

# **SKRIPSI**

## **PERANCANGAN DAN PEMBUATAN TRAFFIC LIGHT TERINTEGRASI MELALUI KOMUNIKASI SERIAL BERBASIS AT89S51**



**Disusun Oleh :**

**ROFI ARISANDI**

**05.12.204**

**JURUSAN TEKNIK ELEKTRO S-1  
KONSENTRASI TEKNIK ELEKTRONIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI NASIONAL MALANG  
2010**

3010

PERUSAHAAN TEKNOLOGI INFORMATIKA  
ALTERNATIF TEKNOLOGI MADRASAH  
KONVENSIONAL DENGAN EFEKTIVITAS  
SISTEM BERSAMA SISTEM 2-1

02'15'301

DATA VENTURE

SYSTEM (10) :

REKAM BERKAS VIGOR

REKAM BERKAS VIGOR BERKAS VIGOR  
REKAM BERKAS VIGOR BERKAS VIGOR  
REKAM BERKAS VIGOR BERKAS VIGOR

02'15'301

**PERANCANGAN DAN PEMBUATAN  
TRAFFIC LIGHT TERINTEGRASI MELALUI  
KOMUNIKASI SERIAL BERBASIS AT89S51**

**SKRIPSI**

**Disusun Oleh :  
ROFI ARISANDI  
05.12.204**



**JURUSAN TEKNIK ELEKTRO S-1  
KONSENTRASI TEKNIK ELEKTRONIKA  
FAKULTAS TEKNOLOGI INDUSTRI  
INSTITUT TEKNOLOGI NASIONAL MALANG  
2010**



LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN *TRAFFIC LIGHT*  
TERINTEGRASI MELALUI KOMUNIKASI SERIAL  
BERBASIS AT89S51

SKRIPSI

*Disusun dan Diajukan Sebagai Salah Satu Syarat Untuk Memperoleh  
Gelara Sarjana Teknik Elektronika Strata Satu (S-1)*

Disusun Oleh :

**ROFI ARISANDI**

NIM : 05.12.204

Diperiksa dan Disetujui

Dosen Pembimbing



**I. Komang Somawirata, ST, MT**

NIP.P. 103 0100 361

Mengetahui,

**Ketua Jurusan Teknik Elektro S-1**



**I. F. Fudi Limpraptono, MT.**

NIP.Y. 103 9500 274

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

**2010**





INSTITUT TEKNOLOGI NASIONAL MALANG  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO S-1  
Jl. Karanglo km 2, Malang

**BERITA ACARA UJIAN SKRIPSI  
FAKULTAS TEKNOLOGI INDUSTRI**

Nama : Rofi Arisandi  
NIM : 05.12.204  
Jurusan : Teknik Elektro S-1  
Konsentrasi : Teknik Elektronika  
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN *TRAFFIC LIGHT*  
TERINTEGRASI MELALUI KOMUNIKASI SERIAL  
BERBASIS AT89S51**

Dipertahankan di hadapan Majelis Penguji Skripsi Jenjang Strata Satu (S-1) pada :

Hari : Selasa  
Tanggal : 09 Februari 2010  
Dengan Nilai : 88,8 (A) *By*



**Ketua Majelis Penguji**  
Ir. H. Sidik Noertjahjono, MT.  
NIP.Y 102 8700 163



**Sekretaris Majelis Penguji**  
Ir. F. Yudi Limpraptono, MT.  
NIP.Y 103 9500 274

**Penguji I**

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

**Penguji II**

Sotyohadi ST.  
NIP.Y. 103 9700 309

## ABSTRAK

### PERANCANGAN DAN PEMBUATAN *TRAFFIC LIGHT* TERINTEGRASI MELALUI KOMUNIKASI SERIAL BERBASIS AT89S51

Rofi Arisandi, 0512204

Jurusan Teknik Elektro S-1, Konsentrasi Elektronika, Fakultas Teknologi Industri  
Dosen Pembimbing : I. Komang Somawirata, ST, MT.

Kata Kunci : Mikrokontroler AT89S51, EEPROM, Serial

Sistem lampu pengatur jalan (*traffic light*). Merupakan salah satu komponen dari sistem transportasi yang memiliki peranan penting dalam menjaga jalannya arus lalu lintas kendaraan. Namun semakin banyaknya kendaraan yang ada di jalan sering terjadi kemacetan. Pada perancangan ini akan dirancang sistem pengatur *traffic light* yang saling terhubung melalui komunikasi serial.

Sistem pengontrol terdiri dari 3 buah mikrokontroler yang terdiri dari sebuah mikrokontroler master dan 2 buah mikrokontroler slave. Mikrokontroler master sebagai pengendali utama yang akan memberikan perintah pada mikrokontroler slave melalui komunikasi serial RS485. Mikrokontroler slave akan menjalankan program *traffic light default* jika tidak ada perintah perubahan nilai *timer traffic light*. Pada sistem ini pengontrol dapat melihat status *traffic* melalui LCD.

Pada saat ada perubahan nilai timer kondisi *traffic* tidak terpengaruhi. Dan nilai error kondisi pada sistem adalah sebesar 1,476%.

# **KATA PENGANTAR**

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

## **“PERANCANGAN DAN PEMBUATAN TRAFFIC LIGHT TERINTEGRASI MELALUI KOMUNIKASI SERIAL BERBASIS AT89S51”**

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 terselesainya Skripsi ini, penulis mengucapkan terima kasih kepada :

- Bapak Prof. Dr. Eng Ir. Abraham Lomi, MSEE. selaku Rektor Institut Teknologi Nasional Malang.
- Bapak Ir. Sidik Noetjahjono, MT selaku Dekan Fakultas Teknologi Industri Institut Teknologi Nasional Malang
- Bapak Ir. F.Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S1 Konsentrasi Teknik Elektronika.
- Bapak I Komang Somawirata, ST, MT selaku Dosen Pembimbing yang telah memberikan bimbingan, pengarahan, serta ilmu-ilmu yang sangat berharga sehingga 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 mohon maaf kepada semua pihak bila mana selama penyusunan skripsi ini penyusun membuat kesalahan secara tidak sengaja dan semoga skripsi ini dapat bermanfaat bagi kita semua.

Malang, 2010

Penulis

# DAFTAR ISI

	<b>Halaman</b>
<b>HALAMAN JUDUL</b>	
<b>LEMBAR PERSETUJUAN</b> .....	i
<b>ABSTRAK</b> .....	ii
<b>KATA PENGANTAR</b> .....	iii
<b>DAFTAR ISI</b> .....	v
<b>DAFTAR GAMBAR</b> .....	x
<b>DAFTAR TABEL</b> .....	xiii
<b>DAFTAR LAMPIRAN</b> .....	xiv
<b>BAB I PENDAHULUAN</b>	
1.1. Latar Belakang .....	1
1.2. Rumusan Masalah .....	2
1.3. Batasan Masalah.....	2
1.4. Tujuan.....	3
1.5. Metodologi .....	3
1.6. Sistematika Pembahasan .....	4
<b>BAB II LANDASAN TEORI</b>	
2.1. Pendahuluan .....	6
2.2. LED .....	6

2.3. RS485 .....	7
2.4. Keypad .....	9
2.5. LCD ( <i>Liquid Crystal Display</i> ).....	10
2.5.1. Blok Diagram dari LCD YJ162A .....	12
2.5.2. Register .....	13
2.5.3. Encoder .....	14
2.6. Mikrokontroler AT89S51 .....	14
2.6.1. Perangkat Keras Mikrokontroler AT89S51 .....	15
2.6.2. Arsitektur Mikrokontroler AT89S51 .....	16
2.6.3. Konfigurasi Pin Mikrokontroler AT89S51 .....	18
2.6.4. Unit Pengolah Pusat (Central Processing Unit CPU).....	21
2.6.5. Organisasi Memori .....	22
2.6.6. Memori Program .....	23
2.6.7. Memori Data .....	24
2.6.7.1. Memori Data Internal .....	24
2.6.7.2. Memori Data Luar .....	31
2.6.8. Metode Pemrograman .....	31
2.7. EEPROM AT47C64 .....	29

### **BAB III PERANCANGAN DAN PEMBUATAN ALAT**

3.1. Pendahuluan .....	38
3.2. Perancangan Sistem Kerja.....	38
3.3. Perancangan Perangkat Keras .....	41
3.3.1. Rangkaian LED .....	41



3.3.2. Minimum Sistem AT89S51 (master) .....	43
3.3.3. Minimum Sistem AT89S51 (slave).....	47
3.3.4. Perancangan Rangkaian LCD .....	49
3.3.5. Perancangan Rangkaian Keypad .....	51
3.3.6. Perancangan Rangkaian RS 485 .....	53
3.3.7. Perancangan Rangkaian EEPROM AT24C64 .....	54
3.4. Perancangan Perangkat Lunak .....	56

## **BAB IV PENGUJIAN DAN HASIL ANALISA**

4.1. Pengujian Rangkaian Minimum Sistem Mikrokontroler AT89S51.....	64
4.1.1 Tujuan.....	64
4.1.2. Alat dan Bahan .....	64
4.1.3. Langkah – Langkah Pengujian .....	64
4.1.4. Hasil dan Analisa Pengujian .....	53
4.2. Pengujian Rangkaian Komunikasi Serial RS 485 .....	66
4.2.1 Tujuan.....	66
4.2.2. Alat dan Bahan .....	66
4.2.3. Langkah – Langkah Pengujian .....	66
4.2.4. Hasil dan Analisa Pengujian .....	67
4.3. Pengujian Keypad .....	68
4.3.1. Tujuan.....	68
4.3.2. Alat dan Bahan .....	68
4.3.3. Langkah – Langkah Pengujian .....	68
4.3.4. Hasil dan Analisa Pengujian .....	69

4.4. Pengujian Rangkaian LCD .....	70
4.4.1. Tujuan.....	70
4.4.2. Alat dan Bahan .....	70
4.4.3. Langkah – Langkah Pengujian.....	70
4.4.4. Hasil dan Analisa Pengujian .....	71
4.5. Pengujian Timer .....	71
4.5.1. Tujuan.....	71
4.5.2. Alat dan Bahan .....	71
4.5.3. Langkah – Langkah Pengujian.....	72
4.5.4. Hasil Pengujian .....	72
4.5.5. Analisa Hasil Pengujian .....	73
4.6. Pengujian Keseluruhan Sistem.....	75
4.6.1. Tujuan.....	75
4.6.2. Alat dan Bahan .....	75
4.6.3. Langkah – Langkah Pengujian.....	75
4.6.4. Hasil dan Analisa Pengujian .....	76
4.6.4.1. Hasil dan Analisa Pengujian Traffic 1 .....	79
4.6.4.1. Hasil dan Analisa Pengujian Traffic 2 .....	80
4.6.4.1. Hasil dan Analisa Pengujian Traffic 3 .....	82
4.6.4.1. Hasil dan Analisa Pengujian pengiriman dan Penerimaan Serial ....	83

## **BAB V PENUTUP**

5.1 Kesimpulan.....	85
5.2 Saran.....	85

**DAFTAR PUSTAKA ..... 86**

**LAMPIRAN ..... 87**



## DAFTAR GAMBAR

	<b>Halaman</b>
Gambar 2.1 Simbol LED.....	6
Gambar 2.2 Typical Operating Circuit.....	8
Gambar 2.3 IC MAX 485.....	8
Gambar 2.4 Keypad Matrik 4x4.....	9
Gambar 2.5 Blok Diagram Dari YJ162A.....	12
Gambar 2.6 Blok Diagram Mikrokontroler AT89S51 .....	16
Gambar 2.7 Konfigurasi Pin Mikrokontroler AT89S51 .....	18
Gambar 2.8 Stuktur Memori Program dan Data pada AT89S51 .....	22
Gambar 2.9 Memori Program .....	23
Gambar 2.10 Memori data Internal .....	25
Gambar 2.11 Pemrograman Mode Parallel dan Mode Serial AT89S51 .....	33
Gambar 2.12 EEPROM AT24C64.....	34
Gambar 2.13 Penulisan Secara Byte .....	35
Gamabr 2.14 Penulisan Secara Page .....	35
Gambar 2.15 Pembacaan Data Secara Current Read .....	36
Gambar 2.16 Pembacaan Data Secara Random Read.....	36
Gambar 2.17 Pembacaan Data Secara Sequential Read.....	37
Gambar 3.1 Diagram Blok Sistem .....	40
Gambar 3.2 Rangkaian LED .....	42
Gambar 3.3 Minimum Sistem AT89S51 ( <i>master</i> ) Dan Alokasi Pin .....	44

Gambar 3.4 Rangkaian Reset dan Osilator pada Minimum Sistem AT98S51 .....	46
Gambar 3.5 Minimum Sistem AT89S51 ( <i>slave</i> ) Dan Alokasi Pin .....	48
Gambar 3.6 Rangkaian LCD.....	50
Gambar 3.7 Susunan Baris dan kolom Keypad 4x4 .....	52
Gambar 3.8 Port Keypad pada Minimum Sistem AT89S51.....	52
Gambar 3.9 Rangkaian RS 485 .....	54
Gambar 3.10 Rangkaian EEPROM AT24C64.....	55
Gambar 3.11 Flow chart Proses Jalanya Sistem .....	57
Gambar 3.12 Flow chart Proses Pembacaan Keypad.....	59
Gambar 3.13 Flow chart Komunikasi Serial RS 485 .....	60
Gambar 3.14 Flow chart Seting Delay Default .....	61
Gambar 3.15 Flow chat Seting Yellow .....	62
Gambar 3.16 Flow chart Proses Inisialisasi LCD .....	63
Gambar 4.1. Diagram Blok Pengujian Mikrokontroller.....	65
Gambar 4.2. Blok Diagram Pengujian Serial RS 485 .....	67
Gambar 4.3. Hasil Pengujian Serial Pada Mikrokontroller slave .....	67
Gambar 4.4. Blok Diagram Pengujian Keypad.....	68
Gambar 4.5. Blok Diagram Pengujian LCD .....	70
Gambar 4.6. Hasil Pengujian LCD.....	56
Gambar 4.7. Tampilan LCD “Judul” .....	76
Gambar 4.8. Tampilan LCD “Disusun dan Nama” .....	77
Gambar 4.9. Pilihan Menu Pada Sistem .....	77
Gambar 4.10. Tamplan LCD Pilihan Menu”Seting Traffic” .....	78

Gambar 4.11. Tampilan LCD Menu “Yellow Light” .....	78
Gambar 4.12. Denah Penempatan Traffic Light .....	79
Gambar 4.13. Hasil Pengiriman Data Serial Melalui RS485 .....	83
Gambar 4.14. Hasil Penerimaan Status Timer Traffic .....	84

## DAFTAR TABEL

	<b>Halaman</b>
Tabel 2.1. Pin-pin LCD dan Konfigurasinya .....	11
Tabel 2.2. Fungsi Khusus pada Port 1 .....	19
Tabel 2.3. Fungsi Khusus pada Port 3 .....	19
Tabel 2.4. Special Fungsi Registrar (SFR) .....	26
Tabel 4.1. Hasil Pengujian Sistem Mikrokontroler .....	65
Tabel 4.2. Hasil Pengujian Keypad .....	69
Tabel 4.3. Hasil Pengujian Timer.....	72
Tabel 4.4. Perbandingan Hasil Pengujian dan Analisa Pengujian Timer.....	74
Tabel 4.5. Hasil Pengujian Traffic 1 .....	79
Tabel 4.6. Hasil Pengujian Traffic 2 .....	81
Tabel 4.7. Hasil Pengujian Traffic 3 .....	82

## DAFTAR LAMPIRAN

Lampiran 1 Formulir Bimbingan Skripsi

Lampiran 2 Formulir Perbaikan Skripsi

Lampiran 3 Skematik Alat

Lampiran 4 Foto Alat

Lampiran 5 *Listing Program*

Lampiran 6 DataSheet

# BAB I

## PENDAHULUAN

### 1.1. Latar Belakang

Perkembangan ilmu pengetahuan dan teknologi pada saat ini semakin berkembang dengan pesat seiring dengan perkembangan zaman terutama dibidang elektronik. Salah satu perkembangan yang paling menonjol saat ini adalah perkembangan sistem control. Dengan adanya sistem ini dapat memudahkan kita mengontrol peralatan elektronik. Kenyamanan pengguna alat control menjadikan setiap aspek yang berada disekitar kita.

Sistem lampu pengatur jalan (*traffic light*). Merupakan salah satu komponen dari sistem transportasi di dunia pada umumnya dan di Indonesia pada khususnya, yang memiliki peranan penting dalam menjaga jalanya arus lalu lintas kendaraan. Namun semakin banyaknya kendaraan yang ada di jalan sering terjadi kemacetan pada jam-jam sibuk orang berangkat sekolah, kerja, kuliah atau pekerjaan yang lainnya ataupun pada hari libur nasional, dimana banyak orang bepergian. Hal ini menyebabkan pengaturan lalu lintas cukup melelahkan.

Bertitik tolak dari masalah-masalah diatas, maka dirancang “ *Traffic Light Terintegrasi Melalui Komunikasi Serial Berbasis AT89S51* ”. Dalam perancangan alat ini akan dijelaskan bagaimana alat ini dipasang pada lampu lalu lintas (*Traffic Light*), mengirimkan data dari mikrokontroler *master* dan menerima data dari mikrokontroler *slave*.

## 1.2. Rumusan Masalah

Permasalahan yang ditangani pada pembuatan skripsi ini adalah karena pada saat ini lampu lalu lintas (*traffic light*) telah banyak digunakan namun masih dikontrol secara terpisah. Dan bagaimana agar dapat membuat sebuah alat yang mudah digunakan, dan memiliki fitur sesuai dengan kebutuhan dengan memanfaatkan Mikrokontroler AT89S51 dan memprogram Mikrokontroler tersebut agar bisa diaplikasikan pada alat pengatur lampu lalu lintas (*traffic light*) serta ditampilkan pada LCD dan mengkondisikan sesuai dengan kondisi yang diinginkan. Masalah yang ditangani ialah pembuatan perangkat keras (*hardware*) adalah lebih mengfungsikan Mikrokontroler AT89S51.

Sehubungan dengan permasalahan diatas, maka dalam skripsi ini dipilih judul:

### **PERANCANGAN DAN PEMBUATAN *TRAFFIC LIGHT* TERINTEGRASI MELALUI KOMUNIKASI SERIAL BERBASIS AT89S51**

## 1.3. Batasan Masalah

Dalam menyusun tugas akhir ini diperlukan suatu batasan masalah agar tidak menyimpang dari ruang lingkup yang akan dibahas. Adapun batasan masalahnya:

- Menggunakan Mikrokontroler Atmel AT89S51
- Menggunakan MAX RS485 sebagai komunikasi serial antar mikrokontroler.

- Menggunakan 2 buah persimpangan 4 jalur dan 1 buah persimpangan 3 jalur sebagai simulasi yang akan dikendalikan.
- Menggunakan bahasa pemrograman *assembler*.
- Tidak membahas software
- Tidak membahas jaringan dan power supply
- Tidak membahas penempatan alat.

#### **1.4. Tujuan dan Manfaat**

Adapun tujuan yang ingin dicapai dalam skripsi ini adalah :

- Mempermudah petugas lalulintas dalam proses pengaturan arus lalulintas.
- Mengaplikasikan mikrokontroller sebagai alat pengatur lampu lalulintas (*traffic light*).

#### **1.5. Metodologi**

Untuk mencapai dari skripsi ini, maka metodologi dalam skripsi ini adalah sebagai berikut :

##### ***a. Studi Literatur***

Dengan mempelajari beberapa kepustakaan yang mendukung, baik kepustakaan tentang perangkat keras dari sistem yang dibuat.

##### ***b. Perancangan Hardware dan Validasi Hardware***

Melakukan kegiatan pembuatan Alat dan Program *bascom*.



*c. Pengujian dan Analisa*

Setelah sistem selesai dibuat, maka diadakan pengujian untuk mendapat kekurangan dari sistem yang telah dirancang.

*d. Penyusunan Laporan*

Penyusunan laporan dibuat dengan mengikuti sistematika pembahasan yang telah ditetapkan

## **1.6. Sistematika Pembahasan**

Sistematika pembahasan dari skripsi ini terdiri dari pokok pembahasan yang saling berkaitan antara satu dengan lainnya, yaitu :

### **BAB I Pendahuluan**

Pada bab ini dibahas tentang latar belakang permasalahan, rumusan masalah, batasan masalah, sistematika pembahasan dari alat yang direncanakan.

### **BAB II Teori Dasar**

Pada bab ini dibahas tentang teori-teori yang mendukung dalam perencanaan dan pembuatan alat ini

### **BAB III Perencanaan Dan Pembuatan Alat**

Pada bab ini dibahas tentang perencanaan dan pembuatan keseluruhan sistem perangkat keras (hardware) dan perangkat lunak (software).

#### **BAB IV Pengujian Alat**

Pada bab ini dibahas tentang proses serta hasil dari pengujian alat, yang didasarkan oleh pengukuran-pengukuran.

#### **BAB V Penutup**

Pada bab ini akan disampaikan kesimpulan dari perencanaan dan pembuatan sistem ini.

## BAB II

### DASAR TEORI

#### 2.1. Pendahuluan

Dalam merencanakan dan merealisasikan sistem ini dibutuhkan pemahaman mengenai pengetahuan yang berhubungan dengan aplikasi. Pemahaman tersebut akan sangat bermanfaat dalam perancangan perangkat keras maupun perangkat lunak Pada bab ini akan dibahas mengenai teori penunjang dari peralatan yang direncanakan. Pokok pembahasan pada bab ini adalah :

1. LED
2. *Serial Interface RS 485*
3. *Keypad*
4. *LCD YJ162A*
5. Mikrokontroler ATMEL AT89S51
6. EEPROM AT24C64

#### 2.2. LED

LED (*Light Emitting Diode*) merupakan komponen yang dapat mengeluarkan emisi cahaya. LED merupakan temuan lain dari dioda, untuk mendapatkan emisi cahaya maka doping yang dipakai adalah gallium, arsenic, dan fosfor. Jenis doping yang berbeda akan menghasilkan warna cahaya yang berbeda.



**Gambar 2.1. Simbol LED**

Sumber : Widodo Budiharto, *Elektronika Digital*, Andi, Yogyakarta, 2005, h 45

Dalam memilih LED selain warna, perlu diperhatikan tegangan kerja, arus maksimum, dan disipasi daya. Hal ini dikarenakan akan mempengaruhi fungsi dari LED sebagai aplikasinya.

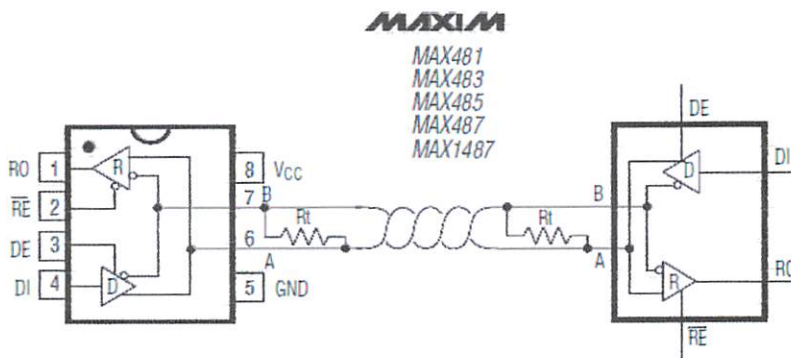
### 2.3. RS 485

RS-485 adalah interface yang bukan merupakan standar peralatan pada Home PC, tapi sangat umum dalam akusisdunia data. RS-485 akan mendukung 32 drivers dan 32 receivers (*bi-directional – half duplex* – komunikasi multidrop pada *single* atau *dual Twisted Pair Cable*). Sistem RS-485 dapat dikomunikasikan dalam model 2 atau 4 kabel. Panjang maksimal kabel dapat lebih dari 4000 kaki, dikarenakan penggunaan sistem transmisi voltase yang berbeda. Penggunaan yang umum untuk RS-485 adalah single PC yang dihubungkan kepada beberapa device yang berbagi dengan kabel yang sama. Dapat dikatakan RS-485 merupakan sistem komunikasi *party-lined*.

RS-485 memenuhi kebutuhan untuk jaringan komunikasi multipoint dan standarnya menspesifikasi sampai 32 driver dan 32 receiver pada suatu bus single (*2-wire*). Dengan pengenalan repeater dan impedansi tinggi driver/receiver secara otomatis, batasan ini dapat diperluas menjadi ratusan (bahkan ribuan) node pada suatu jaringan. RS-485 memperluas range mode untuk baik driver maupun receivers dalam *mode tri-state* dan dengan power off. Driver RS-485 juga dapat menyimpan masalah data *collision (bus contention)* dan kondisi *fault-bus*. Untuk menyelesaikan masalah data collision yang sering muncul dalam unit hardware jaringan multidrop (*converter, repeater, microprocessor control*) dapat dibentuk

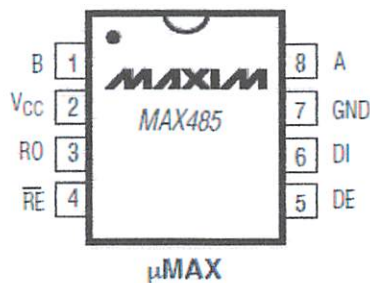
untuk di dalam suatu mode receive sampai mereka siap untuk mentransmisikan data. Sistem master single menawarkan alat sederhana dan langsung untuk menghindari data *collision* dalam *2 wire half duplex* sistem multidrop. Master ini mengawali permintaan komunikasi pada “*Slave Node*” yang ditujukan pada unit ini. Hardware ini mendeteksi start-bit dari transmisi dan secara otomatis mengaktifkan transmitter RS-485.

IC (*Integrated Circuit*) untuk serial RS-485 yang digunakan dalam Skripsi ini adalah jenis IC MAX 485 (Gambar 2.3.). IC ini memenuhi standar RS-485 yang dikeluarkan oleh *Maxim*.



**Gambar 2.2. Typical Operating Circuit**

Sumber : [Data Sheet Maxim Integrated Products](#)



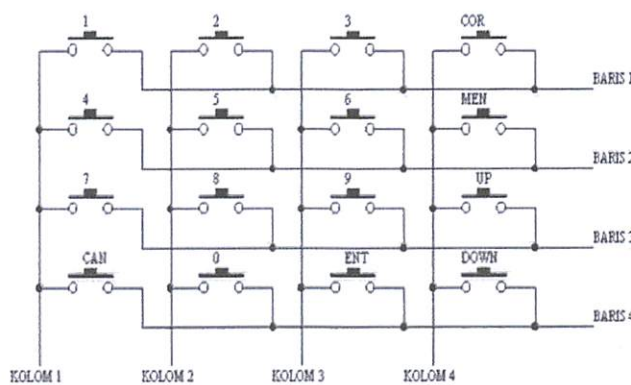
**Gambar 2.3. IC MAX 485**

Sumber : [Data Sheet Maxim Integrated Products](#)

Jika diperhatikan dalam Gambar 2.2, *output* dan *complementary output* dari *driver* maupun *receiver* digabung menjadi satu secara internal dalam IC. Ini menunjukkan bahwa IC MAX 485 dirancang untuk sistem *2-wire (half-duplex)*. Kelebihan IC MAX 485 adalah adanya antisipasi terhadap kondisi *fail-safe* yakni kondisi dimana bus jaringan berada dalam keadaan terbuka. Keadaan ini dapat disebabkan terputusnya jaringan atau karena seluruh *node* berada dalam keadaan siap menerima data sehingga tidak ada satu pun *driver* yang aktif. Secara otomatis *output* dari *receiver* memiliki logika 1, yang jika diterjemahkan dalam komunikasi serial adalah kondisi *idle*. Sifat ini menyebabkan IC MAX 485 tidak memerlukan tambahan resistor bias lagi yang biasanya digunakan untuk melakukan *pull up* dan *pull down* dalam bus jaringan, agar *output receiver* berada dalam kondisi *idle*.

## 2.4. Keypad

*Keypad* merupakan komponen yang digunakan sebagai sarana untuk memasukkan data ke komputer atau minimum system mikrokontroler.



**Gambar 2.4. Keypad Matrik 4x4**  
 Sumber : [www.wikipedia.org](http://www.wikipedia.org), keypad

Rangkaian ini dapat dianalogikan dengan empat buah kabel terbuka yang disilangkan dengan empat buah kabel terbuka yang lain (diletakan di atasnya). Perlakuan ini akan memperoleh 16 titik persilangan. bila pada suatu titik, kabel yang bersilangan itu disentuh (salah satu ditekan sehingga menyentuh kabel yang bersilangan dibawahnya), maka diasumsikan bahwa tombol *keypad* pada posisi persilangan tersebut ditekan.

## 2.5. LCD (*Liquid Crystal Display*)

*Liquid Crystal Display* adalah modul tampilan berkonsumsi daya yang relatif rendah dan terdapat sebuah controller CMOS didalamnya. Kontroler tersebut sebagai pembangkit karakter dari ROM/RAM dan *display* data RAM. Semua fungsi tampilan dikontrol oleh suatu intruksi dan modul LCD dapat dengan mudah untuk diinterfacekan dengan mikroprocessor/mikrocontroller. Input yang diperlukan untuk mengendalikan modul ini berupa bus data yang termultipleks dengan bus alamat dan 3 bit sinyal kontrol. Pengendali *dot matrik* LCD dilakukan secara internal pada modul LCD sendiri.

LCD merupakan suatu bentuk kristal cair yang akan beremulsi apabila dikenakan tegangan padanya. Tampilannya ini berupa *dot matrik* 5 x LCD sehingga jenis huruf yang dapat ditampilkan akan lebih banyak dan lebih baik resolusinya jika dibandingkan dengan 7 segment. Ciri-ciri dari LCD YJ162A:

- Terdiri dari 32 karakter yang dibagi menjadi 2 baris dengan display dot matrik 5 X 7 ditambah cursor
- Karakter generator ROM dengan 192 karakter

- Karakter generator RAM dengan 8 tipe karakter
- 80 X 8 bit display data RAM
- Dapat diinterfacekan dengan MPU 8 atau 4 bit
- Dilengkapi fungsi tambahan : Display clear, cursor home, display ON/OFF, cursor ON/ OFF, display character blink, cursor shift dan display shift
- Internal data
- Internal otomatis dan reset pada power ON
- +5 V power supply tunggal

Berikut ini merupakan pin-pin LCD beserta konfigurasi:

**Tabel 2-1. Pin-Pin LCD Dan Konfigurasinya**

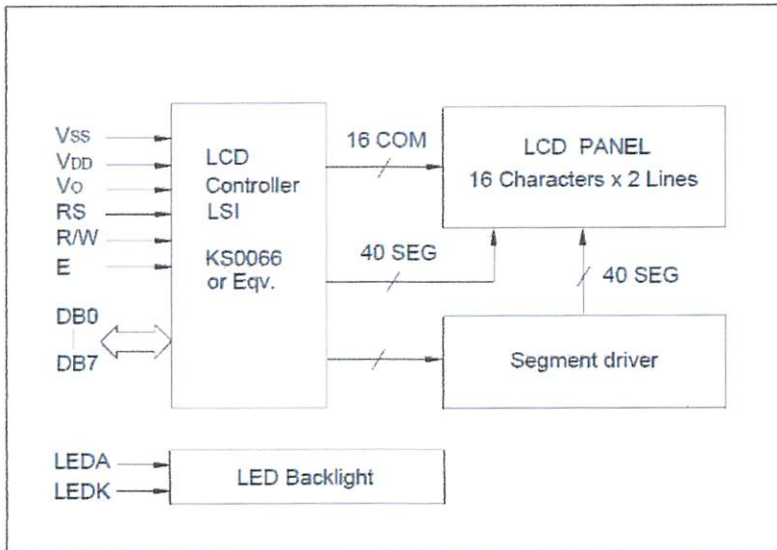
Sumber : LCD Module User Manual

Nama pin	jumlah	I/O	Tujuan	Fungsi
DB0-DB3	4	I/O	MPU	Trisate bidirectional lower data bus: data dibaca dari modul ke MPU atau dari MPU ditulis ke modul melalui bus
DB4-DB7	4	I/O	MPU	Tristatebidirectional upper fourdata bus: data dibaca dari modul ke MPU atau dari MPU ditulis ke modul melalui bus
E	1	Input	MPU	Sinyal operasi dimulai: sinyal aktif baca/tulis
R/W	1	Input	MPU	Sinyalpilih data dan tulis (0:tulis,1:baca)
RS	1	-	Power supply	Sinyal pilih register 0: Instruction register (write) Busy flag dan address



				counter (read) 1:Data register (write dan read)
Vo	1	-	Power supply	Penyetelan kontras pada tampilan LCD
VDD	1	-	Power supply	+ 5V
VSS	1	-	Power supply	Ground 0V
LEDA	1	-	Power supply	Led power supply (+5v)
LEDK	1	-	Power supply	Led power supply (0v)

**2.5.1. Blok Diagram dari LCD YJ162A**



**Gambar 2.5. Blok Diagram dari LCD YJ162A**

Sumber : [Data sheet LCD YJ162A](#)

### 2.5.2. Register

Control LCD mempunyai 2 register 8 bit yaitu *Instruction register* (IR) dan *Data Register* (DR). Kedua register tersebut dipilih melalui *Register Select* (RS). IR menyimpan kode instruksi seperti Display clear dan cursor shift, dan alamat informasi dari Display Data RAM (DD RAM) dan character generator RAM (CG RAM)

DR menyimpan data sementara untuk ditulis ke DD RAM atau CG RAM, atau dibaca dari DD RAM atau CG RAM. Ketika data ditulis ke DD RAM atau CG RAM dari MPU, data di DR secara otomatis ditulis ke DD RAM atau CG RAM dengan operasi internal. Tetapi ketika data dibaca dari DD RAM atau CG RAM maka alamat data ditulis pada IR. Data tersebut akan dimasukkan ke DR dan MPU akan membaca data dari DR, Setelah operasi pembacaan, alamat berikutnya diset data dari DD RAM atau CG RAM pada alamat tersebut akan dimasukkan ke DR untuk operasi berikutnya.

Display data RAM (DD RAM) mempunyai kapasitas area 80 X 8 bit. Beberapa area dari DDRAM yang tidak digunakan untuk display dapat digunakan sebagai General data RAM.

Pada LCD masing-masing pin mempunyai range alamat tersendiri, alamat itu diekspresikan dengan bilangan heksa. Untuk line 1 range alamat berkisar antara 00h-0fh sedangkan untuk line 2 alamat berkisar antara 40h-4fh

### **2.5.3. Encoder**

Informasi atau data ada kalanya dikodekan dalam bentuk kode. Untuk mengkode data digital tersebut digunakan suatu encoder. Salah satu contoh dari encoder adalah encoder 16 to 4 line yang digunakan untuk mengkode 16 saluran data desimal menjadi 4 saluran data BCD. Keluaran encoder merupakan kode dari salah satu input yang diaktifkan. Encoder mempunyai  $2^N$  masukan dan N buah line keluaran.

### **2.6. Mikrokontroler AT98S51**

Sebuah mikrokontroler selain memiliki CPU juga dilengkapi memori dan input output yang merupakan kelengkapan sebagai sistem minimum mikrokomputer sehingga sebuah mikrokontroler dapat dikatakan sebagai mikrokomputer dalam kepingan tunggal (*Single Chip Mikrokomputer*) yang dapat berdiri sendiri/stand alone.

Mikrokontroler AT89S51 adalah mikrokontroler ATMEL yang kompatibel dengan mikrokontroler keluarga MCS-51, yang membutuhkan data rendah dan memiliki performance yang tinggi dan merupakan mikrokontroler 8 bit yang dilengkapi 4 Kbyte EEPROM (Electrical Erasable Programmable Read Only Memory) dan 128 byte RAM internal, program memory dapat diprogram ulang dalam sistem atau menggunakan Programmer Nonvolatile Memory Konvensional. Dengan menggunakan sistem *ISP (In- System Programming)*, maka pengisian program dapat dilakukan secara *On The Fly* yaitu pengisian program secara langsung pada mikrokontroler yang sedang terpasang pada rangkaian aplikasi

Dalam sistem mikrokontroller terdapat 2 hal yang mendasar, yaitu perangkat keras (hardware) dan perangkat lunak (software) yang keduanya saling berkaitan dan mendukung satu sama lain.

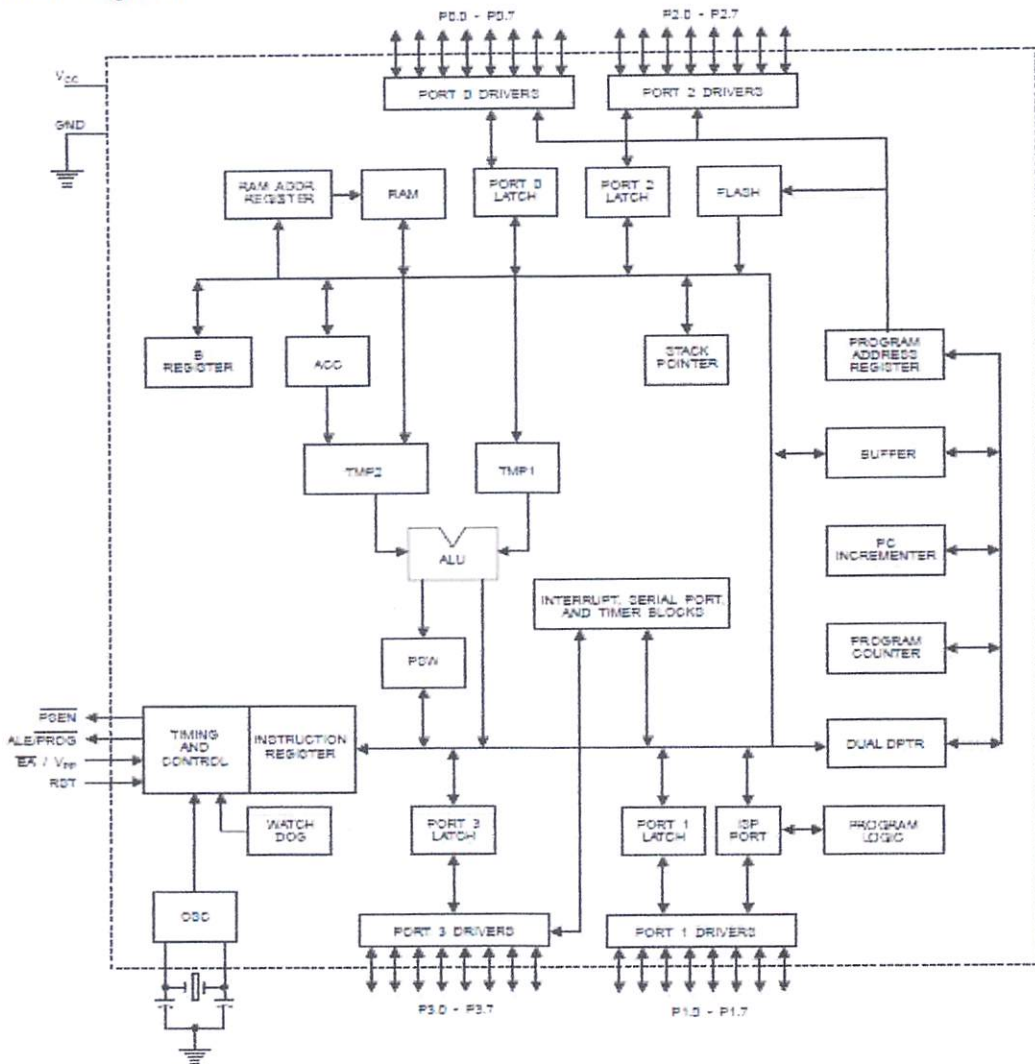
### **2.6.1. Perangkat Keras Mikrokontroller AT89S51**

Mikrokontroller AT89S51 memiliki :

- Sebuah CPU (Central Processing Unit).
- RAM (Random Access Memory).
- EEPROM.
- Empat buah Programmable Port I/O, masing-masing terdiri atas 8 jalur
- I/O yang dapat diprogram dan dapat dialamat per-bit/bit addressible.
- Dua buah Timer/Counter 16 bit.
- Sumber Interrupt.
- Port Komunikasi Serial.
- Osilator dan Clock.

## 2.6.2. Arsitektur Mikrokontroler AT89S51

Block Diagram



**Gambar 2.6. Blok Diagram Mikrokontroler AT89S51**

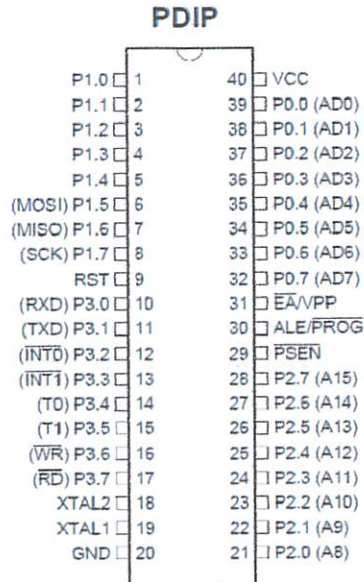
Sumber : Data Sheet Atmel AT 89S51

Arsitektur mikrokontroler AT89S51 adalah sebagai berikut :

1. Sesuai dengan keluarga mikrokontroler MCS 51 lainnya
2. Memiliki memori sistem terprogram (ISP) 4 Kbyte berjenis flash
3. Terdapat memori *flash* yang terintegrasi dalam sistem, dapat diulang 1000 kali

4. Jangkauan operasi tegangan antara 4.0 Volt sampai 5.5 Volt
5. Beroperasi statis penuh pada frekuensi 0 sampai 33 MHz
6. Terdapat tiga tingkat Kunci Memori Program
7. Terdapat 128 x 8 bit RAM internal
8. Terdapat 32 pin masukan/keluaran (*I/O*) yang dapat diprogram.
9. Memiliki 2 buah pewaktu 16 bit *timer/counter*
10. Memiliki 6 buah sumber penyelaan (*interupsi*)
11. Terdapat kanal UART (Universal Asynchronous receiver / transmitter)
12. Daya rendah saat mode menganggur (*idle*) dan pengaman dari tegangan jatuh (*Power Down Mode*)
13. Memiliki *Recovery* penyela dari mode daya jatuh
14. Memiliki pewaktu *Watchdog*
15. Bendera (penanda) daya mati
16. Waktu pemrograman cepat
17. Pemrograman ISP yang fleksibel (*Mode Byte* dan *Page*)

### 2.6.3. Konfigurasi Pin Mikrokontroler AT89S51



**Gambar 2.7. Konfigurasi Pin Mikrokontroler AT89S51**

Sumber : Data Sheet Atmel AT 89S51

Dari gambar tersebut dapat dijelaskan secara fungsional konfigurasi pin-pin AT89S51 sebagai berikut :

1. Pin 40 (VCC), merupakan *port* masukan tegangan positif (*power supply*).
2. Pin 20 (GND), merupakan *port* masukan tegangan negative.
3. Pin 32-39 (Port 0), merupakan 8 bit Port Input/Output dua arah (*bidirectional*), dikonfigurasi sebagai multipkeks bus alamat rendah/LSB (A0-A7) dan data selama pengaksesan program memory dan data memory eksternal. Pada mode ini port 0 mempunyai *pull up* dalam.
4. Pin 1-8 (Port 1), merupakan port input/output 8 bit dua arah (*bidirectional*) dengan *pull-up* dalam. Tiap pin pada port 1 dapat mengendalikan 4 masukan TTL. Bila logika 1 dikirim ke pin port 1, maka dapat digunakan

sebagai masukan dengan tahanan *pull-up* dalam. Pada mode pemrograman, port 1 digunakan sebagai alamat *byte* rendah. Dengan fungsi khusus seperti terlihat pada table 2-2.

**Table 2-2. Fungsi Khusus Pada Port 1**

Port	Fungsi Penggnti
P1. 5	<i>MOSI (Master data Output, Slave data Input untuk kanal SPI)</i>
P1. 6	<i>MISO (Master data Input, Slave data Output untuk kanal SPI)</i>
P1. 7	<i>SCK (Master Clock Output, Slave Clock Input untuk kanal SPI)</i>

- Pin 21-28 (Port 2), merupakan Port I/O dua arah (bidirectional) dengan internal pull-up, mengeluarkan address tinggi (MSB) selama pengambilan (fetch) program memory eksternal dan selama pengaksesan data memory. Port 2 mengeluarkan P@SFR (Special Function Register).Menerima address tinggi dan beberapa sinyal kontrol selama pemrograman dan verifikasi.
- Pin 10-17 (Port 3), merupakan Port I/O 8 bit dua arah (bidirectional) dengan internal pull-up, namun memepunyai beberapa fungsi khusus. Fungsi-fungsi khusus tersebut diuraikan dalam tabel 2-3.

**Tabel 2-3. Tabel fungsi khusus pada Port 3**

Sumber : [www.Atmel.com](http://www.Atmel.com), Data Sheet Book AT89S51

Port	Fungsi Khusus
P3.0	<i>RXD (Receive Data):</i> Port penerima data serial
P3.1	<i>TXD (Transmit Data):</i> Port pengiriman data serial
P3.2	$\overline{\text{INT0}}$ ( <i>Interupt 0</i> ): Port external Interupt 0 (masukan interupsi 0)



P3.3	$\overline{\text{INT1}}$ ( <i>Interrupt 1</i> ): Port external Interupt 1 (masukan interupsi 1)
P3.4	T0 ( <i>Timer 0</i> ): Port Eksternal timer 0 input (masukan dari pewaktu/pencacah (0).
P3.5	T1 ( <i>Timer 1</i> ): Port Eksternal timer 1 input (masukan dari pewaktu/pencacah (1).
P3.6	$\overline{\text{WR}}$ ( <i>Write</i> ): Sinyal penulisan data luar
P3.7	$\overline{\text{RD}}$ ( <i>Read</i> ): Sinyal Pembaca memori data luar

7. Pin 9 (RST), merupakan pin Input yang aktif tinggi (high), pin ini aktif tinggi selama siklus mesin, ketika osilator bekerja akan mereset peralatan.
8. Pin 18 (XTAL2), merupakan Output osilator dan dihubungkan dengan kristal bila menggunakan osilator internal.
9. Pin 19 (XTAL2), merupakan Input Osilator dan dihubungkan, dengan kristal bila menggunakan osilator internal.
10. Pin 29 ( $\overline{\text{PSEN}}$  = *Program Store Enable*), Pin ini berfungsi pada saat mengeksekusi program yang terletak pada memori eksternal. PSEN akan aktif dua kali setiap cycle.
11. Pin 30 ( $\overline{\text{ALE/PROG}}$  = *Address Latch Enable*), Pulsa input ALE digunakan untuk proses latching byte address rendah/LSB (A0-A7) selama pengaksesan ke eksternal memory. Pin ALE juga dipakai untuk pulsa program selama proses pemrograman memori *flash*.
12. Pin 31 ( $\overline{\text{EA/Vpp}}$  = *External Access Enable*), Pada kondisi low, pin ini akan berfungsi sebagai EA yaitu mikrokontroler akan menjalankan

program yang ada pada memori eksternal setelah sistem di-reset. Jika berkondisi high, pin ini akan berfungsi untuk menjalankan program yang ada pada memori internal. Pada saat flash programming pin ini akan mendapat tegangan 12 volt (VP).

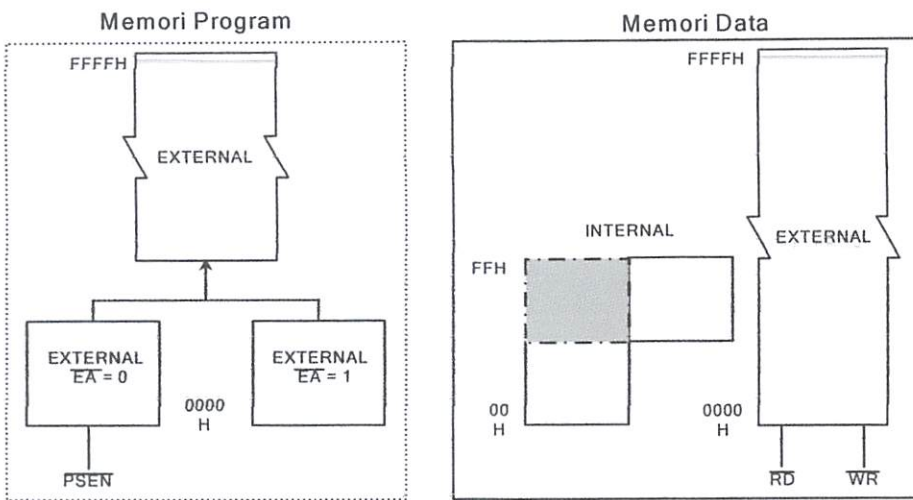
#### **2.6.4. Unit Pengolah Pusat (Central Processing Unit CPU)**

Unit pengolah pusat yang dimiliki oleh mikrokontroler AT89S51 merupakan CPU 8 bit yang dilengkapi oscilator internal. Oscilator internal yang dimiliki AT 89S51 secara teori mampu bekerja dari 0 MHz sampai 33 MHz. Pada prakteknya ada batas kerja minimum disebabkan oleh karena sifat CPU yang tidak statis. CPU banyak menggunakan rangkaian dinamik yang berupa sel-sel penyimpanan sementara yang dibentuk oleh rangkaian kapasitansi ke ground. Dengan frekwensi yang terlalu rendah isi sel penyimpanan tidak benar lagi karena adanya arus bocor. Akibatnya level tegangan pada simpul akan menurun dan tidak benar lagi isinya, sehingga CPU tidak mampu mengingat perintah yang baru saja dilaksanakan.

CPU mampu melaksanakan perintah dalam siklus mesin, tergantung jenis perintah yang dilaksanakan. Satu siklus mesin terdiri dari 12 periode oscilator, sehingga untuk frekwensi oscilator 12 MHz, satu siklus mesin membutuhkan waktu 1  $\mu$ s. Waktu pelaksanaan perintah terpendek adalah 1 siklus mesin dan yang terpanjang adalah 4 siklus mesin.

### 2.6.5. Organisasi Memory

Mikrokontroler AT89S51 memiliki ruang alamat memori data dan program yang terpisah, sebagaimana yang ditunjukkan pada gambar 2.8. Pemisahan memori program dan data tersebut membolehkan memori data diakses dengan alamat 8 bit, sehingga dapat dengan cepat dan dengan mudah disimpan dan dimanipulasi oleh CPU 8 bit. Namun demikian, alamat memori data 16 bit bisa juga dihasilkan melalui register DPTR (Data Pointer Register)



**Gambar 2.8. Stuktur Memori Program dan Data pada AT89S51**

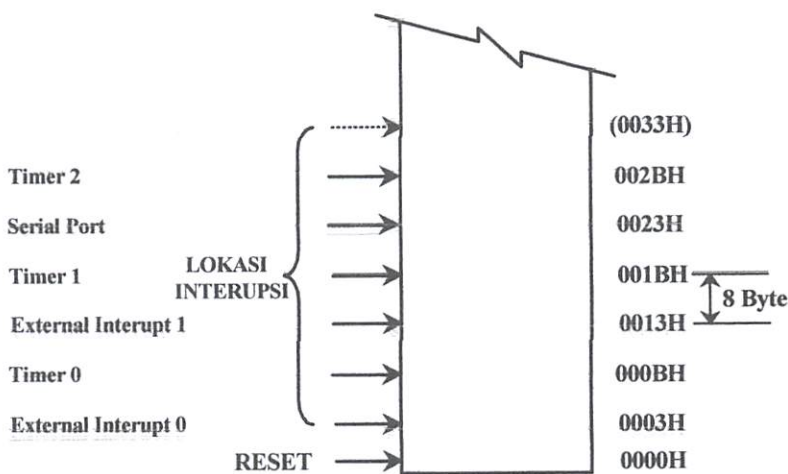
Sumber : Agfianto Eko Putra, Belajar Mikroprosesor AT89S51/52/55, Gaya Media, Yogyakarta, 2002,h 3

Memori program hanya dapat dibaca dan terdapat memori program yang bisa di akses langsung hingga 64K Byte. Sinyal pembacaan untuk membaca memori program luar yaitu menggunakan sinyal *Program Store Enable* ( $\overline{\text{PSEN}}$ ).

Memori data menempati suatu ruang alamat yang terpisah dari memori program. Memori data yang dapat diakses secara langsung dapat hingga 64K byte. CPU akan memeberikan sinyal baca ( $\overline{\text{RD}}$ ) dan tulis ( $\overline{\text{WR}}$ ), selama mengakses memori data.

### 2.6.6. Memori Program

AT89S51 memiliki memori program *Flash PEROM (Programmable and Erassable Read Only Memori)* sebesar 4K *byte*, yaitu ROM yang dapat ditulis ulang atau dihapus menggunakan sebuah perangkat programmer. Pada gambar 2.9. ditunjukkan pemetaan bagian bawah dari memori program. Setelah reset, CPU segera mengerjakan program mulai dari lokasi 0000H.



**Gambar 2.9. Memori Program**

Sumber : Agfianto Eko Putra, Belajar Mikroprosesor AT89S51/52/55, Gaya Media, Yogyakarta, 2002.h 4

Sebagaimana ditunjukkan pada gambar 2.9., masing-masing interupsi disimpan pada lokasi yang sudah tetap dalam memori program. Sebuah interupsi menyebabkan CPU melompat ke lokasi interupsi yang bersangkutan, yaitu letak subrutin layanan interupsi tersebut. Misalnya external interrupt 0 disimpan pada lokasi 0003H, jika interupsi ini digunakan maka layanan rutin interupsi ini harus dituliskan pada lokasi ini, jika tidak maka lokasi tersebut dipakai sebagai memori program serbaguna (untuk keperluan lain).

Lokasi-lokasi layanan interupsi tersebut menempati lokasi-lokasi dengan jarak 8 byte. 0003H untuk eksternal interupt 0, 000BH untuk timer 0, 0013H untuk eksternal interupt 1, dan seterusnya. Jika seluruh rutin layanan interupsi sangat pendek (kurang dari 8 byte), maka seluruh rutin bisa disimpan pada lokasi yang bersangkutan (sesuai dengan interupsi yang digunakan). Jika terlalu panjang (lebih dari 8 byte) maka harus digunakan suatu perintah lompat ke lokasi rutin interupsi yang sebenarnya (di lokasi dalam memori program).

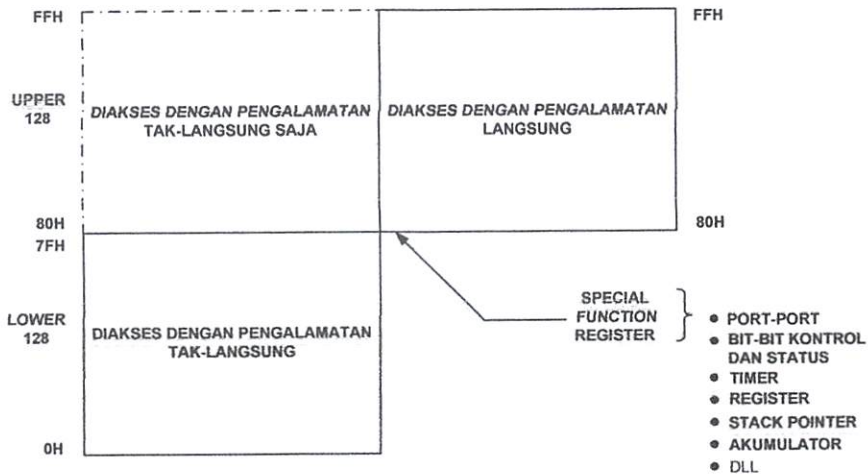
Alamat-alamat yang paling bawah dari memori program dapat berada dalam flash on-chip maupun eksternal. Untuk melakukan hal ini dilakukan pengkabelan pada pin EA atau External Access ke Vcc (akses memori internal) atau ke GND (akses memori eksternal) sesuai dengan kebutuhan.

### **2.6.7. Memori Data**

Memori data mikrokontroler AT89S51 memungkinkan memori data untuk diakses alamat 8 bit. Sekalipun demikian, alamat data memori 16 bit dapat dihasilkan melalui register *DPTR* ( *Data Pointer Register* ). Memori data terbagi menjadi dua, yaitu memori data *internal* (dalam) dan memori data *external* (luar) :

#### **2.6.7.1. Memori Data Internal**

Memori data *internal* dipetakan seperti pada gambar 2.10. Ruang memorinya dibagi menjadi 3 blok, yaitu sebagai lower 128, upper 128, dan ruang SFR (spesial function register).



**Gambar 2.10. Memori Data Internal**

Sumber : Agfianto Eko Putra, Belajar Mikroprosesor AT89S51/52/55, Gaya Media, Yogyakarta, 2002.h 7

Alamat memori data *internal* selalu 8-bit atau 1-byte, yang konsekuensinya hanya mampu mengalami hingga 256 byte. dimana 128 bytes bagian atas menempati alamat parallel ke *special function register* (SFR). Artinya 128 bagian atas mempunyai alamat yang sama dengan SFR namun secara fisik terpisah dari SFR. Ketika instruksi mengakses lokasi internal diatas 7FH, mode alamat yang digunakan pada instruksi menentukan apakah CPU mengakses 128 bytes atas atau SFR. Instruksi yang menggunakan pengalamatan langsung akan mengakses ruang SFR. Pengaksesan memori dapat dilakukan dengan langsung (*direct addressing*) atau pengaksesan secara tak-langsung (*indirect addressing*).

RAM dalam 128 byte bagian bawah dibagi menjadi 3 bagian :

1. 32 byte bawah dikelompokkan menjadi empat bank, masing-masing terdiri 8 register yang diberi nama R0...R7 dengan alamat 00H s/d 1FH.
2. 16 byte diatas bank register merupakan suatu blok ruangan memori yang dapat teralamat per-bit (*bit addressable*) dengan alamat 20H sampai dengan 2FH.

3. 80 byte RAM serba guna dengan alamat 30H s/d 7FH.

RAM internal 128 byte bagian atas digunakan sebagai register fungsi khusus yang disebut SFR (*Special Function Register*) yaitu register yang berhubungan dengan fasilitas-fasilitas perangkat keras pada mikrokontroler AT89S51. Register fungsi khusus (SFR) hanya dapat dialamati secara pengalamatan langsung dan sebagian register SFR dapat dialamati secara byte maupun bit.

Table 2-4. Social Function Register (SFR)

Simbol	Nama Register	Nilai Pada Saat Reset	Alamat
ACC	Accumulator	00 <sub>H</sub>	E0 <sub>H</sub>
B	Register B	00 <sub>H</sub>	F0 <sub>H</sub>
PSW	Program Status Word	00 <sub>H</sub>	D0 <sub>H</sub>
SP	Stack Pointer	07 <sub>H</sub>	81 <sub>H</sub>
DPTR	Data Pointer 2 Byte		
DPL	Bit rendah	0000 <sub>H</sub>	82 <sub>H</sub>
DPH	Bit Tinggi	0000 <sub>H</sub>	83 <sub>H</sub>
P0	Port 0	0FF <sub>H</sub>	80 <sub>H</sub>
P1	Port 1	0FF <sub>H</sub>	90 <sub>H</sub>
P2	Port 2	0FF <sub>H</sub>	A0 <sub>H</sub>
P3	Port 3	0FF <sub>H</sub>	B0 <sub>H</sub>
IP	Interupt Periority Control	XXX00000 <sub>B</sub>	D8 <sub>H</sub>
IE	Interupt Enable Control	0XX00000 <sub>B</sub>	A8 <sub>H</sub>
TMOD	Timer/Counter Mode Control	00 <sub>H</sub>	89 <sub>H</sub>
TCON	Timer/Counter Control	00 <sub>H</sub>	88 <sub>H</sub>
TH0	Timer/Counter 0 High Control	00 <sub>H</sub>	8C <sub>H</sub>
TL0	Timer/Counter 0 Low Control	00 <sub>H</sub>	8A <sub>H</sub>

TH1	Timer/Counter 1 High Control	00 <sub>H</sub>	8D <sub>H</sub>
TL1	Timer/Counter 1 Low Control	00 <sub>H</sub>	8B <sub>H</sub>
SCON	Serial Control	00 <sub>H</sub>	98 <sub>H</sub>
SBUF	Serial Data Buffer	Independen	99 <sub>H</sub>
PCON	Power Control		87 <sub>H</sub>

Berikut ini dibahas secara singkat fungsi dan sifat masing-masing register dalam register fungsi khusus (SFR):

### 1. Akumulator (ACC)

Merupakan register untuk penyimpanan data sementara, untuk operasi aritmatika dan logika. Instruksi mengacu sebagai register A, dan menempati lokasi E0H.

### 2. Register B

Register digunakan bersama akumulator untuk proses aritmatik, selain difungsikan sebagai register biasa, register ini menempati lokasi F0H

### 3. PSW (*Program Status Word*)

Merupakan register 8 bit (lokasi D0H), berisi bit-bit status yang berkaitan dengan kondisi CPU pada saat itu. Status yang tersimpan dalam PSW meliputi : *carry bit*, *the auxiliary carry* ( operasi BCD), 2-bit pemilih bank register, *overflow flag*, sebuah bit paritas, dan dua *flag* status yang bisa didefinisikan sendiri (*user-definable*).



#### 4. Stack Pointer (SP)

Merupakan register 8 bit (lokasi 81H) merupakan register dengan panjang 8-bit, digunakan dalam proses simpan dan ambil dari atau ke *stack*. Nilai dari *stack pointer* akan dinaikan sebelum disimpan bila diberi perintah *push* dan akan berkurang bila diberi perintah *pop*. Lokasi dari *stack pointer* ditentukan dengan *software*

#### 5. Data Pointer (DPTR)

DPTR terdiri dari dua buah register yaitu DPH untuk *byte* tinggi dan DPL untuk *byte* rendah, masing-masing berada di lokasi 83H dan 82H. bersama-sama membentuk register yang mampu menyimpan alamat 16-bit yang berfungsi untuk mengakses lokasi kode alamat dan lokasi alamat memori eksternal. DPTR dapat dimanipulasi sebagai register 16-bit atau sebagai register 8-bit yang terpisah.

#### 6. Port 0, Port 1, Port 2, dan Port 3

P0, P1, P2 dan P3 masing-masing menempati lokasi 80H, 90H, A0H, dan B0H, merupakan pengunci (*latches*), yang digunakan untuk menyimpan data yang akan dibaca atau ditulis dari atau ke Port, untuk masing-masing P0, P1, P2 dan P3.

#### 7. Serial Port UART (*Universal Asynchronous Receiver/Transmitter*)

Serial Port ini di kontrol oleh SCON (*Serial Port Control*) yang berlokasi di 98H, dan SBUF berlokasi di 99H, yang sebenarnya terdiri dari dua register yang terpisah yaitu register penyangga pengirim (*transmit buffer*)

dan penyangga penerima (*receive buffer*). Berfungsi menampung data sementara dari hasil penerimaan atau pengiriman data serial dari atau ke terminal data serial. Setelah serial port di konfigurasi (mode operasi dan *baud rate*), maka penulisan ke SBUF akan memulai pengiriman secara serial. Untuk dapat menggunakan serial port ini maka harus mengetahui bit-bit SFR yang bersangkutan.

#### 8. Timer Register

Merupakan register yang berfungsi sebagai pengatur ragam kerja pewaktu/pencacah pada mikrokontroler AT89S51. Register ini mempunyai dua buah 16 bit *timer/counter*, yaitu timer 0 dan timer 1. timer 0 terletak di alamat 8AH (TL0) dan 8CH (TH0), dan timer 1 terletak di alamat 8BH (TL1) dan 8DH (TH1).

#### 9. Timer mode

Merupakan register yang berada dikokasi 89H yang berfungsi untuk mengatur jenis timer yang digunakan (timer 0 atau timer1).

#### 10. PCON (*Power Control*)

Merupakan register yang berlokasi di 87H dan berfungsi untuk mengatur kebutuhan daya yang memungkinkan mikrokontroler ke dalam mode *idle* atau *sleep* yang mana akan lebih menghemat pemakaian daya. Selain itu pada PCON juga terdapat register untuk mengatur *Baud Rate* pada serial port.

## 11. Register Interupsi

Mikrokontroler AT89S51 memiliki 5 buah interupsi dengan dua level interupsi. Interupsi akan selalu nonaktif setiap kali system di reset. Register yang berhubungan dengan intrupsi adalah *Interrupt Enable Register* (IE) atau register pengaktif yang berada dilokasi A8H, dan berfungsi untuk mengatur keaktifan dari interupt dan *Interrupt Priority Register* (IP) atau register prioritas interrupt yang berada dilokasi B8H.

## 12. WDT (*Watchdog Timer*)

Timer ini merupakan metode *recovery* dalam suatu situasi ketika CPU mengalami gangguan dikarenakan oleh perangkat lunak. Timer ini terdiri dari suatu pencacah 14 bit dan SFR WDTRST (*Watchdog Timer Reset*). WDT dalam kondisi normal tidak di fungsikan. Untuk mefungsikannya pemakai harus menuliskan 01EH dan 0E1H dalam register WDTRST (SFR pada lokasi 0A6H). Saat WDT digunakan, timer akan naik setiap putaran mesin selama osilator sedang berjalan.

## 13. AUXR (*Auxiliary Register*)

Merupakan register tambahan dengan lokasi 8EH, yang digunakan untuk mengatur sendiri bit *Interrupt Enable* pada register IE (*Interrupt Enable*). Register ini berisi Bit (*Diseble/Enable ALE*), DISTRO (*Diseble/Enable Reset Out*) dan WDIDLE (*Diseble/Enable WDT dalam Mode Idle*).

#### 14. AUXR1 (*Auxiliary Register 1*)

Merupakan register tambahan berlokasi A2H, digunakan untuk memilih DPTR (*Data Pointer Register Select*), bila DPS=0, maka DPTR yang digunakan DP0L, DP0H, dan bila DPS=1, yang digunakan DP1L, DP1H.

#### 2.6.7.2. Memori Data Luar

Mikrokontroler AT89S51 memiliki program yang di baca dari *Flash* memori data luar, Port 0 (multiplex) bus alamat rendah/LSB (A0-A7) dan data selama pengaksesan program memory dan data memory eksternal, dan Port 2 sebagai bus alamat tinggi/MSB (A8-A15). Kapasitas dari memori data luar yang di hubungkan dapat mencapai 64Kbyte.

#### 2.6.8. Metode Pemrograman

Mikrokontroler AT89S51 mampu diprogram pada mode *high voltage* (12V) melalui mode *parallel programming*, adapun mode *low voltage* (5V) melalui mode *serial programming*. Pemrograman *Flash* mode serial dapat di program menggunakan antar muka ISP serial dengan menghubungkan RST ke VCC. Antarmuka ini terdiri dari SCK, MOSI sebagai masukan dan MISO sebagai keluaran . setelah RST di set tinggi, interrupt pengaktifan pemrograman perlu untuk dieksekusi sebelum operasi lain dapat dieksekusi. Frekuensi serial SCK maksimum kurang dari 1/16 dari frekuensi kristal. Apabila detak osilator 33 MHz frekuensi SCK maksimum adalah 2 MHz. Pemrograman mode serial dapat di lihat pada gambar 2.11.

Proses pemrograman secara serial adalah sebagai berikut :

#### 1. Urutan Power-Up

Berikan catu daya pada penyemat VCC dan Ground, set penyemat reset ke tinggi, jika kristal tidak di pasang pada XTAL1 dan XTAL2 maka diberi detak 3MHz sampai 33 MHz dan tunggu selama 10 milidetik sampai detak siap.

#### 2. Pengaktifan

Aktifkan *Serial Programming* dengan mengirim instruksi pengaktif pemrograman serial pada penyemat MOSI/P1.5. Frekuensi untuk penggeseran (*shift*) diberikan pada penyemat SCK/P1.7.

#### 3. Pemrograman

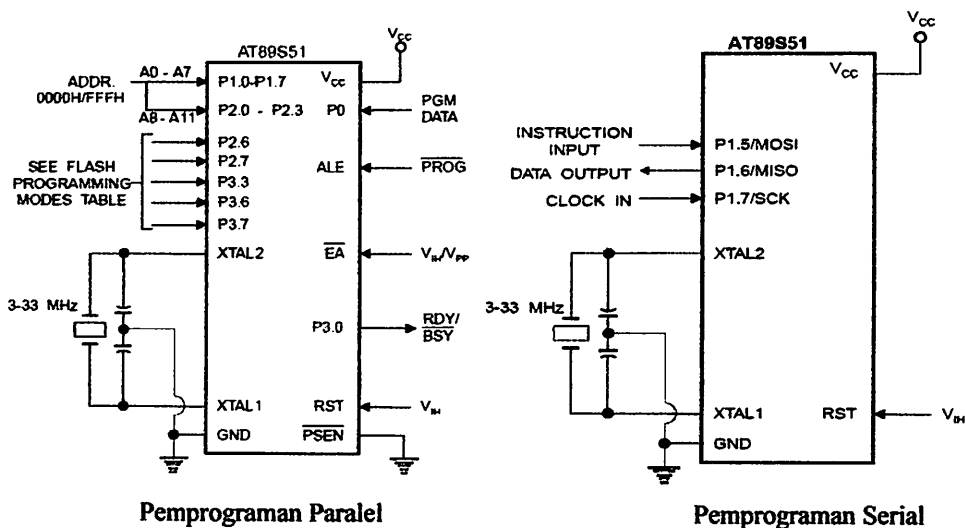
Array kode di program 1 byte pada waktu yang sama dalam mode byte atau *page* manapun.

#### 4. Vertifikasi

Suatu lokasi memori dapat di verifikasi sendiri dengan menggunakan instruksi pembacaan yang mengirimkan data pada alamat terpilih melalui penyemat MISO/P1.6 secara serial.

#### 5. Urutan Power-Off

Set XTAL1 ke rendah (jika tidak ada kristal yang terpasang). Buat RST ke rendah untuk membuat operasi mikrokontroler menjadi normal serta matikan catu daya.

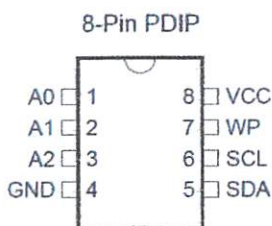


**Gambar 2.11. Pemrograman Mode Paralel dan Mode Serial AT89S51**

## 2.7. EEPROM AT24C64

Serial EEPROM tipe 24C64 adalah merupakan memori serial yang menggunakan teknologi I2C di mana dengan adanya penggunaan teknologi tersebut, jumlah I/O yang digunakan untuk meng-akses memori tersebut semakin sedikit. Hal ini sangat bermanfaat bagi sebuah sistem yang memerlukan banyak I/O. Penggunaan I/O yang semakin sedikit untuk mengakses memori, akan menyediakan lebih banyak I/O yang dapat digunakan untuk keperluan lain.

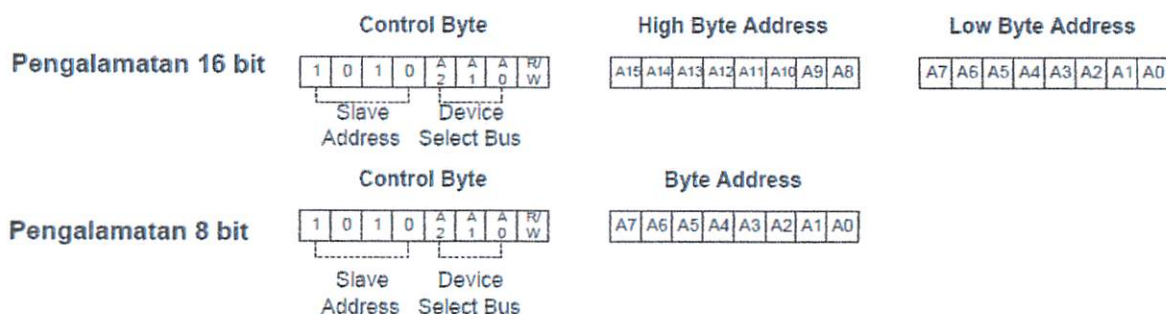
I2C adalah teknologi komunikasi serial yang ditemukan oleh Philips pada tahun 1992 dan direvisi hingga versi 2.1 yang terbaru pada tahun 2000. Teknologi ini hanya menggunakan 2 buah jalur I/O yaitu SDA dan SCL. SDA merupakan jalur data pada komunikasi I2C sedangkan SCL merupakan jalur clock di mana sinyal clock akan selalu muncul untuk setiap bit dari pengiriman data.



**Gambar 2.12. EEPROM AT24C64**  
 Sumber : [Data sheet ATMEL AT24C64](#)

EEPROM AT24C64 memiliki kapasitas penyimpanan sebesar 8Kbyte yang dikelompokkan sebagai 128 word setiap 64 bit.

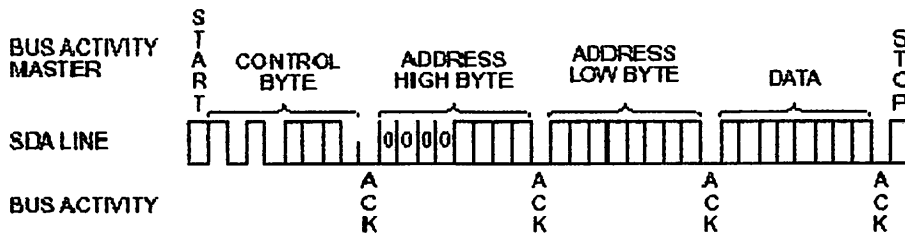
Serial EEPROM I2C berdasarkan pengalamatannya terdiri dari 2 jenis yaitu **pengalamatan 8 bit** yang digunakan untuk Serial EEPROM dengan kapasitas memori sebesar 128 byte hingga 2Kb dan **pengalamatan 16 bit** untuk Serial EEPROM 4Kb hingga 512Kb.



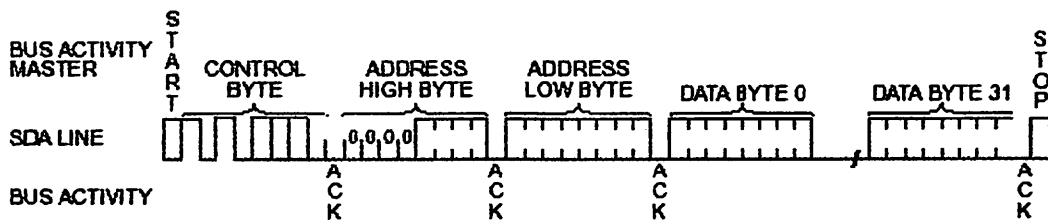
Pada pengalamatan 16 bit terdapat 3 buah paket 8 bit data yang harus dikirimkan ke Serial EEPROM yaitu Control byte, High Byte Address dan Low Byte Address, sedangkan pada pengalamatan 8 bit hanya diperlukan 2 buah paket 8 bit data yaitu Control Byte dan Byte Address saja. Control byte terdiri dari **Slave Address**, **Device Select Bus** bit-bit pengatur alamat dari Serial EEPROM dalam satu jalur bus, **R/W** bit penentu proses penulisan atau pembacaan data dari Serial EEPROM.

Pembacaan maupun penulisan data ke dalam Serial EEPROM, selalu diawali dengan pengiriman Control Byte dan Address Byte. Hanya pada penulisan data akan dilanjutkan dengan pengiriman data 8 bit, sedangkan sebaliknya pada pembacaan akan dilanjutkan dengan pengambilan data 8 bit. Bit R/W pada Control Byte akan berlogika 1 untuk pembacaan data dan berlogika 0 untuk penulisan data.

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



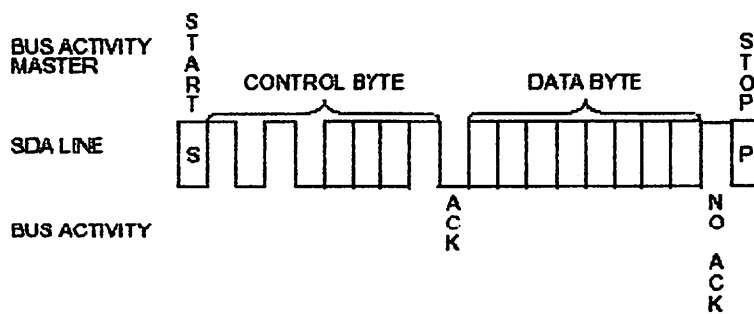
**Gambar 2.13. Penulisan Secara *Byte***



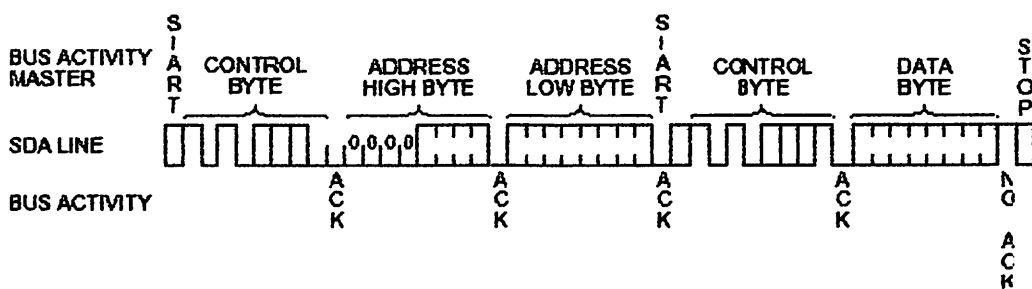
**Gambar 2.14. Penulisan Secara *Page***



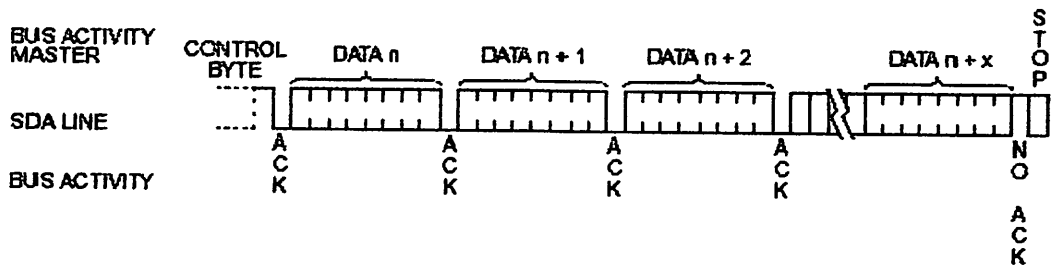
Pembacaan data dapat dilakukan secara *Current Address Read* (pembacaan alamat saat ini), *Random Read* (pembacaan secara acak), maupun secara *Sequential Read* (Pembacaan secara menyeluruh) . Pada *Current Address Read*, data yang dibaca adalah data pada alamat yang terakhir kali diakses saat itu, sedangkan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



**Gambar 2.15. Pembacaan Data Secara *Current Read***



**Gambar 2.16. Pembacaan Data Secara *Random Read***



**Gambar 2.17. Pembacaan Data Secara *Sequential Read***

## BAB III

### PERANCANGAN DAN PEMBUATAN ALAT

#### 3.1. Pendahuluan

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

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

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

Pada perangkat keras sendiri terdiri dari beberapa rangkaian antara lain, rangkaian LED, minimum sistem mikrokontroler, rangkaian LCD, rangkaian keypad, dan serial interface. Pada perancangan perangkat lunak akan meliputi diagram alir dan *software* secara umum

#### 3.2. Perancangan Sistem Kerja

Prinsip kerja dari *traffic light* terintegrasi melalui komunikasi serial adalah melakukan kontrol terhadap lampu lalu lintas (*traffic light*) yang terhubung dalam jaringan secara terpusat atau dikontrol oleh sebuah *controller*. *Controller* terdiri

dari sebuah mikrokontroler sebagai master, yang bertugas untuk melakukan pengontrolan. Dan dua buah mikrokontroler sebagai slave yaitu slave 1 dan 2 memiliki program dasar untuk *traffic light* yang nantinya akan dikontrol oleh mikrokontroler master.

Pada saat alat pertama kali dinyalakan, mikrokontroler slave 1 dan 2 akan menjalankan program *traffic light* sehingga *traffic light* akan langsung bekerja tanpa menunggu perintah dari master. Jika kita ingin merubah kondisi dari *traffic light* berupa perubahan kondisi dari merah ke kuning, dan ke hijau atau kondisi dinama dibutuhkan *traffic light* dalam keadaan kuning semua, mikrokontroler *master* akan memberikan perintah kepada mikrokontroler *slave* melalui masukan dari keypad dan akan di tampilkan pada LCD sebagai *display*

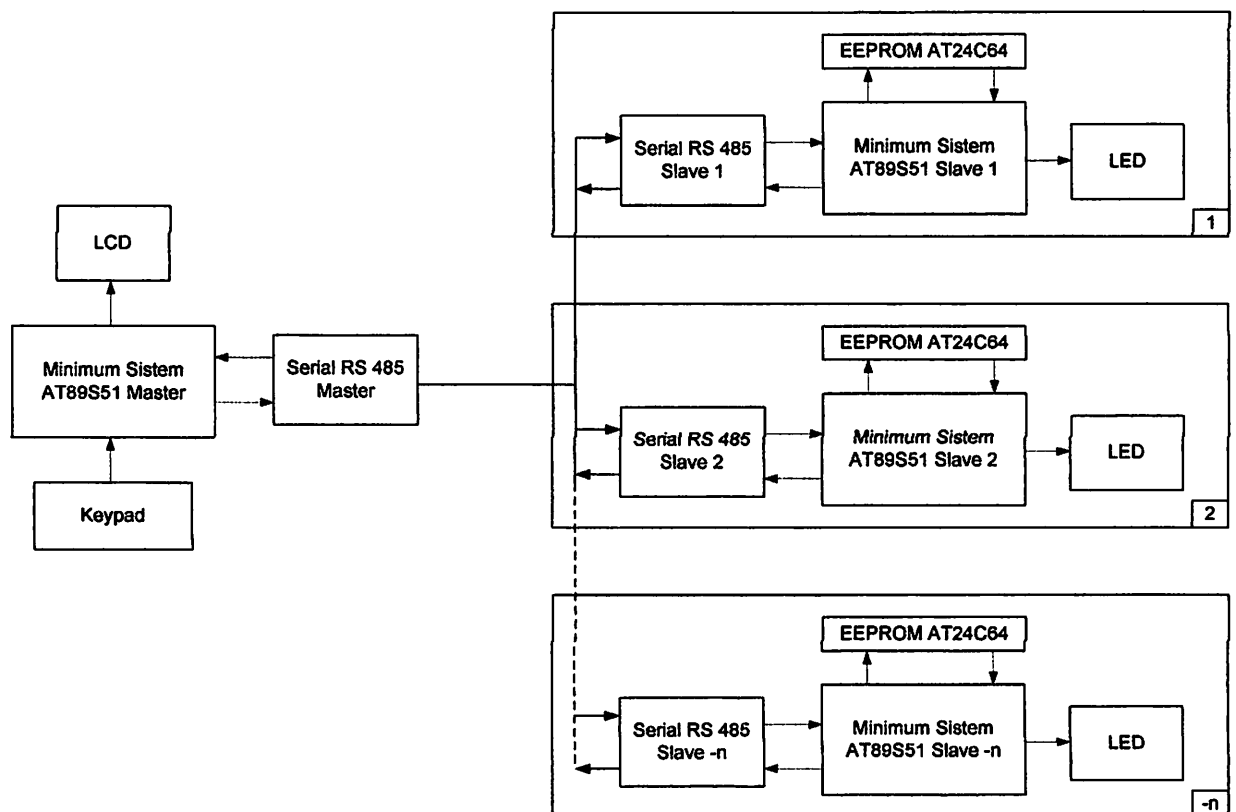
Mikrokontroler slave 1 dan 2 juga menerima perintah status dimana jika perintah ini diberikan maka mikrokontroler slave akan memberikan status kepada mikrokontroler master dan statusnya akan di tampilkan pada LCD sebagai *display*.

Komunikasi yang digunakan adalah dengan menggunakan komunikasi serial secara *half duplex*, karena RS485 tidak dapat berkomunikasi secara bersamaan (*full duplex*), serial akan selalu dalam keadaan menunggu dalam artian sebagai penerima, jika terdapat suatu kondisi dimana dia harus mengirim data maka fungsi penerimaannya di matikan sementara agar tidak terjadi kesalahan. Serial *slave* akan memberikan masukan kepada mikrokontroler slave berupa perintah dari mikrokontroler *master* melalui serial *master*. Dimana perintah tersebut diawali dengan pengiriman inisialisasi dari *slave* agar terhindar dari salah pengiriman data kepada mikrokontroler *slave* yang dituju. Begitu pula sebaliknya,

ketika slave akan memberi tahukan kondisi atau statusnya, *slave* akan terlebih dahulu mengirimkan inisial atau nomer *slave* agar tidak terjadi kesalahan pembacaan status dari *slave* oleh mikrokontroler *master*.

Pada sistem ini digunakan 2 buah minimum sistem mikrokontroler AT89S51 sebagai slave yang akan mensimulasikan 2 buah *traffic light* persimpangan 4 jalur dan 1 buah *traffic light* persimpangan 3 jalur. 1 buah minimum sistem mikrokontroler AT98S51 sebagai master. Keypad sebagai masukan, dan LCD sebagai display.

Dari prinsip kerja sistem alat ini dapat digambarkan secara blok diagram seperti pada gambar 3.1 :



**Gambar 3.1 Diagram Blok Sistem**

Fungsi dari tiap – tiap blok diagram dijelaskan sebagai berikut :

- Led berfungsi sebagai penunjuk keadaan lampu lalu lintas
- Mikrokontroler AT89S51 slave 1 dan 2 berfungsi sebagai pemroses sinyal inputan dari serial *slave* 1 dan 2 sehingga menjadi suatu outputan.
- Serial RS485 *slave* 1 dan 2 berfungsi sebagai hubungan *serial* komunikasi antara mikrokontroler *slave* dengan mikrokontroler *master*.
- Serial RS485 Master berfungsi untuk menghubungkan secara serial antara mikrokontroler *master* dengan mikrokontroler *slave* melewati serial *slave*.
- Mikrokontroler AT89S51 master berfungsi sebagai pusat kontrol untuk memberikan perintah kepada mikrokontroler slave.
- LCD berfungsi sebagai untuk menampilkan menu, perintah serta kondisi atau status.
- Keypad berfungsi untuk memberi masukan perintah.
- EEPROM AT24C64 berfungsi sebagai menyimpan data sementara status *traffic light* dan menyimpan program *default traffic light*.

### 3.3. Perancangan Perangkat Keras

#### 3.3.1. Rangkaian LED

Rangkaian LED yang digunakan adalah *Common Anode (CA)*, untuk menghidupkan LED data yang diberikan atau dituliskan berlogika '0' (aktif *low*). Hal ini dikarenakan jika diberikan logika '0' pada salah satu port, *internal pullup*

pada port tersebut akan di *pulled-low* (secara *internal*), sehingga LED yang terhubung secara *common anode* akan menyala lebih terang. Sedangkan jika digunakan rangkaian *common katode* untuk menyalakan LED dibutuhkan logika '1' (aktif high) yang akan menyebabkan port berimpedansi yang dikarenakan adanya *internal pullup*, disamping itu juga arus keluaran dari port terlalu kecil karena adanya impedansi dalam. Dengan tegangan 5 volt diperlukan resistor sebesar 330 ohm sebagai pembatas arus, dimana akan diperoler arus sebesar 15mA yang cukup untuk menyalakan LED yang biasanya membutuhkan arus sekitar 10mA.

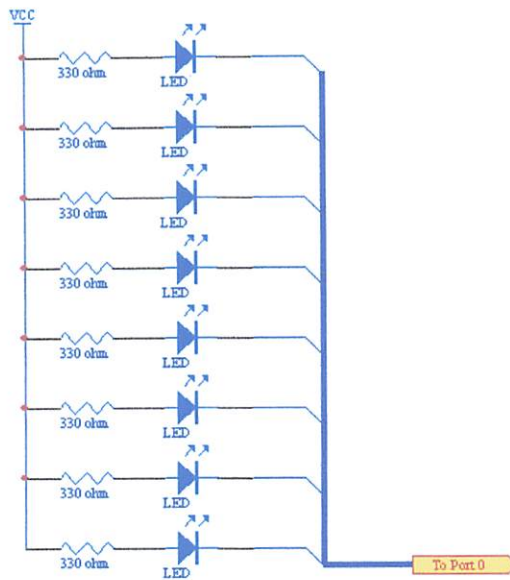
$$V = I \times R$$

Dimana  $V = 5$  volt,

$$R = 330 \text{ ohm}$$

Maka  $V = I \times R$

$$\begin{aligned} I &= \frac{V}{R} \\ &= \frac{5}{330} \\ &= 0,015\text{A atau } 15\text{mA} \end{aligned}$$

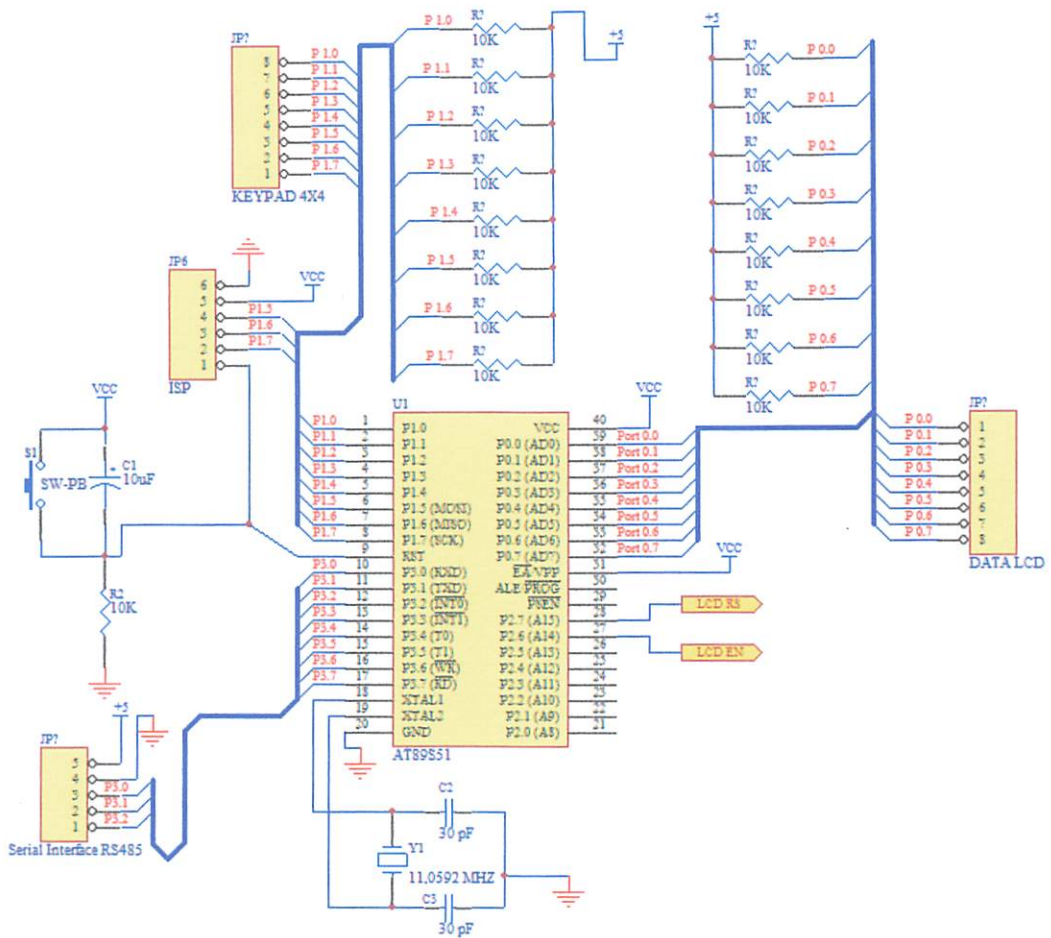


**Gambar 3.2. Rangkaian LED**

### 3.3.2. Minimum Sistem AT98S51 (*master*)

Mikrokontroller yang digunakan ialah mikrokontroller keluarga ATMEL AT89S51. Data yang diperoleh dari masukan keypad akan diolah oleh MCU AT89S51 ini sebagai pemroses pengendalian maupun pemroses data jawaban atas status sistem. Untuk dapat berkomunikasi dengan mikrokontroller *slave* digunakan port serial mikrokontroller AT89S51.





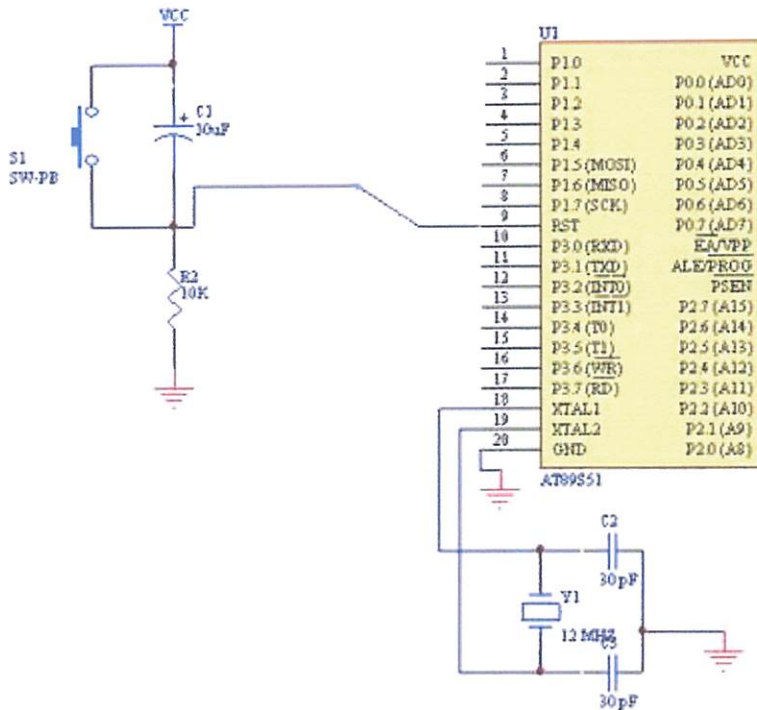
**Gambar 3.3. Minimum Sistem AT89S51 (master) Dan Alokasi Pin**

Alokasi penggunaan pin mikrokontroller pada minimum sistem sebagai master:

- Port 0.0-0.7 digunakan sebagai port output (keluaran) yang akan mengeluarkan data ke LCD YJ162A.
- Port 2.7 digunakan sebagai port output operasi RS (Register Select) pada LCD.

- Port 2.6 digunakan sebagai port output untuk operasi *enable* LCD (aktif atau tidaknya LCD).
- $\overline{EA}$  (*External Access*) Port  $\overline{EA}$  dihubungkan ke Vcc .Jika  $\overline{EA}$  diberikan logika tinggi, maka mikrokontroller akan mengakses program dari ROM internal (EPROM/flash memory). Jika  $\overline{EA}$  diberikan logika rendah, maka mikrokontroller akan mengakses program dari memori eksternal.
- Port XTAL1 dan XTAL2 sebagai masukan dari rangkaian osilator kristal. Rangkaian osilator kristal terdiri atas osilator 11,0592 MHz, kapasitor C<sub>1</sub> dan C<sub>2</sub> yang masing-masing bernilai 33 pF yang akan membangkitkan pulsa *clock* yang digunakan sebagai penggerak bagi sejumlah operasi internal CPU.
- Port 3.2 (INT0) digunakan sebagai port kontrol pengiriman atau penerima data serial RS485.
- Port 3.1. (TXD) digunakan sebagai pin TX atau mengirimkan data pada komunikasi data serial.
- Port 3.0. (RXD) digunakan sebagai pin RX atau untuk menerima data pada komunikasi data serial.
- Port RST (reset) digunakan Input reset master untuk AT89S51.
- Port 1.0-1.7 digunakan sebagai port input yang mendapat masukan dari keypad matrix 4x4. Disamping itu juga digunakan sebagai port pemrograman (ISP).

Perancangan rangkaian reset pada minimum sistem mikrokontroler AT98S51 ialah dengan memberikan logika low pada pin reset mikrokontroler AT89S51. Berikut ialah gambar rancangan rangkaian reset pada AT98S51 :



**Gambar 3.4. Rangkaian Reset dan Osilator pada Minimum Sistem AT98S51**

Dalam rangkaian digunakan kristal dengan harga 11,0592 MHz, maka program akan dijalankan pada setiap langkahnya selama 1,085 µs. Siklus tersebut diambil berdasarkan ketentuan mikrokontroller AT89S51 yaitu 12 clock = 1 siklus mesin, sedangkan frekwensi yang digunakan 12 MHz, maka waktu yang dipakai dalam setiap 1 siklus mesin adalah 1µs. Dengan demikian perhitungannya dapat dilihat sebagai berikut:

$$f = 11,0592 \text{ MHz}$$

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

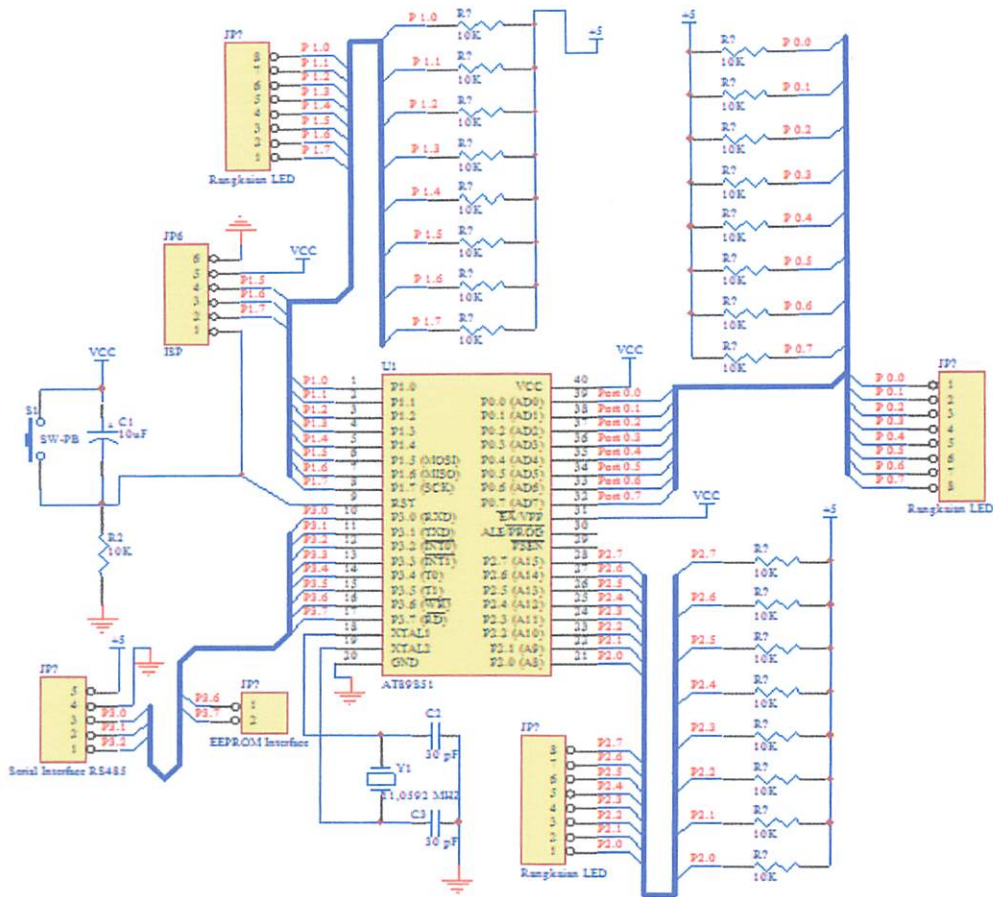
$$T = \frac{1}{11,0592 \times 10^6}$$

Karena 1 siklus mesin = 12 T maka,

$$1 \text{ siklus mesin} = 12 \times \frac{1}{11.0592 \times 10^6} = 1,085 \mu\text{s}.$$

### 3.3.3. Minimum Sistem AT98S51 (*slave*)

Pada minimum sistem AT98S51 sebagai *slave* tidak terdapat LCD dan keypad sehingga port yang pada minimum sistem master digunakan sebagai data LCD dan masukan keypad 4x4, pada minimum system *slave* digunakan sebagai port untuk rangkaian LED sebagai *traffic light*. Berikut gambar rangkaian dari minimum sistem AT89S51 sebagai *slave* :



Gambar 3.5. Minimum Sistem AT89S51 (slave) Dan Alokasi Pin

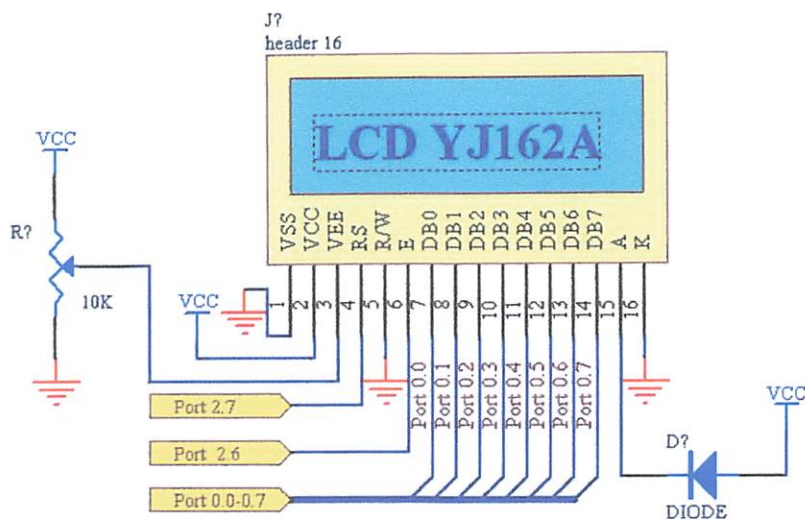
Alokasi penggunaan pin mikrokontroler sebagai *slave* :

- Port 0.0-0.7 digunakan sebagai port output (keluaran) yang akan mengeluarkan data ke rangkaian LED.
- Port 2.0-2.7 digunakan sebagai port output (keluaran) yang akan mengeluarkan data ke rangkaian LED.
- $\overline{EA}$  (*External Access*) Port  $\overline{EA}$  dihubungkan ke Vcc .Jika  $\overline{EA}$  diberikan logika tinggi, maka mikrokontroler akan mengakses program dari ROM internal (EPROM/flash memory). Jika  $\overline{EA}$  diberikan logika rendah, maka mikrokontroler akan mengakses program dari memori eksternal.

- Port XTAL1 dan XTAL2 sebagai masukan dari rangkaian osilator kristal. Rangkaian osilator kristal terdiri atas osilator 11,0592 MHz, kapasitor  $C_1$  dan  $C_2$  yang masing-masing bernilai 33 pF yang akan membangkitkan pulsa *clock* yang digunakan sebagai penggerak bagi sejumlah operasi internal CPU.
- Port 3.2 (INT0) digunakan sebagai port kontrol pengiriman atau penerima data serial RS485.
- Port 3.1. (TXD) digunakan sebagai pin TX atau mengirimkan data pada komunikasi data serial.
- Port 3.0. (RXD) digunakan sebagai pin RX atau untuk menerima data pada komunikasi data serial.
- Port RST (reset) digunakan Input reset master untuk AT89S51.
- Port 1.0-1.7 digunakan sebagai port output (keluaran) yang akan mengeluarkan data ke rangkaian LED. Disamping itu juga digunakan sebagai port pemrograman (ISP).

#### **3.3.4. Perancangan Rangkaian LCD**

LCD yang dipergunakan adalah sebuah LCD dot matrix 16X2 baris dengan konsumsi daya rendah. Modul LCD Character dapat dengan mudah dihubungkan dengan mikrokontroler seperti AT89S51. LCD yang akan digunakan ini mempunyai lebar display 2 baris 16 kolom atau biasa disebut sebagai LCD Character 2x16, dengan 16 pin konektor.



**Gambar 3.6. Rangkaian LCD**

Display karakter pada LCD diatur oleh pin EN, RS dan RW Jalur EN dinamakan Enable dihubungkan dengan mikrokontroler pada port 2.6. Jalur ini digunakan untuk memberitahu LCD bahwa anda sedang mengirimkan sebuah data. Untuk mengirimkan data ke LCD, maka melalui program EN harus dibuat logika low “0” dan set pada dua jalur kontrol yang lain RS dihubungkan pada port 2.7 dan RW. Ketika dua jalur yang lain telah siap, set EN dengan logika “1” dan tunggu untuk sejumlah waktu tertentu ( sesuai dengan datasheet dari LCD tersebut ) dan berikutnya set EN ke logika low “0” lagi. Jalur RS adalah jalur Register Select. Ketika RS berlogika low “0”, data akan dianggap sebagai sebuah perintah atau instruksi khusus ( seperti clear screen, posisi kursor dll ). Ketika RS berlogika high “1”, data yang dikirim adalah data text yang akan ditampilkan pada display LCD. Sebagai contoh, untuk menampilkan huruf “T” pada layar LCD



maka RS harus diset logika high "1". Jalur RW adalah jalur kontrol Read/ Write. Ketika RW berlogika low (0), maka informasi pada bus data akan dituliskan pada layar LCD. Ketika RW berlogika high "1", maka program akan melakukan pembacaan memori dari LCD. Sedangkan pada aplikasi umum pin RW selalu diberi logika low "0".

Pada akhirnya, bus data terdiri dari 4 atau 8 jalur ( bergantung pada mode operasi yang dipilih oleh user ). Pada kasus bus data 8 bit, jalur diacukan sebagai DB0 s/d DB7 Beberapa perintah dasar yang harus dipahami adalah inisialisasi LCD Character.

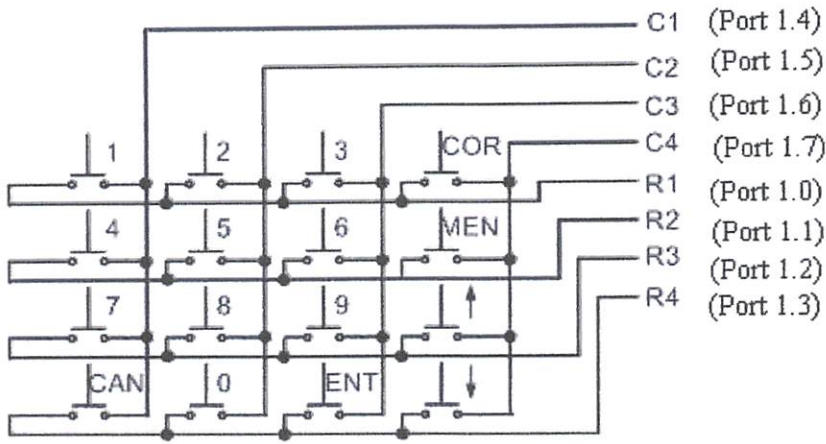
### **3.3.5. Perancangan Rangkaian Keypad**

Rangkaian *Keypad* dalam perencanaan ini menggunakan *Keypad matrik* 4x4 (4 baris dan 4 kolom). Berisikan angka 0 sampai 9. dan 6 tombol tyang tersisa dimanfaatkan untuk CAN, ENT, MEN, UP, DOWN, dan COR.

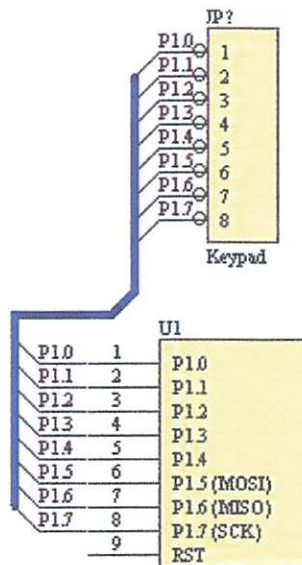
Keypad sesungguhnya terdiri dari sejumlah saklar, yang terhubung sebagai baris dan kolom dengan susuan seperti yang ditunjukkan pada gambar 3.7. Teknik pembacaan dari *Keypad* ini, yaitu model *scanning* pada jalur baris dan pembacaan jalur kolom, agar mikrokontroller dapat melakukan scan keypad, maka port mengeluarkan salah satu bit dari 4 bit yang terhubung pada kolom dengan logika low "0" dan selanjutnya membaca 4 bit pada baris untuk menguji jika ada tombol yang ditekan pada kolom tersebut. Sebagai konsekuensi, selama tidak ada tombol



yang ditekan, maka mikrokontroller akan melihat sebagai logika high "1" pada setiap pin yang terhubung ke baris.



Gambar 3.7. Susuna Baris dan Kolom Keypad 4x4



Gambar 3.8. Port Keypad pada Minimum Sistem AT98S51

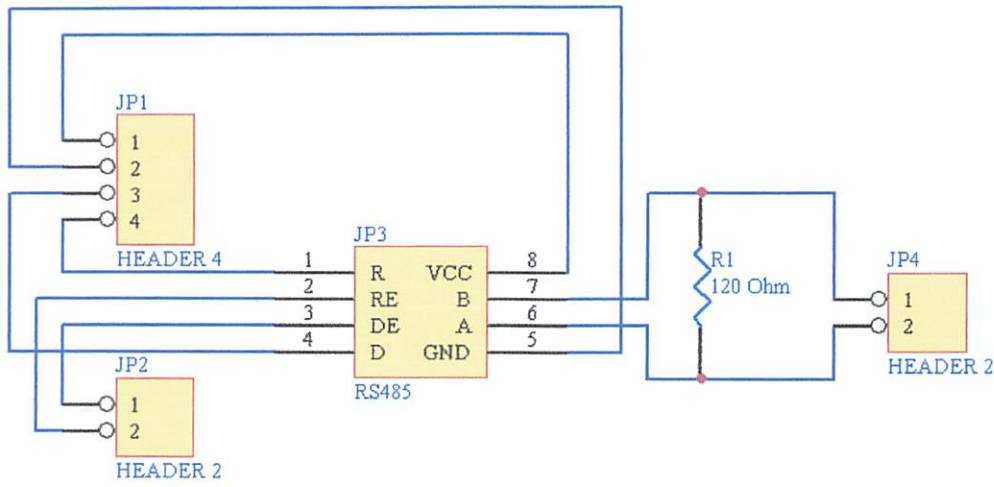
### 3.3.6. Perancangan Rangkaian RS 485

Pada sistem kendali ini terdapat lebih dari satu alat (*device*) maka tidak cukup 1 karakter pada setiap penerimaan/pengiriman data. Alat yang dapat menangani lebih dari 2 node dapat digunakan RS 485. Setiap pengiriman data, node mengirim beberapa karakter yang berisi alamat unik node, tujuan, data dan sebagainya tergantung keperluan. MAX485 dirancang untuk komunikasi data dua arah (*half duplex*) pada jalur transmisi bus *multipoint*. MAX485 dapat juga digunakan sebagai *line repeaters*, dengan panjang kabel lebih dari 4000 kaki atau setara dengan 1,3 Km. Untuk memperkecil pemantulan, pada kedua sisi *line* sebaiknya diberi resistansi sebesar impedansi karakteristiknya (umumnya 120 ohm), dan panjang *line* utama sebaiknya sependek mungkin.

Dalam desain yang dibuat dalam skripsi ini pin *driver enable* (DE) dan *receiver enable* (RE) digabung menjadi satu dihubungkan pada port INT0 (P 3.2). Dengan demikian ketika *driver* aktif, saat itu juga *receiver* akan mati, demikian pula sebaliknya. Sistem ini akan mengakibatkan komputer tidak akan menerima pantulan (*echo*) dari data yang dikirimkannya sendiri. Sedangkan pin R0 dihubungkan ke port RXD penerima data, jika nilai arus pada inputan A lebih besar dari B, maka R0 akan berlogika 1 atau high, dan sebaliknya apabila nilai A lebih kecil dari B maka R0 akan berlogika 0 atau low. VCC pada kaki 8 akan diberikan tegangan 5V dan kaki no 5 akan dihubungkan dengan Ground. Sedangkan kaki no 6,7 (A,B) di hubungkan dengan pin A dan B pada slave.

Pada *Internal Termination*, RS 485 ini juga disediakan slot untuk diisi resistor *termination* (gambar 3.9.). Dengan demikian resistor *termination* tidak

perlu lagi ditambahkan pada kabel jaringan. Nilai resistor *termination* mengikuti nilai standar EIA/TIA, yaitu 120 ohm (dengan asumsi impedansi karakteristik kabel dari 100 – 120 ohm). Jika resistor *termination* tidak diperlukan (*node* bukan merupakan *node* yang terujung dalam bus) maka slot ini dapat dikosongkan.



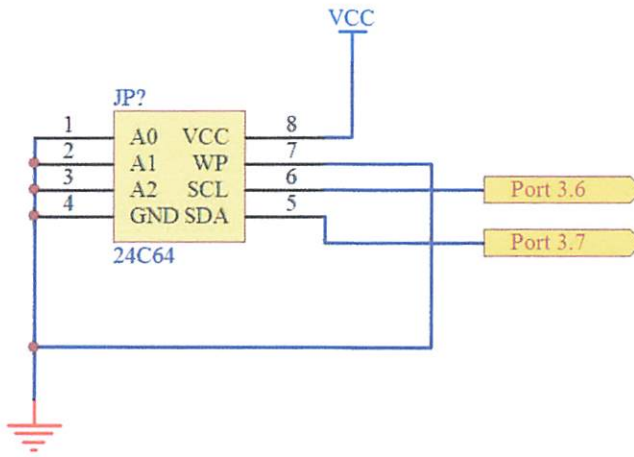
**Gambar 3.9. Rangkaian RS 485**

### 3.3.7. Perancangan Rangkaian EEPROM AT24C64

Pada sistem ini digunakan EEPROM AT24C64 sebagai tempat untuk menyimpan data sementara status dari *traffic light* dan tempat untuk menyimpan program *default traffic light*. AT24C64 merupakan memori serial yang berkapasitas sebesar 8K bit dengan menggunakan teknologi I2C, teknologi ini hanya menggunakan 2 buah jalur I/O yaitu SDA dan SCL. Dimana SDA merupakan jalur data pada komunikasi I2C sedangkan SCL merupakan jalur clock di mana sinyal clock akan selalu muncul untuk setiap bit dari

pengiriman data.

Rangkaian EEPROM AT24C64, jalur serial data dihubungkan ke port 3.7 pada mikrokontroller dan jalur serial clock dihubungkan pada port 3.6 mikrokontroller. Pin WP, dan A0 sampai A2 dihubungkan ke ground. Hal ini dikarenakan EEPROM akan digunakan dengan mode pembacaan data terakhir diakses (*Current Address Read*).

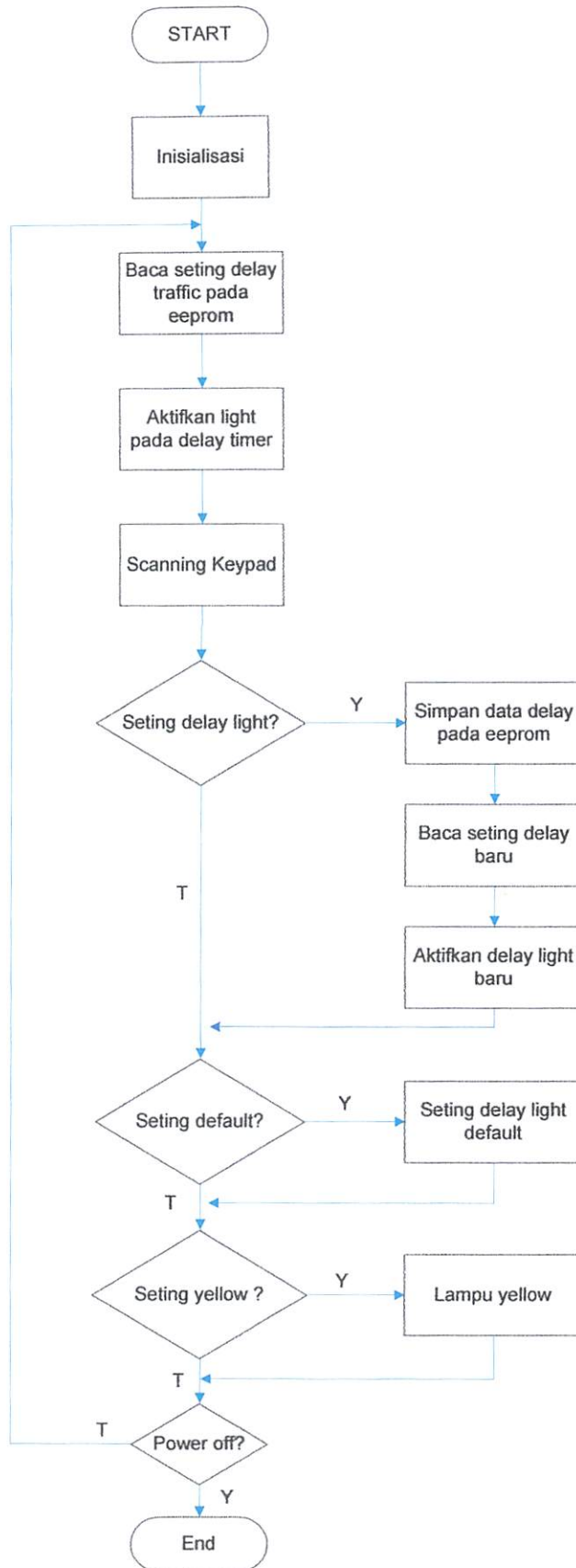


**Gambar 3.10. Rangkaian EEPROM AT24C64**

### 3.4. Perancangan Perangkat Lunak

Sebagai pendukung perangkat keras, perangkat lunak berfungsi sebagai penunjang agar system dapat berjalan dengan benar dan sesuai dengan keinginan. Dalam perancangan system perangkat lunak, setiap urutan instruksi harus didefinisikan secara benar dan jelas. Hal ini dikarenakan mikrokontroller akan membaca instuksi berdasarkan urutan, susunan logika perancangan yang salah tidak dapat diketahui oleh Mikrokontroller. Selama instruksi yang diterima sesuai dengan aturannya, Mikrokontroller tetap mengerjakan instruksi tersebut. Kesalahan seperti ini baru diketahui ketika kerja sistem bekerja tidak sesuai dengan perancangan.

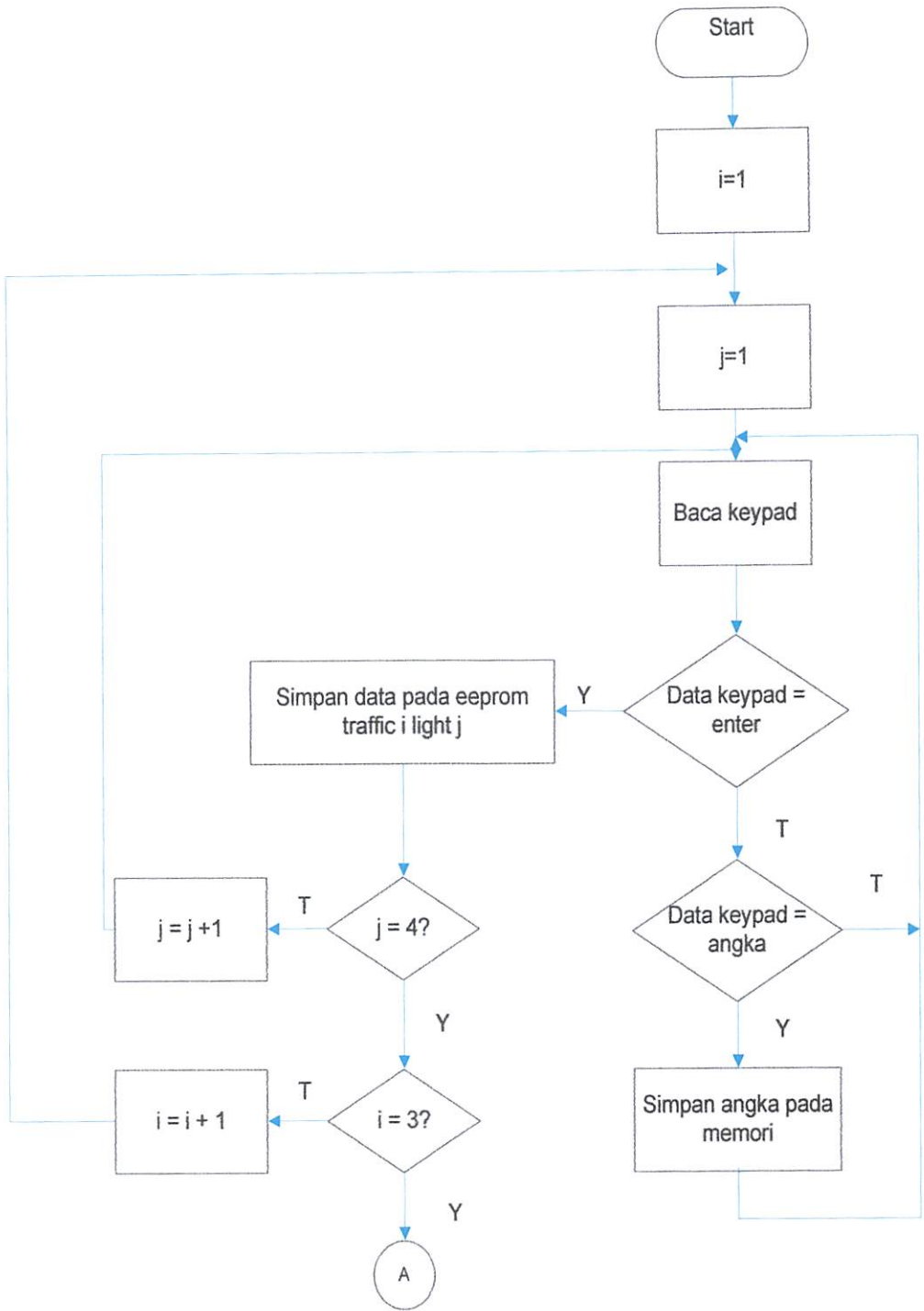
Perancangan perangkat lunak (*software*) yang digunakan dalam perencanaan dan pembuatan alat akan dipaparkan dalam *flowchart* sistem secara keseluruhan. Pembuatan *software* hanya dilakukan pada mikrokontroler menggunakan bahasa pemrograman *assembler*. Diagram alir atau *Flowchart* program secara keseluruhan adalah sebagai berikut:



**Gambar 3.11. Flow chart Proses Jalannya Sistem**

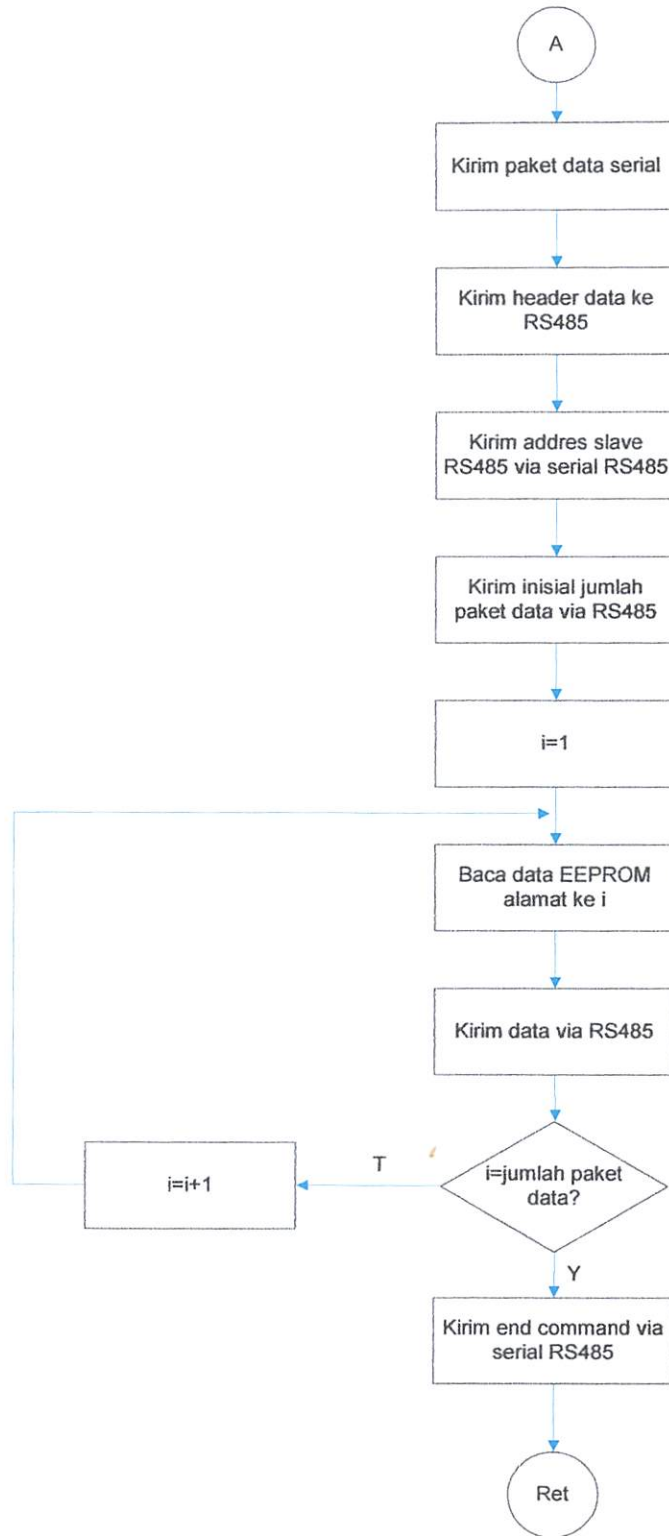
Program yang digunakan adalah bahasa *assembly*, merupakan suatu program yang terdiri dari label-label, mnemonik dan lain sebagainya. Masing-masing pernyataan berhubungan dengan suatu instruksi mesin. Gambar 3.10 dapat dijelaskan sebagai berikut :

1. Start.
2. Inisialisasi LCD, serial.
3. Membaca setingan delay *traffic* pada EEPROM.
4. Mengaktifkan light pada delay timer.
5. Melakukan pengecekan penekanan keypad.
6. Apakah Setting delay light.
7. Jika “ya” menyimpan data delay pada EEPROM.
8. Membaca setting delay baru.
9. Mengaktifkan light dengan delay timer baru.
10. Jika “tidak” Apakah setting dengan *default*.
11. Jika “ya” setting delay light dengan delay *default*.
12. Jika “tidak” Apakah setting *yellow*.
13. Jika “ya” setting *light yellow*.
14. Jika “tidak” apakah power off.
15. Jika “tidak” melakukan pengulangan membaca setting delay pada EEPROM.
16. Jika “ya” program selesai.

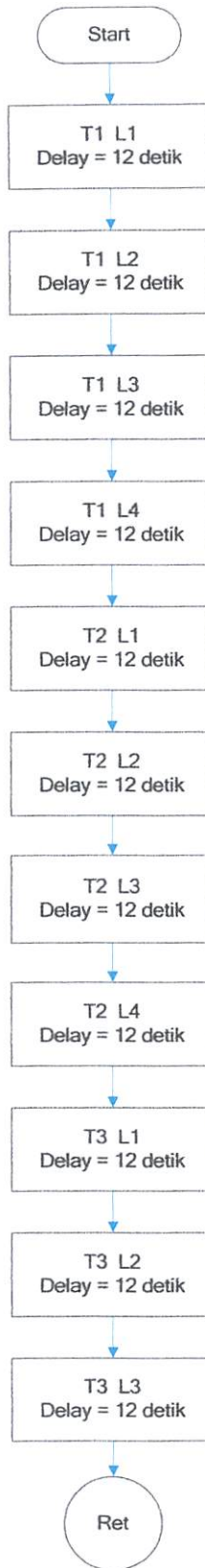


**Gambar 3.12. Flow chart Proses Pembacaan Keypad**

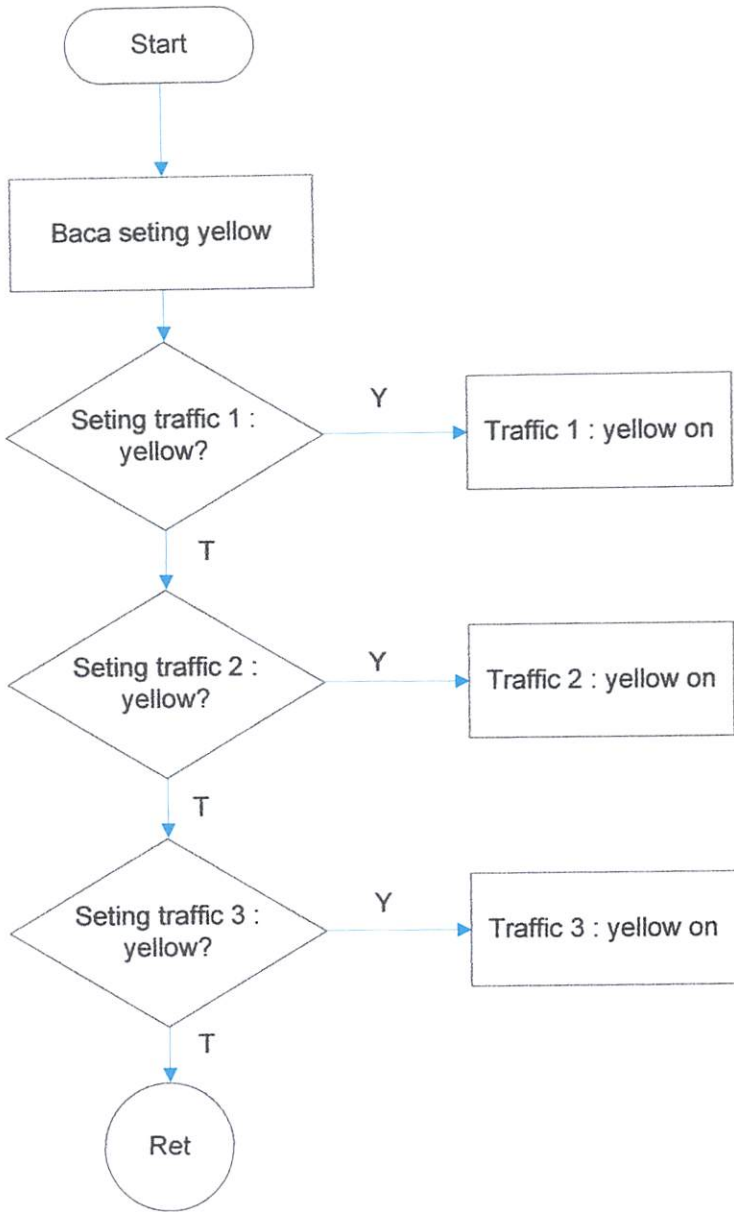




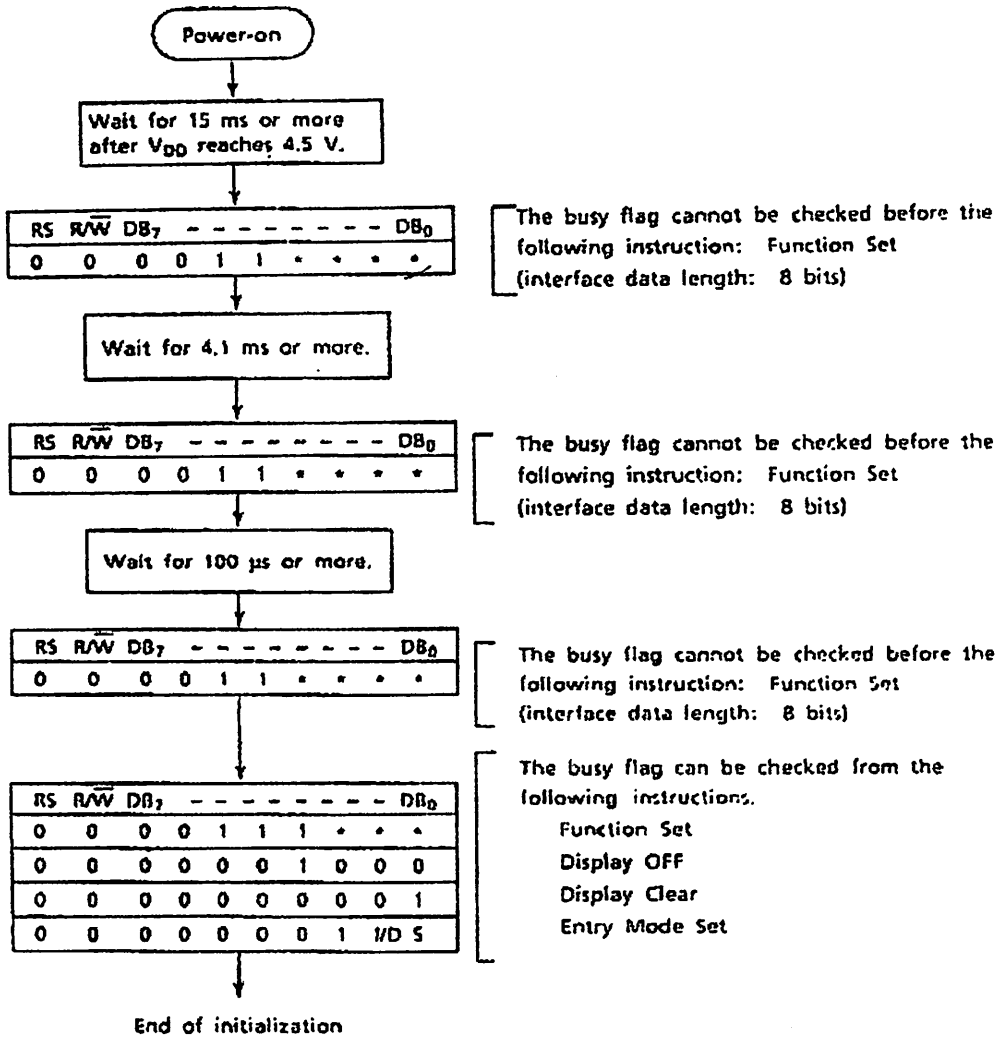
**Gambar 3.13. Flow chart Komunikasi Serial RS485**



**Gambar 3.14. Flow chart Setting *Delay Default***



**Gambar 3.15. Flow chart Setting *Yellow***



Gambar 3.16. Flow chart Proses Inisialisasi LCD

## **BAB IV**

### **PENGUJIAN DAN HASIL ANALISA**

Pengujian dilakukan untuk mengetahui sejauh mana peralatan dapat bekerja sesuai dengan perencanaan. Langkah pengujian dilakukan melalui 2 tahap, yakni pengujian pada setiap blok dan pengujian pada sistem keseluruhan. Tahap pertama dimaksudkan untuk mengetahui sejauh mana blok-blok rangkaian dapat berjalan, sedangkan tahap kedua dilakukan setelah diperoleh kepastian bahwa tiap blok rangkaian telah berjalan sesuai rencana.

#### **4.1. Pengujian Rangkaian Minimum Sistem Mikrokontroller AT89S51**

##### **4.1.1. Tujuan**

Tujuan dari pengujian minimum sistem mikrokonroller AT 89S51 adalah untuk mengetahui apakah minimum sistem dapat bekerja dengan baik atau tidak.

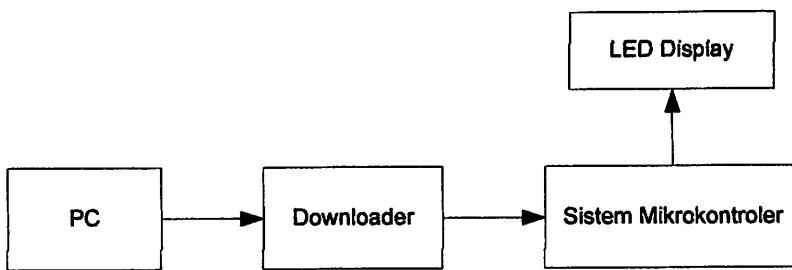
##### **4.1.2. Alat dan Bahan**

1. *Personal Computer* (PC) yang dilengkapi LPT1
2. *Downloader* mikrokontroller ATMEL AT89S51
3. *ISP programmer* dengan *connector* LPT1
4. Minimum sistem Mikrokontroler
5. Catu daya 5 volt

##### **4.1.3. Langkah Pengujian**

Tahapan Pengujian mikrokontroller ATMEL AT89S51 sebagai berikut:

1. Menghubungkan alat dan bahan seperti gambar 4–1.
2. Memberikan catu daya 5 volt.
3. Membuat program yang digunakan untuk menguji mikrokontroler.  
Program yang digunakan dalam pengujian mikrokontroler ini merupakan program yang sederhana yang meletakkan  $0F_H$  dan  $F0_H$  secara bergantian pada *Port 0* ATMEL AT89S51.
4. Mengamati keluaran pada LED Display.



**Gambar 4.1. Diagram Blok Pengujian Mikrokontroler**

#### 4.1.4. Hasil dan Analisa Pengujian

**Tabel 4.1. Hasil Pengujian Sistem Mikrokontroler**

Kondisi	Keluaran pada 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

Keterangan :

- Kondisi bit *low* ( 0 ) = LED menyala
- Kondisi bit *high* ( 1 ) = LED mati

Dari hasil pengujian pada tabel 4-1 dapat dilihat bahwa *port* 0 mengeluarkan logika  $F0_H$  dan  $OF_H$  secara bergantian sesuai dengan isi program. Sehingga diketahui bahwa minimum system bekerja sesuai perancangan.

## **4.2. Pengujian Rangkaian Komunikasi Serial RS485**

### **4.2.1. Tujuan**

Tujuan dari pengujian rangkaian komunikasi serial RS485 adalah untuk mengetahui apakah data yang dikirimkan oleh master dan diterima oleh slave sudah benar.

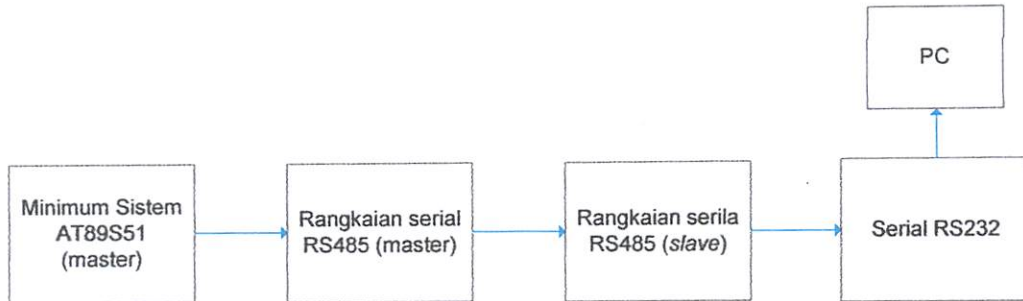
### **4.2.2. Alat dan Bahan**

1. Minimum sistem AT89S51
2. Rangkaian Serial RS485
3. Serial RS232
4. Kabel Serial RS485 dan RS232
5. PC
6. Catu Daya 5Volt

### **4.2.3. Langkah Pengujian**

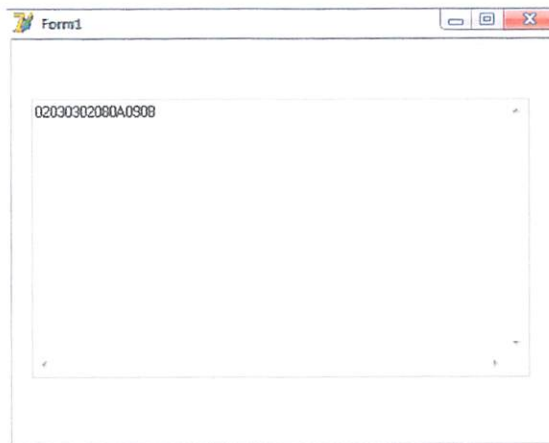
1. Menghubungkan alat dan bahan seperti gambar 4.2

2. Menbuat program mikrokontroler AT89S51 yang berfungsi untuk mengirimkan data (02030302080A090B).
3. Mengisikan program dan melakukan penekanan pada keypad
4. Mengamati hasil pengujian



**Gambar 4.2. Blok Diagram Pengujian Serial RS485**

#### 4.2.4. Hasil dan Analisa Pengujian



**Gambar 4.3. Hasil Pengujian Serial pada Mikrokontroler *slave***

Dari hasil pengamatan di ketahui bahwa data yang dikirim dari minimum sistem *master* adalah sama dengan data yang dikeluarkan oleh PC (Personal Computer). Dengan demikian serial RS485 telah dapat mengirim data atau



menerima data dengan baik pada kecepatan data 9600 bps, dan pada percobaan diatas tidak terdapat kesalahan data atau error yang terjadi sebesar 0%.

### 4.3. Pengujian Keypad

#### 4.3.1. Tujuan

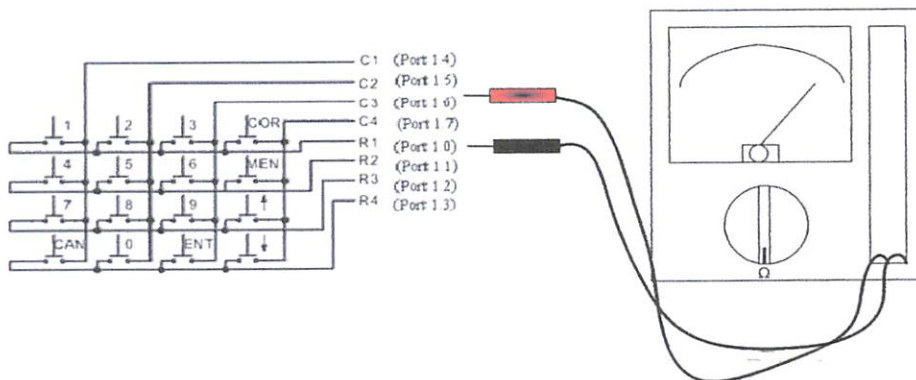
Tujuan dari pengujian keypad adalah untuk mengetahui konfigurasi logika dari keypad pada saat ada penekanan tombol. Dan apakah rangkaian keypad sudah benar.

#### 4.3.2. Alat dan Bahan

1. Keypad Matrix 4x4
2. Multitester SANWA TX360TRF

#### 4.3.3. Langkah Pengujian

1. Menghubungkan keypad dengan multitester pada gambar 4.4.
3. Mengamati hasil pengujian



Gambar 4.4. Blok Diagram Pengujian Keypad

#### 4.3.4. Hasil dan Analisa Pengujian

Table 4.2. Hasil Pengujian *Keypad*

BUTTON	C1	C2	C3	C4	B1	B2	B3	B4
1	1	0	0	0	1	0	0	0
2	0	1	0	0	1	0	0	0
3	0	0	1	0	1	0	0	0
MEN	0	0	0	1	1	0	0	0
4	1	0	0	0	0	1	0	0
5	0	1	0	0	0	1	0	0
6	0	0	1	0	0	1	0	0
COR	0	0	0	1	0	1	0	0
7	1	0	0	0	0	0	1	0
8	0	1	0	0	0	0	1	0
9	0	0	1	0	0	0	1	0
↑	0	0	0	1	0	0	1	0
CAN	1	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	1
ENT	0	0	1	0	0	0	0	1
↓	0	0	0	1	0	0	0	1

Dari hasil pengujian didapat bahwa rangkaian keypad seperti pada tabel 4.3, dengan demikian maka dapat dikatakan bahwa rangkaian keypad dapat bekerja dengan baik.

## 4.4. Pengujian Rangkaian LCD

### 4.4.1. Tujuan

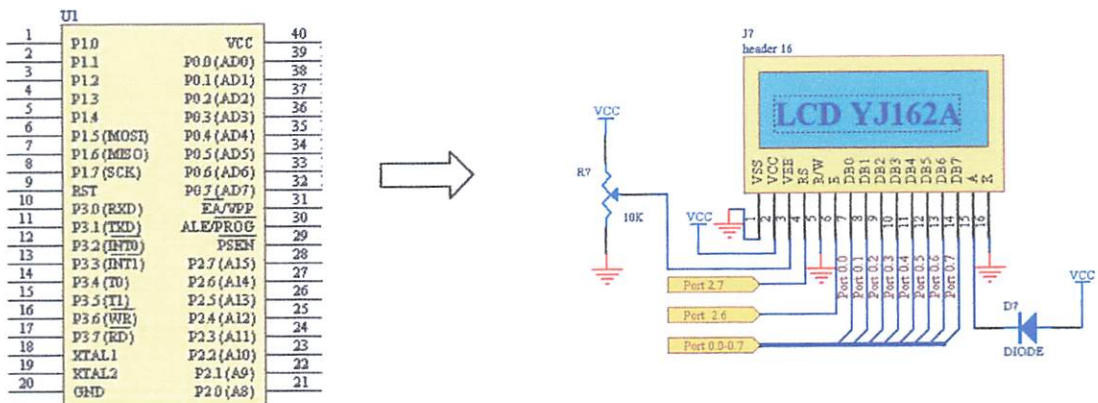
Tujuan dari pengujian rangkaian LCD ini adalah untuk mengetahui apakah rangkaian LCD yang dirancang dapat bekerja sesuai dengan yang direncanakan.

### 4.4.2. Alat dan Bahan

1. Rangkaian LCD
2. Rangkaian Mikrokontroler AT89S51
3. Catu daya 5 Volt

### 4.4.3. Langkah Pengujian

1. Mempersiapkan rangkaian pengujian seperti Rangkaian pengujian LCD pada gambar 4.5.
2. Mempersiapkan *software* mikrokontroler yang berfungsi untuk inialisasi LCD dan menampilkan suatu kata-kata pada baris pertama bertuliskan “SISTEM CONTROL”, baris kedua bertuliskan “Traffic Light”.
3. Mengamati hasil pengujian.



Gambar 4.5. Blok Diagram Pengujian LCD

#### 4.4.4. Hasil dan Analisa Pengujian

Setelah melakukan pengujian LCD diatas didapatkan tulisan “SISTEM CONTROL’ pada baris pertama dan “Traffic Light” pada baris kedua . Dengan demikian setelah mengetahui hasil pengujian rangkaian LCD dapat disimpulkan bahwa rangkaian LCD tersebut telah berfungsi sesuai dengan yang direncanakan.



**Gambar 4.6. Hasil Pengujian LCD**

#### 4.5. Pengujian Timer

##### 4.5.1. Tujuan

Tujuan dari pengujian timer adalah:

1. Mengetahui delay timer pada *traffic light*
2. Membandingkan delay hasil pengukuran dengan data masukan.

##### 4.5.2. Alat dan Bahan

1. Minimum sistem AT89S51.
2. Rangkaian LED
3. Stopwatch.
4. Catu daya 5volt.

### 4.5.3. Langkah Pengujian

1. Mempersiapkan rangkaian pegujian
2. Mempersiapkan program untuk menguji timer.
3. Mengisikan program.
4. Mengamati hasilnya dan mengukur dengan stopwatch.

### 4.5.4. Hasil Pengujian

**Table 4.3. Hasil Pengujian Timer**

Pengujian ke	Masukan ( detik)	Pengukuran Stopwatch (detik)
1	11	11,12
2	16	16,15
3	18	18,21
4	20	20,24
5	25	25,25
6	27	27,3
7	33	33,42
8	38	38,33
9	45	45,47
10	50	50,49

#### 4.5.5. Analisa Hasil Pengujian

Timer berdasarkan perhitungan teori sebagai berikut :

$$f = 11,0592 \text{ MHz}$$

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

$$T = \frac{1}{11,0592 \times 10^6}$$

Karena 1 siklus mesin = 12 T maka,

$$1 \text{ siklus mesin} = 12 \times \frac{1}{11.0592 \times 10^6} = 1,085 \mu\text{S.}$$

$$\text{Sehingga timer} = 1,085 \mu\text{S} \times 200 \times 4609$$

$$= 1,000153 \text{ S} \approx 1\text{S}$$

Dari hasil perhitungan timer yang dimasukkan melalui keypad 1 kali (1x) data yang diberikan melalui keypad. Namun pada hasil pengukuran menunjukkan adanya selisih waktu timer antara teori dan pengukuran. Toleransi error pada pengukuran diatas dirumuskan sebagai berikut :

$$\text{Error} = \frac{\text{Hasil\_Ukur} - \text{Hasil\_Teori}}{\text{Hasil\_teori}} \times 100 \%$$

Dengan demikian hasil dari pengukuran dan perhitungan ditunjukkan dalam table 4.5.

**Table 4.4. Perbandingan Hasil Pengujian dan Analisa Pengujian Timer**

Pengujian Ke	Data Masukan (detik)	Hasil Teori (detik)	Hasil Pengukuran Stopwatch (detik)	Error (%)
1	11	11	11,12	1.09
2	16	16	16,15	0.99
3	18	18	18,21	1
4	20	20	20,24	1,2
5	25	25	25,25	1
6	27	27	27,3	1,1
7	33	33	33,42	1.27
8	38	38	38,33	0,86
9	45	45	45,47	1.04
10	50	50	50,49	0,98

$$\begin{aligned} \text{Rata-rata error} &= \frac{\sum \text{nilai\_error}}{\sum \text{pengujian}} \\ &= \frac{10,53}{10} \\ &= \mathbf{1,053\%} \end{aligned}$$

Dari hasil analisa diatas dapat diketahui bahwa error yang terjadi bekisar antara 0,86% sampai dengan 1,27%. Dengan rata error rany terjadi sebesar 1,053%. Error yang terjadi dikarenakan adanya ketidak tepatan dalam penggunaan *stopwatch*.

## **4.6. Pengujian Keseluruhan Sistem**

### **4.6.1. Tujuan**

Tujuan dari pengujian keseluruhan sistem adalah untuk mengetahui apakah setiap blok dapat berjalan dengan benar dan keseluruhan sistem bekerja sesuai dengan perancangan.

### **4.6.2. Alat dan Bahan**

1. Rangkaian minimum sistem AT89S51 (master).
2. Rangkaian minimum sistem AT89S51 (slave).
3. Rangkaian Serial RS485 (master).
4. Rangkaian Serial RS485 (slave).
5. Rangkaian Serial RS232
6. Rangkaian LED.
7. Rangkaian LCD.
8. Rangkaian Keypad.
9. Catu daya 5 Volt
10. Stopwatch.

### **4.6.3. Langkah Pengujian**

1. Menghubungkan rangkaian keypad matrik 4x4 ke port 1 pada minimum sistem AT89S51 (master).
2. Menghubungkan data LCD ke port 0 pada minimum sistem AT89S51 (master).
3. Menghubungkan RS LCD ke port 2.7 dan EN LCD pada port 2.6 pada minimum sistem AT89S51 (master).



4. Menghubungkan jalur data A dan B serial RS485 master ke RS485 slave.
5. Menghubungkan pin D1 dan R0 serial RS485 master ke pin Rx dan Tx pada minimum sistem AT89S51 master.
6. Menghubungkan pin D1 dan R0 serial RS485 slave ke pin Rx dan Tx pada minimum sistem AT89S51 slave.
7. Menghubungkan pin DE dan RE serial RS485 master ke pin 3.2 pada minimum sistem AT89S51 master.
8. Menghubungkan pin DE dan RE serial RS485 slave ke pin 3.2 pada minimum sistem AT89S51 slave.
9. Menghubungkan rangkaian LED ke port 0, 1, dan 2 pada minimum sistem AT89S51 slave.
10. Mengamati hasil pengujian.

#### **4.6.4. Hasil dan Analisa Pengujian**

Setelah semua Rangkaian dihubungkan dan diberikan catu daya maka sistem akan menampilkan tampilan sebagai berikut:



**Gambar 4.7. Tampilan LCD “Judul”**



**Gambar 4.8. Tampilan LCD “Disusun dan Nama “**

Pilihan menu pada sistem terdapat 2 pilihan, yaitu default seting, dan seting traffic.



**Tabel 4.9. Pilihan Menu Pada Sistem**

No	Menu
1	Default seting
2	Seting traffic

Pada pilihan pertama “*default seting*” akan menjalankan program *traffic light* dengan timer yang sudah ditentukan nilainya. Program *default* akan dijalankan langsung ketika sistem diberi catu daya, dan diinginkan untuk menyeting awal *traffic light* ketika arus lalulintas kembali normal setelah adanya perubahan nilai timer pada setingan traffic.

Pada menu pilihan kedua “seting traffic”. Didalamnya terdapat pilihan-pilihan menu untuk melakukan perubahan seting traffic yang ingin diubah besarnya nilai delay traffic, dengan memasukan angka melalui penekanan keypad. Begitu seting traffic selesai akan kembali ke menu awal dan delay yang baru dimasukan akan dijalankan.



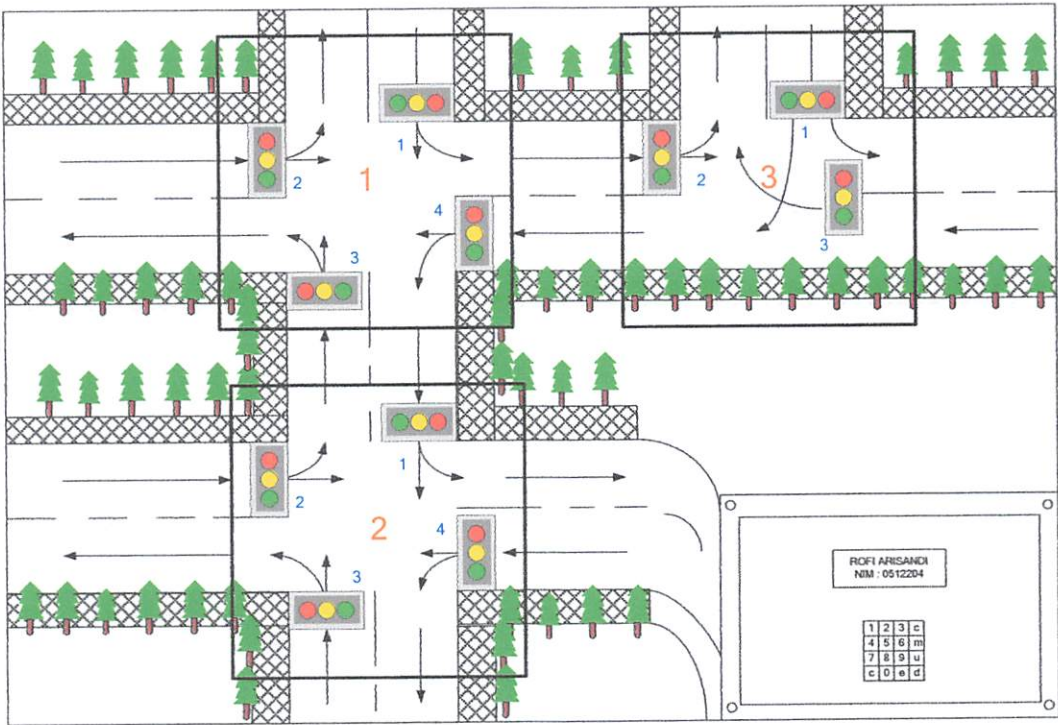
**Gambar 4.10. Tampilan LCD Pilihan Menu “Seting Traffic”**

Pada “Status Traffic” akan menunjukkan status atau kondisi dari traffic yang didalamnya terdapat pilihan untuk mengetahui kondisi dari traffic 1, 2, atau 3.

Pada menu “Yellow Light” akan menyebabkan lampu kuning pada traffic 1, 2, atau 3 menyala.



**Gambar 11. Tampilan LCD Menu “Yellow Light”**



**Gambar 4.12. Denah Penempatan Traffic Light**

#### 4.6.4.1. Hasil dan Analisa Pengujian Traffic 1

Pada pengujian traffic 1 diberikan nilai timer secara acak sehingga diperoleh data seperti pada tabel 4.5.

**Tabel 4.5. Hasil Pengujian Traffic 1**

Pengujian ke	Light	Data masukan (detik)	Hasil pengujian (detik)	Error (%)
1	1	10	10,16	1,6
2		16	16,18	1,125
3		21	21,18	0,85
4		33	33,54	1,63

5	2	12	12,10	0,833
6		16	16,14	0,87
7		20	20,25	1,25
8		30	30,20	1
9	3	10	10,21	2,1
10		18	18,18	1
11		20	20,35	1,75
12		28	28,49	1,75
13	4	11	11,27	2,45
14		17	17,36	2,11
15		29	29,50	1,72
16		31	31,27	0,87

Dari tabel hasil percobaan diatas dapat diketahui bahwa prosentase error yang terjadi antara 0,833% sampai dengan 2,45%. Total error yang terjadi sebesar = 22,163%. Maka rata-rata prosentase error total sebesar yang terjadi 1,385% .

#### **4.6.4.2. Hasil dan Analisa Pengujian Traffic 2**

Pada pengujian traffic 2 diberikan nilai timer secara acak sehingga diperoleh data seperti pada tabel 4.6.

**Tabel 4.6. Hasil Pengujian Traffic 2**

Pengujian ke	Light	Data masukan (detik)	Hasil pengujian (detik)	Error (%)
1	1	10	10,22	2,2
2		16	16,14	0,875
3		21	21,18	0,857
4		33	33,55	1,66
5	2	12	12,10	0,833
6		16	16,45	2,81
7		20	20,55	2,75
8		30	30,20	0,66
9	3	10	10,14	1,4
10		18	18,21	1,16
11		20	20,31	1,55
12		28	28,29	1,03
13	4	11	11,19	1,72
14		17	17,46	2,7
15		29	29,37	1,27
16		31	31,27	0,87

Dari tabel hasil percobaan diatas dapat diketahui bahwa total error yang terjadi pada pengujian traffic 2 sebesar 24,345%. rata-rata prosentase error total sebesar 1,521%

#### 4.6.4.3. Hasil dan Analisa Pengujian Traffic 3

Pada pengujian traffic 3 diberikan nilai timer secara acak sehingga diperoleh data seperti pada tabel 4.7.

**Tabel 4.7. Hasil Pengujian Traffic 3**

Pengujian ke	Light	Data masukan (detik)	Hasil pengujian (detik)	Error (%)
1	1	10	10,09	0,9
2		16	16,13	0,81
3		21	21,18	0,857
4		33	33,65	1,96
5	2	12	12,10	0,833
6		16	16,51	3,18
7		20	20,24	1,2
8		30	30,20	0,66
9	3	10	10,11	1,1
10		18	18,21	1,16
11		20	20,43	2,15
12		28	28,61	2,17

Dari tabel hasil pengujian traffic 3 diketahui prosentase error pada traffic 3 sebesar 16,98%. Sehingga diperoleh rata-rata prosentase error total sebesar 1,06%.

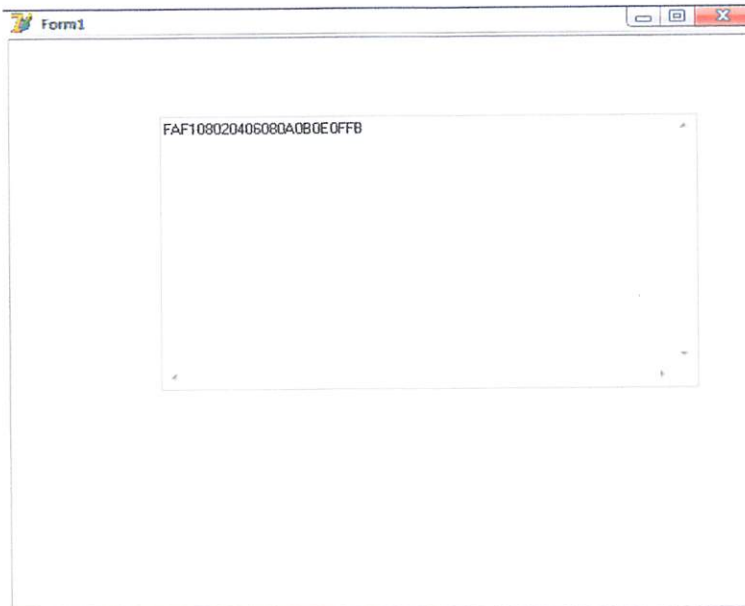


Pengujian timer pada ketiga traffic pada masing-masing lampu diperoleh nilai error pada ketiga traffic adalah sebesar 1,9%, 3,39%, dan 1,93%. Sehingga error total yang terjadi pada timer dalam sistem sebesar :

$$\begin{aligned}
 \text{Error total timer} &= \frac{\sum \text{error\_traffic}}{\sum \text{traffic}} \\
 &= \frac{1,385\% + 1,521\% + 1,06\%}{3} \\
 &= 1.476\%
 \end{aligned}$$

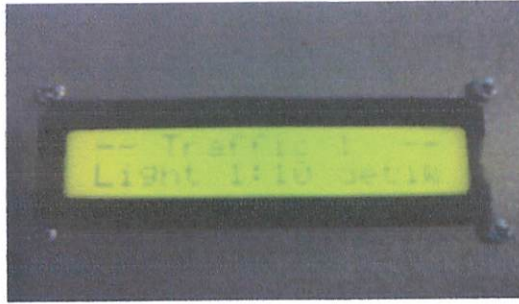
#### 4.6.4.4. Hasil dan Analisa Pengujian Pengiriman dan Penerimaan Data Serial.

Pada pengujian pengiriman data, dilakukan setingan pada traffic dengan memasukan nilai timer baru. Sedangkan pada pengujian penerimaan data serial dengan melihat status traffic melalui LCD. Sehingga diperoleh hasil seperti pada gambar 4.13



**Gambar 4.13. Hasil Pengiriman Data Serial Melalui RS485**





**Gambar 4.14. Hasil Penerimaan Status Timer Traffic**

Dari pengujian pengiriman dan penerimaan data serial melalui RS485, diketahui bahwa data yang dikirim dan diterima oleh mikrokontroler master adalah sama, sehingga dapat dikatakan pada pengujian error yang terjadi sebesar 0%.

Dari hasil pengujian keseluruhan system mulai dari pengujian timer pada traffic 1, 2, dan 3 serta pengujian pengiriman dan penerimaan data serial. Dapat diperoleh prosentase error sistem

$$\begin{aligned} \text{Error Sistem} &= \frac{\sum \text{error}}{\sum \text{pengujian}} \\ &= \frac{1,385\% + 1,521\% + 1,06\% + 0\%}{4} \\ &= 1,476\% \end{aligned}$$

Sehingga diketahui besarnya prosentase error pada system sebesar **1,476%**.

## **BAB V**

### **PENUTUP**

#### **5.1. Kesimpulan**

Setelah dilakukan pengujian terhadap keseluruhan dari sistem alat *traffic light* terintegrasi melalui komunikasi serial berbasis AT89S51, dapat diambil beberapa kesimpulan diantaranya:

1. Pada proses penyetingan *delay* pada salah satu *traffic* tidak mempengaruhi *traffic* yang lainnya.
2. Pada saat ada dilakukan setingan nilai timer baru, sistem tidak menjalankan timer yang baru. Tetapi menyelesaikan pergantian lampu dengan nilai timer lama, setelah pergantian selesai atau kembali pada *traffic* pertama yang menyala hijau, timer setingan terbaru akan dijalankan.
3. Nilai total error sistem yang terjadi adalah sebesar 1,476%.

#### **5.2. Saran**

1. Untuk memperoleh hasil yang lebih baik dalam pengukuran, digunakan alat yang lebih akurat.
2. Dalam pengembangannya dapat ditambahkan penunjuk waktu *delay*, kamera dan aplikasi web.

## DAFTAR PUSTAKA

- [1] [www.Alldatasheet.com](http://www.Alldatasheet.com)
- [2] [www.My-tutorialcafe.com](http://www.My-tutorialcafe.com)
- [3] [DatasheetMikrokontrolerAt89s51.http:// www.ATMEL.com](http://www.ATMEL.com)
- [4] Widodo Budiharto, Sigit Firmansyah, Elektronika Digital dan Mikroprosesor, Andi, Yogyakarta, 1990.
- [5] Agfianto, Belajar Mikrokontroler, Gava Media, Yogyakarta, 2004.
- [6] Tim Lab Mikroprosesor, Pemrograman Mikrokontroler AT89S51, Andi, Yogyakarta, 2007.
- [7] [DatasheetAt24C64.http:// www.ATMEL.com](http://www.ATMEL.com)
- [8] an0055, teknik akses EEPROM. [http:// www.Delta-Elektronik.com](http://www.Delta-Elektronik.com)
- [9] [Half-duplex.htm:// www.wikipedia.com](http://www.wikipedia.com)
- [10] [RS-485.html:// www.lammertbier.nl](http://www.lammertbier.nl)
- [11] [RS-485\\_wiki.htm:// www.wikipedia.com](http://www.wikipedia.com)
- [12] [MAX485.pdf .:// www. Maxim.com](http://www.Maxim.com)

# LAMPIRAN



## FORMULIR BIMBINGAN SKRIPSI

Nama : ROFI ARISANDI  
Nim : 05.12.204  
Masa Bimbingan : 14 September 2009 s/d 14 Maret 2010  
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN TRAFFIC LIGHT  
TERINTEGRASI MELALUI KOMUNIKASI SERIAL  
BERBASIS AT89S51**

No.	Tanggal	Uraian	Paraf Pembimbing
1.	20-10-09	BAB I	
2.	31-10-09	BAB II	
3.	19-11-09	BAB III	
4.	05-01-10	BAB IV	
5.	08-01-10	Pengujian Keypad	
6.	08-01-10	Pengujian Timer	
7.	19-01-10	Abstraksi	
8.	21-01-10	BAB V	
9.	21-01-10	Kesimpulan	
10.			

Malang, Februari 2010  
Dosen Pembimbing,


**(I. Komang Somawirata, ST, MT)**  
NIP. 103 0100 361



**INSTITUT TEKNOLOGI NASIONAL MALANG**  
**FAKULTAS TEKNOLOGI INDUSTRI**  
**JURUSAN TEKNIK ELEKTRO S-1**  
Jl. Karanglo KM.2 Malang

## FORMULIR PERBAIKAN SKRIPSI

Nama : Rofi Arisandi  
Nim : 05.12.204  
Jurusan : Teknik Elektro S-1  
Konsentrasi : Teknik Elektronika  
Masa Bimbingan : 14 September 2009 – 14 Maret 2010  
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN *TRAFFIC LIGHT***  
**TERINTEGRASI MELALUI KOMUNIKASI SERIAL**  
**BERBASIS AT89S51**

No	Penguji	Tanggal	Uraian	Paraf
1.	Penguji I	09 Februari 2010	Rangkaian Lengkap	

**Disetujui**

**Penguji I**

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

**Mengetahui**

**Dosen Pembimbing**

  
**I. Komang Somawirata ST, MT.**  
NIP.Y. 103 0100 361



INSTITUT TEKNOLOGI NASIONAL MALANG  
FAKULTAS TEKNOLOGI INDUSTRI  
JURUSAN TEKNIK ELEKTRO

## Formulir Perbaikan Ujian Skripsi

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

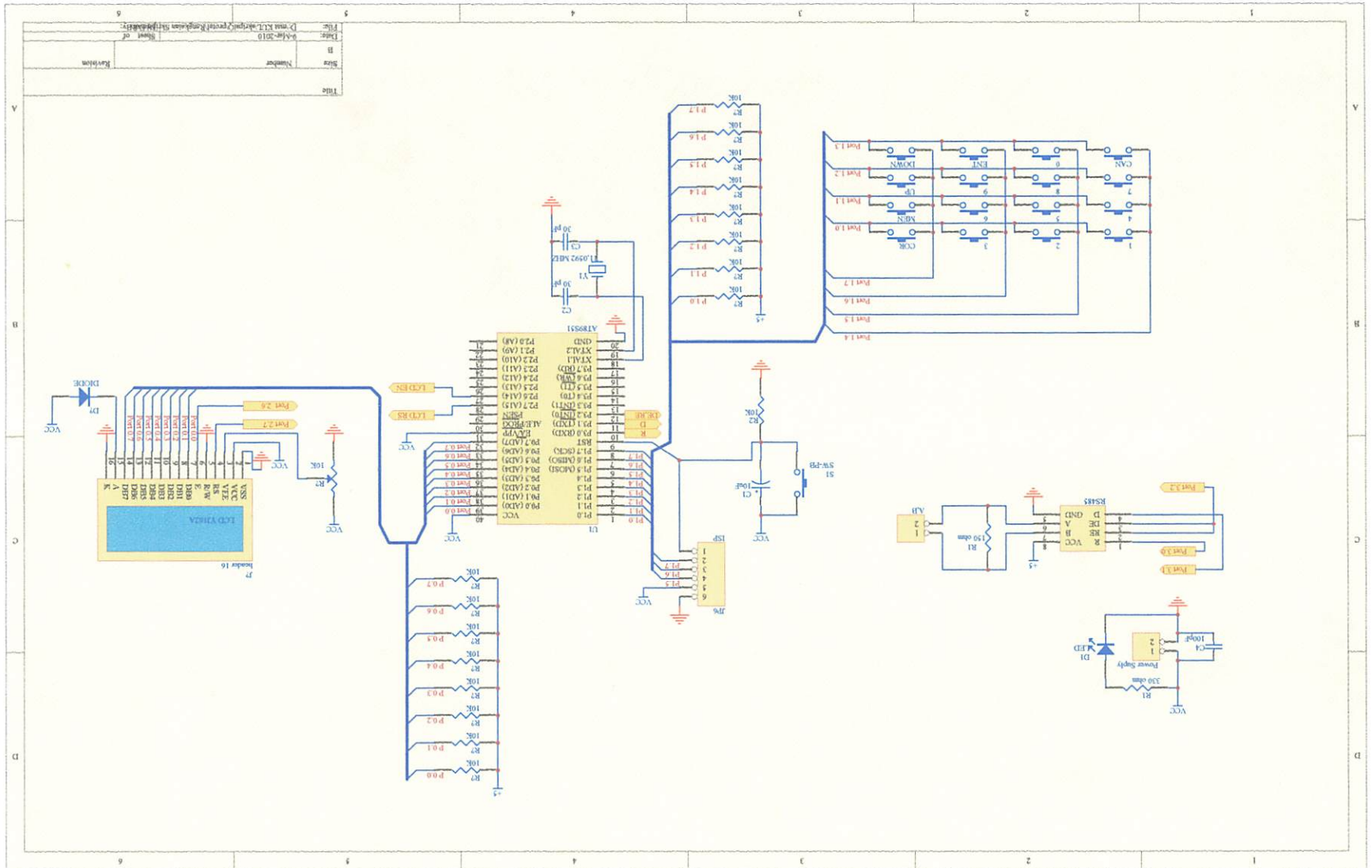
NAMA : NOFI ARISANDI  
NIM : 0512205  
Perbaikan meliputi :

\*) RENCANA LEMBAR P.

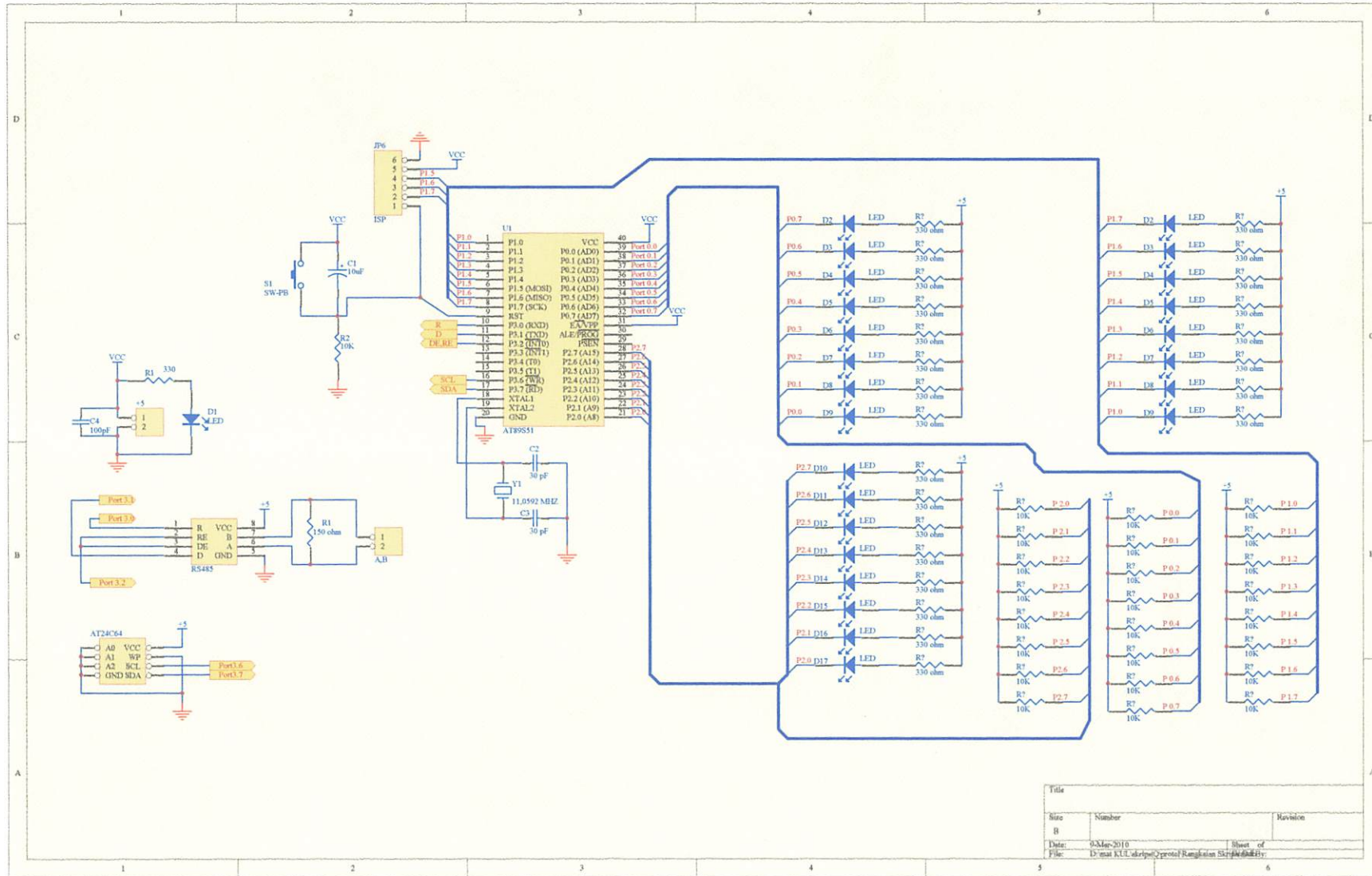
## **LAMPIRAN SKEMATIK**



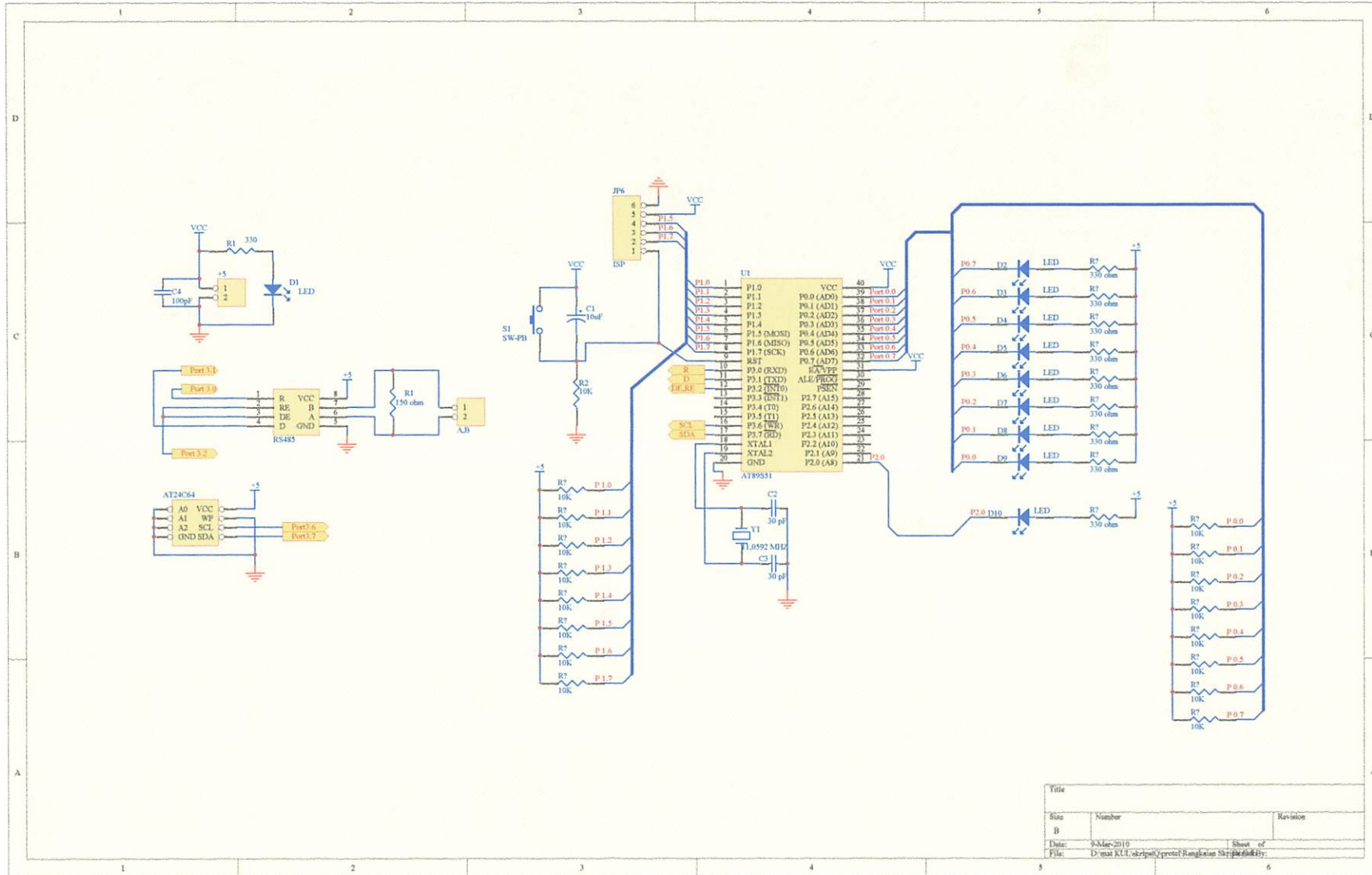
# Rangkaian Lengkap Master



# Rangkaian Lengkap Slave 1



# Rangkaian Lengkap Slave 2



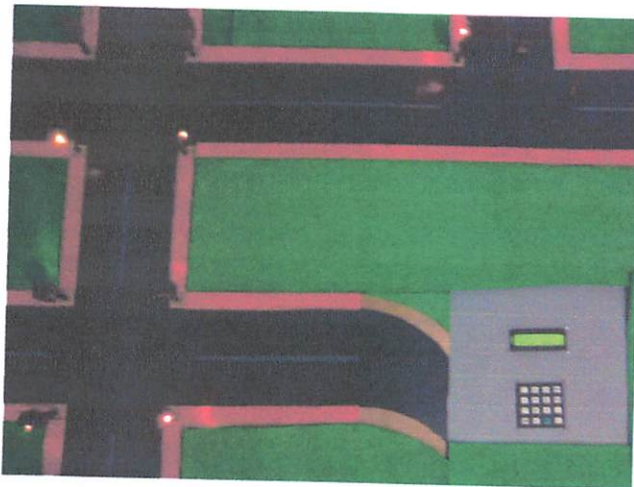
Title		
Status	Number	Revision
B		
Date:	30 May 2010	Sheet 4 of 4
File:	D:\msd\KCU\skripsi\proyek\Rangkaian Slave 2\proyek2.Rbr	

## **LAMPIRAN FOTO ALAT**

## Lampiran Gambar



Gambar Box Kontrol



Gambar Keseluruhan

## **LAMPIRAN LISTING PROGRAM**

## Listing Program Master

LCD\_EN BIT P2.6

LCD\_RS BIT P2.7

KEY BIT 20H.0

SETTING BIT 20H.1

SET\_HEADER BIT 20H.2

SELECT BIT 20H.3

STATUS BIT 20H.4

COLUM4 BIT P1.7

COLUM3 BIT P1.6

COLUM2 BIT P1.5

COLUM1 BIT P1.4

ENABLE\_RX BIT P3.2 ;SELECTOR RS 485

-----

; INISIAL DATA KEYPAD

-----

COR EQU 0FAH

MEN EQU 0FBH

CAN EQU 0FCH

UP EQU 0FDH

DOWN EQU 0FEH

ENT EQU 0FFH

TULIS EQU 30H  
POSISI EQU 31H  
STATUS\_TRAFFIC EQU 32H  
BATAS EQU 33H  
STEP1 EQU 34H  
STEP2 EQU 35H  
TRAFFIC EQU 36H  
LIGHT EQU 37H  
HITUNG EQU 38H

SET\_TRAFFIC1\_L1 EQU 38H  
SET\_TRAFFIC1\_L2 EQU 39H  
SET\_TRAFFIC1\_L3 EQU 3AH  
SET\_TRAFFIC1\_L4 EQU 3BH

SET\_TRAFFIC2\_L1 EQU 3CH  
SET\_TRAFFIC2\_L2 EQU 3DH  
SET\_TRAFFIC2\_L3 EQU 3EH  
SET\_TRAFFIC2\_L4 EQU 3FH

SET\_TRAFFIC3\_L1 EQU 40H  
SET\_TRAFFIC3\_L2 EQU 41H  
SET\_TRAFFIC3\_L3 EQU 42H

KURSOR EQU 43H



SLAVE\_NUMBER EQU 44H ; POSISI NO REFERENSI SLAVE  
RS 485

PACKET\_DATA EQU 45H

TEMP EQU 46H

HEADER EQU 0FAH ;CODE AWAL COMMAND

SLAVE\_NUMBER1 EQU 0F1H ; NO REFERENSI SLAVE 1 RS 485

SLAVE\_NUMBER2 EQU 0F2H ; NO REFERENSI SLAVE 2 RS 485

END\_COMMAND EQU 0FBH

YELLOW\_LIGHT\_ON1 EQU 0FCH

YELLOW\_LIGHT\_ON2 EQU 0FDH

YELLOW\_LIGHT\_OFF1 EQU 0FEH

YELLOW\_LIGHT\_OFF2 EQU 0FFH

STATUS\_DELAY EQU 0F9H

SET\_DEFAULT EQU 0F8H

ORG 00H

LJMP START

ORG 0BH

LJMP Timer

```
ORG 23H
LJMP SERIAL
```

```
ORG 100H
```

```
START:SETBEA
```

```
MOV TMOD,#21H
MOV TH1,#0FDH      ;9600BPS
SETB TR1
```

```
MOV TH0,#0H
MOV TL0,#0H
SETB ET0
```

```
MOV HITUNG,#10
```

```
MOV SCON,#50H
MOV  A,#03Fh      ;FUNCTION SET LCD
CALL     INSTRUKSI_LCD
CALL     INSTRUKSI_LCD
CALL DELAY
MOV  A,#0Ch      ;ON/OFF CONTROL,CURSOR
CALL     INSTRUKSI_LCD
MOV  A,#06h
CALL     INSTRUKSI_LCD
MOV  A,#01h      ;CLEAR DISPLAY
CALL     INSTRUKSI_LCD
```

```
MOV A,#0Ch
CALL INSTRUKSI_LCD
CALL DELAY
MOV DPTR,#JUDUL
CALL DISPLAY
CALL LONG_DELAY
```

```
CALL DISPLAY
CALL LONG_DELAY
```

```
CALL DISPLAY
CALL LONG_DELAY
```

```
SETB SET_HEADER ;RS485 READY
SETB SELECT
CLR ENABLE_RX
DJNZ R7,$
SETB ES
```

MENU\_AWAL:

```
MOV DPTR,#AWAL
CALL BARISA
CALL BARISB
```

CEKT:CALL BACA\_KEYPAD

```
JB KEY,CEKT
```

CEKM1: CJNE A,#MEN,CEKT

```
MOV POSISI,#1
```

```

MOV DPTR,#MENU1
CALL DISPLAY
CEK2: CALL BACA_KEYPAD
JB KEY,CEK2
CJNE A,#DOWN,TDOWN ;TAMPILAN TURUN SAAT
TEKAN PANAHAH DOWN
MOV A,POSISI
CJNE A,#2,NAIKIN ;JUMLAH MENU
JMP CEK2
NAIKIN:INC POSISI
CALL TAMPIL_MENU1
JMP CEK2

TDOWN:
CJNE A,#UP,TCAN
MOV A,POSISI
CJNE A,#1,TURUNIN
JMP CEK2

TURUNIN:
DEC POSISI
CALL TAMPIL_MENU1
JMP CEK2

TCAN: CJNE A,#CAN,TEN
JMP MENU_AWAL

TEN: CJNE A,#ENT,CEK2

```

CALL TAMPIL\_MENU2

JMP CEK2

-----  
; PILIHAN MENU AWAL  
-----

TAMPIL\_MENU1:

MOV A,POSISI

CJNE A,#1,MN2

MOV DPTR,#MENU1

JMP TPL

MN2: CJNE A,#2,MN

MOV DPTR,#MENU2

TPL: CALL BARISA

CALL BARISB

MN: RET

TAMPIL\_MENU2:

MOV A,POSISI

CJNE A,#1,TRAF1

CALL DEFAULT\_SETING

RET

TRAF1: CJNE A,#2,TRAF2

CALL TRAFFIC1

RET

TRAF2: CJNE A,#3,TRAF

TRAF: RET

TRAFFIC1:

MOV STEP1,POSISI

MOV POSISI,#1

CALL TAMPIL\_MENUTRAF1

CEK6: CALL BACA\_KEYPAD

JB KEY,CEK6

CJNE A,#DOWN,TDOWN1 ;TAMPILAN TURUN SAAT  
TEKAN PANAHAH DOWN

MOV A,POSISI

CJNE A,#3,NAIKIN1 ;3 menu

JMP CEK6

NAIKIN1:

INC POSISI

CALL TAMPIL\_MENUTRAF1

JMP CEK6

TDOWN1:

CJNE A,#UP,TCAN1

MOV A,POSISI

CJNE A,#1,TURUNIN1

JMP CEK6

TURUNIN1:

DEC POSISI

CALL TAMPIL\_MENUTRAF1

JMP CEK6

TCAN1: CJNE A,#CAN,TEN1

MOV POSISI,STEP1

CALL TAMPIL\_MENU1

RET

TEN1: CJNE A,#ENT,CEK6

CALL TAMPIL\_MENUTRAF2 ;MASUK SUB MENU  
PULIHAN SETING, STATUS DAN DELAY TRAFFIC

CALL KIRIM\_SETING

CALL DELAY

MOV POSISI,STEP1

CALL TAMPIL\_MENU1

RET

-----  
;ROUTINE PENGIRIMAN SETING DELAY TRAFFIC DARI MASTER KE  
SLAVE

-----  
KIRIM\_SETING:

CLR TR0 ;TIMER NON AKTIF

MOV SLAVE\_NUMBER,#SLAVE\_NUMBER1 ;KIRIM KE SLAVE  
1 RS485

MOV PACKET\_DATA,#8 ;JUMLAH PAKET DATA

MOV TEMP,#SET\_TRAFFIC1\_L1 ;ALAMAT PKET DATA

CALL SEND485

CALL DELAY

MOV SLAVE\_NUMBER,#SLAVE\_NUMBER2 ;KIRIM KE  
SLAVE 2 RS485

```
MOV PACKET_DATA,#3           ;JUMLAH PAKET DATA
MOV TEMP,#SET_TRAFFIC3_L1    ;ALAMAT PKET DATA
CALL SEND485
CALL DELAY
RET
```

```
TIMER:    CLR  TR0
          PUSH ACC
OUTP:POP  ACC
          SETB TR0
          RETI
```

CEKASCII:

```
MOV TEMP,A
ANL  A,#0F0H
SWAP A
ORL  A,#30H
CALL DATA_LCD
MOV  A,TEMP
ANL  A,#0FH
ORL  A,#30H
CALL DATA_LCD
MOV  A,#':'
CALL DATA_LCD
RET
```



-----  
;MENU TRAFIC PILIHAN DELAY,STATUS, YELLOW LAMPU 1,2,3,4  
-----

TAMPIL\_MENUTRAF1:

MOV A,POSISI

CJNE A,#1,MN2T

MOV DPTR,#MENUTRAF1

JMP TPLT

MN2T: CJNE A,#2,MN3T

MOV DPTR,#MENUTRAF2

JMP TPLT

MN3T: CJNE A,#3,MNT

MOV DPTR,#MENUTRAF3

TPLT: CALL BARISA

CALL BARISB

MNT: RET

TAMPIL\_MENUTRAF2:

MOV STEP2,POSISI

MOV A,POSISI

CJNE A,#1,STATUSD

CALL MENU\_SETING\_TRAFFIC ;MASUK MENU  
PENGISIAN SETING

MOV POSISI,STEP2

CALL TAMPIL\_MENUTRAF1

RET

STATUSD:

```
CJNE A,#2,YELLOWD
CALL MENU_STATUS_DELAY
MOV POSISI,STEP2
CALL TAMPIL_MENUTRAF1
RET
```

YELLOWD:

```
CJNE A,#3,STAOUT
CALL SCAN_YELLOW_LIGHT
MOV POSISI,STEP2
CALL TAMPIL_MENUTRAF1
```

STAOUT:

```
RET
```

SEND\_DEFAULT\_TRAFFIC12:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER1
CALL TX_SEND
MOV A,#SET_DEFAULT
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

SEND\_DEFAULT\_TRAFFIC3:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER2
CALL TX_SEND
MOV A,#SET_DEFAULT
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

SEND\_REQUEST\_DATA\_DELAY1:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER1
CALL TX_SEND
MOV A,#STATUS_DELAY
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

SEND\_REQUEST\_DATA\_DELAY2:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER2
CALL TX_SEND
MOV A,#STATUS_DELAY
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

MENU\_STATUS\_DELAY:

```
CALL SEND_REQUEST_DATA_DELAY1
MOV DPTR,#LOADING
CALL DISPLAY
CALL LONG_DELAY
CALL SEND_REQUEST_DATA_DELAY2
CALL LONG_DELAY
MOV DPTR,#DELAY_TRAFFIC
CALL DISPLAY
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC1_L1
CALL TAMPIL_ASCII
```

CEK9: CALL BACA\_KEYPAD

JB KEY,CEK9

;TAMPILKAN DATA SETING T1L2

MOV DPTR,#DELAY\_TRAFFIC

CALL DISPLAY

MOV A,#0C6H

CALL INSTRUKSI\_LCD

MOV A,#2'

CALL DATA\_LCD

MOV A,#0C8H

CALL INSTRUKSI\_LCD

MOV A,SET\_TRAFFIC1\_L2

CALL TAMPIL\_ASCII

CEK9A: CALL BACA\_KEYPAD

JB KEY,CEK9A

;TAMPILKAN DATA SETING T1L3

MOV DPTR,#DELAY\_TRAFFIC

CALL DISPLAY

CALL DATA\_LCD

MOV A,#0C6H

CALL INSTRUKSI\_LCD

MOV A,#3'

CALL DATA\_LCD

```
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC1_L3
CALL TAMPIL_ASCII
CEK9B: . CALL BACA_KEYPAD
JB KEY,CEK9B
```

```
;TAMPILKAN DATA SETING T1L4
```

```
MOV DPTR,#DELAY_TRAFFIC
CALL DISPLAY
MOV A,#0C6H
CALL INSTRUKSI_LCD
MOV A,#'4'
CALL DATA_LCD
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC1_L4
CALL TAMPIL_ASCII
CEK9C: CALL BACA_KEYPAD
JB KEY,CEK9C
```

```
;TAMPILKAN DATA SETING T2L1
```

```
MOV DPTR,#DELAY_TRAFFIC
CALL DISPLAY
MOV A,#8BH
CALL INSTRUKSI_LCD
```

```

MOV A,#'2'                ;TRAFFIC 2
CALL DATA_LCD
MOV A,#0C6H
CALL INSTRUKSI_LCD
MOV A,#'1'                ;LIGHT 1
CALL DATA_LCD
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC2_L1
CALL TAMPIL_ASCII
CEK9D:  CALL BACA_KEYPAD
        JB  KEY,CEK9D

;TAMPILKAN DATA SETING T2L2
MOV DPTR,#DELAY_TRAFFIC
CALL DISPLAY
MOV A,#8BH
CALL INSTRUKSI_LCD
MOV A,#'2'
CALL DATA_LCD
MOV A,#0C6H
CALL INSTRUKSI_LCD
MOV A,#'2'                ;LIGHT 2
CALL DATA_LCD
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC2_L2

```

```

CALL TAMPIL_ASCII
CEK9E:    CALL BACA_KEYPAD
          JB    KEY,CEK9E

          ;TAMPILKAN DATA SETING T2L3
          MOV  DPTR,#DELAY_TRAFFIC
          CALL DISPLAY
          MOV  A,#8BH
          CALL INSTRUKSI_LCD
          MOV  A,#'2'
          CALL DATA_LCD
          MOV  A,#0C6H
          CALL INSTRUKSI_LCD
          MOV  A,#'3'           ;LIGHT 3
          CALL DATA_LCD
          MOV  A,#0C8H
          CALL INSTRUKSI_LCD
          MOV  A,SET_TRAFFIC2_L3
          CALL TAMPIL_ASCII
CEK9F:    CALL BACA_KEYPAD
          JB    KEY,CEK9F

          ;TAMPILKAN DATA SETING T2L4
          MOV  DPTR,#DELAY_TRAFFIC
          CALL DISPLAY
          MOV  A,#8BH
          CALL INSTRUKSI_LCD

```



```
MOV A,#'2'  
CALL DATA_LCD  
MOV A,#0C6H  
CALL INSTRUKSI_LCD  
MOV A,#'4' ;LIGHT 4  
CALL DATA_LCD  
MOV A,#0C8H  
CALL INSTRUKSI_LCD  
MOV A,SET_TRAFFIC2_L4  
CALL TAMPIL_ASCII  
CEK9G: CALL BACA_KEYPAD  
JB KEY,CEK9G
```

```
;TAMPILKAN DATA SETING T3L1  
MOV DPTR,#DELAY_TRAFFIC  
CALL DISPLAY  
MOV A,#8BH  
CALL INSTRUKSI_LCD  
MOV A,#'3'  
CALL DATA_LCD  
MOV A,#0C6H  
CALL INSTRUKSI_LCD  
MOV A,#'1' ;LIGHT 1  
CALL DATA_LCD  
MOV A,#0C8H  
CALL INSTRUKSI_LCD  
MOV A,SET_TRAFFIC3_L1
```

CALL TAMPIL\_ASCII

CEK9H: CALL BACA\_KEYPAD

JB KEY,CEK9H

;TAMPILKAN DATA SETING T3L2

MOV DPTR,#DELAY\_TRAFFIC

CALL DISPLAY

MOV A,#8BH

CALL INSTRUKSI\_LCD

MOV A,#'3'

CALL DATA\_LCD

MOV A,#0C6H

CALL INSTRUKSI\_LCD

MOV A,#'2' ;LIGHT 2

CALL DATA\_LCD

MOV A,#0C8H

CALL INSTRUKSI\_LCD

MOV A,SET\_TRAFFIC3\_L2

CALL TAMPIL\_ASCII

CEK9I: CALL BACA\_KEYPAD

JB KEY,CEK9I

;TAMPILKAN DATA SETING T3L3

MOV DPTR,#DELAY\_TRAFFIC

CALL DISPLAY

```

MOV A,#8BH
CALL INSTRUKSI_LCD
MOV A,#'3'
CALL DATA_LCD
MOV A,#0C6H
CALL INSTRUKSI_LCD
MOV A,#'3' ;LIGHT 3
CALL DATA_LCD
MOV A,#0C8H
CALL INSTRUKSI_LCD
MOV A,SET_TRAFFIC3_L3
CALL TAMPIL_ASCII
CEK9J: CALL BACA_KEYPAD
JB KEY,CEK9J
RET

```

TAMPIL\_ASCII:

```

CLR C
DA A ;KONVERSI DESIMAL
MOV B,A
ANL A,#0F0H
SWAP A
CJNE A,#0,NO
JMP LJT
NO: ORL A,#30H
CALL DATA_LCD
LJT: MOV A,B

```

```
ANL A,#0FH
ORL A,#30H
CALL DATA_LCD
RET
```

-----

;MENU PILIHAN SETING YELLOW

-----

SCAN\_YELLOW\_LIGHT:

```
CALL TAMPIL_INFO_YELLOW
MOV A,#1'
CALL DATA_LCD
CALL PROSES_YELLOW
JB STATUS,YELLOW_OFF1
CALL KIRIM_YELLOW_TRAFFIC_ON1
CALL TAMPIL_MODE_YELLOW
MOV A,#1'
CALL DATA_LCD
JMP YELLOW_KE2
```

YELLOW\_OFF1:

```
CALL KIRIM_YELLOW_TRAFFIC_OFF1
CALL TAMPIL_MODE_NORMAL
MOV A,#1'
CALL DATA_LCD
```

YELLOW\_KE2:

```
CALL LONG_DELAY
```

```
CALL TAMPIL_INFO_YELLOW
MOV A,#'2'
CALL DATA_LCD
CALL PROSES_YELLOW
JB STATUS,YELLOW_OFF2
CALL KIRIM_YELLOW_TRAFFIC_ON2
CALL TAMPIL_MODE_YELLOW
MOV A,#'2'
CALL DATA_LCD
JMP YELLOW_KE3
```

YELLOW\_OFF2:

```
CALL KIRIM_YELLOW_TRAFFIC_OFF2
CALL TAMPIL_MODE_NORMAL
MOV A,#'2'
CALL DATA_LCD
```

YELLOW\_KE3:

```
CALL LONG_DELAY
CALL TAMPIL_INFO_YELLOW
MOV A,#'3'
CALL DATA_LCD
CALL PROSES_YELLOW
JB STATUS,YELLOW_OFF3
CALL KIRIM_YELLOW_TRAFFIC_ON3
CALL TAMPIL_MODE_YELLOW
MOV A,#'1'
CALL DATA_LCD
```

JMP OUTYEL

YELLOW\_OFF3:

CALL KIRIM\_YELLOW\_TRAFFIC\_OFF3

CALL TAMPIL\_MODE\_NORMAL

MOV A,#'3'

CALL DATA\_LCD

OUTYEL:

CALL LONG\_DELAY

RET

TAMPIL\_MODE\_YELLOW:

MOV DPTR,#INFO\_YELLOW

CALL BARISA

MOV DPTR,#YL

CALL BARISB

MOV A,#8FH

CALL INSTRUKSI\_LCD

RET

TAMPIL\_MODE\_NORMAL:

MOV DPTR,#INFO\_YELLOW

CALL BARISA

MOV DPTR,#NM

CALL BARISB

MOV A,#8FH

CALL INSTRUKSI\_LCD

RET

PROSES\_YELLOW:

SETB STATUS

CEK8: CALL BACA\_KEYPAD

JB KEY,CEK8

CAN1B: CJNE A,#CAN,EN1B

RET

EN1B: CJNE A,#ENT,CEK8

CLR STATUS

RET

-----

;STATUS COMMAND YELLOW

KIRIM\_YELLOW\_TRAFFIC\_ON1:

CLR ES

MOV A,#HEADER

CALL TX\_SEND

MOV A,#SLAVE\_NUMBER1 ;TRAFFIC 1 NUMBER

CALL TX\_SEND

MOV A,#YELLOW\_LIGHT\_ON1

CALL TX\_SEND

MOV A,#END\_COMMAND

CALL TX\_SEND

SETB ES

RET

KIRIM\_YELLOW\_TRAFFIC\_OFF1:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER1 ;TRAFFIC 1 NUMBER
CALL TX_SEND
MOV A,#YELLOW_LIGHT_OFF1
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

KIRIM\_YELLOW\_TRAFFIC\_ON2:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER1 ;TRAFFIC 1 NUMBER
CALL TX_SEND
MOV A,#YELLOW_LIGHT_ON2
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

KIRIM\_YELLOW\_TRAFFIC\_OFF2:

```
CLR ES
```



```
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER1           ;TRAFFIC 1 NUMBER
CALL TX_SEND
MOV A,#YELLOW_LIGHT_OFF2
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

KIRIM\_YELLOW\_TRAFFIC\_ON3:

```
CLR ES
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER2           ;TRAFFIC 1 NUMBER
CALL TX_SEND
MOV A,#YELLOW_LIGHT_ON1
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

KIRIM\_YELLOW\_TRAFFIC\_OFF3:

```
CLR ES
MOV A,#HEADER
```

```
CALL TX_SEND
MOV A,#SLAVE_NUMBER2           ;TRAFFIC 1 NUMBER
CALL TX_SEND
MOV A,#YELLOW_LIGHT_OFF1
CALL TX_SEND
MOV A,#END_COMMAND
CALL TX_SEND
SETB ES
RET
```

TAMPIL\_INFO\_YELLOW:

```
MOV DPTR,#TRAF_YELLOW
CALL DISPLAY
MOV A,#8FH
CALL INSTRUKSI_LCD
RET
```

MENU\_SETING\_TRAFFIC:

```
MOV TRAFFIC,#'1'
MOV LIGHT,#'1'
CALL MENU_DELAY_TRAFFIC
JB SETING,LIGHT2
MOV SET_TRAFFIC1_L1,TEMP
```

LIGHT2:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'1'
MOV LIGHT,#'2'
```

```
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT3
MOV SET_TRAFFIC1_L2,TEMP
```

LIGHT3:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'1'
MOV LIGHT,#'3'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT4
MOV SET_TRAFFIC1_L3,TEMP
```

LIGHT4:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'1'
MOV LIGHT,#'4'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT5
MOV SET_TRAFFIC1_L4,TEMP
```

LIGHT5:

```
MOV TRAFFIC,#'2'
MOV LIGHT,#'1'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT6
MOV SET_TRAFFIC2_L1,TEMP
```

LIGHT6:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'2'
MOV LIGHT,#'2'
```

```
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT7
MOV SET_TRAFFIC2_L2,TEMP
```

LIGHT7:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'2'
MOV LIGHT,#'3'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT8
MOV SET_TRAFFIC2_L3,TEMP
```

LIGHT8:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'2'
MOV LIGHT,#'4'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT9
MOV SET_TRAFFIC2_L4,TEMP
```

LIGHT9:

```
MOV TRAFFIC,#'3'
MOV LIGHT,#'1'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT10
MOV SET_TRAFFIC3_L1,TEMP
```

LIGHT10:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'3'
MOV LIGHT,#'2'
```

```
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHT11
MOV SET_TRAFFIC3_L2,TEMP
```

LIGHT11:

```
MOV STEP2,POSISI
MOV TRAFFIC,#'3'
MOV LIGHT,#'3'
CALL MENU_DELAY_TRAFFIC
JB  SETING,LIGHTR
MOV SET_TRAFFIC3_L3,TEMP
```

LIGHTR:

```
MOV POSISI,STEP2
CALL TAMPIL_MENUTRAF1
RET
```

MENU\_DELAY\_TRAFFIC:

```
MOV DPTR,#LIGHTS
CALL DISPLAY
MOV A,#8FH
CALL INSTRUKSI_LCD
MOV A,TRAFFIC
CALL DATA_LCD
MOV A,#0C6H
CALL INSTRUKSI_LCD
MOV A,LIGHT
CALL DATA_LCD
MOV A,#0C8H
```

```

CALL INSTRUKSI_LCD
CALL CURSOR_ON
MOV R0,#TEMP
MOV R2,#1                ;POSISI PENOMORAN KURSOR
SETB SETING
CEK7: CALL BACA_KEYPAD
    JB  KEY,CEK7
    CJNE A,#DOWN,TDOWN2    ;TAMPILAN TURUN SAAT
TEKAN PANAHAH DOWN
    JMP CEK7
TDOWN2:
    CJNE A,#UP,TCAN2
    JMP CEK7
TCAN2:  CJNE A,#CAN,TEN2
    CJNE R2,#1,HAPUS
    CALL CURSOR_OFF
    RET
HAPUS:CJNER2,#2,HAPUS1
    MOV A,#0C8H
    CALL INSTRUKSI_LCD
    MOV A,#'
    CALL DATA_LCD
    MOV A,#0C8H
    CALL INSTRUKSI_LCD
    CALL CURSOR_ON

```

```
DEC R2
JMP CEK7
```

HAPUS1:

```
CJNE R2,#3,CEK7
MOV A,#0C9H
CALL INSTRUKSI_LCD
MOV A,#' '
CALL DATA_LCD
MOV A,#0C9H
CALL INSTRUKSI_LCD
CALL CURSOR_ON
DEC R2
JMP CEK7
```

TEN2: CJNE A,#ENT,MEN2

```
CJNE R2,#1,CKNT2
JMP CEK7
```

CKNT2: CJNE R2,#2,CKNTR

```
MOV A,TEMP
MOV TEMP+1,A
MOV TEMP,#0
```

CKNTR:

```
MOV R0,#TEMP
MOV A,@R0
ANL A,#0FH
SWAP A
MOV TEMP,A
```

```
INC R0
MOV A,@R0
ANL A,#0FH
ORL A,TEMP
MOV TEMP,A ;HASIL DESIMAL HEX
MOV R0,#0
CLR A
```

BANDING:

```
CJNE A,TEMP,TIDAKSAMA
MOV TEMP,R0 ;HASIL KONVERSI DESIMAL KE
HEKSA
CLR SETING ;TANDA SETING OKE
CALL CURSOR_OFF
RET
```

TIDAKSAMA:

```
ADD A,#1
DA A
INC R0 ;NAIKKAN NILAI HEKSA
JMP BANDING
```

```
MEN2: CJNE A,#MEN,COR2
JMP CEK7
RET
```

```
COR2: CJNE A,#COR,ISI2
JMP CEK7
```



RET

ISI2: CJNE R2,#3,OKE2

JMP CEK7

OKE2: CJNE R2,#2,OKE2A

MOV B,A

CALL CURSOR\_OFF

MOV A,B

OKE2A: MOV @R0,A

ORL A,#30H

CALL DATA\_LCD

INC R0

INC R2

;NAIKKAN POSISI KURSOR

JMP CEK7

CURSOR\_ON:

MOV A,#0DH

CALL INSTRUKSI\_LCD

RET

CURSOR\_OFF:

MOV A,#0CH

CALL INSTRUKSI\_LCD

RET

DEFAULT\_SETING:

```
MOV DPTR,#DEFAULT
CALL DISPLAY
CEK5: CALL BACA_KEYPAD
JB KEY,CEK5
CJNE A,#CAN,CKMN3
CALL TAMPIL_MENU1
RET
CKMN3:CJNE A,#ENT,CEK5
CALL KIRIM_SETING_AWAL
MOV DPTR,#INFO_DEFAULT
CALL DISPLAY
CALL LONG_DELAY
CALL TAMPIL_MENU1
RET
```

;------

; KIRIM SETING AWAL KE SLAVE RS485

;------

KIRIM\_SETING\_AWAL:

```
CALL SEND_DEFAULT_TRAFFIC12
DJNZ R6,$
DJNZ R6,$
CALL SEND_DEFAULT_TRAFFIC3
RET
```

BACA\_KEYPAD:

```
SETB KEY
```

```

MOV P1,#11111110B           ;SCAN BARIS 1
JB  COLUM1,NO2
CLR  KEY
MOV  A,#1
JMP  TAHAN
NO2: JB  COLUM2,NO3
CLR  KEY
MOV  A,#2
JMP  TAHAN
NO3: JB  COLUM3,CCOR
CLR  KEY
MOV  A,#3
JMP  TAHAN
CCOR:  JB  COLUM4,NO4
CLR  KEY
MOV  A,#COR
JMP  TAHAN
NO4: MOV P1,#11111101B       ;SCAN BARIS 2
JB  COLUM1,NO5
CLR  KEY
MOV  A,#4
JMP  TAHAN
NO5: JB  COLUM2,NO6
CLR  KEY
MOV  A,#5
JMP  TAHAN
NO6: JB  COLUM3,CMEN

```

CLR KEY

MOV A,#6

JMP TAHAN

CMEN: JB COLUM4,NO7

CLR KEY

MOV A,#MEN

JMP TAHAN

NO7: MOV P1,#11111011B ;SCAN BARIS 3

JB COLUM1,NO8

CLR KEY

MOV A,#7

JMP TAHAN

NO8: JB COLUM2,NO9

CLR KEY

MOV A,#8

JMP TAHAN

NO9: JB COLUM3,CUP

CLR KEY

MOV A,#9

JMP TAHAN

CUP: JB COLUM4,CCAN

CLR KEY

MOV A,#UP

JMP TAHAN

CCAN: MOV P1,#11110111B ;SCAN BARIS 4

JB COLUM1,NOL

CLR KEY

```
    MOV A,#CAN
    JMP TAHAN
NOL: JB  COLUM2,CENT
    CLR KEY
    CLR A
    JMP TAHAN
CENT:JB  COLUM3,CDN
    CLR KEY
    MOV A,#ENT
    JMP TAHAN
CDN: JB  COLUM4,TAHAN
    CLR KEY
    MOV A,#DOWN
TAHAN:
    JNB COLUM1,$
    JNB COLUM2,$
    JNB COLUM3,$
    JNB COLUM4,$
WES: RET

LONG_DELAY:
    MOV R5,#60
LD:  CALL DELAY
    DJNZ R5,LD
    RET

DELAY:  MOV R7,#150
```

DELAY1: MOV R6,#50H

DJNZ R6,\$

DJNZ R7,DELAY1

RET

SEND: CLR ES

CALL TX\_SEND

SETB ES

RET

TX\_SEND:

SETB ENABLE\_RX

NOP

NOP

NOP

NOP

NOP

NOP

MOV SBUF,A

JNB TI,\$

CLR TI

NOP

NOP

NOP

CLR ENABLE\_RX

RET

```
;-----  
;   TRANSFER DATA SERIAL  
;-----
```

SEND485:

SETB ENABLE\_RX

CLR ES

MOV A,#END\_COMMAND

CALL TX\_SEND

MOV A,#HEADER

CALL TX\_SEND

MOV A,#HEADER

CALL TX\_SEND

MOV A,SLAVE\_NUMBER

CALL TX\_SEND

MOV A,PACKET\_DATA

CALL TX\_SEND

NOP

NOP

MOV R2,PACKET\_DATA ;JUMLAH PAKET DATA

MOV R0,TEMP ;ALAMAT DATA

LOOP:MOV A,@R0

CALL TX\_SEND

INC R0

DJNZ R2,LOOP

MOV A,#END\_COMMAND

CALL TX\_SEND

```
CLR  ENABLE_RX
SETB ES
RET
```

```
;-----
;  ROUTINE interupsi serial
;-----
```

SERIAL:

```
CLR  ES
PUSH ACC
JNB  RI,$
MOV  A,SBUF
CLR  RI
JB   SET_HEADER,CEKS
CJNE A,#END_COMMAND,ISI
SETB SET_HEADER
SETB SELECT
CALL SIMPAN
JMP  OUTS
```

```
ISI:  MOV  @R0,A
      INC  R0
      JMP  OUTS
```

```
CEKS: CJNE A,#HEADER,OUTS      ;JIKA  HEADER  DIKENAL,
PROSES RESET BACA DATA
      CLR  SET_HEADER
```



```

MOV R0,#TEMP
OUTS:POP ACC
SETB ES
RETI
.
SLAVE2:
CJNE A,#SLAVE_NUMBER2,OUTPK ;CEK APAKAH DARI
SLAVE2
MOV A,@R0
MOV PACKET_DATA,A ;SIMPAN PAKET DATA
INC R0

MOV R1,#SET_TRAFFIC3_L1
JMP LOOPSAVE

SIMPAN:
MOV R0,#TEMP
MOV A,@R0
INC R0
CJNE A,#SLAVE_NUMBER1,SLAVE2 ;CEK APAKAH DARI
SLAVE1
MOV A,@R0
MOV PACKET_DATA,A ;SIMPAN PAKET DATA
INC R0

MOV R1,#SET_TRAFFIC1_L1
LOOPSAVE:
MOV A,@R0 ;BACA PAKET DATA

```

```
MOV @R1,A ;SIMPAN DI TARGET  
LIGHT DELAY
```

```
INC R0
```

```
INC R1
```

```
DJNZ PACKET_DATA,LOOPSAVE
```

```
OUTPK:
```

```
RET
```

```
;
```

```
; Routine untuk menulis instruksi ke LCD
```

```
;
```

```
INSTRUKSI_LCD:
```

```
CLR LCD_RS
```

```
MOV P0,A ;instruksi ke LCD
```

```
CLR LCD_EN ;module
```

```
SETB LCD_EN
```

```
DJNZ R7,$
```

```
DJNZ R7,$
```

```
RET
```

```
DISPLAY:
```

```
CALL BARISA
```

```
CALL BARISB
```

```
RET
```

```
;
```

```
; Routine untuk menulis data ke LCD
```

```
;
```

DATA\_LCD:

```
SETB LCD_RS
MOV P0,A ;data ke LCD
CLR LCD_EN ;module
SETB LCD_EN
DJNZ r7,$
DJNZ R7,$
RET
```

;

;menuliskan text judul di LCD atas

;

BARISA:

```
MOV TULIS,#16 ; tulis pada baris 1 sebanyak 16 char
MOV A,#80h ; mulai dari kiri atas
CALL INSTRUKSI_LCD
```

TULIS1: CLR A

```
MOVC A,@A+DPTR
INC DPTR
CALL DATA_LCD
DJNZ TULIS,TULIS1
RET
```

;

; menuliskan data dari ram terima

; ke LCD baris 2 sebanyak 16 alamat

---

BARISB:

MOV TULIS,#16 ; tulis pada baris 2 sebanyak 16 char  
MOV A,#0C0H ; mulai dari kiri bawah  
CALL INSTRUKSI\_LCD

TULIS2: CLR A

MOVC A,@A+DPTR  
INC DPTR  
CALL DATA\_LCD  
DJNZ TULIS,TULIS2  
RET

JUDUL: DB 'SISTEM CONTROL'

DB 'Traffic Light'

DB 'Di susun oleh: '

DB 'Rofi Arisandi '

DB 'NIM: '

DB '0512204 '

AWAL: DB 'Tekan MENU '

DB 'Untuk Seting '

LOADING:

DB 'Loading.....'

DB 'Please Wait. '

DELAY\_TRAFFIC:

DB '-- Traffic 1 --'

DB 'Light 1: detik'

MENU1:

DB 'Default Seting ',7FH

DB 'Seting Traffic '

MENU2:

DB 'Default Seting '

DB 'Seting Traffic ',7FH

DEFAULT:

DB 'Default Seting ?'

DB '[CAN] [ENT]'

INFO\_DEFAULT:

DB 'Seting ke '

DB 'pengaturan awal '

MENUTRAF1:

DB 'Seting Traffic ',7FH

DB 'Status Traffic '

MENUTRAF2:

DB 'Status Traffic ',7FH

DB 'Yellow Light '

MENUTRAF3:

DB 'Status Traffic '  
DB 'Yellow Light ',7FH

LIGHTS:-

DB 'Seting Traffic '  
DB 'Light : detik'

TRAF\_YELLOW:

DB 'Yellow Traffic 1'  
DB '[CAN] [ENT]'

INFO\_YELLOW:

DB 'Status Traffic 1'  
NM: DB 'Mode : Normal '  
YL: DB 'Mode : Yellow '

end

## Listing Program Slave 1

```
;-----  
; PIN I2C SEEPROM  
;-----  
SCL          BIT    P3.7  
SDA          BIT    P3.6  
;-----  
;PIN CONTROL MAX 485  
;-----  
ENABLE_RX    BIT    P3.2          ;SELECTOR RS 485  
  
SELECT       BIT    20H.0  
SET_HEADER   BIT    20H.1  
  
CEK_STATUS   BIT    20H.2  
YELLOW_ON1   BIT    20H.3  
YELLOW_ON2   BIT    20H.4  
TOGGLE1      BIT    20H.5  
TOGGLE2      BIT    20H.6  
  
KUNING1      BIT    20H.7  
KUNING2      BIT    21H.0  
KUNING3      BIT    21H.1  
KUNING4      BIT    21H.2
```

KUNING1B	BIT	21H.3
KUNING2B	BIT	21H.4
KUNING3B	BIT	21H.5
KUNING4B	BIT	21H.6
UPDATE	BIT	21H.7
UPLOAD	BIT	22H.0

RED1	BIT	P0.0
YELLOW1	BIT	P0.1
GREEN1	BIT	P0.2

RED2	BIT	P0.3
YELLOW2	BIT	P0.4
GREEN2	BIT	P0.5

RED3	BIT	P0.6
YELLOW3	BIT	P0.7
GREEN3	BIT	P2.0

RED4	BIT	P2.1
YELLOW4	BIT	P2.2
GREEN4	BIT	P2.3

RED1B	BIT	P2.4
YELLOW1B	BIT	P2.5



GREEN1B BIT P2.6

RED2B BIT P2.7

YELLOW2B BIT P1.0

GREEN2B BIT P1.1

RED3B BIT P1.2

YELLOW3B BIT P1.3

GREEN3B BIT P1.4

RED4B BIT P1.5

YELLOW4B BIT P1.6

GREEN4B BIT P1.7

TIMER EQU -4609

HEADER EQU 0FAH ;CODE AWAL COMMAND

SLAVE\_NUMBER EQU 0F1H ; NO REFERENSI SLAVE RS 485

END\_COMMAND EQU 0FBH

YELLOW\_LIGHT\_ON1 EQU 0FCH

YELLOW\_LIGHT\_ON2 EQU 0FDH

YELLOW\_LIGHT\_OFF1 EQU 0FEH

YELLOW\_LIGHT\_OFF2 EQU 0FFH

STATUS\_DELAY EQU 0F9H

SET\_DEFAULT EQU 0F8H

;INISIAL ADDRESS SEEROM

STATUS\_SETING EQU 0

BACKUP\_SETING EQU 1

DATA\_SERIAL EQU 30H

COUNT\_TIMER EQU 31H

PAKET EQU 32H

TIME\_RED1 EQU 33H

TIME\_RED2 EQU 34H

TIME\_RED3 EQU 35H

TIME\_RED4 EQU 36H

TIME\_RED1B EQU 37H

TIME\_RED2B EQU 38H

TIME\_RED3B EQU 39H

TIME\_RED4B EQU 3AH

SETING EQU 3BH

CONTROL1 EQU 3CH

CONTROL2 EQU 3DH

RELOAD1 EQU 3EH

RELOAD2 EQU 3FH

TIME\_YELLOW EQU 40H

TIME\_YELLOW2 EQU 41H

TULIS EQU 42H

BROM EQU 43H

TEMP EQU 44H

ORG 00H

LJMP START

ORG 0BH

LJMP CEK\_TIMER

ORG 23H

LJMP CEK\_SERIAL

ORG 100H

START:SETBEA

MOV TMOD,#21H

MOV TH1,#0FDH ;BAUD RATE 9600

SETB TR1

SETB ET0

MOV TH0,#HIGH TIMER

MOV TL0,#LOW TIMER

MOV COUNT\_TIMER,#200

MOV SCON,#50H ;SERIAL MODE 2

SETB ES

SETB SET\_HEADER

SETB SELECT

CLR ENABLE\_RX

```

DJNZ R7,$
SETB TR0
CALL BACA_ROM_SETING
CALL BACA_ROM_STATUS
MOV A,20H
ORL A,#11100111B
MOV 20H,A
CLR KUNING1B
CLR KUNING1
WAIT:JB  UPLOAD,$
DJNZ R7,$
CLR ES
SETB ENABLE_RX ;RS485 AKTIF
CALL TX_SEND
MOV A,#HEADER
CALL TX_SEND
MOV A,#SLAVE_NUMBER ;SLAVE 1
CALL TX_SEND
MOV A,#8
CALL TX_SEND
CALL SEND_DATA_SETING
MOV A,#END_COMMAND
CALL TX_SEND
CLR ENABLE_RX
SETB UPLOAD

```

SETB ES

JMP WAIT

SETING\_DEFAULT:

MOV CONTROL1,#1

MOV CONTROL2,#1

MOV TIME\_RED1,#12

MOV TIME\_RED2,#12

MOV TIME\_RED3,#12

MOV TIME\_RED4,#12

MOV TIME\_RED1B,#12

MOV TIME\_RED2B,#12

MOV TIME\_RED3B,#12

MOV TIME\_RED4B,#12

MOV RELOAD1,TIME\_RED1

MOV RELOAD2,TIME\_RED1B

CLR KUNING1B

CLR KUNING1

SETB YELLOW\_ON1 ;0 KUNING SAJA, 1 NORMAL

SETB YELLOW\_ON2 ;0 KUNING SAJA, 1 NORMAL

MOV TIME\_YELLOW,#1

MOV TIME\_YELLOW2,#1

RET

OUTJMP:

JMP OUTTIM

TOGGLE\_Y1:

SETB RED1

SETB RED2

SETB RED3

SETB RED4

SETB GREEN1

SETB GREEN2

SETB GREEN3

SETB GREEN4

JB TOGGLE1,ONY1

SETB YELLOW1

SETB YELLOW2

SETB YELLOW3

SETB YELLOW4

RET

ONY1:CLR YELLOW1

CLR YELLOW2

CLR YELLOW3

CLR YELLOW4

RET

TOGGLE\_Y2:

SETB RED1B

SETB RED2B  
SETB RED3B  
SETB RED4B  
SETB GREEN1B  
SETB GREEN2B  
SETB GREEN3B  
SETB GREEN4B  
JB TOGGLE2,ONY2  
SETB YELLOW1B  
SETB YELLOW2B  
SETB YELLOW3B  
SETB YELLOW4B  
RET

ONY2:CLR YELLOW1B  
CLR YELLOW2B  
CLR YELLOW3B  
CLR YELLOW4B  
RET

RESET\_TRAFFIC1:

MOV COUNT\_TIMER,#100  
MOV TIME\_YELLOW,#1  
CLR KUNING1  
MOV CONTROL1,#1  
MOV RELOAD1,TIME\_RED1

RET

RESET\_TRAFFIC2:

```
MOV COUNT_TIMER,#199
MOV TIME_YELLOW2,#1
CLR KUNING1B
MOV CONTROL2,#1
MOV RELOAD2,TIME_RED1B
RET
```

CEK\_TIMER:

```
CLR TR0
PUSH ACC
MOV TH0,#HIGH TIMER
MOV TL0,#LOW TIMER
DJNZ COUNT_TIMER,OUTJMP
MOV COUNT_TIMER,#199
JB YELLOW_ON1,NORM1 ;JIKA 0 KUNING AJA
CPL TOGGLE1 ;MATI HIDUP BERGANTIAN
CALL TOGGLE_Y1
JMP CHANNEL2
```

NORM1:

```
MOV A,CONTROL1
CJNE A,#1,POS2
JB KUNING1,N1
```



```

CPL YELLOW1
DJNZ TIME_YELLOW,OUT1A
MOV TIME_YELLOW,#6
SETB KUNING1
N1: CALL GREEN1_NYALA
DJNZ RELOAD1,OUT1A
CLR KUNING2
SETB GREEN1
SETB YELLOW1
CLR RED1
MOV RELOAD1,TIME_RED
MOV CONTROL1,#2 ;SIAP UTK LAMPU SELANJUTNYA
OUT1A:
JMP CHANNEL2
POS2: CJNE A,#2,POS3
JB KUNING2,N2
CPL YELLOW2
DJNZ TIME_YELLOW,OUT1A
MOV TIME_YELLOW,#6
SETB KUNING2
N2: CALL GREEN2_NYALA
DJNZ RELOAD1,CHANNEL2
CLR KUNING3 ;FLAG UTK KUNING SELANJUNYA
SETB GREEN2
SETB YELLOW2

```

CLR RED2

MOV RELOAD1,TIME\_RED3

MOV CONTROL1,#3 ;SIAP UTK LAMPU SELANJUTNYA

JMP CHANNEL2

POS3: CJNE A,#3,POS4

JB KUNING3,N3

CPL YELLOW3

DJNZ TIME\_YELLOW,CHANNEL2

MOV TIME\_YELLOW,#6

SETB KUNING3

N3: CALL GREEN3\_NYALA

DJNZ RELOAD1,CHANNEL2

CLR KUNING4

SETB GREEN3

SETB YELLOW3

CLR RED3

MOV RELOAD1,TIME\_RED4

MOV CONTROL1,#4 ;SIAP UTK LAMPU SELANJUTNYA

JMP CHANNEL2

POS4: CJNE A,#4,CHANNEL2

JB KUNING4,N4

CPL YELLOW4

DJNZ TIME\_YELLOW,CHANNEL2

MOV TIME\_YELLOW,#6

SETB KUNING4

```
N4: CALL GREEN4_NYALA
      DJNZ RELOAD1,CHANNEL2
      CLR KUNING1
      SETB GREEN4
      SETB YELLOW4
      CLR RED4
      MOV RELOAD1,TIME_RED1
      MOV CONTROL1,#1 ;SIAP UTK LAMPU SELANJUTNYA
```

CHANNEL2:

```
JB YELLOW_ON2,NORM2 ;JIKA 0 KUNING AJA
CPL TOGGLE2 ;MATI HIDUP BERGANTIAN
CALL TOGGLE_Y2
JMP OUTTIM
```

NORM2:MOV A,CONTROL2

```
CJNE A,#1,POS2B
JB KUNING1B,N1B
CPL YELLOW1B
DJNZ TIME_YELLOW2,OUT1B
MOV TIME_YELLOW2,#6
SETB KUNING1B
```

N1B: CALL GREEN1B\_NYALA

```
DJNZ RELOAD2,OUT1B
SETB GREEN1B
CLR KUNING2B
```

```

SETB YELLOW1B
CLR RED1B
MOV RELOAD2,TIME_RED2B
MOV CONTROL2,#2 ;SIAP UTK LAMPU SELANJUTNYA
OUT1B:JMP OUTTIM
POS2B: CJNE A,#2,POS3B
JB KUNING2B,N2B
CPL YELLOW2B
DJNZ TIME_YELLOW2,OUTTIM
MOV TIME_YELLOW2,#6
SETB KUNING2B
N2B: CALL GREEN2B_NYALA
DJNZ RELOAD2,OUTTIM
CLR KUNING3B
SETB GREEN2B
SETB YELLOW2B
CLR RED2B
MOV RELOAD2,TIME_RED3B
MOV CONTROL2,#3 ;SIAP UTK LAMPU SELANJUTNYA
JMP OUTTIM
POS3B:
CJNE A,#3,POS4B
JB KUNING3B,N3B
CPL YELLOW3B
DJNZ TIME_YELLOW2,OUTTIM

```

```
MOV TIME_YELLOW2,#6
SETB KUNING3B
N3B: CALL GREEN3B_NYALA
DJNZ RELOAD2,OUTTIM
CLR KUNING4B
SETB GREEN3B
SETB YELLOW3B
CLR RED3B
MOV RELOAD2,TIME_RED4B
MOV CONTROL2,#4 ;SIAP UTK LAMPU SELANJUTNYA
JMP OUTTIM
```

POS4B:

```
CJNE A,#4,OUTTIM
JB KUNING4B,N4B
CPL YELLOW4B
DJNZ TIME_YELLOW2,OUTTIM
MOV TIME_YELLOW2,#6
SETB KUNING4B
N4B: CALL GREEN4B_NYALA
DJNZ RELOAD2,OUTTIM
CLR KUNING1B
SETB GREEN4B
SETB YELLOW4B
CLR RED4B
MOV RELOAD2,TIME_RED1B
```

MOV CONTROL2,#1

;SIAP UTK LAMPU SELANJUTNYA

OUTTIM:

POP ACC

SETB TR0

RETI

GREEN1\_NYALA:

MOV P0,#0FFH

SETB P2.0

SETB P2.1

SETB P2.2

SETB P2.3

CLR GREEN1

CLR RED2

CLR RED3

CLR RED4

RET

GREEN2\_NYALA:

MOV P0,#0FFH

SETB P2.0

SETB P2.1

SETB P2.2

SETB P2.3

```
CLR RED1
CLR GREEN2
CLR RED3
CLR RED4
RET
```

GREEN3\_NYALA:

```
MOV P0,#0FFH
SETB P2.0
SETB P2.1
SETB P2.2
SETB P2.3
CLR RED1
CLR RED2
CLR GREEN3
CLR RED4
RET
```

GREEN4\_NYALA:

```
MOV P0,#0FFH
SETB P2.0
SETB P2.1
SETB P2.2
SETB P2.3
CLR RED1
```

```
CLR RED2
CLR RED3
CLR GREEN4
RET
```

GREEN1B\_NYALA:

```
MOV P1,#0FFH
SETB P2.4
SETB P2.5
SETB P2.6
SETB P2.7
CLR GREEN1B
CLR RED2B
CLR RED3B
CLR RED4B
RET
```

GREEN2B\_NYALA:

```
MOV P1,#0FFH
SETB P2.4
SETB P2.5
SETB P2.6
SETB P2.7
CLR RED1B
CLR GREEN2B
```



CLR RED3B

CLR RED4B

RET

GREEN3B\_NYALA:

MOV P1,#0FFH

SETB P2.4

SETB P2.5

SETB P2.6

SETB P2.7

CLR RED1B

CLR RED2B

CLR GREEN3B

CLR RED4B

RET

GREEN4B\_NYALA:

MOV P1,#0FFH

SETB P2.4

SETB P2.5

SETB P2.6

SETB P2.7

CLR RED1B

CLR RED2B

CLR RED3B

CLR GREEN4B

RET

NYALA\_YELLOW1B:

CLR YELLOW1B

CALL DELAY1S

SETB YELLOW1B

CALL DELAY1S

CLR YELLOW1B

RET

NYALA\_YELLOW2B:

CLR YELLOW2B

CALL DELAY1S

SETB YELLOW2B

CALL DELAY1S

CLR YELLOW2B

RET

NYALA\_YELLOW3B:

CLR YELLOW3B

CALL DELAY1S

SETB YELLOW3B

CALL DELAY1S

CLR YELLOW3B

RET

NYALA\_YELLOW4B:

CLR YELLOW4B

CALL DELAY1S

SETB YELLOW4B

CALL DELAY1S

CLR YELLOW4B

RET

DELAY1S:

MOV R5,#22

LD: CALL DELAY

DJNZ R5,LD

RET

DELAY:MOVR7,#150

DELAY1:

MOV R6,#50H

DJNZ R6,\$

DJNZ R7,DELAY1

RET

TX\_SEND:

NOP

```

NOP
MOV SBUF,A
JNB TI,$
CLR TI
NOP
RET

```

```

;-----
;   TRANSFER DATA SERIAL
;-----

```

```

SEND:SETB ENABLE_RX
      CLR ES
      MOV B,A
      MOV TEMP,#HEADER
      MOV TEMP+1,#SLAVE_NUMBER
      MOV TEMP+2,A
      MOV TEMP+3,#END_COMMAND
      NOP
      NOP
      NOP
      MOV R0,#TEMP
LOOP:MOV A,@R0
      CALL TX_SEND
      INC R0
      CJNE R0,#TEMP+4,LOOP

```

CLR ENABLE\_RX

SETB ES

RET

-----

; ROUTINE interupsi serial

-----

CEK\_SERIAL:

CLR ES

PUSH ACC

JNB RI,\$

MOV A,SBUF

CLR RI

JB SET\_HEADER,CEK

JB SELECT,CEKNUMBER

CJNE A,#END\_COMMAND,ISIN

SETB SET\_HEADER

SETB SELECT

JB UPDATE,NONE

CALL CEK\_OUT

NONE: SETB TR0

JMP OUTS

ISIN: CJNE A,#YELLOW\_LIGHT\_ON1,ISIN2

CLR YELLOW\_ON1 ;TRAFFIC1 MODE YELLOW

```

;CALL TX_SEND
CALL SIMPAN_ROM_STATUS
CALL RESET_TRAFFIC1
JMP OUTS
ISIN2: CJNE A,#YELLOW_LIGHT_ON2,ISIN3
CLR YELLOW_ON2 ;TRAFFIC2 MODE YELLOW
;CALL TX_SEND
CALL SIMPAN_ROM_STATUS
CALL RESET_TRAFFIC2
JMP OUTS
ISIN3: CJNE A,#YELLOW_LIGHT_OFF1,ISIN4
SETB YELLOW_ON1 ;TRAFFIC1 MODE NORMAL
;CALL TX_SEND
CALL SIMPAN_ROM_STATUS
CALL RESET_TRAFFIC1
JMP OUTS
ISIN4: CJNE A,#YELLOW_LIGHT_OFF2,ISIN5
SETB YELLOW_ON2 ;TRAFFIC2 MODE NORMAL
;CALL TX_SEND
CALL SIMPAN_ROM_STATUS
CALL RESET_TRAFFIC2
JMP OUTS
ISIN5: CJNE A,#STATUS_DELAY,ISIN6
CLR UPLOAD
JMP OUTS

```

ISIN6: CJNE A,#SET\_DEFAULT,ISI

CALL SETING\_DEFAULT

CALL SIMPAN\_ROM\_SETING

JMP OUTS

ISI: MOV @R0,A

INC R0

CLR UPDATE ;STATUS UPDATE WAKTU TRAFFIC

JMP OUTS

CEKNUMBER:

CJNE A,#SLAVE\_NUMBER,OUTS

CLR TR0

SETB UPLOAD

SETB UPDATE

MOV R0,#TEMP

CLR SELECT ;FLAG DATA DI IJTNKAN MASUK

JMP OUTS

CEK: CJNE A,#HEADER,OUTS

CLR SET\_HEADER

OUTS:POP ACC

SETB ES

RETI

CEK\_OUT:

MOV R4,TEMP ; JUMLAH PAKET DATA

```

MOV R0,#TEMP+1                ; DATA PAKET KE 1
MOV R1,#TIME_RED1            ; POSISI MEMORY TIMER TRAFFIC
COPY:MOV A,@R0
MOV @R1,A
INC R0
INC R1
DJNZ R4,COPY                  ;ULANG HINGGA PAKET SELESAI
; CALL RESET_TRAFFIC1
; CALL RESET_TRAFFIC2
CALL SIMPAN_ROM_SETING
RET

```

SEND\_DATA\_SETING:

```

SETB ENABLE_RX                ;RS485 AKTIF
MOV B,#8                       ; JUMLAH PAKET DATA
MOV R1,#TIME_RED1            ; POSISI MEMORY TIMER TRAFFIC
COP: MOV A,@R1
CALL TX_SEND
INC R1
DJNZ B,COP                     ;ULANG HINGGA PAKET SELESAI
CLR ENABLE_RX                 ;RS485 RECEIVE
RET

```

SIMPAN\_ROM\_SETING:

```

MOV TEMP,#8

```



MOV DPTR,#BACKUP\_SETING

CALL INISIAL\_I2C

MOV R1,#TIME\_RED1 ; POSISI MEMORY TIMER TRAFFIC

COPYR:MOVA,@R1

CALL WRITE\_I2C

INC R1

DJNZ TEMP,COPYR ;ULANG HINGGA PAKET SELESAI

MOV A,#0FFH

CALL WRITE\_I2C

CALL STOP\_I2C

RET

BACA\_ROM\_SETING:

MOV R0,#TIME\_RED1

MOV DPTR,#BACKUP\_SETING

CALL BACA\_I2C

LAGI2: CALL READ

MOV @R0,A

INC R0

CJNE A,#0FFH,LAGIAN

CALL NACK

CALL STOP\_I2C

RET

LAGIAN:CALL ACK

JMP LAGI2

BACA\_ROM\_STATUS:

MOV DPTR,#STATUS\_SETING

CALL BACA\_I2C

CALL READ

MOV 20H,A

CALL NACK

CALL STOP\_I2C

CALL RESET\_TRAFFIC1

CALL RESET\_TRAFFIC2

RET

;------

;ROUTINE SIMPAN I2C SERIAL EEPROM

;------

SIMPAN\_ROM\_STATUS:

MOV DPTR,#STATUS\_SETING

CALL INISIAL\_I2C

MOV A,20H ;DATA SETING YELLOW

CALL WRITE\_I2C

CALL STOP\_I2C

RET

;------

;ROUTINE I2C SERIAL EEPROM

;-----

INISIAL\_I2C:

```
CALL START_I2C
MOV A,#10100000B
CALL WRITE_I2C
MOV A,DPH ;TULIS ADDRESS HIGH
CALL WRITE_I2C
MOV A,DPL ;TULIS ADDRESS LOW
CALL WRITE_I2C
RET
```

BACA\_I2C:

```
CALL START_I2C
MOV A,#10100000B ;TULIS POSISI ALAMAT/ADDRESS
CALL WRITE_I2C
MOV A,DPH ;TULIS ADDRESS HIGH
CALL WRITE_I2C
MOV A,DPL ;TULIS ADDRESS LOW
CALL WRITE_I2C
CALL START_I2C
MOV A,#10100001B ;PERINTAH READ
CALL WRITE_I2C
RET
```

START\_I2C:

SETB SDA

SETB SCL

CALL ROM\_DELAY

CLR SDA

NOP

NOP

NOP

CLR SCL

CALL ROM\_DELAY

RET

STOP\_I2C:

CLR SDA

SETB SCL

CALL ROM\_DELAY

SETB SDA

CALL ROM\_DELAY

CLR SCL

RET

WRITE\_I2C:

MOV BROM,A

MOV TULIS,#8

OUTC: RLC A

```
MOV SDA,C
CALL ROM_DELAY
SETB SCL
NOP
NOP
NOP
NOP
CLR SCL
DJNZ TULIS,OUTC
SETB SDA
NOP
SETB SCL
NOP
NOP
NOP
MOV C,SDA
CLR SCL
MOV A,BROM
RET
```

```
READ:    MOV TULIS,#8
```

```
SETB SDA
```

```
INBIT:CALL ROM_DELAY
```

```
SETB SCL
```

```
NOP
```

```

NOP
NOP
MOV C,SDA
RLC A
CLR SCL
DJNZ TULIS,INBIT
RET

```

```

ACK: CLR SDA
      CALL ROM_DELAY
      SETB SCL
      NOP
      CLR SCL
      RET

```

```

NACK: SETB SDA
      CALL ROM_DELAY
      SETB SCL
      NOP
      CLR SCL
      RET

```

```

ROM_DELAY:
      DJNZ R7,$
      RET

```

## Listing Program Slave 2

```
;-----  
; PIN I2C SEEPROM  
;-----  
  
SCL          BIT    P3.7  
SDA          BIT    P3.6  
  
;-----  
;PIN CONTROL MAX 485  
;-----  
ENABLE_RX    BIT    P3.2          ;SELECTOR RS 485  
  
SELECT       BIT    20H.0  
SET_HEADER   BIT    20H.1  
  
CEK_STATUS   BIT    20H.2  
YELLOW_ON1   BIT    20H.3  
YELLOW_ON2   BIT    20H.4  
TOGGLE1      BIT    20H.5  
TOGGLE2      BIT    20H.6  
  
KUNING1      BIT    20H.7  
KUNING2      BIT    21H.0  
KUNING3      BIT    21H.1
```

UPDATE BIT 21H.7

UPLOAD BIT 22H.0

RED1 BIT P0.0

YELLOW1 BIT P0.1

GREEN1 BIT P0.2

RED2 BIT P0.3

YELLOW2 BIT P0.4

GREEN2 BIT P0.5

RED3 BIT P0.6

YELLOW3 BIT P0.7

GREEN3 BIT P2.0

TIMER EQU -4609

HEADER EQU 0FAH ;CODE AWAL COMMAND

SLAVE\_NUMBER EQU 0F2H ; NO REFERENSI SLAVE RS 485

END\_COMMAND EQU 0FBH

YELLOW\_LIGHT\_ON1 EQU 0FCH

YELLOW\_LIGHT\_ON2 EQU 0FDH

YELLOW\_LIGHT\_OFF1 EQU 0FEH

YELLOW\_LIGHT\_OFF2 EQU 0FFH



STATUS\_DELAY EQU 0F9H  
SET\_DEFAULT EQU 0F8H

;INISIAL ADDRESS SEEROM

STATUS\_SETING EQU 0

BACKUP\_SETING EQU 1

DATA\_SERIAL EQU 30H

COUNT\_TIMER EQU 31H

PAKET EQU 32H

TIME\_RED1 EQU 33H

TIME\_RED2 EQU 34H

TIME\_RED3 EQU 35H

SETING EQU 3BH

CONTROL1 EQU 3CH

RELOAD1 EQU 3EH

TIME\_YELLOW EQU 40H

TULIS EQU 42H

BROM EQU 43H

TEMP EQU 44H

ORG 00H

LJMP START

ORG 0BH

LJMP CEK\_TIMER

ORG 23H

LJMP CEK\_SERIAL

ORG 100H

START:SETBEA

MOV TMOD,#21H

MOV TH1,#0FDH ;BAUD RATE 9600

SETB TR1

SETB ET0

MOV TH0,#HIGH TIMER

MOV TL0,#LOW TIMER

MOV COUNT\_TIMER,#200

MOV SCON,#50H ;SERIAL MODE 2

SETB ES

SETB SET\_HEADER

SETB SELECT

CLR ENABLE\_RX

DJNZ R7,\$

SETB TR0

CALL BACA\_ROM\_SETTING

CALL BACA\_ROM\_STATUS

MOV A,20H

ORL A,#11100111B

```

    CLR  KUNING1
WAIT: JB  UPLOAD,$
    DJNZ R7,$
    CLR  ES
    SETB ENABLE_RX                ;RS485 TX
    CALL TX_SEND
    MOV  A,#HEADER
    CALL TX_SEND
    MOV  A,#SLAVE_NUMBER        ;SLAVE 2
    CALL TX_SEND
    MOV  A,#3                    ;JUMLAH PAKET
    CALL TX_SEND
    CALL SEND_DATA_SETING
    MOV  A,#END_COMMAND
    CALL TX_SEND
    SETB UPLOAD
    CLR  ENABLE_RX
    SETB ES
    JMP  WAIT

```

SETTING\_DEFAULT:

```

    MOV  CONTROL1,#1
    MOV  TIME_RED1,#12
    MOV  TIME_RED2,#12

```

```

MOV 20H,A
CLR KUNING1
WAIT:JB  UPLOAD,$
      DJNZ R7,$
      CLR  ES
      SETB ENABLE_RX                ;RS485 TX
      CALL TX_SEND
      MOV  A,#HEADER
      CALL TX_SEND
      MOV  A,#SLAVE_NUMBER          ;SLAVE 2
      CALL TX_SEND
      MOV  A,#3                      ;JUMLAH PAKET
      CALL TX_SEND
      CALL SEND_DATA_SETING
      MOV  A,#END_COMMAND
      CALL TX_SEND
      SETB UPLOAD
      CLR  ENABLE_RX
      SETB ES
      JMP  WAIT

```

SETTING\_DEFAULT:

```

MOV CONTROL1,#1
MOV TIME_RED1,#12
MOV TIME_RED2,#12

```

```
MOV TIME_RED3,#12
MOV RELOAD1,TIME_RED1
CLR KUNING1
SETB YELLOW_ON1           ;0 KUNING SAJA, 1 NORMAL
MOV TIME_YELLOW,#1
RET
```

TOGGLE\_Y1:

```
SETB RED1
SETB RED2
SETB RED3
SETB GREEN1
SETB GREEN2
SETB GREEN3
JB TOGGLE1,ONY1
SETB YELLOW1
SETB YELLOW2
SETB YELLOW3
RET
```

ONY1:CLR YELLOW1

CLR YELLOW2

CLR YELLOW3

RET

RESET\_TRAFFIC1:

```
MOV COUNT_TIMER,#199
MOV TIME_YELLOW,#1
CLR KUNING1
MOV CONTROL1,#1
MOV RELOAD1,TIME_RED1
RET
```

CEK\_TIMER:

```
CLR TR0
PUSH ACC
MOV TH0,#HIGH TIMER
MOV TL0,#LOW TIMER
DJNZ COUNT_TIMER,OUTJ
MOV COUNT_TIMER,#199
JB YELLOW_ON1,NORM1 ;JIKA 0 KUNING AJA
CPL TOGGLE1 ;MATI HIDUP BERGANTIAN
CALL TOGGLE_Y1
```

OUTJ: JMP OUTTIM

NORM1:

```
MOV A,CONTROL1
CJNE A,#1,POS2
JB KUNING1,N1
CPL YELLOW1
DJNZ TIME_YELLOW,OUT1A
MOV TIME_YELLOW,#6
```

```

SETB KUNING1
N1: CALL GREEN1_NYALA
    DJNZ RELOAD1,OUT1A
    CLR KUNING2
    SETB GREEN1
    SETB YELLOW1
    CLR RED1
    MOV RELOAD1,TIME_RED2
    MOV CONTROL1,#2          ;SIAP UTK LAMPU SELANJUTNYA
OUT1A:
    JMP OUTTIM
POS2: CJNE A,#2,POS3
    JB KUNING2,N2
    CPL YELLOW2
    DJNZ TIME_YELLOW,OUT1A
    MOV TIME_YELLOW,#6
    SETB KUNING2
N2: CALL GREEN2_NYALA
    DJNZ RELOAD1,OUTTIM
    CLR KUNING3          ;FLAG UTK KUNING SELANJUNYA
    SETB GREEN2
    SETB YELLOW2
    CLR RED2
    MOV RELOAD1,TIME_RED3
    MOV CONTROL1,#3      ;SIAP UTK LAMPU SELANJUTNYA

```

```

    JMP   OUTTIM
POS3: CJNE A,#3,OUTTIM
      JB   KUNING3,N3
      CPL  YELLOW3
      DJNZ TIME_YELLOW,OUTTIM
      MOV  TIME_YELLOW,#6
      SETB KUNING3
N3:   CALL GREEN3_NYALA
      DJNZ RELOAD1,OUTTIM
      CLR  KUNING1
      SETB GREEN3
      SETB YELLOW3
      CLR  RED3
      MOV  RELOAD1,TIME_RED1
      MOV  CONTROL1,#1      ;SIAP UTK LAMPU SELANJUTNYA
OUTTIM:
      POP  ACC
      SETB TR0
      RETI

```

```

GREEN1_NYALA:

```

```

    MOV  P0,#0FFH
    SETB P2.0
    SETB P2.1

```



```
SETB P2.2
SETB P2.3
CLR GREEN1
CLR RED2
CLR RED3
RET
```

GREEN2\_NYALA:

```
MOV P0,#0FFH
SETB P2.0
SETB P2.1
SETB P2.2
SETB P2.3
CLR RED1
CLR GREEN2
CLR RED3
RET
```

GREEN3\_NYALA:

```
MOV P0,#0FFH
SETB P2.0
SETB P2.1
SETB P2.2
SETB P2.3
CLR RED1
```

```
CLR RED2
CLR GREEN3
RET
```

DELAY1S:

```
MOV R5,#22
LD: CALL DELAY
DJNZ R5,LD
RET
```

DELAY:MOVR7,#150

DELAY1:

```
MOV R6,#50H
DJNZ R6,$
DJNZ R7,DELAY1
RET
```

TX\_SEND:

```
NOP
NOP
MOV SBUF,A
JNB TI,$
CLR TI
NOP
```

```
    NOP
    CLR  ENABLE_RX
    RET
```

```
;-----
;    TRANSFER DATA SERIAL
;-----
```

```
SEND:SETB ENABLE_RX
      CLR  ES
      MOV  B,A
      MOV  TEMP,#HEADER
      MOV  TEMP+1,#SLAVE_NUMBER
      MOV  TEMP+2,A
      MOV  TEMP+3,#END_COMMAND
      NOP
      NOP
      NOP
      MOV  R0,#TEMP
LOOP:MOV  A,@R0
      CALL TX_SEND
      INC  R0
      CJNE R0,#TEMP+4,LOOP
      CLR  ENABLE_RX
      SETB ES
      RET
```

; ROUTINE interupsi serial

;-----

CEK\_SERIAL:

CLR ES

PUSH ACC

JNB RI,\$

MOV A,SBUF

CLR RI

JB SET\_HEADER,CEK

JB SELECT,CEKNUMBER

CJNE A,#END\_COMMAND,ISIN

SETB SET\_HEADER

SETB SELECT

JB UPDATE,NONE

CALL CEK\_OUT

NONE: SETB TR0

JMP OUTS

ISIN: CJNE A,#YELLOW\_LIGHT\_ON1,ISIN2

CLR YELLOW\_ON1 ;TRAFFIC1 MODE YELLOW

;CALL TX\_SEND

CALL SIMPAN\_ROM\_STATUS

CALL RESET\_TRAFFIC1

```

    JMP  OUTS
ISIN2: CJNE A,#YELLOW_LIGHT_OFF1,ISIN3
        SETB YELLOW_ON1           ;TRAFFIC1 MODE NORMAL
        ;CALL TX_SEND
        CALL SIMPAN_ROM_STATUS
        CALL RESET_TRAFFIC1
        JMP  OUTS
ISIN3: CJNE A,#STATUS_DELAY,ISIN4
        CLR  UPLOAD
        JMP  OUTS
ISIN4: CJNE A,#SET_DEFAULT,ISI
        CALL SETING_DEFAULT
        CALL SIMPAN_ROM_SETING
        JMP  OUTS
ISI:   MOV  @R0,A
        INC  R0
        CLR  UPDATE           ;STATUS UPDATE WAKTU TRAFFIC
        JMP  OUTS

```

CEKNUMBER:

```

    CJNE A,#SLAVE_NUMBER,OUTS
    CLR  TR0
    SETB UPLOAD
    SETB UPDATE
    MOV  R0,#TEMP

```

```

        CLR  SELECT          ;FLAG DATA DI IJINKAN MASUK
        JMP  OUTS
CEK:   CJNE  A,#HEADER,OUTS
        CLR  SET_HEADER
OUTS:  POP  ACC
        SETB ES
        RETI

CEK_OUT:
        MOV  R4,TEMP          ; JUMLAH PAKET DATA
        MOV  R0,#TEMP+1      ; DATA PAKET KE 1
        MOV  R1,#TIME_RED1   ; POSISI MEMORY TIMER
TRAFFIC
COPY:  MOV  A,@R0
        MOV  @R1,A
        INC  R0
        INC  R1
        DJNZ R4,COPY         ;ULANG HINGGA PAKET SELESAI
;;     CALL RESET_TRAFFIC1
        CALL SIMPAN_ROM_SETING
        RET

SEND_DATA_SETING:
        MOV  B,#3            ; JUMLAH PAKET DATA
        MOV  R1,#TIME_RED1   ; POSISI MEMORY TIMER
TRAFFIC

```

```
COP: MOV A,@R1
      CALL TX_SEND
      INC R1
      DJNZ B,COP          ;ULANG HINGGA PAKET SELESAI
      RET
```

```
SIMPAN_ROM_SETING:
```

```
      MOV TEMP,#3
      MOV DPTR,#BACKUP_SETING
      CALL INISIAL_I2C
      MOV R1,#TIME_RED1   ; POSISI MEMORY TIMER
TRAFFIC
```

```
COPYR:MOVA,@R1
```

```
      CALL WRITE_I2C
      INC R1
      DJNZ TEMP,COPYR     ;ULANG HINGGA PAKET
SELESAI
```

```
      MOV A,#0FFH
      CALL WRITE_I2C
      CALL STOP_I2C
      RET
```

```
BACA_ROM_SETING:
```

```
      MOV R0,#TIME_RED1
      MOV DPTR,#BACKUP_SETING
      CALL BACA_I2C
```

```
LAGI2:    CALL READ
          MOV  @R0,A
          INC  R0
          CJNE A,#0FFH,LAGIAN
          CALL NACK
          CALL STOP_I2C
          RET
```

```
LAGIAN:CALL  ACK
        JMP  LAGI2
```

```
BACA_ROM_STATUS:
          MOV  DPTR,#STATUS_SETING
          CALL BACA_I2C
          CALL READ
          MOV  20H,A
          CALL NACK
          CALL STOP_I2C
          CALL RESET_TRAFFIC1
          RET
```

-----

;ROUTINE SIMPAN I2C SERIAL EEPROM

-----

```
SIMPAN_ROM_STATUS:
```



```
MOV DPTR,#STATUS_SETING
CALL INISIAL_I2C
MOV A,20H           ;DATA SETING YELLOW
CALL WRITE_I2C
CALL STOP_I2C
RET
```

```
;-----
```

```
;ROUTINE I2C SERIAL EEPROM
```

```
;-----
```

```
INISIAL_I2C:
```

```
CALL START_I2C
MOV A,#10100000B
CALL WRITE_I2C
MOV A,DPH           ;TULIS ADDRESS HIGH
CALL WRITE_I2C
MOV A,DPL           ;TULIS ADDRESS LOW
CALL WRITE_I2C
RET
```

```
BACA_I2C:
```

```
CALL START_I2C
MOV A,#10100000B           ;TULIS POSISI
ALAMAT/ADDRESS
CALL WRITE_I2C
```

```
MOV A,DPH                ;TULIS ADDRESS HIGH
CALL WRITE_I2C
MOV A,DPL                ;TULIS ADDRESS LOW
CALL WRITE_I2C
CALL START_I2C
MOV A,#10100001B        ;PERINTAH READ
CALL WRITE_I2C
RET
```

START\_I2C:

```
SETB SDA
SETB SCL
CALL ROM_DELAY
CLR SDA
NOP
NOP
NOP
CLR SCL
CALL ROM_DELAY
RET
```

STOP\_I2C:

```
CLR SDA
SETB SCL
```

```
CALL ROM_DELAY
SETB SDA
CALL ROM_DELAY
CLR SCL
RET
```

WRITE\_I2C:

```
MOV BROM,A
MOV TULIS,#8
OUTC: RLC A
MOV SDA,C
CALL ROM_DELAY
SETB SCL
NOP
NOP
NOP
NOP
CLR SCL
DJNZ TULIS,OUTC
SETB SDA
NOP
SETB SCL
NOP
NOP
NOP
```

```
MOV C,SDA
CLR SCL
MOV A,BROM
RET
```

```
READ: MOV TULIS,#8
```

```
SETB SDA
```

```
INBIT:CALL ROM_DELAY
```

```
SETB SCL
```

```
NOP
```

```
NOP
```

```
NOP
```

```
MOV C,SDA
```

```
RLC A
```

```
CLR SCL
```

```
DJNZ TULIS,INBIT
```

```
RET
```

```
ACK: CLR SDA
```

```
CALL ROM_DELAY
```

```
SETB SCL
```

```
NOP
```

```
CLR SCL
```

```
RET
```

```
NACK:    SETB SDA
         CALL ROM_DELAY
         SETB SCL
         NOP
         CLR  SCL
         RET
```

```
ROM_DELAY:
```

```
    DJNZ R7,$
    RET
```

```
END
```

## **LAMPIRAN DATASHEET**



# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## General Description

MAX481, MAX483, MAX485, MAX487-MAX491, and MAX1487 are low-power transceivers for RS-485 and RS-422 communication. Each part contains one driver and one receiver. The MAX483, MAX487, MAX488, and MAX489 are reduced slew-rate drivers that minimize EMI and signal reflections caused by improperly terminated cables, allowing error-free data transmission up to 250kbps. The driver slew rates of the MAX481, MAX485, MAX490, MAX491, and MAX1487 are not limited, allowing them to transmit up to 2.5Mbps.

These transceivers draw between 120µA and 500µA of quiescent current when unloaded or fully loaded with disabled receivers. Additionally, the MAX481, MAX483, and MAX487 feature a low-current shutdown mode in which they consume only 0.1µA. All parts operate from a single 5V supply.

These transceivers are short-circuit current limited and are protected against excessive power dissipation by thermal shutdown circuitry that places the driver outputs into a high-impedance state. The receiver input has a fail-safe feature that guarantees a logic-high output if the input is open circuit.

The MAX487 and MAX1487 feature quarter-unit-load receiver input impedance, allowing up to 128 MAX487/MAX1487 transceivers on the bus. Full-duplex communications are obtained using the MAX488-MAX491, while the MAX481, MAX483, MAX485, MAX487, and MAX1487 are designed for half-duplex applications.

## Applications

- Low-Power RS-485 Transceivers
- Low-Power RS-422 Transceivers
- Level Translators
- Transceivers for EMI-Sensitive Applications
- Industrial-Control Local Area Networks

## Features

- ♦ In µMAX Package: Smallest 8-Pin SO
- ♦ Slew-Rate Limited for Error-Free Data Transmission (MAX483/487/488/489)
- ♦ 0.1µA Low-Current Shutdown Mode (MAX481/483/487)
- ♦ Low Quiescent Current:
  - 120µA (MAX483/487/488/489)
  - 230µA (MAX1487)
  - 300µA (MAX481/485/490/491)
- ♦ -7V to +12V Common-Mode Input Voltage Range
- ♦ Three-State Outputs
- ♦ 30ns Propagation Delays, 5ns Skew (MAX481/485/490/491/1487)
- ♦ Full-Duplex and Half-Duplex Versions Available
- ♦ Operate from a Single 5V Supply
- ♦ Allows up to 128 Transceivers on the Bus (MAX487/MAX1487)
- ♦ Current-Limiting and Thermal Shutdown for Driver Overload Protection

## Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX481CPA	0°C to +70°C	8 Plastic DIP
MAX481CSA	0°C to +70°C	8 SO
MAX481CUA	0°C to +70°C	8 µMAX
MAX481C/D	0°C to +70°C	Dice*

Ordering information continued at end of data sheet.  
 \* Contact factory for dice specifications.

## Selection Table

PART NUMBER	HALF/FULL DUPLEX	DATA RATE (Mbps)	SLEW-RATE LIMITED	LOW-POWER SHUTDOWN	RECEIVER/DRIVER ENABLE	QUIESCENT CURRENT (µA)	NUMBER OF TRANSMITTERS ON BUS	PIN COUNT
MAX481	Half	2.5	No	Yes	Yes	300	32	8
MAX483	Half	0.25	Yes	Yes	Yes	120	32	8
MAX485	Half	2.5	No	No	Yes	300	32	8
MAX487	Half	0.25	Yes	Yes	Yes	120	128	8
MAX488	Full	0.25	Yes	No	No	120	32	8
MAX489	Full	0.25	Yes	No	Yes	120	32	14
MAX490	Full	2.5	No	No	No	300	32	8
MAX491	Full	2.5	No	No	Yes	300	32	14
MAX1487	Half	2.5	No	No	Yes	230	128	8



Maxim Integrated Products 1

For free samples & the latest literature: <http://www.maxim-ic.com>, or phone 1-800-998-8800

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V <sub>CC</sub> ).....	12V	14-Pin SO (derate 8.33mW/°C above +70°C).....	667mW
Control Input Voltage ( $\overline{RE}$ , DE).....	-0.5V to (V <sub>CC</sub> + 0.5V)	8-Pin $\mu$ MAX (derate 4.1mW/°C above +70°C).....	830mW
Driver Input Voltage (DI).....	-0.5V to (V <sub>CC</sub> + 0.5V)	8-Pin CERDIP (derate 8.00mW/°C above +70°C).....	640mW
Driver Output Voltage (A, B).....	-8V to +12.5V	14-Pin CERDIP (derate 9.09mW/°C above +70°C).....	727mW
Receiver Input Voltage (A, B).....	-8V to +12.5V	Operating Temperature Ranges	
Receiver Output Voltage (RO).....	-0.5V to (V <sub>CC</sub> + 0.5V)	MAX4__C_/MAX1487C_A.....	0°C to +70°C
Continuous Power Dissipation (T <sub>A</sub> = +70°C)		MAX4__E_/MAX1487E_A.....	-40°C to +85°C
in Plastic DIP (derate 9.09mW/°C above +70°C).....	727mW	MAX4__MJ_/MAX1487MJA.....	-55°C to +125°C
in Plastic DIP (derate 10.00mW/°C above +70°C).....	800mW	Storage Temperature Range.....	
in SO (derate 5.88mW/°C above +70°C).....	471mW	-65°C to +160°C	
		Lead Temperature (soldering, 10sec).....	
		+300°C	

Values 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 in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## ELECTRICAL CHARACTERISTICS

V<sub>CC</sub> = 5V  $\pm$ 5%, T<sub>A</sub> = T<sub>MIN</sub> to T<sub>MAX</sub>, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Differential Driver Output (no load)	V <sub>OD1</sub>				5	V
Differential Driver Output (with load)	V <sub>OD2</sub>	R = 50 $\Omega$ (RS-422)	2			V
		R = 27 $\Omega$ (RS-485), Figure 4	1.5		5	V
Change in Magnitude of Driver Differential Output Voltage for Complementary Output States	$\Delta$ V <sub>OD</sub>	R = 27 $\Omega$ or 50 $\Omega$ , Figure 4			0.2	V
Average Common-Mode Output Voltage	V <sub>OC</sub>	R = 27 $\Omega$ or 50 $\Omega$ , Figure 4			3	V
Change in Magnitude of Driver Common-Mode Output Voltage Complementary Output States	$\Delta$ V <sub>OC</sub>	R = 27 $\Omega$ or 50 $\Omega$ , Figure 4			0.2	V
Input High Voltage	V <sub>IH</sub>	DE, DI, $\overline{RE}$	2.0			V
Input Low Voltage	V <sub>IL</sub>	DE, DI, $\overline{RE}$			0.8	V
Input Current	I <sub>IN1</sub>	DE, DI, $\overline{RE}$			$\pm$ 2	$\mu$ A
Input Current (A, B)	I <sub>IN2</sub>	DE = 0V; V <sub>CC</sub> = 0V or 5.25V, all devices except MAX487/MAX1487	V <sub>IN</sub> = 12V		1.0	mA
			V <sub>IN</sub> = -7V		-0.8	
		MAX487/MAX1487, DE = 0V, V <sub>CC</sub> = 0V or 5.25V	V <sub>IN</sub> = 12V		0.25	mA
			V <sub>IN</sub> = -7V		-0.2	
Receiver Differential Threshold Voltage	V <sub>TH</sub>	-7V $\leq$ V <sub>CM</sub> $\leq$ 12V	-0.2		0.2	V
Receiver Input Hysteresis	$\Delta$ V <sub>TH</sub>	V <sub>CM</sub> = 0V		70		mV
Receiver Output High Voltage	V <sub>OH</sub>	I <sub>O</sub> = -4mA, V <sub>ID</sub> = 200mV	3.5			V
Receiver Output Low Voltage	V <sub>OL</sub>	I <sub>O</sub> = 4mA, V <sub>ID</sub> = -200mV			0.4	V
Receiver Standby Current (high impedance) Output Current at Receiver	I <sub>OZR</sub>	0.4V $\leq$ V <sub>O</sub> $\leq$ 2.4V			$\pm$ 1	$\mu$ A
Receiver Input Resistance	R <sub>IN</sub>	-7V $\leq$ V <sub>CM</sub> $\leq$ 12V, all devices except MAX487/MAX1487	12			k $\Omega$
		-7V $\leq$ V <sub>CM</sub> $\leq$ 12V, MAX487/MAX1487	48			k $\Omega$



# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

## ELECTRICAL CHARACTERISTICS (continued)

( $V_{CC} = 5V \pm 5\%$ ,  $T_A = T_{MIN}$  to  $T_{MAX}$ , unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Load Supply Current (Note 3)	I <sub>CC</sub>	MAX488/MAX489, DE, DI, RE = 0V or V <sub>CC</sub>		120	250	μA
		MAX490/MAX491, DE, DI, RE = 0V or V <sub>CC</sub>		300	500	
		MAX481/MAX485, RE = 0V or V <sub>CC</sub>	DE = V <sub>CC</sub>	500	900	
			DE = 0V	300	500	
		MAX1487, RE = 0V or V <sub>CC</sub>	DE = V <sub>CC</sub>	300	500	
			DE = 0V	230	400	
		MAX483/MAX487, RE = 0V or V <sub>CC</sub>	DE = 5V	MAX483 MAX487	350 250	
DE = 0V			120	250		
Supply Current in Shutdown	ISHDN	MAX481/483/487, DE = 0V, RE = V <sub>CC</sub>		0.1	10	μA
Receiver Short-Circuit Current, = High	I <sub>OSD1</sub>	-7V ≤ V <sub>O</sub> ≤ 12V (Note 4)	35		250	mA
Receiver Short-Circuit Current, = Low	I <sub>OSD2</sub>	-7V ≤ V <sub>O</sub> ≤ 12V (Note 4)	35		250	mA
Receiver Short-Circuit Current	I <sub>OSR</sub>	0V ≤ V <sub>O</sub> ≤ V <sub>CC</sub>	7		95	mA

## SWITCHING CHARACTERISTICS—MAX481/MAX485, MAX490/MAX491, MAX1487

( $V_{CC} = 5V \pm 5\%$ ,  $T_A = T_{MIN}$  to  $T_{MAX}$ , unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS	
Receiver Input to Output	t <sub>PLH</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	10	30	60	ns	
	t <sub>PHL</sub>		10	30	60		
Receiver Output Skew to Output	t <sub>SKEW</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF		5	10	ns	
Receiver Rise or Fall Time	t <sub>R</sub> , t <sub>F</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	MAX481, MAX485, MAX1487	3	15	40	ns
			MAX490C/E, MAX491C/E	5	15	25	
			MAX490M, MAX491M	3	15	40	
Receiver Enable to Output High	t <sub>ZH</sub>	Figures 7 and 9, C <sub>L</sub> = 100pF, S <sub>2</sub> closed		40	70	ns	
Receiver Enable to Output Low	t <sub>ZL</sub>	Figures 7 and 9, C <sub>L</sub> = 100pF, S <sub>1</sub> closed		40	70	ns	
Receiver Disable Time from Low	t <sub>LZ</sub>	Figures 7 and 9, C <sub>L</sub> = 15pF, S <sub>1</sub> closed		40	70	ns	
Receiver Disable Time from High	t <sub>HZ</sub>	Figures 7 and 9, C <sub>L</sub> = 15pF, S <sub>2</sub> closed		40	70	ns	
Receiver Input to Output	t <sub>PLH</sub> , t <sub>PHL</sub>	Figures 6 and 10, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	MAX481, MAX485, MAX1487	20	90	200	ns
			MAX490C/E, MAX491C/E	20	90	150	
			MAX490M, MAX491M	20	90	200	
Receiver Input to Output Skew	t <sub>SKD</sub>	Figures 6 and 10, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF		13		ns	
Receiver Enable to Output Low	t <sub>ZL</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S <sub>1</sub> closed		20	50	ns	
Receiver Enable to Output High	t <sub>ZH</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S <sub>2</sub> closed		20	50	ns	
Receiver Disable Time from Low	t <sub>LZ</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S <sub>1</sub> closed		20	50	ns	
Receiver Disable Time from High	t <sub>HZ</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S <sub>2</sub> closed		20	50	ns	
Maximum Data Rate	f <sub>MAX</sub>		2.5			Mbps	
Receiver Enable to Shutdown	t <sub>SHDN</sub>	MAX481 (Note 5)	50	200	600	ns	

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## SWITCHING CHARACTERISTICS—MAX481/MAX485, MAX490/MAX491, MAX1487 (continued)

V<sub>CC</sub> = 5V ±5%, T<sub>A</sub> = T<sub>MIN</sub> to T<sub>MAX</sub>, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Receiver Enable from Shutdown to Input High (MAX481)	t <sub>ZH</sub> (SHDN)	Figures 7 and 9, C <sub>L</sub> = 100pF, S2 closed		40	100	ns
Receiver Enable from Shutdown to Input Low (MAX481)	t <sub>ZL</sub> (SHDN)	Figures 7 and 9, C <sub>L</sub> = 100pF, S1 closed		40	100	ns
Receiver Enable from Shutdown Output High (MAX481)	t <sub>ZH</sub> (SHDN)	Figures 5 and 11, C <sub>L</sub> = 15pF, S2 closed, A - B = 2V		300	1000	ns
Receiver Enable from Shutdown Output Low (MAX481)	t <sub>ZL</sub> (SHDN)	Figures 5 and 11, C <sub>L</sub> = 15pF, S1 closed, B - A = 2V		300	1000	ns

## SWITCHING CHARACTERISTICS—MAX483, MAX487/MAX488/MAX489

V<sub>CC</sub> = 5V ±5%, T<sub>A</sub> = T<sub>MIN</sub> to T<sub>MAX</sub>, unless otherwise noted.) (Notes 1, 2)

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Receiver Input to Output	t <sub>PLH</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	250	800	2000	ns
	t <sub>PHL</sub>		250	800	2000	
Receiver Output Skew to Output	t <sub>SKEW</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF		100	800	ns
Receiver Rise or Fall Time	t <sub>R</sub> , t <sub>F</sub>	Figures 6 and 8, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	250		2000	ns
Receiver Enable to Output High	t <sub>ZH</sub>	Figures 7 and 9, C <sub>L</sub> = 100pF, S2 closed	250		2000	ns
Receiver Enable to Output Low	t <sub>ZL</sub>	Figures 7 and 9, C <sub>L</sub> = 100pF, S1 closed	250		2000	ns
Receiver Disable Time from Low	t <sub>LZ</sub>	Figures 7 and 9, C <sub>L</sub> = 15pF, S1 closed	300		3000	ns
Receiver Disable Time from High	t <sub>HZ</sub>	Figures 7 and 9, C <sub>L</sub> = 15pF, S2 closed	300		3000	ns
Receiver Input to Output	t <sub>PLH</sub>	Figures 6 and 10, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF	250		2000	ns
	t <sub>PHL</sub>		250		2000	
Receiver Input-to-Output Differential Receiver Skew	t <sub>SKD</sub>	Figures 6 and 10, R <sub>DIFF</sub> = 54Ω, C <sub>L1</sub> = C <sub>L2</sub> = 100pF		100		ns
Receiver Enable to Output Low	t <sub>ZL</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S1 closed		20	50	ns
Receiver Enable to Output High	t <sub>ZH</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S2 closed		20	50	ns
Receiver Disable Time from Low	t <sub>LZ</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S1 closed		20	50	ns
Receiver Disable Time from High	t <sub>HZ</sub>	Figures 5 and 11, C <sub>R</sub> L = 15pF, S2 closed		20	50	ns
Maximum Data Rate	f <sub>MAX</sub>	t <sub>PLH</sub> , t <sub>PHL</sub> < 50% of data period	250			kbps
Time to Shutdown	t <sub>SHDN</sub>	MAX483/MAX487 (Note 5)	50	200	600	ns
Receiver Enable from Shutdown to Input High	t <sub>ZH</sub> (SHDN)	MAX483/MAX487, Figures 7 and 9, C <sub>L</sub> = 100pF, S2 closed			2000	ns
Receiver Enable from Shutdown to Input Low	t <sub>ZL</sub> (SHDN)	MAX483/MAX487, Figures 7 and 9, C <sub>L</sub> = 100pF, S1 closed			2000	ns
Receiver Enable from Shutdown Output High	t <sub>ZH</sub> (SHDN)	MAX483/MAX487, Figures 5 and 11, C <sub>L</sub> = 15pF, S2 closed			2500	ns
Receiver Enable from Shutdown Output Low	t <sub>ZL</sub> (SHDN)	MAX483/MAX487, Figures 5 and 11, C <sub>L</sub> = 15pF, S1 closed			2500	ns

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

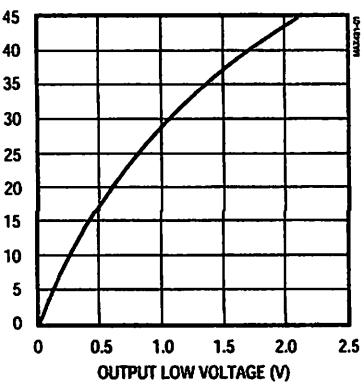
## TESTS FOR ELECTRICAL/SWITCHING CHARACTERISTICS

- 1: All currents into device pins are positive; all currents out of device pins are negative. All voltages are referenced to device ground unless otherwise specified.
- 2: All typical specifications are given for  $V_{CC} = 5V$  and  $T_A = +25^\circ C$ .
- 3: Supply current specification is valid for loaded transmitters when  $DE = 0V$ .
- 4: Applies to peak current. See *Typical Operating Characteristics*.
- 5: The MAX481/MAX483/MAX487 are put into shutdown by bringing  $\overline{RE}$  high and  $DE$  low. If the inputs are in this state for less than 50ns, the parts are guaranteed not to enter shutdown. If the inputs are in this state for at least 600ns, the parts are guaranteed to have entered shutdown. See *Low-Power Shutdown Mode* section.

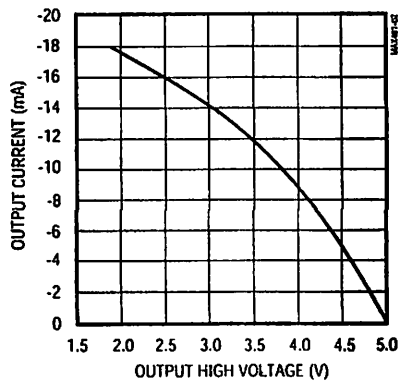
## Typical Operating Characteristics

( $V_{CC} = 5V$ ,  $T_A = +25^\circ C$ , unless otherwise noted.)

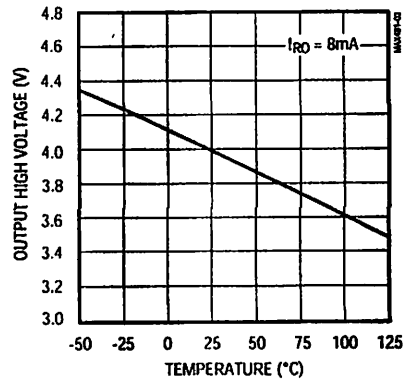
OUTPUT CURRENT vs. RECEIVER OUTPUT LOW VOLTAGE



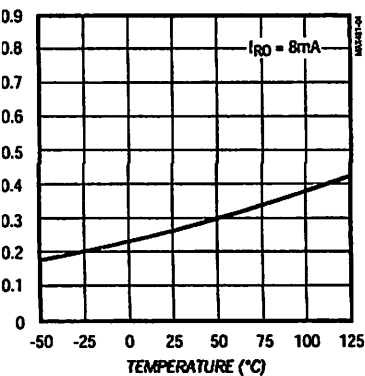
OUTPUT CURRENT vs. RECEIVER OUTPUT HIGH VOLTAGE



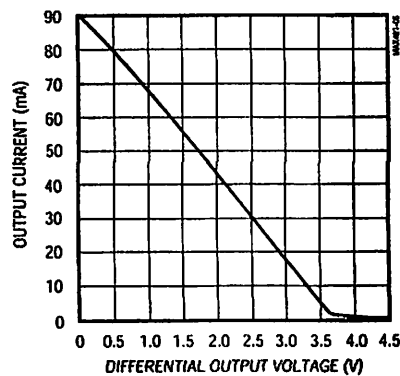
RECEIVER OUTPUT HIGH VOLTAGE vs. TEMPERATURE



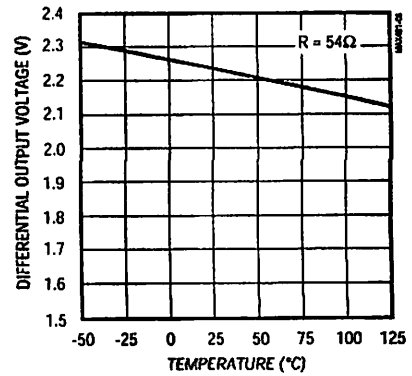
RECEIVER OUTPUT LOW VOLTAGE vs. TEMPERATURE



DRIVER OUTPUT CURRENT vs. DIFFERENTIAL OUTPUT VOLTAGE



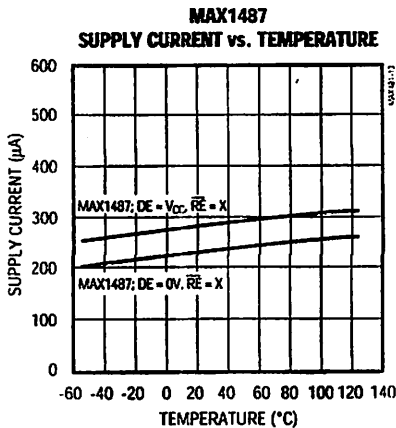
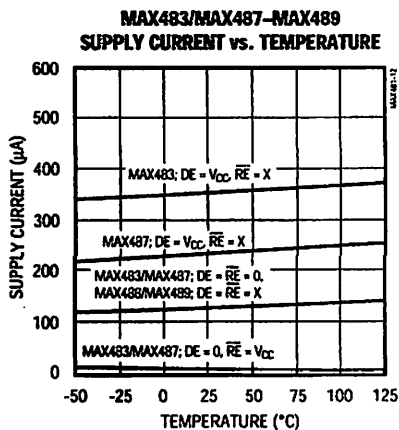
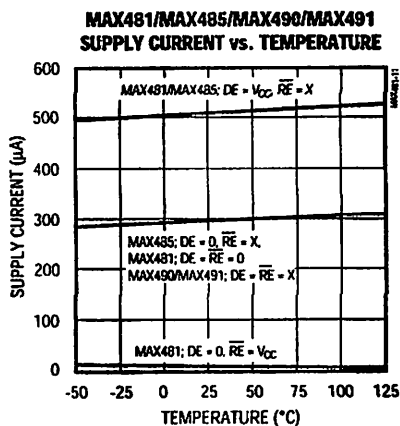
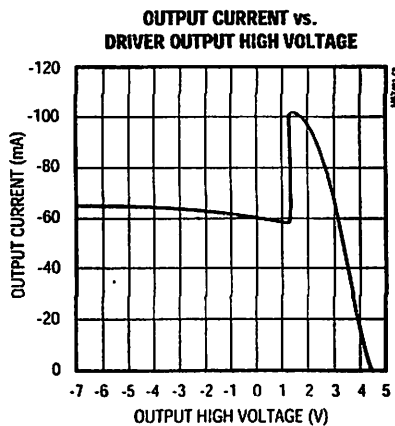
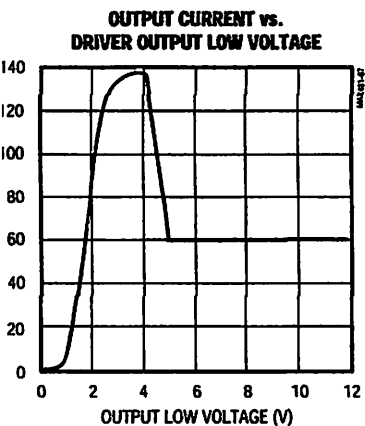
DRIVER DIFFERENTIAL OUTPUT VOLTAGE vs. TEMPERATURE



# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Typical Operating Characteristics (continued)

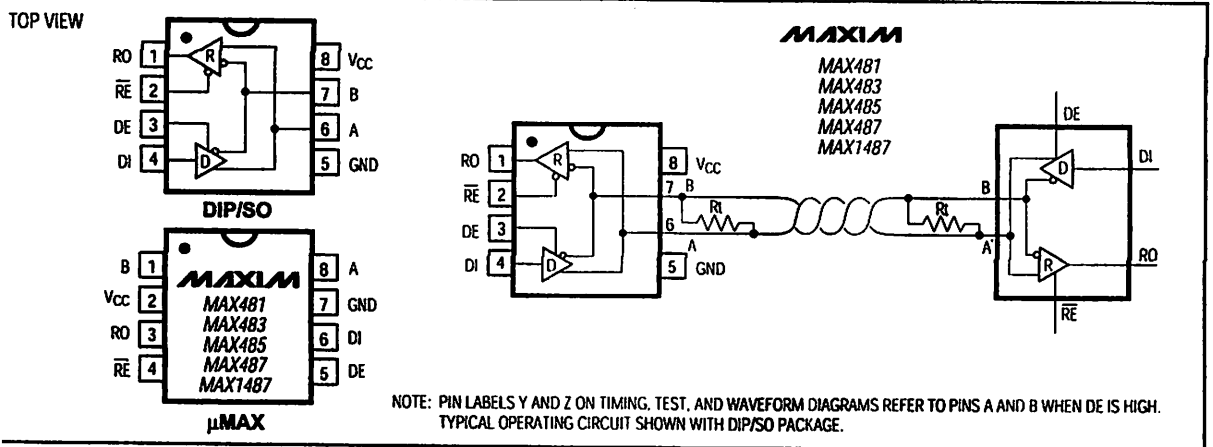
$V_{CC} = 5V$ ,  $T_A = +25^\circ C$ , unless otherwise noted.)



# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Pin Description

PIN					NAME	FUNCTION
X481/MAX483/ X485/MAX487/ MAX1487	μMAX	MAX488/ MAX490		MAX489/ MAX491		
/SO	μMAX	DIP/SO	μMAX	DIP/SO		
1	3	2	4	2	RO	Receiver Output: If A > B by 200mV, RO will be high; if A < B by 200mV, RO will be low.
2	4	—	—	3	$\overline{RE}$	Receiver Output Enable. RO is enabled when $\overline{RE}$ is low; RO is high impedance when $\overline{RE}$ is high.
3	5	—	—	4	DE	Driver Output Enable. The driver outputs, Y and Z, are enabled by bringing DE high. They are high impedance when DE is low. If the driver outputs are enabled, the parts function as line drivers. While they are high impedance, they function as line receivers if $\overline{RE}$ is low.
4	6	3	5	5	DI	Driver Input. A low on DI forces output Y low and output Z high. Similarly, a high on DI forces output Y high and output Z low.
5	7	4	6	6, 7	GND	Ground
—	—	5	7	9	Y	Noninverting Driver Output
—	—	6	8	10	Z	Inverting Driver Output
5	8	—	—	—	A	Noninverting Receiver Input and Noninverting Driver Output
—	—	8	2	12	A	Noninverting Receiver Input
7	1	—	—	—	B	Inverting Receiver Input and Inverting Driver Output
—	—	7	1	11	B	Inverting Receiver Input
1	2	1	3	14	VCC	Positive Supply: $4.75V \leq V_{CC} \leq 5.25V$
—	—	—	—	1, 8, 13	N.C.	No Connect—not internally connected



1. MAX481/MAX483/MAX485/MAX487/MAX1487 Pin Configuration and Typical Operating Circuit

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

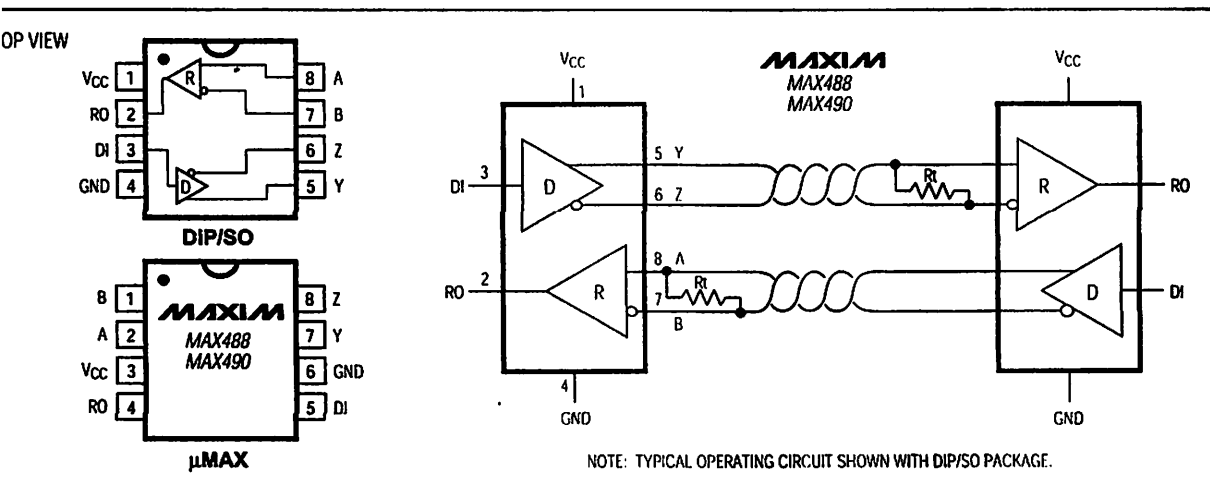


Figure 2. MAX488/MAX490 Pin Configuration and Typical Operating Circuit

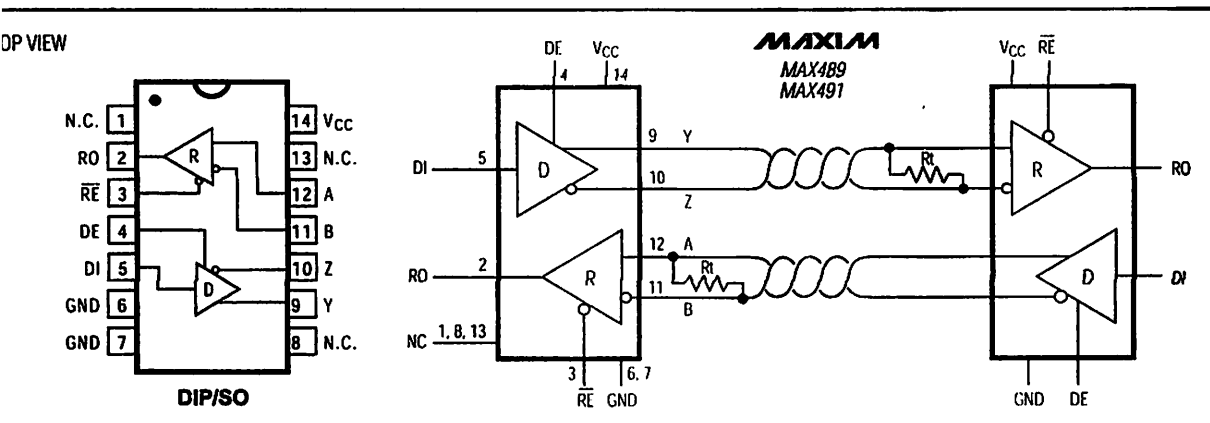


Figure 3. MAX489/MAX491 Pin Configuration and Typical Operating Circuit

## Applications Information

MAX481/MAX483/MAX485/MAX487-MAX491 and MAX1487 are low-power transceivers for RS-485 and RS-422 communications. The MAX481, MAX485, MAX490, MAX491, and MAX1487 can transmit and receive at data rates up to 2.5Mbps, while the MAX483, MAX487, MAX488, and MAX489 are specified for data rates up to 100kbps. The MAX488-MAX491 are full-duplex transceivers while the MAX481, MAX483, MAX485, MAX487, MAX488, MAX489, and MAX1487 are half-duplex. In addition, Driver Enable (DE) and Receiver Enable (RE) pins are included on the MAX481, MAX483, MAX485, MAX487, MAX489, MAX491, and MAX1487. When disabled, the driver and receiver outputs are high impedance.

## MAX487/MAX1487: 128 Transceivers on the Bus

The 48kΩ, 1/4-unit-load receiver input impedance of the MAX487 and MAX1487 allows up to 128 transceivers on a bus, compared to the 1-unit load (12kΩ input impedance) of standard RS-485 drivers (32 transceivers maximum). Any combination of MAX487/MAX1487 and other RS-485 transceivers with a total of 32 unit loads or less can be put on the bus. The MAX481/MAX483/MAX485 and MAX488-MAX491 have standard 12kΩ Receiver Input Impedance.

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Test Circuits

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

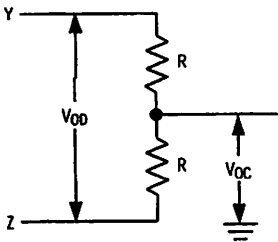


Figure 4. Driver DC Test Load

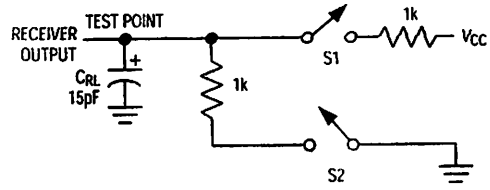


Figure 5. Receiver Timing Test Load

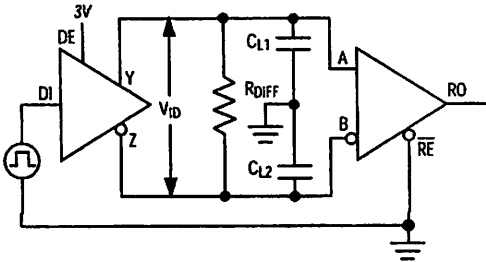


Figure 6. Driver/Receiver Timing Test Circuit

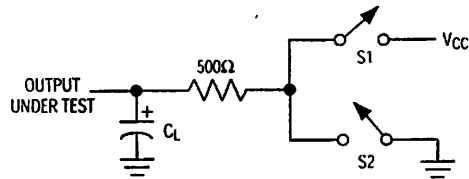


Figure 7. Driver Timing Test Load

### MAX483/MAX487/MAX488/MAX489: Reduced EMI and Reflections

MAX483 and MAX487-MAX489 are slew-rate limiting devices that minimize EMI and reduce reflections caused by improperly terminated cables. Figure 12 shows the driver output waveform and its Fourier analysis of a 10kHz signal transmitted by a MAX481, MAX485, MAX488, MAX489, MAX491, or MAX1487. High-frequency har-

monics with large amplitudes are evident. Figure 13 shows the same information displayed for a MAX483, MAX487, MAX488, or MAX489 transmitting under the same conditions. Figure 13's high-frequency harmonics have much lower amplitudes, and the potential for EMI is significantly reduced.

# Low-Power, Slew-Rate-Limited S-485/RS-422 Transceivers

## Switching Waveforms

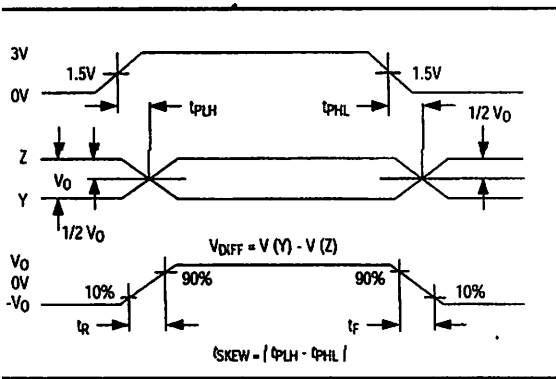


Figure 8. Driver Propagation Delays

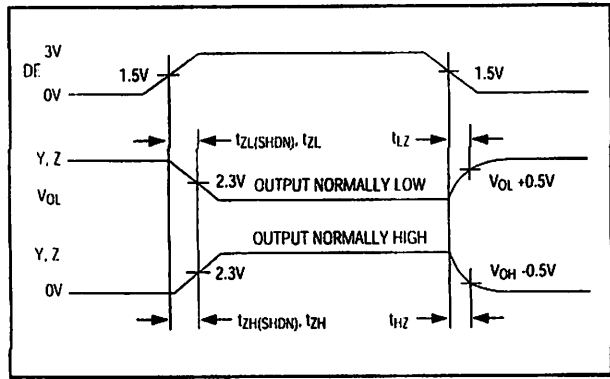


Figure 9. Driver Enable and Disable Times (except MAX488 and MAX490)

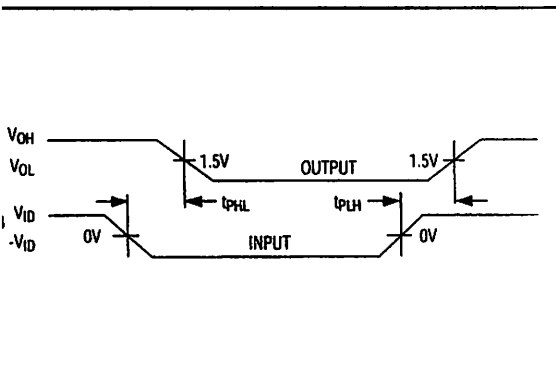


Figure 10. Receiver Propagation Delays

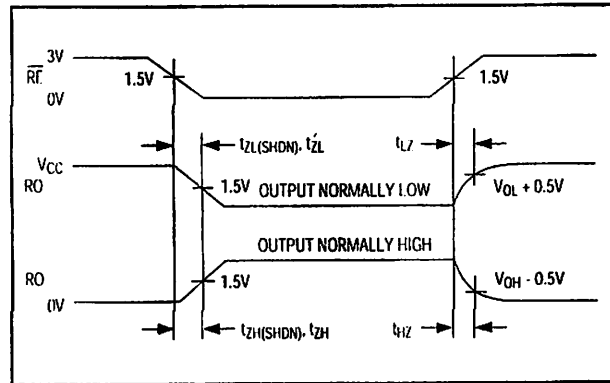


Figure 11. Receiver Enable and Disable Times (except MAX488 and MAX490)

## Function Tables (MAX481/MAX483/MAX485/MAX487/MAX1487)

Table 1. Transmitting

INPUTS			OUTPUTS	
RE	DE	DI	Z	Y
X	1	1	0	1
X	1	0	1	0
0	0	X	High-Z	High-Z
1	0	X	High-Z*	High-Z*

Don't care  
 High-Z = High impedance  
 \* Shutdown mode for MAX481/MAX483/MAX487

Table 2. Receiving

INPUTS			OUTPUT
RE	DE	A-B	RO
0	0	$\geq +0.2V$	1
0	0	$\leq -0.2V$	0
0	0	Inputs open	1
1	0	X	High-Z*

X = Don't care  
 High-Z = High impedance  
 \* Shutdown mode for MAX481/MAX483/MAX487



## Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

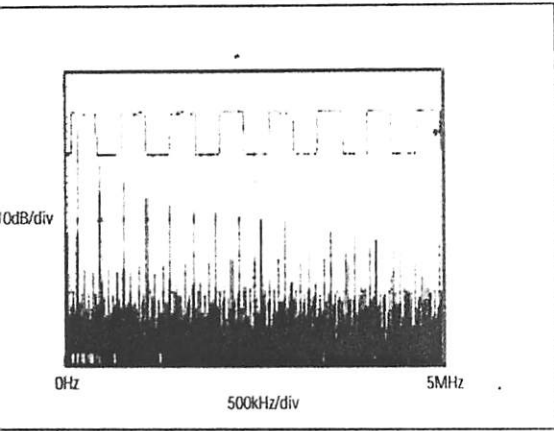


Figure 12. Driver Output Waveform and FFT Plot of MAX481/485/MAX490/MAX491/MAX1487 Transmitting a 150kHz Signal

### Low-Power Shutdown Mode (MAX481/MAX483/MAX487)

Low-power shutdown mode is initiated by bringing  $\overline{RE}$  high and DE low. The devices will not shut down unless both the driver and receiver are disabled. In shutdown, the devices typically draw only 0.1 $\mu$ A of supply current.

$\overline{RE}$  and DE may be driven simultaneously; the parts are guaranteed not to enter shutdown if  $\overline{RE}$  is high and DE low for less than 50ns. If the inputs are in this state for at least 600ns, the parts are guaranteed to enter shutdown.

For the MAX481, MAX483, and MAX487, the  $t_{ZH}$  and  $t_{ZL}$  enable times assume the part was not in the low-power shutdown state (the MAX485/MAX488–MAX491 and MAX1487 can not be shut down). The  $t_{ZH}(SHDN)$  and  $t_{ZL}(SHDN)$  enable times assume the parts were shut down (see *Electrical Characteristics*).

It takes the drivers and receivers longer to become enabled from the low-power shutdown state ( $t_{ZH}(SHDN)$ ,  $t_{ZL}(SHDN)$ ) than from the operating mode ( $t_{ZH}$ ,  $t_{ZL}$ ). (The parts are in operating mode if the  $\overline{RE}$ ,  $\overline{RD}$  inputs equal a logical 0,1 or 1,1 or 0, 0.)

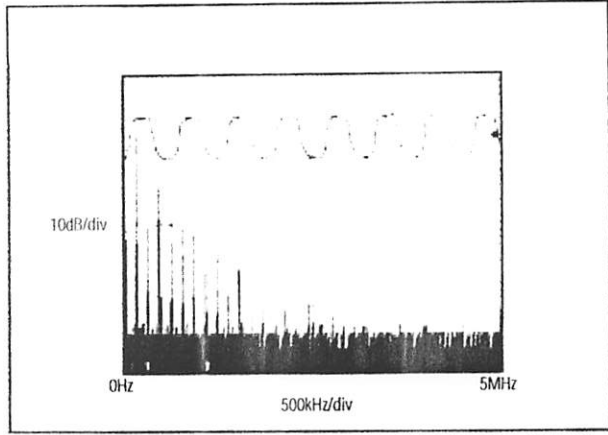


Figure 13. Driver Output Waveform and FFT Plot of MAX483/485/MAX487–MAX489 Transmitting a 150kHz Signal

### Driver Output Protection

Excessive output current and power dissipation caused by faults or by bus contention are prevented by two mechanisms. A foldback current limit on the output stage provides immediate protection against short circuits over the whole common-mode voltage range (see *Typical Operating Characteristics*). In addition, a thermal shutdown circuit forces the driver outputs into a high-impedance state if the die temperature rises excessively.

### Propagation Delay

Many digital encoding schemes depend on the difference between the driver and receiver propagation delay times. Typical propagation delays are shown in Figures 15–18 using Figure 14's test circuit.

The difference in receiver delay times,  $|t_{PLH} - t_{PHL}|$ , is typically under 13ns for the MAX481, MAX485, MAX490, MAX491, and MAX1487 and is typically less than 100ns for the MAX483 and MAX487–MAX489.

The driver skew times are typically 5ns (10ns max) for the MAX481, MAX485, MAX490, MAX491, and MAX1487, and are typically 100ns (800ns max) for the MAX483 and MAX487–MAX489.

MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

# Low-Power, Slew-Rate-Limited S-485/RS-422 Transceivers

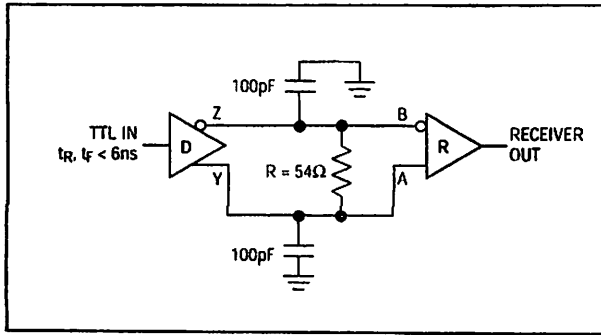


Figure 14. Receiver Propagation Delay Test Circuit

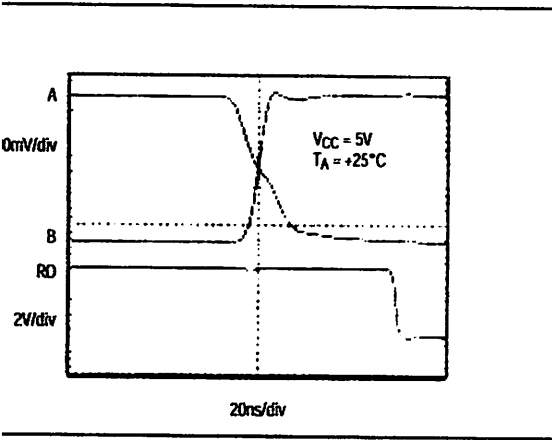


Figure 15. MAX481/MAX485/MAX490/MAX491/MAX1487 Receiver tPHL

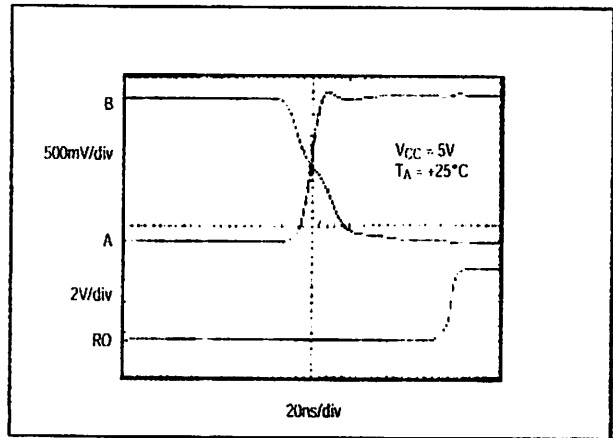


Figure 16. MAX481/MAX485/MAX490/MAX491/MAX1487 Receiver tPLH

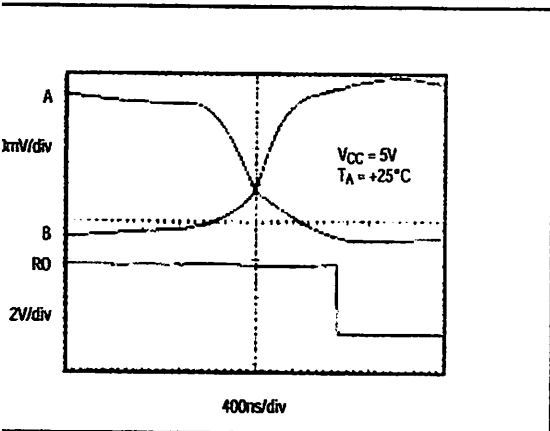


Figure 17. MAX483, MAX487-MAX489 Receiver tPHL

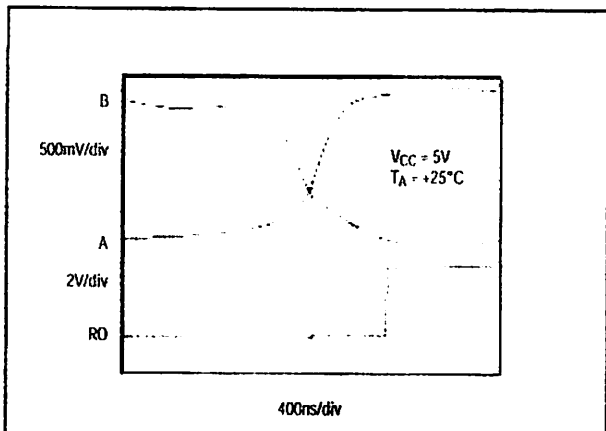


Figure 18. MAX483, MAX487-MAX489 Receiver tPLH

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Line Length vs. Data Rate

RS-485/RS-422 standard covers line lengths up to 4000 feet. For line lengths greater than 4000 feet, see Figure 23.

Figures 19 and 20 show the system differential voltage of the parts driving 4000 feet of 26AWG twisted-pair cable at 110kHz into 120Ω loads.

## Typical Applications

MAX481, MAX483, MAX485, MAX487-MAX491, and MAX1487 transceivers are designed for bidirectional data communications on multipoint bus transmission lines.

Figures 21 and 22 show typical network applications circuits. These parts can also be used as line repeaters, with cable lengths longer than 4000 feet, as shown in Figure 23.

To minimize reflections, the line should be terminated at both ends in its characteristic impedance, and stub lengths off the main line should be kept as short as possible. The slew-rate-limited MAX483 and MAX487-MAX489 are more tolerant of imperfect termination.

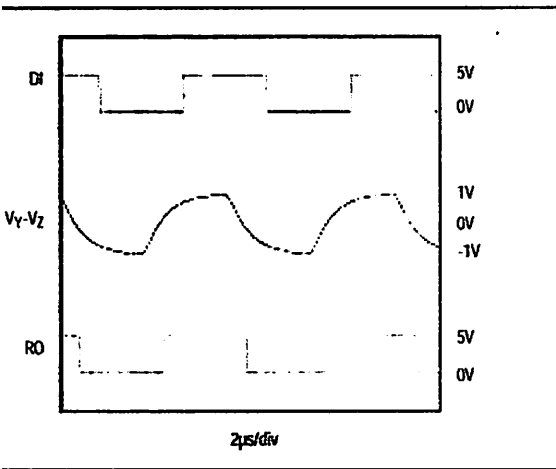


Figure 19. MAX481/MAX485/MAX490/MAX491/MAX1487 System Differential Voltage at 110kHz Driving 4000ft of Cable

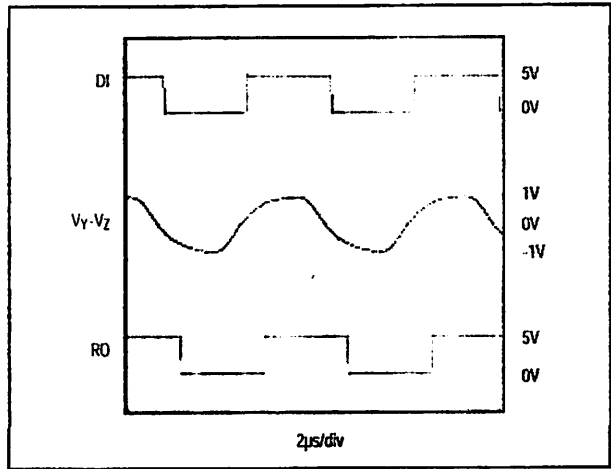


Figure 20. MAX483, MAX487-MAX489 System Differential Voltage at 110kHz Driving 4000ft of Cable

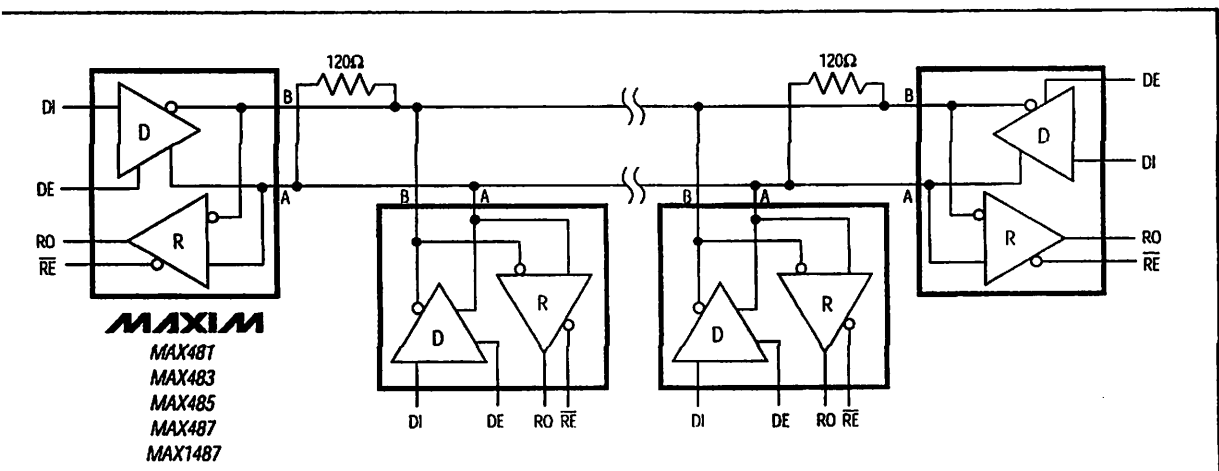
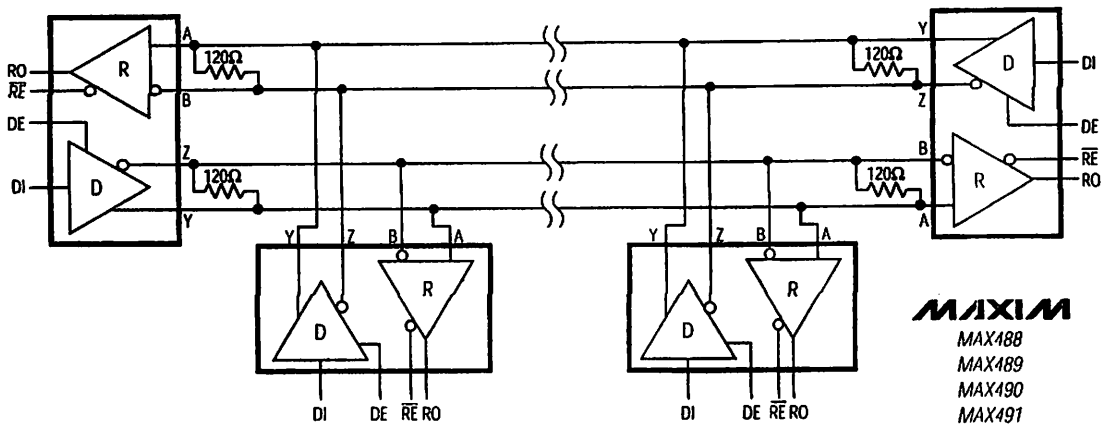


Figure 21. MAX481/MAX483/MAX485/MAX487/MAX1487 Typical Half-Duplex RS-485 Network

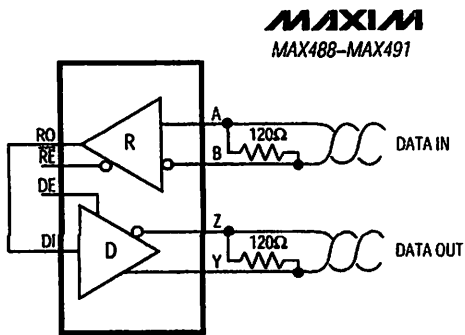
MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers



NOTE:  $\overline{RE}$  AND DE ON MAX489/MAX491 ONLY.

Figure 22. MAX488-MAX491 Full-Duplex RS-485 Network



NOTE:  $\overline{RE}$  AND DE ON MAX489/MAX491 ONLY.

Figure 23. Line Repeater for MAX488-MAX491

## Isolated RS-485

For isolated RS-485 applications, see the MAX253 and MAX1480 data sheets.

# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
AX481EPA	-40°C to +85°C	8 Plastic DIP
AX481ESA	-40°C to +85°C	8 SO
AX481MJA	-55°C to +125°C	8 CERDIP
<b>AX483CPA</b>	0°C to +70°C	8 Plastic DIP
AX483CSA	0°C to +70°C	8 SO
AX483CUA	0°C to +70°C	8 $\mu$ MAX
AX483C/D	0°C to +70°C	Dice*
AX483EPA	-40°C to +85°C	8 Plastic DIP
AX483ESA	-40°C to +85°C	8 SO
AX483MJA	-55°C to +125°C	8 CERDIP
<b>AX485CPA</b>	0°C to +70°C	8 Plastic DIP
AX485CSA	0°C to +70°C	8 SO
AX485CUA	0°C to +70°C	8 $\mu$ MAX
AX485C/D	0°C to +70°C	Dice*
AX485EPA	-40°C to +85°C	8 Plastic DIP
AX485ESA	-40°C to +85°C	8 SO
AX485MJA	-55°C to +125°C	8 CERDIP
<b>AX487CPA</b>	0°C to +70°C	8 Plastic DIP
AX487CSA	0°C to +70°C	8 SO
AX487CUA	0°C to +70°C	8 $\mu$ MAX
AX487C/D	0°C to +70°C	Dice*
AX487EPA	-40°C to +85°C	8 Plastic DIP
AX487ESA	-40°C to +85°C	8 SO
AX487MJA	-55°C to +125°C	8 CERDIP
<b>AX488CPA</b>	0°C to +70°C	8 Plastic DIP
AX488CSA	0°C to +70°C	8 SO
AX488CUA	0°C to +70°C	8 $\mu$ MAX
AX488C/D	0°C to +70°C	Dice*
AX488EPA	-40°C to +85°C	8 Plastic DIP
AX488ESA	-40°C to +85°C	8 SO
AX488MJA	-55°C to +125°C	8 CERDIP
<b>AX489CPD</b>	0°C to +70°C	14 Plastic DIP
AX489CSD	0°C to +70°C	14 SO
AX489C/D	0°C to +70°C	Dice*
AX489EPD	-40°C to +85°C	14 Plastic DIP
AX489ESD	-40°C to +85°C	14 SO
AX489MJD	-55°C to +125°C	14 CERDIP

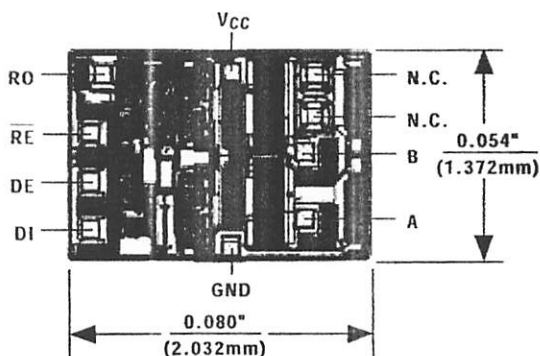
## Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
<b>MAX490CPA</b>	0°C to +70°C	8 Plastic DIP
MAX490CSA	0°C to +70°C	8 SO
MAX490CUA	0°C to +70°C	8 $\mu$ MAX
MAX490C/D	0°C to +70°C	Dice*
MAX490EPA	-40°C to +85°C	8 Plastic DIP
MAX490ESA	-40°C to +85°C	8 SO
MAX490MJA	-55°C to +125°C	8 CERDIP
<b>MAX491CPD</b>	0°C to +70°C	14 Plastic DIP
MAX491CSD	0°C to +70°C	14 SO
MAX491C/D	0°C to +70°C	Dice*
MAX491EPD	-40°C to +85°C	14 Plastic DIP
MAX491ESD	-40°C to +85°C	14 SO
MAX491MJD	-55°C to +125°C	14 CERDIP
<b>MAX1487CPA</b>	0°C to +70°C	8 Plastic DIP
MAX1487CSA	0°C to +70°C	8 SO
MAX1487CUA	0°C to +70°C	8 $\mu$ MAX
MAX1487C/D	0°C to +70°C	Dice*
MAX1487EPA	-40°C to +85°C	8 Plastic DIP
MAX1487ESA	-40°C to +85°C	8 SO
MAX1487MJA	-55°C to +125°C	8 CERDIP

\* Contact factory for dice specifications.

## Chip Topographies

MAX481/MAX483/MAX485/MAX487/MAX1487

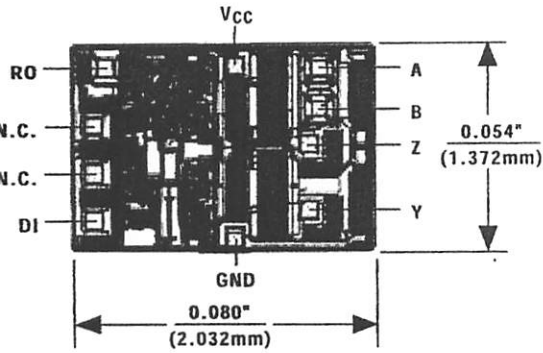


MAX481/MAX483/MAX485/MAX487-MAX491/MAX1487

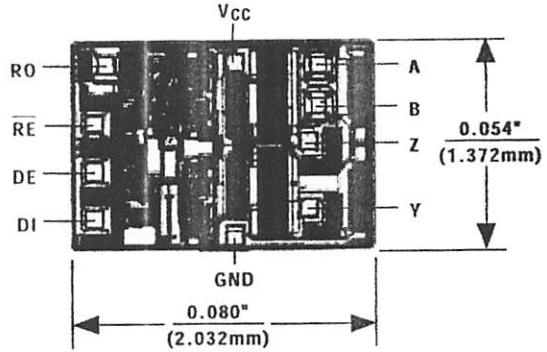
# Low-Power, Slew-Rate-Limited RS-485/RS-422 Transceivers

## Chip Topographies (continued)

MAX488/MAX490

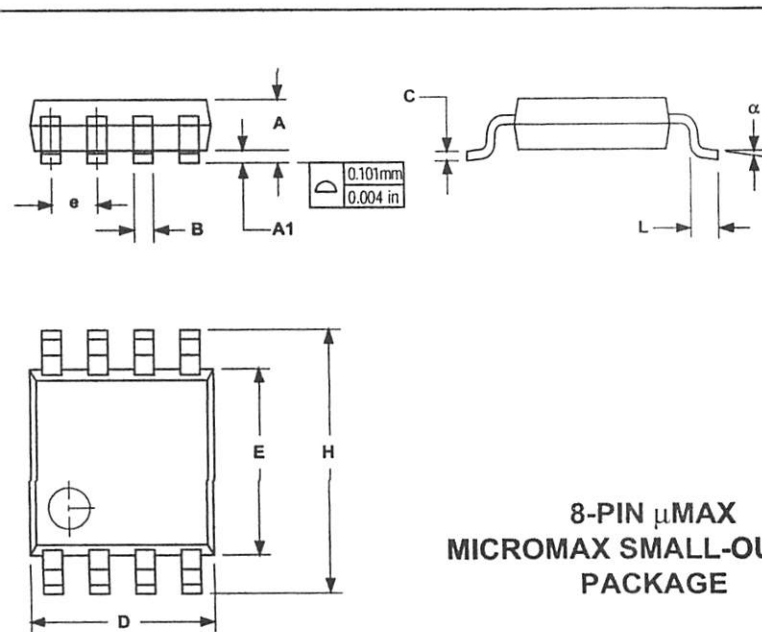


MAX489/MAX491



TRANSISTOR COUNT: 248  
SUBSTRATE CONNECTED TO GND

## Package Information



DIM	INCHES		MILLIMETERS	
	MIN	MAX	MIN	MAX
A	0.036	0.044	0.91	1.11
A1	0.004	0.008	0.10	0.20
B	0.010	0.014	0.25	0.36
C	0.005	0.007	0.13	0.18
D	0.116	0.120	2.95	3.05
E	0.116	0.120	2.95	3.05
e	0.0256		0.65	
H	0.188	0.198	4.78	5.03
L	0.016	0.026	0.41	0.66
α	0°	6°	0°	6°

21-0036D

8-PIN  $\mu$ MAX  
MICROMAX SMALL-OUTLINE  
PACKAGE

Maxim cannot assume responsibility for use of any circuitry other than circuitry entirely embodied in a Maxim product. No circuit patent licenses are granted. Maxim reserves the right to change the circuitry and specifications without notice at any time.

Maxim Integrated Products, 120 San Gabriel Drive, Sunnyvale, CA 94086 (408) 737-7600

## Features

Compatible with MCS-51® Products  
4K Bytes of In-System Programmable (ISP) Flash Memory  
Endurance: 1000 Write/Erase Cycles  
3 to 5.5V Operating Range  
Static Operation: 0 Hz to 33 MHz  
5-level Program Memory Lock  
8-bit Internal RAM  
8 Programmable I/O Lines  
16-bit Timer/Counters  
5 Interrupt Sources  
Full Duplex UART Serial Channel  
Power Idle and Power-down Modes  
Rapid Recovery from Power-down Mode  
Watchdog Timer  
Two Data Pointers  
Power-off Flag  
In-System Programming Time  
Byte and Page ISP Programming (Byte and Page Mode)

## Description

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

The AT89S51 provides the following standard features: 4K bytes of Flash, 128 bytes of internal RAM, 8 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a five-level interrupt architecture, a full duplex serial port, on-chip oscillator, and low power consumption. In addition, the AT89S51 is designed with static logic for operation at zero frequency and supports two software selectable power saving modes. The Idle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and interrupt system to continue functioning. The Power-down mode saves the RAM content, freezes the oscillator, disabling all other chip functions until the next external or hardware reset.



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

### AT89S51

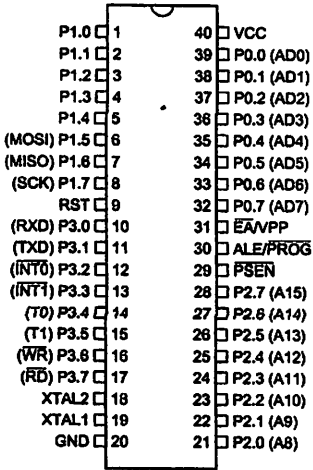
Rev. 2487A-10/01



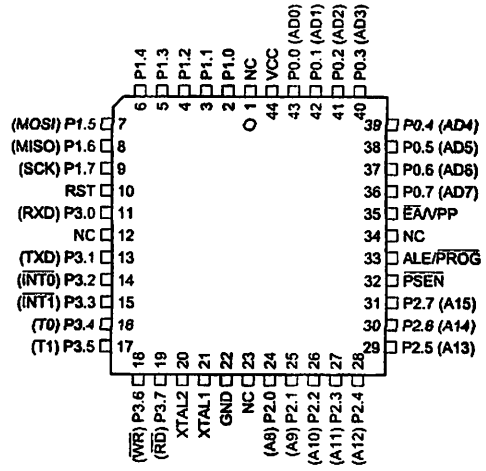


# Configurations

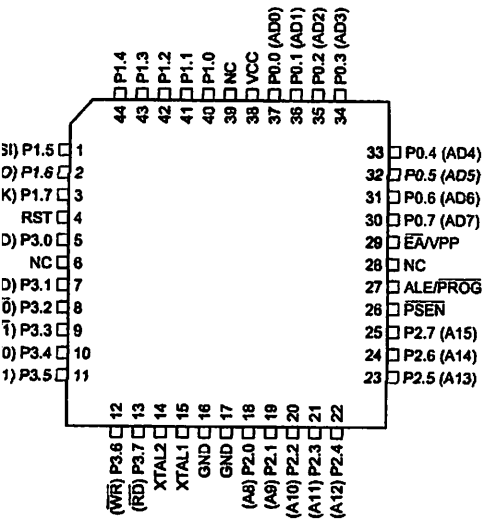
**PDIP**



**PLCC**

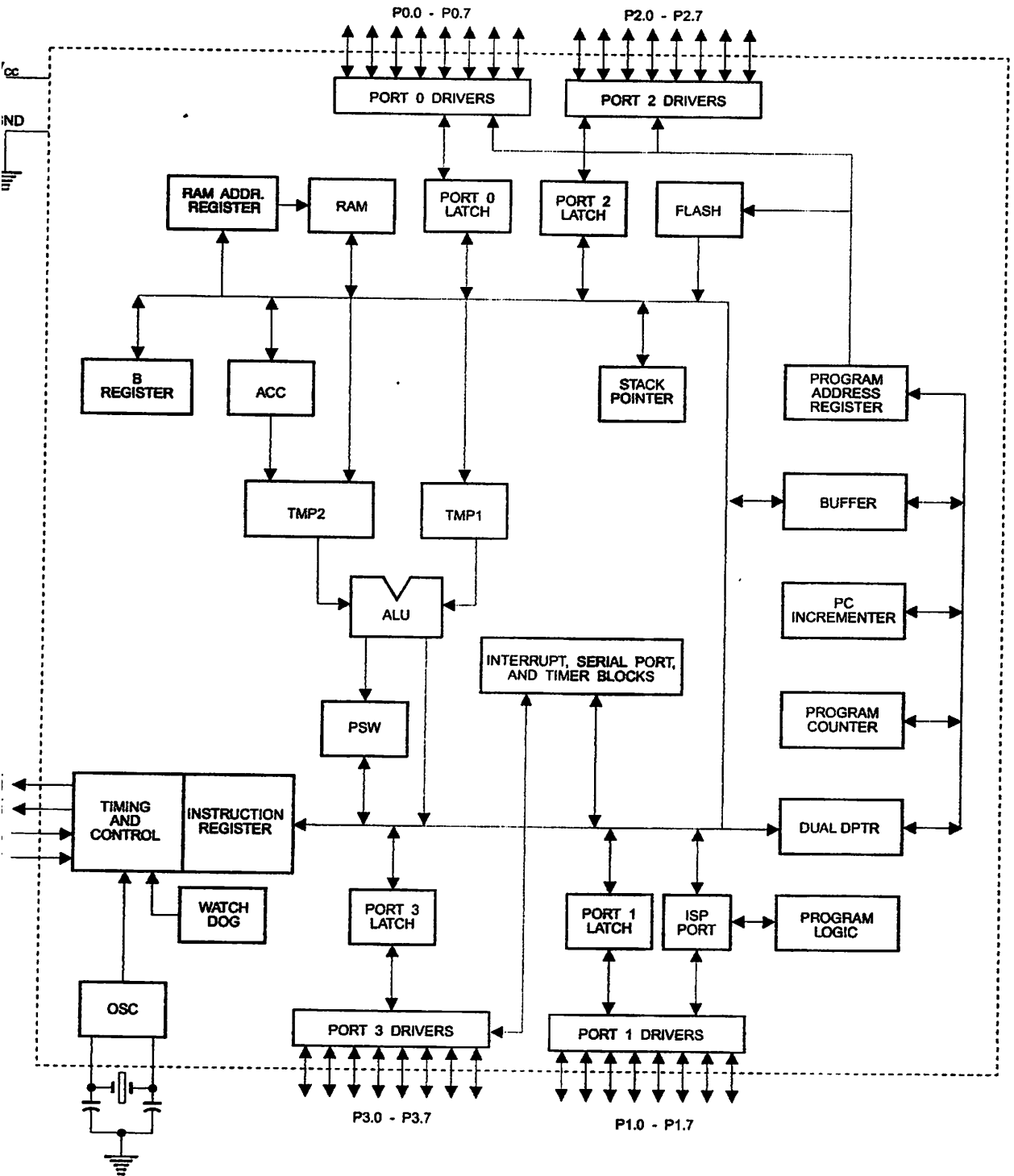


**TQFP**





Diagram





## Description

Supply voltage.

Ground.

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

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

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

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

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

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

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

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

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

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

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

Port 3 also serves the functions of various special features of the AT89S51, as shown in the following table.

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

Reset input. A high on this pin for two machine cycles while the oscillator is running resets the device. This pin drives High for 98 oscillator periods after the Watchdog times out. The DISRTO bit in SFR AUXR (address 8EH) can be used to disable this feature. In the default state of bit DISRTO, the RESET HIGH out feature is enabled.

**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 ( $\overline{\text{PROG}}$ ) during Flash programming.

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

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

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

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

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

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

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

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

Output from the inverting oscillator amplifier



al  
ion  
ters

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

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

AT89S51 SFR Map and Reset Values

								0FFH
B 00000000								0F7H
								0EFH
ACC 00000000								0E7H
								0DFH
PSW 00000000								0D7H
								0CFH
								0C7H
IP XX000000								0BFH
P3 11111111								0B7H
IE 0X000000								0AFH
P2 11111111		AUXR1 XXXXXXXX0				WDTRST XXXXXXXX		0A7H
SCON 00000000	SBUF XXXXXXXX							9FH
P1 11111111								97H
TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXX00XX0		8FH
P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000		PCON 0XXX0000	87H

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

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

**Table 2. AUXR: Auxiliary Register**

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

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



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

**Table 3. AUXR1: Auxiliary Register 1**

AUXR1								
Address = A2H								
Reset Value = XXXXXX0B								
Not Bit Addressable								
Bit	7	6	5	4	3	2	1	DPS
	-	-	-	-	-	-	-	
-	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.

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

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

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

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

**Using the WDT**  
To enable the WDT, a user must write 01EH and 0E1H in sequence to the WDTRST register (SFR location 0A6H). When the WDT is enabled, the user needs to service it by writing 01EH and 0E1H to WDTRST to avoid a WDT overflow. The 14-bit counter overflows when it reaches 16383 (3FFFH), and this will reset the device. When the WDT is enabled, it will increment every machine cycle while the oscillator is running. This means the user must reset the WDT at least every 16383 machine cycles. To reset the WDT the user must write 01EH and 0E1H to WDTRST. WDTRST is a write-only register. The WDT counter cannot be read or written. When WDT overflows, it will generate an output RESET pulse at the RST pin. The RESET pulse duration is  $98 \times TOSC$ , where  $TOSC = 1/FOSC$ . To make the best use of the WDT, it

should be serviced in those sections of code that will periodically be executed within the time required to prevent a WDT reset.

## During Power-down Mode

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

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

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

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

The UART in the AT89S51 operates the same way as the UART in the AT89C51. For further information on the UART operation, refer to the ATMEL Web site (<http://www.atmel.com>). From the home page, select 'Products', then '8051-Architecture Flash Microcontroller', then 'Product Overview'.

## Timer 0 and 1

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

## Interrupts

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

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

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

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



**Table 4.** Interrupt Enable (IE) Register

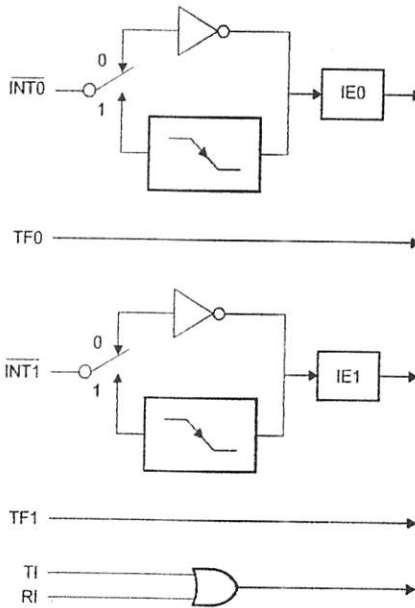
(MSB)				(LSB)			
EA	-	-	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.  
 Enable Bit = 0 disables the interrupt.

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

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

**Figure 1.** Interrupt Sources

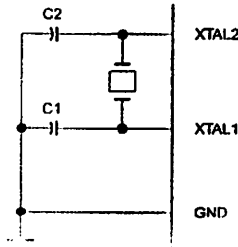




ator  
characteristics

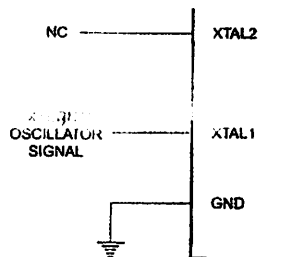
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected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

Figure 2. Oscillator Connections



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

Figure 3. External Clock Drive Configuration



ode

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

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

-down

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



**Table 5. Status of External Pins During Idle and Power-down Modes**

Mode	Program Memory	ALE	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

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

**Table 6. Lock Bit Protection Modes**

Program Lock Bits				Protection Type
	LB1	LB2	LB3	
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, 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 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 EA must agree with the current logic level at that pin in order for the device to function properly.

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

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

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

1. Input the desired memory location on the address lines.
2. Input the appropriate data byte on the data lines.
3. Activate the correct combination of control signals.
4. Raise EA/Vpp to 12V.
5. Pulse ALE/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.

**Data Polling:** The AT89S51 features 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. Data Polling may begin any time after a write cycle has been initiated.

**Ready/Busy:** The progress of byte programming can also be monitored by the  $\overline{RDY/BSY}$  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  $\overline{READY}$ .

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

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

(000H) = 1EH indicates manufactured by Atmel  
 (100H) = 51H indicates 89S51  
 (200H) = 06H

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

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

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

Programming  
 Mode

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

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

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

Programming  
 Sequence

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

1. Power-up sequence:
  - Apply power between VCC and GND pins.
  - Set RST pin to "H".
  - If a crystal is not connected across pins XTAL1 and XTAL2, apply a 3 MHz to 33 MHz clock to XTAL1 pin and wait for at least 10 milliseconds.
2. Enable serial programming by sending the Programming Enable serial instruction to pin MOSI/P1.5. The frequency of the shift clock supplied at pin SCK/P1.7 needs to be less than the CPU clock at XTAL1 divided by 16.
3. The Code array is programmed one byte at a time in either the Byte or Page mode. The write cycle is self-timed and typically takes less than 0.5 ms at 5V.
4. Any memory location can be verified by using the Read instruction that returns the content at the selected address at serial output MISO/P1.6.
5. At the end of a programming session, RST can be set low to commence normal device operation.





Power-off sequence (if needed):  
 Set XTAL1 to "L" (if a crystal is not used).  
 Set RST to "L".  
 Turn V<sub>CC</sub> power off.

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

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

Programming  
 Instruction Set

Programming  
 Instruction Set –  
 Serial Mode

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

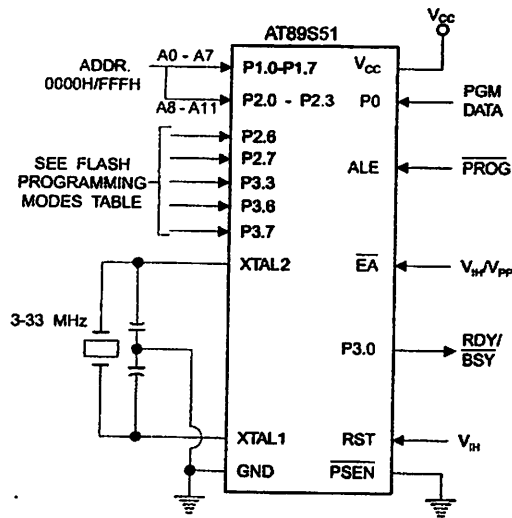
All major programming vendors offer worldwide support for the Atmel microcontroller series. Please contact your local programming vendor for the appropriate software revision.

Flash Programming Modes

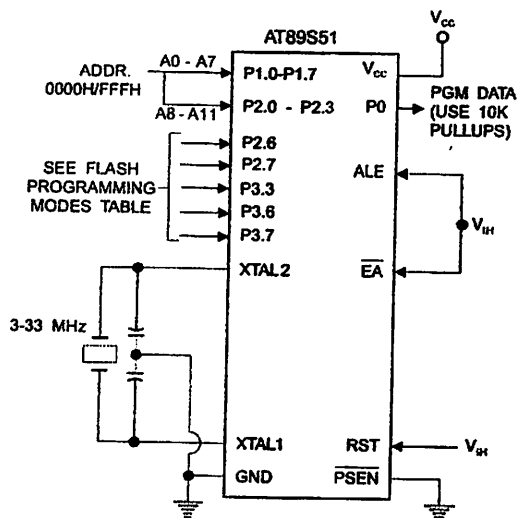
	V <sub>CC</sub>	RST	$\overline{\text{PSEN}}$	ALE/ PROG	$\overline{\text{EA}}/$ V <sub>PP</sub>	P2.6	P2.7	P3.3	P3.6	P3.7	P0.7-0 Data	P2.3-0	P1.7-0
												Address	
Write Code Data	5V	H	L		12V	L	H	H	H	H	D <sub>IN</sub>	A11-8	A7-0
Read Code Data	5V	H	L	H	H	L	L	L	H	H	D <sub>OUT</sub>	A11-8	A7-0
Bit 1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

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

**Figure 4. Programming the Flash Memory (Parallel Mode)**



**Figure 5. Verifying the Flash Memory (Parallel Mode)**



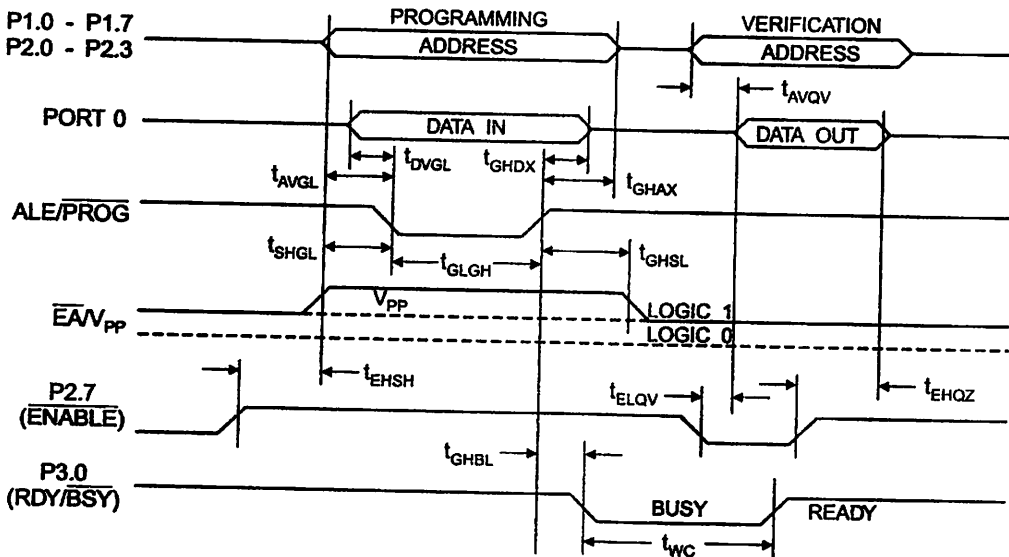


## Programming and Verification Characteristics (Parallel Mode)

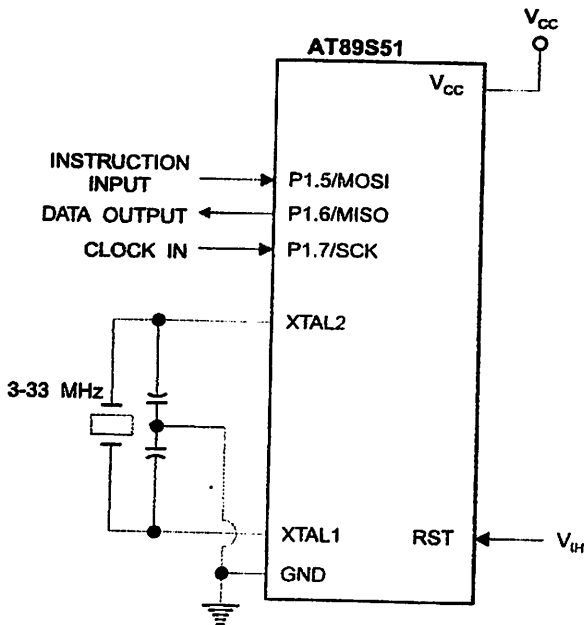
0 to 30°C,  $V_{CC} = 4.5$  to 5.5V

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

### Flash Programming and Verification Waveforms – Parallel Mode

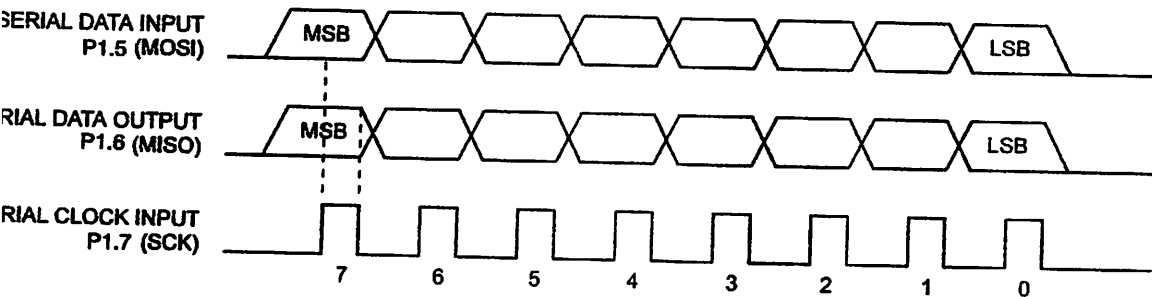


## Flash Memory Serial Downloading



## Programming and Verification Waveforms – Serial Mode

### Serial Programming Waveforms





### Serial Programming Instruction Set

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

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

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



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

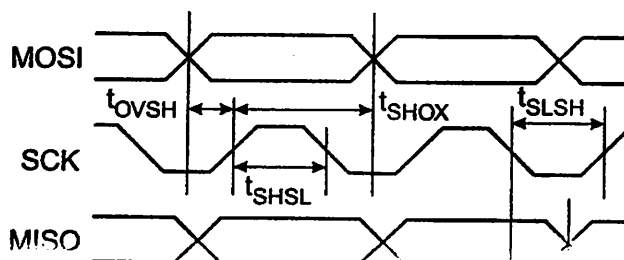
After Reset signal is high, SCK should be low for at least 64 system clocks before it goes high to clock in the enable data bytes. No pulsing of Reset signal is necessary. SCK should be no faster than 1/16 of the system clock at XTAL1.

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



## 1 Programming Characteristics

### 9. Serial Programming Timing



9. Serial Programming Characteristics,  $T_A = -40^\circ\text{C}$  to  $85^\circ\text{C}$ ,  $V_{CC} = 4.0 - 5.5\text{V}$  (Unless Otherwise Noted)

Parameter	Min	Typ	Max	Units
Oscillator Frequency	0		33	MHz
Oscillator Period	30			ns
SCK Pulse Width High	$8 t_{CLCL}$			ns
SCK Pulse Width Low	$8 t_{CLCL}$			ns
MOSI Setup to SCK High	$t_{CLCL}$			ns
MOSI Hold after SCK High	$2 t_{CLCL}$			ns
SCK Low to MISO Valid	10	16	32	ns
Chip Erase Instruction Cycle Time			500	ms
Serial Byte Write Cycle Time			$64 t_{CLCL} + 400$	$\mu\text{s}$



## Absolute Maximum Ratings\*

Storage Temperature.....	-55°C to +125°C
Operating Temperature.....	-65°C to +150°C
Voltage on Any Pin with Respect to Ground.....	-1.0V to +7.0V
Maximum Operating Voltage.....	6.6V
Maximum Input 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 does not imply functional operation of the device at these or any other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

## Characteristics

Conditions shown in this table are valid for  $T_A = -40^\circ\text{C}$  to  $85^\circ\text{C}$  and  $V_{CC} = 4.0\text{V}$  to  $5.5\text{V}$ , unless otherwise noted.

Parameter	Condition	Min	Max	Units
Input Low Voltage	(Except $\bar{E}A$ )	-0.5	$0.2 V_{CC} - 0.1$	V
Input Low Voltage ( $\bar{E}A$ )		-0.5	$0.2 V_{CC} - 0.3$	V
Input High Voltage	(Except XTAL1, RST)	$0.2 V_{CC} + 0.9$	$V_{CC} + 0.5$	V
Input High Voltage	(XTAL1, RST)	$0.7 V_{CC}$	$V_{CC} + 0.5$	V
Output Low Voltage <sup>(1)</sup> (Ports 1,2,3)	$I_{OL} = 1.6 \text{ mA}$		0.45	V
Output Low Voltage <sup>(1)</sup> (Port 0, ALE, $\bar{P}SEN$ )	$I_{OL} = 3.2 \text{ mA}$		0.45	V
Output High Voltage (Ports 1,2,3, ALE, $\bar{P}SEN$ )	$I_{OH} = -60 \mu\text{A}, V_{CC} = 5\text{V} \pm 10\%$	2.4		V
	$I_{OH} = -25 \mu\text{A}$	$0.75 V_{CC}$		V
	$I_{OH} = -10 \mu\text{A}$	$0.9 V_{CC}$		V
Output High Voltage (Port 0 in External Bus Mode)	$I_{OH} = -800 \mu\text{A}, V_{CC} = 5\text{V} \pm 10\%$	2.4		V
	$I_{OH} = -300 \mu\text{A}$	$0.75 V_{CC}$		V
	$I_{OH} = -80 \mu\text{A}$	$0.9 V_{CC}$		V
Logical 0 Input Current (Ports 1,2,3)	$V_{IN} = 0.45\text{V}$		-50	$\mu\text{A}$
Logical 1 to 0 Transition Current (Ports 1,2,3)	$V_{IN} = 2\text{V}, V_{CC} = 5\text{V} \pm 10\%$		-650	$\mu\text{A}$
Input Leakage Current (Port 0, $\bar{E}A$ )	$0.45 < V_{IN} < V_{CC}$		$\pm 10$	$\mu\text{A}$
Reset Pulldown Resistor		50	300	$\text{K}\Omega$
Pin Capacitance	Test Freq. = 1 MHz, $T_A = 25^\circ\text{C}$		10	pF
Power Supply Current	Active Mode, 12 MHz		25	mA
	Idle Mode, 12 MHz		6.5	mA
Power-down Mode <sup>(2)</sup>	$V_{CC} = 5.5\text{V}$		50	$\mu\text{A}$

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.

Minimum  $V_{CC}$  for Power-down is 2V.

# AT89S51

**Characteristics**

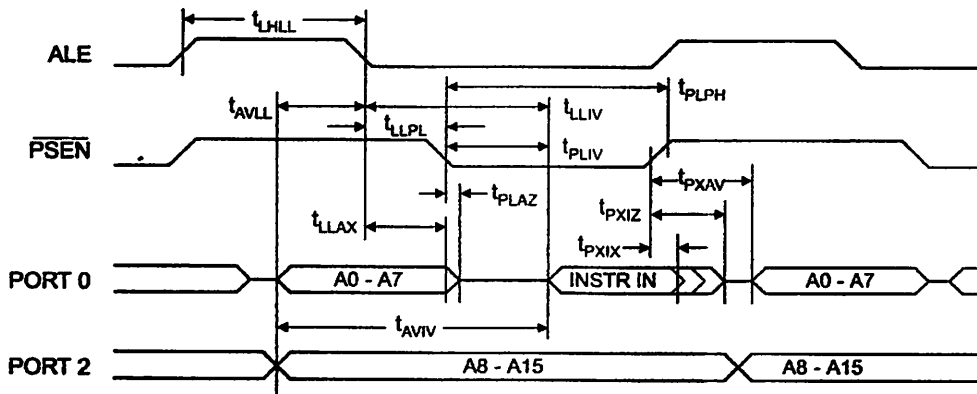
Operating conditions, load capacitance for Port 0, ALE/ $\overline{\text{PROG}}$ , and  $\overline{\text{PSEN}}$  = 100 pF; load capacitance for all other 80 pF.

**Read Program and Data Memory Characteristics**

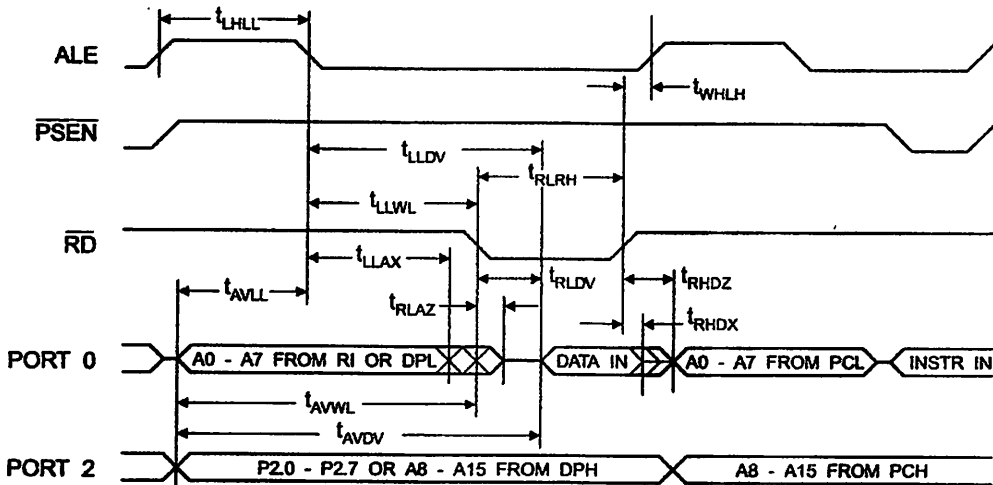
Parameter	12 MHz Oscillator		Variable Oscillator		Units
	Min	Max	Min	Max	
Oscillator Frequency			0	33	MHz
ALE Pulse Width	127		$2t_{\text{CLCL}}-40$		ns
Address Valid to ALE Low	43		$t_{\text{CLCL}}-25$		ns
Address Hold After ALE Low	48		$t_{\text{CLCL}}-25$		ns
ALE Low to Valid Instruction In		233	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



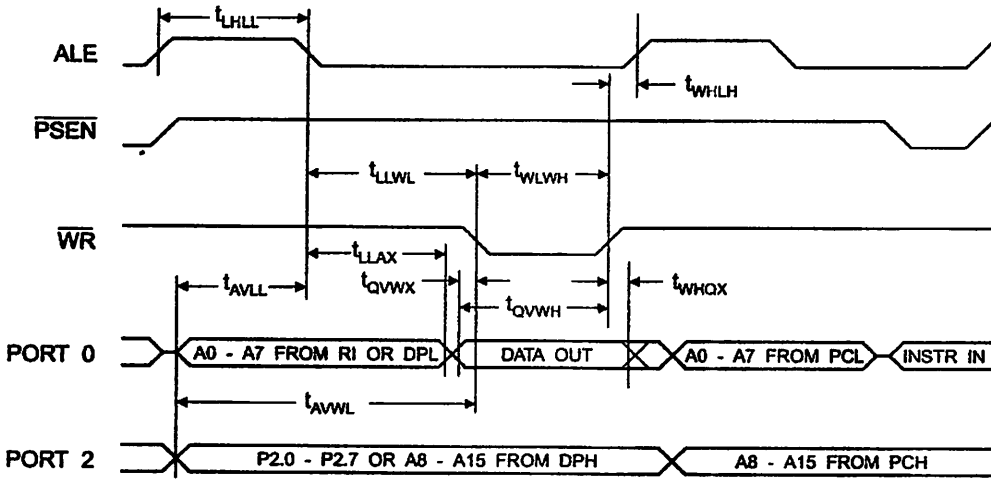
## Program Memory Read Cycle



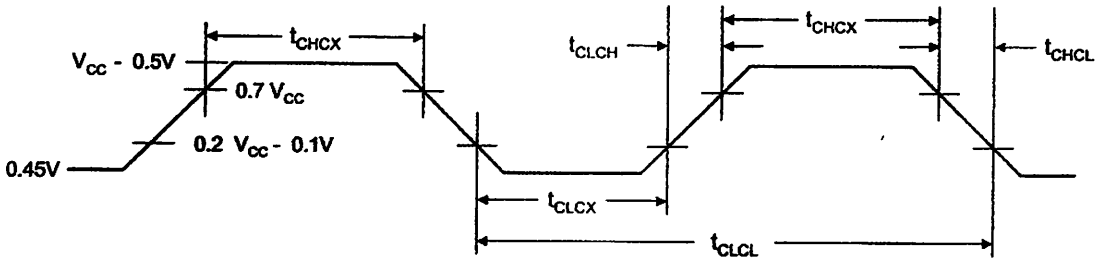
## Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

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





## Operating Information

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

= Preliminary Availability

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

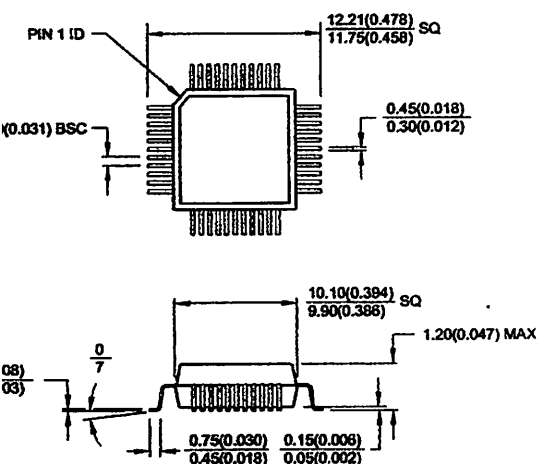




## gging Information

44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Package (TQFP)

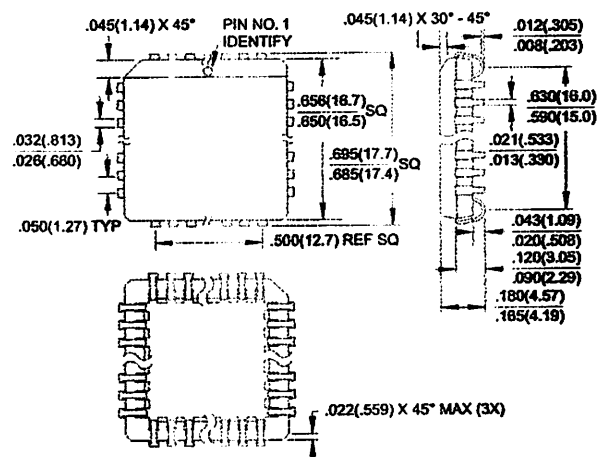
Dimensions in Millimeters and (Inches)\*



Controlling dimension: millimeters

44J, 44-lead, Plastic J-leaded Chip Carrier (PLCC)

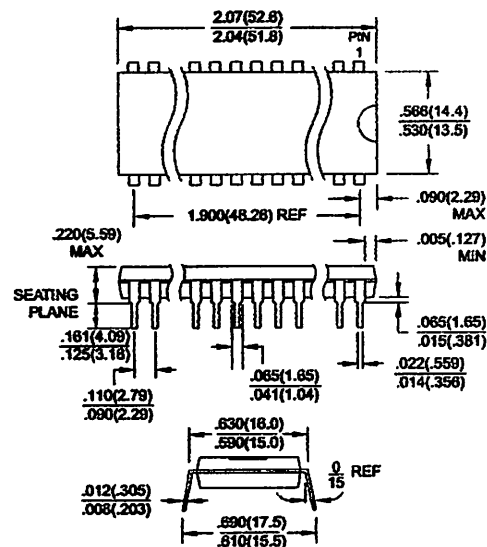
Dimensions in Inches and (Millimeters)



44-pin, 0.600" Wide, Plastic Dual In-line Package (PDIP)

Dimensions in Inches and (Millimeters)

ANSI STANDARD MS-011 AC







## Headquarters

**Atmel Headquarters**  
Orchard Parkway  
Foster, CA 95131  
TEL (408) 441-0311  
FAX (408) 487-2600

**Atmel SarL**  
Rue des Arsenaux 41  
Postale 80  
1705 Fribourg  
Switzerland  
TEL (41) 26-426-5555  
FAX (41) 26-426-5500

**Atmel Asia, Ltd.**  
Room 1219  
Chemical Golden Plaza  
Jardine Road Tsimshatsui  
Kowloon  
Hong Kong  
TEL (852) 2721-9778  
FAX (852) 2722-1369

**Atmel Japan K.K.**  
Aonuma Shinkawa Bldg.  
3 Shinkawa  
Chiyoda-ku, Tokyo 104-0033  
Japan  
TEL (81) 3-3523-3551  
FAX (81) 3-3523-7581

## Atmel Product Operations

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

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

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

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

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

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

---

*e-mail*  
[literature@atmel.com](mailto:literature@atmel.com)

*Web Site*  
<http://www.atmel.com>

## Atmel Corporation 2001.

Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard warranty detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors that may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and does not make any commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use as critical components in life support devices or systems.

Atmel is the registered trademark of Atmel.

Atmel is the registered trademark of Intel Corporation. Terms and product names in this document may be the trademarks of others.



Printed on recycled paper.

2487A-10/01/xM

## Features

- Voltage and Standard-Voltage Operation
- ( $V_{CC} = 2.7V$  to  $5.5V$ )
- ( $V_{CC} = 1.8V$  to  $5.5V$ )
- Low Power Devices ( $I_{SB} = 2 \mu A$  at  $5.5V$ ) Available
- Highly Organized  $4096 \times 8$ ,  $8192 \times 8$
- Serial Interface
- Write Enable Pin, Filtered Inputs for Noise Suppression
- Optional Data Transfer Protocol
- Low Voltage (1.8V, 2.5V, 2.7V) and 400 kHz (5V) Clock Rate
- Write Protect Pin for Hardware Data Protection
- Partial Page Write Mode (Partial Page Writes Allowed)
- Fast Write Cycle (10 ms max)
- High Reliability
- Endurance: 1 Million Write Cycles
- Data Retention: 100 Years
- Automotive Grade and Extended Temperature Devices Available
- JEDEC PDIP, 8-Pin JEDEC SOIC, 8-Pin EIAJ SOIC, 8-Pin TSSOP Packages

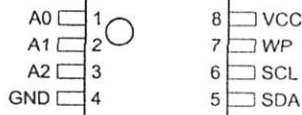
## Description

AT24C32/64 provides 32,768/65,536 bits of serial electrically erasable and programmable read only memory (EEPROM) organized as 4096/8192 words of 8 bits. The device's cascadable feature allows up to 8 devices to share a common 2-wire serial interface. The device is optimized for use in many industrial and commercial applications where low power and low voltage operation are essential. The AT24C32/64 is available in space saving 8-pin JEDEC PDIP, 8-pin JEDEC SOIC, 8-pin EIAJ SOIC, 8-pin TSSOP (AT24C64) packages and is accessed via a 2-wire serial interface. In addition, the entire family is available in 2.7V (2.7V to 5.5V) and 1.8V (1.8V to 5.5V) versions.

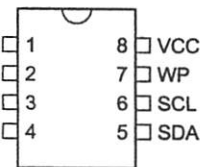
## Pin Configurations

Pin	Function
1, 2, 3	Address Inputs
4	Serial Data
5	Serial Clock Input
6, 7, 8	Write Protect

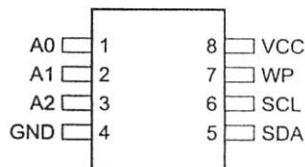
8-Pin TSSOP



8-Pin PDIP



8-Pin SOIC



## 2-Wire Serial EEPROM

32K (4096 x 8)

64K (8192 x 8)

AT24C32

AT24C64

Rev. 0336K-SEEPR-7/03



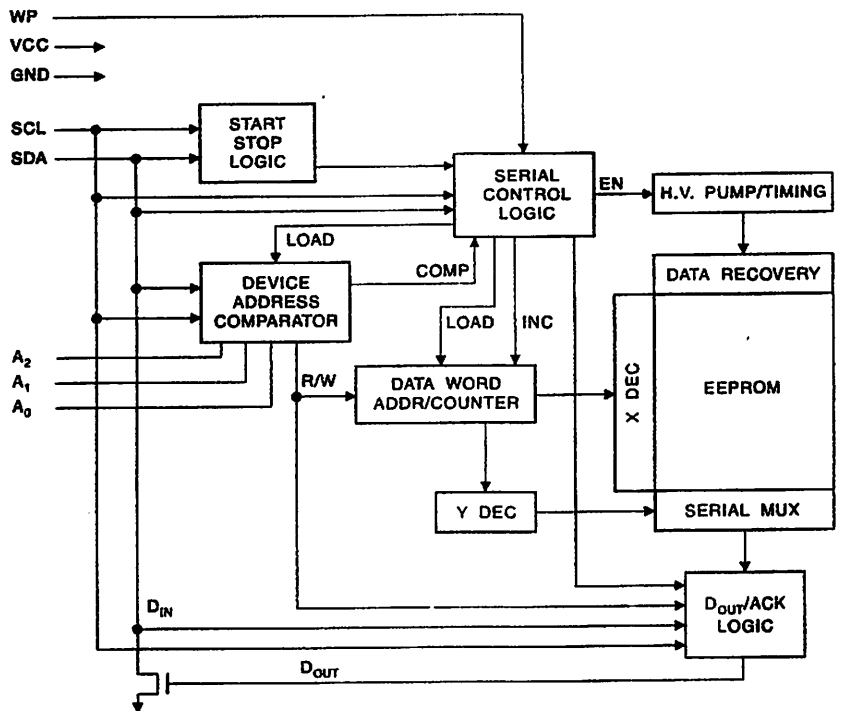


### Absolute Maximum Ratings\*

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 .....	6.25V
Maximum Input Current .....	5.0 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.

### Block Diagram



**Description**

**SERIAL CLOCK (SCL):** The SCL input is used to positive edge clock data into each EEPROM device and negative edge clock data out of each device.

**SERIAL DATA (SDA):** The SDA pin is bidirectional for serial data transfer. This pin is open-drain driven and may be wire-ORed with any number of other open-drain or open collector devices.

**DEVICE/PAGE ADDRESSES (A<sub>2</sub>, A<sub>1</sub>, A<sub>0</sub>):** The A<sub>2</sub>, A<sub>1</sub> and A<sub>0</sub> pins are device address inputs that are hard wired or left not connected for hardware compatibility with AT24C16. When the pins are hardwired, as many as eight 32K/64K devices may be addressed on a single bus system (device addressing is discussed in detail under the Device Addressing section). When the pins are not hardwired, the default A<sub>2</sub>, A<sub>1</sub>, and A<sub>0</sub> are zero.

**WRITE PROTECT (WP):** The write protect input, when tied to GND, allows normal write operations. When WP is tied high to V<sub>CC</sub>, all write operations to the upper quadrant (8/16K bits) of memory are inhibited. If left unconnected, WP is internally pulled down to GND.

**Memory Organization**

**AT24C32/64, 32K/64K SERIAL EEPROM:** The 32K/64K is internally organized as 256 pages of 32 bytes each. Random word addressing requires a 12/13 bit data word address.



## Capacitance<sup>(1)</sup>

over recommended operating range from  $T_A = 25^\circ\text{C}$ ,  $f = 1.0\text{ MHz}$ ,  $V_{CC} = +1.8\text{V}$ .

Test Condition	Max	Units	Conditions
Input/Output Capacitance (SDA)	8	pF	$V_{IO} = 0\text{V}$
Input Capacitance ( $A_0, A_1, A_2, \text{SCL}$ )	6	pF	$V_{IN} = 0\text{V}$

<sup>1</sup> This parameter is characterized and is not 100% tested.

## Characteristics

over recommended operating range from:  $T_{AI} = -40^\circ\text{C}$  to  $+85^\circ\text{C}$ ,  $V_{CC} = +1.8\text{V}$  to  $+5.5\text{V}$ ,  $T_{AC} = 0^\circ\text{C}$  to  $+70^\circ\text{C}$ ,  $V_{CC} = 1.8\text{V}$  to  $+5.5\text{V}$  (unless otherwise noted).

Parameter	Test Condition		Min	Typ	Max	Units
Supply Voltage			1.8		5.5	V
Supply Voltage			2.5		5.5	V
Supply Voltage			2.7		5.5	V
Supply Voltage			4.5		5.5	V
Supply Current	$V_{CC} = 5.0\text{V}$	READ at 100 kHz		0.4	1.0	mA
Supply Current	$V_{CC} = 5.0\text{V}$	WRITE at 100 kHz		2.0	3.0	mA
Standby Current (1.8V option)	$V_{CC} = 1.8\text{V}$	$V_{IN} = V_{CC}$ or $V_{SS}$			0.1	$\mu\text{A}$
	$V_{CC} = 5.5\text{V}$				2.0	
Standby Current (2.5V option)	$V_{CC} = 2.5\text{V}$	$V_{IN} = V_{CC}$ or $V_{SS}$			0.5	$\mu\text{A}$
	$V_{CC} = 5.5\text{V}$				2.0	
Standby Current (2.7V option)	$V_{CC} = 2.7\text{V}$	$V_{IN} = V_{CC}$ or $V_{SS}$			0.5	$\mu\text{A}$
	$V_{CC} = 5.5\text{V}$				2.0	
Standby Current (5V option)	$V_{CC} = 4.5 - 5.5\text{V}$	$V_{IN} = V_{CC}$ or $V_{SS}$		20	35	$\mu\text{A}$
Input Leakage Current	$V_{IN} = V_{CC}$ or $V_{SS}$			0.10	3.0	$\mu\text{A}$
Output Leakage Current	$V_{OUT} = V_{CC}$ or $V_{SS}$			0.05	3.0	$\mu\text{A}$
Input Low Level <sup>(1)</sup>			-0.6		$V_{CC} \times 0.3$	V
Input High Level <sup>(1)</sup>			$V_{CC} \times 0.7$		$V_{CC} + 0.5$	V
Output Low Level	$V_{CC} = 3.0\text{V}$	$I_{OL} = 2.1\text{ mA}$			0.4	V
Output Low Level	$V_{CC} = 1.8\text{V}$	$I_{OL} = 0.15\text{ mA}$			0.2	V

1.  $V_{IL}$  min and  $V_{IH}$  max are reference only and are not tested.

## Characteristics

Operating over recommended operating range from  $T_A = -40^\circ\text{C}$  to  $+85^\circ\text{C}$ ,  $V_{CC} = +1.8\text{V}$  to  $+5.5\text{V}$ ,  $C_L = 1$  TTL Gate and 100 pF (unless otherwise noted).

Parameter*	1.8-volt		2.7-, 2.5-volt		5.0-volt		Units
	Min	Max	Min	Max	Min	Max	
Clock Frequency, SCL		100		100		400	kHz
Clock Pulse Width Low	4.7		4.7		1.2		$\mu\text{s}$
Clock Pulse Width High	4.0		4.0		0.6		$\mu\text{s}$
Noise Suppression Time <sup>(1)</sup>		100		100		50	ns
Clock Low to Data Out Valid	0.1	4.5	0.1	4.5	0.1	0.9	$\mu\text{s}$
Time the bus must be free before a new transmission can start <sup>(1)</sup>	4.7		4.7		1.2		$\mu\text{s}$
Start Hold Time	4.0		4.0		0.6		$\mu\text{s}$
Start Set-up Time	4.7		4.7		0.6		$\mu\text{s}$
Data In Hold Time	0		0		0		$\mu\text{s}$
Data In Set-up Time	200		200		100		ns
Inputs Rise Time <sup>(1)</sup>		1.0		1.0		0.3	$\mu\text{s}$
Inputs Fall Time <sup>(1)</sup>		300		300		300	ns
Stop Set-up Time	4.7		4.7		0.6		$\mu\text{s}$
Data Out Hold Time	100		100		50		ns
Write Cycle Time		20		10		10	ms
Write Cycles <sup>(1)</sup>	5.0V, 25°C, Page Mode	1M		1M		1M	Write Cycles

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



## Operation

**CLOCK and DATA TRANSITIONS:** The SDA pin is normally pulled high with an external device. Data on the SDA pin may change only during SCL low time periods (refer to Data Validity timing diagram). Data changes during SCL high periods will indicate a start or stop condition as defined below.

**START CONDITION:** A high-to-low transition of SDA with SCL high is a start condition which must precede any other command (refer to Start and Stop Definition timing diagram).

**STOP CONDITION:** A low-to-high transition of SDA with SCL high is a stop condition. After a read sequence, the stop command will place the EEPROM in a standby power mode (refer to Start and Stop Definition timing diagram).

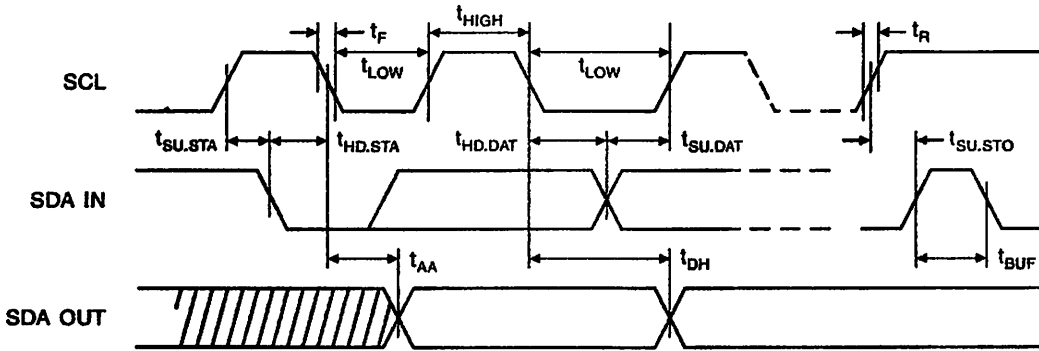
**ACKNOWLEDGE:** All addresses and data words are serially transmitted to and from the EEPROM in 8-bit words. The EEPROM sends a zero during the ninth clock cycle to acknowledge that it has received each word.

**STANDBY MODE:** The AT24C32/64 features a low power standby mode which is enabled: a) upon power-up and b) after the receipt of the STOP bit and the completion of any internal operations.

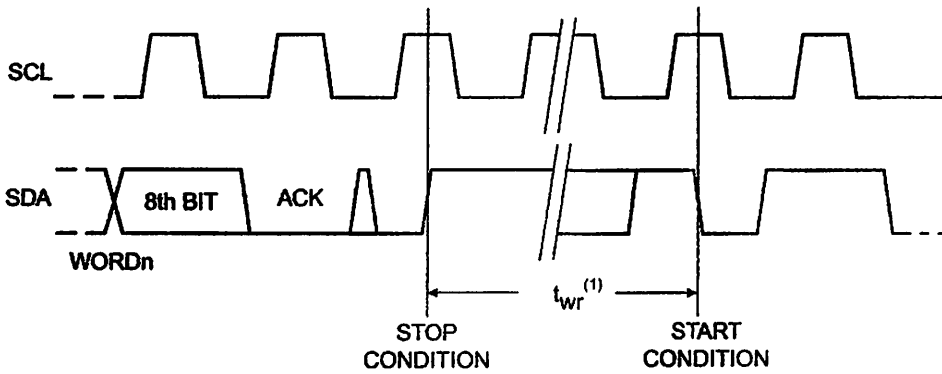
**MEMORY RESET:** After an interruption in protocol, power loss or system reset, any 2-wire part can be reset by following these steps:

(a) Clock up to 9 cycles, (b) look for SDA high in each cycle while SCL is high and then (c) create a start condition as SDA is high.

**Timing**  
**Serial Clock, SDA: Serial Data I/O**



**Cycle Timing**  
**Serial Clock, SDA: Serial Data I/O**

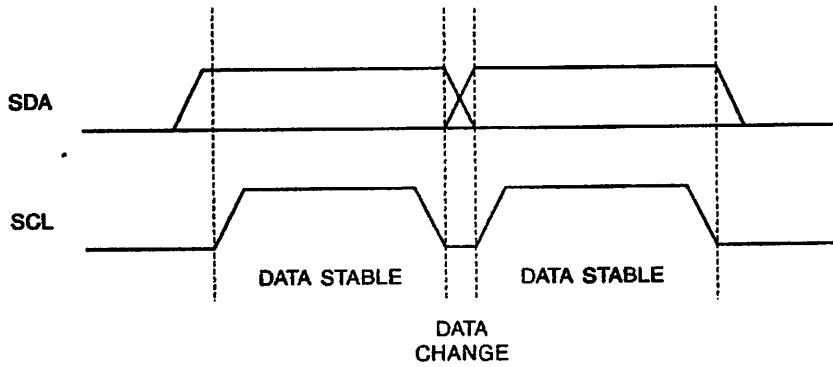


The write cycle time  $t_{WR}$  is the time from a valid stop condition of a write sequence to the end of the internal clear/write cycle.

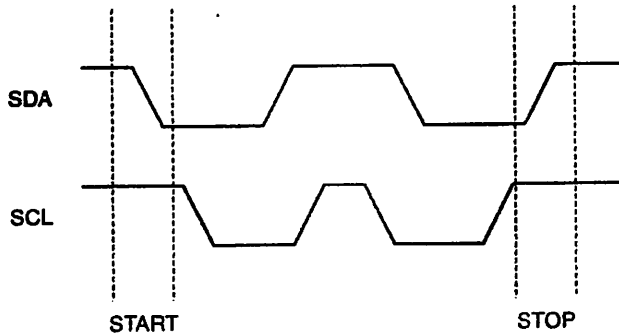




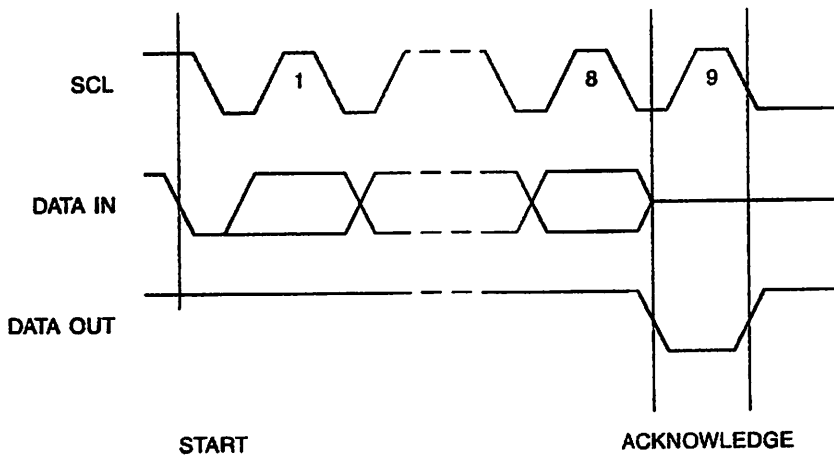
## Validity



## Start and Stop Definition



## Bit Acknowledge



## Addressing

The 32K/64K EEPROM requires an 8-bit device address word following a start condition to enable the chip for a read or write operation (refer to Figure 1). The device address word consists of a mandatory one, zero sequence for the first four most significant bits as shown. This is common to all 2-wire EEPROM devices.

The 32K/64K uses the three device address bits A2, A1, A0 to allow as many as eight devices on the same bus. These bits must compare to their corresponding hardwired input pins. The A2, A1, and A0 pins use an internal proprietary circuit that biases them to a logic low condition if the pins are allowed to float.

The eighth bit of the device address is the read/write operation select bit. A read operation is initiated if this bit is high and a write operation is initiated if this bit is low.

Upon a compare of the device address, the EEPROM will output a zero. If a compare is not made, the device will return to standby state.

**NOISE PROTECTION:** Special internal circuitry placed on the SDA and SCL pins prevent small noise spikes from activating the device. A low- $V_{CC}$  detector (5-volt option) resets the device to prevent data corruption in a noisy environment.

**DATA SECURITY:** The AT24C32/64 has a hardware data protection scheme that allows the user to write protect the upper quadrant (8/16K bits) of memory when the WP pin is at  $V_{CC}$ .

## Operations

**BYTE WRITE:** A write operation requires two 8-bit data word addresses following the device address word and acknowledgment. Upon receipt of this address, the EEPROM will again respond with a zero and then clock in the first 8-bit data word. Following receipt of the 8-bit data word, the EEPROM will output a zero and the addressing device, such as a microcontroller, must terminate the write sequence with a stop condition. At this time the EEPROM enters an internally-timed write cycle,  $t_{WR}$ , to the nonvolatile memory. All inputs are disabled during this write cycle and the EEPROM will not respond until the write is complete (refer to Figure 2).

**PAGE WRITE:** The 32K/64K EEPROM is capable of 32-byte page writes.

A page write is initiated the same way as a byte write, but the microcontroller does not send a stop condition after the first data word is clocked in. Instead, after the EEPROM acknowledges receipt of the first data word, the microcontroller can transmit up to 31 more data words. The EEPROM will respond with a zero after each data word received. The microcontroller must terminate the page write sequence with a stop condition (refer to Figure 3).

The data word address lower 5 bits are internally incremented following the receipt of each data word. The higher data word address bits are not incremented, retaining the memory page row location. When the word address, internally generated, reaches the page boundary, the following byte is placed at the beginning of the same page. If more than 32 data words are transmitted to the EEPROM, the data word address will "roll over" and previous data will be overwritten.

**ACKNOWLEDGE POLLING:** Once the internally-timed write cycle has started and the EEPROM inputs are disabled, acknowledge polling can be initiated. This involves sending a start condition followed by the device address word. The read/write bit is representative of the operation desired. Only if the internal write cycle has completed will the EEPROM respond with a zero, allowing the read or write sequence to continue.





tions

Read operations are initiated the same way as write operations with the exception that the read/write select bit in the device address word is set to one. There are three read operations: current address read, random address read and sequential read.

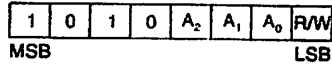
**CURRENT ADDRESS READ:** The internal data word address counter maintains the last address accessed during the last read or write operation, incremented by one. This address stays valid between operations as long as the chip power is maintained. The address "roll over" during read is from the last byte of the last memory page, to the first byte of the first page. The address "roll over" during write is from the last byte of the current page to the first byte of the same page.

Once the device address with the read/write select bit set to one is clocked in and acknowledged by the EEPROM, the current address data word is serially clocked out. The microcontroller does not respond with an input zero but does generate a following stop condition (refer to Figure 4).

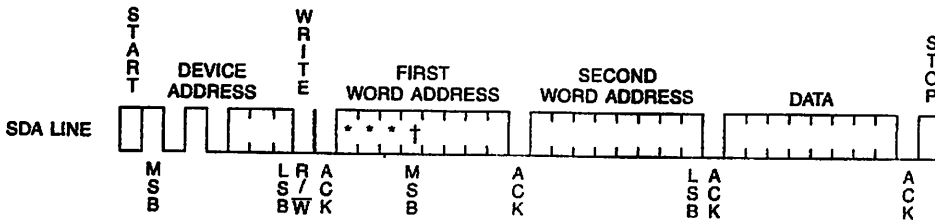
**RANDOM READ:** A random read requires a "dummy" byte write sequence to load in the data word address. Once the device address word and data word address are clocked in and acknowledged by the EEPROM, the microcontroller must generate another start condition. The microcontroller now initiates a current address read by sending a device address with the read/write select bit high. The EEPROM acknowledges the device address and serially clocks out the data word. The microcontroller does not respond with a zero but does generate a following stop condition (refer to Figure 5).

**SEQUENTIAL READ:** Sequential reads are initiated by either a current address read or a random address read. After the microcontroller receives a data word, it responds with an acknowledge. As long as the EEPROM receives an acknowledge, it will continue to increment the data word address and serially clock out sequential data words. When the memory address limit is reached, the data word address will "roll over" and the sequential read will continue. The sequential read operation is terminated when the microcontroller does not respond with a zero but does generate a following stop condition (refer to Figure 6).

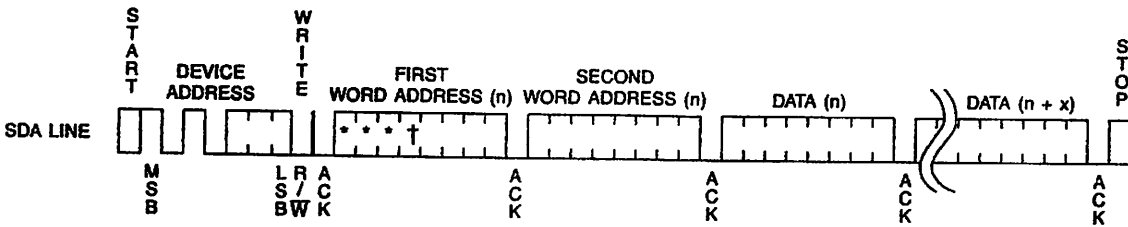
## Device Address



## Byte Write



## Page Write

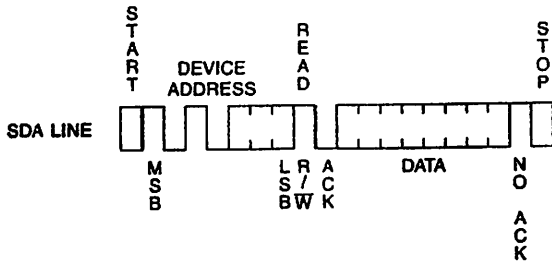


\* = DON'T CARE bits

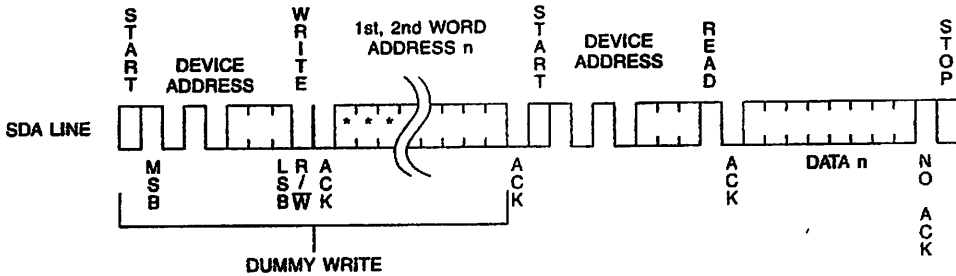
† = DON'T CARE bits for the 32K



Current Address Read

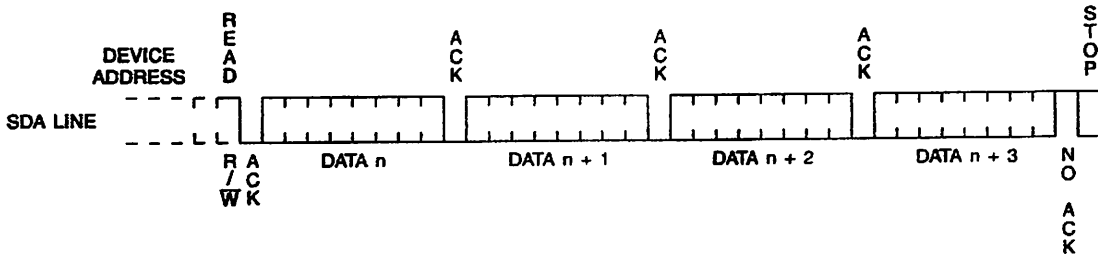


Random Read



\* = DON'T CARE bits

Sequential Read



**Ordering Information**

Ordering Code	Package	Operation Range
AT24C32-10PI-2.7	8P3	Industrial (-40°C to 85°C)
AT24C32N-10SI-2.7	8S1	
AT24C32W-10SI-2.7	8S2	
AT24C32-10PI-1.8	8P3	Industrial (-40°C to 85°C)
AT24C32N-10SI-1.8	8S1	
AT24C32W-10SI-1.8	8S2	

For 2.7V devices used in the 4.5V to 5.5V range, please refer to performance values in the AC and DC characteristics tables.

Package Type	
	8-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
	8-lead, 0.150" Wide, Plastic Gull Wing Small Outline (JEDEC SOIC)
	8-lead, 0.200" Wide, Plastic Gull Wing Small Outline (EIAJ SOIC)
Options	
	Low Voltage (2.7V to 5.5V)
	Low Voltage (1.8V to 5.5V)





## 64 Ordering Information

Part Code	Package	Operation Range
C64-10PI-2.7 C64N-10SI-2.7 C64W-10SI-2.7 C64-10TI-2.7	8P3 8S1 8S2 8A2	Industrial (-40°C to 85°C)
C64-10PI-1.8 C64N-10SI-1.8 C64W-10SI-1.8 C64-10TI-1.8	8P3 8S1 8S2 8A2	Industrial (-40°C to 85°C)

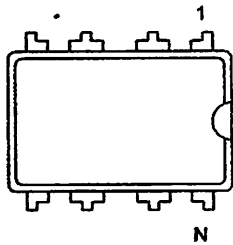
For 2.7V devices used in the 4.5V to 5.5V range, please refer to performance values in the AC and DC characteristics tables.

Package Type	
	8-lead, 0.300" Wide, Plastic Dual Inline Package (PDIP)
	8-lead, 0.150" Wide, Plastic Gull Wing Small Outline (JEDEC SOIC)
	8-lead, 0.200" Wide, Plastic Gull Wing Small Outline (EIAJ SOIC)
	8-lead, 0.170" Wide, Thin Shrink Small Outline Package (TSSOP)
Options	
	Low Voltage (2.7V to 5.5V)
	Low Voltage (1.8V to 5.5V)

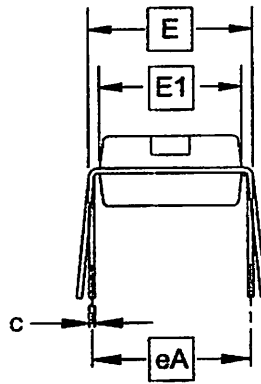
# AT24C32/64

ging Information

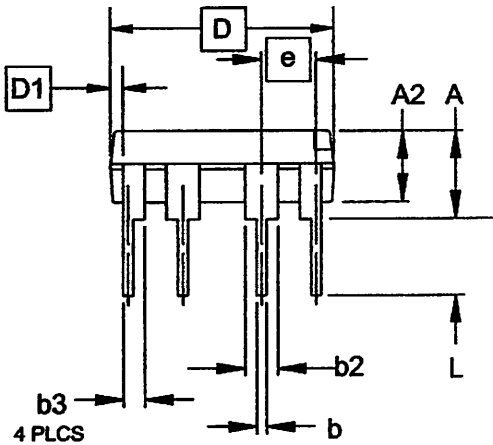
PDIP



Top View



End View



Side View

COMMON DIMENSIONS  
(Unit of Measure = inches)

SYMBOL	MIN	NOM	MAX	NOTE
A			0.210	2
A2	0.115	0.130	0.195	
b	0.014	0.018	0.022	5
b2	0.045	0.060	0.070	6
b3	0.030	0.039	0.045	6
c	0.008	0.010	0.014	
D	0.355	0.365	0.400	3
D1	0.005			3
E	0.300	0.310	0.325	4
E1	0.240	0.250	0.280	3
e	0.100 BSC			
eA	0.300 BSC			4
L	0.115	0.130	0.150	2

1. This drawing is for general information only; refer to JEDEC Drawing MS-001, Variation BA for additional information.
2. Dimensions A and L are measured with the package seated in JEDEC seating plane Gauge GS-3.
3. D, D1 and E1 dimensions do not include mold Flash or protrusions. Mold Flash or protrusions shall not exceed 0.010 inch.
4. E and eA measured with the leads constrained to be perpendicular to datum.
5. Pointed or rounded lead tips are preferred to ease insertion.
6. b2 and b3 maximum dimensions do not include Dambar protrusions. Dambar protrusions shall not exceed 0.010 (0.25 mm).

01/09/02

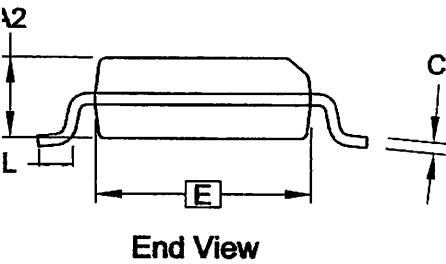
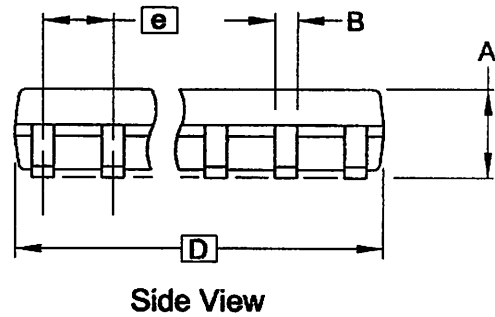
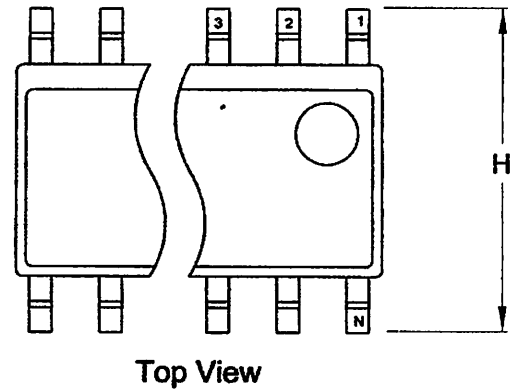
<p>2325 Orchard Parkway San Jose, CA 95131</p>	<p>TITLE 8P3, 8-lead, 0.300" Wide Body, Plastic Dual In-line Package (PDIP)</p>	<p>DRAWING NO. 8P3</p>	<p>REV. B</p>
--	---	----------------------------	-------------------







**JEDEC SOIC**



**COMMON DIMENSIONS**  
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	-	-	1.75	
B	-	-	0.51	
C	-	-	0.25	
D	-	-	5.00	
E	-	-	4.00	
e	1.27 BSC			
H	-	-	6.20	
L	-	-	1.27	

This drawing is for general information only. Refer to JEDEC Drawing MS-012 for proper dimensions, tolerances, datums, etc.

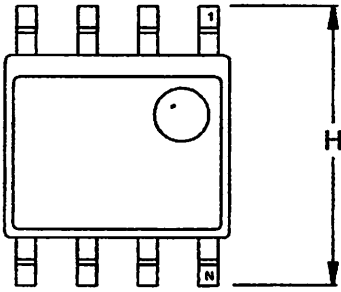
10/10/01

2325 Orchard Parkway San Jose, CA 95131	<b>TITLE</b> <b>8S1</b> , 8-lead (0.150" Wide Body), Plastic Gull Wing Small Outline (JEDEC SOIC)	<b>DRAWING NO.</b> 8S1	<b>REV.</b> A
--	---	---------------------------	------------------

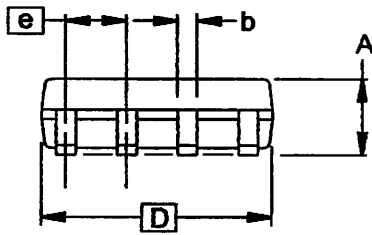
**AT24C32/64**

0336K-SEEPR-7/03

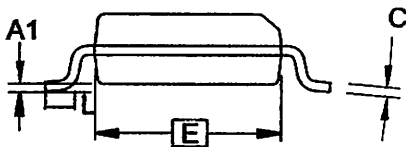
EIAJ SOIC



Top View



Side View



End View

COMMON DIMENSIONS  
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	1.78		2.03	
A1	0.05		0.33	
b	0.35		0.51	5
C	0.18		0.25	5
D	5.13		5.38	
E	5.13		5.41	2, 3
H	7.62		8.38	
L	0.51		0.89	
e	1.27 BSC			4

1. This drawing is for general information only; refer to EIAJ Drawing EDR-7320 for additional information.
2. Mismatch of the upper and lower dies and resin burrs aren't included.
3. It is recommended that upper and lower cavities be equal. If they are different, the larger dimension shall be regarded.
4. Determines the true geometric position.
5. Values b, C apply to pb/Sn solder plated terminal. The standard thickness of the solder layer shall be 0.010 +0.010/-0.005 mm.

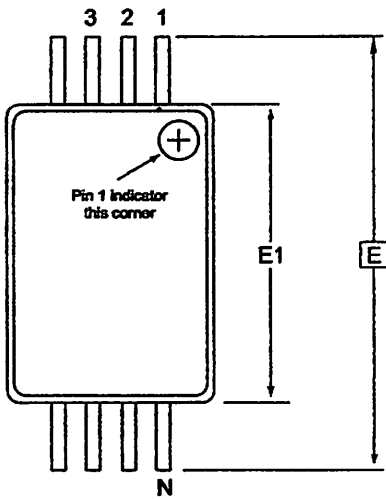
5/2/02

2325 Orchard Parkway San Jose, CA 95131	<b>TITLE</b> 8S2, 8-lead, 0.209" Body, Plastic Small Outline Package (EIAJ)	<b>DRAWING NO.</b> 8S2	<b>REV.</b> B
--	---	---------------------------	------------------

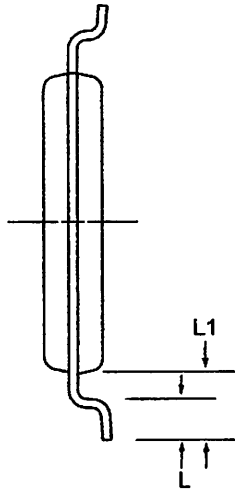




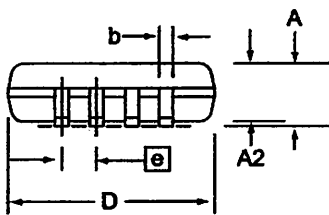
TSSOP



Top View



End View



Side View

**COMMON DIMENSIONS**  
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
D	2.90	3.00	3.10	2, 5
E	6.40 BSC			
E1	4.30	4.40	4.50	3, 5
A	-	-	1.20	
A2	0.80	1.00	1.05	
b	0.19	-	0.30	4
e	0.65 BSC			
L	0.45	0.60	0.75	
L1	1.00 REF			

1. This drawing is for general information only. Refer to JEDEC Drawing MO-153, Variation AA, for proper dimensions, tolerances, datums, etc.
2. Dimension D does not include mold Flash, protrusions or gate burrs. Mold Flash, protrusions and gate burrs shall not exceed 0.15 mm (0.006 in) per side.
3. Dimension E1 does not include inter-lead Flash or protrusions. Inter-lead Flash and protrusions shall not exceed 0.25 mm (0.010 in) per side.
4. Dimension b does not include Dambar protrusion. Allowable Dambar protrusion shall be 0.08 mm total in excess of the b dimension at maximum material condition. Dambar cannot be located on the lower radius of the foot. Minimum space between protrusion and adjacent lead is 0.07 mm.
5. Dimension D and E1 to be determined at Datum Plane H.

5/30/02

<p>2325 Orchard Parkway San Jose, CA 95131</p>	<p><b>TITLE</b> 8A2, 8-lead, 4.4 mm Body, Plastic Thin Shrink Small Outline Package (TSSOP)</p>	<p><b>DRAWING NO.</b> 8A2</p>	<p><b>REV.</b> B</p>
--	---	-----------------------------------	--------------------------

AT24C32/64

0338K-SEEPR-7/03



## Corporation

Orchard Parkway  
Folsom, CA 95131  
Tel: (408) 441-0311  
Fax: (408) 487-2600

## Regional Headquarters

Sarl  
des Arsenaux 41  
Postale 80  
1705 Fribourg  
Switzerland  
Tel: (41) 26-426-5555  
Fax: (41) 26-426-5500

1219  
Chemical Golden Plaza  
Lody Road Tsimshatsui  
Kowloon  
Hong Kong  
Tel: (852) 2721-9778  
Fax: (852) 2722-1369

Onetsu Shinkawa Bldg.  
1-1-1 Shinkawa  
Nagasaki, Tokyo 104-0033

Tel: (81) 3-3523-3551  
Fax: (81) 3-3523-7581

## Atmel Operations

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

**Microcontrollers**  
2325 Orchard Parkway  
San Jose, CA 95131  
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  
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  
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

**e-mail**  
[literature@atmel.com](mailto:literature@atmel.com)

**Web Site**  
<http://www.atmel.com>

Atmel Corporation makes no warranty for the use of its products, other than those expressly contained in the Company's standard terms and conditions which is detailed in Atmel's Terms and Conditions located on the Company's web site. The Company assumes no responsibility for any errors that may appear in this document, reserves the right to change devices or specifications detailed herein at any time without notice, and makes no commitment to update the information contained herein. No licenses to patents or other intellectual property of Atmel are granted by the Company in connection with the sale of Atmel products, expressly or by implication. Atmel's products are not authorized for use in life support devices or systems.

© Atmel Corporation 2003. All rights reserved. Atmel® and combinations thereof, are the registered trademarks of Atmel Corporation or its subsidiaries. Other terms and product names may be the trademarks of their respective owners.



Printed on recycled paper.

0336K-SEEPR-7/03

xM