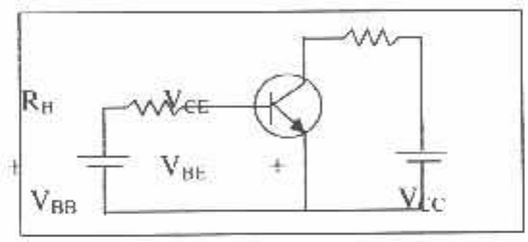
Rangkaian dasar dari MAX 232 dapat dilihat pada gambar berikut ini:



Gambar 2.12. Rangkaian Operasi MAX 232
Sumber : *Maxim Data Sheet*

2.6. Transistor sebagai Driver/saklar

Transistor merupakan salah satu dari komponen aktif yang banyak sekali digunakan. Beberapa fungsi transistor yaitu dapat digunakan sebagai rangkaian *Driver* ataupun sebagai saklar. Rangkaian *Driver* merupakan suatu rangkaian yang digunakan untuk menggerakkan peralatan lain yang membutuhkan arus atau tegangan yang lebih besar. Dalam rangkaian *driver* ini kaki basis transistor dikontrol dengan memberi pulsa rendah dan pulsa tinggi [14]. Rangkaian *driver* ditunjukkan pada gambar berikut ini :



Gambar 2.13 Rangkaian Driver
Sumber: Prinsip-prinsip elektronika

Pada gambar terlihat V_{BB} akan memberi tegangan maju kepada dioda emitor melalui resistor R_B. Untuk transistor *silicon* V_{BE} berkisar antara 0,6-0,7 Volt. Arus basis dan kolektor dapat dicari dengan persamaan dibawah ini :

$$V_{BB} - I_B \cdot R_B - V_{BE} = 0$$

$$I_B = \frac{V_{BB} - V_{BE}}{R_B} (Ampere) \quad \text{.....(14-1)}$$

Dimana : V_{BB} = Tegangan bias Basis (Volt)

V_{BE} = Tegangan Basis Emitor (Volt)

R_B = Resistor pada Basis (Ohm)

I_B = Arus Basis (Ampere)

Sedangkan arus kolektor didapat dengan persamaan dibawah ini :

$I_C = \beta . I_B$ (Ampere).....(14-2)

Dimana : I_C = Arus Kolector (Ampere)

β – Beta

Arus Kolektor diatas akan menimbulkan tegangan sebesar I_C x R_C pada resistor. Karena itu tegangan emitor menjadi :

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

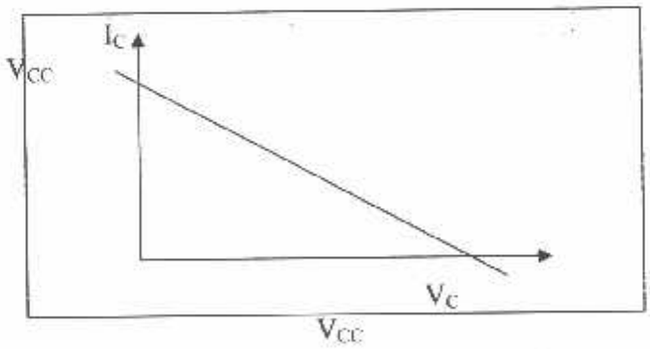
$V_{CE} = V_{CC} - I_C . R_C$ (Volt).....(14-3)

Dimana : V_{CE} = Tegangan Kolektor Emitor (Volt)

V_{CC} = Tegangan sumber (Volt)

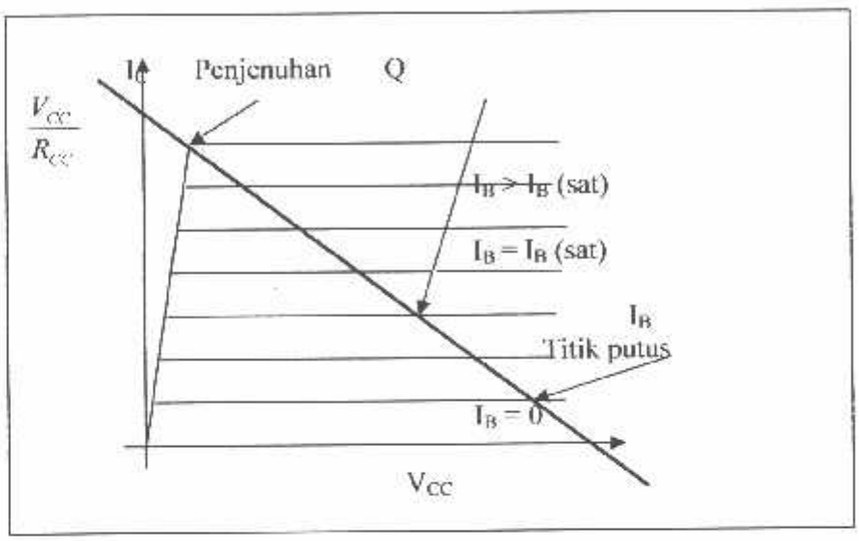
R_C = Resistor pada Kolektor (Ohm)

Hubungan antara I_C dan V_{CE} pada garis beban DC dari transistor pada hambatan beban tertentu. Gambar 2-11 garis beban DC tersebut dapat dilukiskan sebagai berikut :



Gambar 2.14 Garis Beban DC
Sumber: Prinsip-prinsip elektronika

Garis Beban DC tersebut mempunyai titik potong dengan sumbu tegak pada $I_C = V_{CC}/R_C$ untuk $V_{CE} = 0$ dan titik potong dengan sumbu datar pada $V_{CE}=V_{CC}$ untuk $I_C = 0$



Gambar 2.15 Titik jenuh dan titik putus pada gari beban DC
Sumber: Prinsip-prinsip elektronika

Pada gambar 2-12 tampak bahwa titk potong antara garis beban DC dan kurva I_B =0 dikenal sebagai titik *Cut off*. Pada keadaan ini $V_{CE} = V_{CC}$. Titik jenuh adalah titik potong kurva I_B pada ujung teratas pada garis beban DC. Pada keadaan ini I_C dan V_{CE} dalam keadaan jenuh. I_C pada titik ini diperoleh :

$$I_{C(sat)} = \frac{V_{CC} - V_{CE(sat)}}{R_C} (Ampere) \quad \text{.....(14-4)}$$

Sedangkan titik Q adalah semua titik potong kurva I_B yang terletak diantara titik *Cut off* dan titik jenuh merupakan daerah aktif transistor[14].

2.7. RELAY

Relay disebut juga saklar magnet, yang dapat mengontrol (*on* atau *off*) peralatan dari jarak jauh. *Relay* ini tersusun dari sebuah kumparan dengan inti besi lunak dan plat (Wasito S, 1996:145). Prinsip kerja *Relay* adalah apabila terdapat arus yang menuju kumparan maka akan timbul suatu medan magnet. Medan magnet tersebut selanjutnya membuat inti besi menjadi magnet. Inti besi menarik pegas kontak menyebabkan posisi saklar berubah. Apabila tegangan pada kumparan tersebut dihilangkan maka pada kumparan tidak terjadi induksi magnetik yang menyebabkan kontak kembali ke posisi awal. (Wasito S.,1996: 145).

Relay dapat dibagi menjadi beberapa jenis yaitu :

1. SPST (*Single-Pole Single-Throw Switch*)
2. SPDT (*Single-Pole Double-Throw Switch*)
3. DPST (*Double-Pole Single-Throw Switch*)
4. DPDT (*Double-Pole Double-Throw Switch*)

Gambar 2.16 jenis-jenis *relay*
Sumber: Vademekum Elektronika

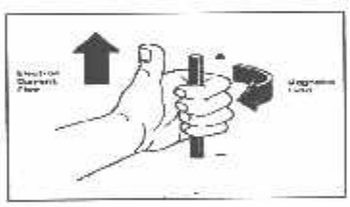
Relay dibedakan menjadi dua berdasarkan kondisi awal saat *relay* tidak bekerja, yaitu *relay normally open* (NO) dan *relay normally closed* (NC). *Relay*

disebut *normally open* yaitu jika kumparan *relay* diberi tegangan maka *relay* akan bekerja dan akan menutup kontak-kontaknya, sehingga kontak NO akan menutup. Sedangkan *relay* disebut *normally closed* jika kumparan *relay* diberi tegangan maka *relay* bekerja dan membuka kontak-kontaknya, sehingga kontak NC akan membuka.

2.8 Motor DC

2.8.1 Teori Dasar Motor DC

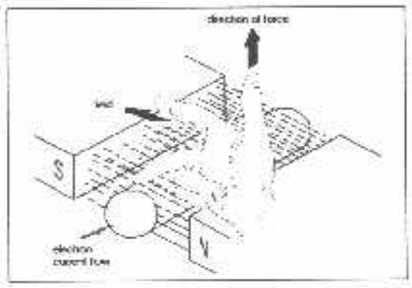
Setiap arus yang mengalir melalui sebuah konduktor akan menimbulkan medan magnet. Arah medan magnet dapat ditentukan dengan kaidah tangan kiri. Ibu jari tangan menunjukkan arah aliran arus listrik sedangkan jari-jari yang lain menunjukkan arah medan magnet yang timbul, seperti yang ditunjukkan oleh gambar 2.17 berikut ini:



Gambar 2.17 Kaidah Tangan Kiri
Sumber: Fisika

Jika suatu konduktor yang dialiri arus listrik ditempatkan dalam sebuah medan magnet, kombinasi medan magnet akan ditunjukkan oleh gambar 2.18. Arah aliran arus listrik dalam konduktor ditunjukkan dengan tanda "x" atau ".". Tanda "x" menunjukkan arah arus listrik mengalir menjauhi pembaca gambar, tanda "." menunjukkan arah arus listrik mengalir mendekati pembaca gambar.

Sebuah cara lagi untuk menunjukkan hubungan antara arus listrik yang mengalir didalam sebuah konduktor, medan magnet dan arah gerak, adalah kaidah tangan kanan untuk motor seperti yang diperlihatkan pada gambar 2.18 berikut :



Gambar 2.18 Kaidah Tangan Kanan Untuk Motor
Sumber: Elektronika dalam Industri

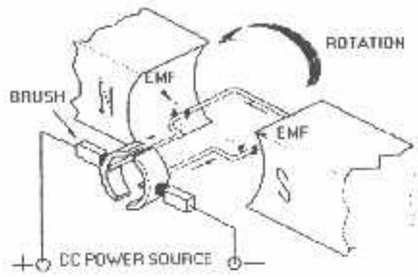
Kaidah tangan kanan untuk motor menunjukkan arah arus yang mengalir didalam sebuah konduktor yang berada dalam medan magnet. Jari tengah menunjukkan arah arus yang mengalir pada konduktor, jari telunjuk menunjukkan arah medan magnet dan ibu jari menunjukkan arah gaya putar. Adapun besarnya gaya yang bekerja pada konduktor tersebut dapat dirumuskan dengan :

F = B . I . L (Newton)

Dimana : B = kerapatan fluks magnet (weber)

L = panjang konduktor (meter)

I = arus listrik (ampere)



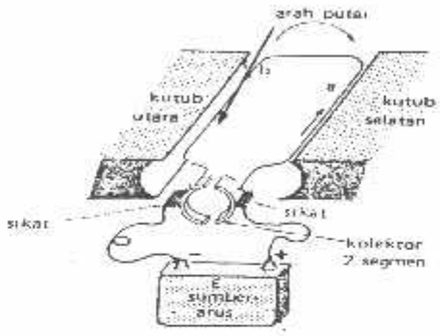
Gambar 2.19 Konstruksi Dasar Motor DC
Sumber: Elektronika dalam Industri

Pada gambar 2.19 diatas tampak sebuah konstruksi dasar motor DC, pada gambar diatas terlihat bahwa pada saat terminal motor diberi tegangan dc, maka arus elektron akan mengalir melalui konduktor dari terminal negatif menuju ke terminal positif. Karena konduktor berada diantara medan magnet, maka akan timbul medan magnet juga pada konduktor yang arahnya seperti terlihat pada gambar 2.19 diatas. Arah garis gaya medan magnet yang dihasilkan oleh magnet permanen adalah dari kutub utara menuju ke selatan. Sementara pada konduktor yang dekat dengan kutub selatan, arah garis gaya magnet disisi sebelah bawah searah dengan garis gaya magnet permanen sedangkan di sisi sebelah atas arah garis gaya magnet berlawanan arah dengan garis gaya magnet permanen. Ini menyebabkan medan magnet disisi sebelah bawah lebih rapat daripada sisi sebelah atas. Dengan demikian konduktor akan terdorong ke arah atas. Sementara pada konduktor yang dekat dengan kutub utara, arah garis gaya magnet disisi sebelah atas searah dengan garis gaya magnet permanen sedangkan di sisi sebelah bawah arah garis gaya magnet berlawanan arah dengan garis gaya magnet permanen. Ini menyebabkan medan magnet disisi sebelah atas lebih rapat daripada sisi sebelah bawah. Dengan demikian konduktor akan terdorong ke arah

bawah. Pada akhirnya konduktor akan membentuk gerakan berputar berlawanan dengan jarum jam seperti terlihat pada gambar 2.19 diatas.

2.8.2 Cara Kerja Motor DC

Adapun cara kerja motor DC dapat dilihat pada gambar dibawah ini



Gambar 2.20 Dasar Kontruksi Motor DC
Sumber: Elektronika dalam Industri

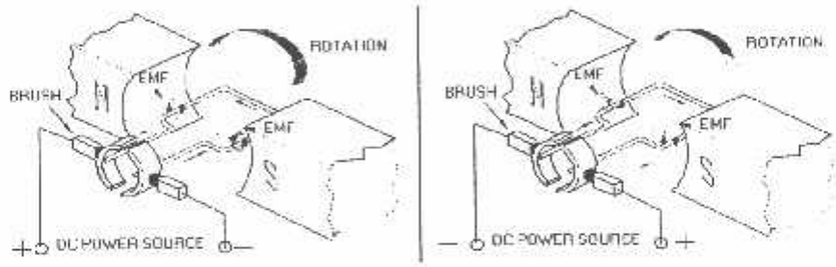
Ada satu lilit kawat a – b berada di dalam medan magnet. Lilitan ini dapat berputar dengan bebas, lilitan ini biasa disebut dengan jangkat (*armour*).

Pada jangkar dimasukkan arus yang berasal dari sumber (baterai) E. koneksi baterai dengan jangkar melalui sikat-sikat. Sikat-sikat ini terpasang pada sebuah cincin yang terbelah dua. yang disebut kolektir. Adapun tujuan dari kontruksi ini adalah agar liliran kawat dapat berputar apabila ada arus listrik yang melewatinya.

Pada kawat yang berada di kanan arus mengalir dari depan ke belakang dalam kawat yang di kiri, arus mengalir dari belakang ke depan kawat a dan b secara berganti-gantian berada di kiri dan kanan. Karena itu arah arus di a dan arah arus di b selalu membolak balik. Pembalikan arah arus itu terjadi pada saat

lilitan kawat melintasi possisi vertikal. Disini kolektor berfungsi bagaikan penyearah mekanik. Flux magnet yang ditimbulkan magnet permanen disebut medan magnetnya motor. Dalam gambar arah fluk magnetik adalah dari kiri ke kanan. Adapun gaya yang bekerja pada penghantar b adalah ke atas, sementara gaya yang bekerja pada penghantar a adalah ke bawah . Gaya-gaya yang bekerja sama kuatnya, jadi ada kopel yang bekerja pada kawat sehingga lilitan pun dapat berputar. Setelah berputar 90^0 arah arus berbalik, pada saat itu penghantar a dan penghantar b bertukar tempat. Akibatnya arah gerak putaran tidak berubah.

2.8.3 Pengendalian Arah Putaran Motor DC



Gambar 2.21 Pengendalian Arah Motor DC
Sumber: Elektronika dalam Industri

Dari gambar 2.21 diatas, agar arah putaran motor DC berubah, maka polaritas tegangan pada terminal motor harus dibalik.

BAB III

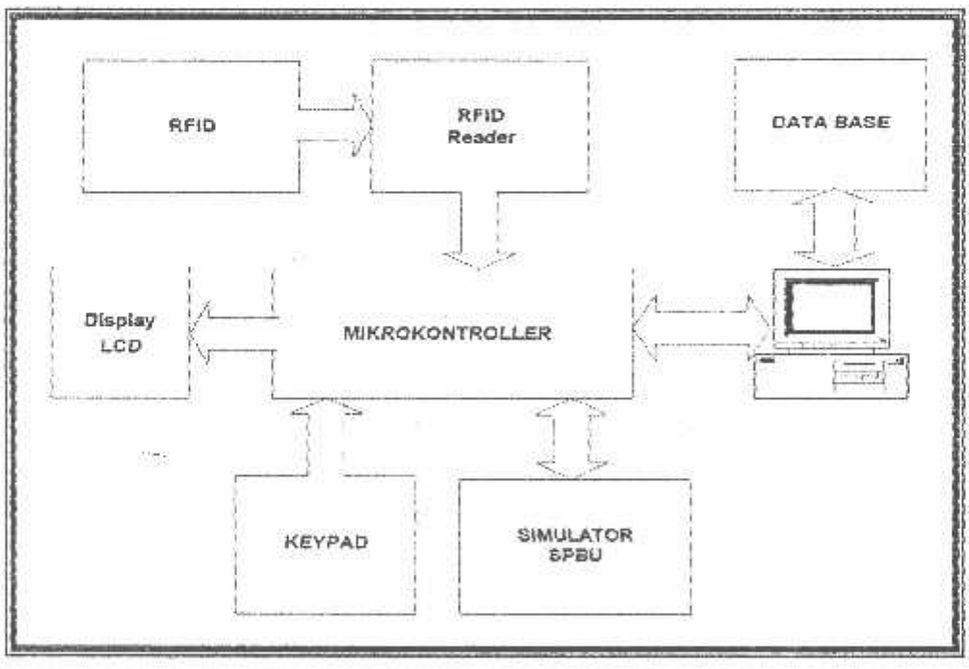
PERANCANGAN HARDWARE DAN SOFTWARE

Bab ini akan membahas tentang perencanaan dan perancangan alat yang meliputi perencanaan perangkat keras (*Hardware*) dan perangkat lunak (*Software*) dari Sistem layanan prabayar pada SPBU menggunakan tags RFID secara otomatis ini. Perancangan secara keseluruhan dapat dibagi menjadi dua bagian, yaitu :

1. Perancangan Hardware
2. Perancangan Software

3.1. PERANCANGAN HARDWARE

Blok diagram sistem sebagai berikut :



Gambar 3.1 Diagram Blok Sistem
Sumber : *Perencanaan dan pembuatan*

Dari gambar blok diagram 3-1 dapat dijelaskan cara kerjanya sebagai berikut:

1. Tag RFID (*Radio Frequensi Identification*)

   Ini adalah devais yang menyimpan informasi untuk identifikasi yang dalam perancangan ini obyek yang digunakan adalah identitas pelanggan . Tag RFID sering juga disebut sebagai *transponder*.

2. Pembaca RFID (*Radio Frequensi Identification*)

   Devais yang kompatibel dengan tag RFID yang akan berkomunikasi secara *wireless* dengan tag. Berikut juga akan membaca dan mengidentifikasi infornasA g.

3. PC sebagai software aplikasi

   Aplikasi pada sebuah workstation atau PC yang digunakan untuk menyimpan *Data Base* pelanggan melalui tag dan pembaca RFID sehingga dapat juga sebagai tempat pengolahan data agar bisa diaplikasikan oleh mikrokontroller pada tampilan maupun ke simulasi alat.

4. Mikrokontroller

   Mengontrol simulator dan juga berfungsi sebagai reset, memproses data, melakukan komunikasi dari RFID Reader, mengolah inputan dari keypad dan penulisan dan intruksi pada tampilan LCD.

5. LCD (*Liquid Crystal Display*)

   Menampilkan data base pelanggan, saldo yang dimiliki pelanggan dan jumlah liter yang diinginkan.

6. Keypad

   Memasukkan PIN number dan jumlah liter yang diinginkan.

7. Simulator SPBU

Rangkaian driver untuk pengendalian pompa air sebagai outputan yang diterima dari mikrokontroller.

Prinsip Kerja :

Pada saat kartu tag pelanggan masuk dalam zona elektromagnetik dari *RFID Reader* maka akan terjadi pembacaan ID dari tag tersebut oleh *RFID Reader* yang kemudian format datanya langsung dikomunikasikan ke mikrokontroller. Pada mikrokontroller tersebut terdapat salah satu Pin yang difungsikan sebagai CS (*chip select*) yang berfungsi untuk memilih jalur komunikasi antara *RFID Reader dan RS 232*. Setelah itu pelanggan dapat memasukkan nomor PIN dan kemudian jumlah liter yang diinginkan melalui *Keypad* untuk memastikan keamanan dari kartu tag tersebut. Data dari *Keypad* tersebut diproses oleh mikrokontroller. Kemudian mikrokontroller akan berkomunikasi dengan *Data Base* pada PC melalui *RS 232*. oleh mikrokontroller nilai voucer dari *data base*, jumlah liter bahan baker yang diinginkan, jumlah nilai voucer setelah melakukan pengisian tersebut ditampilakan pada LCD. Dan mikrokontroller juga berfungsi mengendalikan driver relay pompa pada saat melakukan pengisian.

3.1.1. Perencanaan Rangkaian Antarmuka *Tag* dan *Reader* RFID

Mekanisme pembacaan antarmuka RFID adalah menggunakan *Tag* pasif, yaitu *tag* yang tidak memiliki catu daya sendiri serta tidak dapat menginisiasi komunikasi dengan *reader*. Sebagai gantinya, *tag* merespon emisi frekuensi radio dan menurunkan dayanya dari gelombang-gelombang energi yang dipancarkan oleg *reader*. Sebuah *tag* RFID atau *transponder*, terdiri atas sebuah mikro (*microchip*) dan sebuah antena . *Chip* mikro itu sendiri dapat berukuran sekecil butiran pasir, seukuran 0.4 mm. Chip tersebut menyimpan nomor seri yang unik atau informasi lainnya tergantung kepada tipe memorinya. Tipe memori itu sendiri dapat *read-only*, *read-write*, atau *write-onceread-many*. Antena terbuat dari bahan *mete coil* (baca BAB II) yang terpasang pada chip mikro mengirimkan informasi dari chip ke *reader* pada frekuensi standart 125 KHz. Biasanya rentang pembacaan diindikasikan dengan besarnya antena. Antena yang lebih besar mengindikasikan rentang pembacaan yang lebih jauh. *Tag* tersebut terpasang atau tertanam dalam obyek yang akan diidentifikasi. *Tag* dapat di*scan* dengan *reader* bergerak maupun stasioner menggunakan gelombang radio

3.1.1.1. Kartu *(tag)* RFID

Kartu (*tag*) pada alat ini menggunakan media RFID yang berfungsi agar pelanggan dapat bertransaksi sebagaimana yang dijelskan pada penjelasan diatas (baca: Mekanisme Kerja). output dengan format ASCII, karena output ini sangat mudah untuk dihubungkan pada mikrokontroler atau PC menggunakan komunikasi serial UART.

Misalkan ambil 10 data karakter ASCII. Dalam contoh ini berarti data tersebut adalah:

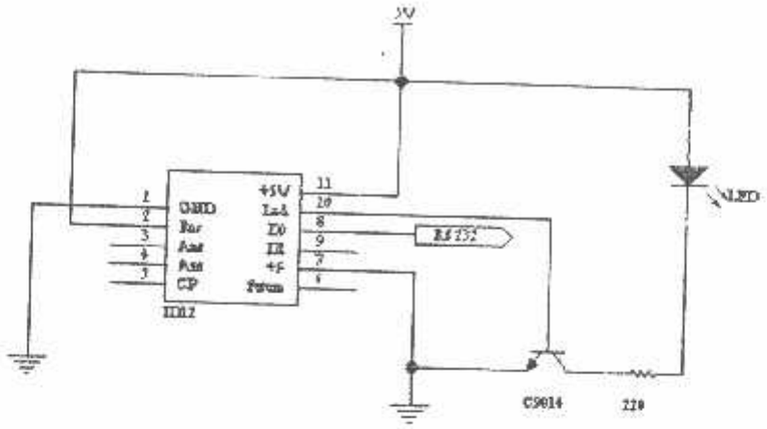
30	34	36	32	30	31	44	37	36	43	Data Heksa
		6	2	0	1	D	7	6	C	Data ASCII

(Baca: BAB II pembacaan format data). Hasil konversi dari data heksa ke dalam data ASCII adalah "6201D76C". Dirubah menjadi 1644287852 desimal (ini merupakan nomor kartu sebenarnya yang tertera pada badan kartu tsb).

Kartu tersebut nantinya akan dapat digunakan dan langsung diidentifikasi oleh pembaca RFID (*reader*) dan system akan memproses lebih lanjut dan akhirnya dapat dipergunakan oleh pelanggan.

3.1.1.2. Pembaca *(Reader)* RFID

Pembaca (*reader*) RFID ini akan digunakan sebagai media untuk mengenali dan melakukan pembacaan ID (*serial number*) yang tersimpan secara permanent di dalam kartu (*tag*) RFID melalui gelombang elektromagnetik yang dipancarkan oleh RFID (*reader*) dalam radius tertentu. ID yang termuat di dalamnya akan diteruskan ke mikrokontroller atau ke PC yang kemudian akan digunakan sebagai data yang diperlukan. Gambar 3.2 dibawah ini adalah rangkaian dari pembaca (*reader*) RFID.



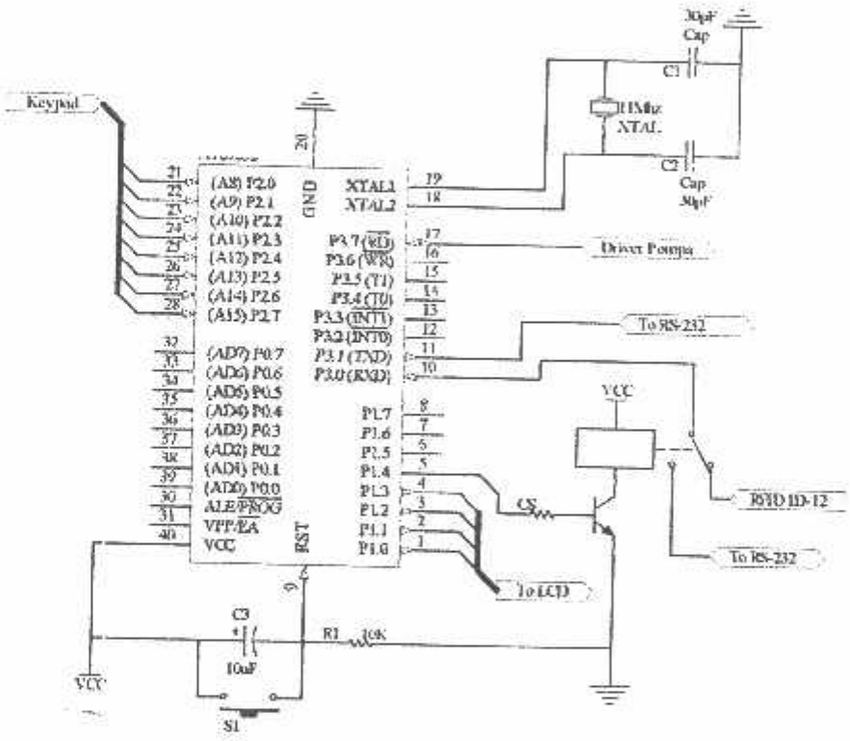
Gambar 3.2 Rangkaian Antarmuka RFID (*reader*)
Sumber : Perencanaan dan pembuatan

3.1.2. Perancangan Mikrokontroller AT89S52

Mikrokontroller AT89S52 adalah suatu chip IC yang terdiri dari 40 pin, dalam perencanaan alat ini mikrokontroller digunakan sebagai pusat pengendali kerja dari alat yang dibuat tersebut, disinilah tersimpan program-program (*software*) perintah serta alamat yang akan dituju program. Agar mikrokontroller dapat dipergunakan dengan baik dalam perancangan alat ini diperlukan beberapa elemen penting seperti *Vcc, Ground*, rangkaian *Reset*, rangkaian *clock* dan juga Port I/O yang digunakan dapat dilihat pada Gambar 3.3, dan dijelaskan sebagai berikut:

- Port 1.0 s/d Port 1.3 merupakan port yang digunakan sebagai data LCD.
- Port 1.4 Difungsikan sebagai CS (*chip select*) yang digunakan untuk memilih jalur komunikasi antara RFID *Reader* dan RS 232.
- Port 2.0 s/d Port 2.7 digunakan sebagai Input dari data Keypad.
- Port 3.0 digunakan untuk penerima data serial (Rx).

- Port 3.1 digunakan untuk pengiriaman data serial (Tx)
- Port 3.7 digunakan untuk pengendali dari driver pompa.
- Pin 18 dan 19 dihubungkan dengan rangkaian *clock* atau *oscillator eksternal.*
- Pin 9 (RESET), reset aktif tinggi yang terhubung dengan rangkaian power on reset dan jika diaktifkan akan mereset mikrokontroller AT89S52.
- Pin 20 (GND) digunakan sebagai ground
- Pin 40 (VCC) digunakan sebagai VCC dengan tegangan supply +5 V



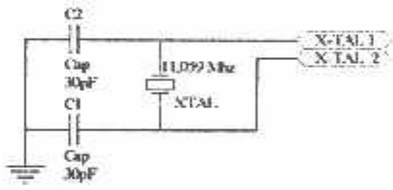
Gambar 3.3 Rancangan pemakaian *port-port* mikrokontroller AT89S52
Sumber : Perencanaan dan pembuatan

3.1.2.1. Rangkaian *Clock dan Reset* Minimum Sistem

Rangkaian yang mendukung mikrokontroller ada dua, yaitu rangkaian *clock* dan *reset.*

- *Clock* (X-TAL 1 dan X-TAL 2 pada pin 18 ,19)

Kecepatan proses yang dilakukan oleh mikrokontroller ditenyukan ole sumber *clock* (pewaktuan) yang mengendalikan mikrokontroller tersebut. System yang dirancang ini akan menggunakan *oscillator* internal yang sudah tersedia dalam chip mikrokontroller. Dalam perencanaan system ini frekuensi oscillator yang digunakan sebesar 11,059 MHz dan kapasitor penstabil sebesar 30pF. Berikut adalh gambar unruk rangkaian *clock.*



Gambar 3.4 Rangkaian *Clock*
Sumber : Perencanaan dan pembuatan

Berdasarkan rangkaian diatas maka didapatkan :

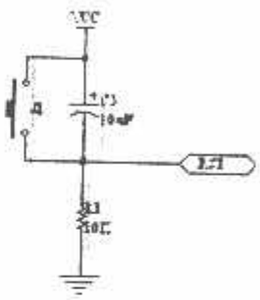
$$\text{Clock} = T = \frac{1}{F}$$

$$= \frac{1}{11{,}059}$$

$$= 0{,}09\ \mu s$$

- Rangkaian *Reset* (pin 9)

Untuk me*reset* mikrokontroller AT 89S52, maka RST (pin 9) diberi logika tinggi selama sekurangnya dua siklus mesin (*24 periode oscillator*). Untuk membangkitkan sinyal *reset* dibutuhkan rangkaian *reset* yang diharapkan akan mempunyai kemampuan *power ON Reset*, Reset terjadi saat *Power* diaktifkan seperti yang ditunjukkan gambar di bawah ini.



Gambar 3.5 Rangakaian *Reset*
Sumber : Perencanaan dan pembuatan

Pada rangkaian *reset* diatas didapatkan Fo dari persamaan berikut :

Jika diketahui nilai R = 10 KΩ, dan C = 10 μF

$$Fo = \frac{1}{1{,}1.RC}$$

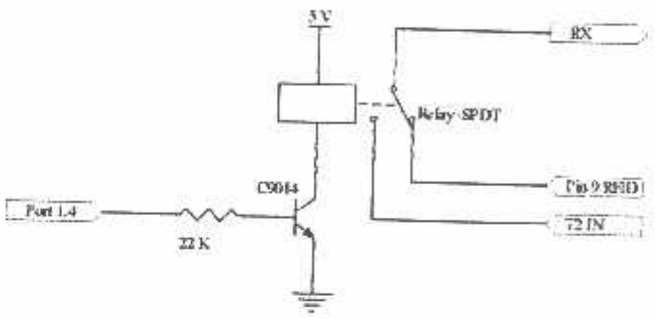
$$Fo = \frac{1}{1{,}1.10^{3}.10^{-6}}$$

$$Fo = 9{,}09\ Hz$$

3.1.3. Rangkaian Switch Data

Rangkaian switch data disini digunakan untuk memilih jalur komunikasi data dengan menggunakan CS (Chip Select) pada Port 1.4 pada mikrokontroller

AT 89S52 yang dihubungkan ke sebuah transistor type C 9014 dimana transistor tersebut berfungsi sebagai driver relay SPDT dengan catu daya 5 V. seperti yang dijelaskan pada gambar dibawah iniu :



Gambar 3.6 Rangkaian Switch Data
Sumber : Perencanaan dan pembuatan

Pada (C 9014 *Data Sheet*) transistor memiliki nilai Hfe = 80, dan $R_{relay} = 300\ \Omega$

$$I_C = \frac{Vcc - Vce}{Rrelay} \quad ; I_B = \frac{Ic}{Hfe}$$

$$I_C = \frac{5 - 0{,}25}{300} = 15{,}2\ \text{mA} \quad ; I_B = \frac{15{,}2mA}{80} = 0{,}1975\ \text{mA}$$

Setelh didapatkan nilai I_B , maka dapat ditentukan nilai R_B :

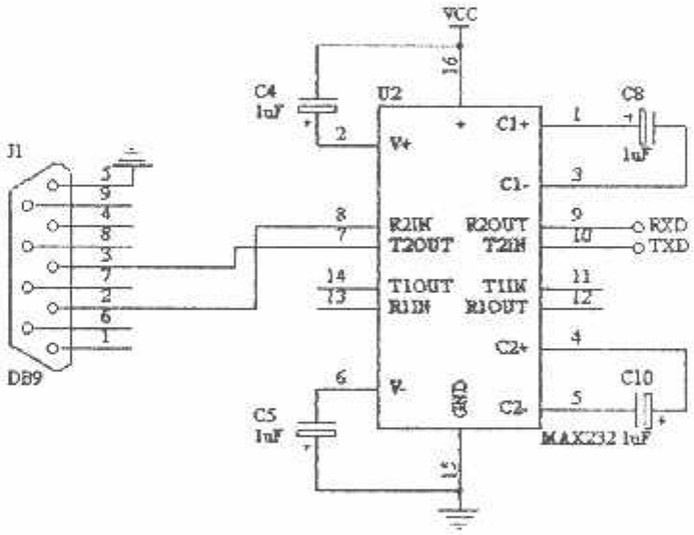
$$R_B = \frac{Vbb - Vbe}{Ib}$$

$$R_B = \frac{5 - 0{,}7}{0{,}1975mA} = 21{,}77\ \text{K}\Omega = 22\ \text{K}\Omega$$

Dengan didapatkanya nilai $R_B = 22\ \text{K}\Omega$

3.1.4. RS 232

Sebelum diinputkan ke komputer dibutuhkan rangkaian converter tegangan. Mikrokontroller mempunyai output logika high dihasilkan dari tegangan 5 volt dan logika low sebesar 0 volt, tegangan ini akan sering mengakibatkan terjadinya kesalahan didalam pengiriman dan penerimaan data dikarenakan rugi-rugi dari kabel. RS 232 berfungsi untuk memperlebar range tegangan karena berada dikisaran +10V dan -10V, dengan range yang lebar ini kesalahan karena rugi-rugi system komunikasi dari mokrokontroller ke komputer tidak mempengaruhi nilai data yang dikirim.



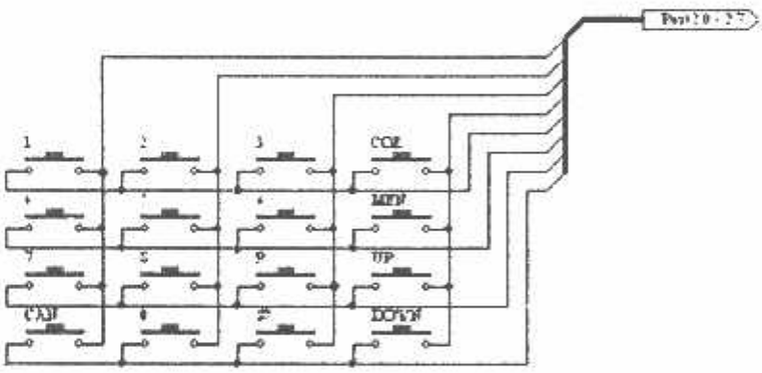
Gambar 3.7 Rangkaian RS 232
Sumber : Perencanaan dan pembuatan

Kabel penghubung dengan komputer digunakan DB 9. DB 9 yang di mikrokontroller pin 5 dihubungkan dengan ground, pin 2 dihubungkan dengan

pin 8 dari RS 232 dan pin 3 dihubungkan dengan pin 7. DB 9 yang dihubungkan dengan komputer adalan pin 2 dihubungkan dengan pin 3 dari DB 9 mikrokontroller dan sebaliknya pin 2 dihubungkan dengan pin 3, ini dilakukan supaya pengiriman data dari mikrokontroller di terima di penerima dari komputer dan sebaliknya.

3.1.5. Keypad

Perancangan *keypad* dirancang memiliki konfigurasi matrik 4X4, sehingga akan diperoleh tombol sebanyak 16 buah. Masing-masing tombol tersebut digunakan sebagai masukkan data.



Gambar 3.8. Rangkaian Keypad
Sumber : Perencanaan dan pembuatan

Adapun cara kerja keypad yang direncanakan dapat dijelaskan sebagai berikut : Setiap kali penekanan tombol akan terjadi suatu persilangan antara baris X dengan kolom Y. kondisi logic hasil penekanan tombol key pad tersebut dihubungkan pada Port Input (P2.0 – P2.7) melalui kaki X1-X4 dan Y1-Y4.

Keadaan penekanan tombol persilangan antara baris X dan kolom Y akan dibaca dan untuk sementara disimpan dimemory internal mikrokontroller schingga persilangan antara baris dan kolom dapat dikirimkan ke MCU.

Berikut tabel hasil scanning data oleh keypad 4X4 :

Tabel 3-1 Kombinasi Output Keypad Matrix 4X4
Sumber : Perencanaan dan pembuatan

NUM	SCANNING				DATA OUTPUT			
COT	0	1	1	1	0	1	1	1
3	0	1	1	1	1	0	1	1
2	0	1	1	1	1	1	0	1
1	0	1	1	1	1	1	1	0
MEN	1	0	1	1	0	1	1	1
6	1	0	1	1	1	0	1	1
5	1	0	1	1	1	1	0	1
4	1	0	1	1	1	1	1	0
UP	1	1	0	1	0	1	1	1
9	1	1	0	1	1	0	1	1
8	1	1	0	1	1	1	0	1
7	1	1	0	1	1	1	1	0
DOWN	1	1	1	0	0	1	1	1
ENT	1	1	1	0	1	0	1	1
O	1	1	1	0	1	1	0	1
CAN	1	1	1	0	1	1	1	0

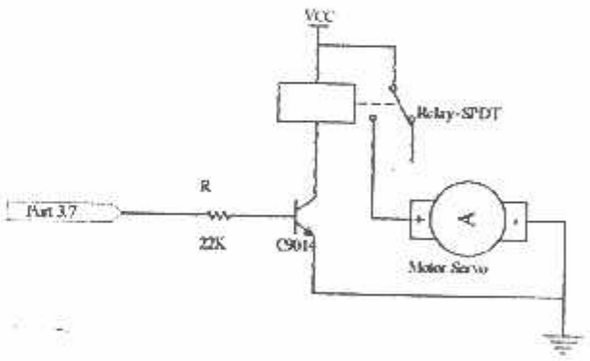
3.1.6. Rangkaian *Driver Relay*

Pada Perencanaan alat ini digunakan sebuah *driver relay*. Driver ini berfungsi sebagai penggerak *relay*. Dalam rangkaian drver relay ini digunakan transistor type C 9014 yang berfungsi sebagai *switch*. Jika ada sinyal dari mikrokontroller maka transistor akan *saturasi* sehingga *relay* akan menjadi aktif dan sebaliknya jika tidak ada sinyal dari mikrokontroller maka transistor akan *cut-off* sehingga *relay* tidak aktif.

Relay yang digunakan dalam rangkaian ini adalah jenis SPDT *(Single Pole Dual Terminal)* yang memiliki resistansi *Rrelay* sebesar 800 Ω.dan bekerja pada catu daya +12 V. Sedangkan transistot type C 9014 yang digunakan pada rangkaian *driver relay* mempunyai spesifikasi sebagai berikut :

VCE saturasi : 0,25 V
Hfe : 80
Vbe : 0,7 V

Berikut rangkaian driver relay dapat ditunjukkan pada gambar dibawah ini :



Gambar 3.9 Rangkaian Pengendali *Relay*
Sumber : Perencanaan dan pembuatan

Untuk menentukan nilai I_c, I_b, dan R_b agar rangkaian dapat bekerja seperti yang diharapkan dapat diperoleh dari persamaan dibawah (Malvino, 1981.122).

Jika $R_c = R_{relay}$

Maka : $I_c = \frac{Vcc - Vce}{Rrelay}$

$= \frac{12 - 0{,}25}{800} = 14{,}68$ mA

R_b dapat dicari dengan mencari I_b terlebih dahulu dengan menggunakan persamaan dibawah ini :

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

$$I_b = \frac{14{,}68}{80} = 0.1835 \text{ mA}$$

Untuk mencari nilai R_b apabila diketahui nilai $V_{BB} = 5$ V digunakan persamaan sebagai berikut (Malvino, 1981:121) :

$V_{BB} = V_B + V_{BE}$

$5 = I_B \,.\, R_B + 0{,}7$

$5 = 0{,}1835 \text{ mA} \,.\, R_B + 0{,}7$

$R_B = \frac{5 - 0{,}7}{0{,}1835 x 10^{-3}}$

$= \frac{4{,}3 x 10^3}{0{,}1835} = 23{,}43 \text{ K}\Omega = 22 \text{ K}\Omega$

3.1.7. Rangkaian LCD (*Liquid Crystal Display*)

LCD *Display module* M 1632 buatan *Seiko instrument Inc.* terdiri dari 2 bagian, yang pertama merupakan panel LCD sebagai media penampil informasi dalam bentuk huruf atau angka 2 baris, masing-masing baris bisa menampung 16 huruf atau angka.

Bagian kedua merupakan sebuah sistem yang dibentuk dengan mikrokontroller yang ditempekan dibalik panel LCD, berfungsi mengatur tampilan informasi serta berfungsi mengatur komunikasi M 1632 dengan mikrokontroller utama. Dengan demikian pemakaian M 1632 menjadi sederhana dibandingkan dengan sistem lain, karena M 1632 cukup mengirim kode ASCII dari informasi yang ditampilkan seperti layaknya memakai sebuah printer.

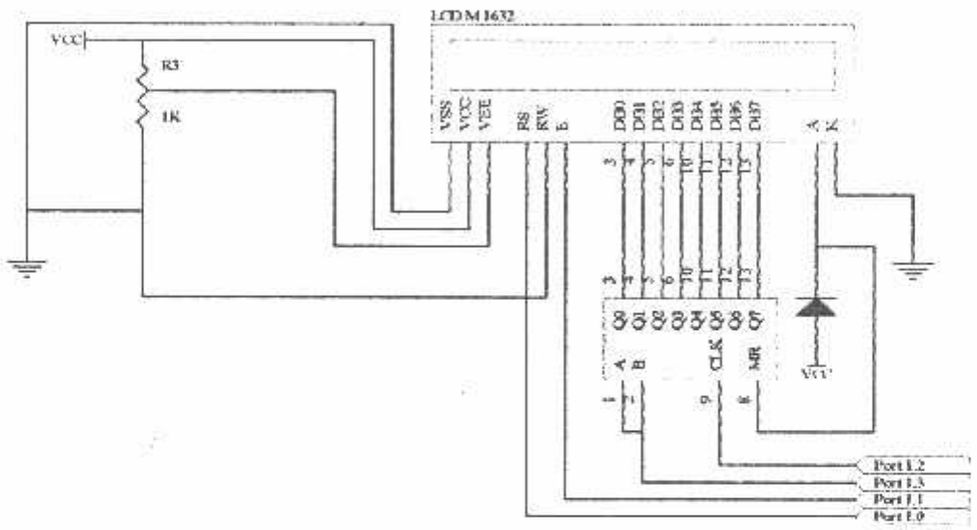
Rangkaian LCD M 1632 ini adalah komponen *display* yang umum digunakan. *Display* LCD M 1632 ini memiliki ROM sebagai penyimpanan karakter sebanyak 192 buah. Sebelum mengoperasikan LCD sebagai penampil karakter, terlebih dahulu ditentukan format penulisan LCD.

Dalam penulisan format LCD terada beberapa aturan yang diberikan oleh pabrik pembuatnya (dalam *data sheet*) yaiitu:

1. Menentukan jalur bit data yang akan digunakan.
2. Membersihkan layar *display* dari dari karakter *blank* .
3. Menentukan alamat baris pertama dan baris ke dua.
4. Dalam penulisan karakter menggunkan *cursor* atau tidak

Jika penginilisialisasian sudah selesai, langkah selanjutnya adalah menulis karakter yang diinginkan. Misalnya tampilan yang diinginkan adalah "123", maka

format data yang ditansfer ke jalur data adalah format data BCD angka 123 yang masing-masing disertai data posisi baris. Data yang dikirim ke LCD cukup satu kali, selanjutnya data akan terus tampil berulang-ulang oleh LCD itu sendiri selama tidak ada instruksi untuk membersihkan layar. Hubungan pin data dengan dengan pin control LCD dengan MCU ditunjukkan pada gambar di bawah ini :



Gambar 3.10 Rangkaian LCD M1632
Sumber : Perencanaan dan pembuatan

Fungsi dari masing-masing pin LCD yang digunakan adalah :

- Pin RS dihubungkan dengan *port* 1.0 dari MCU untuk membedakan sinyal antara instruksi program atau instruksi penulisan data
- Pin E dihubungkan dengan *port* 1.1 dari MCU untuk memberikan instruksi penulisan pada alamat LCD

- Pin DB0 – DB7 dihubungkan pada Q0 – Q7 IC DM 74LS164 *shif register* dimana input A dan B terhubung pada *port* 1.3 dan CLK terhubung pada *port* 1.2 sebagai pengontrollan data yang akan masuk ke LCD

3.2. Perancangan Perangkat Lunak

3.2.1. Perangkat Lunak Mikrokontroller AT89S52

Perangkat Lunak yang digunakan untuk minimum sistem AT89S52 ini adalah menggunakan bahasa pemrograman bahasa C. Program yang ditulis dengan pemrograman bahasa C Label; Kode *Mnemonic* dan lain sebagainya, pada umumnya di amakan sebagai Program Sumber (*Source Code*) yang belum bisa diterima oleh prosesor untuk dijalankan sebagai program, tetapi harus diterjemahkan dulu menjadi bahasa mesin dalam bentuk kode hexa atau biner.

Pada skripsi ini Program Sumber diterjemahkan atau di*compile* menggunakan program *Keile51* bila *compile* tidak ada tanda *compile error* maka program benar, untuk mensimulasikan program mengunakan program TS Emulator 8051 V.1.1. Hasil *compile* berektensi "*Hex*", "*Obj*", "*Lst*", dan "*Bin*".

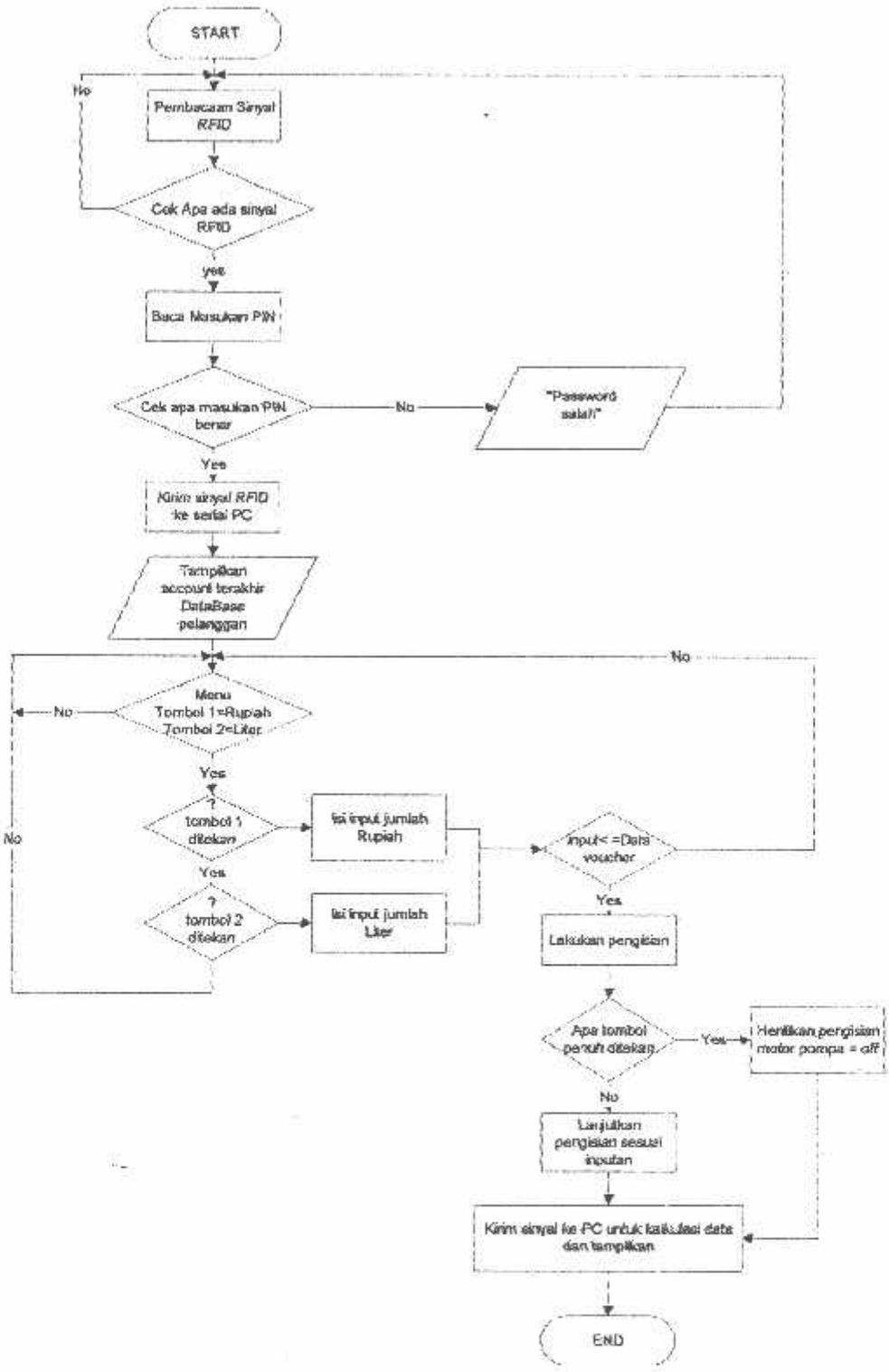
File yang berektensi "*Hex*" berisikan kode-kode bahasa mesin. Kode-kode bahasa mesin inilah yang dimasukkan ke memori program (ROM). Dalam dunia mikrokontroller biasanya file *Hex* ini diisikan ke EPROM, dan khusus untuk

akan diisikan pada memori program bersangkutan. File ini sangat berguna untuk dokumentasi dan sarana untuk menelusuri program yang ditulis apabila terjadi kesalahan.

Perlu diperhatikan adalah setiap prosessor mempunyai konsentrasi yang berbeda, intruksi untuk mengendalikan masing-masing prosesor juga berlainan dengan demikian bahasa *Assembler* untuk masing-masing prosesor juga berbeda, yang sama hanyalah pola dasar cara penulisan program saja.

3.2.2. Perangkat Lunak Pada PC

Perangkat lunak pada PC digunakan untuk program aplikasi komunikasi data serial mikrokontroller, agar PC dapat mengenali dan memproses sinyal data serial dari mikrokontroller atau sebaliknya dalam perencanaan ini digunakan bahasa pemrograman *Borland Delphi* yang juga berfungsi sebagai *database*.



Gambar 3.11 *Flowchart* Mikrokontroller
Sumber : Perencanaan dan Pembuatan

Gambar 3.12 *Flowchart Software* Pada PC
Sumber : Perencanaan dan Pembuatan

BAB IV

ANALISA DAN PENGUJIAN ALAT

4.1 Pendahuluan

Tahapan yang paling penting pada perancangan alat ini adalah pengujian. Dari pengujian dapat diketahui apakah alat yang dibuat bekerja dengan baik atau masih kurang sempurna.

Pada bab ini akan dibahas tentang pengujian alat yang telah dirancang dengan tujuan agar perancangan dan pembuatan dengan hasil keperluan yang ada, dipandang dari segi perancangan *hardware* maupun *software* dapat sesuai dengan dengan kondisi yang diinginkan. Dengan diadakan pengujian alat, maka pada skripsi ini dapat membuktikan bahwa alat yang dibuat dapat berjalan sesuai dengan kondisi masukan yang ada, sehingga memberikan keluaran yang sesuai pula. Bagian-bagian yang diuji dari peralatan ini adalah :

1. RFID (*Radio Frequensi Identification*).
2. Pengujian Mikrokontroller.
3. Pengujian Rangkaian serial 232.
4. Keypad.
5. Tampilan LCD.
6. Keseluruhan Sistem

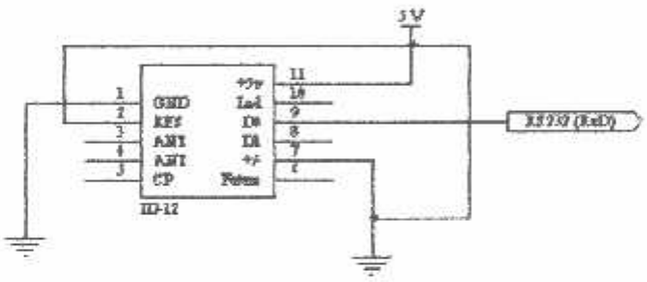
4.2. Pengujian RFID

4.2.1. Tujuan

Tujuan dari pengujian ini adalah untuk mengetahui apakah *tag* RFID bisa dibaca oleh *Reader* RFID. Adapun cara pengujianya adalah merangkai rangkaian RFID dan kemudian menghubungkan ke COM1 PC. Untuk menguji *reader* bisa membaca kartu RFID dilakukan melalui komunikasi *Hyper Terminal.*

4.2.2. Prosedur pengujian

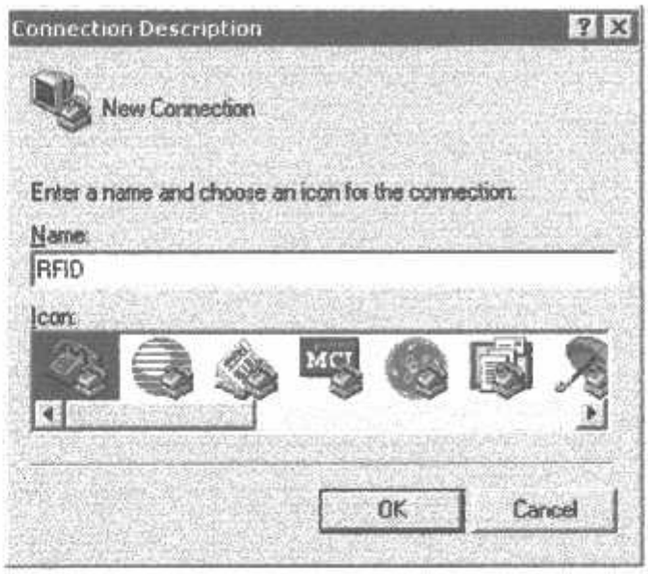
a. Menghubungkan rangkaian RFID ke COM1 PC.



Gambar 4.1 Rangkaian Pengujian *RFID*
Sumber : Pengujian

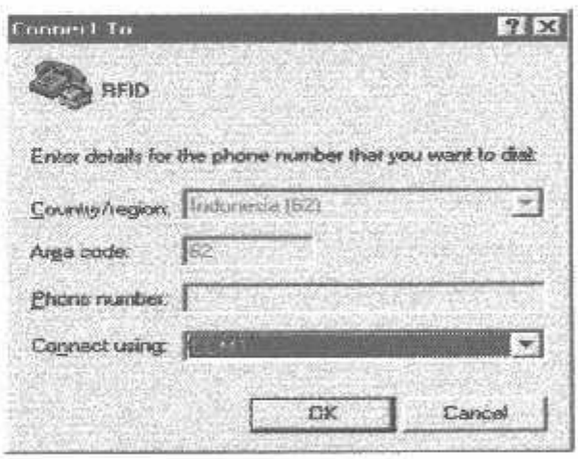
b. Membuka *Hyper Terminal* (*start* → *all program* → *accessories* → *communication* → *hyper terminal*)

c. Memberi nama dan memilih *icon* pada *Connection Description.*



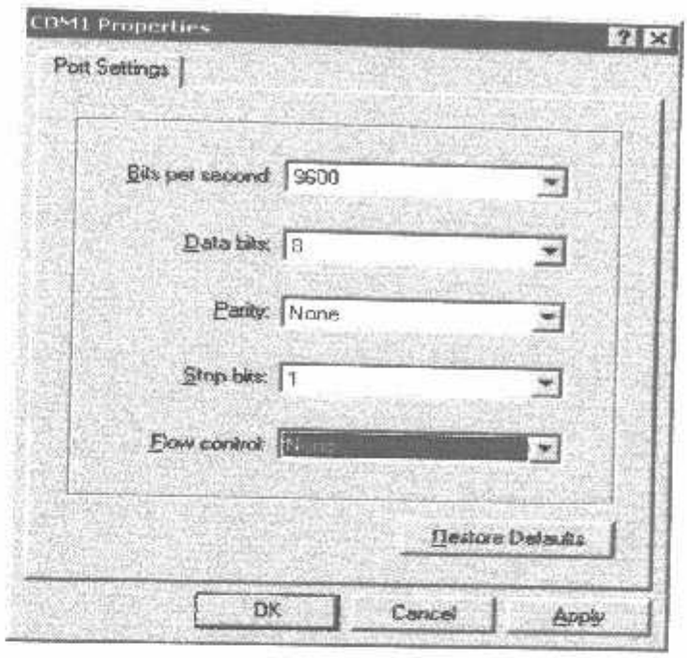
Gambar 4.2 Kotak Dialog *Conection Description*
Sumber : Pengujian

d. Memilih COM1 pada kotak dialog connect to



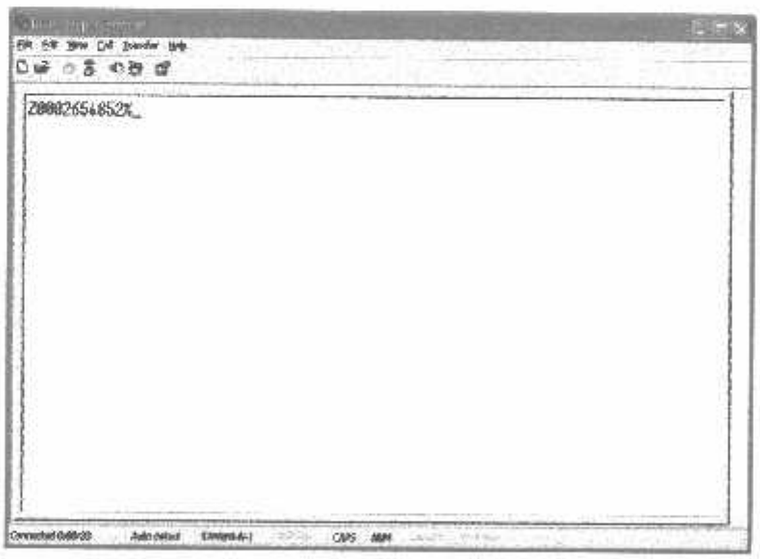
Gambar 4.3 Kotak Dialog *Connect to*
Sumber : Pengujian

e. Pada COM1 *propertis* mengubah *bit rate per second* menjadi 9600 dan *flow control* menjadi *none*.



Gambar 4.4 Kotak Dialog COM1 *Propertis*
Sumber : Pengujian

f. Menempatkan kartu pada jarak yang dijangkau reader sehingga menampilkan angka dari kartu tersebut.



Gambar 4.5 Kotak Dialog Hasil *Identifikasi Reader* terhadap Kartu
Sumber : Pengujian

4.2.3. Hasil Pengujian Pembacaan RFID

Tabel 4-1 Tabel Hasil Pengujian Pembacaan RFID
Sumber : Pengujian

Jarak	Percobaan									
	1	2	3	4	5	6	7	8	9	10
1 cm	√	√	√	√	√	√	√	√	√	√
2 cm	√	√	√	√	√	√	√	√	√	√
3 cm	√	√	√	√	√	√	√	√	√	√
4 cm	√	√	√	√	√	√	√	√	√	√
5 cm	√	√	√	√	√	√	√	√	√	√
6 cm	–	–	–	√	–	–	–	√	–	√
7 cm	√	–	–	–	–	√	–	–	–	–
8 cm	–	–	–	–	–	–	–	–	–	–
9 cm	–	–	–	–	–	–	–	–	–	–
10 cm	–	–	–	–	–	–	–	–	–	–

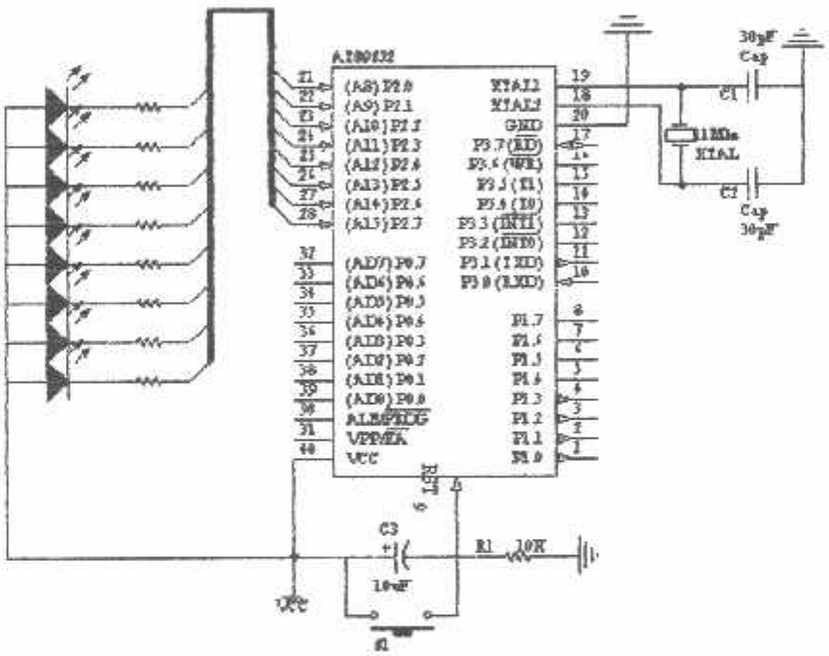
Tabel di atas merupakan hasil pengujian dimana kartu yang menghadap *reader* adalah bagian depan. Jarak yang baik untuk bisa teridentifikasi adalah 5 cm. untuk bagian belakang menghasilkan data yang sama, tetapi untuk pengujian dimana kartu tegak lurus dengan *reader* hanya bisa saat kartu berjarak sangat dekat dengan *reader* (menempel).

4.3. Pengujian Mikrokontroller AT89S52

4.3.1. Tujuan Pengujian

Pengujian ini dilakukan untuk mengetahui apakah rangkaian mikrokontroller yang dibuat dapat bekerja sesuai dengan inputan program yang di download di dalamnya.

4.3.2. Prosedur pengujian :



Gambar 4.6 Rangkaian Pengujian Mikrokontroller
Sumber : Pengujian

1. Membuat rangkaian seperti diagram blok pada gambar 4.6
2. Memberi catu daya 5V
3. Membuat program sederhanadengan memberi inputan $0F_h$ dan $F0_h$ pada port 1 AT89C51. Program yang dibuat :

```
#include <8051.h>
{           int count;
            TMOD=0x10;
            For  (count=1;count<=;count++)
        {   TH1=0x3c;
            TL1=0xaf;
            TR1=1;
            While (TF1==0);
            TR1=0;
            TF1=0;
        }
}

 Void main ()
{               unsigned char data;
                While (1)
            {   nilai=0x0f;
                P1=nilai;
                Dolay_50ms (10);
                Nilai=0xf0;
                P1=nilai;
            }
}
```

4. Download program diatas.

5. Mengamati keluaran pada LED display.

Tabel 4-2 Keluaran LED Display
Sumber : Pengujian

Kondisi	Keluaran Led Display							
	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
Satu	1	1	1	1	0	0	0	0
Dua	0	0	0	0	1	1	1	1

4.3.3. Analisa Hasil Pengujian

Pada pengujian rangkaian terlihat bahwa Port 1 memberikan keluaran logika $0F_h$ dan F0h sehingga nyala Led seperti pada tabel 4.3 apabila diketahui Led display *active low*.

4.4. Pengujian Komunikasi Serial

4.4.1. Tujuan

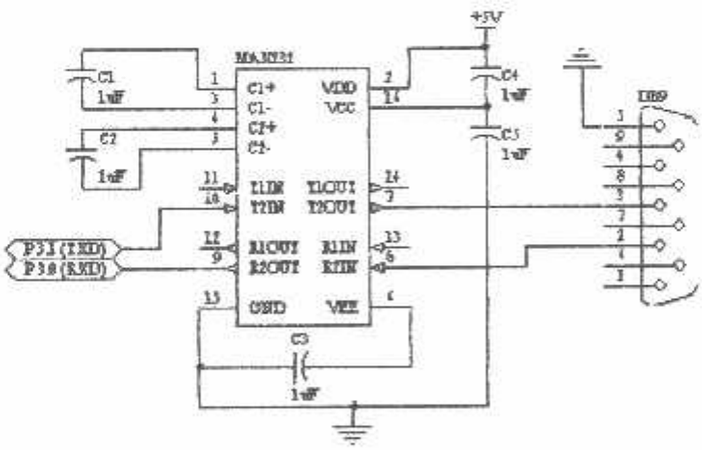
Untuk mengetahui apakah data yang dikirim dari MCU ke PC dapat diterima dengan benar dengan melakukan simulasi pada komputer.

4.4.2. Peralatan yang digunakan

1. Komputer
2. Sistem mikrokontroller dengan antarmuka RS232.
3. Writer downloader

4.4.3. Prosedur pengujian

1. Menyusun rangkaian seperti pada Gambar 4.7.



Gambar 4.7 Rangkaian Pengujian *RS 232*
Sumber : Pengujian

2. Membuat program transfer data pada sistem mikrokontroller seperti yang ditunjukkan dalam Gambar 4.8 Dengan program tersebut komputer mengirim data '1234567890' ke alat dan oleh alat akan dikembalikan lagi ke komputer.
3. Download program ke Writer Downloader dan eksekusi program.
4. Mencatat hasil yang terlihat dalam layar computer

```
#include <8051.h>
void delay_lus (int n)
{
        int count;
        for (count=1;count<=n;count++)
        {
                _asm
                        Nop
                _endasm;
        }
}
void delay_50ms (int n)
{
        int count;
        for (count=1;count<=n;count++)
        {
                TH0=0x3c;
                TL0=0xaf;
                TR0=1;
                While (TF0==0)
                TR0=0;
                TF0=0;
        }
}
void init_timer_serial ()
{
        TMOD=0x21;
        TH1=0xf3;
        SCON=0x52;
        PCON=0x00;
        TR1=1;
}
void serial_transmit (char karakter)
{
        T1=0;
        SBUF=karakter;
        While (T1==0);
}
void main ()
{
        code char nilai []="1234567890,";
        int I;
        init_timer_serial ();
        do
        {       i=0;
                while (nilai (i)!=',');
                i++;
                delay_lus (1);
        }
        Delay_50ms (60);
        } while (1);
}
```

Gambar 4.8. Program Uji Komunikasi Data di Komputer
Sumber : Pengujian

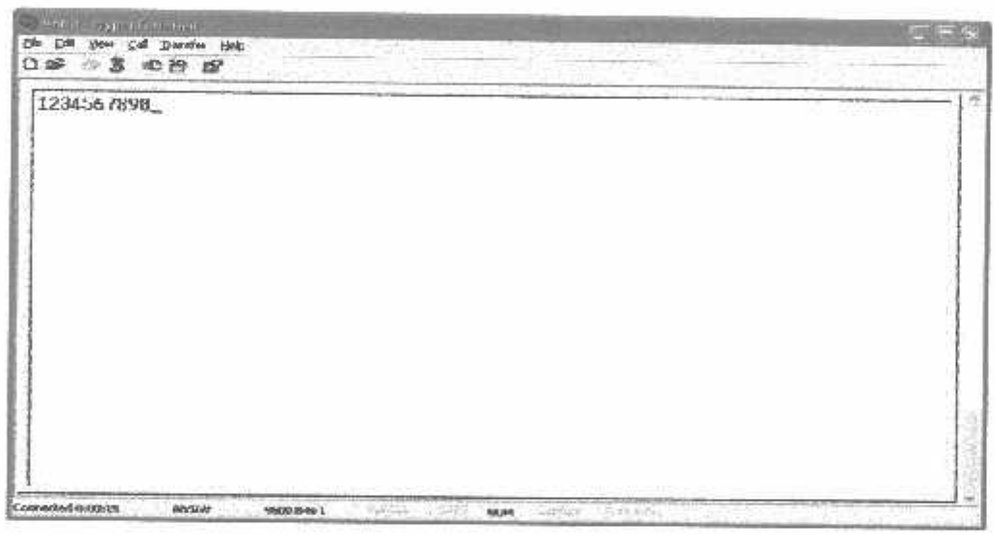
➢ **Hasil Pengujian**

Hasil pengujian transfer data serial ini ditunjukkan Tabel 4-4 dibawah ini :

Tabel 4-3 Hasil Pengujian Transfer Data Serial
Sumber : Pengujian

Data yang dikirim Mikrokontroller	Data yang diterima komputer
1234567890	**1234567890**

Atau dapat dilihat pada tampilan komunikasi *Hyper terminal* seperti Gambar 4.9 dibawah ini :



Gambar 4.9. *Hyper Terminal* Uji Komunikasi Data di Komputer
Sumber : Pengujian

4.4.4. Analisis Hasil Pengujian

Hasil pengujian dalam Tabel 4.4 menunjukkan bahwa proses pengiriman data serial dengan menggunakan RS 232 ke alat telah benar dan sesuai.

4.5. Pengujian *keypad*

4.5.1. Tujuan Pengujian

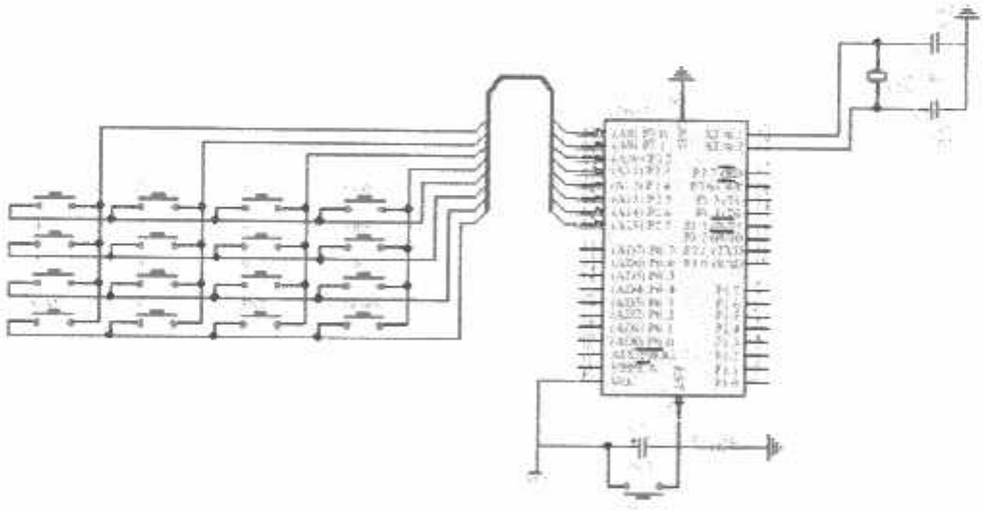
Pada peralatan yang direncakan, keypad merupakan komponen yang sangat penting karena digunakan sebagai sarana untuk menginput data. Tujuan dari pengujian ini adalah untuk mengetahui apakah data *output* setelah penekanan *keypad* sesuai dengan data pada program.

4.5.2. Peralatan yang dibutuhkan

1. *Keypad* 4X4.
2. Minimum sistem mikrokontroler AT89S52.
3. Multimeter digital
4. Catu daya +5V

4.5.3. Prosedur Pengujian

1. Menyusun komponen-komponen yang dibutuhkan sesuai dengan rangkaian berikut ini:



4.10. Rangkaian Pengujian *Keypad*
Sumber : Pengujian

2. Membuat *listing* program seperti dibawah ini:

```
#include<at89x51.h>
unsigned char temp;
char Tombolnya()
{
        P2=0xFE                 ;
        switch ( (P2>>4) & 0x0F )
        {
        case 0x0E:
                return '1'      ;
                break           ;
        case 0x0D:
                return '2'      ;
                break           ;
        case 0x0B:
                return '3'      ;
                break           ;
        case 0x07:
                return 'R'      ;
                break           ;
        }
        P2=0xFD                         ;
        switch ( (P2>>4) & 0x0F )
        {
        case 0x0E:
                return '4'      ;
                break           ;
        case 0x0D:
                return '5'      ;
                break           ;
        case 0x0B:
                return '6'      ;
                break           ;
        case 0x07:
                return 'M'      ;
                break           ;
        }
        P2=0xFB                         ;
        switch ( (P2>>4) & 0x0F )
        {
        case 0x0E:
                return '7'      ;
                break           ;
        case 0x0D:
                return '8'      ;
```

```
            break           ;
        case 0x0B:
            return '9'      ;
            break           ;
        case 0x07:
            return 'U'      ;
            break           ;
        }
        P2=0xF7                         ;
        switch ( (P2>>4) & 0x0F )
        {
        case 0x0E:
            return 'N'      ;
            break           ;
        case 0x0D:
            return '0'      ;
            break           ;
        case 0x0B:
            return 'E'      ;
            break           ;
        case 0x07:
            return '.'      ;
            break           ;
        default :
            return 0xff     ;
            break           ;
        }
}
void main()
{
 while(1)
 {
 temp=Tombolnya();
 P1=temp;
 }
}
```

3. Meng-*compile* dan men-*download*-kan program yang telah dibuat tadi kedalam mikrokontroler AT89S52.

4. Melakukan penekanan pada keypad, mengukur tegangan keluaran di port *1*.

Tabel 4.4 Pengujian Keypad

Input Keypad	Output							
	P1.0	P1.1	P1.2	P1.3	P1.4	P1.5	P1.6	P1.7
	Teg	Teg	Teg	Teg	Teg	Teg	Teg	Teg
1	4.52	0.68	0.69	0.68	4.52	4.58	0.68	0.69
2	0.68	4.60	4.52	0.69	4.59	4.60	0.70	0.69
3	4.64	4.61	0.61	0.69	4.65	4.64	0.62	0.69
4	0.69	0.64	4.55	0.69	4.61	4.62	0.70	0.69
5	4.64	0.70	4.69	0.70	4.68	4.68	0.70	0.69
6	0.70	4.68	4.69	0.69	4.68	4.68	0.71	0.69
7	4.69	4.71	4.69	0.69	4.72	4.71	0.63	0.69
8	0.69	0.70	0.70	4.62	4.64	4.63	0.70	0.68
9	4.68	0.70	0.70	4.69	4.69	4.68	0.70	0.71
0	0.71	0.69	0.71	0.70	4.69	4.62	0.70	0.71
COR	0.71	4.62	0.69	0.70	4.66	0.72	4.65	0.69
MEN	4.68	0.70	4.65	4.65	0.69	0.71	4.66	0.70
UP	4.71	0.73	4.70	0.73	4.72	0.71	4.71	0.71
DOWN	0.72	4.67	4.62	4.61	0.71	4.61	0.70	0.70
CAN	0.70	4.64	4.68	4.62	0.70	0.72	4.64	0.71
ENT	4.66	0.71	4.65	0.71	0.71	0.72	4.65	0.70
Tanpa Penekanan	4.72	4.63	4.82	4.80	4.82	4.82	4.81	4.82

4.5.4. Analisa

Dari tabel pengujian diatas, dapat dilihat bahwa setiap kali penekanan tombol yang berbeda, output tegangan di port 1 juga berbeda pula. Tegangan output pada port 1 yang mendekati 5 volt dianggap sebagai logika 'HIGH', dan jika mendekati 0 volt dianggap sebagai logika 'LOW'.

4.6. Pengujian LCD

4.6.1. Tujuan Pengujian

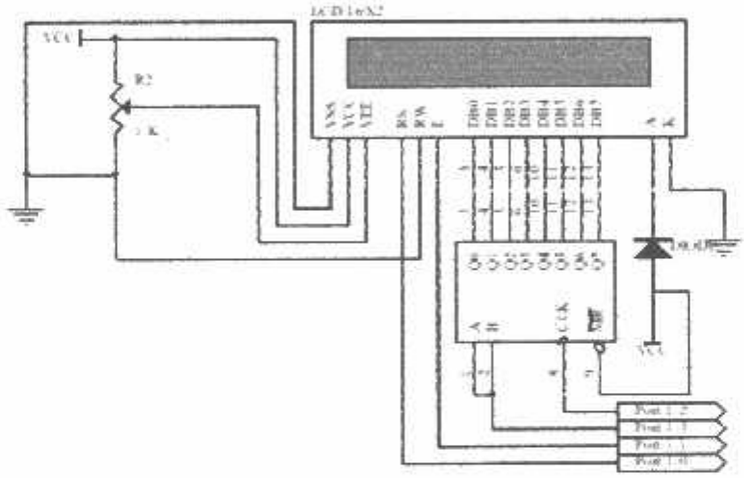
Pengujian ini bertujuan untuk mengetahui apakah rangkaian LCD serial yang direncanakan dapat bekerja dan LCD dapat menampilkan data yang dikirimkan oleh mikrokontroler atau tidak.

4.6.2. Peralatan yang Digunakan

1. LCD.
2. Minimum sistem mikrokontroler AT89S52.
3. IC shift register 8 bit (74LS164)
4. Catu daya +5V

4.6.3. Prosedur Pengujian

1. Merangkai interface LCD dan mikrokontroler sesuai dengan rangkaian yang direncanakan.



Gambar 4.11 Rangkaian interface LCD dan Mikrokontroler AT89C51
Sumber : Pengujian

2. Membuat listing program sebagai berikut:

```
#include<at89x51.h>
#include<LCDS1.C>

void main()
{
        Init_LCD();
        ClearLCD();
        Display_Control(LCDOn,NoBlink,Hide);
        Printxy(1,1,"1234567890NRMU.");
}
```

5. Meng-*compile* dan Men-*download* program tersebut kedalam mikrokontroler AT89S52.
6. Menghubungkan rangkaian dengan catu daya +5V.

Atau dapat dilihat pada tampilan *LCD* seperti Gambar 4.12 dibawah ini :

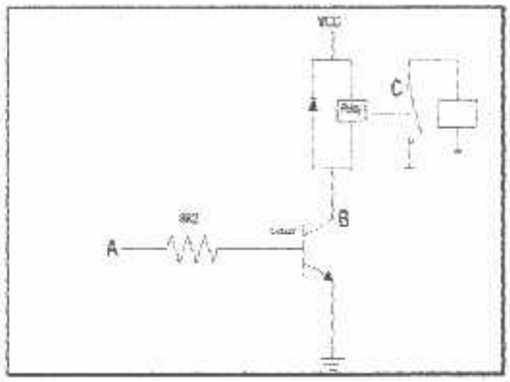
Gambar 4-12 Gambar Pengujian LCD 16x2
Sumber : Pengujian

4.6.4. Analisa

Dari pengujian tersebut diketahui bahwa data dikirimkan dari mikrokontroler ke LCD secara serial dengan menggeser tiap bit data mulai dari LSB hingga MSB. Rangkaian *interface* antara LCD, mikrokontroler AT89S52 bekerja sesuai dengan perencanaan dan LCD dapat menampilkan data yang dikirimkan oleh mikrokontroler.

4.7. Pengujian rangkaian *driver*

Blok pengujian rangkaian driver relai untuk motor pompa ditunjukkan seperti dalam Gambar 4-13.



Gambar 4.13. Pengujian Rangkaian *Driver*
Sumber : Pengujian

4.7.1. Tujuan

Pengujian ini bertujuan untuk mengetahui apakah motor bekerja dengan baik sesuai dengan yang direncanakan.

4.7.2. Langkah pengujiannya adalah sebagai berikut :

1. Menyusun rangkaian seperti gambar 4-4 diatas.
2. Memberikan catu daya pada rangkaian *driver* relay.
3. Mengamati keluaran dari kondisi relay dan output relay.
4. Hasil Pengujian rangkaian driver relay ditunjukkan dalam Tabel 4-4

4.7.3. Hasil Pengujian

Tabel 4-5 Hasil Pengujian Rangkaian *Driver* relay
Sumber : Pengujian

Kondisi A	Kondisi B	Kondisi C	Kondisi Relay
0 Volt	12 Volt	0 (Logika Low)	Relay Off
5 Volt	0 Volt	1 (Logika High)	Relay On

4.7.4 Analisis hasil pengujian

Dari Tabel 4-4 terlihat bahwa Jika driver relay berlogika tinggi ("1") maka kondisi relay on, demikian sebaliknya jika driver relay berlogika rendah ("0") maka kondisi relay off.

4.8. Pengujian Sistem Pembelian Menggunakan Voucer Berbasis RFID.

4.8.1. Tujuan

Tujuan dari pengujian ini adalah untuk mengetahui apakah semua system berjalan dengan normal dan juga untuk mengetahui error yang terjadi.

4.8.2. Prosedur Pengujian

a. Menghubungkan keseluruhan rangkaian sesuai dengan diagram blok.
b. Menjalankan program Database Delphi.
c. Melakukan proses identifikasi.
d. Melewatkan Tag RFID.
e. Melakukan transaksi pembelian.

4.8.3. Hasil Pengujian

1 Tampilan Informasi Pada LCD.

a. Menampilkan Tulisan "RFID SPBU".
b. Menampilkan Hasil Pembacaan Tag RFID.
c. Masukkan PIN.

d. Menampilkan Mode Pilihan Transaksi Dalam Satuan Liter/Rupiah.

e. Menuliskan Besaran Satuan Sesuai Dengan Pilihan Diatas.

f. Menampilkan Perhitungan Dalam Jumlah Liter/Rupiah Sesuai Dengan Keluaran.

2 Tampilan Informasi Pada Program Delphi.

a. Menampilkan no ID dari Tag RFID.

b. Menampilkan Password.

c. Menampilkan Nama.

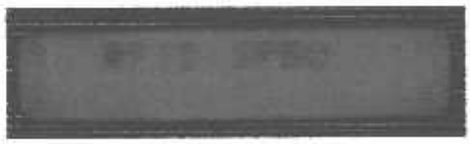
d. Menampilkan Alamat dan Nomor Telepon.

e. Menampilkan Account.

3 Proses transaksi dengan pengaktifan driver relay pada pompa motor.

4.8.4. Proses Identifikasi Dan Sistem Transaksi.

a. Pada saat awal atau *stand by* terdapat tampilan "RFID SPBU" pada LCD seperti gambar di bawah ini :



Gambar 4.14 Tampilan Awal
Sumber : Pengujian

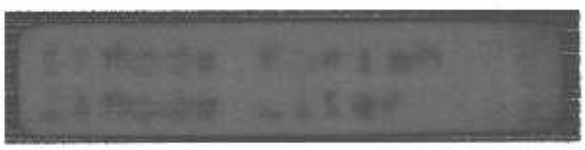
b. Setelah dilakukan percobaan *pengidentifikasian* maka untuk mendapatkan hasil yang maksimal maka kartu harus berada dalam jarak kurang sama dengan 5 cm, seperti yang tercantum dalam table 4-1. Sehingga didapatkan tampilan hasil pembacaan seperti gambar di bawah ini :

Gambar 4.15 Tampilan Hasil *pengidentifikasian*
Sumber : Pengujian

c. Langkah berikut setelah *pengidentifikasian* berhasil yaitu dengan pengisian nomor PIN melalui keypad sesuai tampilan gambar 4.16 kemudian tekan enter.

Gambar 4.16 Tampilan Pengisian Nomor PIN
Sumber : Pengujian

d. Setelah itu pada LCD akan menampilkan mode pilihan transaksi seperti gambar 4.17 yaitu tekan tombol 1 untuk pilihan transaksi Rupiah dan tekan tombol 2 untuk pilih transaksi Liter.



Gambar 4.17 Tampilan Mode Pengisian
Sumber : Pengujian

e. Kemudian menuliskan besaran satuan dari mode pilihan transaksi yang telah dipilih hingga didapat tampilan seperti gambar :

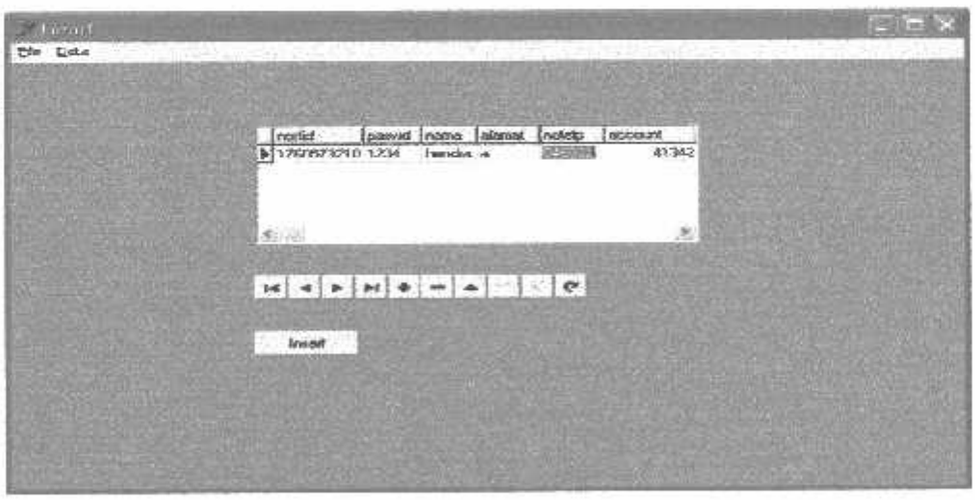
Gambar 4.18 Tampilan Besaran Satuan Yang Diinginkan
Sumber : Pengujian

f. Pada gambar 4.19 LCD menampilkan perhitungan sesuai dengan output yang dikeluarkan melalui pompa air.

Gambar 4.19 Tampilan Perhitungan Keluaran
Sumber : Pengujian

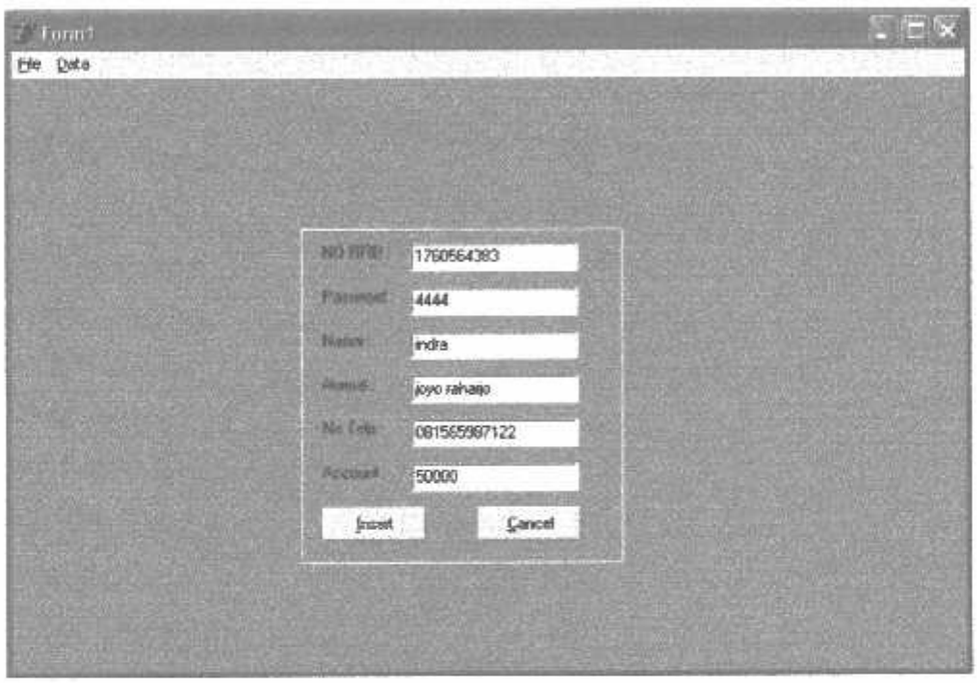
4.8.5. Proses Penambahan Pada Data Base.

a. Pada saat *tag* berada dalam jangkauan *reader* maka akan terjadi proses pengidentifikasi yang bisa memberikan informasi dari pemilik kartu berikut accountnya. Hal tersebut dapat dilihat pada gambar berikut :



Gambar 4.20 Kotak Dialog Database Identitas Pemilik Kartu
Sumber : Pengujian

b. Setelah itu dilakukan percobaan penambahan identitas baru ke dalam Database dengan menekan tombol "*Insert*" pada kotak dialog pada gambar 4.11 diatas. Sehingga akan menampilkan kotak dialog baru seperti yang diperlihatkan pada gambar dibawah :

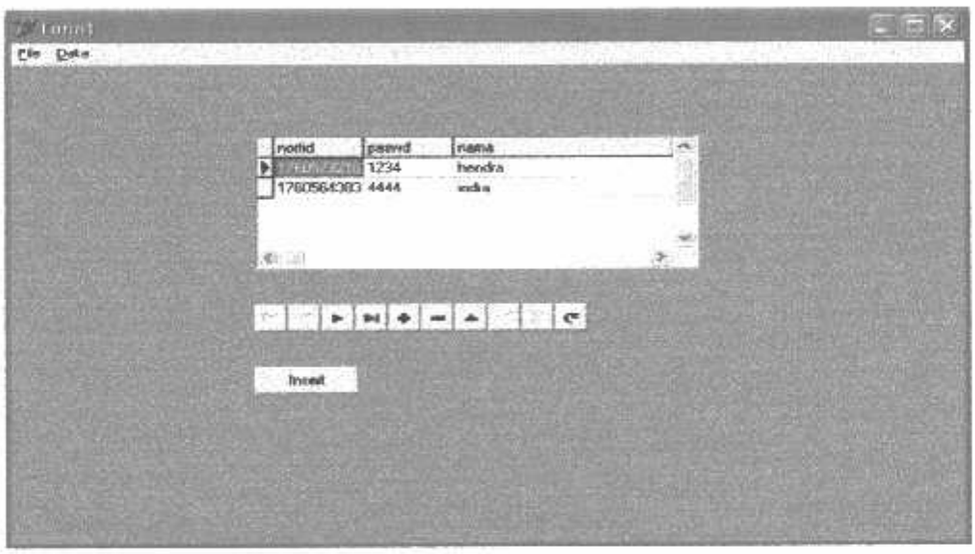


Gambar 4.21 Kotak Dialog Penambahan Identitas Baru.
Sumber : Pengujian

Setelah keluar tampilan kotak dialog seperti pada gambar 4.12 diatas dilanjutkan dengan penulisan. Seperti langkah berikut :

a. Menuliskan no Tag ID.
b. Menuliskan Password ID sesuai permintaan pelanggan.
c. Menuliskan Nama Pelanggan, alamat, serta nomor telepon.
d. Jumlah account prabayar.

c. Setelah melakukan penulisan sesuai langkah diatas, selanjutnya dilakukan penekanan tombol "*Insert*" yang tertera pada kotak dialog seperti pada gambar 4.12 untuk memasukkanya ke dalam *Database* dan menekann tombol "*cancel*" untuk membatalkanya. Sehingga akan tampil kotak dialog data base sebagai berikut :



Gambar 4.22 Kotak Dialog Hasil Penambahan Identitas Baru Pada *Database*
Sumber : Pengujian

4.8.6. Analisa Hasil Pengujian Sistem

Dari pengujian diatas menunjukkan bahwa identitas dari tag yang dituliskan pada kotak dialog pada program Delphi gambar 4.12 akan ditampilkan ke dalam display LCD saat mikrokontroller melakukan pendeteksian, dan pada penekanan nomor PIN pada keypad harus sesuai dengan password yang telah dituliskan pada kotak dialog program Delphi gambar 4.12 agar dapat melanjutkan transaksi.

Sedangkan yang ditunjukkan pada kotak dialog Delphi 4.13, terlihat adanya pengurangan account pelanggan setelah proses pengisian. Dan account terakhir akan ditunjukkan melalui tampilan LCD saat awal terjadi transaksi lagi.

Sistem pada simulasi SPBU ini menggunakan sebuah pompa yang yang dimana pengontrollan keluaran debitnya digunakan program timer/delay pada mikrokontroller. Program timer tersebut didapat dari beberapa sample percobaan seperti table dibawah:

Tabel 4.6 Pengujian Sampel Debit
Sumber : Pengujian

Percobaan	Jumlah Liter	Waktu pada Pompa	Jumlah Liter Output	Selisih Per Liter
I	1 Liter	188 detik	1,033	0,033
II	2 Liter	377 detik	1,9835	0,0165
III	3 Liter	542 detik	3,0385	0,0385
IV	4 Liter	728 detik	3,978	0,022

Dari data sample diatas maka untuk timer pemrograman mikrokontroller digunakan 182 detik/liter, sehingga didapat perhitungan pada program 5,49 mL/detiknya. Dan juga didapat rata-rata selisih per liter sama dengan 0,0275 liter/27,5 mL.

Persentase kesalahan pada driver motor pompa apabila diberi lamanya waktu pengisian per-liternya dengan hasil per-liter yang dikeluarkan dan kelipatannya.

$$\%Error = \frac{Rata - rataSelisih(LiterSebenarnya - LiterOutput)}{LiterSebenarnya} x100\%$$

$$= \frac{0,0275}{1L} x100\% = 2,75\%$$

Sehingga didapatkan error pada pengisian per-liternya adalah:

Jumlah liter x %kesalahan = 2,75% . 1000 mL = 27,5 mL


INSTITUT TEKNOLOGI NASIONAL
ITN
MALANG

BAB V

KESIMPULAN DAN SARAN

5.1 Kesimpulan

Berdasarkan perencanaan dan pembuatan sistem kartu prabayar pada SPBU menggunakan Teknologi *RFID* dapat diambil kesimpulan sebagai berikut:

1. Sistem ini dapat berjalan sebagaimana mestinya software Delphi dapat melakukan select, update Database, dan melakukan komunikasi serial dengan baik, sedangkan Hardware dapat menampilkan menu, mengontrol, dan melakukan komunikasi dengan baik.
2. Pembacaan RFID tag oleh Reader dapat mencapai jarak 7 cm. Sedangkan jark yang bagus dalam pembacaan adalah 5 cm
3. Serta error pengisian tiap liternya adalah 2,75 %. Yaitu sekitar 27,5 ml per pengisian 1 liter.
4. *Database Delphi* berfungsi sebagai tempat menyimpan no *ID*, identitas pelanggan, pengaturan no PIN, jumlah *account voucher* yang diisikan dan semua perhitungan transaksi pembelian.
5. Sistem ini dapat menggantikan metode konvensional dalam pengisian di SPBU menjadi Self-service.

5.2 Saran

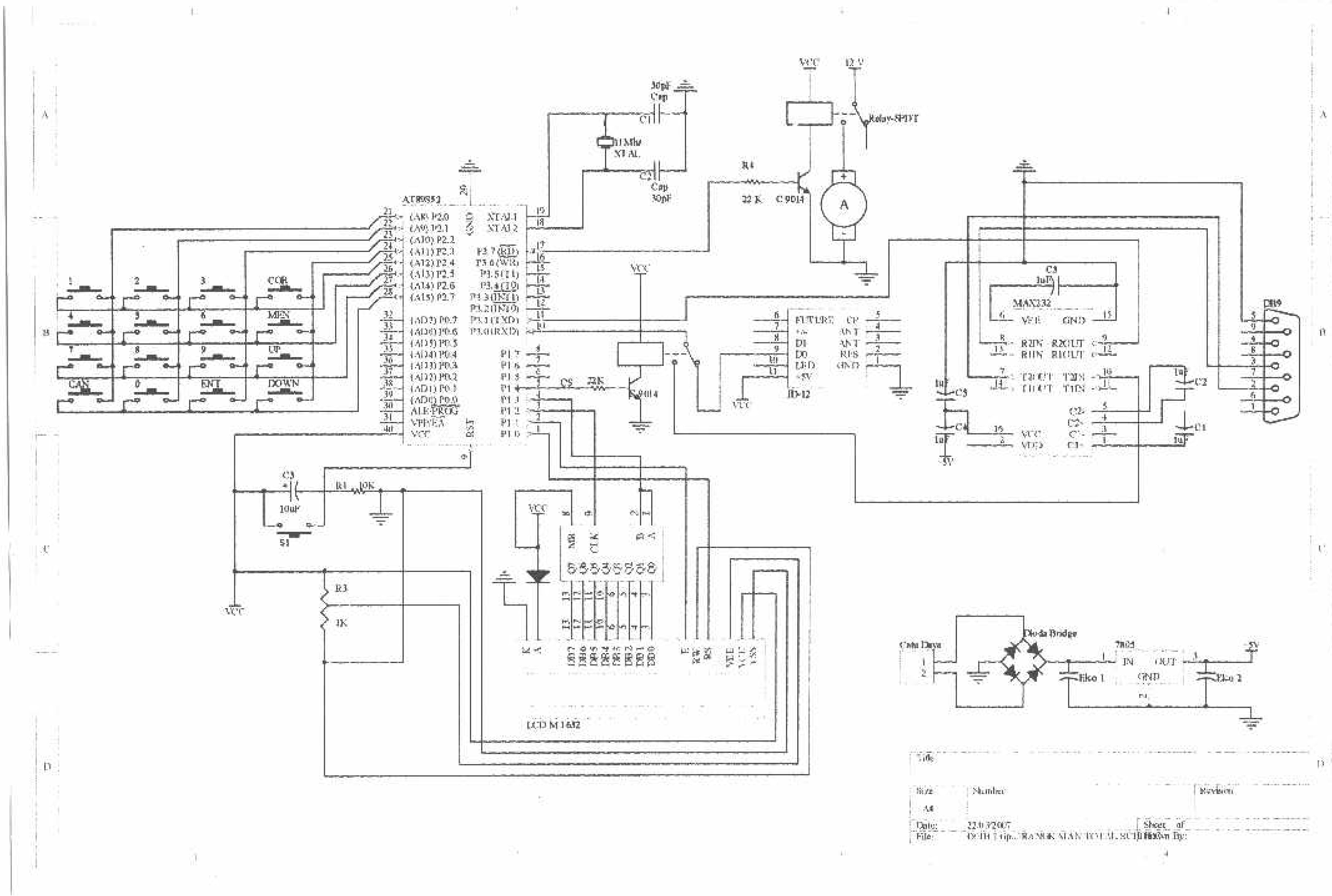
Tujuan utama dari penulisan adalah bagaimana membuat suatu sistem kartu prabayar untuk SPBU dimana konsumen dapat memilih jumlah liter maupun jumlah rupiah sendiri.Saran-saran yang bisa digunakan untuk pengembangan alat ini secara lanjut antara lain:

1. Implementasi secara nyata perlu dilakaukan riset kontrol dengan mesin pengisian BBM yang nyata.
2. Perangkat yang dibuat akan lebih akurat dan lebih presisi lagi jika menggunakan komponen-komponen yang kualitas dan kemampuannya lebih baik.
3. Pengembangan dalam hal pembuatan kartu voucher bisa menggunakan alternatif lain misal : Kartu magnetic, Barcode Reader dsb, yang tingkat efisien yang lebih sesuai dengan pemakaian.
4. Pengembangan pada simulasi SPBU agar lebih presisi sebaiknya digunakan sensor mengenai aliran air, pembesaran tendon agar debit air lebih banyak dan juga mekatronika yang baik sebagai penunjang.

DAFTAR PUSTAKA

1. Atmel. Co, *Data Sheet 8 bit Microcontroller with 4 K bytes Flash AT89S52*, 1998.
2. Agfianto Eko P, *Belajar Mikrokontroller AT89C51/52/55 Teori dan Aplikasi*, Gavamedia, Yogyakarta, 2006
3. M Agus J. Alam, *Belajar Sendiri Borland Delphi 6 & 7*, PT. Elex Media Komputindo, Jakarta 1989.
4. Rivas,Mario, *RFID – its Applications and Benefits*, Philips, 2004
5. ID Series Datasheet *Advanced Digital Reader Technology*, 2004.
6. DM 72LS164 *Data sheet National Semiconduktor*, 1982.
7. LCD *Module User Manual, Seiko Instruments Inc*, 1985.
8. RS 232 *Data sheet Maxim-ic*, www. Maxim-ic.com.
9. Ibnu Malik, Moh dan Anistardi, Bereksperimen dengan Mikrontroller 8031, PT. Elex Media Komputindo, Jakarta, 1997.
10. Frederick J. Bueche, *Fisika*, Edisi kedelapan, Erlangga, Jakarta, 1989
11. Malvino, Albert Paul, Prinsip-Prinsip Elektronik, Edisi ke 2, PT. Erlangga, Jakarta, 1981.
12. Wasito S., Vademekum Elektronika, Edisi kedua, PT Gramedia, Jakarta, 1995.

LAMPIRAN



AT89S52
XTAL1
XTAL2
GND
RST
VCC
ALE/PROG
11Mhz
XTAL
30pF
Cap
C1
C2
R4
22 K
C 9014
Relay-SPDT
12 V
VCC
MAX232
DB9
ID-12
LCD M 1632
Dioda Bridge
7805
IN
OUT
GND
Elco 1
Elco 2
5V
COR
MEN
UP
DOWN
ENT
CAN
S1
10uF
R1
10K
R3
1K
Title
Size
A4
Number
Revision
Date:
22/03/2007
File:
Sheet of
Drawn By:

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

Nama : MAHARDIAN MAHENDRADHATA
NIM : 02.17.019
Masa Bimbingan : 10 Oktober 2006 - 10 April 2007
Judul Skripsi : RANCANG BANGUN SISTEM LAYANAN SPBU PRABAYAR MENGGUNAKAN TEKNOLOGI RFID

No	Tanggal	Uraian	Paraf Pembimbing
1	27-01-2007	BAB I, BAB II, BAB III	
2	9-02-2007	REVISI PENULISAN SUMBER & FLOWCHART	
3	16-02-2007	BAB IV	
4	26-02-2007	REVISI HASIL PENGUJIAN, GAMBAR DISPLAY, UJI KESALAHAN/ ERROR	
5	2-3-2007	BAB V & MAKALAH SEMINAR HASIL	
6	6-3-2007	REVISI KESIMPULAN & SARAN	
7	9-3-2007	SEMUA BAB KELENGKAPAN KESELURUHAN	
8	10-3-2007	ACC ICOMPAC	
9			
10			

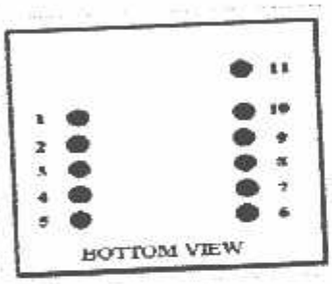
Malang,
Dosen Pembimbing

Joseph Dedy Irawan, ST. MT
Nip. 132 315 178

Form S-4b

dan aplikasi perangkat lunak untuk menyimpan data pada *server database*. Pada prakteknya *tag reader* dapat berupa perangkat keras yang terletak pada suatu tempat yang tetap. Pada aplikasinya *tag reader* dapat membaca sendiri *tag* yang dideteksi *(smart self). Tag reader smart self* dapat mendeteksi ketika ada penambahan *tag* atau ada *tag* yang keluar. Pada dasarnya *tag reader* merupakan suatu peralatan yang sederhana dan dapat digabungkan kedalam perlengkapan *mobile* seperti telepon selular atau PDAs.

Saluran (*channel*) dari *reader* ke *tag* disebut dengan saluran *forward* (*forward channel*), saluran dari *tag* ke *reader* disebut dengan saluran *bacward* (*bacward channel).*



Gambar 2.2 Gambar ID-12
Sumber : Data Sheet ID-12

Spesifikasi Reader ID-12 :

- Power Requirement : 5V@13mA nominal
- Card Format : Temec Q5555 or compatible
- Frequency : 125KHz
- Encoding : Manchester 62bit,modulus 64
- I/O Output Current : 20mA sink/source
- Drive Current : 300mA
- Antenna Volt : 100 Volt PKPK