SKRIPSI

PERENCANAAN DAN PEMBUATAN SISTEM PEMESANAN MAKANAN PADA RESTORAN BERBASIS MIKROKONTROLLER AT89S51

Disusun Oleh :

SEPVIYAN YUDIANTHO

03.17.019

JURUSAN TEKNIK ELEKTRO S -1

KONSENTRASI TEKNIK ELEKTRONIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2009

LEMBAR PERSETUJUAN

PERENCANAAN DAN PEMBUATAN SISTEM PEMESANAN MAKANAN PADA RESTORAN BERBASIS MIKROKONTROLLER AT89S51

SKRIPSI

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

Disusun Oleh :

SEPVIYAN YUDIANTHO
03.17.019

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
M.Ibrahim Ashari, ST, MT	**Irmalia Suryani Faradisa, ST, MT**
NIP.Y.1030100358	**NIP. P. 1030000365**

Mengetahui

Ketua Jurusan Teknik Elektro S-1

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

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

2009

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

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	: Sepviyan Yudiantho
NIM	: 03.17.019
Jurusan	: Teknik Elektro S-1
Konsentrasi	: Teknik Elektronika
Masa Bimbingan	: 9 November 2008 s/d 9 mei 2009
Judul Skripsi	: Perencanaan dan pembuatan sistem pemesanan makanan pada restoran berbasis mikrokontroller AT89S51

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

Hari	: Selasa
Tanggal	: 17 Maret 2009
Dengan Nilai	: 75,75 (B+)

Ketua Majelis Pengguji

Ir.H.Sidik Noertjahjono, MT
NIP.Y. 1028700163

Sekretaris Majelis Pengguji

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

Penguji I

Sotyohadi, ST
NIP. Y. 1039700309

Penguji II

Joseph Dedy Irawan, ST, MT
NIP. 132315178

ABSTRAKSI

PERENCANAAN DAN PEMBUATAN SISTEM PEMESANAN MAKANAN PADA RESTORAN BERBASIS MIKROKONTROLLER AT89S51

SEPVIYAN YUDIANTHO

03.17.019

Jurusan Teknik Elektronika S1- Institut Teknologi Nasional Malang
Jl.Raya Karanglo Km.02 Malang, Indonesia
pluto_59@yahoo.co.id

Abstrak

Sistem pelayanan restaurant di era modern ini masih menggunakan sistem manual. Dalam proyek akhir ini akan menyajikan penggunaan modul sistem minimum Mikrokontroller AT89S51untuk perancangan dan pembuatan sistem minimum untuk pemesanan menu pada meja konsumen. *Microsoft Access* digunakan sebagai *database* dan *Delphi 7.0* difungsikan untuk form pemanggil data data dari *Microsoft Access* dengan memanfaatkan fasilitas. Data Manager yang ada di *Delphi 7.0.* Masukan dari meja makan yang berisikan kode menu dan jumlah menu. Akan ditampilkan pada LCD dan dengan menggunakan komunikasi serial RS 485 DB9, data akan dikirimkan dari mikroprosesor ke PC bagian dapur dengan menggunakan komunikasi yang sama.

Kata kunci : Mikrokontroller AT89S51, LCD, Keypad, Delphi 7.0, Microsoft Acces.

Abstrack

The restaurant service system in this modern era stills use the manual system. In this Preface Of The Final Project Paper present using of Microcontroller AT89S51 in designing and making of minimum system for ordering menu on consumer desk. Microsoft Access used a database and Delphie 7.0 functioned for the form of caller of data from Microsoft Access by exploiting l Data Manager facility exist in Delphi 7.0. Input from the table about the codes of food will be show to the LCD. This information from the table about the codes of food will be show to the LCD. This information from the table will be sent with RS-485,DB9 to PC Kitchen by using same communication.

***Keywords** : Microcontroller AT89S51, LCD, Keypad, Delphi 7.0, Microsoft Access.*

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat-Mu Ya Allah yang telah memberikan rahmat dan hidayah-Mu, sehingga saya dapat menyelesaikan skripsi yang berjudul **" Perencanaan dan pembuatan sistem pemesanan makanan pada restoran berbasis mikrokontroller AT89S51 "** ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Elektronika ITN Malang dan untuk mencapai gelar Sarjana Teknik.

Keberhasilan peyelesaian laporan skripsi ini tidak lepas dari dukungan dan bantuan berbagai pihak. Untuk itu penulis menyampaikan terima kasih kepada :

1. Bapak Prof. Dr. Eng. Ir. Abraham Lomi, MSEE selaku Rektor ITN Malang.
2. Bapak, Ir.H.Sidik Noertjahjono, MT selaku selaku Dekan Fakultas Teknologi Industri.
3. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
4. Bapak M.Ibrahin Ashari,ST, MT selaku Dosen Pembimbing I.
5. Ibu Irmalia Suryani Faradisa, ST, MT selaku Dosen Pembimbing II.
6. Kedua orang tuaku dan kakak-kakakku yang telah memberikan dukungan.
7. Semua temen – temen yang telah membantu dalam penyelesaian penyusunan skripsi ini.

Penulis telah berusaha semaksimal mungkin dan menyadari sepenuhnya akan keterbatasan pengetahuan dalam menyelesaikan laporan ini. Untuk itu

Penulis mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

Harapan penulis semoga laporan skripsi ini memberikan manfaat bagi perkembangan ilmu pengetahuan dan pembaca.

Malang, Maret 2009

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Dalam perkembangan dunia saat ini sudah demikian pesatnya dengan dukungan ilmu teknik elektronika. Teknologi menjadi suatu media penerapan yang membantu manusia dalam hampir seluruh aspek kehidupan. Salah satunya dalam pemesanan makanan pada restoran, dimana pelayan restoran tidak perlu datang untuk memberikan menu makanan. Selain kurang efesien biasanya pengunjung lama menunggu pelayan restoran datang. Hal ini membuat pengunjung agak kecewa dengan pelayanan yang diberikan oleh restoran tersebut.

Untuk itu penulis bermaksud membuat suatu alat pemesanan makanan pada meja restoran berbasis mikrokontroller AT89S51. Dengan penggunaan teknologi seperti ini memudahkan pelayanan yang cepat dan lebih mudah. Karena pemesanan makanan langsung kedapur restoran tanpa pelayan mengantar menu kedapur.

1.2. Rumusan Masalah

Berdasarkan latar belakang yang telah diuraikan pada bagian sebelumnya, maka permasalahannya adalah :

1. Bagaimana merancang dan membuat rangkaian *hardware.*
2. Bagaimana merancang dan membuat perangkat lunak atau *software* pada mikrokontroller yang mengendalikan semua kerja sistem.

1.3. Batasan Masalah

Untuk menyederhanakan pembahasan maka diberikan beberapa batasan masalah sebagai berikut:

1. Tidak membahas pemesan pindah meja.
2. Sebagai pengontrolnya menggunakan mikrokontroller AT89S51
3. Sebagai tempat keypad digunakan papan tombol
4. Sebagai penghubung atau komunikasi menggunakan RS-485
5. Tidak membahas catu daya.

1.4. Tujuan

Tujuan penulisan adalah untuk membuat alat pelayanan menu restoran dan papan tombol pada masing – masing meja menggunakan serial multipoint RS-485 berbasis mikrokontroller AT89S51.

1.5. Metodologi Penelitian

Untuk merealisasikan sistem yang dirancang maka dilakukan langkah – langkah:

1. Studi *literature*, yaitu melakukan pencarian informasi dan data – data dari referensi yang berhubungan dengan ala seperti menu – menu cepat saji, mikrokontroller dan komunikasi serial multipont RS-485.
2. Perancangan software dan validasi *software*, yaitu melakukan kegiatan pembutan progaram, rangkain perblok hingga pembuatan PCB rangkaian keseluruhan sampai melakukan perakitan komponen.

3. Pengujian dan analisa, pada bagian ini melakukan uji hardware dan software, kemudian melakukan analisis berdasarkan hasil pengujian yang dilakukan.
4. Pengujian alat dilakukan untuk mengetahui apakah alat yang dibuat telah sesuai dengan yang direncanakan.
5. Pembahasan analisa data menggunakan progam *Delphi.*
6. Kesimpulan saran.

1.8. Sistematika Penulisan

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 Landasan Teori

Pada bab ini dibahas tentang teori-teori yang mendukung dalam perencanaan dan pembuatan alat ini yang meliputi rangkaian mikrokontroller AT89S51.

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 dan percobaan.

BAB V Penutup

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

BAB II

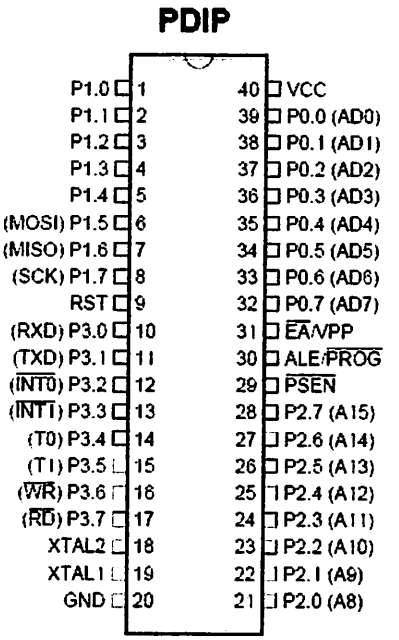
DASAR TEORI

2.1. Umum.

Pada bab ini akan diberikan teori dasar yang melandasi permasalahan dan penyelesaiannya yang diangkat dalam proyek akhir ini. Teori dasar yang diberikan meliputi: pembuatan *data base*, yang memberikan definisi dan klasifikasi tentang penyimpanan serta koneksi database antara *Delphi* dan *Microsoft access*. Selanjutnya, diberikan tentang teori komunikasi data serial, pararel, *connector interface* RS 485, pengaksesan *port serial* pada *Delphi* serta komunikasi serial pada Mikrokontroller AT89S51.

2.2. Mikrokontroller AT89S51.

AT89S51 adalah *mikrokontroller* keluaran *Atmel* dengan 4K *byte Flash* PEROM *(Programmable and Erasable Read Only Memory)*, AT89S51 merupakan memori dengan teknologi *nonvolatile memory*, isi memori tersebut dapat diisi ulang ataupun dihapus berkali-kali. Memori ini biasa digunakan untuk menyimpan instruksi (perintah) berstandar MCS-51 code sehingga memungkinkan mikrokontroller ini untuk bekerja dalam mode *single chip operation* (mode operasi keping tunggal) yang tidak memerlukan *external memory* (memori luar) unruk menyimpan *source code* tersebut. Gambar 2.1 menjukkan konfigurasi pin mikrokontroller AT89S51.



Gambar 2.1. *Konfigurasi Pin Mikrokontroller AT89S51*

Deskripsi Mikrokontroller AT89S51:

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

Port 0 dapat berfungsi sebagai I/O biasa, *low order multiplex addres/data* atau pun menerima kode *byte* pada saat *Flash Programming*. Pada saat sebagai I/O biasa port ini dapat memberikan *output sink* ke delapan buah *Transistor Transistor Logic* (TTL) input atau dapat diubah sebagai input dengan memberikan logika 1 pada port tersebut.

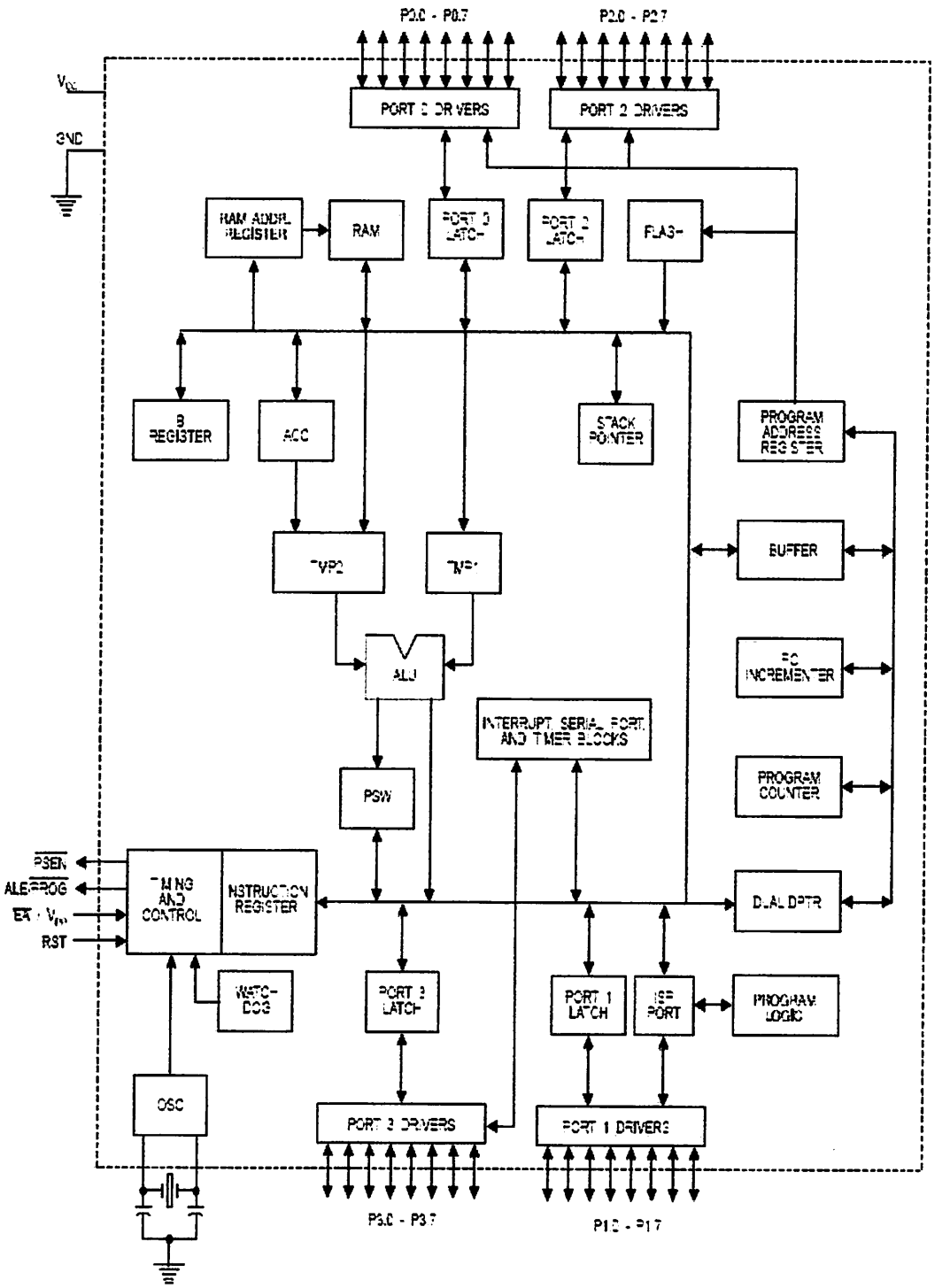
- Port 1, yaitu pin p1.0...p1.7 Port 1 berfungsi sebagai I/O biasa atau menerima *low order address byte* selama pada saat *Flash Programming*. Port ini mempunyai internal pull updan berfungsi sebagai input dengan memberikan logika 1. Sebagai output port ini dapat memberikan *output sink* keempat buah input TTL. Fasilitas khusus dari port 1 ini

adalah adanya *In-System Programming*, yaitu port 1.5 sebagai MOSI, port 1.6 sebagai MISO, port 1.7 sebagai SCK.

- Port 2, yaitu mulai pin p2.0...p2.7 Port 2 berfungsi sebagai I\O biasa atau *high order address*, pada saat mengakses memori secara 16 bit (Movx DPTR). Pada saat mengakses memori secara 8 bit (Mov Rn), port ini akan mengeluarkan sisi dari *Special Function Register*. Port ini mempunyai pull up dan berfungsi sebagai input dengan memberikan logika 1. Sebagai output, port ini dapat memberikan output sink keempat buah input TTL.
- Pin 3.0, sebagai RXD *(Port Serial Input)*.
- Pin 3.1, sebagai TXD *(Port Seial Output)*.
- Pin 3.2, sebagai INT0 *(Port External Interupt 0)*.
- Pin 3.3, sebagai INT1 *(Port External Interupt 1)*.
- Pin 3.4, sebagai T0 *(Port External Timer 0)*.
- Pin 3.5, sebagai T1 *(Port External Timer 1)*.
- Pin 3.6, sebagai WR *(External Data Memory Write Strobe)*.
- Pin 3.7, sebagai RD *(External Data Memory Read Strobe)*.
- Pin 9, sebagai RST Reset akan aktif dengan memberikan input high selama 2 cycle.
- Pin 30, sebagai ALE/PROG Pin ini dapat berfungsi sebagai *Address Latch Enable* (ALE) yang me-*latch low byte* address pada saat mengakses *memori external*. Sedangkan pada saat Flash Programming (PROG) berfungsi sebagai *pulse input*. Pada operasi normal ALE akan mengeluarkan sinyal *clock* sebesar 1/16 *frekwensi oscillator*, kecuali pada saat mengakses *memori external*. Sinyal *clock* pada saat ini dapat pula di *disable* dengan men-set bit 0 *Special Function Register*.

- Pin 29, sebagai PSEN Pin ini berfungsi pada saat mengeksekusi program yang terletak pada *memori eksteranal.* PSEN akan aktif dua kali setiap *cycle.*
- Pin 31, Sebagai EA/VPP Pada kondisi *low*, pin ini akan berfungsi sebagai EA yaitu mikrokontroller akan menjalankan program yang ada pada memori eksternal setelah sistem di *reset.* Jika berkondisi *high*, pin ini akan berfungsi untuk menjalankan program yang ada pada *memori internal.* Pada saat *Flash Programming* pin ini akan mendapat tegangan 12 Volt (VPP).
- Pin 19, sebagai XTALL1 *(Input Oscillator).*
- Pin 18, sebagai XTALL2 *(Output Oscillator).*

Dengan keistimewaan diatas pembuatan alat menggunakan AT89S51 menjadi lebih sederhana dan tidak memerlukan IC pendukung yang banyak. Adapun gambar 2.2 menunjukan blok Diagram dari Mikrokontroler AT89S51 adalah sebagai berikut:



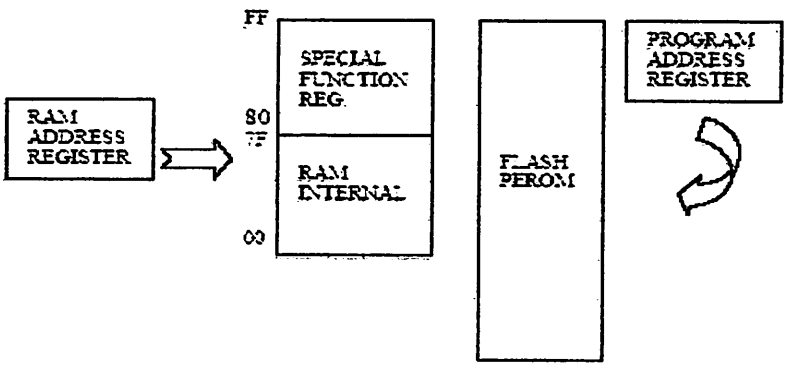
Gambar 2.2. Blok Diagram Mikrokontroler AT89S51

2.2.1. Struktur Memori.

AT89S51 mempunyai stuktur memori yang terdiri atas :

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

Gambar 2.3 menjukkan struktur memori AT89S51.



Gambar 2.3. *Struktur Memori AT89S51*

2.2.2. Organisasi Memory.

Organisasi memori pada mikrokontroler AT89S51 dapat dibagi menjadi dua bagian besar yaitu memori program dan memori data. Pembagian tersebut didasarkan atas fungsi dari penyimpanan data maupun program. *Memori program* digunakan untuk menyimpan instruksi-instruksi yang akan dijalankan oleh mikrokontroler, sedangkan *memori* data digunakan sebagai tempat yang sedang diolah mikrokontroler.

Program mikrokontroler disimpan dalam memori program berupa ROM. Mikrokontroller AT89S51 dilengkapi dengan *ROM internal*, sehingga untuk menyimpan program tidak digunakan *ROM eksternal* yang terpisah dari mikrokontroller. Agar tidak menggunakan *memori program eksternal*, EA (*Eksternal Address enable*) dihubungkan dengan Vcc.

Memori program mikrokontroller menggunakan alamat 16 bit mulai 0000H-FFFFH, sehingga kapasitas penyimpanan program maksimal adalah 4Kb. Sinyal /PSEN (*Program Store Enable*) tidak digunakan jika digunakan *memori program internal.*

Selain memori program mikrokontroller AT89S51 juga memiliki *data internal* 128 *byte* dan mampu mengakses memori data eksternal sebesar 64 Kb. Semua *memori data internal* dapat dialamati dengan data langsung atau tidak langsung. Ciri dari pengalamatan langsung adalah *operand* alamat *register* yang berisi alamat data yang akan diolah. Sebagian memori tersebut dapat dialamati dengan pengalamatan *register*, dan sebagian lagi dapat dialamati dengan memori satu bit. Untuk membaca data digunakan sinyal /RD sedangkan untuk menulis digunakan sinyal /RW.

2.2.3. SFR (*Special Function Register*).

Register fungsi khusus (*Special Function Register*) terletak pada 128 byte bagian atas memori data internal dan berisi register-register untuk pelayanan latch port, timer, program status words, control peripheral dan sebagainya. Alamat register fungsi khusus ditunjukan pada tabel 2.1.

Tabel 2.1 Special Function Register

Simbol	Nama Register	Alamat
ACC	Accumulator	$E0_H$
B	Register B	$F0_H$
PSW	Program Status Word	$D0_H$
SP	Stack Pointer	81_H
DPL	Bit rendah	82_H
DPH	Bit Tinggi	83_H
P0	Port 0	80_H
P1	Port 1	90_H
P2	Port 2	$A0_H$
P3	Port 3	$B0_H$
IP	Interupt Periority Control	$B8_H$
IE	Interupt Enable Control	$A8_H$
TMOD	Timer/Counter Mode Control	89_H
TCON	Timer/Counter Control	88_H
TH0	Timer/Counter 0 High Control	$8C_H$
TL0	Timer/Counter 0 Low Control	$8A_H$
TH1	Timer/Counter 1 High Control	$8D_H$
TL1	Timer/Counter 1 Low Control	$8B_H$
SCON	Serial Control	98_H
SBUF	Serial Data Buffer	99_H
PCON	Power Control	87_H

Beberapa macam register fungsi khusus yang sering digunakan adalah sebagai berikut ini:

- *Accumulator* (ACC) merupakan *register* untuk penambahan dan pengurangan. Perintah *mnemonic* untuk mengakses *akumulator* disederhanakan sebagai A.
- *Register B* merupakan *register* khusus yang berfungsi melayani operasi perkalian dan pembagian.
- *Stack Pointer* (SP) merupakan *register* 8 bit yang dapat diletakkan di alamat mana pun pada RAM internal.
- *Data Pointer* (DPTR) terdiri dari dua *register*, yaitu untuk *byte* tinggi (Data Pointer High, DPH) dan byte rendah (Data Pointer Low, DPL) yang berfungsi untuk mengunci alamat 16 bit.
- *Port 0* sampai *Port 3* merupakan *register* yang berfungsi untuk membaca dan mengeluarkan data pada port 0, 1, 2, 3. Masing-masing register ini dapat dialamati *per-byte* mau pun per-bit.
- *Control Register* terdiri dari *register* yang mempunyai fungsi kontrol. Untuk mengontrol sistem interupsi, terdapat dua *register* khusus, yaitu *register* IP (*Interupt Priority*) dan register IE (*Interupt Enable*). Untuk mengontrol pelayanan *timer/counter* terdapat *register* khusus, yaitu register TCON (*timer/counter control*) serta pelayanan port serial menggunakan register SCON (*Serial Port Control*).

- **2.2.4. Sistem Interupsi.**

Mikrokontroller AT89S51 mempunyai 5 buah sumber interupsi yang dapat membangkitkan permintaan *interupsi*, yaitu INT0, INT1, T1, T2 dan Port Serial. Saat

terjadinya *interupsi* mikrokontroller secara otomatis akan menuju ke sub rutin pada alamat tersebut. Setelah interupsi selesai dikerjakan, mikrokontroller akan mengerjakan program semula. Tiap-tiap sumber *interupsi* dapat *enable* atau *disable* secara *software.*

Tingkat prioritas semua sumber *interupt* dapat diprogram sendiri-sendiri dengan *set* atau *clear* bit pada (*Interupt Priority*). Jika dua permintaan *interusi* dengan tingkat prioritas yang berbeda diterima secara bersamaan, permintaan *interupsi* dengan prioritas tertinggi yang akan dilayani. Jika permintaan *interupsi* dengan *prioritas* yang sama diterima bersamaan, akan dilakukan polling untuk menentukan mana yang akan dilayani. *Bit-bit* pada IP adalah sebagai berikut:

–	–	–	**PS**	**PT1**	**PX1**	**PT0**	**PX0**

Priority bit = 1 menandakan prioritas tinggi

Priority bit = 0 menandakan prioritas rendah

Simbol	**Posisi**	**Fungsi**
–	IP.7	Kosong
–	IP.6	Kosong
–	IP.5	Kosong
PS	IP.4	Bit prioritas interupsi port serial
PT1	IP.3	Bit prioritas interupsi Timer 1
PX1	IP.2	Bit prioritas interupsi
PT0	IP.1	Bit prioritas interupsi Timer 0
PX0	IP.0	Bit prioritas interupsi

Tabel 2.2 Tingkatan Prioritas Interupsi.

Prioritas Interupsi	Sumber Interupsi	Alamat Vektor
1	IE0 (Interupsi eksternal 0)	0003_H
2	TF0 (timer overflow flag 0)	$000B_H$
3	IE1 (interupsi eksternal 1)	0013_H
4	TF1 (timer overflow flag 1)	$001B_H$
5	R1 dan T1	0023_H

2.3 Komunikasi Data Serial

Komunikasi *data serial* sangat berbeda dengan format pemindahan *data pararel.* Disini pengiriman *bit-bit* tidak dilakukan sekaligus seperti pada saluran *pararel,* tetapi setiap *bit* dikirimkan satu persatu melalui saluran tunggal. Dalam pengiriman data secara *serial* harus ada *sinkronisasi* atau penyesuaian antara pengirim dan penerima agar data yang dikirimkan dapat diterima dengan tepat dan benar oleh penerima. Dalam komunikasi secara serial terdapat tiga macam mode *transmisi serial* dalam *mentransmisikan bit-bit* data yaitu : *syncronous, asynchronous da isochronous.* Dalam proyek akhir ini hanya membahas *mode transmisi asyncronous* saja yang digunakan. *Transmisi serial mode asyncronous* digunakan bila pengiriman data dilakukan satu karakter tiap pengiriman. Antara satu karakter dengan karakter lainnya tidak ada waktu

antara yang tetap. Karakter dapat dikirimkan sekaligus atau pun beberapa karakter kemudian berhenti unutk waktu yang tidak tentu, kemudian dikirimkan sisanya. Dengan demikian *bit bit* data ini dikirimkan dengan periode yang acak sehingga pada sisi penerima data akan diterima kapan saja. Ada pun sinkronisasi yang terjadi pada *transmisi serial asinchronous* adalah dengan memberikan *bit bit* penanda awal dari data dan penanda akhir dari data pada sisi pengirim maupun dari sisi penerima.

2.3.1. Metode Komunikasi

Metode yang dimaksud adalah:

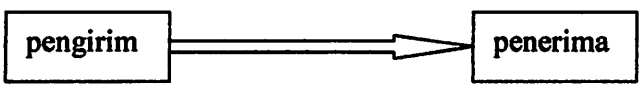
a. Mode Komunikasi *Simplex.*

b. Mode Komunikasi *Half Duplex* .

c. Mode Komunukasi *Full Duplex.*

2.3.1.1 Mode Komunikasi *Simplex*

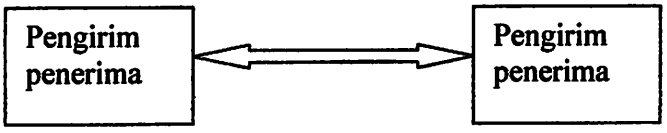
Pada mode komunikasi *simplex* ini jalur komunikasi yang diberikan adalah satu arah seperti yang ditunjukkan gambar dibawah. Dalam komunikasi pengirim dan penerima adalah permanen, dimana pada pihak pengirim adalah selalu mengirim data, dan pihak penerima adalah selalu menerima data. Gambar 2.4 menunjukkan hubungan *simplex.*



Gambar 2.4. *Hubungan Simplex*

2.3.1.2 Mode Komunikasi *Half Duplex*

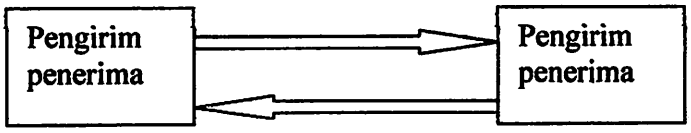
Pada mode komunikasi *half-duplex* ini komunikasi data dilakukan dalam dua arah dimana pada masing masing pihak bisa sebagai penerima atau sebagai pengirim data seperti yang ditunjukkan pada gambar 2.5. Namun pada mode ini tidak dapat dilakukan secara bersamaan tetapi secara bergantian atau disebut *two way alternative*. Jika salah satu sebagai pengirim maka yang lain adalah sebagai penerima demikian sebaliknya. Gambar 2.5 menunjukan hubungan *half-duplex*.



Gambar 2.5. *Hubungan Half-duplex*

2.3.1.3 Mode komunikasi *Full-Duplex*

Mode komunikasi *full duplex* dilakukan dalam dua arah yang terjadi secara bersamaan seperti ditunjukkan pada gambar 2.6.



Gambar 2.6. *Hubungan Full-Duplex*

2.3.2 Format data *komunikasi serial*

Dalam teknik komunikasi dara serial dikenal istilah format data serial. Format data serial ini terdiri dari parameter - parameter yang dipakai untuk menentukan bentuk

data serial yang akan dikomunikasikan. Ada beberapa macam format data serial yang dapat digunakan, dimana elemen elemennya terdiri dari :

a) Kecepatan mobilisasi data per bit (*baut rate*).

b) Jumlah bit data per karakter (*data length*).

c) Pariti yang digunakan.

d) Jumlah *stop bit* dan *start bit*.

2.3.2.1 Kecepatan *mobilisasi* data *per bit*

Laju data serial sering kali dinyatakan dalam satuan *baud*. Laju *baud* dalam kanal komunikasi merupakan laju tercepat dari perpindahan bit. Laju pemindahan bit kanal jaringan biasanya lebih rendah dari laju *baud*. Keterlambatan tersebut karena bit ekstra ditambahkan untuk keperluan pewaktuan. Dalam system kecepatan tinggi, pemindahan juga diperlambat oleh tundaan aktif proses pengolahan.

Data serial dapat dimobilisasikan pada berbagai *baud rate*. *Baud rate* yang digunakan dalam teknik komunikasi data serial *null modem asycronous* adalah 300 bps sampai 19200 bps.

2.3.2.2 Jumlah bit data per karakter

Dalam komunikasi data serial mode asycronous biasa berlangsung transmisi data yang dikemas dalam bentuk karakter. Dalam satu karakter diperbolehkan terdiri dari beberapa variasi jumlah bit. Dari sekian variasi yang diperbolehkan diantarannya adalah terdiri dari 7 bit dan 8 bit (panjang data karakternya saja). Kedua variasi ini adalah yang paling sering digunakan dalam komunikasi serial. Tetapi meskipun demikian tidak

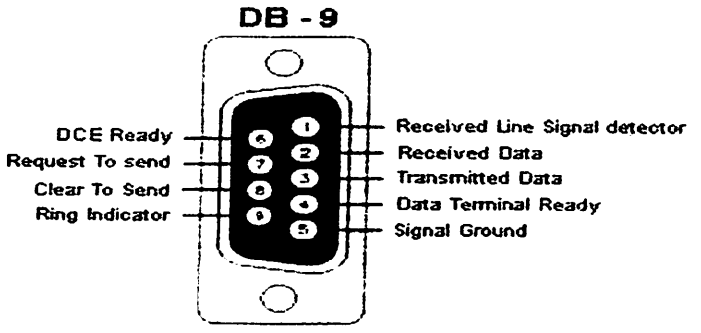
menutup kemungkinan jika diinginkan untuk menggunakan variasi jumlah bit yang lain asalkan masih diperkenakan dalam penulisan inisialisasi peralatan serial yang bersangkutan dalam komunikasi serial. Tetapi meskipun demikian tidak menutup kemungkinan jika diinginkan untuk menggunakaan variasi jumlah bit yang lain asalkan masih diperkenakan dalam penulisan inisialisasi peralatan serial yang bersangkutan.

2.3.2.3 *Parity bit*

Bit parity adalah bit yang digunakan sebagai alat pemeriksaan kesalahan sederhana dalam proses transmisi data digital. Pemakaian parity ini akan diikut sertakan dalam proses penulisan inisialisasi peralatan serial yang bersangkutan. bit parity ini akan diletakan setelah susunan bit data. Kemungkinan dari jenis ini ada tiga macam ,yaitu parity ganjil, parity genap dan tanpa parity (tidak diikutkan dalam pemeriksaan kesalahan) level digital .

2.3.2.4 Konfigurasi *port serial*

Gambar 2.7 menunjukkan *konektor port serial DB-9* pada bagian belakang CPU. Pada computer IBM PC kompatibel biasanya kita dapat menemukan dua konektor port serial DB-9 yang biasa dinamai COM 1 dan COM 2.



Gambar 2.7. *Konektor serial DB-9* pada bagian belakang CPU

Tabel 2.3 Konfigurasi pin dan nama sinyal konektor serial

Nomor Pin	Nama Sinyal	Direction	Keterangan
1	DCD	In	Data Carrier Detect / Received Line Sinyal Detect
2	RxD	In	Receive Data
3	TxD	Out	Transmit Data
4	DTR	Out	Data Terminal Ready
5	GND	-	Ground
6	DSR	In	Data Set Ready
7	RST	Out	Request to Send
8	CTS	In	Clear to Send
9	RI	In	Ring Indicator

Keterangan mengenai fungsi saluran RS-232 pada *konektor DB-9* adalah sebagai berikut:

- *Received Line Signa detect,* dengan saluran ini DCE membertahukan ke DTE bahwa pada terminal masukan ada data masukan.
- *Received data,* digunakan DTE pada saat menerima data dari DCE.
- *Transmite data,* digunakan DTE pada saat mengirim data ke DCE.
- *Data terminal ready,* pada saluran ini DTE memberitahukan kesiapan terminalnya.

- *Signal groud,* merupakan saluran *groud.*
- *Ring indicator,* pada saluran ini DCE memberitahu ke DTE bahwa sebuah stasiun menghendaki hubungan dengannya.
- *Clear to send,* dengan saluran ini DCE memberitahukan bahwa DTE sudah dapat memulai pengiriman data.
- *Request to send,*dengan saluran ini DCE diminta mengirim data oleh DTE.
- *DCE ready,*merupakan sinyal aktif yang menunjukan bahwa DCE sudah siap.

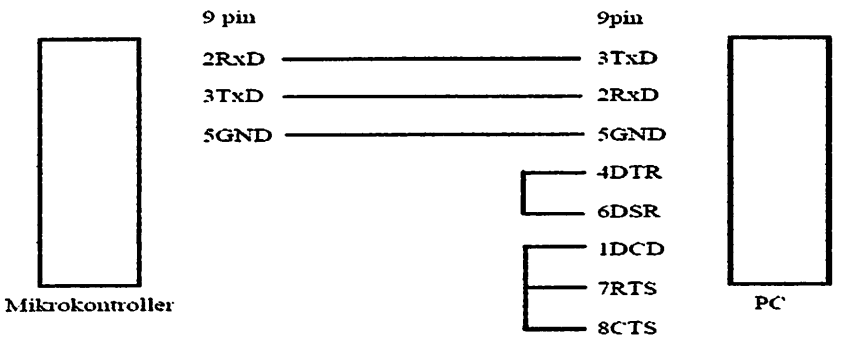
2.3.3. Komunikasi Serial Pada Mikrokontroller

Komunikasi antar PC dengan mikrokontroller adalah komunikasi serial dimana menggunakan RS-232 sebagai *konektor interfacenya.*

Pada Mikrokontroller terdapat parameter-parameter *setup window* yang harus diset sesuai dengan parameter *MSCom.* Parameter-parameter tersebut antara lain:

1. *Transmission speed* : pilih 2400.
2. *Data length* : pilih 8 bit sebagai bit datanaya.
3. *Stop bit* : pilih 1.
4. *Parity check* : pilih NONE.
5. *Flow control* : pilih NONE.

Untuk melakukan pengiriman data ke Mikrokontroller maka untuk RS-232 sebagai konektor interfacenya hanya menggunakan 3 pin saja. Gambar 2.8 koneksi RS-232 pada mikrokontroller dan PC.



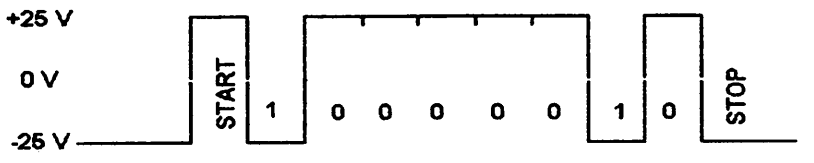
Gambar 2.8. *Koneksi RS-232* pada mikrokontroller dan PC

2.3.4. Konektor Interface RS-232.

Karakteristik dari RS-232 memiliki ketentuan level tegangan sebagai berikut:

1. Logika '1' disebut 'mark'terletak antara -3 Volt hingga -25 Volt.
2. Logika '0' disebut 'space'terletak antara +3 Volt hingga +25 Volt.
3. Daerah tegangan antara -3 Volt hingga +3 Volt adalah invalid level,yaitu daerah tegangan yang tidak memiliki level logika pasti sehingga harus dihindari. Demikian juga, level tegangan lebih negatif dari -25 Volt atau lebih positif dari +25 volt juga harus dihindari karena tegangan tersebut dapat merusak line driver pada saluran RS-232.

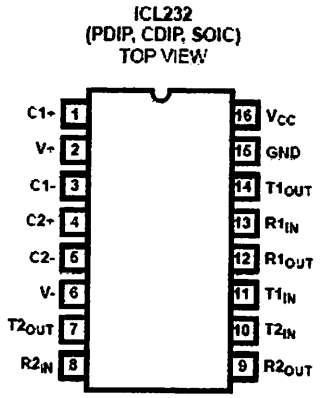
Gambar 2.9 berikut ini adalah contoh level tegangan RS-232 pada pengiriman huruf 'A'dalam format ASCII tanpa bit paritas.



Gambar 2.9. *Level tegangan RS-232* pada pengiriman huruf 'A'
tanpa bit paritas

2.3.4.1. *IC Serial MAX 232*

Komunikasi serial RS-232 merupakan komunikator yang menghubungkan antara terminal data dari suatu peralatan dan peralatan ini menjalankan pertukaran data biner secara serial. Sedangkan IC yang dipakai untuk komunikasi juga menyediakan pemrosesan data dan protocol, sedang yang laen berupa interface ke jalur komunikasi secara fisik. Bagian yang menangani komunikasi dapat dihubungkan dengan berbagai palikasi yang berhubungan dengan elektronik, tetapi memiliki kondisi arus dan tegangan yang tak menentu. IC serial RS 232 dipakai sebagai interface (antar muka) dari PC ke perangkat luar (level TTL) atau sebaliknya dari perangkat luar ke PC. Tegangan yang ada pada RS 232 berbeda dengan level tegangan digital. Tegangan RS 232 tersebut antara +3 volt sampai +15 volt untuk logoka "0" dan -3 sampai dengan -15 volt untuk logika "1". Tegangan yang cukup tinggi ini mengakibatkan data dapat ditransmisikan cukup jauh. Gambar 2.10 menunjukan *Interface MAX 232.*



Gambar 2.10 *Interface MAX 232*

IC MAX 232 mempunyai 16 kaki dengan supplay tegangan sebesar 5 volt. Kaki ke-16 digunakan sebagai input tegangan (Vcc), kaki ke-15 sebagai *ground* (GND). Kaki 8 dan 13 digunakan sebagai input RS-232. RS-232 merupakan suatu interface yang

digunakan untuk menghubungkan antar terminal data dari suatu peralatan dan peralatan komunikasi data yang menjalankan pertukaran data biner secara serial.

2.3.5. RS-485/RS-422 *Transceiver*

Penggerak komunikasi RS-485/RS-422 adalah pembantu komunikasi serial seperti halnya RS-232 yang sudah dipakai dalam transfer data serial sekarang ini. Perbedaan antara keduanya terletak pada panjang saluran yang dipakai, maksimum bit rate, level tegangan data "1" dan "0", kebocoran daya dan masukaan penerima, mengenai perbedaan tersebut.

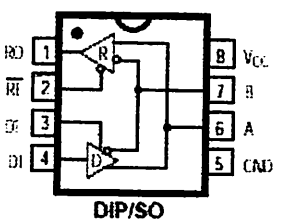
Panjang saluran yang di maksud adalah batas panjangnya saluran yang digunakan antara satu driver RS-485 ke driver RS-483 yang lain yang masih diperbolehkan, ini berkaitan dengan kualitas data yang diterima driver penerima. Tabel 2.4 menunjukkan karakteristik beberapa tipe komunikasi serial.

Table 2.4 karakteristik beberapa tipe komunikasi serial

Karekteristik	RS 232	RS 485	RS 423
Saluran panjang	100ft		
Maksimum bit/dctik	2×10^4	10^5	10^6
Data "1"=making	-1,5 - -36v	$V_A > V_B$	V_A = -
Data "0"=spancing	+1,5v +3,6v		
Hubungan pendek	100	100	100
Kebocoran bila daya dimatikan,maksimum tegangan yang diberikan pada yang tak diberi daya	300	100	100
Masukan penerima	1,5 Ujung tunggal	100*m*v diferensial	100*m*v diferensial

Sumber : zaks D, 1987

Untuk menampilkan data biner dibutuhkan dua besaran tegangan (V_A dan V_B). Pada RS-485/RS-422 kondisi "1" atau mark dinyatakan dengan membuat tegangan saluran B lebih besar dari saluran A(V_A dan V_B), sedangkan kondisi "0" atau space dinyatakan dengan membuat saluran A lebih besar dari saluran B($V_A > V_B$). Syaratnya tegangan antara dua saluran tersebut ($V_{A\,B}$) harus lebih besar dari 0,4 V dan lebih baik kecil dari 12 V. keuntungan transmisi *sinyal diferensial* adalah berkurangnya noise dalam satu saluran sinyal, akan berkurang secara sama pada saluran sinyal yang lain [Douglas V.Hall ,1992]. Gambar 2.11 menunjukan interface MAX 485.



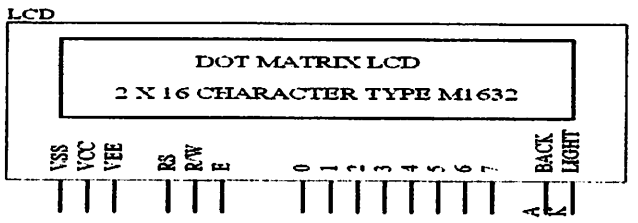
Gambar 2.11 Interface MAX 485

2.5. LCD (*Liquid Crytal Display*)

Liquid Crystal Display adalah modul tampilan yang mempunyai konsumsi daya yang relatif rendah dan terdapat sebuah *kontroller CMOS* didalamnya. kontroller tersebut sebagai pembangkit *ROM/RAM* dan *display data RAM*. Semua fungsi tampilan dikontrol oleh suatu instruksi,sehingga modul LCD dapat dengan mudah diinterfacekan dengan MPU. Ciri-ciri dari LCD M1632:

- 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 berserta konfigurasinya:



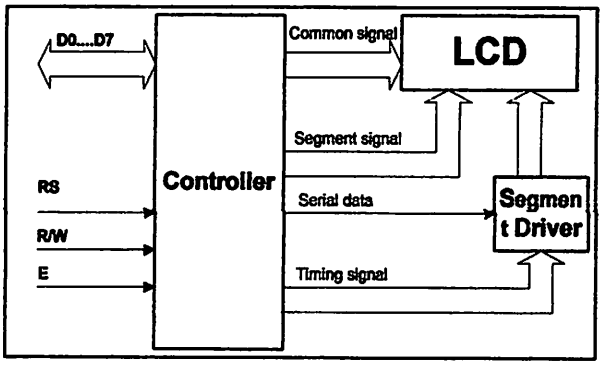
Gambar 2.12 Pin Pada LCD

LCD ini mempunyai 16 pin yang dihubungkan dengan perangkat keras *prosessor* penunjang. Adapun fungsi dari masing-masing pin, ditunjukkan dalam tabel 2.5 berikut:

Tabel 2.5 Fungsi Pin LCD M1632

No	Nama Penyemat	Fungsi
1	Vss	Terminal ground
2	Vcc	Tegangan catu +5 volt
3	Vee	Drive LCD
4	RS	Sinyal pemilih register 0: Instruksi register (tulis) 1: Data Register (tulis dan baca)
5	R/W	Sinyal seleksi tulis atau baca 0: Tulis 1: Baca
6	E	Sinyal operasi awal, sinyal ini mengaktifkan data tulis dan baca
7 – 14	DB0-DB7	Merupakan saluran data, berisi perintah dan data.
15	V+ BL	Pengendali kecerahan latar belakang LCD 4 - 4,42 V dan 50 – 500 mA
16	V-BL	Pengendali kecerahan latar belakang LCD 0 V

Masukan yang diperlukan untuk mengendalikan modul berupa bus data yang masih *termutiplek* dengan bus alamat serta 3 bit sinyal kontrol. Sementara pengendalian LCD dilakukan secara *internal* oleh *kontroler* yang sudah terpasang dalam modul LCD. Diagram blok untuk LCD dapat dilihat dalam Gambar 2.13.



Gambar 2.13 Diagram Blok LCD M1632

2.6. *Keypad* 4x4

Teknik yang sering digunakan dalam perancangan *keypad* adalah teknik *multiplexing* empat buah jalur baris dan empat jalur kolom. Bila baris dan kolom ini disidangkan maka akan terbentuk titik-titik potong yang membentuk *matriks* 4x4. seperti pada Gambar 2.13.

1	2	3	COR
4	5	6	MEN
7	8	9	↑
CAN	0	ENT	↓

Gambar 2.14 Penampang Dasar *Keypad*

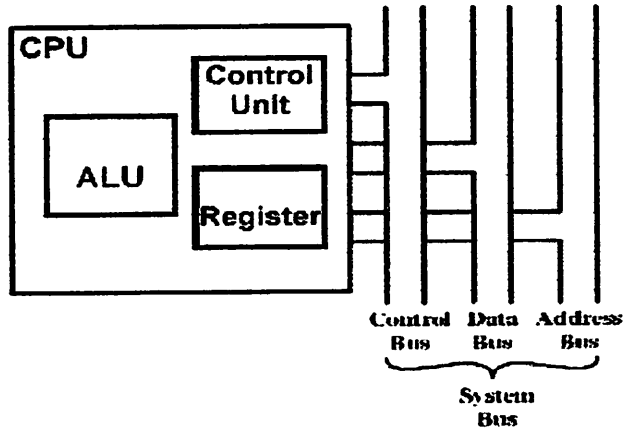
Rangkaian ini dapat dianalogikan dengan empat buah kabel terbuka yang disilangkan dengan empat kabel terbuka lainnya. Bila pada suatu titik, kabel yang disilangkan itu disentuhkan maka diasumsikan bahwa tombol *keypad* pada posisi yang-bersilangan tersebut ditekan. Untuk mengidentifikasikan letak atau posisi tombol yang

ditekan itu, bahwa susunan *matrix keypad* itu membetuk koordinat (x,y) dalam dua dimensi, informasi posisi yang diperlukan adalah informasi tentang nilai x dan y.

2.7. Personal Computer

2.7.1. Struktur dan fungsi CPU.

Mengerti dan fungsi CPU yaitu dapat melakukan *fech instruksi, interperenter intruksi, fech data, exusidata,* dan penyimpanan kembali. Mengerti aliran data pada siklus tak langsung, siklus *interupt*. Mengerti *pipeling* dan mengerti menangani percabangan pada *pipeling*. Agar dapat menjalakan tugas CPU, CPU harus menyimpan data untuk sementara waktu. CPU harus mengiat lokasi *instruksi* terakhir sehingga CPU dapat mengambil instruksi berikutnya. CPU perlu menyimpan instruksi dan data untuk sementara waktu pada saat instruksi sedang dieksekusi. CPU memerlukan memori internal berukuran kecil yang dikenal dengan *register*. Blok diagram CPU sebagai berikut.



Gambar 2.15 Blok diagram CPU

ALU

- Melakukan *komputasi/* pengolahan data berdasarkan *instruksi* yang diberikan padanya.
- Komponen – komponen utama CPU
    1. Arithmetik dan logic unit.
    2. Register.
    3. Control Unit (CU).
- Contol unit
    1. Mengontrol perpindahan data dan *instruksi* ke CPU atau dari CPU dan pengontrol operasi ALU.
    2. Selain itu menunjukkan memori internal minimum yang terdiri dari beberapa lokasi penyimpanan yang disebut *register*.

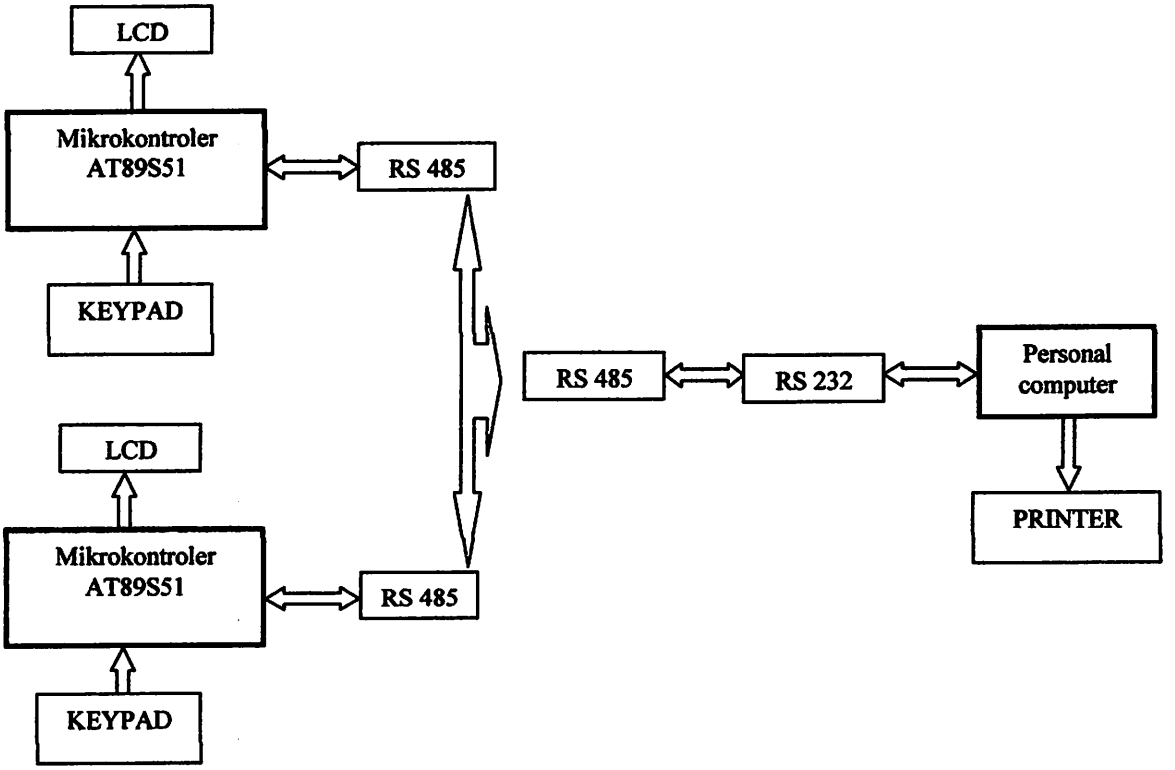
BAB III

PERENCANAAN DAN PEMBUATAN ALAT

Bab ini akan membahas tentang perencanaan dan perancangan alat yang meliputi perencanaan perangkat keras (*Hardware*) dan perangkat lunak (*Software*) mikrokontroler. Perancangan secara keseluruhan dapat dibagi menjadi dua bagian,yaitu:

1. Perancangan Perangkat Keras (*Hardware*).
2. Perancangan Perangkat Lunak (*Software*).

3.1. Perancangan Perangkat Keras (*Hardware*).



Gambar 3.1 Diagram Blok Sistem.

Keterangan dari diagram blok :

- *Keypad*

  Keypad yang digunakan berupa keypad matrik 4 x 4. Berfungsi sebagai inputan data pada mikrokontroller. Inputan yang dimasukan berupa jumlah menu yang akan dipesan.

- Mikrokontroler

  Sebagai pengolah semua data dan inputan dari sensor dan keypad kemudian menampilkannya pada LCD. Mikrokontroller berfungsi sebagai pengontrol semua sistem, dalam IC ini terdapat 4 *Kbyte EEPROM*, 128 *byte RAM internal*, 32 *Programmable I/O lines*, 4 Kb Flash memory dan 2 buah *timer / counter* 16 bit.

- RS-485

  RS-485 digunakan sebagai converter antara meja klien 1 dengan meja klien 2.

- RS-232

  RS-232 sebagai konverter antara RS-485 dengan komputer.

- Printer

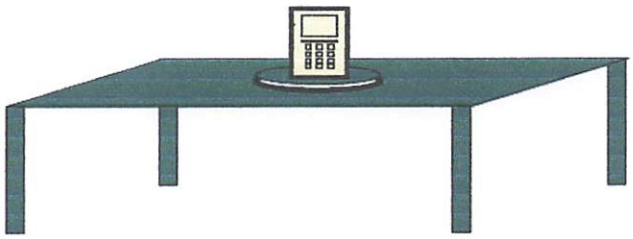
  Digunakan untuk mencetak masukan dari komputer.

  Dari gambar blok diagram 3.1 dapat dijelaskan cara kerjanya sebagai berikut:

Prinsip kerja:

Sistem alat ini bekerja pada saat kita menekan tombol menu pada keypad maka LCD akan menampilkan menu makanan yang sudah diprogam oleh mikro kita tinggal memilih menu apa yang kita inginkan dan berapa jumlah makanan atau minuman yang kita pesan. setelah selesai kita memilih menu yang kita pesan kita tekan tombol enter, setelah itu tekan tombol exit. maka mikro akan memberikan konfirmasi kepada pemesan

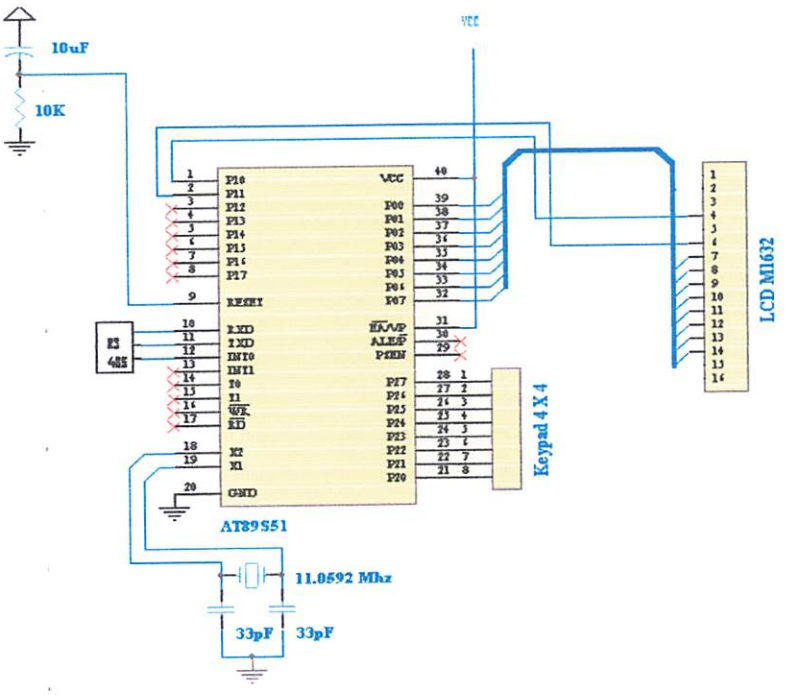
yang ditampilkan oleh LCD. Jika kita memilih ya maka mikro akan menirimkan data. jika kita tekan tombol tidak maka mikro tidak akan mengirim data tersebut. Data yang dikirim diterima oleh komputer yang berada dimeja kasir restoran. Data tersebut diproses untuk dicetak, data tersebut dicetak oleh printer. Adapun miniatur tata peletakan alat pada meja restaurant.



Gambar 3.2. Miniatur alat pada meja.

3.2. Mikrokontroller AT89S51

Dalam perancangan dan pembuatan rangkain ini menggunakan mikrokontroller AT89S51 sebagai pengontrol masukan dan keluaran. IC ini memiliki fasilitas EEPROM sebagai memori program sebagai dapat mempermudah dalam proses perencanaan sistem alat ini. Adapun aplikasi mikrokontroller AT89S51 dalam perancangan sebagai berikut:



Gambar 3.3. Menunjukan masing – masing port.

Agar sebuah mikrokontroller dapat bekerja sebagai pengontrol, maka kaki – kaki/ port mikrokontroller dihubungkan pada rangkaian – rangkaian *eksternal.* Pada perancangan ini dipaparkan port apa saja yang akan digunakan pada mikrokontroller AT89S51 yaitu:

1. P0 (P0.0 sampai P0.7) sebagai out put untuk tampilan LCD.

   P2 (P2.0 sampai P2.7) sebagai input untuk tombol.

   P1 (P1.0) digunakan untuk *enable* sedangkan untuk P1 (P1.1) digunakan untuk RST LCD.

2. Kaki pin 9 (RST) dihubungkan kerangkaian *reset* untuk me*reset hardware* baik secara sofware atau pun manual, maka pin RST ini berlogika '1' (aktif high).

3. Kaki pin 18 (XTAL 2) dan kaki pin (XTAL 1) dihubungkan kerangkaian osilator crystal untuk memberikan clock inputan yang dibutuhkan oleh mikrokontroller AT89S51.
4. Kaki pin 20 (GND) dihubungkan ke ground.
5. Kaki pin 40 (VCC) dihubungkan kesumber tegangan 5V DC.
6. Kaki pin 11 (TXD) digunakan untuk mengirim data ke PC.

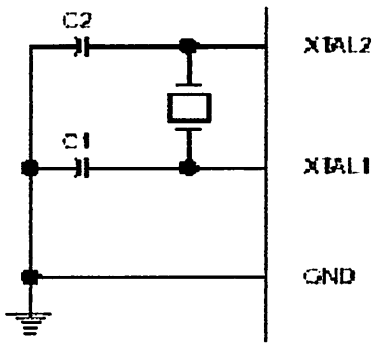
3.2.1. Rangkaian Clock Minimun Sistem.

Mikrokontroller AT89S51 ini memiliki internal clock generator yang berfungsi sebagai sumber *clock* yang di perlukan. Rangkaian ini terdiri dari dua buah *kapasitor* dan sebuah *kristal*, dengan ketentuan :

C1 dan C2 = 30 pF ±10 pF untuk *kristal*

= 40 pF ±10 pF untuk keramik *resonators*

Dalam minimun sistem ini menggunakan kristal 11,0592 Mhz dan C1 = C2 sebesar 33 pF. Dengan rangkaian sebagai berikut :



Gambar 3.4. Rangkaian Clock

Dengan menggunakan nilai kristal diatas maka dapat dihitung waktu yang diperlukan untuk satu siklus mesin.

Diketahui : F = 11 Mhz

Sehingga : $Tme = \frac{Cx12}{f_{kristal}}$

$$Tme = \frac{1x12}{11Khz} = \frac{12}{11Khz}$$

$Tme = 1{,}09\ \mu S$

Maka satu siklus mesin dari mikrokontroller AT89S51 adalah sebesar *Tme* = 1,09 µS.

Kecepatan proses dilakukan oleh mikrokontroller ditentukan oleh sumber *clock* (pewaktuan) yang mengendalikan mikrokontroller tersebut. Sistem yang dirancang ini akan menggunakan osilator internal yang sudah tersedia didalam chip AT89S51. untuk menentukan *frekuensi* osilatornya, cukup dengan cara menghubungkan kristal pada pin XTAL 1 dan XTAL 2 serta dua buah kapasitor ke *ground.* Besar crystal disesuaikan dengan kecepatan yang diharapkan untuk transfer data melalui pin serial interface AT89S51 tersebut. Sistem ini dirancang untuk memiliki kemampuan *baud rate* sebesar 2400 bps, sehingga dipilih crystal dengan nilai 12 MHZ.

3.2.2. Rangkaian Reset.

Rangkaian *reset* dalam mikrokontroller AT89S51 akan melakukan *reset* setelah catu daya dihidupkan. Pada saat kondisi *reset* maka faktor reset pada alamat 0000H akan dituju oleh mikrokontroller AT89S51 (dalam hal ini program counter) agar program yang terdapat didalam mikrokontroller kembali ke kondisi semula atau dengan kata lain mikrokontroller mengakses awal dari program yang telah diisi didalamnya. Didalam *reset*

ini akan mengunakan beberapa macam cara untuk mereset miktrokontroller AT89S51. Cara pertama mengunakan *switch* (manual), dimana *user* yang akan mengoperasikan *switch* ini. Cara ke dua mengunakan capasitor 47 uF, dimana *kapasitor* tersebut akan berkondisi aktif *high* selama beberapa detik.

Besarnya nilai tahanan dan kapasitor pada rangkaian *reset* akan menentukan lamanya waktu pulsa *reset*. Sedangkan untuk mencari *frekuensi* dari *reset* tersebut dengan mengunakan rumus sebagai berikut:

$$Fo = \frac{1}{1{,}1 * R_1 * C_1}$$

Sehingga dengan komponen *resistor* dengan nilai 10 KΩ dan *kapasitor* dengan nilai 10 µF akan dihasilkan *frekuensi*:

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

$$F_0 = 90{,}9Hz$$

Maka periode *clock* $= \frac{1}{F}$

$$T = \frac{1}{90.9} = 0{,}01S$$

Dengan frekuensi yang dihasilkan serta periode *clock* yang telah diketahui maka akan memberikan siklus ke mikrokontroler untuk bekerja terus sampai tombol *reset* tersebut ditekan.

Karena crystal yang digunakan mempunyai *frekuensi* sebesar 11,0595 MHZ maka satu periode membutuhkan waktu sebesar:

$$T = \frac{1}{fXTAL} = \frac{1}{11{,}0592MHZ} = 9{,}042 \times 10^{-8}$$

Sehinga waktu minimal logika yang dibutuhkan untuk m*ereset* mikrokontroller adalah:

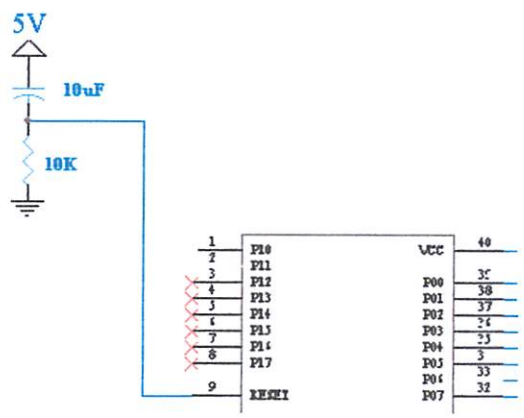
Reset (min) = $T \times$ periode yang digunakan

$$=9{,}042 \times 10^{-8} \times 24 = 2{,}17\ \mu s$$

Jika mikrokontroller membutuhkan waktu minimal $2{,}17\ \mu s$ untuk m*ereset*. Waktu minimal inilah yang dijadikan pedoman untuk menentukan R dan C dengan menentukan nilai R = 10 k dan C =$10\ \mu F$ maka.

$t = 0{,}357 R.C = 0{,}357 \times 10 \times 10^{3} \times 10 \times 10^{-6} = 35{,}7\ \mu s$

Jadi dengan nilai komponen R=10k dan C =$10\ \mu F$ dapat memenuhi syarat minimal untuk waktu yang dibutuhkan oleh mikrokontroller. Gambar 3.5 menunjukaan rangkaian *reset* :



Gambar 3.5. Rangkaian Reset.

3.2.3. Rangkaian LCD M1632 (*Liquid Crystal Display*).

LCD diperlukan untuk menampilkan nilai karakter input yang akan diproses dan data karakter output dari hasil pengukuran supaya hasil proses dan pengukuran bisa dipahami oleh manusia. Rangkaian LCD ini dalam pengoperasiannya memerlukan 8 bit

data dan 3 bit kontrol. Bagian utama dari rangkaian ini adalah penampil karakter LCD 16 x 2 baris. RS (*Register Select*) dan *Enable* pada pin 4 dan pin 6 yang merupakan kontrol dari LCD. Saluran data (*data bus*) dihubungkan ke port 0 mikrokontroller. Untuk pin R/W akan berlogika low (0) apabila dihubungkan dengan ground maka LCD difungsikan hanya untuk menuliskan program atau data ke display. Untuk mengambil data dari mikrokontroller maka pin-pin data dihubungkan dengan port dari mikrokontroller.

Pada lembaran *data sheet* modul LCD M1632 *SEIKO INSTRUMENT INC* disebutkan bahwa:

Power supply LCD meliputi :

Vss = 0 V

Vcc = 5 V ± 5% (2mA)

Power supply back light :

V + BL = 4 - 4,2 V (50 sampai 200 mA)

V BL = 0 V (GND)

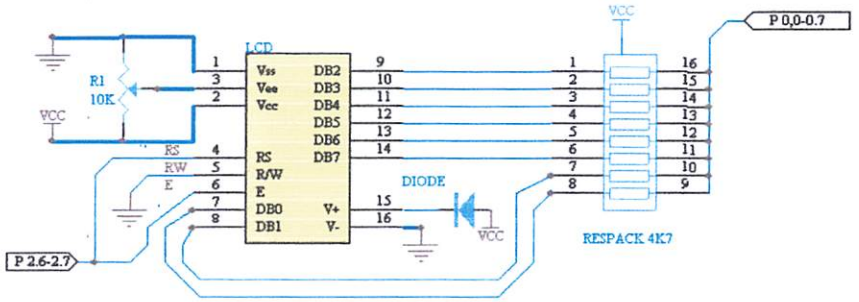
Pada input V + BL dipasang sebuah dioda 1N4002 (bahan *silicon* dengan Vd = 0,65 V sampai 0,7 V). tujuannya adalah didapatkan tegangan V + BL sebesar 4 - 4,2 V dengan perhitungan sebagai berikut :

$$Vcc = Vd + (V + BL)$$

$$5 = 0,7 + (V + BL)$$

$$(V + BL) = 5 - 0,7 = 4,3 \text{ Volt.}$$

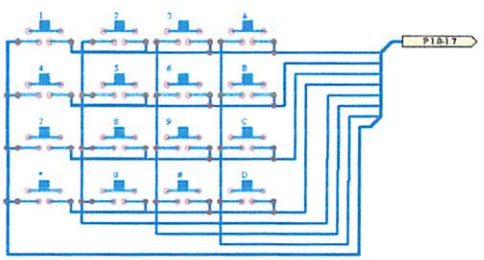
Dipilih *dioda* 1N4002 karena arus maksimum yang biasa dilewatkan oleh *dioda* ini sebesar 1 A, berikut ini adalah rangkaian lengkap modul LCD yang digunakan dalam perancangan. Gambar 3.6 menunjukaan rangkaian LCD :



Gambar 3.6. Rangkaian LCD

3.2.4. Rangkaian *Keypad.*

Keypad yang digunakan adalah *keypad* matriks 4x4. Port yang digunakan untuk sinyal port 1.4 – port 1.7 dari mikrokontroller masuk ke kelompok baris *keypad*, sedangkan kelompok kolom *keypad* dihubungkan ke port 1.0 – port 1.3 mikrokontroller. Untuk fungsi dari tombol-tombol *keypad* tergantung pada pemrogram. Berikut blok diagram dari penyambungan keypad ke mikrokontroller.



Gambar 3.7. Blok Diagram Hubungan *Keypad* Dengan Mikrokontroller.

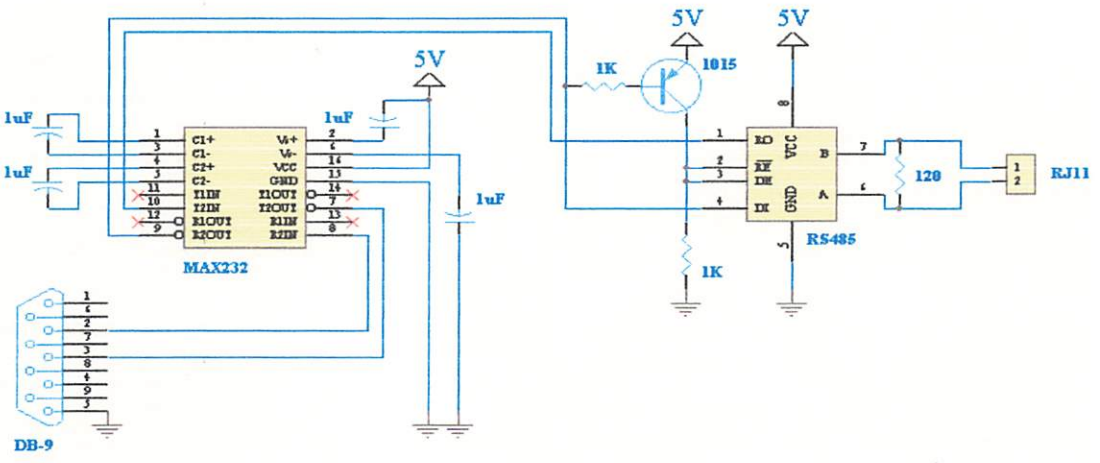
Apabila terjadi penekanan tombol maka data yang dihasilkan dalam bentuk *hexadesimal* akan diterjemahkan oleh mikrokontroller menjadi desimal. Dari kombinasi penekanan tersebut menghasilkan 16 tombol atau kemungkinan yang akan terbentuk

karakter angka dan simbol. Teknik pembacaan pada *keypad* ini, yaitu model *scaning* 4 jalur baris dan 4 jalur kolom. Bila baris dan kolom ini disilangkan maka akan terbentuk titik-titik potong yang membentuk *matrik* 4x 4.

3.2.5. Perencanaan dan Pembuataan Rangkaian Konverter Saluran Transmisi RS-485

Dalam jaringan yang direncanakan terdapat dua buah titik *client* dan MCU bagian meja pembeli dan titik server berupa MCU bagian operator yang bertindak sebagai kasir. Pada setiap titik mikrokontroller, *port* komunikasi serial digunakan bekerja pada level tegangan TTL sehingga dapat langsung dihubungkan ke*port*/ masukan dan keluaran TTL dari *transiver* RS-485.

Rangkaian antaramuka kesaluran transmisi RS-485 pada masing-masing mikrokontroller ditunjukan dalam gambar 3.8 berikut.



Gambar 3.8. Rangkaian *converter* saluran transmisi RS-485

3.3. Perangkat Lunak Mikrokontroler.

Untuk mendukung *hardware* yang sudah dibuat, maka dibutuhkan perangkat lunak (*software*) supaya perangkat keras tersebut bisa berjalan sesuai dengan tujuan. Mikrokontroler dapat mengendalikan seluruh sistem apabila ada urutan instruksi yang mendefinisikan secara jelas urutan kerja yang harus dilaksanakan. Dalam perancangan alat ini perangkat lunak yang digunakan adalah bahasa pemrograman *Delphi.*

Urutan instruksi ini sangat penting untuk didefenisikan, karena mikrokontroller bekerja secara pasti berdasarkan urutan insruksi ini. 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 aplikasi tidak sesuai dengan spesifikasi awal. Oleh karena itu, perancangan perangkat keras sangat menentukan dalam keberhasilan pembuatan perangkat lunak, sama pentingnya dengan perancangan perangkat keras. Sebuah mikrokontroller tidak akan bekerja bila tidak diberikan program. Program tersebut memberitahukan apa yang harus dilakukan oleh mikrokontroller.

3.3.1. Langkah Pembuatan Program.

- Buat diagram alir dari program yang dibuat.
- Mengubah flowchart ke dalam bahasa pemprograman Delphi sesuai urutan jalannya program.
- Mengkompilasi program yang dibuat sampai menghasilkan struktur program yang dihasilkan.
- Mendownload program yang telah sesuai dengan apa yang diharapkan ke dalam mikrokontroler dengan menggunakan downloader sesuai tipe mikro

yang digunakan.

3.4. Desain komunikasi client dan server.

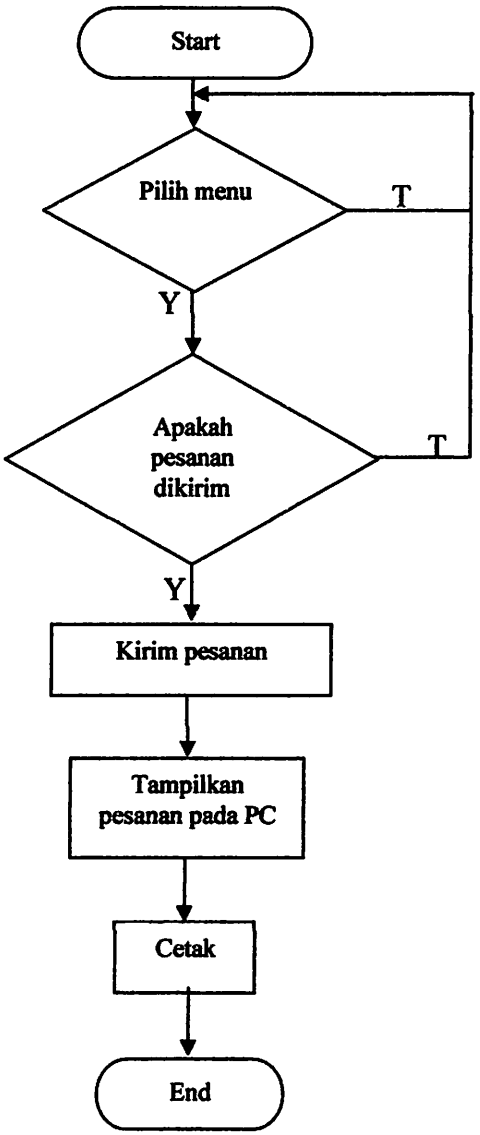
Pada saat progam dinyalakan server request E_0 dan E_1 pada masing-masing meja, untuk menanyakan E_0 apakah ada pesanan pada meja 1. Server akan menunggu jawaban pada meja 1 selama 1 detik. Jika ada jawaban maka data disimpan pada database untuk diprint. Data pesanan terdiri dari jenis makanan dan minuman (merupakan urutan dari menu), jumlah porsi/buah, contohnya ada pesanan pecel 2 porsi/buah dan es teh 2 porsi/buah jadi data yang dikirim oleh mikro adalah E_0 (2,2,21,2). Angka 2 pertama adalah urutan daftar menu sedangkan angka 2 yang kedua adalah jumlah pesanan. Jika tidak ada jawaban maka server akan melanjutkan request pada E_1 apakah ada pesanan pada meja 2, jika ada pesanan pada meja 2 maka server akan mengambil data pesanan dan disimpan pada database untuk diprint. Jika tidak ada maka server akan mengcounter ulang pada meja 1.

Untuk mengupdate menu pada client 1 dan 2, PC request F_0 untuk meja 1 dan F_1 `untuk meja 2. untuk menu yang akan diupdate. Untuk update menu hanya dapat diupdate pada server aja, jika menu yang masih ada ditandai dengan true atau logic "1" sedangkan yang tidak ada ditandai false atau logic "0". Jadi server hanya mengirimkan logic "1" dan "0" pada masing-masing meja. Contoh menu pecel habis kita lihat pecel dalam daptar menu berada pada urutan menu no 2. jadi uratan tersebut yang diberi logic "0" . Maka server mengirimkan F_0 (1,0,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,) pada meja client 1.

3.5. Personal Computer

Untuk menjalankan progam Delphi pada komputer dibutuhkan spesifikasi minimnya menggunakan pentium 3, memory RAM 256Mb, hardisk 80Gb. Kurang dari spesifikasi tersebut progam tidak dapat jalan dengan sempurnya atau kemungkinan adanya error pada saat program Delphi berjalan. Untuk itu spesifikasi diatas perlu diperhatikan.

3.6 Diagram alir secara umum



Gambar 3.9 *flowchart* secara umum.

BAB IV

PENGUKURAN DAN PENGUJIAN

4.1. Umum

Untuk memastikan apakah sistem pengiriman data menu masing – masing meja dapat bekerja sesuai dengan spesifikasi perencanaan,diperlukan serangkaian pengujian dan pengukuran.

Bab pengujian dan pengukuran ini menguraikan tentang bagian alat yang diuji, tujuan pengujian, langkah-langkah pengujian dan hasil pengujian yang menunjukkan unjuk kerja dari tiap-tiap bagian alat. Pembahasan dalam bab ini dibagi menurut pembagian alat yang diuji untuk mengetahui untuk kerja sistem secara keseluruhan.
Pada bab ini membahas cara pegujian dan analisa dari alat yang dirancang, sehingga dapat diketahui apakah alat tersebut dapat bekerja sesuai dengan yang telah direncanakan. Dalam rangka pengujian alat tersebut diuraikan percobaan yang dilakukan untuk mengetahui respon dari keseluruhan alat yang telah dirancang.

Untuk mengetahui kemampuan alat dan sistem kerja sesuai dengan program yang telah dibuat maka dilakukan pengujian pada alat dan sistem kerja alat dengan prosedur pengujian sebagai berikut:

1. Pengujian perangkat keras
2. Pengujian sistem secara keseluruhan

4.2. Pengujian Hardware

Pengujian perangkat keras ini mencakup pengujian rangkaian elektronika pada masing-masing blok maupun blok secara keseluruhan yang telah dirancang dengan menggunakan multimeter digital

.

4.2.1 Pengujian LCD

4.2.1.1. Tujuan

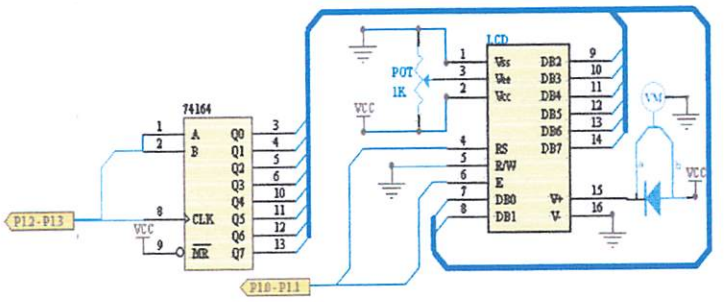
Adapun tujuan dari pengujian rangkaian ini untuk mengetahui kondisi keluaran LCD yaitu sebagai tampilan,juga mencatat nilai tegangan yang masuk pada LCD sebelum dan sesudah melewati diode.

4.2.1.2. Alat-alat yang digunakan

- LCD
- Rangkaian mikrokontroler AT 89S51.
- Catu daya.

4.2.1.3. Prosedur pengujian

1. Menyusun rangkaian pengujian seperti pada gambar 4.1



Gambar 4.1 Rangkaian Pengujian LCD

2. Membuat *software* pengujian rangkaian LCD, program ini berisi inisialisasi *mikrokontroler* dan LCD.

```
mulai:  mov     DPTR,#nama
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#nim
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   delay2
```

3. Mengaktifkan catu daya.
4. Mengoperasikan program dan hasil keluaran akan ditunjukkan pada layar penampil kristal cair.

4.2.1.4. Hasil pengujian

Dari hasil pengujian maka didapatkan tampilan seperti yang terlihat pada gambar berikut ini:

Gambar 4.2 Tampilan Pengujian LCD

4.2.2. Pengujian Keypad 4 x 4

4.2.2.1. Tujuan

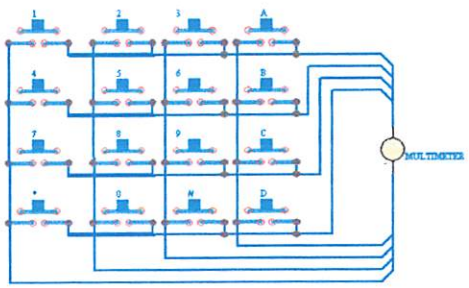
Untuk menguji apakah tombol keypad dapat bekerja sebagai inputan,dan mensimulasikan tombol yang ditekan melalui suara buzzer pada multimeter digital.

4.2.2.2. Alat yang digunakan

- Multimeter digital
- *Keypad*

4.2.2.3. Prosedur Pengujian

1. Menyusun rangkaian pengujian keypad seperti pada gambar 4.3.



Gambar 4.3 Pengujian Rangkaian *Keypad.*

2. Memberikan kombinasi masukan dengan menekan tombol-tombol *keypad.*
3. Mengamati hasil penekanan *keypad.* Kemudian mencatat hasil pengamatan pada tabel 4.1.

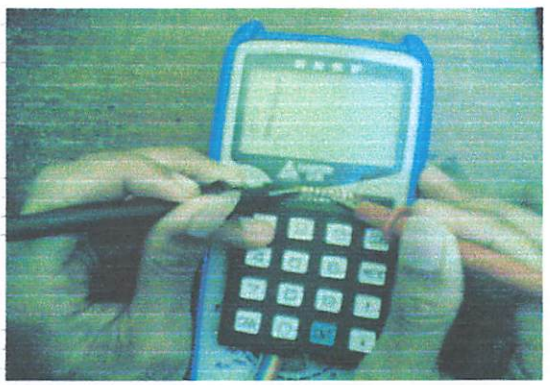
4.2.2.4. Hasil Pengujian

Tabel 4.1 Hasil Pengujian *Keypad.*

TOMBOL	BARIS				KOLOM			
	1	2	3	4	1	2	3	4
1	1	0	0	0	1	0	0	0
2	1	0	0	0	0	1	0	0
3	1	0	0	0	0	0	1	0

A	1	0	0	0	0	0	0	1
4	0	1	0	0	1	0	0	0
5	0	1	0	0	0	1	0	0
6	0	1	0	0	0	0	1	0
B	0	1	0	0	0	0	0	1
7	0	0	1	0	1	0	0	0
8	0	0	1	0	0	1	0	0
9	0	0	1	0	0	0	1	0
C	0	0	1	0	0	0	0	1
#	0	0	0	1	1	0	0	0
0	0	0	0	1	0	1	0	0
*	0	0	0	1	0	0	1	0
D	0	0	0	1	0	0	0	1

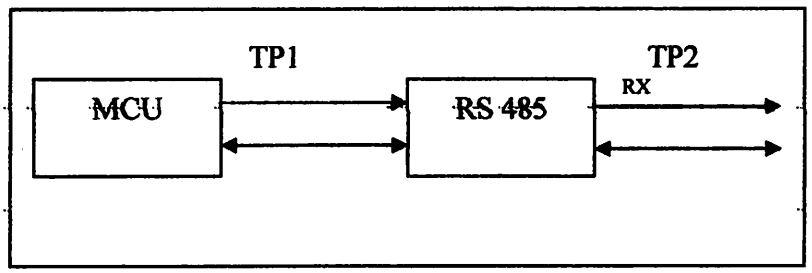
*Keterangan : cara membaca tabel diatas adalah jika antara baris dan kolom terhubung (1) maka akan membentuk matrik baris dan kolom sesuai penekanan tombol *keypad.*



Gambar 4.4 Pengecekan Jalur *Keypad* Dengan Multimeter.

4.2.3. Pengujian jalur komunikasi serial

Pengujian ini bertujuan untuk mengetahui besaran elektrik yang digunakan dalam komunikasi data serial, sehingga dapat diidentifikasikan sebagai logika '1' dan logika '0'. Pengujian ini terdapat pada titik hubung antara MCU dan RS 485 jalur komunikasi RS 485. Untuk lebih jelasnya dapat dilihat dalam gambar 4.5 berikut:



Gambar 4.5 Titik pengujian jalur komunikasi serial

Tabel 4.2 Hasil pengujian jalur komunikasi serial

Pada Saat	Hasil Pengujian	
	TP1	TP2
1	1 (5 volt)	1 (4,8 volt)
2	0 (0 volt)	0 (0,2 volt)

Analisa penujian

Pada saat penyalaan pertama ditampilkan menu makanan, serta harga pada LCD. Kita memilih menu tersebut dan masukan kedalam kolom pemesanan ketika kita keluar dari menu tersebut. MCU menerima dari sentral dan mencocokkannya dengan address yang dia punya. Ketika data yang masuk sama dengan address yang dia punya maka MCU mengirimkan data balasan yang berarti siap berkomunikasi untuk mengirimkan

data pemesanan. Ketika data yang dikirimkam tidak sama dengan address yang dia punya maka MCU tidak mengirimkan data tersebut.

Pengiriman data tersebut dilakukan secara bertahap (step by step) mulai dari pemesanan pertama sampai dengan kelima, jika tidak ada pesanan maka data yang dikirim adalah 0 (kosong). Ketika central menerima data dari meja pemesan maka akan diterima oleh MCU untuk dicetak.

4.3. Pengujian sistem keseluruhan komunikasi RS 485

Pada saat penyalaan pertama ditampilkan menu makanan minuman serta harga pada LCD. Kita memilih menu tersebut dan memasukannya kedalam kolom pemesanan, ketika kita keluar dari menu tersebut, MCU menerima dari server mencocokkannya dengan address yang dia punya. Ketika data yang masuk sama dengan address yang dia punya. Maka MCU mengirimkan data balasan yang berarti siap berkomunikasi untuk mengirimkan data pesanan. Ketika data yang masuk tidak sama dengan address yang dia punya maka MCU tidak menghiraukan data tersebut.

Pengiriman data tersebut dilakukan secara bertahap(step by step) mulai dari pemesanan pertama sampai dengan kelima, jika tidak ada pesanan maka data yang dikirim adalah 0 (kosong).

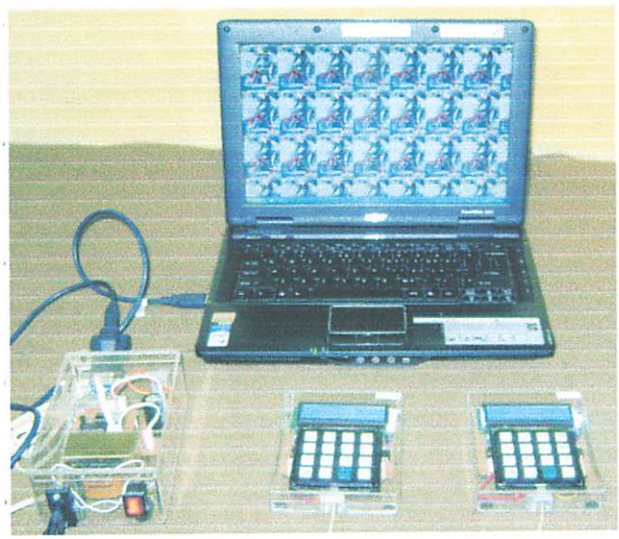
Ketika server menerima data dari setiap meja berupa pesanan maka MCU server mencetaknya keprinter.

4.4. Pengujian sistem secara menyeluruh

Pengujian rangkaian secara keseluruhan dilakukan dengan menghubungkan Masing-masing rangkaian atau blok dan menjalankan perangkat lunak yang dibuat Pengujian ini dimaksudkan untuk mengetahui apakah peralatan yang dibuat telah sesuai Dengan perecanaan :

1. Pengujian alat digunakan pada daerah yang sama dengan beberapa kali pengukuran
2. Memberi tegangan catu daya 12 Volt DC
3. Menghidupkan saklar dengan mengamati LCD
4. Mengisi data melalui *keypad*
5. Komputer sebagai server yang berada pada kasir

Dari hasil pengujian alat secara keseluruhan didapatkan hasil yang maksimal. Jadi dalam pembuatan alat ini sudah pada tahap yang sempurna tanpa ada gagal pada pengiriman data. Gambar 4.6 menunjukan gambar keseluruhan alat.



Gambar 4.6 Alat keseluruhan

BAB V

PENUTUP

5.1. Kesimpulan

Setelah melakukan perencanaan dan pengujian sistem pengontrol untuk restoran menggunakan mikrokontroller AT89S51 dapat diambil kesimpulan sebagai berikut:

1. Rangkaian keypad dan penampilan LCD dapat bekerja sesuai dengan perencanaan yaitu sebagai masukan input dan LCD sebagai penampilan dari menu-menu yang ada pada restoran.
2. Mikrokontroller dalam proses pengontrolan sistem dan keypad sebagai masukan dan program menu ditampilkan pada LCD, sedangkan komunikasi dengan server menggunakan serial RS-485.
3. Untuk menghindari keterlambatan pada saat pengiriman data jarak jangkauan kabel maksimal 15 meter.

5.2. Saran

Meskipun hasil pengujian dan analisis sudah menyampai keadaan yang diharapkan namun masih banyak perbaiakan atau pengembangan seperti menu dapat diganti dengan menu lain, harga dapat diganti untuk lebih meningkatkan lagi performasi alat.

DAFTAR PUSTAKA

[1] Elektronika-elektronika.blogspot.com2007\2007_02_01_archive.html

[2] http://www.atmel.com

[3] Leon W. Couch, II. (1995). *Digital and Analog Communication Systems.* (3rd edition). Upper \\ Saddle River, New Jersey, 07458: Prentice Hall.

[4] Ir.Melani Satyoadi (2004). *Elektronika digital.* Andi, Yogyakarta, 2004

[5] Albert Paul Malvino, Hanapi Gunawan, Prinsip-prinsip Elektronika, Erlangga, Jakarta, 1990

[6] Hafindo, Pelatihan Microcontroller MCS-51 Programming and Interfacing, Hafindo Electronic & Education, Malang, 2001.

[7] http//www.data sheet AT89S51.com

[8] http//www.popnas.com

[9] http//www.jatengfast.com

[10] http//www.dallas.com

LAMPIRAN - LAMPIRAN

INSTITUT TEKNOLOGI NASIONAL
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO
JL.Raya Karanglo Km 2 MALANG

PERSETUJUAN PERBAIKAN SKRIPSI

Dari hasil Ujian Kompreherensip Jenjang Strata Satu (S1) Jurusan Teknik Elektro Konsentrasi Elektronika yang diselenggarakan pada :

Hari : Selasa
Tanggal : 17 Maret 2009

Telah dilaksanakan perbaikan skripsi oleh saudara :

Nama : Sepviyan Yudiantho
NIM : 03.17.019

Perbaikan tersebut meliputi :

Penguji	Materi Perbaikan	Paraf
I	• Gambar diagram blok sistem, karena hubungan antara RS 485 pada dasarnya parallel.	
II	• Pengujian. • Desain komunikasi (client dan server).	

Malang, Maret 2009

Disetujui Oleh

Penguji I	Penguji II
Sotyohadi, ST	**Joseph Dedy Irawan, ST MT**
NIP.Y.1039700309	NIP : 132315178

Mengetahui

Dosen Pembimbing I	Dosen Pembimbing II
M.Ibrahim Ashari, ST, MT	**Irmalia Suryani Faradisa, ST, MT**
NIP.Y.1030100358	NIP.P.103000035

INSTITUT TEKNOLOGI NASIONAL
Jl. Raya Karanglo Km 2
MALANG

FORM BIMBINGAN SKRIPSI

Nama : **SEPVIYAN YUDIANTO**
NIM : **03.17.019**
Masa Bimbingan : **9 November 2008 s/d 9 Mei 2009**
Judul : **Perancangan dan pembuatan sistem pemesanan makanan pada restoran berbasis microcontroller AT89S51**

NO	Tanggal	Uraian	Paraf Pembibing
1	17 feb'09	Revisi Bab 1 dan Bab 2.	
2	18 feb'09	acc Bab 1 dan Bab 2	
3	20 feb'09	acc Bab III	
4	2 maret'09	Bab IV dan Bab V revisi tulisan dan alat	
5	11 maret'09	Revisi makalah seminar	
6	13 maret'09	acc makalah seminar	
7			
8			
9			
10			

Malang,
Dosen Pembimbing

IBRAHIM ASHARI,ST,MT
NIP.Y.1030100358

Form S-4a

INSTITUT TEKNOLOGI NASIONAL
Jl. Raya Karanglo Km 2
MALANG

FORM BIMBINGAN SKRIPSI

Nama : **SEPVIYAN YUDIANTO**
NIM : **03.17.019**
Masa Bimbingan : **9 November 2008 s/d 9 Mei 2009**
Judul : **Perancangan dan pembuatan sistem pemesanan makanan pada restoran berbasis microcontroller AT89S51**

NO	Tanggal	Uraian	Paraf Pembibing
1	24/05/02	BAB I, II, III Seperti makalah.	
2	30/05/01	BAB I acc. II → PC tambahkan	
3		III → flow chart ? belajar.	
4	07/03/05	Acc BAB II, III Revisi BAB IV, V	
5	16/03/03	Acc. ujian kompre	
6			
7			
8			
9			
10			

Malang,
Dosen Pembimbing

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

Form S-4a

Features

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

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 Atmel's high-density nonvolatile memory technology and is compatible with the industry-standard 80C51 instruction set and pinout. The on-chip Flash allows the program memory 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 monolithic chip, the Atmel AT89S51 is a powerful microcontroller which provides a highly-flexible and cost-effective solution to many embedded control applications.

The AT89S51 provides the following standard features: 4K bytes of Flash, 128 bytes of RAM, 32 I/O lines, Watchdog timer, two data pointers, two 16-bit timer/counters, a five-vector two-level interrupt architecture, a full duplex serial port, on-chip oscillator, and clock circuitry. In addition, the AT89S51 is designed with static logic for operation down to 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 contents but freezes the oscillator, disabling all other chip functions until the next external interrupt or hardware reset.

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

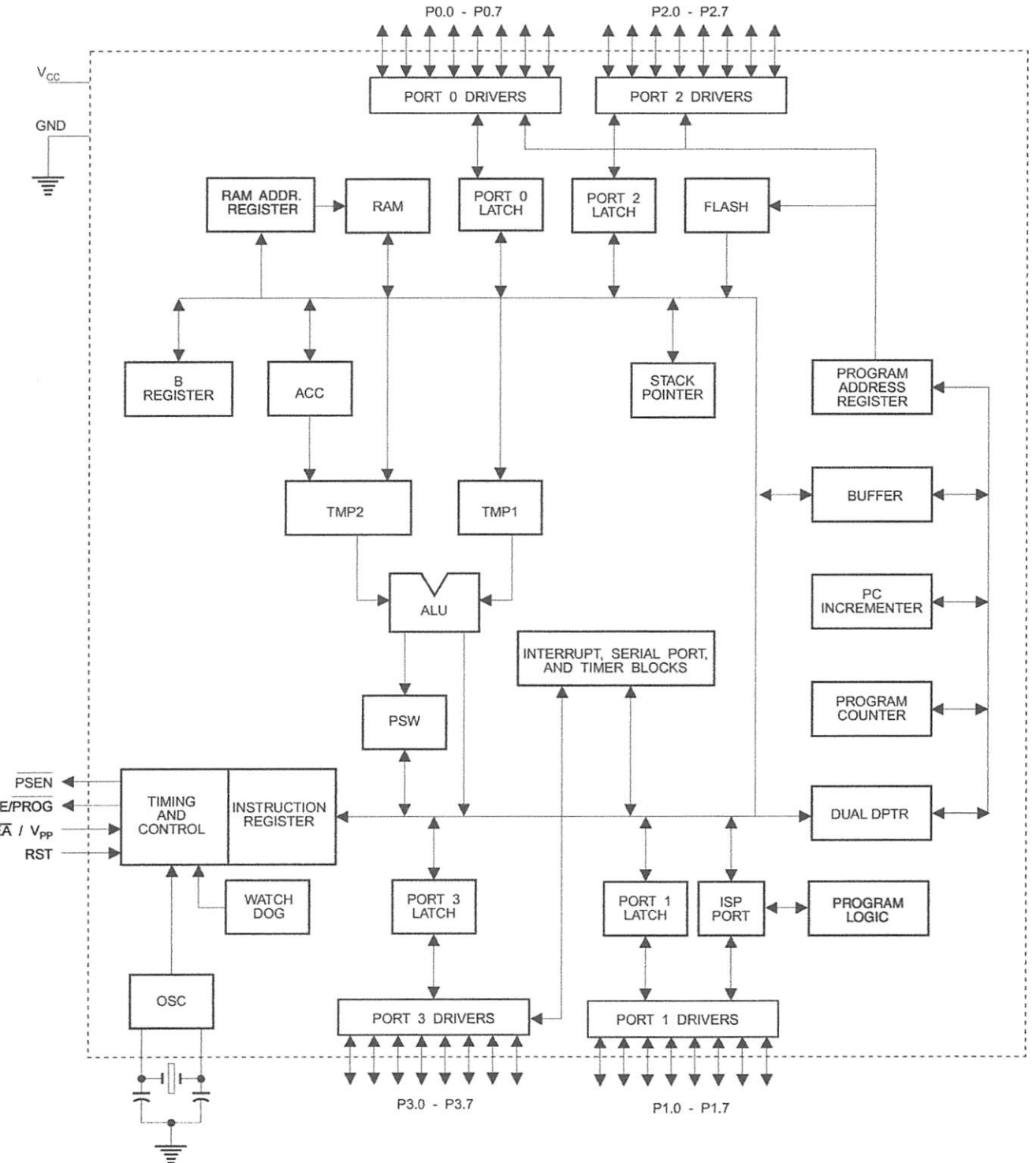
AT89S51

Rev. 2487A–10/01

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

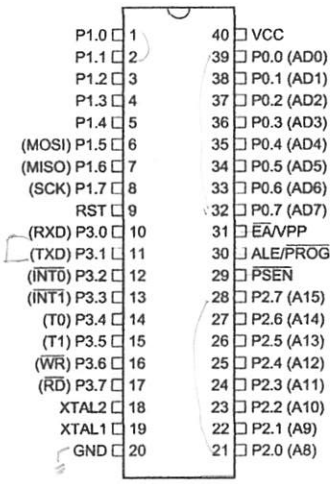
AT89S51

ock Diagram

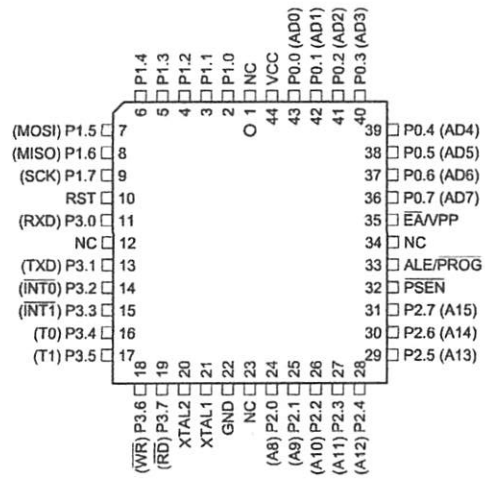

P0.0 - P0.7
P2.0 - P2.7
V_{CC}
GND
PORT 0 DRIVERS
PORT 2 DRIVERS
RAM ADDR. REGISTER
RAM
PORT 0 LATCH
PORT 2 LATCH
FLASH
B REGISTER
ACC
STACK POINTER
PROGRAM ADDRESS REGISTER
BUFFER
TMP2
TMP1
ALU
PC INCREMENTER
INTERRUPT, SERIAL PORT, AND TIMER BLOCKS
PSW
PROGRAM COUNTER
PSEN
E/PROG
A / V_{PP}
RST
TIMING AND CONTROL
INSTRUCTION REGISTER
DUAL DPTR
WATCH DOG
PORT 3 LATCH
PORT 1 LATCH
ISP PORT
PROGRAM LOGIC
OSC
PORT 3 DRIVERS
PORT 1 DRIVERS
P3.0 - P3.7
P1.0 - P1.7

n Configurations

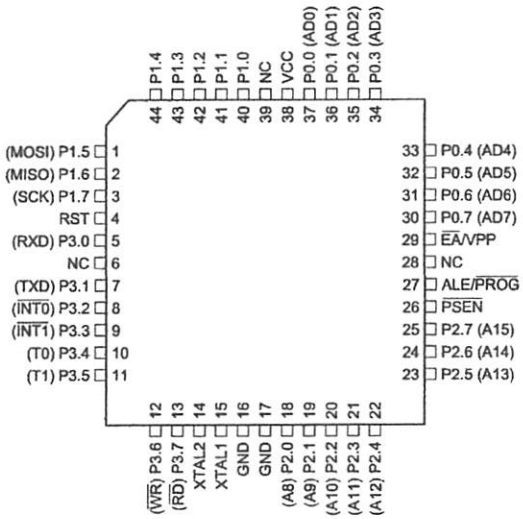
PDIP



PLCC



TQFP



n Description

C Supply voltage.

ID Ground.

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

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

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

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

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

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

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

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

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

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

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

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

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

T

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.

E/$\overline{PROG}$

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

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

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

$\overline{EN}$

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

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

/VPP

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

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

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

AL1

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

AL2

Output from the inverting oscillator amplifier

pecial
nction
egisters

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.

ble 1. AT89S51 SFR Map and Reset Values

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

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

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

Table 2. AUXR: Auxiliary Register

AUXR Address = 8EH Reset Value = XXX00XX0B

Not Bit Addressable

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

– Reserved for future expansion

DISALE Disable/Enable ALE

DISALE Operating Mode

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

1 ALE is active only during a MOVX or MOVC instruction

DISRTO Disable/Enable Reset out

DISRTO

0 Reset pin is driven High after WDT times out

1 Reset pin is input only

WDIDLE Disable/Enable WDT in IDLE mode

WDIDLE

0 WDT continues to count in IDLE mode

1 WDT halts counting in IDLE mode

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

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

Table 3. AUXR1: Auxiliary Register 1

AUXR1

Address = A2H

Reset Value = XXXXXXX0B

Not Bit Addressable

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

– Reserved for future expansion

DPS Data Pointer Register Select

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

ɨmory ganization

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.

ɔgram 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.

ta 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.

ɨtchdog ner ne-time ɨabled with ɨset-out)

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

ɨng the WDT

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

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

DT During ower-down d Idle

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

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

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

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

ART

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

mer 0 and 1

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

terrupts

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

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

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

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

Table 4. Interrupt Enable (IE) Register

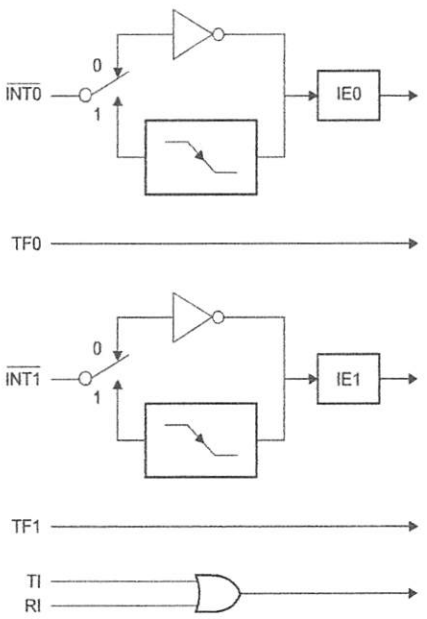
(MSB)							(LSB)
EA	–	–	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

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

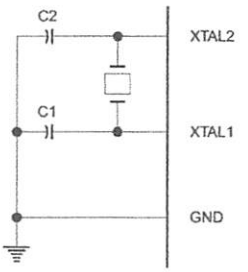
Figure 1. Interrupt Sources



scillator haracteristics

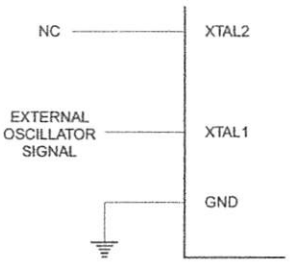
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 2. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected 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



e Mode

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

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

wer-down ode

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

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

Mode	Program Memory	ALE	$\overline{PSEN}$	PORT0	PORT1	PORT2	PORT3
Idle	Internal	1	1	Data	Data	Data	Data
Idle	External	1	1	Float	Data	Address	Data
Power-down	Internal	0	0	Data	Data	Data	Data
Power-down	External	0	0	Float	Data	Data	Data

rogram emory Lock ts

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

Table 6. Lock Bit Protection Modes

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No program lock features
2	P	U	U	MOVC instructions executed from external program memory are disabled from fetching code bytes from internal memory, $\overline{EA}$ is sampled and latched on reset, and further programming of the Flash memory is disabled
3	P	P	U	Same as mode 2, but verify is also disabled
4	P	P	P	Same as mode 3, but external execution is also disabled

When lock bit 1 is programmed, the logic level at the $\overline{EA}$ pin is sampled and latched during reset. If the device is powered up without a reset, the latch initializes to a random value and holds that value until reset is activated. The latched value of $\overline{EA}$ must agree with the current logic level at that pin in order for the device to function properly.

rogramming e Flash – arallel Mode

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

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

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

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

$\overline{Data}$ Polling: The AT89S51 features $\overline{Data}$ Polling to indicate the end of a byte write cycle. During a write cycle, an attempted read of the last byte written will result in the complement of the written data on P0.7. Once the write cycle has been completed, true data is valid on all outputs, and the next cycle may begin. $\overline{Data}$ Polling may begin any time after a write cycle has been initiated.

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

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

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

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

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

ogramming e Flash – rial Mode

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

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

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

rial ogramming gorithm

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

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

Power-off sequence (if needed):

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

Set RST to "L".

Turn V_{CC} power off.

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

rial ogramming struction Set

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

ogramming terface – rallel 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.

ble 7. Flash Programming Modes

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

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

Figure 4. Programming the Flash Memory (Parallel Mode)

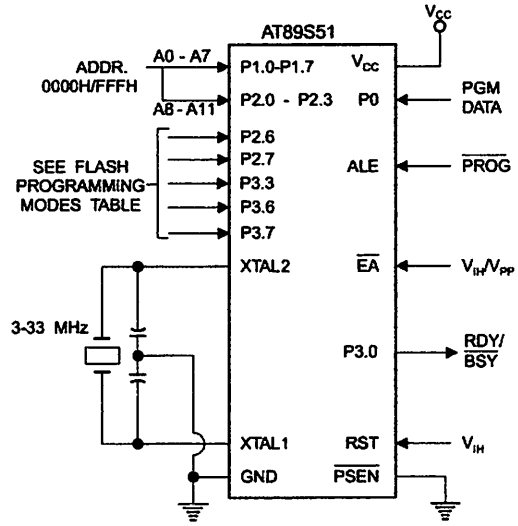
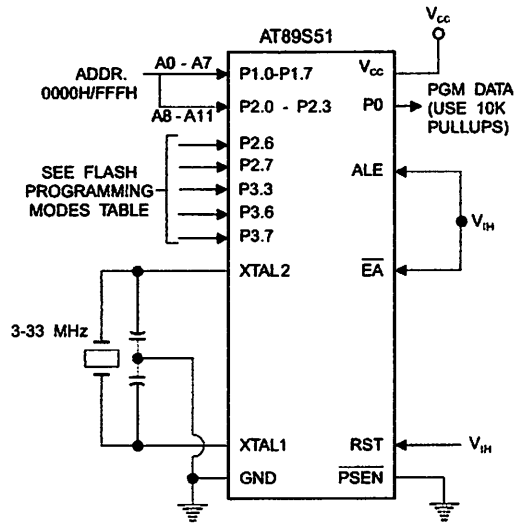


Figure 5. Verifying the Flash Memory (Parallel Mode)

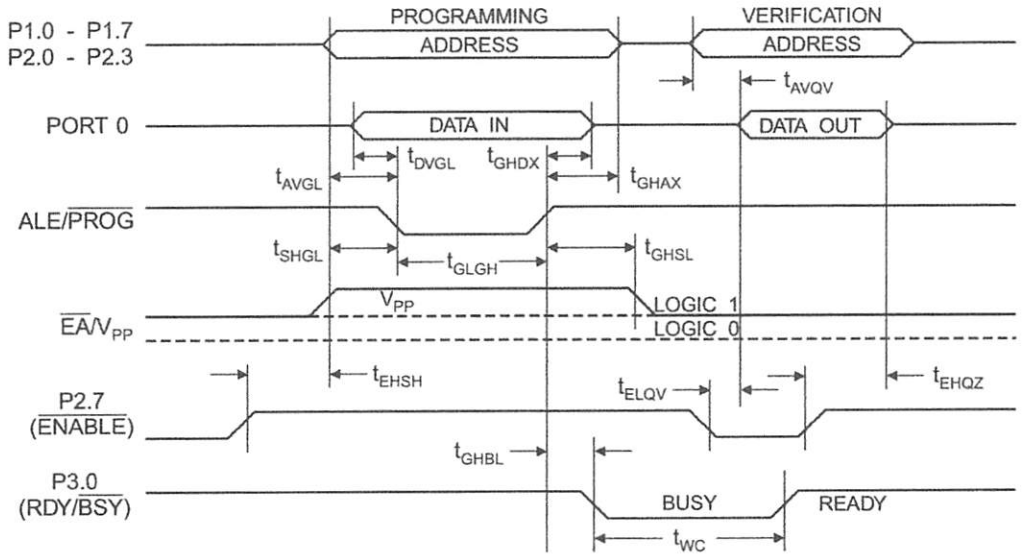


lash Programming and Verification Characteristics (Parallel Mode)

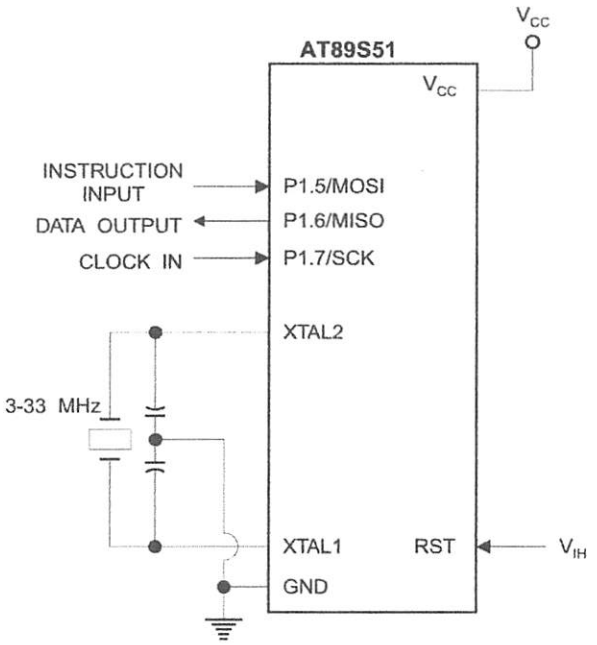
= 20°C to 30°C, V_{CC} = 4.5 to 5.5V

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

gure 6. Flash Programming and Verification Waveforms – Parallel Mode

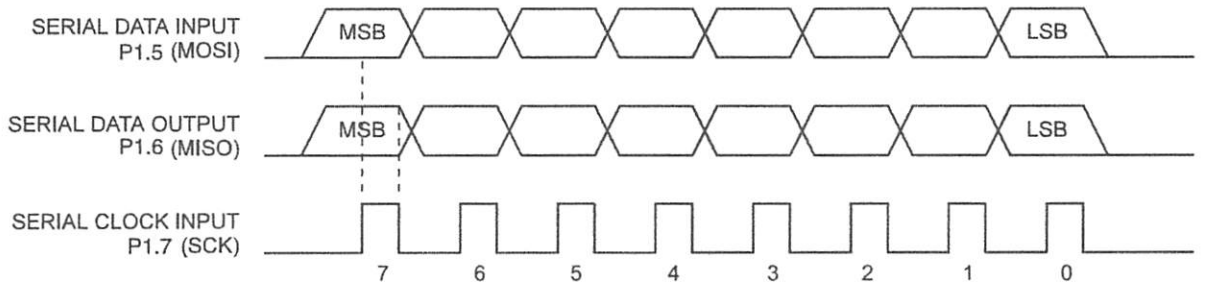


gure 7. Flash Memory Serial Downloading



lash Programming and Verification Waveforms – Serial Mode

gure 8. Serial Programming Waveforms



ble 8. Serial Programming Instruction Set

	Instruction Format				
ıstruction	**Byte 1**	**Byte 2**	**Byte 3**	**Byte 4**	**Operation**
'rogramming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx 0110 1001 (Output)	Enable Serial Programming while RST is high
:hip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx	Chip Erase Flash memory array
:ead Program Memory 3yte Mode)	0010 0000	xxxx A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Read data from Program memory in the byte mode
/rite Program Memory 3yte Mode)	0100 0000	xxxx A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0	Write data to Program memory in the byte mode
/rite Lock Bits[2]	1010 1100	1110 00 B1 B2	xxxx xxxx	xxxx xxxx	Write Lock bits. See Note (2).
:ead 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")
:ead Signature Bytes[1]	0010 1000	xxx A5 A4 A3 A2 A1	A0 xxx xxxx	Signature Byte	Read Signature Byte
:ead Program Memory 'age 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)
/rite Program Memory 'age 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)

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

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

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

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

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

Serial Programming Characteristics

Figure 9. Serial Programming Timing

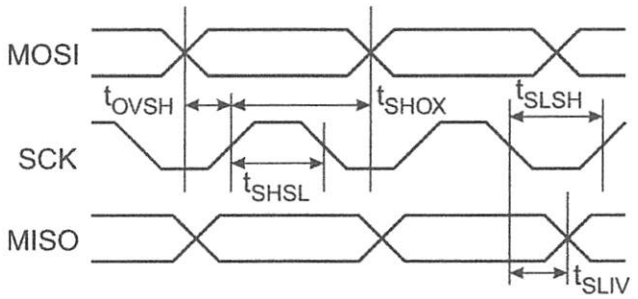


Table 9. Serial Programming Characteristics, T_A = -40° C to 85° C, V_{CC} = 4.0 - 5.5V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
$1/t_{CLCL}$	Oscillator Frequency	0		33	MHz
t_{CLCL}	Oscillator Period	30			ns
t_{SHSL}	SCK Pulse Width High	8 t_{CLCL}			ns
t_{SLSH}	SCK Pulse Width Low	8 t_{CLCL}			ns
t_{OVSH}	MOSI Setup to SCK High	t_{CLCL}			ns
t_{SHOX}	MOSI Hold after SCK High	2 t_{CLCL}			ns
t_{SLIV}	SCK Low to MISO Valid	10	16	32	ns
t_{ERASE}	Chip Erase Instruction Cycle Time			500	ms
t_{SWC}	Serial Byte Write Cycle Time			64 t_{CLCL} + 400	µs

bsolute Maximum Ratings*

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

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

C Characteristics

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

ymbol	Parameter	Condition	Min	Max	Units
IL	Input Low Voltage	(Except $\overline{EA}$)	-0.5	0.2 V_{CC}-0.1	V
IL1	Input Low Voltage ($\overline{EA}$)		-0.5	0.2 V_{CC}-0.3	V
IH	Input High Voltage	(Except XTAL1, RST)	0.2 V_{CC}+0.9	V_{CC}+0.5	V
IH1	Input High Voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC}+0.5	V
OL	Output Low Voltage[1] (Ports 1,2,3)	I_{OL} = 1.6 mA		0.45	V
OL1	Output Low Voltage[1] (Port 0, ALE, $\overline{PSEN}$)	I_{OL} = 3.2 mA		0.45	V
OH	Output High Voltage (Ports 1,2,3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, V_{CC} = 5V ±10%	2.4		V
		I_{OH} = -25 µA	0.75 V_{CC}		V
		I_{OH} = -10 µA	0.9 V_{CC}		V
OH1	Output High Voltage (Port 0 in External Bus Mode)	I_{OH} = -800 µA, V_{CC} = 5V ±10%	2.4		V
		I_{OH} = -300 µA	0.75 V_{CC}		V
		I_{OH} = -80 µA	0.9 V_{CC}		V
	Logical 0 Input Current (Ports 1,2,3)	V_{IN} = 0.45V		-50	µA
L	Logical 1 to 0 Transition Current (Ports 1,2,3)	V_{IN} = 2V, V_{CC} = 5V ±10%		-650	µA
I	Input Leakage Current (Port 0, $\overline{EA}$)	0.45 < V_{IN} < V_{CC}		±10	µA
RST	Reset Pulldown Resistor		50	300	KΩ
IO	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10	pF
CC	Power Supply Current	Active Mode, 12 MHz		25	mA
		Idle Mode, 12 MHz		6.5	mA
	Power-down Mode[2]	V_{CC} = 5.5V		50	µA

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

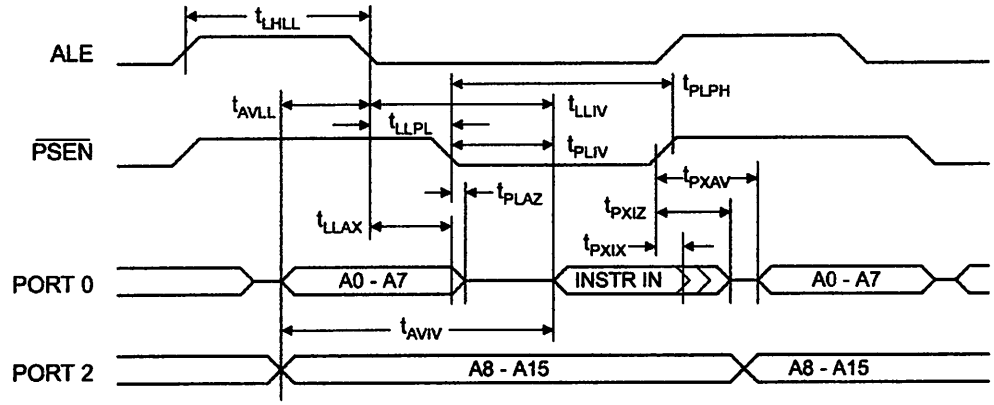
C Characteristics

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

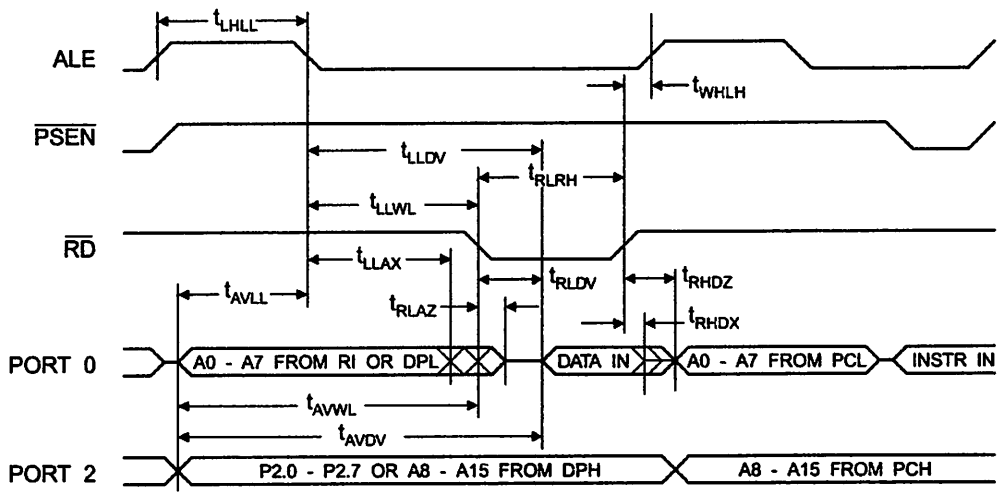
xternal Program and Data Memory Characteristics

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

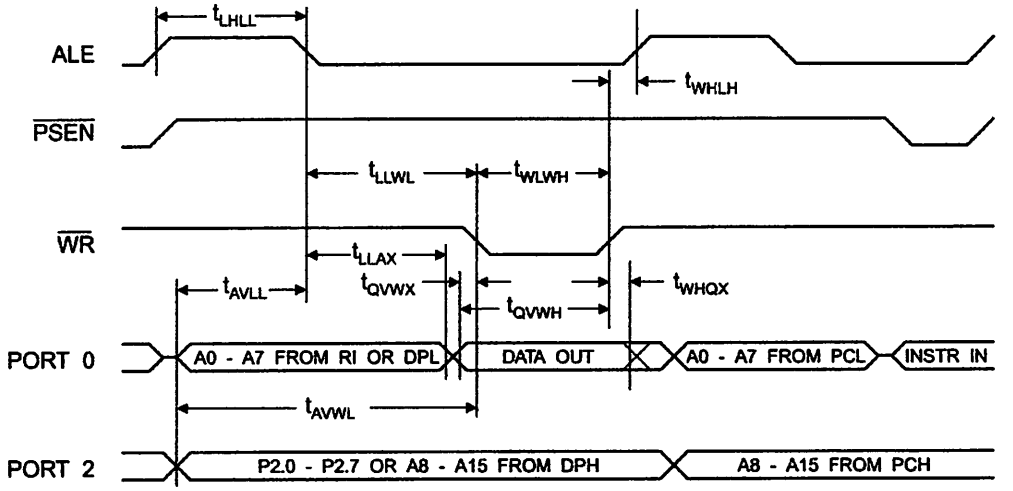
ternal Program Memory Read Cycle



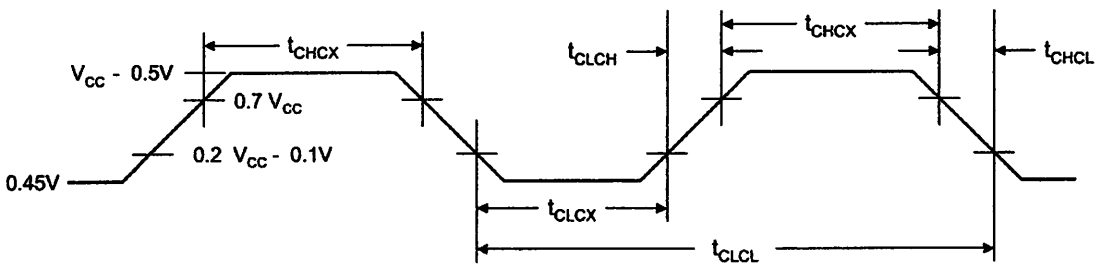
ternal Data Memory Read Cycle



ternal Data Memory Write Cycle



ternal Clock Drive Waveforms



ternal Clock Drive

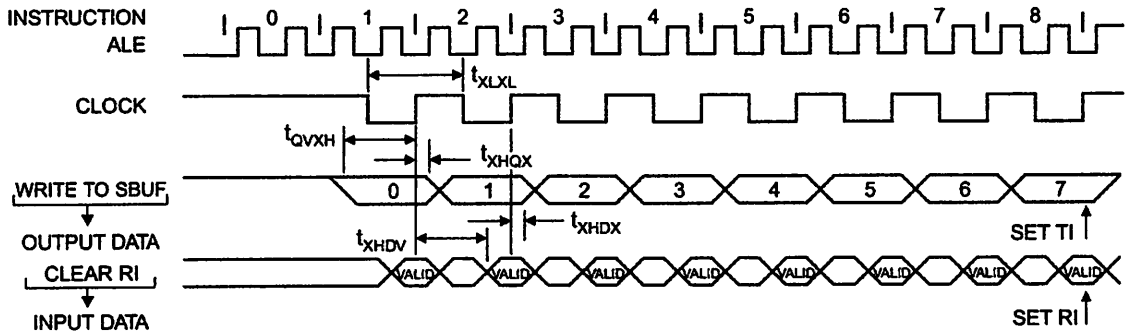
nbol	Parameter	Min	Max	Units
LCL	Oscillator Frequency	0	33	MHz
L	Clock Period	30		ns
CX	High Time	12		ns
X	Low Time	12		ns
H	Rise Time		5	ns
L	Fall Time		5	ns

rial Port Timing: Shift Register Mode Test Conditions

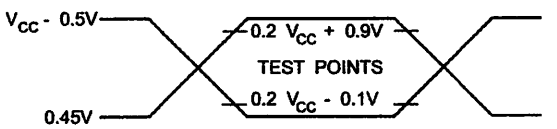
values in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

nbol	Parameter	12 MHz Osc		Variable Oscillator		Units
		Min	Max	Min	Max	
L	Serial Port Clock Cycle Time	1.0		$12t_{CLCL}$		µs
QV	Output Data Setup to Clock Rising Edge	700		$10t_{CLCL}$-133		ns
QX	Output Data Hold After Clock Rising Edge	50		$2t_{CLCL}$-80		ns
XX	Input Data Hold After Clock Rising Edge	0		0		ns
XV	Clock Rising Edge to Input Data Valid		700		$10t_{CLCL}$-133	ns

ift Register Mode Timing Waveforms

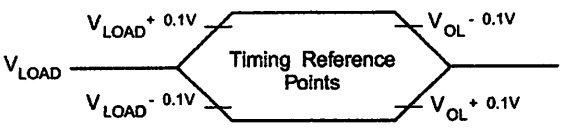


Testing Input/Output Waveforms[1]



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

at Waveforms[1]



: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

dering Information

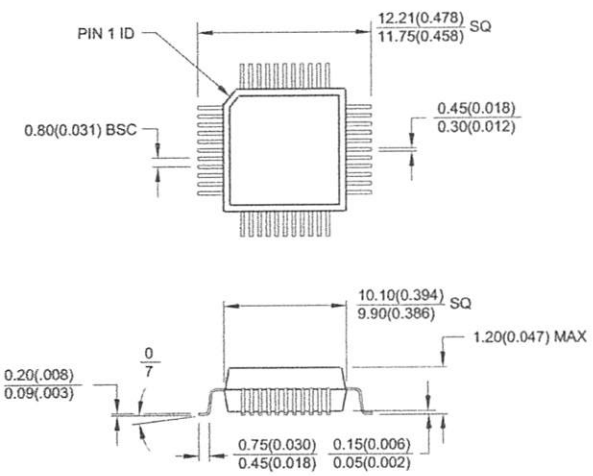
Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	4.0V to 5.5V	AT89S51-24AC AT89S51-24JC AT89S51-24PC	44A 44J 40P6	Commercial (0° C to 70° C)
		AT89S51-24AI AT89S51-24JI AT89S51-24PI	44A 44J 40P6	Industrial (-40° C to 85° C)
33	4.5V to 5.5V	AT89S51-33AC AT89S51-33JC AT89S51-33PC	44A 44J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

= Preliminary Availability

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

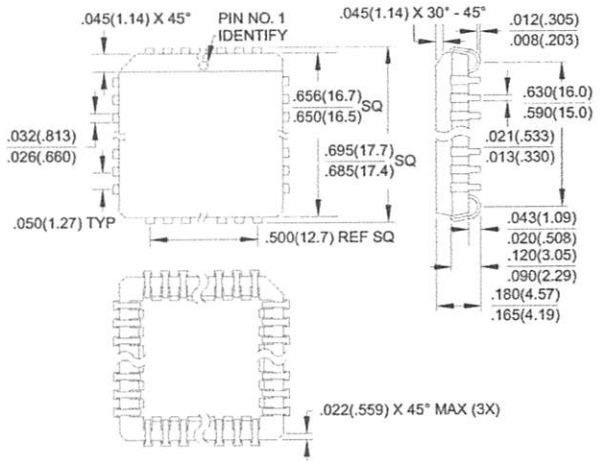
ckaging Information

44A, 44-lead, Thin (1.0 mm) Plastic Gull Wing Quad Flat Package (TQFP)
Dimensions in Millimeters and (Inches)*

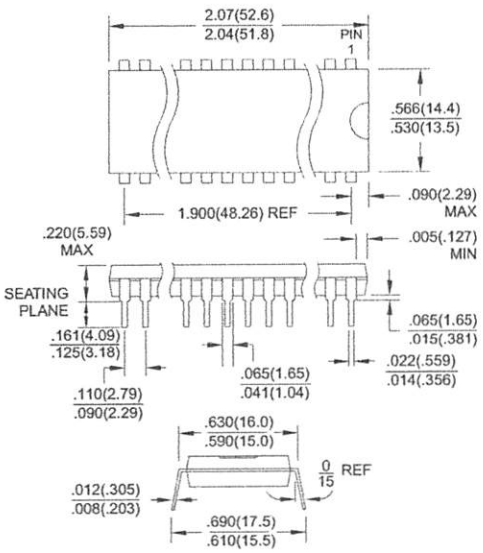


*Controlling dimension: millimeters

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



40P6, 40-pin, 0.600" Wide, Plastic Dual Inline Package (PDIP)
Dimensions in Inches and (Millimeters)
JEDEC STANDARD MS-011 AC



mel Headquarters

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

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

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

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

Atmel Product Operations

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

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

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

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

Atmel Rousset
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

Web Site
http://www.atmel.com

2487A–10/01/xM

AN.No.1632-711E

LIQUID CRYSTAL DISPLAY MODULE

M 1 6 3 2

USER MANUAL

Seiko Instruments Inc.

PREFACE

This manual describes technical informations on functions and instructions of M1632 from Seiko Instruments Inc. Please read this instruction manual carefully to understand all the module functions and make the best use of them. Description details may be changed without notice.

Revision Record

Edition	Revision	Date
1	Original	April 1985
2	Completely revised	Jan. 1987

1. GENERAL

1.1 General

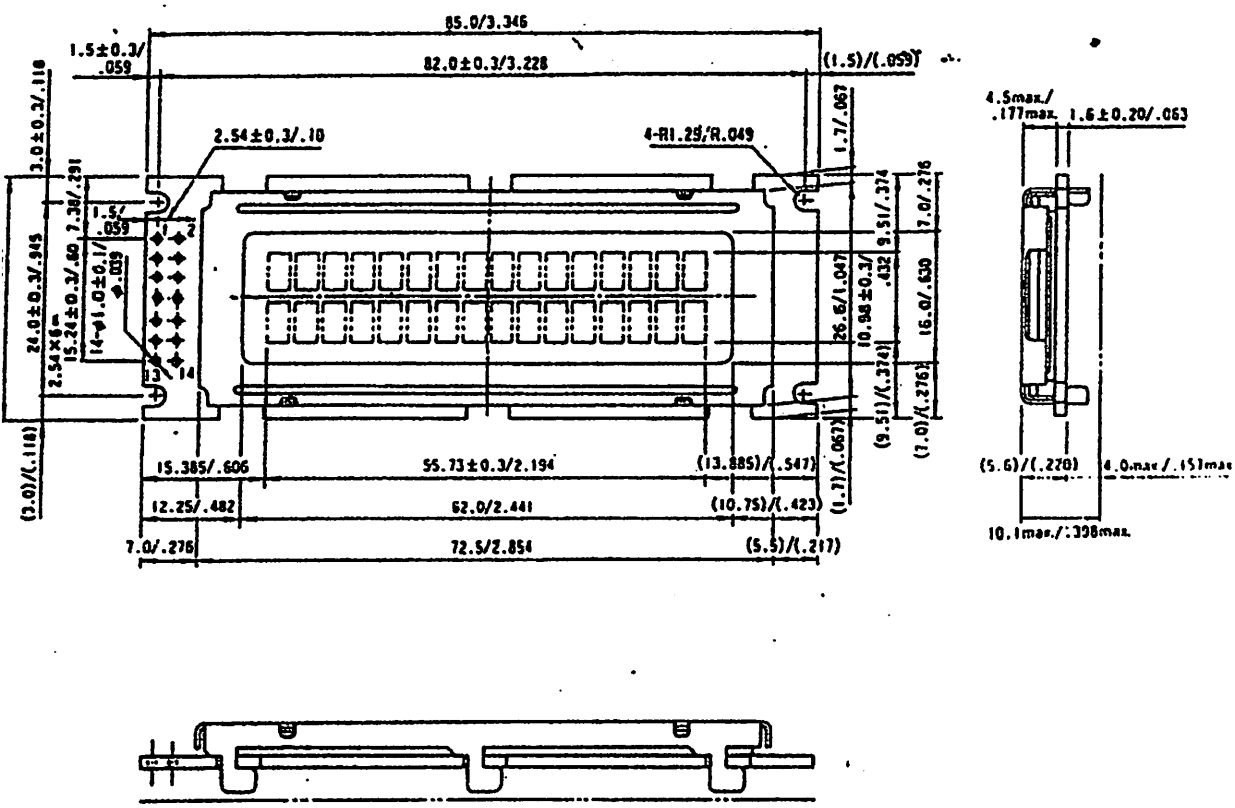
The M1632 is a low-power-consumption dot-matrix liquid crystal display (LCD) module with a high-contrast wide-view TN LCD panel and a CMOS LCD drive controller built in. The controller has a built-in character generator ROM/RAM, and display data RAM. All the display functions are controlled by instructions and the module can easily be interfaced with an MPU. This makes the module applicable to a wide range of purposes including terminal display units for microcomputers and display units for measuring gages.

1.2 Features

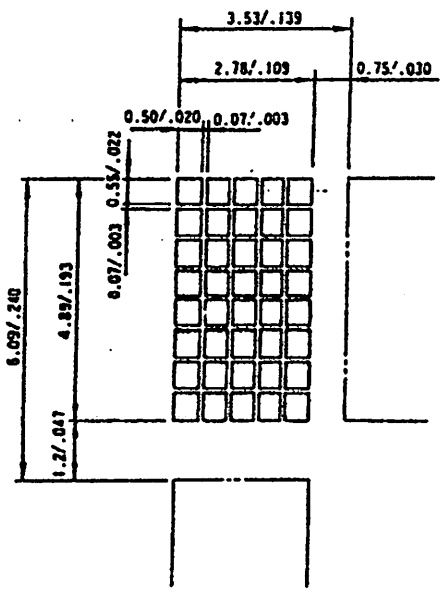
- 16-character, two-line TN liquid crystal display of 5 x 7 dot matrix + cursor
- Duty ratio: 1/16
- Character generator ROM for 192 character types.
  (character font: 5 x 7 dot matrix)
- Character generator RAM for eight character types (program write)
  (character font: 5 x 7 dot matrix)
- 80 x 8 bit display data RAM (80 characters maximum)
- Interface with four-bit and eight-bit MPUs possible
- Display data RAM and character generator RAM readable from MPU
- Many instruction functions

  Display Clear, Cursor Home, Display ON/OFF, Cursor ON/OFF, Display Character Blink, Cursor Shift, and Display Shift
- Built-in oscillator circuit
- +5 V single power supply
- Built-in automatic reset circuit at power-on
- CMOS process
- Operating temperature range: 0°C to 50°C

1.3 Dimensions Diagram



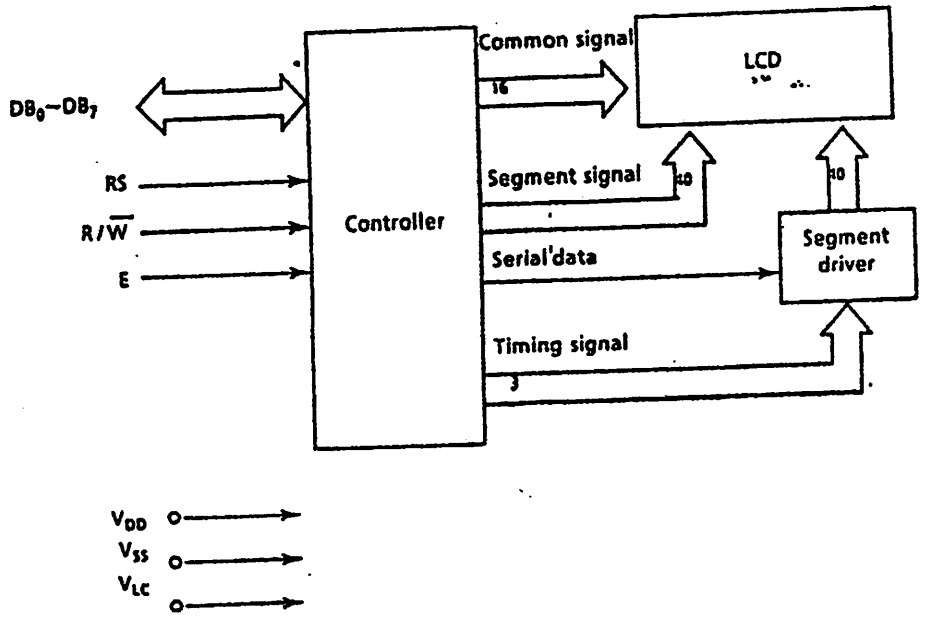
Unit : mm/inch
General tolerance : ±0.5 mm



No.	Symbol	Level	Function	
1	Vss	-	Power Supply	0V (GND)
2	Vcc	-		5V ±10%
3	Vee	-		for LCD Drive
4	RS	H/L	H: Data Input L: Instruction Input	
5	R/W	H/L	H:READ L:WRITE	
6	E	H, ↓	Enable Signal	
7	DB0	H/L	Data Bus	
8	DB1	H/L		
9	DB2	H/L		
10	DB3	H/L		
11	DB4	H/L		
12	DB5	H/L		
13	DB6	H/L		
14	DB7	H/L		
15	V+ BL	-	Back Light Supply	4 - 4.2V 50-200mA
16	V- BL	-		0V (GND)

Figure 1 Dimensions diagran

.4 Block Diagram



.5 Absolute Maximum Ratings

V_{SS} = 0 V

Item	Symbol	Standard	Unit	Remarks
Power supply voltage	V_{DD}	− 0.3 to + 7.0	V	
	V_{LC}	V_{DD} − 13.5 to V_{DD} + 0.3	V	
Input voltage	V_{in}	− 0.3 to V_{DD} + 0.3,	V	
Operating temperature	T_{opr}	0 to + 50	°C	
Storage temperature	T_{stg}	− 20 to + 60	°C	At 50% RH

.6 Electrical Characteristics

V_{DD} = 5 V ± 5%, V_{SS} = 0 V, T_A = 0°C to 50°C

Item		Symbol	Conditions	Standard			Unit
				Min.	Typ.	Max.	
Input voltage	High	V_{IH1}		2.2	–	V_{DD}	V
	Low	V_{IL1}		0	–	0.6	V
Output voltage (TTL)	High	V_{OH1}	$-I_{OH}$ = 0.205 mA	2.4	–	–	V
	Low	V_{OL1}	I_{OL} = 1.2 mA	–	–	0.4	V
Output voltage (CMOS)	High	V_{OH2}	$-I_{OH}$ = 0.04 mA	0.9V_{DD}	–	–	V
	Low	V_{OL2}	I_{OL} = 0.04 mA	–	–	0.1V_{DD}	V
Power supply voltage		V_{DD}		4.75	5.00	5.25	V
		V_{LC}	V_{DD} = 5 V, T_A = 25°C	–	0.25	–	V
Current consumption		I_{DD}		–	2.0	3.0	mA
		I_{LC}	V_{LC} = 0.25V	–	–	1.0	mA
Clock oscillation freq.		f_{osc}	Resistance oscillation	190	270	350	kHz

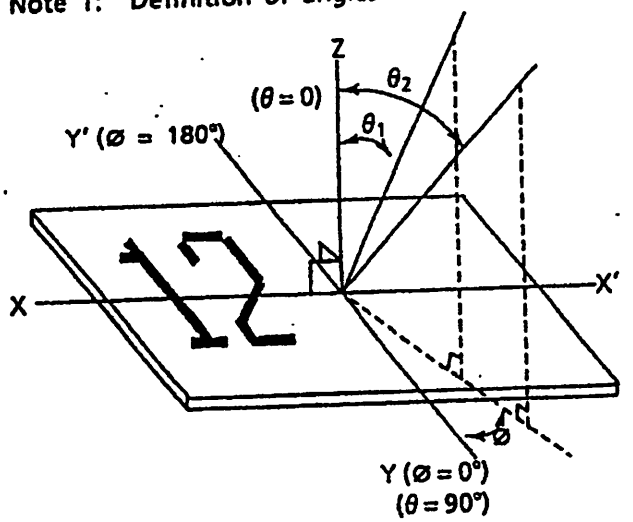
1.7 Optical Characteristics

1.7.1 Optical characteristics

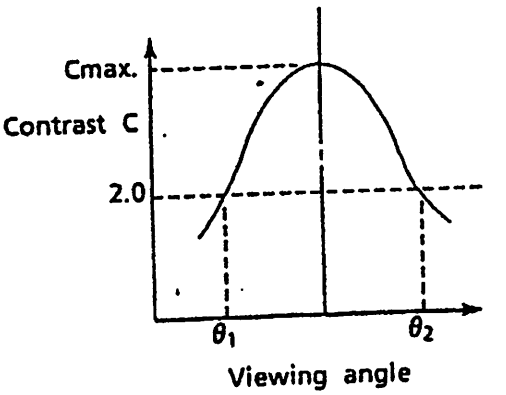
Maximum viewing angle: 6 o'clock ($\varnothing$ = 0°)
T_A = 25°C, V_{opr} = 4.75 V

Item	Symbol	Conditions	Min.	Typ.	Max.	Remarks
Viewing angle	$\theta_2 - \theta_1$	$C \geq 2.0$, $\varnothing = 0°$	35	–	–	See Notes 1 and 2.
Contrast	C	$\theta = 25°$, $\varnothing = 0°$	5	8	–	See Note 3.
Rise time	t_{on}	$\theta = 25°$, $\varnothing = 0°$	–	60 ms	70 ms	See Note 4.
Fall time	t_{off}	$\theta = 25°$, $\varnothing = 0°$	–	150 ms	170 ms	See Note 4.

Note 1: Definition of angles $\varnothing$ and θ

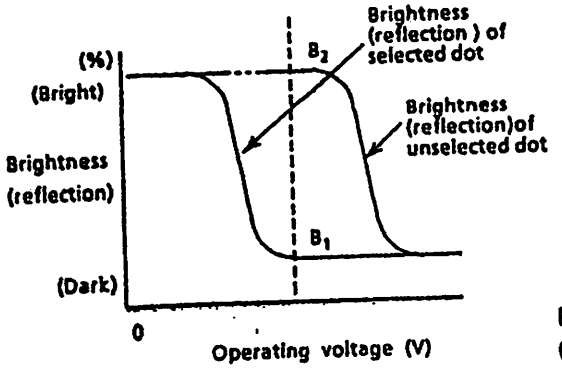


Note 2: Definition of viewing angles θ_1 and θ_2

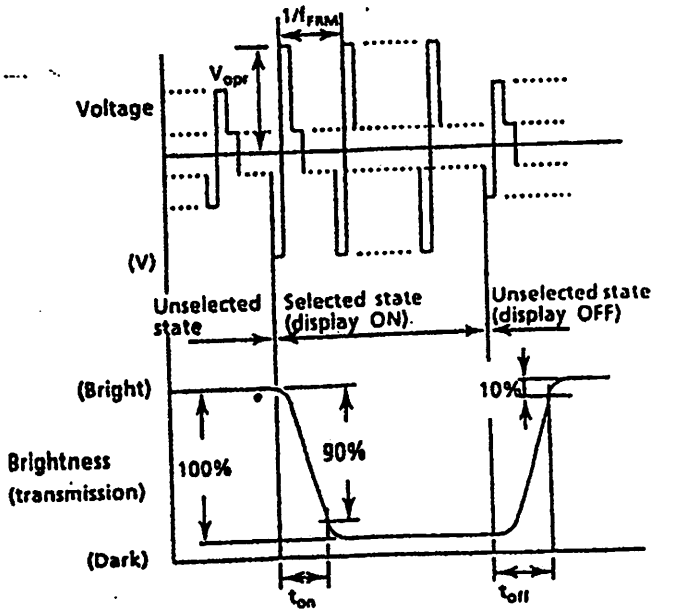


Note 3: Definition of contrast C

$$C = \frac{\text{Brightness (reflection) of unselected dot (B2)}}{\text{Brightness (reflection) of selected dot (B1)}}$$



Note 4: Definition of response time



V_{opr} : Operating voltage (V)
f_{FRM} : Frame frequency (Hz)
t_{on} : Response time (rise)(ms)
t_{off} : Response time (fall)(ms)

1.7.2 Recommended operating voltage

The viewing angle and screen contrast of the LCD panel can be varied by changing the liquid-crystal operating voltage (V_{opr}), that is V_{LC}.

The optical characteristics is influenced by an ambient temperature. The recommended value of V_{opr} for an ambient temperatures are shown below.

Temperature (℃)	0	10	25	40	50
Voltage V_{opr} (V)	5.00	4.90	4.75	4.60	4.50

$V_{opr} = V_{DD} - V_{LC}$

OPERATING INSTRUCTIONS

2.1 Terminal Functions

Table 1 Terminal functions

Signal name	No. of terminals	I/O	Destination	Function
DB_0 to DB_3	4	I/O	MPU	Tristate bidirectional lower four data buses: Data is read from the module to the MPU or written to the module from the MPU through the buses. If the interface data is 4 bits, the signals are not used.
DB_4 to DB_7	4	I/O	MPU	Tristate bidirectional upper four data buses: Data is read from the module to the MPU or written to the module from the MPU through the buses. DB_7 is also used as a busy flag.
E	1	Input	MPU	Operation start signal: The signal activates data write or read.
R/$\overline{W}$	1	Input	MPU	Read (R) and Write ($\overline{W}$) selection signals 0 : Write 1 : Read
RS	1	Input	MPU	Register selection signals 0 : Instruction register (Write) Busy flag and address counter (Read) 1 : Data register (Write and Read)
V_{LC}	1	–	Power supply	Power supply terminal for driving liquid crystal display: The screen contrast can be varied by changing V_{LC}.
V_{DD}	1	–	Power supply	+5 V
V_{SS}	1	–	Power supply	Ground terminal: 0 V

2.2 Basic Operations

2.2.1 Registers

The controller has two kinds of eight-bit registers: the instruction register (IR) and the data register (DR). They are selected by the register select (RS) signal as shown in Table 2.

The IR stores instruction codes such as Display Clear and Cursor Shift, and the address information of display data RAM (DD RAM) and character generator RAM (CG RAM). They can be written from the MPU, but cannot be read to the MPU.

The DR temporarily stores data to be written into DD RAM or CG RAM, or read from DD RAM or CG RAM. When data is written into DD RAM or CG RAM from the MPU, the data in the DR is automatically written into DD RAM or CG RAM by internal operation. However, when data is read from DD RAM or CG RAM, the necessary data address is written into the IR. The specified data is read out to the DR and then the MPU reads it from the DR. After the read operation, the next address is set and DD RAM or CG RAM data at the address is read into the DR for the next read operation.

Table 2 Register selection

RS	R/$\overline{W}$	Operation
0	0	IR selection, IR write. Internal operation : Display clear
0	1	Busy flag (DB_7) and address counter (DB_0 to DB_6) read
1	0	DR selection, DR write. Internal operation : DR to DD RAM or CG RAM
1	1	DR selection, DR read. Internal operation : DD RAM or CG RAM to DR

2.2.2 Busy flag (BF)

The flag indicates whether the module is ready to accept the next instruction. As shown in Table 2, the signal is output to DB_7 if RS = 0 and R/$\overline{W}$ = 1. If the value is 1, the module is working internally and the instruction cannot be accepted. If the value is 0, the next instruction can be written. Therefore, the flag status needs to be checked before executing an instruction. If an instruction is executed without checking the flag status, wait for more than the execution time shown by 2.4 Instruction Outline.

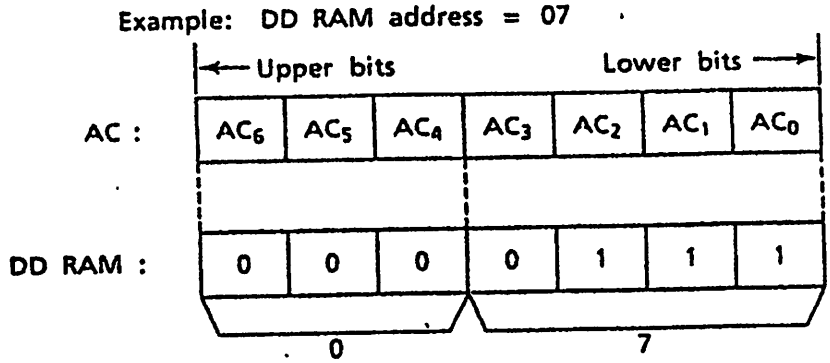
2.2.3 Address counter (AC)

The counter specifies an address when data is written into DD RAM or CG RAM and the data stored in DD RAM or CG RAM is read out. If an Address Set instruction (for DD RAM or CG RAM) is written in the IR, the address information is transferred from the IR to the AC. When display data is written into or read from DD RAM or CG RAM, the AC is automatically incremented or decremented by one according to the Entry Mode Set. The contents of the AC are output to DB_0 to DB_6 as shown in Table 2 if RS = 0 and $R/\overline{W} = 1$.

2.2.4 Display data RAM (DD RAM)

DD RAM has a capacity of up to 80 × 8 bits and stores display data of 80 eight-bit character codes. Some storage areas of DD RAM which are not used for display can be used as general data RAM.

A DD RAM address to be set in the AC is expressed in hexadecimal form as follows.



00_H to $0F_H$ of the DD RAM address is set in the line 1, and 40_H to $4F_H$ in the line 2.

Note : The addresses in the digit 16 of line 1 and the digit 1 of line 2 are not consecutive.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	DD RAM address
Line 2	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	

If the display is shifted, DD RAM address 00_H to 27_H are displayed in line 1 and 40_H to 67_H in line 2. The following figures are examples of display shifts.

*Left shift

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	DD RAM address
Line 2	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	4F	50	

*Right shift

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	Display digit
Line 1	27	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	DD RAM address
Line 2	67	40	41	42	43	44	45	46	47	48	49	4A	4B	4C	4D	4E	

2.5 Character generator ROM (CG ROM)

Character generator ROM generates 192 types of 5 x 7 dot-matrix character patterns from eight-bit character codes.

Table 3 shows the correspondence between the CG ROM character codes and character patterns.

2.6 Character generator RAM (CG RAM)

CG RAM is used to create character patterns freely by programming. Eight types of character patterns can be written.

Table 4 shows the character patterns created from CG RAM addresses and data. To display a created character pattern, the character code in the left column of the table is written into DD RAM corresponding to the display position (digit). The areas not used for display are available as general data RAM.

Table 3 Correspondence between character codes and character patterns

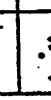
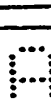
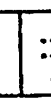
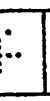
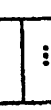
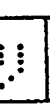
Upper bit 4 bit / Lower 4 bit	0 0000	2 0010	3 0011	4 0100	5 0101	6 0110	7 0111	1010	1011	1100	1101	1110	1111
×××0000	CG RAM (1)												
×××0001	(2)												
×××0010	(3)												
×××0011	(4)												
×××0100	(5)												
×××0101	(6)												
×××0110	(7)												
×××0111	(8)												
×××1000	(1)												
×××1001	(2)												
×××1010	(3)												
×××1011	(4)												
×××1100	(5)												
×××1101	(6)												
×××1110	(7)												
×××1111	(8)												

Table 4 Relationships between CG RAM addresses and character codes (DD RAM) and character patterns (CG RAM data)

Character code (DD RAM data) 7 6 5 4 3 2 1 0 ← Upper bit Lower bit →	CG RAM address 5 4 3 ← Upper bit	2 1 0 Lower bit →	Character pattern (CG RAM data) 7 6 5 4 3 2 1 0 ← Upper bit Lower bit →
0 0 0 0 * 0 0 0	0 0 0	0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	* * * 1 1 1 1 1 0 1 0 0 0 1 1 0 0 0 1 1 1 1 1 0 0 1 1 0 0 1 0 0 1 0 1 0 0 0 1 * * * 0 0 0 0 0
0 0 0 0 * 0 0 1	0 0 1	0 0 0 0 0 1 0 1 0 0 1 1 1 0 0 1 0 1 1 1 0 1 1 1	* * * 1 0 0 0 1 0 1 0 1 0 1 1 1 1 1 0 0 1 0 0 1 1 1 1 1 0 0 1 0 0 0 0 1 0 0 * * * 0 0 0 0 0
0 0 0 0 * 1 1 1	1 1 1	0 0 0 0 0 1 1 0 0 1 0 1 1 1 0 1 1 1	* * * * * *

Example of character pattern (R)

← Cursor position

Example of character pattern (¥)

Notes:
- In CG RAM data, 1 corresponds to Selection and 0 to Non-selection on the display.
- Character code bits 0 to 2 and CG RAM address bits 3 to 5 correspond with each other (three bits, eight types).
- CG RAM address bits 0 to 2 specify a line position for a character pattern. Line 8 of a character pattern is the cursor position where the logical sum of the cursor and CG RAM data is displayed. Set the data of line 8 to 0 to display the cursor. If the data is changed to 1, one bit lights, regardless of the cursor.

The character pattern column positions correspond to CG RAM data bits 0 to 4 and bit 4 comes to the left end. CG RAM data bits 5 to 7 are not displayed but can be used as general data RAM.

When reading a character pattern from CG RAM, set to 0 all of character code bits 4 to 7. Bits 0 to 2 determine which pattern will be read out. Since bit 3 is not valid, 00_H and 08_H select the same character.

3 Timing Characteristics

2.3.1 Write timing characteristics

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, T_A = 0°C to 50°C

Item		Symbol	Standard		Unit
			Min.	Max.	
Enable cycle time		$t_{CYC}E$	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er},t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$ — E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data setup time		t_{DSW}	195	–	ns
Data hold time		t_H	10	–	ns

Write operation

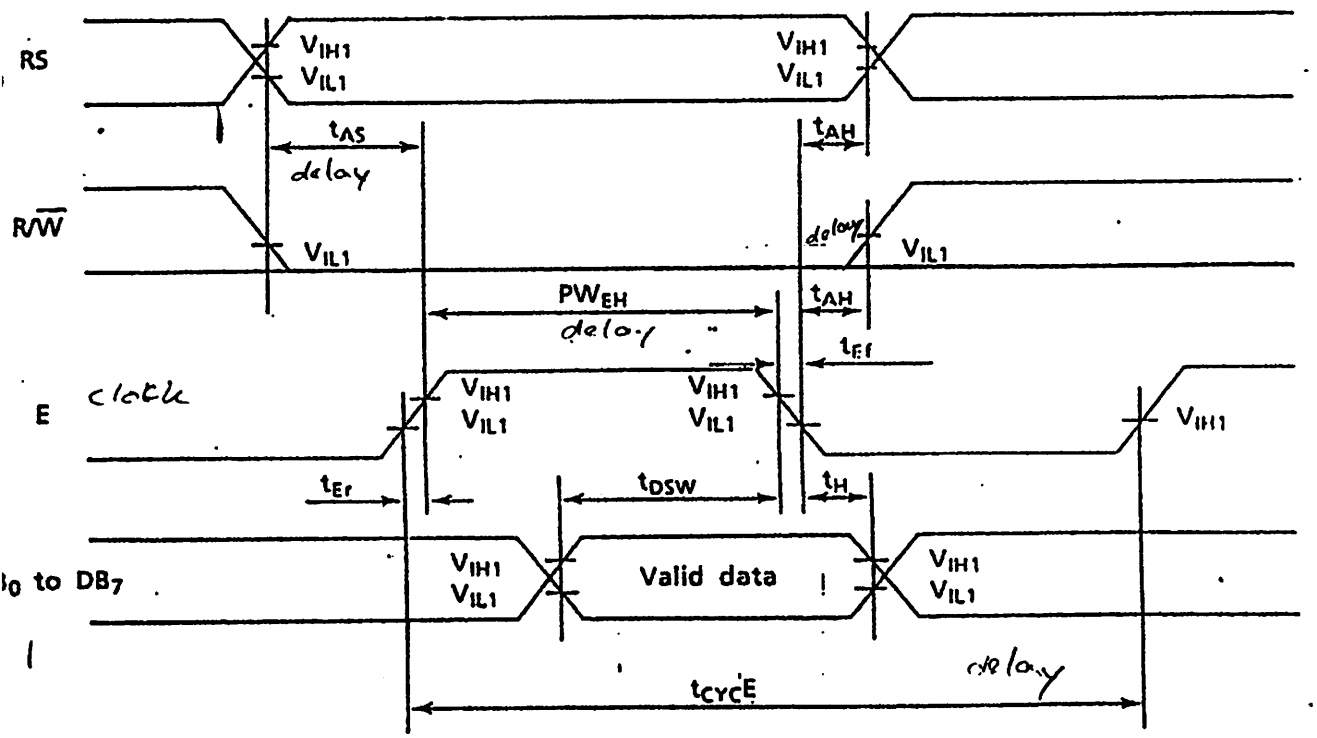


Figure 3 Data write from MPU to module

2.3.2 Read timing characteristics

V_{DD} = 5.0 V ± 5%, V_{SS} = 0 V, T_A = 0°C to 50°C

Item		Symbol	Standard		Unit
			Min.	Max.	
Enable cycle time		$t_{CYC}E$	1000	–	ns
Enable pulse width	High level	PW_{EH}	450	–	ns
Enable rise and fall time		t_{Er}, t_{Ef}	–	25	ns
Setup time	RS, R/$\overline{W}$—E	t_{AS}	140	–	ns
Address hold time		t_{AH}	10	–	ns
Data delay time		t_{DDR}	–	320	ns
Data hold time		t_{H}	20	–	ns

Read operation

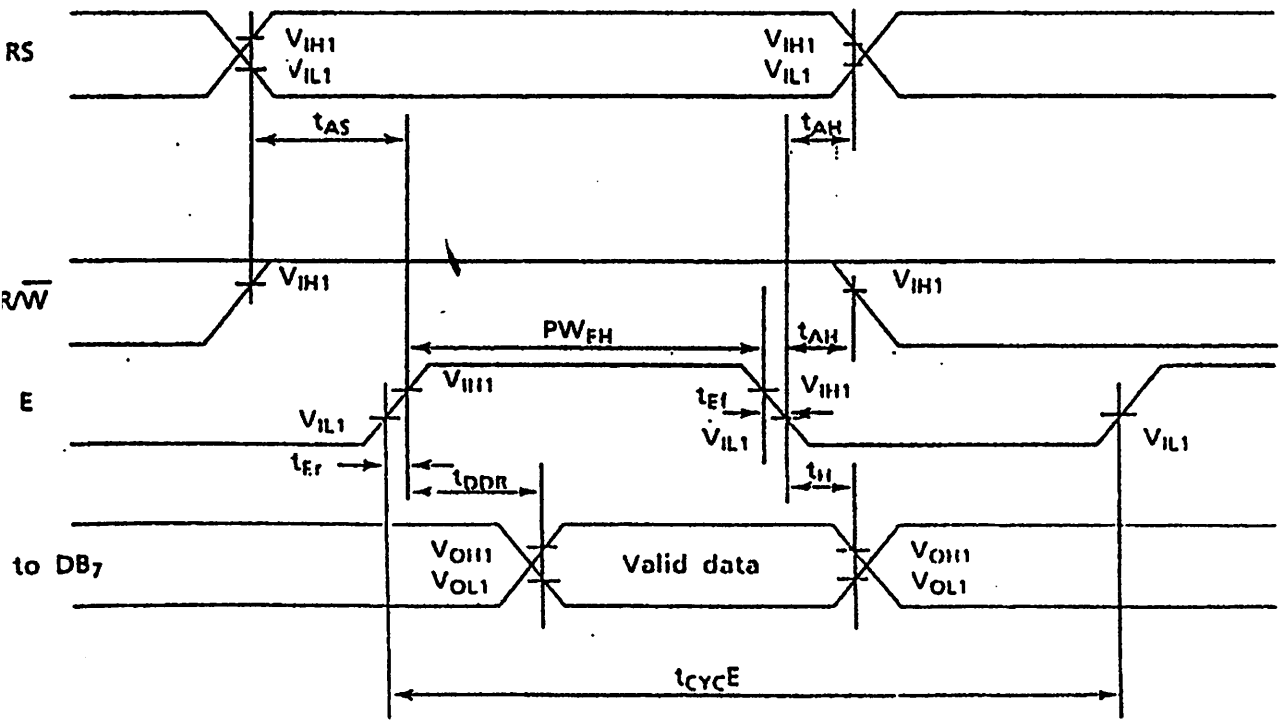


Figure 4 Data read from module to MPU

Instruction Outline

Table 5 List of instructions

Instruction	Code										Function	Execution time
	RS	R/W	DB_7	DB_6	DB_5	DB_4	DB_3	DB_2	DB_1	DB_0		
Display clear	0	0	0	0	0	0	0	0	0	1	Clears all display and returns cursor to home position (address 0)	1.64 ms
Cursor Home	0	0	0	0	0	0	0	0	1	*	Returns cursor to home position. Shifted display returns to home position and DD RAM contents do not change.	1.64 ms
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Sets direction of cursor movement and whether display will be shifted when data is written or read	40 µs
Display ON / OFF control	0	0	0	0	0	0	1	D	C	B	Turns ON/OFF total display (D) and cursor (C), and makes cursor position column start blinking (B)	40 µs
Cursor/Display Shift	0	0	0	0	0	1	S/C	R/L	*	*	Moves cursor and shifts display without changing DD RAM contents	40 µs
Function Set	0	0	0	0	1	DL	1	*	*	*	Sets interface data length (DL)	40 µs
CG RAM Address Set	0	0	0	1	A_{CG}						Sets CG RAM address to start transmitting or receiving CG RAM data	40 µs
DD RAM Address Set	0	0	1	A_{DD}							Sets DD RAM address to start transmitting or receiving DD RAM data	40 µs
BF/Address Read	0	1	BF	AC							Reads BF indicating module in internal operation and AC contents (used for both CG RAM and DD RAM)	0 µs
) Data Write to CG RAM or DD RAM	1	0	Write Data								Writes data into DD RAM or CG RAM	40 µs
) Data Read from CG RAM or DD RAM	1	1	Read Data								Reads data from DD RAM or CG RAM	40 µs

: Invalid bit
G : CG RAM address
D : DD RAM address

I/D = 1 : Increment
I/D = 0 : Decrement

S = 1 : Display shift
S = 0 : No display shift

D = 1 : Display ON
D = 0 : Display OFF

C = 1 : Cursor ON
C = 0 : Cursor OFF

B = 1 : Blink ON
B = 0 : Blink OFF

S/C = 1 : Display shift
S/C = 0 : Cursor movement

R/L = 1 : Right shift
R/L = 0 : Left shift

DL = 1 : 8 bits
DL = 0 : 4 bits

BF = 1 : Internal operation in progress
BF = 0 : Instruction can be accepted

Instruction Details

Display Clear

	RS	R/W	DB_7							DB_0
Code	0	0	0	0	0	0	0	0	0	1

Display Clear clears all display and returns cursor to home position (address 0).

Space code 20 (hexadecimal) is written into all the addresses of DD RAM, and DD RAM address 0 is set to the AC. If shifted, the display returns to the original position. After execution of the Display Clear instruction, the entry mode is incremented.

Note : When executing the Display Clear instruction, follow the restrictions listed in Table 6.

) Cursor Home

	RS	R/W	DB_7							DB_0
Code	0	0	0	0	0	0	0	0	1.	*

* : Invalid bit

Cursor Home returns cursor to home position (address 0).

DD RAM address 0 is set to the AC. The cursor returns to the home position. If shifted, the display returns to the original position. The DD RAM contents do not change. If the cursor or blinking is ON, it returns to the left side.

Note : When executing the Cursor Home instruction, follow the restrictions listed in Table 6.

Table 6 Restrictions on execution of Display Clear and Cursor Home instructions

Conditions of use	Restrictions
When executing the Display Clear or Cursor Home instruction when the display is shifted (after execution of Display Shift instruction)	The Cursor Home instruction should be executed again immediately after the Display Clear or Cursor Home instruction is executed. Do not leave an interval of a multiple of $400/f_{osc}$* second after the first execution. Example: 1.5 ms, 3 ms, 4.5 ms for f_{osc} = 270 kHz *f_{osc}: Oscillation frequency
When 23_H, 27_H, 63_H, or 67_H is used as a DD RAM address to execute Cursor Home instruction	Before executing the Cursor Home instruction, the data of the four DD RAM addresses given at the left should be read and saved. After execution, write the data again in DD RAM. (This restriction is necessary to prevent the contents of the DD RAM addresses from being destroyed after the Cursor Home instruction has been executed.)

3) Entry Mode Set

	RS	R/$\overline{W}$	DB_7							DB_0
Code	0	0	0	0	0	0	0	1	I/D	S

Entry Mode Set sets the direction of cursor movement and whether display will be shifted.

I/D : The DD RAM address is incremented or decremented by one when a character code is written into or read from DD RAM. This is also true for writing into or reading from CG RAM.

When I/D = 1, the address is incremented by one and the cursor or blink moves to the right.

When I/D = 0, the address is decremented by one and the cursor or blink moves to the left.

S : If S = 1, the entire display is shifted either to the right or left for writing into DD RAM. The cursor position does not change, only the display moves. There is no display shift for reading from DD RAM.

When S = 1 and I/D = 1, the display shifts to the left.
When S = 1 and I/D = 0, the display shifts to the right.

If S = 0, the display does not shift.

4) Display ON/OFF Control

	RS	R/$\overline{W}$	DB_7							DB_0
Code	0	0	0	0	0	0	1	D	C	B

Display ON/OFF Control turns the total display and the cursor ON and OFF, and makes the cursor position start blinking. Cursor ON/OFF and blinking is done at the column indicated by the specified DD RAM address by the AC.

D : When D = 1, the display is turned ON.
When D = 0, the display is turned OFF.
If D = 0 is used, display data remains in DD RAM. Change 0 to 1 to display data.

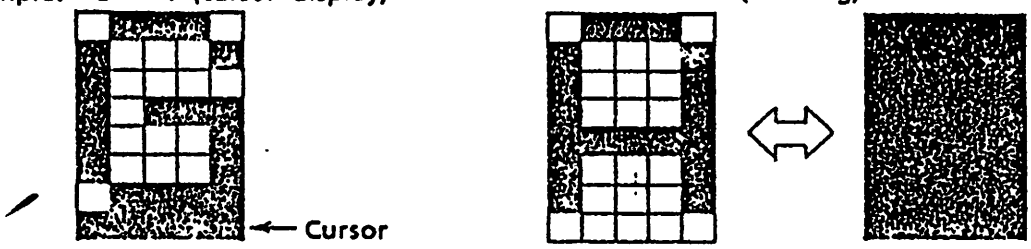
C : When C = 1, the cursor is displayed.
When C = 0, the cursor is not displayed.

The cursor is displayed in the dot line below the 5 x 7 dot-matrix character fonts. If the cursor is OFF, display data is written into DD RAM in the order specified by I/D.

B : When B = 1, the character at the cursor position starts blinking.
When B = 0, it does not blink.

For blinking, all-black dots and the character are switched about every 0.4 seconds. The cursor and blinking can be set at the same time.

Example: C = 1 (cursor display) B = 1 (blinking)



) Cursor/Display Shift

	RS	R/$\overline{W}$	DB_7							DB_0
Code	0	0	0	0	0	1	S/C	R/L	*	*

* :Invalid bit

Cursor/Display Shift moves the cursor and shifts the display without changing the DD RAM contents.

The cursor position and the AC contents match. This instruction is available for display correction and retrieval because the cursor position or display can be shifted without writing or reading display data. Since the DD RAM capacity is 40-character and two lines, the cursor is shifted from digit 40 of line 1 to digit 1 of line 2. Displays of lines 1 and 2 are shifted at the same time. Therefore, the display pattern of line 2 is not shifted to line 1.

S/C	R/L	Operation
0	0	The cursor position is shifted to the left (the AC decrements one).
0	1	The cursor position is shifted to the right (the AC increments one).
1	0	The entire display is shifted to the left with the cursor.
1	1	The entire display is shifted to the right with the cursor.

Note: If only display shift is done, the AC contents do not change.

i) **Function Set**

	RS	R/$\overline{W}$	DB_7							DB_0	
Code	0	0	0	0	1	DL	1	*	*	*	* : Invalid bit

R/$\overline{W}$ = 0

Function Set sets the interface data length.

DL : Interface data length

When DL = 1, the data length is set at eight bits (DB_7 to DB_0).

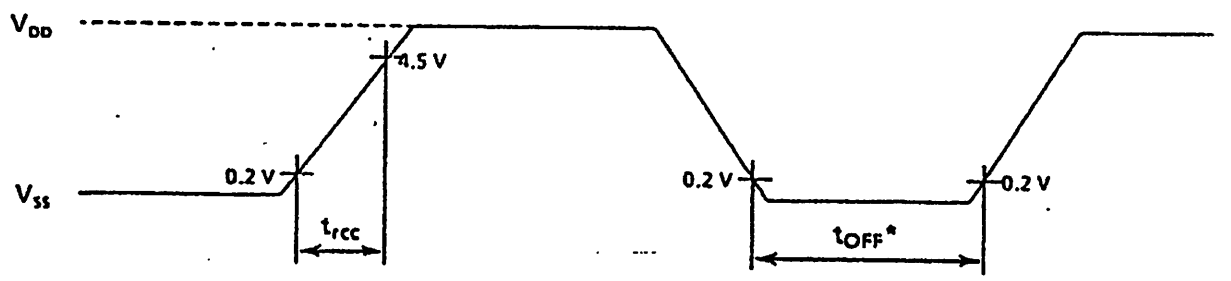
When DL = 0, the data length is set at four bits (DB_7 to DB_4).

The upper four bits are transferred first, then the lower four bits follow.

The Function Set instruction must be executed prior to all other instructions except for Busy Flag/Address Read. If another instruction is executed first, no function instruction except changing the interface data length can be executed.

Remarks: Initialization

The system is automatically initialized at power-on if the following power supply conditions are satisfied.



*t_{OFF}: Time when power supply is OFF if cut instantaneously or turned ON and OFF repeatedly

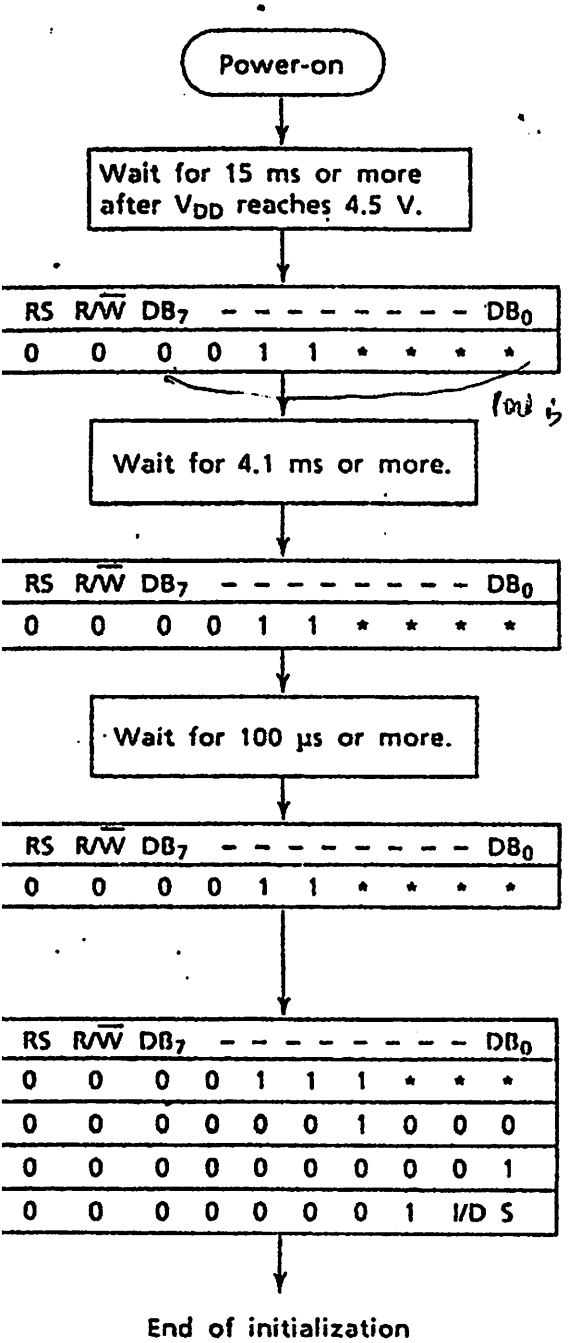
The following instructions are executed for initialization.

- 5 x 7 dot-matrix character font: 1/8 duty
- Display clear
- Function Set — DL = 1: Interface data length: 8 bits
- Display ON/OFF Control — D = 0: Display OFF; C = 0: Cursor OFF; B = 0: Blink OFF
- Entry mode — I/O = 1: Increment; S = 0: No display shift

Since the condition is not suitable for the M1632, further function setting is necessary.

If automatic initialization is not executed because the above power supply conditions are not satisfied, use the instruction from next page on.

i) **Interface data length : Eight bits**



The busy flag cannot be checked before the following instruction: Function Set (interface data length: 8 bits)

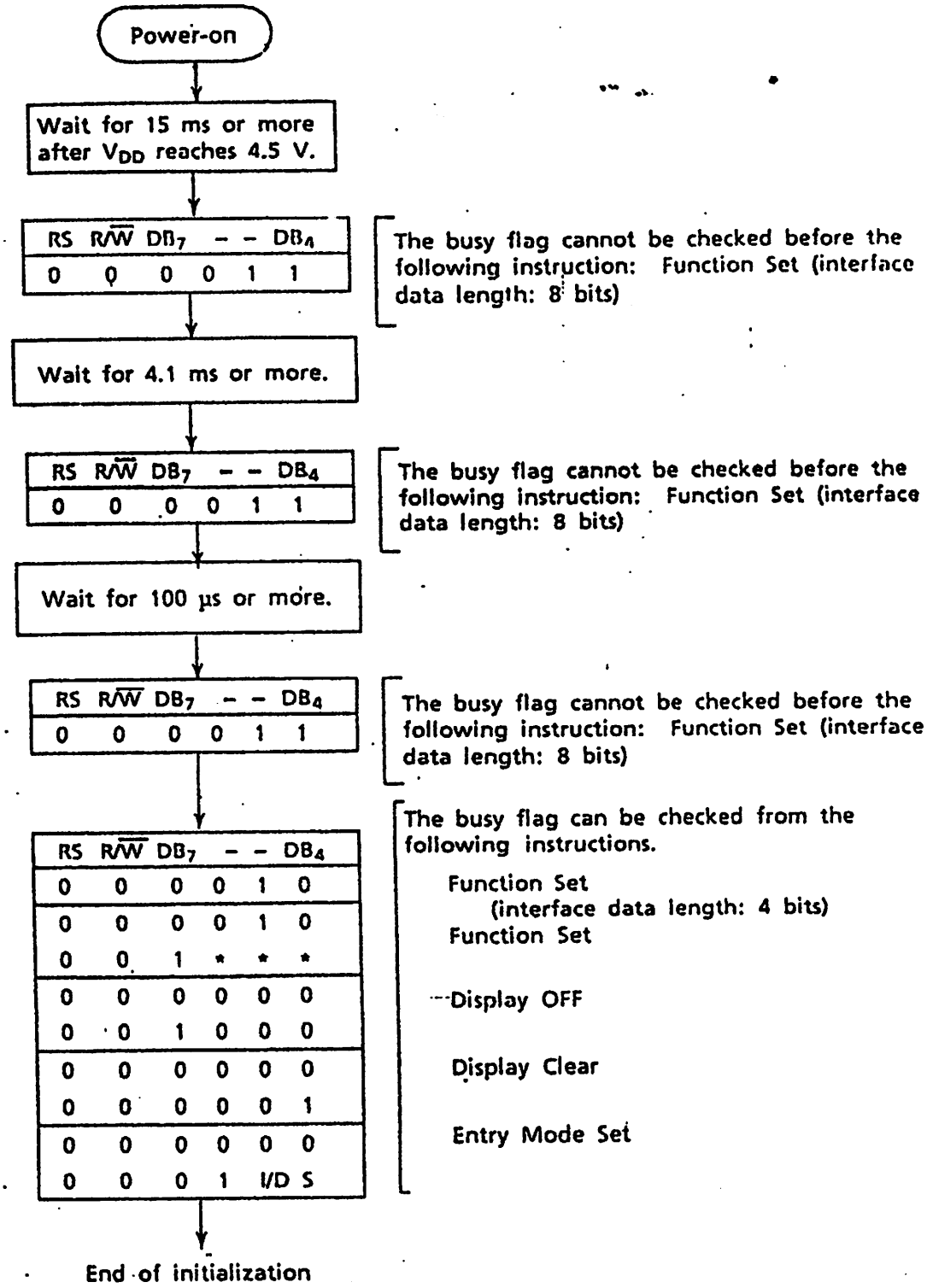
The busy flag cannot be checked before the following instruction: Function Set (interface data length: 8 bits)

The busy flag cannot be checked before the following instruction: Function Set (interface data length: 8 bits)

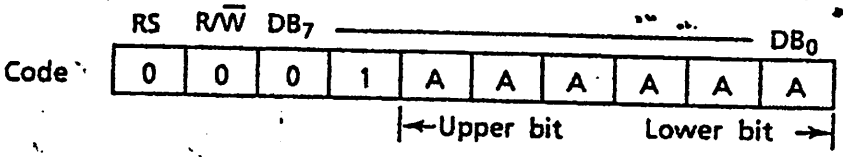
The busy flag can be checked from the following instructions.

- Function Set
- Display OFF
- Display Clear
- Entry Mode Set

Interface data length: Four bits

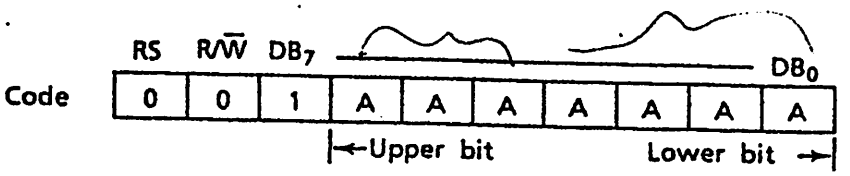


') CG RAM Address Set



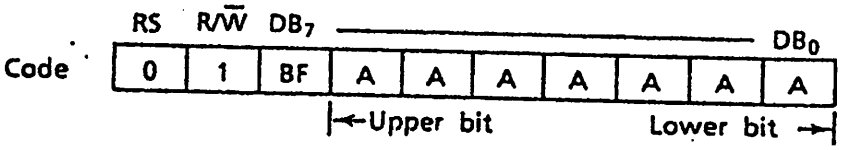
CG RAM addresses expressed as binary AAAAAA are set to the AC. Then data in CG RAM is written from or read to the MPU.

) DD RAM Address Set



DD RAM addresses expressed as binary AAAAAAA are set to the AC. Then data in DD RAM is written from or read to the MPU. The addresses used for display in line 1 (AAAAAAA) are 00_H to 27_H and those for line 2 (AAAAAAA) are 40_H to 67_H.

Busy Flag/Address Read



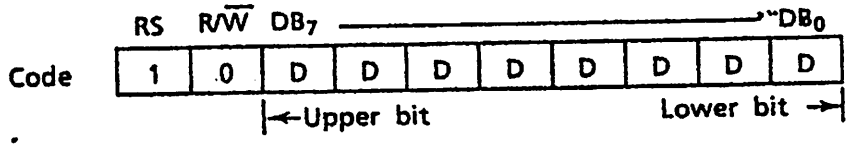
The BF signal is read out, indicating that the module is working internally because of the previous instruction.

When BF = 1, the module is working internally and the next instruction cannot be accepted until the BF value becomes 0.

When BF = 0, the next instruction can be accepted.

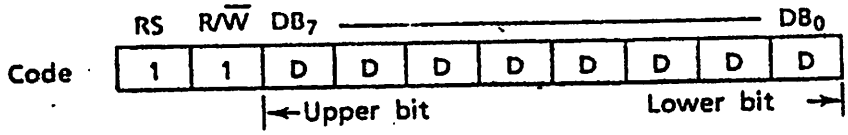
Therefore, make sure that BF = 0 before writing the next instruction. The AC values of binary AAAAAAA are read out at the same time as reading the busy flag. The AC addresses are used for both CG RAM and DD RAM but the address set before execution of the instruction determines which address is to be used.

) Data Write to CG RAM or DD RAM



Binary eight-bit data DDDDDDDD is written into CG RAM or DD RAM. The CG RAM Address Set instruction of (7) or the DD RAM Address Set instruction of (8) before this instruction selects either RAM. After the write operation, the address and display shift are determined by the entry mode setting.

1) Data Read from CG RAM or DD RAM



Binary eight-bit data DDDDDDDD is read from CG RAM or DD RAM. The CG RAM Address Set instruction of (7) or the DD RAM Address Set instruction of (8) before this instruction selects either RAM. In addition, either instruction (7) or (8) must be executed immediately before this instruction. If no address set instruction is executed before a read instruction, the first data read becomes invalid. If read instructions are executed consecutively, data is normally read from the second time. However, if the cursor is shifted by the Cursor Shift instruction when reading DD RAM, there is no need to execute an address set instruction because the Cursor Shift instruction does this.
After the read operation, the address is automatically incremented or decremented by one according to the entry mode, but the display is not shifted.

Note : The AC is automatically incremented or decremented by one according to the entry mode after a write instruction is executed to write data in CG RAM or DD RAM. However, the data of the RAM selected by the AC are not read out even if a read instruction is executed immediately afterwards.

Correct data is read out under the following conditions.

- An address set instruction is executed immediately before readout.
- For DD RAM, the Cursor Shift instruction is executed immediately before readout.
- The second, or later, instruction is executed in consecutive execution of read instructions.

Examples of Instruction Use

1) Interface data length: Eight bits

Instruction	Display	Operation
Power-on RS: — R/W: — DB_7 —— DB_0: —		The built-in reset circuit initializes the module.
Function Set ✓ RS: 0 R/W: 0 DB_7 —— DB_0: 0 0 1 1 1 * * *		The interface data length is set to 8 bits. The character format becomes 5 x 7 dot-matrix at 1/16 duty cycle.
Display ON/OFF Control RS: 0 R/W: 0 DB_7 —— DB_0: 0 0 0 0 1 1 1 0		The display and cursor are turned ON, but nothing is displayed.
Entry Mode Set RS: 0 R/W: 0 DB_7 —— DB_0: 0 0 0 0 0 1 1 0		The address is incremented by one and the cursor shifts to the right in a write operation to internal RAM. The display is not shifted.
Write to CG RAM or DD RAM RS: 1 R/W: 0 DB_7 —— DB_0: 0 1 0 0 1 1 0 0	L_	L is written. The AC is incremented by one and the cursor shifts to the right.
Write to CG RAM or DD RAM RS: 1 R/W: 0 DB_7 —— DB_0: 0 1 0 0 0 0 1 1	LC_	C is written.
⋮	⋮	
Write to CG RAM or DD RAM RS: 1 R/W: 0 DB_7 —— DB_0: 0 0 1 1 0 0 1 0	LCD MODULE M1632	2 is written in digit 16. Cursor disappears.

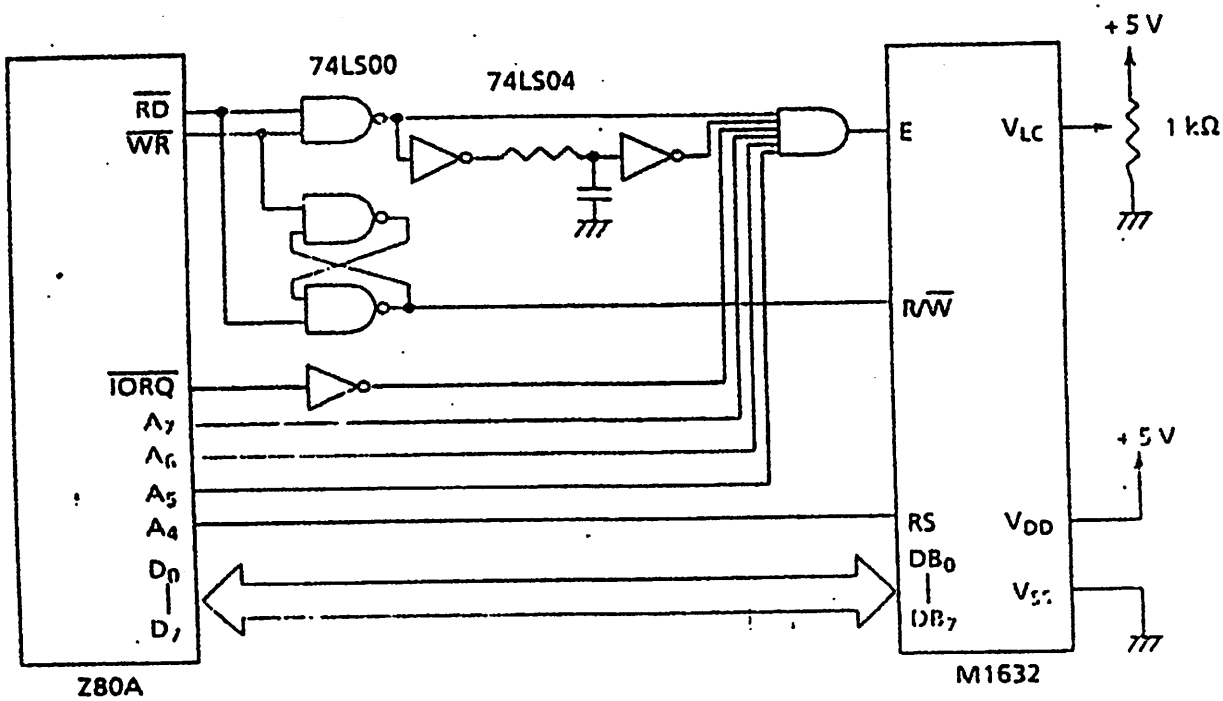
No.	Instruction	Display	Operation
9	DD RAM address set RS: 0, R/W: 0, DB7 —— DB0: 1 1 0 0 0 0 0 0	LCD MODULE M1632 _	The DD RAM address is set so that the cursor appears at digit 1 of line 2.
10	Write to CG RAM or DD RAM RS: 1, R/W: 0, DB7 —— DB0: 0 0 1 1 0 0 0 1	LCD MODULE M1632 1_	1 is written.
11	Write to CG RAM or DD RAM RS: 1, R/W: 0, DB7 —— DB0: 0 0 1 1 0 1 1 0	LCD MODULE M1632 16_	6 is written.
12	⋮	⋮	
13	Write to CG RAM or DD RAM RS: 1, R/W: 0, DB7 —— DB0: 0 1 0 1 0 0 1 1	LCD MODULE M1632 16DIGITS,2LINES	S is written.
14	DD RAM address set RS: 0, R/W: 0, DB7 —— DB0: 1 0 0 0 0 0 0 0	LCD MODULE M1632 16DIGITS,2LINES	The cursor returns to the home position.
15	Display clear RS: 0, R/W: 0, DB7 —— DB0: 0 0 0 0 0 0 0 1	_	All the display disappears and the cursor remains at the home position.
16	⋮	⋮	

(2) Interface data length: Four bits

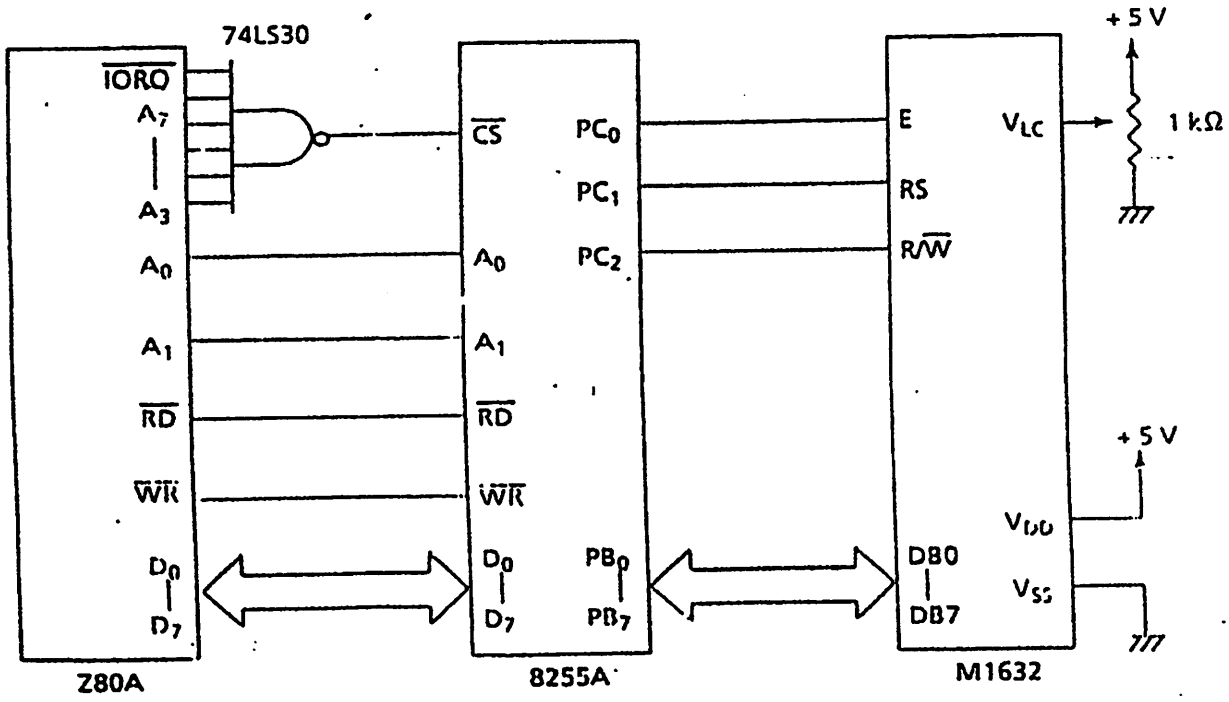
No.	Instruction	Display	Operation
1	Power-on RS: ／ R/W: ／ DB7 — DB4: ／		The built-in reset circuit initializes the module.
2	Function Set RS: 0 R/W: 0 DB7 — DB4: 0 0 1 0		Four-bit operation mode is set. *Eight-bit operation mode is set by initialization, and the instruction is executed only once.
3	Function Set RS: 0 R/W: 0 DB7 — DB4: 0 0 1 0 RS: 0 R/W: 0 DB7 — DB4: 1 * * *		The 4-bit operation mode, 1/16 duty cycle, and 5 x 7 dot-matrix character format are selected. Then 4-bit operation mode starts.
4	Display ON/OFF Control RS: 0 R/W: 0 DB7 — DB4: 0 0 0 0 RS: 0 R/W: 0 DB7 — DB4: 1 1 1 0	_	The display and cursor are turned ON, but nothing is displayed.
5	Entry Mode Set RS: 0 R/W: 0 DB7 — DB4: 0 0 0 0 RS: 0 R/W: 0 DB7 — DB4: 0 1 1 0	_	The address is incremented by one and the cursor shifts to the right in a write operation to internal RAM. The display is not shifted.
6	Write to CG RAM or DD RAM. RS: 1 R/W: 0 DB7 — DB4: 0 1 0 0 RS: 1 R/W: 0 DB7 — DB4: 1 1 0 0	L_	L is written. the AC is incremented by one and the cursor shifts to the right.

MPU Connection Diagrams

2.7.1 Z80A



2.7.2 Z80A and 8255A



19-0410; Rev 3; 7/96

MAXIM

±15kV ESD-Protected, Slew-Rate-Limited, Low-Power, RS-485/RS-422 Transceivers

General Description

The MAX481E, MAX483E, MAX485E, MAX487E–MAX491E, and MAX1487E are low-power transceivers for RS-485 and RS-422 communications in harsh environments. Each driver output and receiver input is protected against ±15kV electrostatic discharge (ESD) shocks, without latchup. These parts contain one driver and one receiver. The MAX483E, MAX487E, MAX488E, and MAX489E feature reduced slew-rate drivers that minimize EMI and reduce reflections caused by improperly terminated cables, thus allowing error-free data transmission up to 250kbps. The driver slew rates of the MAX481E, MAX485E, MAX490E, MAX491E, and MAX1487E are not limited, allowing them to transmit up to 2.5Mbps.

These transceivers draw as little as 120µA supply current when unloaded or when fully loaded with disabled drivers (see *Selection Table*). Additionally, the MAX481E, MAX483E, and MAX487E have a low-current shutdown mode in which they consume only 0.5µA. All parts operate from a single +5V supply.

Drivers are short-circuit current limited, and are protected against excessive power dissipation by thermal shutdown circuitry that places their 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 MAX487E and MAX1487E feature quarter-unit-load receiver input impedance, allowing up to 128 transceivers on the bus. The MAX488E–MAX491E are designed for full-duplex communications, while the MAX481E, MAX483E, MAX485E, MAX487E, and MAX1487E are designed for half-duplex applications. For applications that are not ESD sensitive see the pin- and function-compatible MAX481, MAX483, MAX485, MAX487–MAX491, and MAX1487.

Applications

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

Features

- **ESD Protection: ±15kV—Human Body Model**
- **Slew-Rate Limited for Error-Free Data Transmission (MAX483E/487E/488E/489E)**
- **Low Quiescent Current:**
  120µA (MAX483E/487E/488E/489E)
  230µA (MAX1487E)
  300µA (MAX481E/485E/490E/491E)
- **-7V to +12V Common-Mode Input Voltage Range**
- **Three-State Outputs**
- **30ns Propagation Delays, 5ns Skew (MAX481E/485E/490E/491E/1487E)**
- **Full-Duplex and Half-Duplex Versions Available**
- **Allows up to 128 Transceivers on the Bus (MAX487E/MAX1487E)**
- **Current Limiting and Thermal Shutdown for Driver Overload Protection**

Ordering Information

PART	TEMP. RANGE	PIN-PACKAGE
MAX481ECPA	0°C to +70°C	8 Plastic DIP
MAX481ECSA	0°C to +70°C	8 SO
MAX481EEPA	-40°C to +85°C	8 Plastic DIP
MAX481EESA	-40°C to +85°C	8 SO

Ordering Information continued on last page.

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
MAX481E	Half	2.5	No	Yes	Yes	300	32	8
MAX483E	Half	0.25	Yes	Yes	Yes	120	32	8
MAX485E	Half	2.5	No	No	Yes	300	32	8
MAX487E	Half	0.25	Yes	Yes	Yes	120	128	8
MAX488E	Full	0.25	Yes	No	No	120	32	8
MAX489E	Full	0.25	Yes	No	Yes	120	32	14
MAX490E	Full	2.5	No	No	No	300	32	8
MAX491E	Full	2.5	No	No	Yes	300	32	14
MAX1487E	Half	2.5	No	No	Yes	230	128	8

±15kV ESD-Protected, Slew-Rate-Limited, Low-Power, RS-485/RS-422 Transceivers

ABSOLUTE MAXIMUM RATINGS

Supply Voltage (V_{CC}) 12V
Control Input Voltage ($\overline{RE}$, DE) ..... -0.5V to (V_{CC} + 0.5V)
Driver Input Voltage (DI) ..... -0.5V to (V_{CC} + 0.5V)
Driver Output Voltage (Y, Z; A, B) ..... -8V to +12.5V
Receiver Input Voltage (A, B) ..... -8V to +12.5V
Receiver Output Voltage (RO) ..... -0.5V to (V_{CC} + 0.5V)
Continuous Power Dissipation (T_A = +70°C)
8-Pin Plastic DIP (derate 9.09mW/°C above +70°C) 727mW
14-Pin Plastic DIP (derate 10.00mW/°C above +70°C) ..800mW
8-Pin SO (derate 5.88mW/°C above +70°C) 471mW
14-Pin SO (derate 8.33mW/°C above +70°C) 667mW
Operating Temperature Ranges
MAX4_ _C_ _/MAX1487EC_ A ..... 0°C to +70°C
MAX4_ _E_ _/MAX1487EE_ A ..... -40°C to +85°C
Storage Temperature Range ..... -65°C to +160°C
Lead Temperature (soldering, 10sec) ..... +300°C

Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. These are stress ratings only, and functional operation of the device at these or any other conditions beyond those indicated in the operational sections of the specifications is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

DC ELECTRICAL CHARACTERISTICS

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

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Differential Driver Output (no load)	V_{OD1}					5	V
Differential Driver Output (with load)	V_{OD2}	R = 50Ω (RS-422)		2			V
		R = 27Ω (RS-485), Figure 8		1.5		5	
Change in Magnitude of Driver Differential Output Voltage for Complementary Output States	ΔV_{OD}	R = 27Ω or 50Ω, Figure 8				0.2	V
Driver Common-Mode Output Voltage	V_{OC}	R = 27Ω or 50Ω, Figure 8				3	V
Change in Magnitude of Driver Common-Mode Output Voltage for Complementary Output States	ΔV_{OD}	R = 27Ω or 50Ω, Figure 8				0.2	V
Input High Voltage	V_{IH}	DE, DI, $\overline{RE}$		2.0			V
Input Low Voltage	V_{IL}	DE, DI, $\overline{RE}$				0.8	V
Input Current	I_{IN1}	DE, DI, $\overline{RE}$				±2	µA
Input Current (A, B)	I_{IN2}	DE = 0V; V_{CC} = 0V or 5.25V, all devices except MAX487E/MAX1487E	V_{IN} = 12V			1.0	mA
			V_{IN} = -7V			-0.8	
		MAX487E/MAX1487E, DE = 0V, V_{CC} = 0V or 5.25V	V_{IN} = 12V			0.25	mA
			V_{IN} = -7V			-0.2	
Receiver Differential Threshold Voltage	V_{TH}	-7V ≤ V_{CM} ≤ 12V		-0.2		0.2	V
Receiver Input Hysteresis	ΔV_{TH}	V_{CM} = 0V			70		mV
Receiver Output High Voltage	V_{OH}	I_O = -4mA, V_{ID} = 200mV		3.5			V
Receiver Output Low Voltage	V_{OL}	I_O = 4mA, V_{ID} = -200mV				0.4	V
Three-State (high impedance) Output Current at Receiver	I_{OZR}	0.4V ≤ V_O ≤ 2.4V				±1	µA
Receiver Input Resistance	R_{IN}	-7V ≤ V_{CM} ≤ 12V, all devices except MAX487E/MAX1487E		12			kΩ
		-7V ≤ V_{CM} ≤ 12V, MAX487E/MAX1487E		48			kΩ

DC ELECTRICAL CHARACTERISTICS (continued)

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

PARAMETER	SYMBOL	CONDITIONS			MIN	TYP	MAX	UNITS
No-Load Supply Current (Note 3)	I_{CC}	MAX488E/MAX489E, DE, DI, $\overline{RE}$ = 0V or V_{CC}				120	250	µA
		MAX490E/MAX491E, DE, DI, $\overline{RE}$ = 0V or V_{CC}				300	500	
		MAX481E/MAX485E, $\overline{RE}$ = 0V or V_{CC}	DE = V_{CC}			500	900	
			DE = 0V			300	500	
		MAX1487E, $\overline{RE}$ = 0V or V_{CC}	DE = V_{CC}			300	500	
			DE = 0V			230	400	
		MAX483E/MAX487E, $\overline{RE}$ = 0V or V_{CC}	DE = V_{CC}	MAX483E		350	650	
				MAX487E		250	400	
			DE = 0V			120	250	
Supply Current in Shutdown	I_{SHDN}	MAX481E/483E/487E, DE = 0V, $\overline{RE}$ = V_{CC}				0.5	10	µA
Driver Short-Circuit Current, V_O = High	I_{OSD1}	-7V ≤ V_O ≤12V (Note 4)			35		250	mA
Driver Short-Circuit Current, V_O = Low	I_{OSD2}	-7V ≤ V_O ≤12V (Note 4)			35		250	mA
Receiver Short-Circuit Current	I_{OSR}	0V ≤ V_O ≤ V_{CC}			7		95	mA
ESD Protection		A, B, Y and Z pins, tested using Human Body Model				±15		kV

SWITCHING CHARACTERISTICS—MAX481E/MAX485E, MAX490E/MAX491E, MAX1487E

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

PARAMETER	SYMBOL	CONDITIONS		MIN	TYP	MAX	UNITS
Driver Input to Output	t_{PLH}	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF		10	40	60	ns
	t_{PHL}			10	40	60	
Driver Output Skew to Output	t_{SKEW}	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF			5	10	ns
Driver Rise or Fall Time	t_R, t_F	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	MAX481E, MAX485E, MAX1487E	3	20	40	ns
			MAX490EC/E, MAX491EC/E	5	20	25	
Driver Enable to Output High	t_{ZH}	Figures 11 and 13, C_L = 100pF, S2 closed			45	70	ns
Driver Enable to Output Low	t_{ZL}	Figures 11 and 13, C_L = 100pF, S1 closed			45	70	ns
Driver Disable Time from Low	t_{LZ}	Figures 11 and 13, C_L = 15pF, S1 closed			45	70	ns
Driver Disable Time from High	t_{HZ}	Figures 11 and 13, C_L = 15pF, S2 closed			45	70	ns
Receiver Input to Output	t_{PLH}, t_{PHL}	Figures 10 and 14, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	MAX481E, MAX485E, MAX1487E	20	60	200	ns
			MAX490EC/E, MAX491EC/E	20	60	150	
$\lvert t_{PLH} - t_{PHL} \rvert$ Differential Receiver Skew	t_{SKD}	Figures 10 and 14, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF			5		ns
Receiver Enable to Output Low	t_{ZL}	Figures 9 and 15, C_{RL} = 15pF, S1 closed			20	50	ns
Receiver Enable to Output High	t_{ZH}	Figures 9 and 15, C_{RL} = 15pF, S2 closed			20	50	ns
Receiver Disable Time from Low	t_{LZ}	Figures 9 and 15, C_{RL} = 15pF, S1 closed			20	50	ns
Receiver Disable Time from High	t_{HZ}	Figures 9 and 15, C_{RL} = 15pF, S2 closed			20	50	ns
Maximum Data Rate	f_{MAX}			2.5			Mbps
Time to Shutdown	t_{SHDN}	MAX481E (Note 5)		50	200	600	ns

SWITCHING CHARACTERISTICS—MAX481E/MAX485E, MAX490E/MAX491E, MAX1487E (continued)

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

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Driver Enable from Shutdown to Output High (MAX481E)	$t_{ZH(SHDN)}$	Figures 11 and 13, C_L = 100pF, S2 closed		45	100	ns
Driver Enable from Shutdown to Output Low (MAX481E)	$t_{ZL(SHDN)}$	Figures 11 and 13, C_L = 100pF, S1 closed		45	100	ns
Receiver Enable from Shutdown to Output High (MAX481E)	$t_{ZH(SHDN)}$	Figures 9 and 15, C_L = 15pF, S2 closed, A - B = 2V		225	1000	ns
Receiver Enable from Shutdown to Output Low (MAX481E)	$t_{ZL(SHDN)}$	Figures 9 and 15, C_L = 15pF, S1 closed, B - A = 2V		225	1000	ns

SWITCHING CHARACTERISTICS—MAX483E, MAX487E/MAX488E/MAX489E

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

PARAMETER	SYMBOL	CONDITIONS	MIN	TYP	MAX	UNITS
Driver Input to Output	t_{PLH}	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250	800	2000	ns
	t_{PHL}		250	800	2000	
Driver Output Skew to Output	t_{SKEW}	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF		20	800	ns
Driver Rise or Fall Time	t_R, t_F	Figures 10 and 12, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250		2000	ns
Driver Enable to Output High	t_{ZH}	Figures 11 and 13, C_L = 100pF, S2 closed	250		2000	ns
Driver Enable to Output Low	t_{ZL}	Figures 11 and 13, C_L = 100pF, S1 closed	250		2000	ns
Driver Disable Time from Low	t_{LZ}	Figures 11 and 13, C_L = 15pF, S1 closed	300		3000	ns
Driver Disable Time from High	t_{HZ}	Figures 11 and 13, C_L = 15pF, S2 closed	300		3000	ns
Receiver Input to Output	t_{PLH}	Figures 10 and 14, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF	250		2000	ns
	t_{PHL}		250		2000	
$\lvert t_{PLH} - t_{PHL} \rvert$ Differential Receiver Skew	t_{SKD}	Figures 10 and 14, R_{DIFF} = 54Ω, C_{L1} = C_{L2} = 100pF		100		ns
Receiver Enable to Output Low	t_{ZL}	Figures 9 and 15, C_{RL} = 15pF, S1 closed		25	50	ns
Receiver Enable to Output High	t_{ZH}	Figures 9 and 15, C_{RL} = 15pF, S2 closed		25	50	ns
Receiver Disable Time from Low	t_{LZ}	Figures 9 and 15, C_{RL} = 15pF, S1 closed		25	50	ns
Receiver Disable Time from High	t_{HZ}	Figures 9 and 15, C_{RL} = 15pF, S2 closed		25	50	ns
Maximum Data Rate	f_{MAX}	t_{PLH}, t_{PHL} < 50% of data period	250			kbps
Time to Shutdown	t_{SHDN}	MAX483E/MAX487E (Note 5)	50	200	600	ns
Driver Enable from Shutdown to Output High	$t_{ZH(SHDN)}$	MAX483E/MAX487E, Figures 11 and 13, C_L = 100pF, S2 closed			2000	ns
Driver Enable from Shutdown to Output Low	$t_{ZL(SHDN)}$	MAX483E/MAX487E, Figures 11 and 13, C_L = 100pF, S1 closed			2000	ns
Receiver Enable from Shutdown to Output High	$t_{ZH(SHDN)}$	MAX483E/MAX487E, Figures 9 and 15, C_L = 15pF, S2 closed			2500	ns
Receiver Enable from Shutdown to Output Low	$t_{ZL(SHDN)}$	MAX483E/MAX487E, Figures 9 and 15, C_L = 15pF, S1 closed			2500	ns

NOTES FOR ELECTRICAL/SWITCHING CHARACTERISTICS

Note 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.

Note 2: All typical specifications are given for V_{CC} = 5V and T_A = +25°C.

Note 3: Supply current specification is valid for loaded transmitters when DE = 0V.

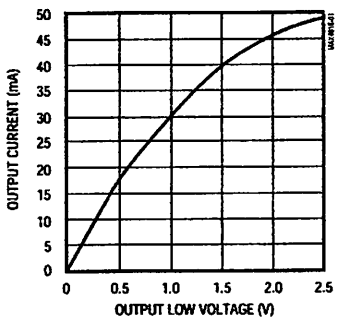
Note 4: Applies to peak current. See *Typical Operating Characteristics*.

Note 5: The MAX481E/MAX483E/MAX487E 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.

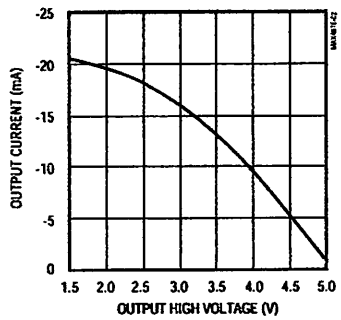
Typical Operating Characteristics

(V_{CC} = 5V, T_A = +25°C, unless otherwise noted.)

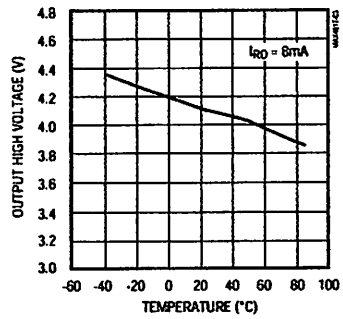
OUTPUT CURRENT vs. RECEIVER OUTPUT LOW VOLTAGE



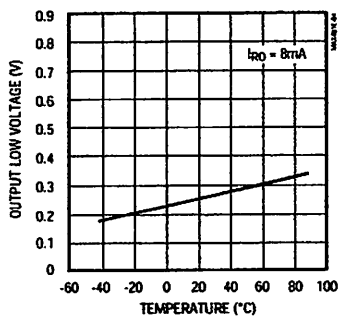
OUTPUT CURRENT vs. RECEIVER OUTPUT HIGH VOLTAGE



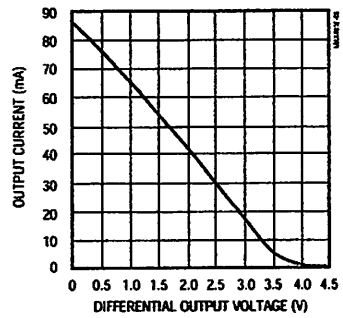
RECEIVER OUTPUT HIGH VOLTAGE vs. TEMPERATURE



RECEIVER OUTPUT LOW VOLTAGE vs. TEMPERATURE



DRIVER OUTPUT CURRENT vs. DIFFERENTIAL OUTPUT VOLTAGE

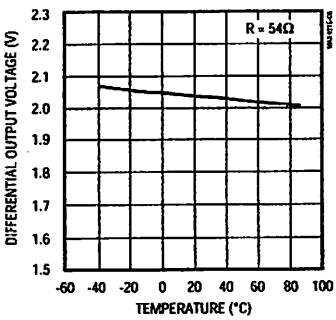


±15kV ESD-Protected, Slew-Rate-Limited, Low-Power, RS-485/RS-422 Transceivers

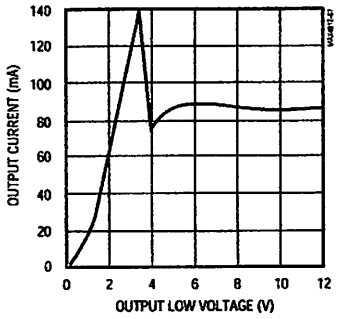
Typical Operating Characteristics (continued)

(V_{CC} = 5V, T_A = +25°C, unless otherwise noted.)

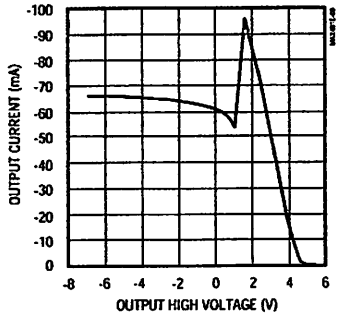
DRIVER DIFFERENTIAL OUTPUT VOLTAGE vs. TEMPERATURE



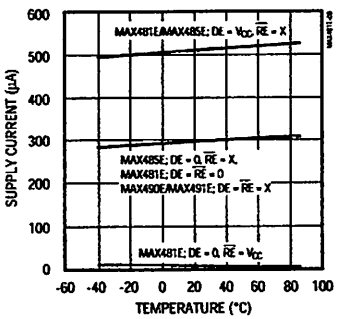
OUTPUT CURRENT vs. DRIVER OUTPUT LOW VOLTAGE



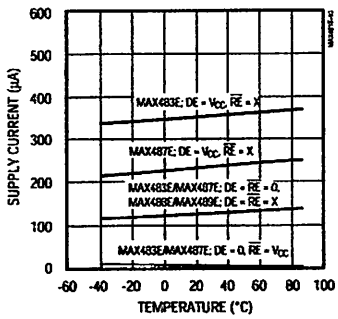
OUTPUT CURRENT vs. DRIVER OUTPUT HIGH VOLTAGE



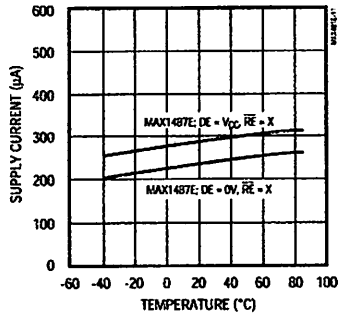
MAX481E/MAX485E/MAX490E/MAX491E SUPPLY CURRENT vs. TEMPERATURE



MAX483E/MAX487E–MAX489E SUPPLY CURRENT vs. TEMPERATURE



MAX1487E SUPPLY CURRENT vs. TEMPERATURE



Pin Description

PIN			NAME	FUNCTION
MAX481E/MAX483E MAX485E/MAX487E MAX1487E	MAX488E MAX490E	MAX489E MAX491E		
1	2	2	RO	Receiver Output: If A > B by 200mV, RO will be high; If A < B by 200mV, RO will be low.
2	—	3	$\overline{RE}$	Receiver Output Enable. RO is enabled when $\overline{RE}$ is low; RO is high impedance when $\overline{RE}$ is high.
3	—	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	3	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	4	6, 7	GND	Ground
—	5	9	Y	Noninverting Driver Output
—	6	10	Z	Inverting Driver Output
6	—	—	A	Noninverting Receiver Input and Noninverting Driver Output
—	8	12	A	Noninverting Receiver Input
7	—	—	B	Inverting Receiver Input and Inverting Driver Output
—	7	11	B	Inverting Receiver Input
8	1	14	V_{CC}	Positive Supply: $4.75V \le V_{CC} \le 5.25V$
—	—	1, 8, 13	N.C.	No Connect—not internally connected

±15kV ESD-Protected, Slew-Rate-Limited, Low-Power, RS-485/RS-422 Transceivers

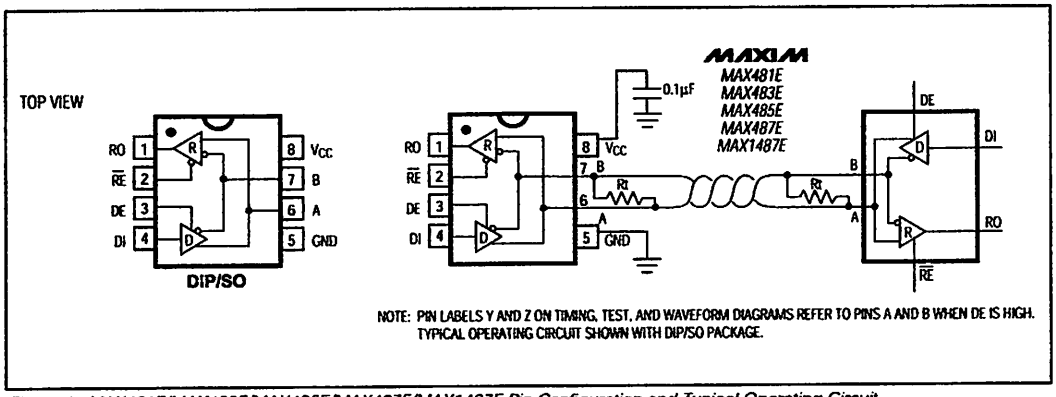


Figure 1. MAX481E/MAX483E/MAX485E/MAX487E/MAX1487E Pin Configuration and Typical Operating Circuit

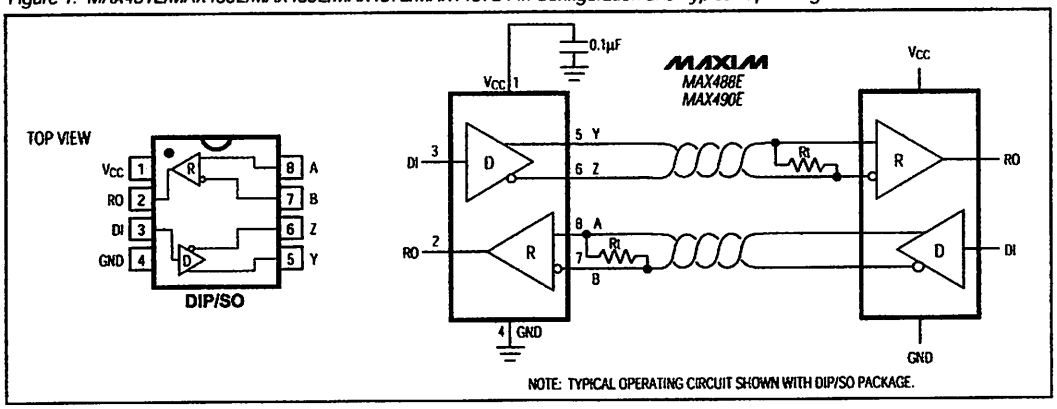


Figure 2. MAX488E/MAX490E Pin Configuration and Typical Operating Circuit

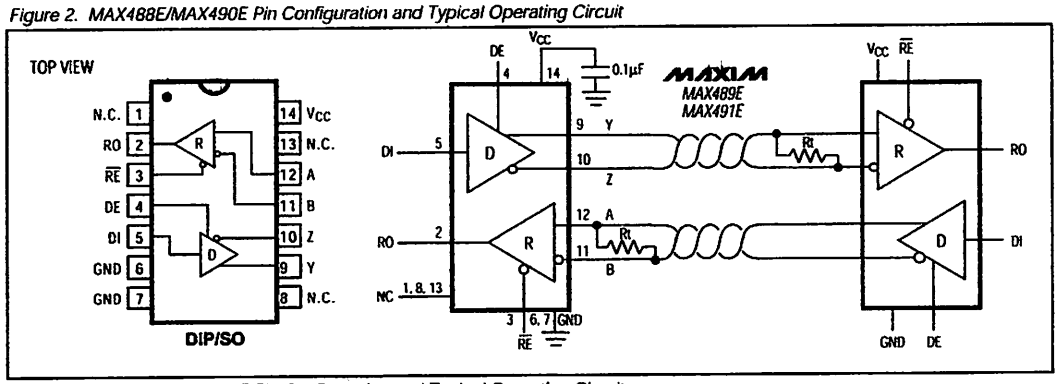


Figure 3. MAX489E/MAX491E Pin Configuration and Typical Operating Circuit

Function Tables (MAX481E/MAX483E/MAX485E/MAX487E/MAX1487E)

Table 1. Transmitting

INPUTS			OUTPUTS	
$\overline{RE}$	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*

X = Don't care
High-Z = High impedance
* Shutdown mode for MAX481E/MAX483E/MAX487E

Table 2. Receiving

INPUTS			OUTPUT
$\overline{RE}$	DE	A-B	RO
0	0	≥ +0.2V	1
0	0	≤ -0.2V	0
0	0	Inputs open	1
1	0	X	High-Z*

X = Don't care
High-Z = High impedance
* Shutdown mode for MAX481E/MAX483E/MAX487E

Applications Information

The MAX481E/MAX483E/MAX485E/MAX487E–MAX491E and MAX1487E are low-power transceivers for RS-485 and RS-422 communications. These "E" versions of the MAX481, MAX483, MAX485, MAX487–MAX491, and MAX1487 provide extra protection against ESD. The rugged MAX481E, MAX483E, MAX485E, MAX497E–MAX491E, and MAX1487E are intended for harsh environments where high-speed communication is important. These devices eliminate the need for transient suppressor diodes and the associated high capacitance loading. The standard (non-"E") MAX481, MAX483, MAX485, MAX487–MAX491, and MAX1487 are recommended for applications where cost is critical.

The MAX481E, MAX485E, MAX490E, MAX491E, and MAX1487E can transmit and receive at data rates up to 2.5Mbps, while the MAX483E, MAX487E, MAX488E, and MAX489E are specified for data rates up to 250kbps. The MAX488E–MAX491E are full-duplex transceivers, while the MAX481E, MAX483E, MAX487E, and MAX1487E are half-duplex. In addition, driver-enable (DE) and receiver-enable (RE) pins are included on the MAX481E, MAX483E, MAX485E, MAX487E, MAX489E, MAX491E, and MAX1487E. When disabled, the driver and receiver outputs are high impedance.

±15kV ESD Protection

As with all Maxim devices, ESD-protection structures are incorporated on all pins to protect against electrostatic discharges encountered during handling and assembly. The driver outputs and receiver inputs have extra protection against static electricity. Maxim's engineers developed state-of-the-art structures to protect these pins against ESD of ±15kV without damage. The ESD structures withstand high ESD in all states: normal operation, shutdown, and powered down. After an ESD event, Maxim's MAX481E, MAX483E, MAX485E, MAX487E–MAX491E, and MAX1487E keep working without latchup.

ESD protection can be tested in various ways; the transmitter outputs and receiver inputs of this product family are characterized for protection to ±15kV using the Human Body Model.

Other ESD test methodologies include IEC10004-2 contact discharge and IEC1000-4-2 air-gap discharge (formerly IEC801-2).

ESD Test Conditions

ESD performance depends on a variety of conditions. Contact Maxim for a reliability report that documents test set-up, test methodology, and test results.

Human Body Model

Figure 4 shows the Human Body Model, and Figure 5 shows the current waveform it generates when discharged into a low impedance. This model consists of a 100pF capacitor charged to the ESD voltage of interest, which is then discharged into the test device through a 1.5kΩ resistor.

IEC1000-4-2

The IEC1000-4-2 standard covers ESD testing and performance of finished equipment; it does not specifically refer to integrated circuits (Figure 6).

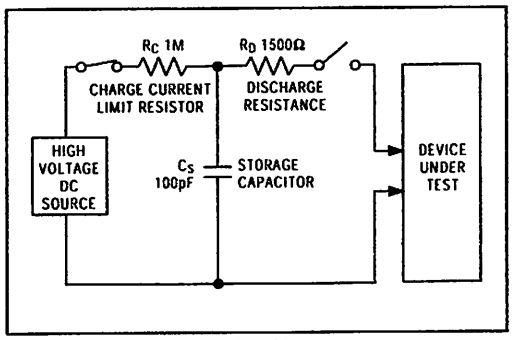


Figure 4. Human Body ESD Test Model

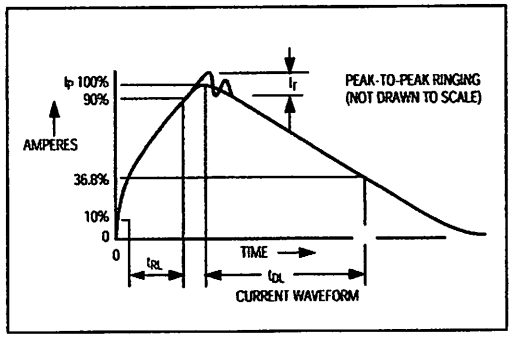


Figure 5. Human Body Model Current Waveform

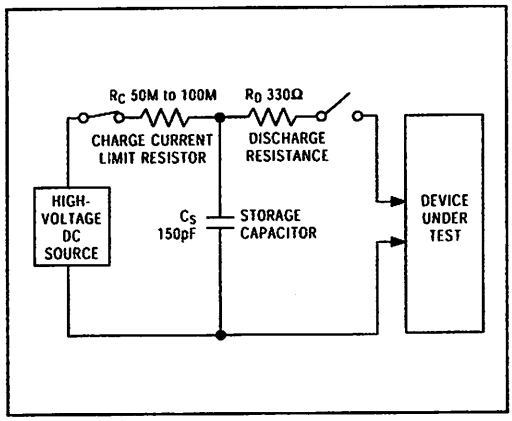


Figure 6. IEC1000-4-2 ESD Test Model

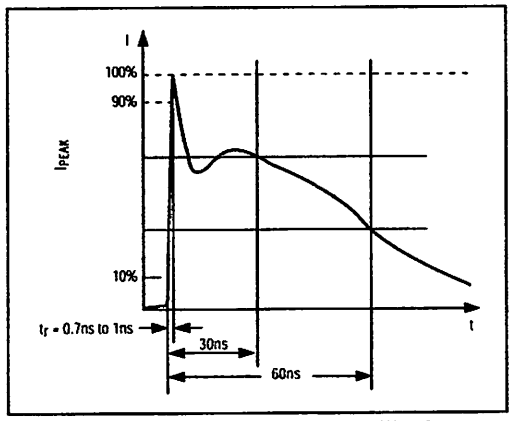


Figure 7. IEC1000-4-2 ESD Generator Current Waveform

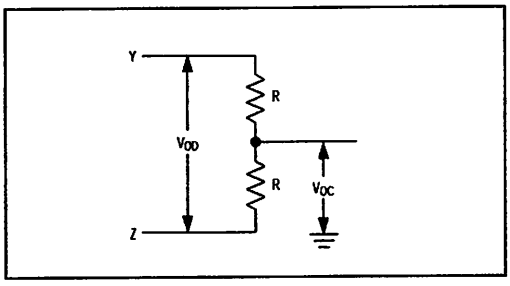


Figure 8. Driver DC Test Load

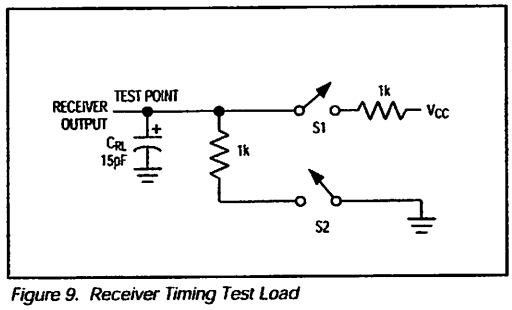


Figure 9. Receiver Timing Test Load

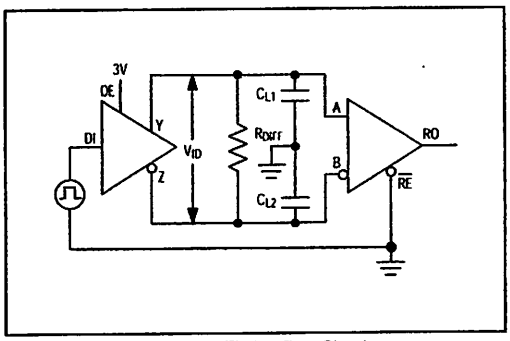


Figure 10. Driver/Receiver Timing Test Circuit

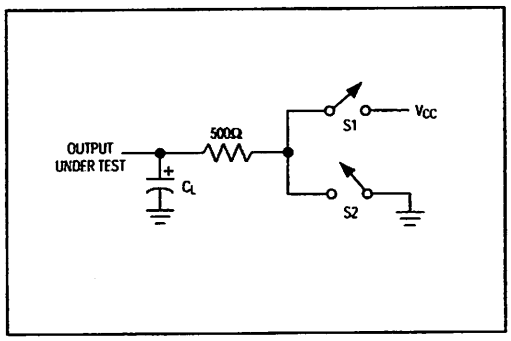


Figure 11. Driver Timing Test Load

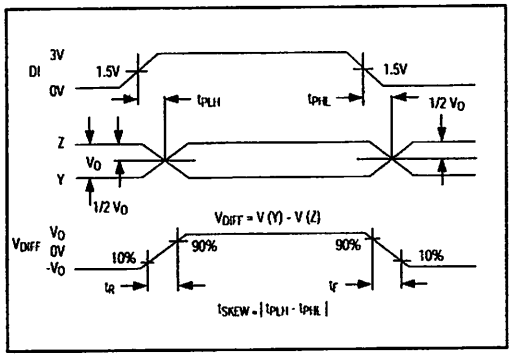


Figure 12. Driver Propagation Delays

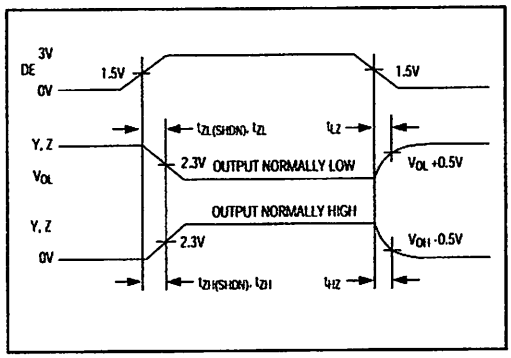


Figure 13. Driver Enable and Disable Times (except MAX488E and MAX490E)

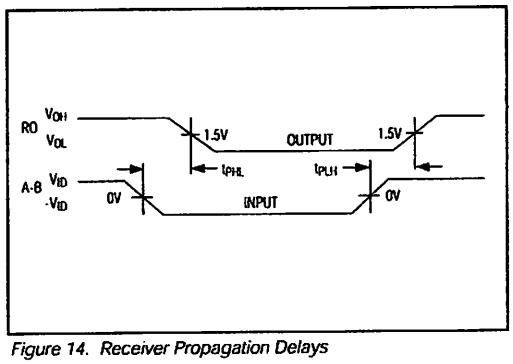


Figure 14. Receiver Propagation Delays

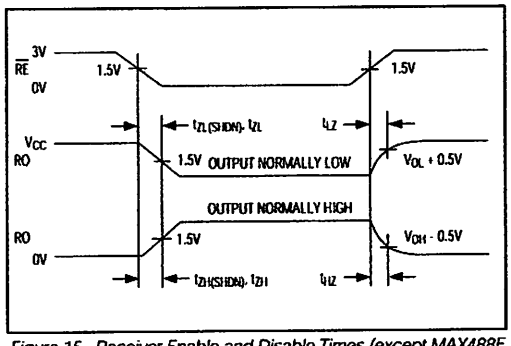


Figure 15. Receiver Enable and Disable Times (except MAX488E and MAX490E)

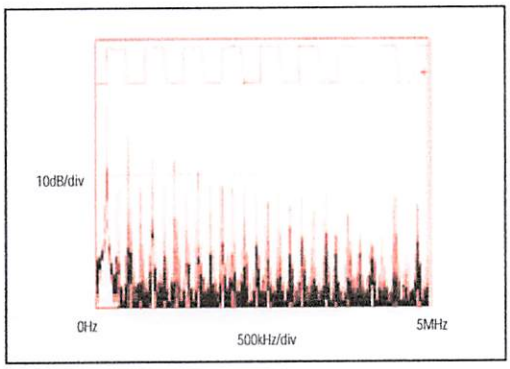


Figure 16. Driver Output Waveform and FFT Plot of MAX485E/MAX490E/MAX491E/MAX1487E Transmitting a 150kHz Signal

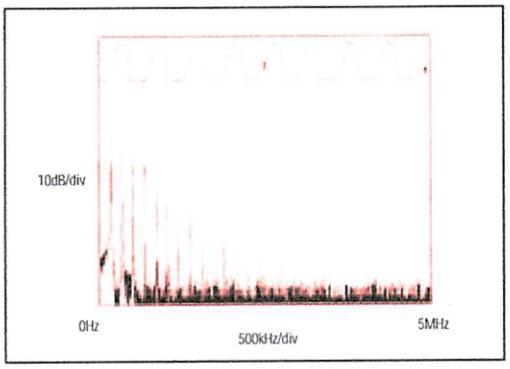


Figure 17. Driver Output Waveform and FFT Plot of MAX483E/MAX487E–MAX489E Transmitting a 150kHz Signal

The major difference between tests done using the Human Body Model and IEC1000-4-2 is higher peak current in IEC1000-4-2, because series resistance is lower in the IEC1000-4-2 model. Hence, the ESD withstand voltage measured to IEC1000-4-2 is generally lower than that measured using the Human Body Model. Figure 7 shows the current waveform for the 8kV IEC1000-4-2 ESD contact-discharge test.

The air-gap test involves approaching the device with a charged probe. The contact-discharge method connects the probe to the device before the probe is energized.

Machine Model

The Machine Model for ESD tests all pins using a 200pF storage capacitor and zero discharge resistance. Its objective is to emulate the stress caused by contact that occurs with handling and assembly during manufacturing. Of course, all pins require this protection during manufacturing—not just inputs and outputs. Therefore, after PC board assembly, the Machine Model is less relevant to I/O ports.

MAX487E/MAX1487E: 128 Transceivers on the Bus

The 48kΩ, 1/4-unit-load receiver input impedance of the MAX487E and MAX1487E 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 MAX487E/MAX1487E and other RS-485 transceivers with a total of 32 unit loads or less can be put on the bus. The MAX481E, MAX483E, MAX485E, and MAX488E–MAX491E have standard 12kΩ receiver input impedance.

MAX483E/MAX487E/MAX488E/MAX489E: Reduced EMI and Reflections

The MAX483E and MAX487E–MAX489E are slew-rate limited, minimizing EMI and reducing reflections caused by improperly terminated cables. Figure 16 shows the driver output waveform and its Fourier analysis of a 150kHz signal transmitted by a MAX481E, MAX485E, MAX490E, MAX491E, or MAX1487E. High-frequency harmonics with large amplitudes are evident. Figure 17 shows the same information displayed for a MAX483E, MAX487E, MAX488E, or MAX489E transmitting under the same conditions. Figure 17's high-frequency harmonics have much lower amplitudes, and the potential for EMI is significantly reduced.

Low-Power Shutdown Mode (MAX481E/MAX483E/MAX487E)

A low-power shutdown mode is initiated by bringing both $\overline{RE}$ 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 0.5µA 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 is 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 MAX481E, MAX483E, and MAX487E, the t_{ZH} and t_{ZL} enable times assume the part was not in the low-power shutdown state (the MAX485E, MAX488E–MAX491E, and MAX1487E can not be shut down). The $t_{ZH(SHDN)}$ and $t_{ZL(SHDN)}$ enable times assume the parts were shut down (see *Electrical Characteristics*).

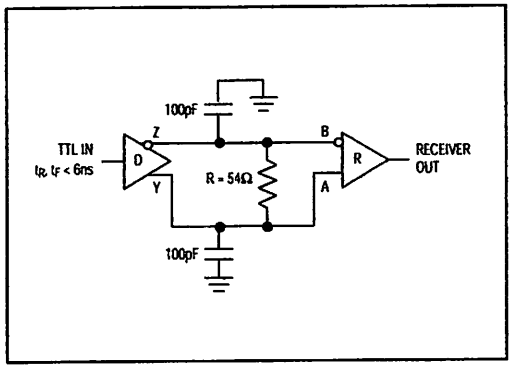


Figure 18. Receiver Propagation Delay Test Circuit

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 $\overline{RE}$, DE inputs equal a logical 0,1 or 1,1 or 0, 0.)

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 19–22 using Figure 18's test circuit.

The difference in receiver delay times, t_{PLH} - t_{PHL}, is typically under 13ns for the MAX481E, MAX485E, MAX490E, MAX491E, and MAX1487E, and is typically less than 100ns for the MAX483E and MAX487E–MAX489E.

The driver skew times are typically 5ns (10ns max) for the MAX481E, MAX485E, MAX490E, MAX491E, and MAX1487E, and are typically 100ns (800ns max) for the MAX483E and MAX487E–MAX489E.

Typical Applications

The MAX481E, MAX483E, MAX485E, MAX487E–MAX491E, and MAX1487E transceivers are designed for bidirectional data communications on multipoint bus transmission lines. Figures 25 and 26 show typical network application circuits. These parts can also be used as line repeaters, with cable lengths longer than 4000 feet.

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 MAX483E and MAX487E–MAX489E are more tolerant of imperfect termination. Bypass the V_{CC} pin with 0.1µF.

Isolated RS-485

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

Line Length vs. Data Rate

The RS-485/RS-422 standard covers line lengths up to 4000 feet. Figures 23 and 24 show the system differential voltage for the parts driving 4000 feet of 26AWG twisted-pair wire at 110kHz into 100Ω loads.

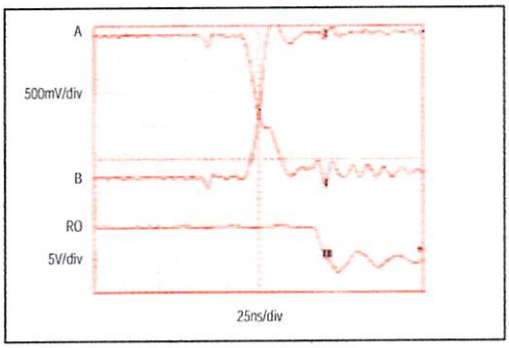


Figure 19. MAX481E/MAX485E/MAX490E/MAX1487E Receiver t_{PHL}

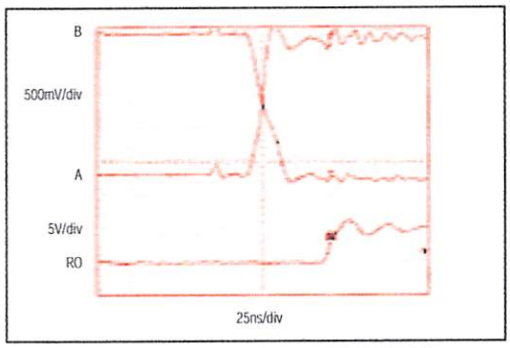


Figure 20. MAX481E/MAX485E/MAX490E/MAX491E/MAX1487E Receiver t_{PLH}

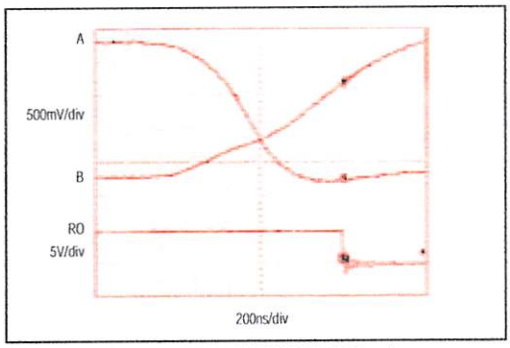


Figure 21. MAX483E/MAX487E–MAX489E Receiver t_{PHL}

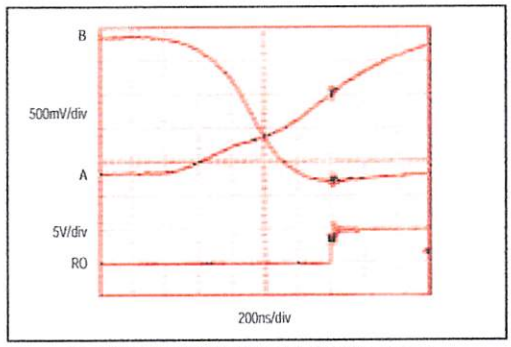


Figure 22. MAX483E/MAX487E–MAX489E Receiver t_{PLH}

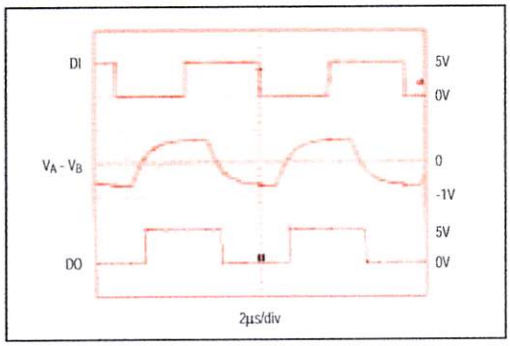


Figure 23. MAX481E/MAX485E/MAX490E/MAX491E/MAX1487E System Differential Voltage at 110kHz Driving 4000ft of Cable

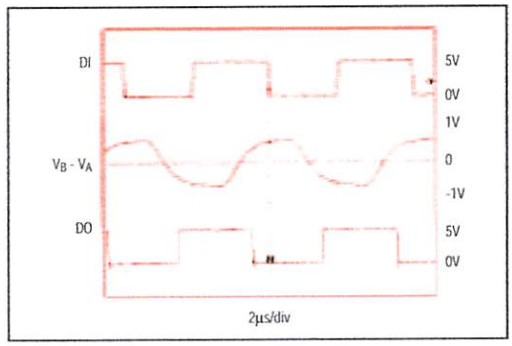


Figure 24. MAX483E/MAX1487E–MAX489E System Differential Voltage at 110kHz Driving 4000ft of Cable

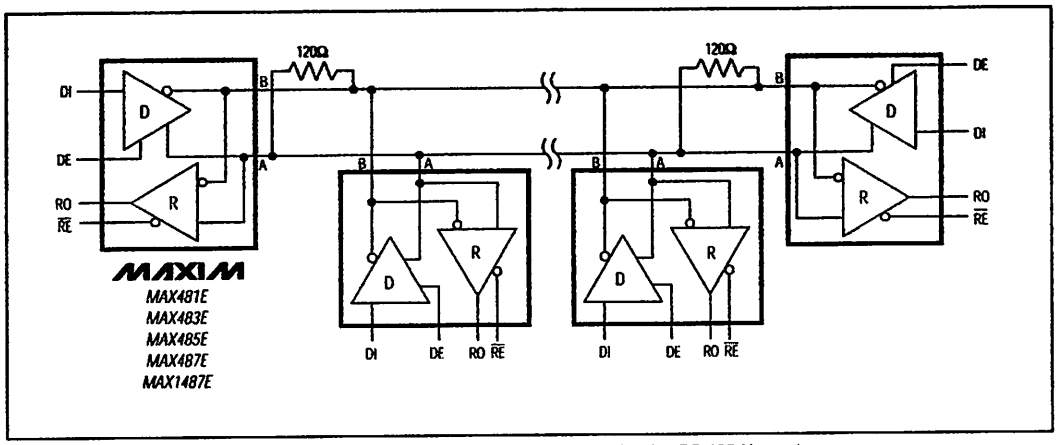


Figure 25. MAX481E/MAX483E/MAX485E/MAX487E/MAX1487E Typical Half-Duplex RS-485 Network

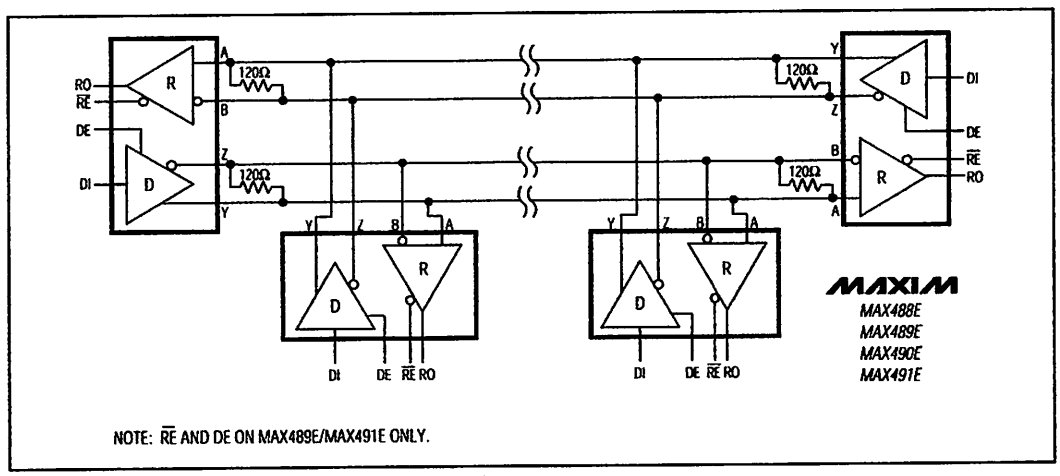


Figure 26. MAX488E–MAX491E Full-Duplex RS-485 Network

±15kV ESD-Protected, Slew-Rate-Limited, Low-Power, RS-485/RS-422 Transceivers

Ordering Information (continued)

PART	TEMP. RANGE	PIN-PACKAGE
MAX483ECPA	0°C to +70°C	8 Plastic DIP
MAX483ECSA	0°C to +70°C	8 SO
MAX483EEPA	-40°C to +85°C	8 Plastic DIP
MAX483EESA	-40°C to +85°C	8 SO
MAX485ECPA	0°C to +70°C	8 Plastic DIP
MAX485ECSA	0°C to +70°C	8 SO
MAX485EEPA	-40°C to +85°C	8 Plastic DIP
MAX485EESA	-40°C to +85°C	8 SO
MAX487ECPA	0°C to +70°C	8 Plastic DIP
MAX487ECSA	0°C to +70°C	8 SO
MAX487EEPA	-40°C to +85°C	8 Plastic DIP
MAX487EESA	-40°C to +85°C	8 SO
MAX488ECPA	0°C to +70°C	8 Plastic DIP
MAX488ECSA	0°C to +70°C	8 SO
MAX488EEPA	-40°C to +85°C	8 Plastic DIP
MAX488EESA	-40°C to +85°C	8 SO

PART	TEMP. RANGE	PIN-PACKAGE
MAX489ECPD	0°C to +70°C	14 Plastic DIP
MAX489ECSD	0°C to +70°C	14 SO
MAX489EEPD	-40°C to +85°C	14 Plastic DIP
MAX489EESD	-40°C to +85°C	14 SO
MAX490ECPA	0°C to +70°C	8 Plastic DIP
MAX490ECSA	0°C to +70°C	8 SO
MAX490EEPA	-40°C to +85°C	8 Plastic DIP
MAX490EESA	-40°C to +85°C	8 SO
MAX491ECPD	0°C to +70°C	14 Plastic DIP
MAX491ECSD	0°C to +70°C	14 SO
MAX491EEPD	-40°C to +85°C	14 Plastic DIP
MAX491EESD	-40°C to +85°C	14 SO
MAX1487ECPA	0°C to +70°C	8 Plastic DIP
MAX1487ECSA	0°C to +70°C	8 SO
MAX1487EEPA	-40°C to +85°C	8 Plastic DIP
MAX1487EESA	-40°C to +85°C	8 SO

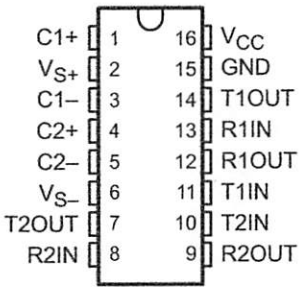
Chip Information

TRANSISTOR COUNT: 295

SLLS047I – FEBRUARY 1989 – REVISED OCTOBER 2002

- Meet or Exceed TIA/EIA-232-F and ITU Recommendation V.28
- Operate With Single 5-V Power Supply
- Operate Up to 120 kbit/s
- Two Drivers and Two Receivers
- ±30-V Input Levels
- Low Supply Current . . . 8 mA Typical
- Designed to be Interchangeable With Maxim MAX232
- ESD Protection Exceeds JESD 22
  – 2000-V Human-Body Model (A114-A)
- Applications
  - TIA/EIA-232-F
  - Battery-Powered Systems
  - Terminals
  - Modems
  - Computers

MAX232 . . . D, DW, N, OR NS PACKAGE
MAX232I . . . D, DW, OR N PACKAGE
(TOP VIEW)



description/ordering information

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

ORDERING INFORMATION

T_A	PACKAGE†		ORDERABLE PART NUMBER	TOP-SIDE MARKING
0°C to 70°C	PDIP (N)	Tube	MAX232N	MAX232N
	SOIC (D)	Tube	MAX232D	MAX232
		Tape and reel	MAX232DR	
	SOIC (DW)	Tube	MAX232DW	MAX232
		Tape and reel	MAX232DWR	
	SOP (NS)	Tape and reel	MAX232NSR	MAX232
–40°C to 85°C	PDIP (N)	Tube	MAX232IN	MAX232IN
	SOIC (D)	Tube	MAX232ID	MAX232I
		Tape and reel	MAX232IDR	
	SOIC (DW)	Tube	MAX232IDW	MAX232I
		Tape and reel	MAX232IDWR	

† Package drawings, standard packing quantities, thermal data, symbolization, and PCB design guidelines are available at www.ti.com/sc/package.

Function Tables

EACH DRIVER

INPUT TIN	OUTPUT TOUT
L	H
H	L

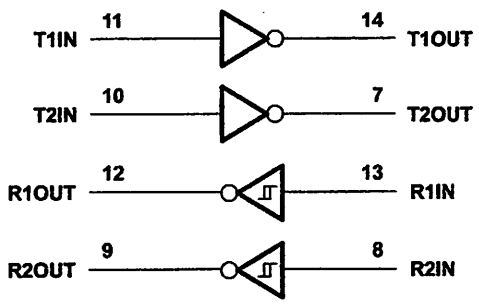
H = high level, L = low level

EACH RECEIVER

INPUT RIN	OUTPUT ROUT
L	H
H	L

H = high level, L = low level

Logic diagram (positive logic)



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

Input supply voltage range, V_{CC} (see Note 1) ..... –0.3 V to 6 V
Positive output supply voltage range, V_{S+} ..... V_{CC} – 0.3 V to 15 V
Negative output supply voltage range, V_{S-} ..... –0.3 V to –15 V
Input voltage range, V_I: Driver ..... –0.3 V to V_{CC} + 0.3 V
Receiver ..... ±30 V
Output voltage range, V_O: T1OUT, T2OUT ..... V_{S-} – 0.3 V to V_{S+} + 0.3 V
R1OUT, R2OUT ..... –0.3 V to V_{CC} + 0.3 V
Short-circuit duration: T1OUT, T2OUT ..... Unlimited
Package thermal impedance, θ_{JA} (see Note 2): D package ..... 73°C/W
DW package ..... 57°C/W
N package ..... 67°C/W
NS package ..... 64°C/W
Lead temperature 1,6 mm (1/16 inch) from case for 10 seconds ..... 260°C
Storage temperature range, T_{stg} ..... –65°C to 150°C

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

NOTE 1: All voltage values are with respect to network ground terminal.
2. The package thermal impedance is calculated in accordance with JESD 51-7.

recommended operating conditions

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

electrical characteristics over recommended ranges of supply voltage and operating free-air temperature (unless otherwise noted) (see Note 3 and Figure 4)

PARAMETER		TEST CONDITIONS	MIN	TYP‡	MAX	UNIT
I_{CC}	Supply current	V_{CC} = 5.5 V, All outputs open, T_A = 25°C		8	10	mA

‡ All typical values are at V_{CC} = 5 V and T_A = 25°C.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

DRIVER SECTION

Electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	T1OUT, T2OUT	R_L = 3 kΩ to GND		5	7		V
V_{OL}	Low-level output voltage‡	T1OUT, T2OUT	R_L = 3 kΩ to GND			–7	–5	V
r_o	Output resistance	T1OUT, T2OUT	V_{S+} = V_{S-} = 0,	V_O = ±2 V	300			Ω
I_{OS}§	Short-circuit output current	T1OUT, T2OUT	V_{CC} = 5.5 V,	V_O = 0		±10		mA
I_{IS}	Short-circuit input current	T1IN, T2IN	V_I = 0				200	µA

All typical values are at V_{CC} = 5 V, T_A = 25°C.
The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
Not more than one output should be shorted at a time.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

Switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3)

PARAMETER		TEST CONDITIONS	MIN	TYP	MAX	UNIT
SR	Driver slew rate	R_L = 3 kΩ to 7 kΩ, See Figure 2			30	V/µs
SR(t)	Driver transition region slew rate	See Figure 3		3		V/µs
	Data rate	One TOUT switching		120		kbit/s

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

RECEIVER SECTION

Electrical characteristics over recommended ranges of supply voltage and operating free-air temperature range (see Note 3)

PARAMETER			TEST CONDITIONS		MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage	R1OUT, R2OUT	I_{OH} = –1 mA		3.5			V
V_{OL}	Low-level output voltage‡	R1OUT, R2OUT	I_{OL} = 3.2 mA				0.4	V
V_{IT+}	Receiver positive-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V,	T_A = 25°C		1.7	2.4	V
V_{IT-}	Receiver negative-going input threshold voltage	R1IN, R2IN	V_{CC} = 5 V,	T_A = 25°C	0.8	1.2		V
V_{hys}	Input hysteresis voltage	R1IN, R2IN	V_{CC} = 5 V		0.2	0.5	1	V
r_i	Receiver input resistance	R1IN, R2IN	V_{CC} = 5,	T_A = 25°C	3	5	7	kΩ

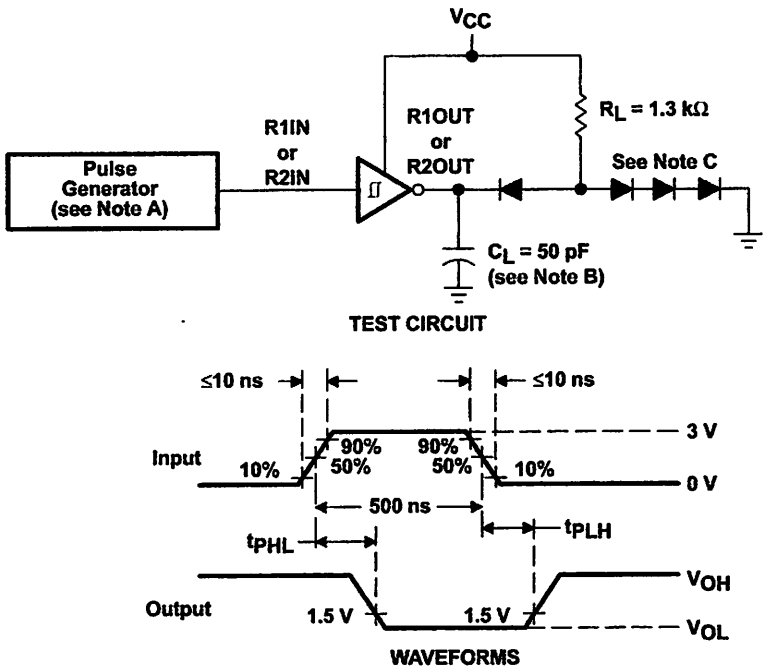
All typical values are at V_{CC} = 5 V, T_A = 25°C.
The algebraic convention, in which the least positive (most negative) value is designated minimum, is used in this data sheet for logic voltage levels only.
NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

Switching characteristics, V_{CC} = 5 V, T_A = 25°C (see Note 3 and Figure 1)

PARAMETER		TYP	UNIT
$t_{PLH(R)}$	Receiver propagation delay time, low- to high-level output	500	ns
$t_{PHL(R)}$	Receiver propagation delay time, high- to low-level output	500	ns

NOTE 3: Test conditions are C1–C4 = 1 µF at V_{CC} = 5 V ± 0.5 V.

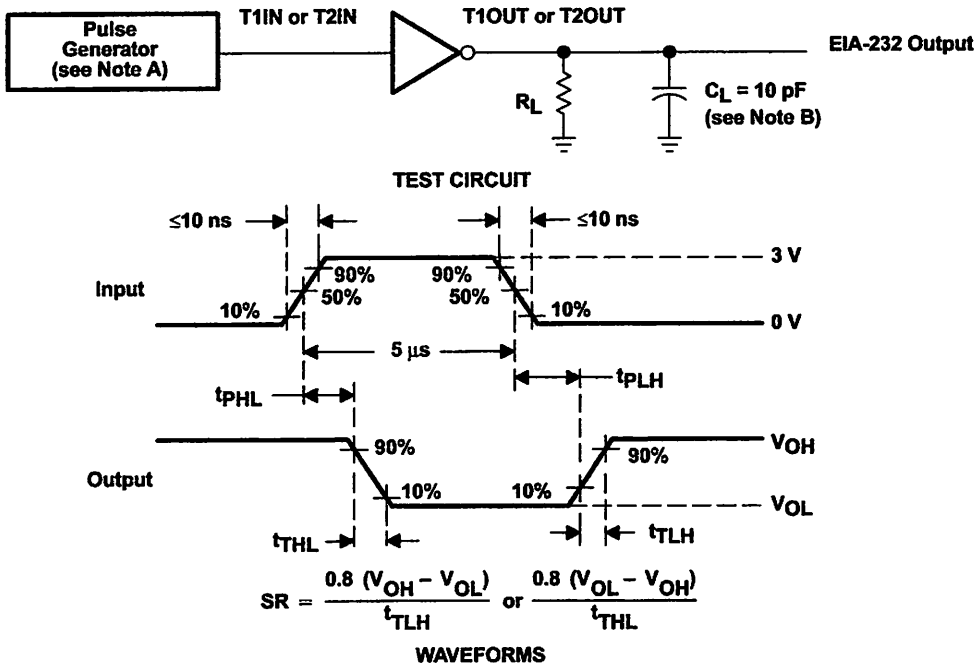
PARAMETER MEASUREMENT INFORMATION



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

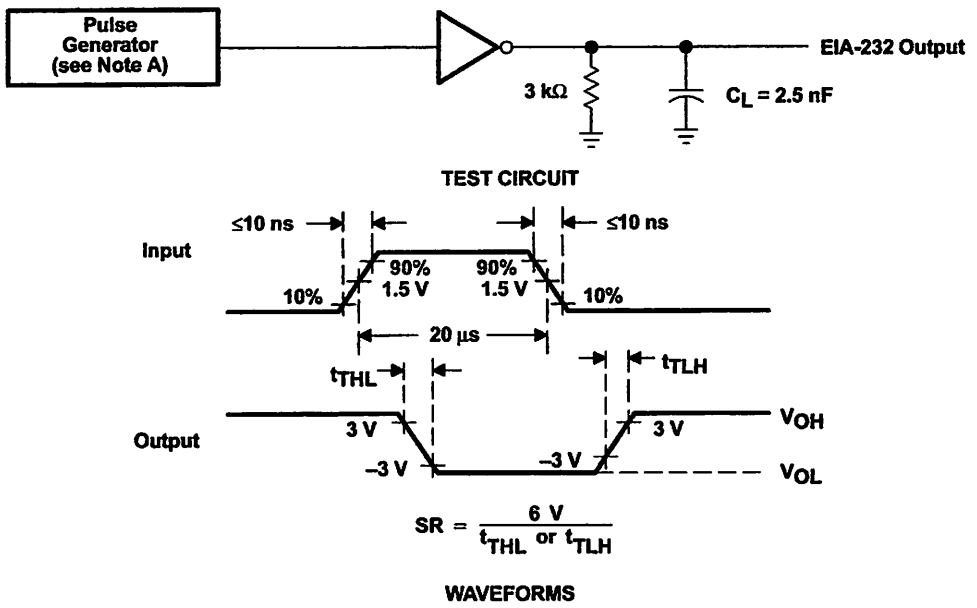
Figure 1. Receiver Test Circuit and Waveforms for t_{PHL} and t_{PLH} Measurements

PARAMETER MEASUREMENT INFORMATION



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

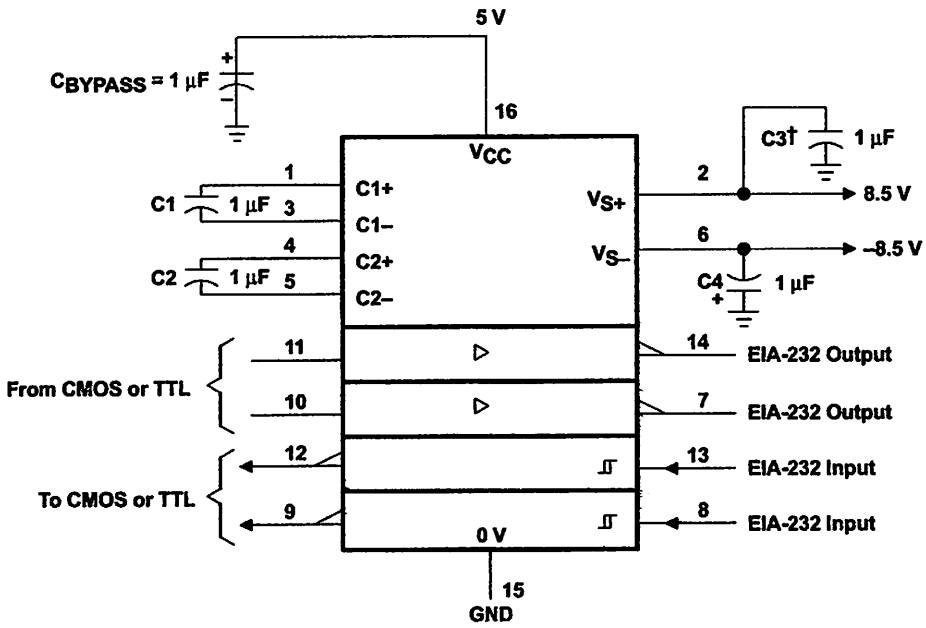
Figure 2. Driver Test Circuit and Waveforms for t_{PHL} and t_{PLH} Measurements (5-µs Input)



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

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

APPLICATION INFORMATION



† C3 can be connected to V_{CC} or GND.

Figure 4. Typical Operating Circuit

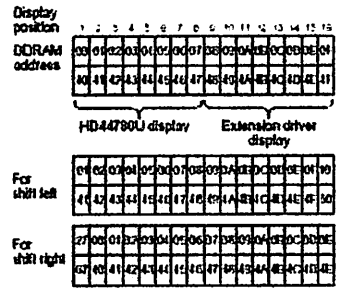
M1632 MODULE LCD 16 X 2 BARIS (M1632)

Deskripsi:

M1632 adalah merupakan modul LCD dengan tampilan 16 x 2 baris dengan konsumsi daya y rendah. Modul ini dilengkapi dengan mikrokontroler yang didisain khusus untuk mengendalikan L(Mikrokontroler HD44780 buatan Hitachi yang berfungsi sebagai pengendali LCD ini mempunyai CGR((Character Generator Read Only Memory), CGRAM (Character Generator Random Access Memory) DDRAM (Display Data Random Access Memory).

DDRAM

DDRAM adalah merupakan memori tempat karakter yang ditampilkan berada. Contoh, un karakter 'A' atau 41H yang ditulis pada alamat 00, maka karakter tersebut akan tampil pada baris pert dan kolom pertama dari LCD. Apabila karakter tersebut ditulis di alamat 40, maka karakter tersebut a tampil pada baris kedua kolom pertama dari LCD.



Gambar 1
DDRAM M1632 (diambil dari data sheet HD44780)

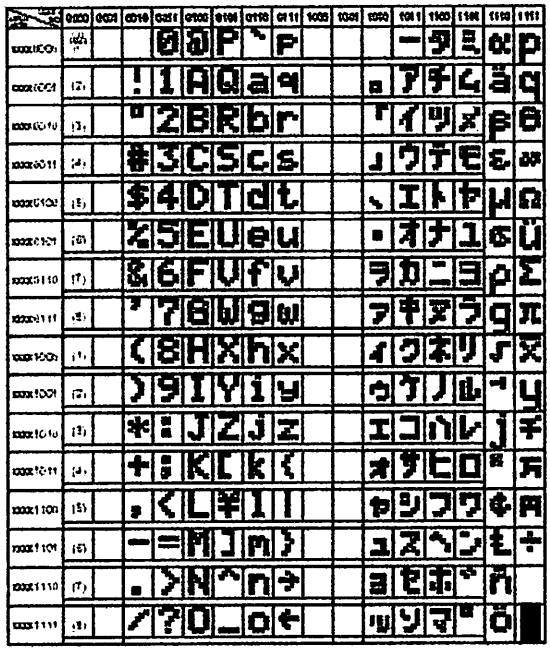
CGRAM

CGRAM adalah merupakan memori untuk menggambarkan **pola sebuah karakter** di m bentuk dari karakter dapat diubah-ubah sesuai keinginan. Namun memori ini akan hilang saat power sup tidak aktif, sehingga pola karakter akan hilang.

CGROM

CGROM adalah merupakan memori untuk menggambarkan **pola sebuah karakter** di mana p tersebut sudah ditentukan secara permanen dari HD44780 sehingga pengguna tidak dapat mengubah l Namun karena ROM bersifat permanen, maka pola karakter tersebut tidak akan hilang walaupun po supply tidak aktif

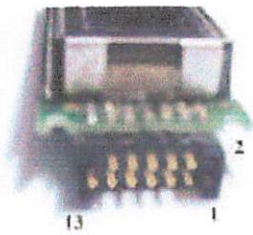
Pada gambar 2, tampak terlihat pola-pola karakter yang tersimpan dalam lokasi-lokasi tert dalam CGROM. Pada saat HD44780 akan menampilkan data 41H yang tersimpan pada DDRAM, m HD44780 akan mengambil data di alamat 41H (0100 0001) yang ada pada CGROM yaitu pola karakter A



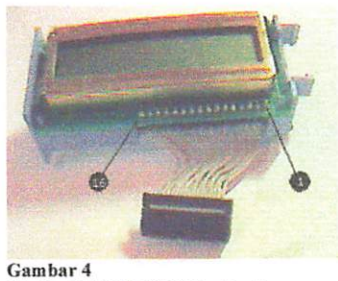
Gambar 2
Hubungan antara CGROM dan DDRAM (diambil dari data sheet HD44780)

Pin Out

No	Nama Pin	Deskripsi
1	VCC	+5V
2	GND	0V
3	VEE	Tegangan Kontras LCD
4	RS	Register Select, 0 = Register Perintah, 1 = Register I
5	R/W	1 = Read, 0 = Write
6	E	Enable Clock LCD, logika 1 setiap kali pengiriman a pembacaan data
7	D0	Data Bus 0
8	D1	Data Bus 1
9	D2	Data Bus 2
10	D3	Data Bus 3
11	D4	Data Bus 4
12	D5	Data Bus 5
13	D6	Data Bus 6
14	D7	Data Bus 7
15	Anoda (Kabel coklat untuk LCD Hitachi)	Tegangan positif backlight
16	Katoda (Kabel merah untuk LCD Hitachi)	Tegangan negatif backlight



Gambar 3
Pin Out M1632 LCD Hitachi



Gambar 4
Pin Out LCD M1632 Standard

Register

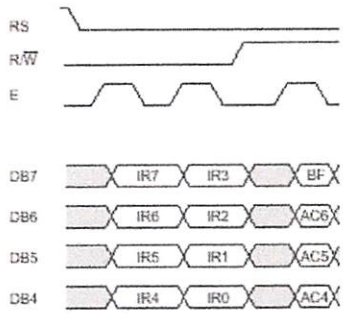
HD44780, mempunyai dua buah Register yang aksesnya diatur dengan menggunakan kaki Pada saat RS berlogika 0, maka register yang diakses adalah Register Perintah dan pada saat RS berlog 1, maka register yang diakses adalah Register Data

Register Perintah

Register ini adalah register di mana perintah-perintah dari mikrokontroler ke HD44780 pada proses penulisan data atau tempat status dari HD44780 dapat dibaca pada saat pembacaan data.

Penulisan Data ke Register Perintah

Penulisan data ke Register Perintah dilakukan dengan tujuan mengatur tampilan LCD, inisiali dan mengatur Address Counter maupun Address Data. Gambar 5 menunjukkan proses penulisan data register perintah dengan menggunakan mode 4 bit interface. Kondisi RS berlogika 0 menunjukkan ak data ke Register Perintah. RW berlogika 0 yang menunjukkan proses penulisan data akan dilakukan. Nib tinggi (bit 7 sampai bit 4) terlebih dahulu dikirimkan dengan diawali pulsa logika 1 pada E Clo Kemudian Nibble rendah (bit 3 sampai bit 0) dikirimkan dengan diawali pulsa logika 1 pada E Clock l Untuk mode 8 bit interface, proses penulisan dapat langsung dilakukan secara 8 bit (bit 7 ... bit 0) diawali sebuah pulsa logika 1 pada E Clock.



Gambar 5
Timing diagram Penulisan Data ke Register Perintah Mode 4 bit Interface

Tabel 1
Perintah-perintah M1632

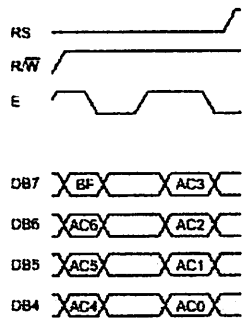
Perintah	D7	D6	D5	D4	D3	D2	D1	D0	Deskripsi
Hapus Display	0	0	0	0	0	0	0	1	Hapus Display dan DDRAM
Posisi Awal	0	0	0	0	0	0	1	X	Set Alamat DDRAM di 0
Set Mode	0	0	0	0	0	1	I/D	S	Atur arah pergeseran cursor dan displ

Display On/OFF	0	0	0	0	1	D	C	B	Atur display (D) On/OFF, cursor ON/OFF, Blinking (B)
Geser Cursor/Display	0	0	0	1	S/C	R/L	X	X	Geser Cursor atau display tanpa men alamat DDRAM
Set Fungsi	0	0	1	DL	N	F	X	X	Atur panjang data, jumlah baris : tampil, dan font karakter
Set Alamat CGRAM	0	1	ACG	ACG	ACG	ACG	ACG	ACG	Data dapat dibaca atau ditulis set alamat diatur
Set Alamat DDRAM	1	ADD	ADD	ADD	ADD	ADD	ADD	ADD	Data dapat dibaca atau ditulis set alamat diatur

X = diabaikan
I/D 1=Increment, 0=Decrement
S 0=Display tidak geser
S/C 1=Display Shift, 0=Geser Cursor
R/L 1=Geser Kiri, 0=Geser Kanan
DL 1=8 bit, 0=4bit
N 1=2 baris, 0=1 baris
F 1=5x10, 0=5x8
D 0=Display OFF, 1=Display ON
C 0=Cursor OFF, 1=Cursor ON
B 0=Blinking OFF, 1=Blinking ON

Pembacaan Data dari Register Perintah

Proses pembacaan data pada register perintah biasa digunakan untuk melihat status busy dari L atau membaca Address Counter. RS diatur pada logika 0 untuk akses ke Register Perintah, R/W diatur p logika 1 yang menunjukkan proses pembacaan data. 4 bit nibble tinggi dibaca dengan diawali pulsa log 1 pada E Clock dan kemudian 4 bit nibble rendah dibaca dengan diawali pulsa logika 1 pada E Clo Untuk Mode 8 bit interface, pembacaan 8 bit (nibble tinggi dan rendah) dilakukan sekaligus dengan diav sebuah pulsa logika 1 pada E Clock.



Gambar 6
Timing Diagram Pembacaan Register Perintah Mode 4 bit Interface

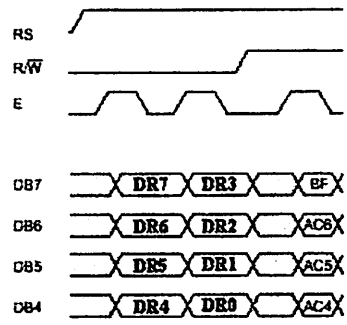
Register Data

Register ini adalah register di mana mikrokontroler dapat menuliskan atau membaca data ke a dari DDRAM. Penulisan data pada register ini akan menempatkan data tersebut ke DDRAM sesuai den alamat yang telah diatur sebelumnya

Penulisan Data ke Register Data

Penulisan data pada Register Data dilakukan untuk mengirimkan data yang akan ditampilkan pa LCD. Proses diawali dengan adanya logika 1 pada RS yang menunjukkan akses ke Register Data, kondisi R/W diatur pada logika 0 yang menunjukkan proses penulisan data. Data 4 bit nibble tinggi (bit 7 hingga

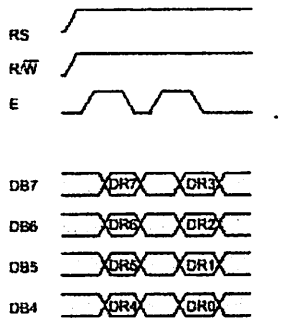
bit 4) dikirim dengan diawali pulsa logika 1 pada sinyal E Clock dan kemudian diikuti 4 bit nibble rendah (bit 3 hingga bit 0) yang juga diawali pulsa logika 1 pada sinyal E Clock.



Gambar 7
Timing Diagram Penulisan Data ke Register Data Mode 4 bit Interface

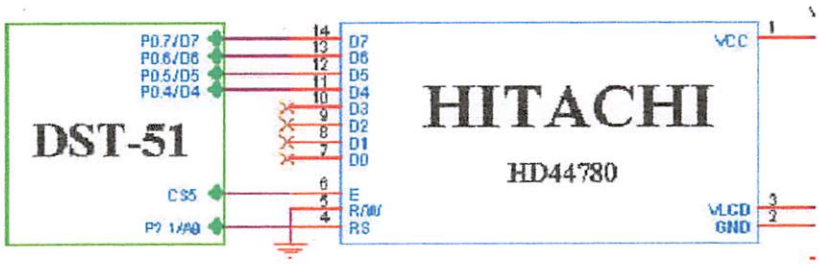
Pembacaan Data dari Register Data

Pembacaan data dari Register Data dilakukan untuk membaca kembali data yang tampil p LCD. Proses dilakukan dengan mengatur RS pada logika 1 yang menunjukkan adanya akses ke Regi Data. Kondisi R/W diatur pada logika tinggi yang menunjukkan adanya proses pembacaan data. Data 4 nibble tinggi (bit 7 hingga bit 4) dibaca dengan diawali adanya pulsa logika 1 pada E Clock dan dilanjut dengan data 4 bit nibble rendah (bit 3 hingga bit 0) yang juga diawali dengan pulsa logika 1 pada E Clock

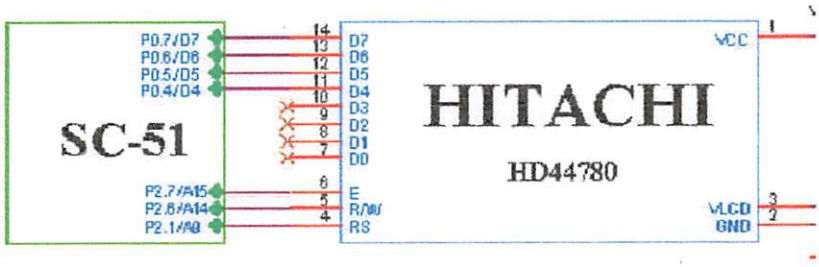


Gambar 8
Timing Diagram Pembacaan Data dari Register Data Mode 4 bit Interface

Antar muka LCD dengan mikrokontroler



Gambar 9
Antar muka dengan Modul DST-51

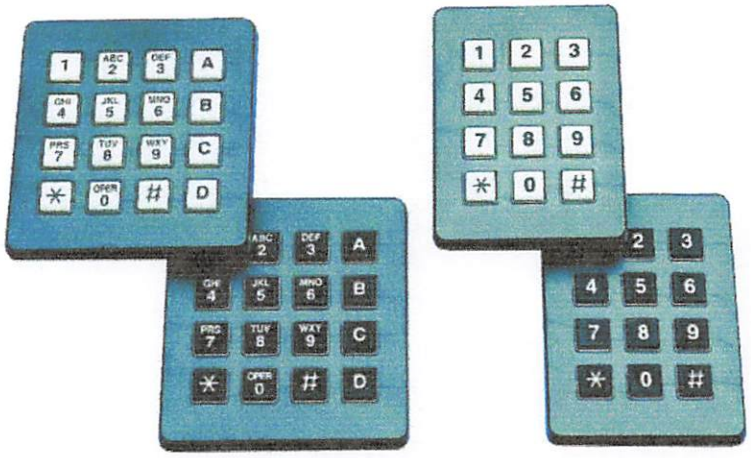


Gambar 10
Antar Muka dengan Modul SC-51 atau AT8951

Program
Rutin-rutin Program untuk DST-51 yang diassembly dengan ALDS atau ASM51
Rutin-rutin Program untuk SC-51/AT8951 yang diassembly dengan ALDS atau ASM51
Rutin delay yang diassembly dengan ALDS atau ASM51
Datasheet HD44780

SERIES 96
Conductive Rubber

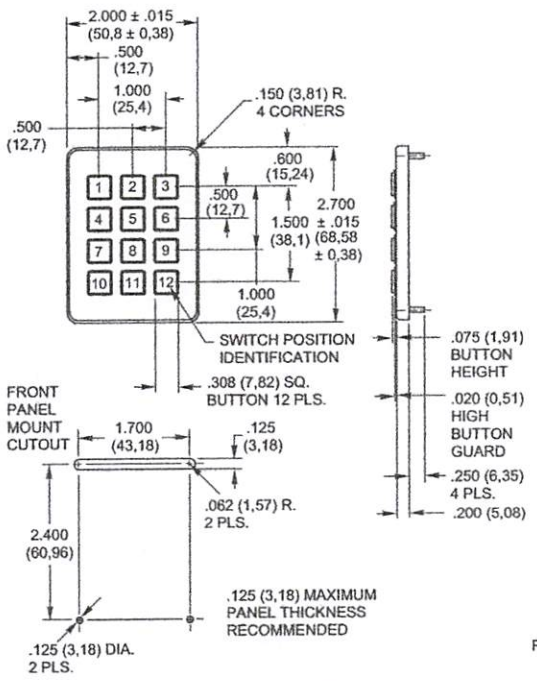
FEATURES
- Quality, Economical Keyboards
- Easily Customized Legends
- Matrix Circuitry
- Backlit and Shielded Options Available
- Termination Mates With Standard Connectors
- Tactile Feedback to Operator
- 1,000,000 Operations per Button
- Compatible With High Resistance Logic Inputs



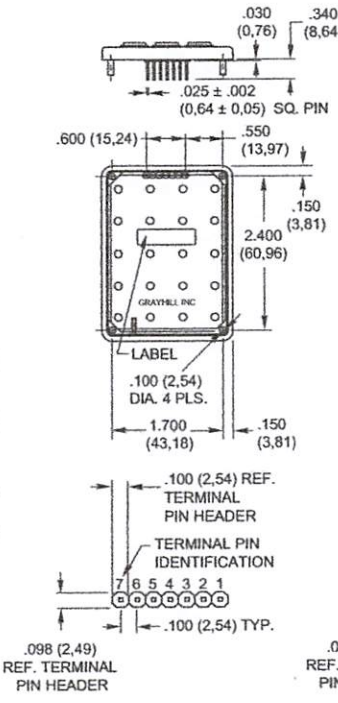
The Series 96 is Grayhill's most economical 3x4 and 4x4 keypad family. The contact system utilizes conductive rubber to mate the appropriate PC board traces. Offered in matrix circuitry, with shielded and backlit options. Built with quality component parts, the Series 96 is subjected to our rigid statistical process control to insure that it meets our reliability standards.

DIMENSIONS In inches (and millimeters)

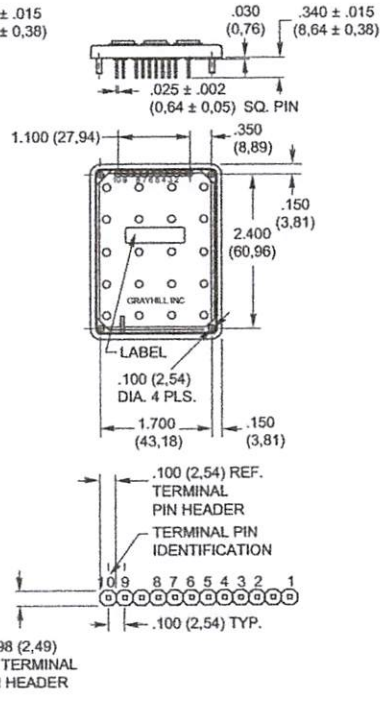
3x4 Front Mount Keyboard



Standard Versions

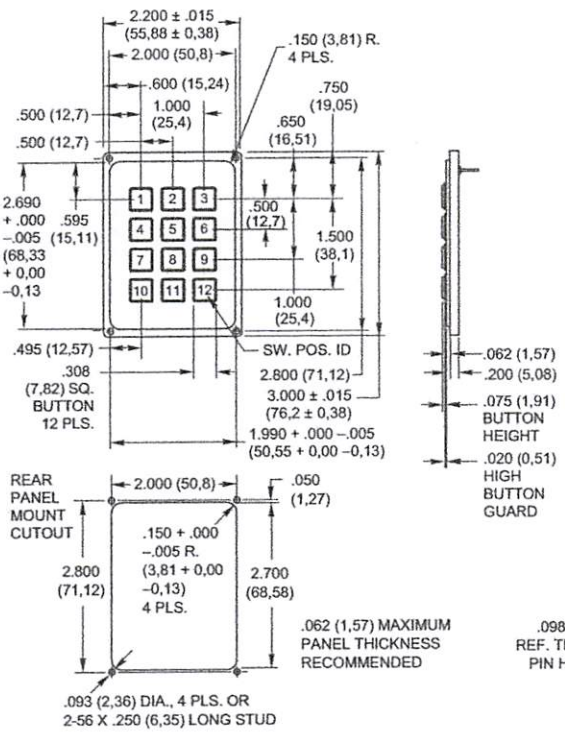


Shielded/Backlit Versions

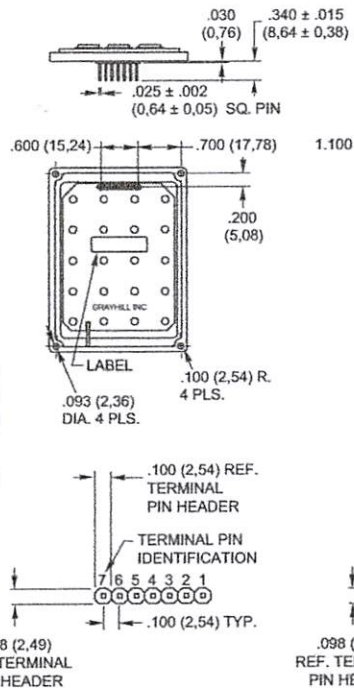


DIMENSIONS In inches (and millimeters)

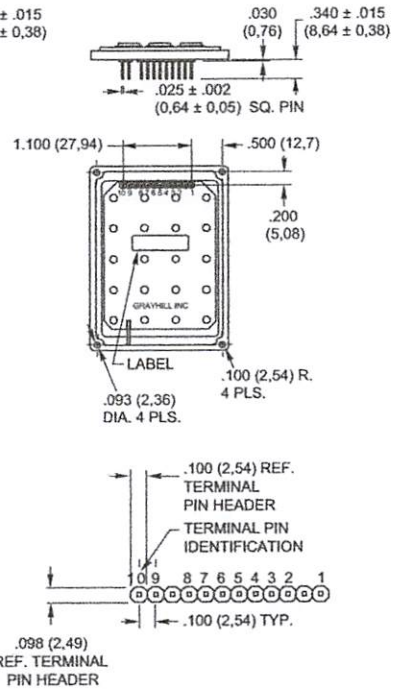
3x4 Rear Mount Keyboard



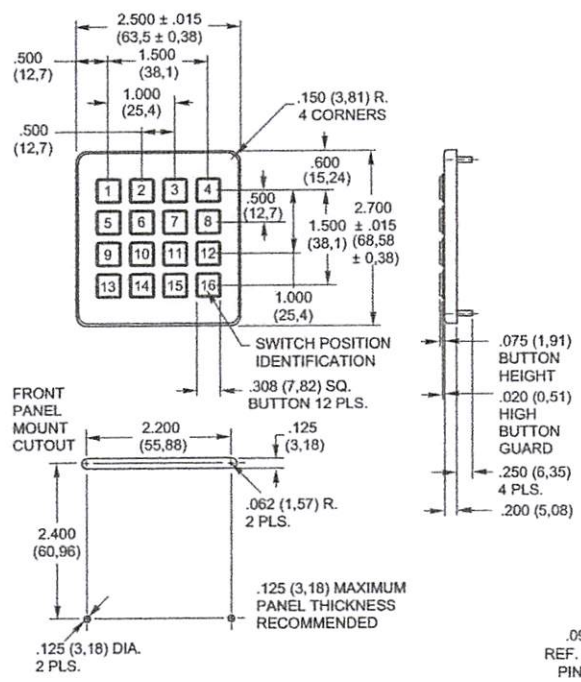
Standard Versions



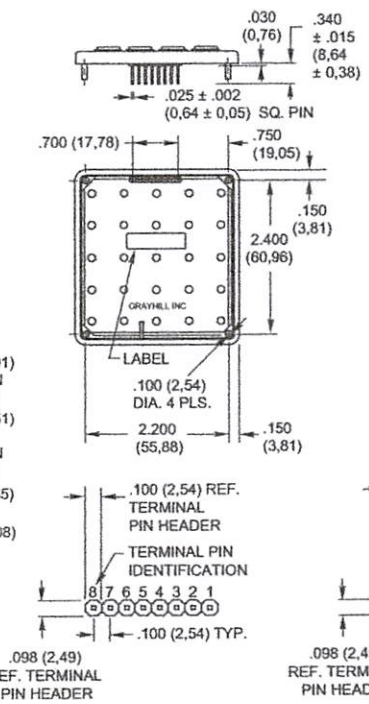
Shielded/Backlit Versions



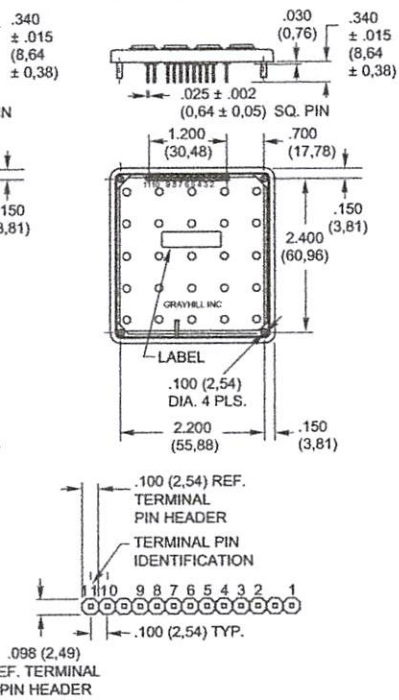
4x4 Front Mount Keyboard



Standard Versions

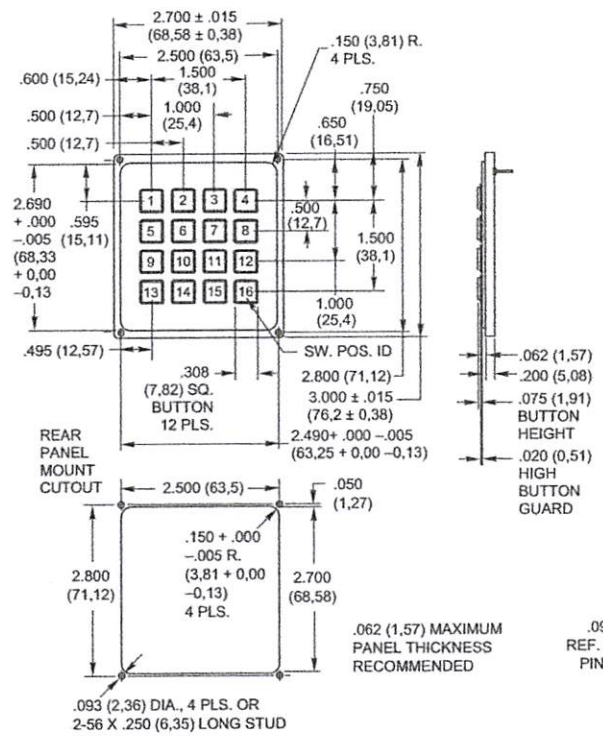


Shielded/Backlit Versions

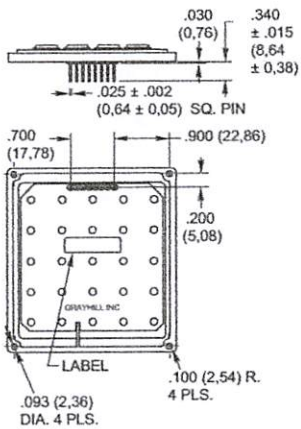


DIMENSIONS In inches (and millimeters)

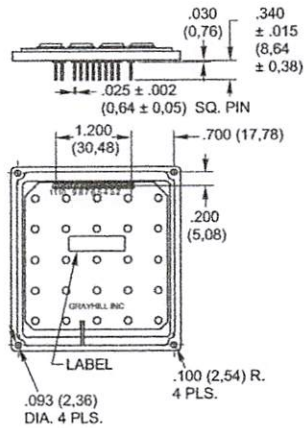
4x4 Rear Mount Keyboard



Standard Versions



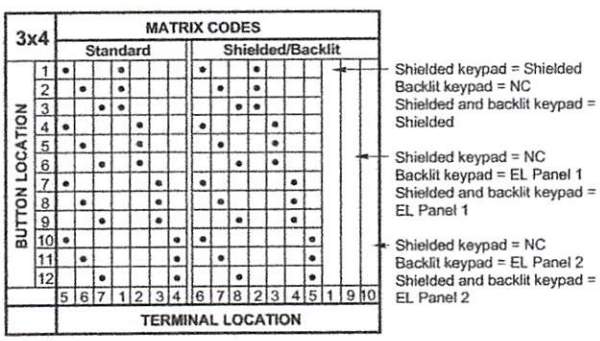
Shielded/Backlit Versions



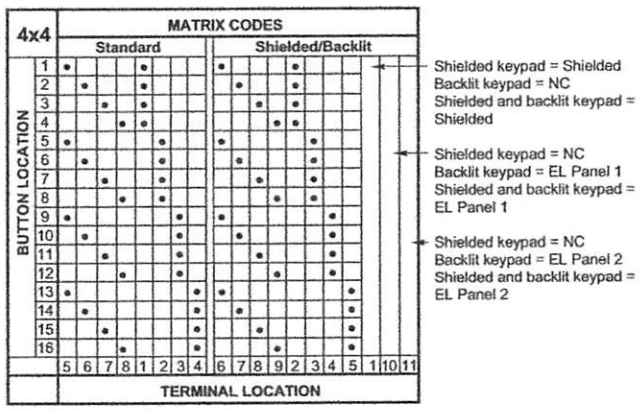
CODE AND TRUTH TABLES

Dots in the chart indicate connected terminals when switch is closed.
Terminals are identified on the keyboard.

12 Button Keypads



16 Button Keypads



SPECIFICATIONS

Rating Criteria

Rating at 12 Vdc: 5 milliamps for .5 seconds
Contact Bounce: < 12 milliseconds
Contact Resistance: < 100 ohms (at stated operating force)
Voltage Breakdown: 250 Vac between components
Mechanical Operation Life: 1,000,000 operations per key
Insulation Resistance: > 10^{12} ohms @ 500 Vdc
Push Out Force Per Pin: 5 lbs.

Operating Features

Travel: .040 minimum
Operating Force: 175 ± 40 grams
Operating Temperature: -30°C to +80°C

Material and Finishes

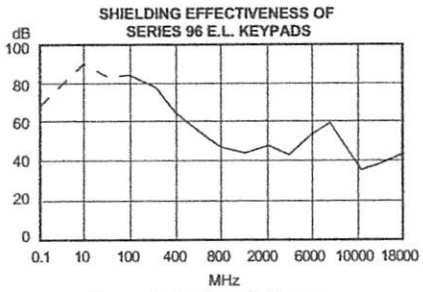
Terminal Pin: Phosphor bronze, solder-plated
PC Board: FR-4 glass cloth epoxy
Keypad: Silicone rubber, durometer 50 ± 5
Housing: ABS, cycolac "KJW"
Housing Color: Black

Shielding Effectiveness

Results shown are typical for a standard Grayhill Series 84S keyboard. A conductive gasket will generally increase the shielding, depending on the size and shape of the gasket and its material. Data derived for E-Field Radiation.

Test Method:
Measurements were made with the keyboard mounted to a brass plate, which in turn was mounted to a shielded enclosure containing the receiving equipment. A signal generator provided the frequency source that was radiated from the transmitting antenna to the enclosed receiving antenna. The spacing between antennas was maintained constant throughout the frequency range. The effectiveness rating is determined by establishing a reference reading without obstruction between the two antennas and determining the difference between that reading and the test setup reading.

Note:
When measured in actual equipment, shielding effectiveness is determined by many factors. This method accurately represents the shielding effectiveness of the Grayhill Series 84S under ideal test conditions.



Frequency M Hz	Rating in dB
0.1	≥ 66.2
10	≥ 94.8
100	90.5
400	64.2
800	42.3
2,000	40.5
6,000	33.1
10,000	34.4
18,000	37.0

STANDARD LEGENDS

Available through Grayhill Distributors

To order one of the configurations below, use the dash number shown here; select the keypad size and code, and order the part number with the appropriate legend dash number.

-102

-006

-152

-056

ORDERING INFORMATION

96AB2-102-FS-EL

Grayhill Series Number
Keyboard Size: A = 3x4, B = 4x4
Circuitry: B2 = Matrix (terminal pin header)

E.L. Panel Backlighting Option
EL = Backlit, Blank = Non-backlit
EMI/RFI Shielding Option
S = Shielded, Blank = Non-shielded
Mounting Option: F = Front panel mount, R = Rear panel mount
Standard Legend Choices
12 Position legends
102 = Black legends on a white button
152 = White legends on a black button
16 Position legends
006 = Black legends on a white button
056 = White legends on a black button

Available from your local Grayhill Distributor. For prices and discounts, contact a local Sales Office, an authorized local Distributor or Grayhill.

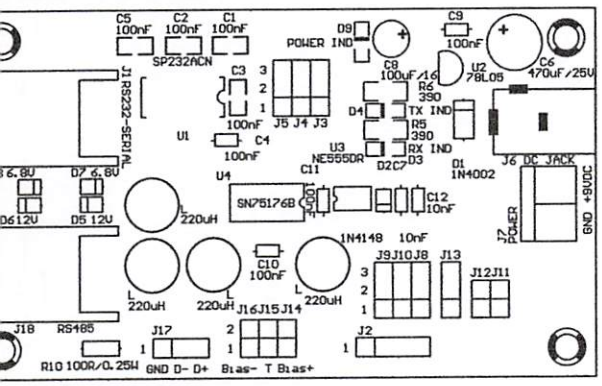
OT-I/O RS232-RS485 Converter

232-RS485 Converter merupakan suatu pengubah
l tegangan dua arah antara RS232/TTL dan RS485. Modul
apat difungsikan sebagai jalur komunikasi antara komputer
mikrokontroler berantarmuka UART RS232 dengan modul
jaringan berantarmuka UART RS485.

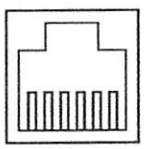
sifikasi

Mengubah level tegangan RS232 atau TTL menjadi RS485
an sebaliknya.
Mengakomodasi baud rate 300 bps hingga 115200 bps.
Dapat dikonfigurasi sebagai DCE (Data Communication
quipment) atau DTE (Data Terminal Equipment).
Arah data pada jalur RS485 dapat dikendalikan secara
anual (sisi RS232/TTL menggunakan 2 jalur data dan 1 jalur
ontrol) ataupun otomatis (sisi RS232/TTL hanya menggunakan
jalur data).
ersedia pengaturan bias+, terminator dan bias- untuk jalur
S485.
Membutuhkan tegangan + 9VDC sebagai catu daya.

Letak & Pengaturan Jumper

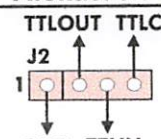
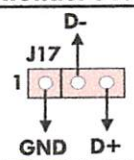


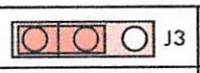
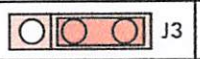
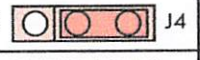
an pin konektor RJ12 (Female) :

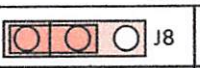
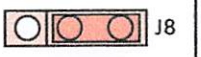
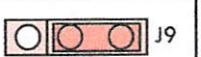
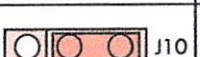


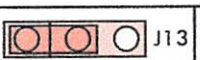
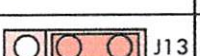
Alokasi RJ12 (J1)		
Pin	**Nama**	**Fungsi**
1	CTS	Pengendali arah data (tergantung jumper J5 & J8)
2	RTS	
3	COM	Jalur Referensi Ground
4	TXD	Jalur data keluar/masuk (tergantung jumper J3, J4, J9 & J10)
5	RXD	
6	DSR	Tidak digunakan

Alokasi RJ12 (J18)		
Pin	**Nama**	**Fungsi**
1	COM	Jalur Referensi
2	D -	Jalur B RS485
3	D -	
4	D +	Jalur A RS485
5	D +	
6	COM	Jalur Referensi

Alokasi J2	Alokasi J17
	

Konfigurasi J3, J4 dan J5 (RS232)	
	Pin RXD (J1) digunakan sebagai jalur data keluar dari modul ini (DCE)
	Pin TXD (J1) digunakan sebagai jalur data keluar dari modul ini (DTE)
	Pin TXD (J1) digunakan sebagai jalur data masuk ke modul ini (DCE)
	Pin RXD (J1) digunakan sebagai jalur data masuk ke modul ini (DTE)
	Pin RTS (J1) digunakan untuk mengendalikan arah data (DCE)
	Pin CTS (J1) digunakan untuk mengendalikan arah data (DTE)

Konfigurasi J8, J9 dan J10	
	Pin TTLCTRL (J2) digunakan untuk mengendalikan arah data
	Pin CTS atau RTS (J1) digunakan untuk mengendalikan arah data (tergantung jumper J5)
	Pin TTLOUT (J2) digunakan sebagai jalur data keluar dari modul ini
	Pin RXD atau TXD (J1) digunakan sebagai jalur data keluar dari modul ini (tergantung jumper J3)
	Pin TTLIN (J2) digunakan sebagai jalur data masuk ke modul ini
	Pin TXD atau RXD (J1) digunakan sebagai jalur data masuk ke modul ini (tergantung jumper J4)

Konfigurasi J13	
	Arah data akan dikendalikan secara otomatis (tanpa memperhatikan posisi jumper J5 dan J8)
	Arah data dikendalikan secara manual (tergantung jumper J5 dan J8)

Jika J13 berada pada posisi 1-2, maka salah satu jumper J11 atau J12 harus terpasang (tidak boleh terpasang keduanya).
J11 terpasang : untuk baud rate ≥19200 bps
J12 terpasang : untuk baud rate ≤9600 bps

Jumper J14, J15, dan J16 digunakan sebagai bias dan terminator. Dalam satu jaringan, hanya boleh terdapat satu bias +, satu bias -, dan 2 terminator (masing-masing di kedua ujung jaringan).
J14 terpasang : D+ (J17 dan J18) terhubung ke bias+

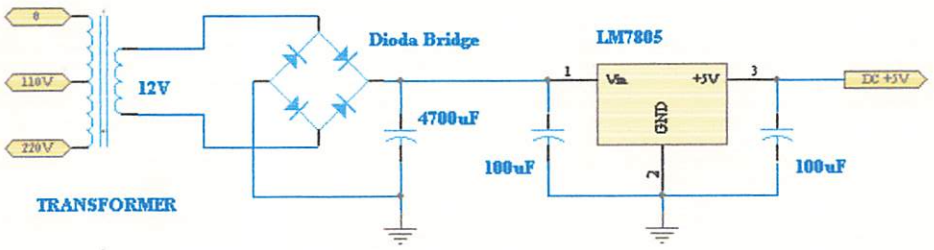
J15 terpasang : D+ dan D- (J17 dan J18) dihubungkan oleh terminator
J16 terpasang : D- (J17 dan J18) terhubung ke bias-

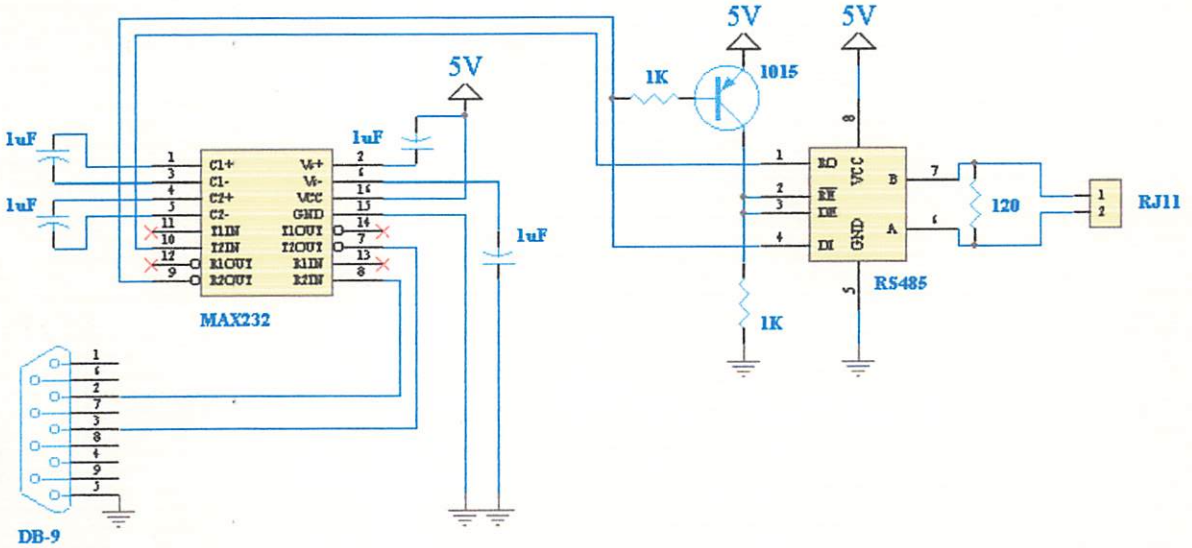
Konfigurasi kabel serial yang terdapat pada paket penjualan adalah sebagai berikut (berfungsi sebagai DCE):

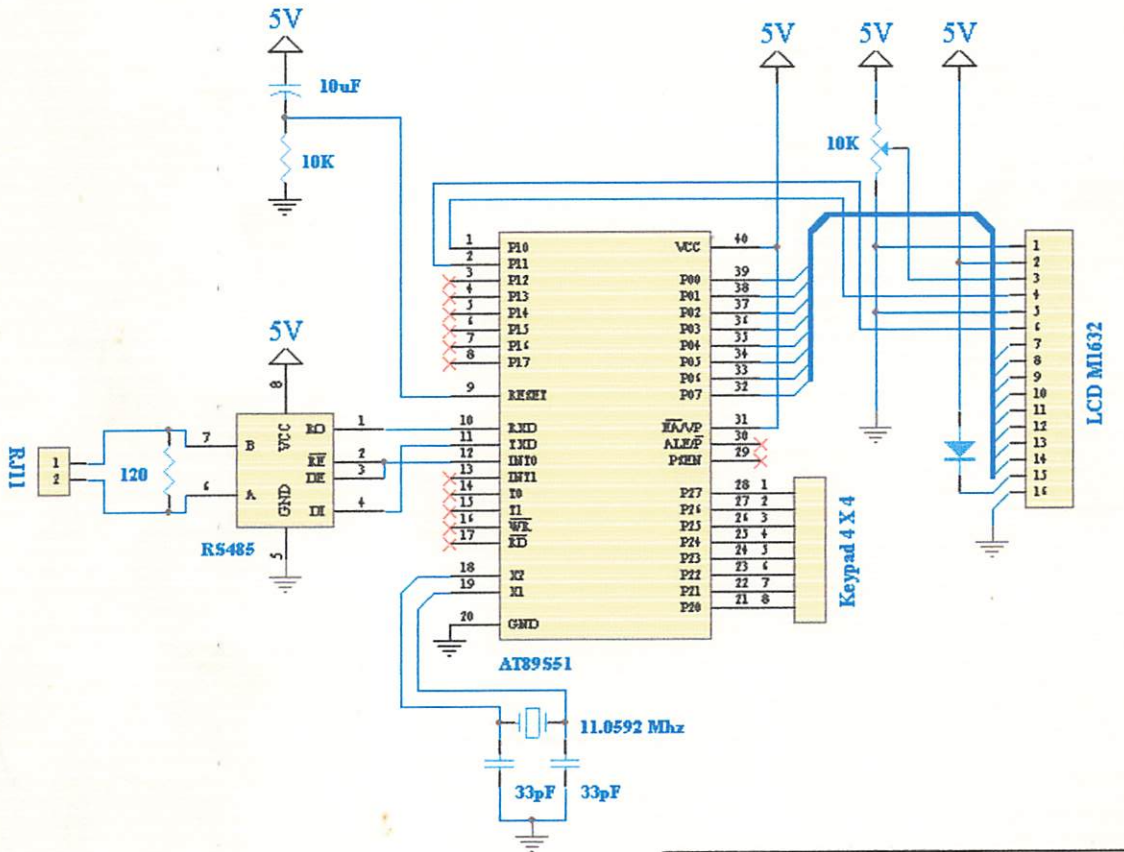
COM port Komputer DB9	RS232-RS485 Converter (J1)
RX (pin2)	RX (pin5)
TX (pin3)	TX (pin4)
GND (pin5)	GND (pin3)
RTS (pin7)	RTS (pin2)

- *Terima Kasih atas kepercayaan Anda menggunakan produk kami, bila ada kesulitan, pertanyaan atau saran mengenai produk ini silahkan menghubungi technical support kami :*

support@innovativeelectronics.com







Title		
Size: B	Number	Revision
Date:	30-Oct-2008	Sheet of
File:	C:\MYDOCU~1\REPVIANY.SCH	Drawn By:

```
unit Unit1;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, ExtCtrls, StdCtrls, Grids, DBGrids, OleCtrls, MSCommLib_TLB,
  Crystal_TLB, DB, ADODB, Buttons, Menus;

type
  RMenu = Record
    No_Menu:string;
    NamaMenu:string;
    Golongan:string;
    Harga:Real;
  end;
  RPesanan = Record
    No_Pesan: string;
    Tgl_Pesan: string;
    No_Meja: string;
    Total: string;
  end;
  RItem_Pesanan = Record
     No_ItemPesan:string;
     No_Pesan:string;
     No_Menu:string;
     Harga:string;
     Jumlah:string;
     SubTotal:string;
  end;
  TForm1 = class(TForm)
    MSComm1: TMSComm;
    Label1: TLabel;
    Timer1: TTimer;
    Label2: TLabel;
    LTanggal: TLabel;
    Label3: TLabel;
    TxtNoMeja1: TEdit;
    Label4: TLabel;
    Label5: TLabel;
    TxtNoPesanan1: TEdit;
    TxtNoMeja2: TEdit;
    TxtNoPesanan2: TEdit;
    Label6: TLabel;
    TxtTotal1: TStaticText;
    Label7: TLabel;
```

```
  TxtTotal2: TStaticText;
  Tabel1: TStringGrid;
  Tabel2: TStringGrid;
  CR1: TCrystalReport;
  AdoConn: TADOConnection;
  Search: TADOQuery;
  CmdSave1: TBitBtn;
  CmdSave2: TBitBtn;
  Timer2: TTimer;
  MainMenu1: TMainMenu;
  SettingMenu1: TMenuItem;
  Exit1: TMenuItem;
  procedure Timer1Timer(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure CmdSave1Click(Sender: TObject);
  procedure CmdSave2Click(Sender: TObject);
  procedure Timer2Timer(Sender: TObject);
  procedure Exit1Click(Sender: TObject);
  procedure SettingMenu1Click(Sender: TObject);
 private
  { Private declarations }
  No_Meja,No1,No2,a:byte;
  Stat_Tabel:Boolean;
  Wkt,Wkt2:integer;
  buffer,Buffer1:string;
  Sqlcmd: string;
  Menu:Rmenu;
  Item_Pesanan:RItem_Pesanan;
  Pesanan:RPesanan;
  Array_Menu:array[1..25]of integer;
 public
  { Public declarations }
  Procedure Atur_Tabel(Tabel: TStringGrid);
  Procedure Setting_Awal1;
  Procedure Setting_Awal2;
  Procedure Setting_Awal;
  Function Find_Menu(No_Menu: string):boolean;
  Function Find_Tabel(Tabel: TStringGrid;No:byte;No_Menu:String):Boolean;
  Function Hitung_Total(Tabel: TStringGrid;No:byte):Real;
  Function Get_No_Pemesanan1():string;
  Function Get_No_Pemesanan2():string;
  Procedure Update_StatusMenu;
 end;

var
 Form1: TForm1;
```

```
implementation

uses Unit2;

{$R *.dfm}
procedure TForm1.Timer1Timer(Sender: TObject);
var
  i:integer;
  No_Menu,Juml_Pesan:integer;
begin
  case wkt of
    1: begin
        if not mscomm1.PortOpen then mscomm1.PortOpen:=true;
        mscomm1.Output:=chr(223+No_Meja);
        inc(wkt);
      end;
    2: begin
        Buffer:= MSComm1.Input;
        If Not (Buffer = '') Then
        begin
          if length(buffer)>=3 then
          begin
            buffer1:='';
            No_Menu:=0;
            For i:= 1 To Length(Buffer) do
              buffer1:= buffer1+ inttostr(ord(buffer[i]));
            For i:=1 to length(buffer1) do
            begin
              if i mod 3 =1 then
              begin
                No_Menu:=(strtoint(buffer1[i])-1) * 5;
              end
              else if i mod 3= 2 then
              begin
                No_Menu:=No_Menu + strtoint(buffer1[i]);
                Menu.No_Menu:= copy('00',1,2-length(inttostr(no_menu))) +
inttostr(no_menu);
                if No_Meja=1 then
                begin
                  if txtNoPesanan1.Text ='' then
txtNopesanan1.text:=Get_No_Pemesanan1;
                  If Find_Menu(Menu.No_Menu) then
                  begin
                    if not find_Tabel(tabel1,No1,Menu.No_Menu) then
                    begin
```

```
                    inc(no1);
                    if No1>1 then Tabel1.RowCount:=Tabel1.RowCount+1;
                    Tabel1.Cells[0,No1]:=inttostr(no1);
                    Tabel1.Cells[1,No1]:=Menu.No_Menu;
                    Tabel1.Cells[2,No1]:= menu.NamaMenu;
                    Tabel1.Cells[3,No1]:= Floattostr(menu.Harga);
                  end;
                end;
              end
              else
              begin
                if txtNoPesanan2.Text ='' then TxtNoPesanan2.text:=
Get_No_Pemesanan2;
                If Find_Menu(Menu.No_Menu) then
                begin
                  if not find_Tabel(tabel2,No2,Menu.No_Menu) then
                  begin
                    inc(no2);
                    if No2>1 then Tabel2.RowCount:=Tabel2.RowCount+1;
                    Tabel2.Cells[0,No2]:=inttostr(no2);
                    Tabel2.Cells[1,No2]:=Menu.No_Menu;
                    Tabel2.Cells[2,No2]:= menu.NamaMenu;
                    Tabel2.Cells[3,No2]:= Floattostr(menu.Harga);
                  end;
                end;
              end
            end
            else if i mod 3= 0 then
            begin
              Juml_Pesan:= strtoint(buffer1[i]);
              if No_Meja=1 then
              begin
                if stat_Tabel then
                begin
                  No1:=a;
                  tabel1.Cells[4,No1]:=inttostr(strtoint(tabel1.Cells[4,No1]) +
Juml_pesan);
                end
                else
                  tabel1.Cells[4,No1]:=inttostr(Juml_pesan);
                tabel1.Cells[5,No1]:=Floattostr(strtofloat(tabel1.Cells[3,No1])*
strtofloat(tabel1.Cells[4,No1]));
                No1:=tabel1.RowCount-1;
                txttotal1.Caption:=floattostr(Hitung_Total(Tabel1,No1));
              end
              else
```

```
                    begin
                      if stat_Tabel then
                      begin
                        No2:=a;
                        tabel2.Cells[4,No2]:=inttostr(strtoint(tabel2.Cells[4,No2]) +
Juml_pesan);
                      end
                      else
                        tabel2.Cells[4,No2]:=inttostr(Juml_pesan);
                      tabel2.Cells[5,No2]:=Floattostr(strtofloat(tabel2.Cells[3,No2])*
strtofloat(tabel2.Cells[4,No2]));
                      No2:=tabel2.RowCount-1;
                      txttotal2.Caption:=floattostr(Hitung_Total(Tabel2,No2));
                    end;
                  end;
                end;
              end;
            end;
            wkt:=1;
            if No_Meja =1 then No_Meja:=2
            else No_Meja:=1;
          end;
    end;
end;

procedure TForm1.FormCreate(Sender: TObject);
begin
    AdoConn.Open;
    if not(Mscomm1.PortOpen) then Mscomm1.PortOpen:=True;
    setting_awal;
end;

procedure TForm1.Atur_Tabel(Tabel: TStringGrid);
var
    i:byte;
begin
    Tabel.ColCount:=6;
    Tabel.RowCount:=2;
    For i:=0 to tabel.ColCount-1 do
      Tabel.Cells[i,1]:='';
    Tabel.cells[0,0]:='No';
    Tabel.cells[1,0]:='No Menu';
    Tabel.cells[2,0]:='Nama Menu';
    Tabel.ColWidths[2]:=110;
    Tabel.cells[3,0]:='Harga';
    Tabel.cells[4,0]:='Jumlah';
```

```
  Tabel.cells[5,0]:='Sub Total';
end;

procedure TForm1.Setting_Awal1;
begin
  No1:=0;
  TxtNoPesanan1.Text:='';
  TxtNoMeja1.Text:='01';
  TxtNoPesanan1.Enabled:=false;
  TxtNoMeja1.Enabled:=false;
  Txttotal1.Caption:='';
  Atur_tabel(tabel1);
end;

function TForm1.Find_Menu(No_Menu: string): boolean;
begin
  Find_Menu:=False;
  with search do
  begin
    if not (trim(No_Menu)='') then
    begin
      Sqlcmd:= 'Select NamaMenu,Harga From Menu Where No_Menu='''+ No_Menu
+''' ';
      close;
      sql.Clear;
      sql.Add(sqlcmd);
      open;
      if not (isempty) then
      begin
        Find_Menu:=True;
        Menu.NamaMenu:= Fieldbyname('NamaMenu').AsString;
        Menu.Harga:= Fieldbyname('Harga').AsCurrency;
      end;
    end;
  end;
end;

function TForm1.Find_Tabel(Tabel: TStringGrid;No:byte;No_Menu:string): Boolean;
Var
  i:integer;
begin
  Find_Tabel:= False;
  Stat_Tabel:= False;
  For i:= 1 To No do
  begin
    If Tabel.Cells[1, i] = No_Menu  Then
```

```
      begin
        Find_Tabel:= True;
        Stat_Tabel:= True;
        break;
      end;
    end;
    a:=i;
end;

function TForm1.Hitung_Total(Tabel: TStringGrid; No: byte): Real;
Var
    i:byte;
    Total:real;
begin
    Total:=0;
    For i:=1 to No do
      Total:=Total + Strtofloat(Tabel.Cells[5,i]);
    Hitung_Total:=Total;
end;

function TForm1.Get_No_Pemesanan1: string;
Var
    Temp:string;
begin
    Get_No_Pemesanan1:='';
    Temp:=Formatdatetime('yyMMdd',date);
    with search do
    begin
      Sqlcmd:= 'Select Top 1 No_Pesan From Pesanan Where No_Pesan like "'+ Temp
+'%" order by No_Pesan desc ';
      close;
      sql.Clear;
      sql.Add(sqlcmd);
      open;
      if not (isempty) then
      begin
        if txtnoPesanan2.Text ='' then
          Get_No_Pemesanan1:=Temp + copy('0000',1,4-
length(Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+1)))+
          Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+1)
        else
          Get_No_Pemesanan1:=Temp + copy('0000',1,4-
length(Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+2)))+
          Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+2);
      end
      else
```

```
    begin
      if txtnoPesanan2.Text ='' then
        Get_No_Pemesanan1:=Temp + '0001'
      else
        Get_No_Pemesanan1:=Temp + '0002';
    end;
  end;
end;

function TForm1.Get_No_Pemesanan2: string;
Var
  Temp:string;
begin
  Get_No_Pemesanan2:='';
  Temp:=Formatdatetime('yyMMdd',date);
  with search do
  begin
    Sqlcmd:= 'Select Top 1 No_Pesan From Pesanan Where No_Pesan like "'+ Temp
+'%" order by No_Pesan desc ';
    close;
    sql.Clear;
    sql.Add(sqlcmd);
    open;
    if not (isempty) then
    begin
      if txtnoPesanan1.Text ='' then
        Get_No_Pemesanan2:=Temp + copy('0000',1,4-
length(Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+1)))+
        Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+1)
      else
        Get_No_Pemesanan2:=Temp + copy('0000',1,4-
length(Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+2)))+
        Floattostr(strtofloat(copy(Fieldbyname('No_Pesan').AsString,7,4))+2)
    end
    else
    begin
      if txtnoPesanan1.Text ='' then
        Get_No_Pemesanan2:=Temp + '0001'
      else
        Get_No_Pemesanan2:=Temp + '0002';
    end

  end;

end;
```

```
procedure TForm1.CmdSave1Click(Sender: TObject);
Var
  i:integer;
begin
  adoconn.BeginTrans;
  With Pesanan do
  begin
    No_Pesan:=txtnoPesanan1.Text;
    Tgl_Pesan:=Ltanggal.Caption;
    No_Meja:=txtnomeja1.Text;
    Total:=txttotal1.Caption;
    sqlcmd:= 'INSERT INTO PESANAN VALUES('''+ No_Pesan +''','''+ tgl_Pesan +''',
' +
        ''''+ No_Meja +''','+ Total +') ';
    adoconn.Execute(sqlcmd);
  end;
  With Item_Pesanan do
  begin
    For i:=1 to No1 do
    begin
      No_ItemPesan:=Pesanan.No_Pesan + copy('00',1,2-length(inttostr(i))) +
inttostr(i);
      No_Pesan:=Pesanan.No_Pesan;
      No_Menu:=Tabel1.Cells[1,i];
      Harga:= Tabel1.Cells[3,i];
      Jumlah:=Tabel1.Cells[4,i];
      SubTotal:=Tabel1.Cells[5,i];
      sqlcmd:='INSERT INTO ITEM_PESANAN VALUES('''+ No_ItemPesan +''','''+
No_Pesan +''', ' +
        ''''+ No_Menu +''','+ Harga +','+ Jumlah +','+ SubTotal +') ';
      adoconn.Execute(sqlcmd);
    end;
  end;
  adoconn.CommitTrans;
  CR1.ReportFileName:= 'report1.rpt';
  CR1.SelectionFormula:='{Pesanan.No_Pesan}='''+ Pesanan.No_Pesan +''' ';
  CR1.Action:=1;
  Setting_Awal1;
end;

procedure TForm1.CmdSave2Click(Sender: TObject);
Var
  i:integer;
begin
  adoconn.BeginTrans;
  With Pesanan do
```

```
  begin
    No_Pesan:=txtnoPesanan2.Text;
    Tgl_Pesan:=Ltanggal.Caption;
    No_Meja:=txtnomeja2.Text;
    Total:=txttotal2.Caption;
    sqlcmd:= 'INSERT INTO PESANAN VALUES('''+ No_Pesan +''','''+ tgl_Pesan +''',
' +
        ''''+ No_Meja +''','+ Total +') ';
    adoconn.Execute(sqlcmd);
  end;
  With Item_Pesanan do
  begin
    For i:=1 to No2 do
    begin
      No_ItemPesan:=Pesanan.No_Pesan + copy('00',1,2-length(inttostr(i))) +
inttostr(i);
      No_Pesan:=Pesanan.No_Pesan;
      No_Menu:=Tabel2.Cells[1,i];
      Harga:= Tabel2.Cells[3,i];
      Jumlah:=Tabel2.Cells[4,i];
      SubTotal:=Tabel2.Cells[5,i];
      sqlcmd:='INSERT INTO ITEM_PESANAN VALUES('''+ No_ItemPesan +''','''+
No_Pesan +''', ' +
        ''''+ No_Menu +''','+ Harga +','+ Jumlah +','+ SubTotal +') ';
      adoconn.Execute(sqlcmd);
    end;
  end;
  adoconn.CommitTrans;
  CR1.ReportFileName:= 'report1.rpt';
  CR1.SelectionFormula:='{Pesanan.No_Pesan}='''+ Pesanan.No_Pesan +''' ';
  CR1.Action:=1;
  Setting_Awal2;
end;

procedure TForm1.Update_StatusMenu;
Var
  Temp:string;
  i:integer;
begin
  For i:=1 to 25 do
    Array_Menu[i]:=1;
  with search do
  begin
    sqlcmd:='Select Top 1 * From Menu order by No_Menu';
    close;
    sql.Clear;
```

```
      sql.Add(sqlcmd);
      open;
      i:=1;
      While not (isempty) do
      begin
        Temp:=Fieldbyname('No_Menu').AsString;
        if FieldbyName('Status').AsBoolean then
          array_Menu[i]:=1
        else
          array_Menu[i]:=2;
        sqlcmd:='Select Top 1 * From Menu Where No_Menu > '"+ Temp +'" order by
No_Menu';
        close;
        sql.Clear;
        sql.Add(sqlcmd);
        open;
        inc(i);
      end;
      Wkt2:=1;
      Timer2.enabled:=True;
   end;
end;

procedure TForm1.Timer2Timer(Sender: TObject);
Var
   Buf,i:integer;
begin
   case wkt2 of
   1: begin
        if not mscomm1.PortOpen then mscomm1.PortOpen:=true;
        mscomm1.Output:= chr(239+no_meja);
        inc(wkt2);
      end;
   2: begin
        buf:=0;
        buffer:=mscomm1.Input;
        If Not (Buffer = '') Then
        begin
          For i:= 1 To Length(Buffer) do
            buf:= ord(buffer[i]);
          if buf=255 then
          begin
            For i:=1 to 25 do
            begin
              mscomm1.Output:=chr(array_menu[i]);
              sleep(50);
```

```
            end;
            if No_Meja=2 then
            begin
              timer2.Enabled:=false;
              wkt:=1;
              mscomm1.PortOpen:=false;
              timer1.Enabled:=true;
            end
            else
            begin
              Wkt2:=1;
              inc(No_Meja);
            end;
          end;
        end;
      end;
    end;
end;

procedure TForm1.Exit1Click(Sender: TObject);
begin
  close;
end;

procedure TForm1.SettingMenu1Click(Sender: TObject);
begin
  timer1.Enabled:=false;
  timer2.Enabled:=false;
  Form1.Hide;
  Form2.show;
end;

procedure TForm1.Setting_Awal;
begin
  No_Meja:=1;
  wkt:=1;
  Wkt2:=1;
  Setting_awal1;
  Setting_awal2;
  Ltanggal.Caption :=formatdatetime('dd/MM/yyyy',date);
  Timer1.Enabled:=false;
  Timer2.Enabled:=false;
  update_StatusMenu;
end;

procedure TForm1.Setting_Awal2;
```

```
begin
  No2:=0;
  TxtNoPesanan2.Text:='';
  TxtNoMeja2.Text:='02';
  TxtNoPesanan2.Enabled:=false;
  TxtNoMeja2.Enabled:=false;
  Txttotal2.Caption:='';
  update_StatusMenu;
  Atur_tabel(tabel2);
end;

end.
```

```
        org     00h
        ljmp    init
;
        org     23h
        clr     ES
        jnb     RI,$
        clr     RI
        mov     R7,SBUF
        setb    ES
        reti
;
        Rest    Bit P1.0            ; reset LCD
        Enbl    Bit P1.1            ; enable LCD
        Slct    Bit P3.2            ; selector RS485 -> TX/RX
        Sm00    Equ 30h             ; status menu
        Sm01    Equ 31h
        Sm02    Equ 32h
        Sm03    Equ 33h
        Sm04    Equ 34h
        Sm05    Equ 35h
        Sm06    Equ 36h
        Sm07    Equ 37h
        Sm08    Equ 38h
        Sm09    Equ 39h
        Sm10    Equ 3Ah
        Sm11    Equ 3Bh
        Sm12    Equ 3Ch
        Sm13    Equ 3Dh
        Sm14    Equ 3Eh
        Sm15    Equ 3Fh
        Sm16    Equ 40h
        Sm17    Equ 41h
        Sm18    Equ 42h
        Sm19    Equ 43h
        Sm20    Equ 44h
        Sm21    Equ 45h
        Sm22    Equ 46h
        Sm23    Equ 47h
        Sm24    Equ 48h
;
        Ps0m    Equ 50h             ; pesan master menu
        Ps0s    Equ 51h             ; pesan sub menu
        Ps0j    Equ 52h             ; pesan jumlah menu
        Ps1m    Equ 53h
        Ps1s    Equ 54h
        Ps1j    Equ 55h
        Ps2m    Equ 56h
        Ps2s    Equ 57h
        Ps2j    Equ 58h
        Ps3m    Equ 59h
        Ps3s    Equ 5Ah
        Ps3j    Equ 5Bh
        Ps4m    Equ 5Ch
        Ps4s    Equ 5Dh
        Ps4j    Equ 5Eh
        Hurf    Equ 60h
        Dly0    Equ 61h
```

```
        Dly1    Equ 62h
        Dly2    Equ 63h
;
init:   lcall   lcd_in              ; inisialisasi LCD
        lcall   srl_in              ; inisialisasi serial
        lcall   stsmnu              ; reset status menu
        lcall   rstpsn              ; reset pesan menu
        mov     R5,#0               ; reset jumlah pesan
        mov     R7,#0FFh            ; reset request data
;
mulai:  mov     DPTR,#nama
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#nim
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   delay2
        mov     DPTR,#jurs
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#univ
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   delay2
;
loop:   mov     DPTR,#judul1
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#judul2
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#judul3
        lcall   delay3
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     DPTR,#judul4
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   delay3
        ljmp    loop
;
mamenu: mov     DPTR,#tmamnu
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
mamen0: cjne    R1,#1,mamen1
        mov     DPTR,#tmamn0
mamen1: cjne    R1,#2,mamen2
        mov     DPTR,#tmamn1
```

```
mamen2: cjne    R1,#3,mamen3
        mov     DPTR,#tmamn2
mamen3: cjne    R1,#4,mamen4
        mov     DPTR,#tmamn3
mamen4: cjne    R1,#5,mamen5
        mov     DPTR,#tmamn4
mamen5: lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   tg_lps
mamen6: lcall   scnkpd
        cjne    R0,#15,mamen8
        dec     R1
        cjne    R1,#0,mamen7
        mov     R1,#1
mamen7: ljmp    mamen0
mamen8: cjne    R0,#16,mamenA
        inc     R1
        cjne    R1,#6,mamen9
        mov     R1,#5
mamen9: ljmp    mamen0
mamenA: cjne    R0,#11,mamenB
        mov     SP,#07h
        mov     R7,#0FFh
        ljmp    loop
mamenB: cjne    R0,#12,mamen6
;
sbmen0: cjne    R1,#1,sbmen1
        mov     DPTR,#tmamn0
sbmen1: cjne    R1,#2,sbmen2
        mov     DPTR,#tmamn1
sbmen2: cjne    R1,#3,sbmen3
        mov     DPTR,#tmamn2
sbmen3: cjne    R1,#4,sbmen4
        mov     DPTR,#tmamn3
sbmen4: cjne    R1,#5,sbmen5
        mov     DPTR,#tmamn4
sbmen5: lcall   line1
        mov     Hurf,#16
        lcall   tulis
        mov     R2,#1
;
sbmen6: cjne    R1,#1,sbmen7
sbmn00: cjne    R2,#1,sbmn01
        mov     R3,Sm00             ;\
admn00: cjne    R3,#1,tdmn00        ; | cek status menu 1
        mov     DPTR,#tsmn00        ; | ada / tidak
tdmn00: cjne    R3,#2,sbmn01        ; |
        mov     DPTR,#mnkosg        ;/
sbmn01: cjne    R2,#2,sbmn02
        mov     R3,Sm01             ;\
admn01: cjne    R3,#1,tdmn01        ; | cek status menu 2
        mov     DPTR,#tsmn01        ; | ada / tidak
tdmn01: cjne    R3,#2,sbmn02        ; |
        mov     DPTR,#mnkosg        ;/
sbmn02: cjne    R2,#3,sbmn03
        mov     R3,Sm02             ;\
```

```
admn02: cjne    R3,#1,tdmn02        ; | cek status menu 3
        mov     DPTR,#tsmn02        ; | ada / tidak
tdmn02: cjne    R3,#2,sbmn03        ; |
        mov     DPTR,#mnkosg        ;/
sbmn03: cjne    R2,#4,sbmn04
        mov     R3,Sm03             ;\
admn03: cjne    R3,#1,tdmn03        ; | cek status menu 4
        mov     DPTR,#tsmn03        ; | ada / tidak
tdmn03: cjne    R3,#2,sbmn04        ; |
        mov     DPTR,#mnkosg        ;/
sbmn04: cjne    R2,#5,sbmn05
        mov     R3,Sm04             ;\
admn04: cjne    R3,#1,tdmn04        ; | cek status menu 5
        mov     DPTR,#tsmn04        ; | ada / tidak
tdmn04: cjne    R3,#2,sbmn05        ; |
        mov     DPTR,#mnkosg        ;/
sbmn05: ljmp    sbmenB
;
sbmen7: cjne    R1,#2,sbmen8
sbmn10: cjne    R2,#1,sbmn11
        mov     R3,Sm05             ;\
admn10: cjne    R3,#1,tdmn10        ; | cek status menu 6
        mov     DPTR,#tsmn10        ; | ada / tidak
tdmn10: cjne    R3,#2,sbmn11        ; |
        mov     DPTR,#mnkosg        ;/
sbmn11: cjne    R2,#2,sbmn12
        mov     R3,Sm06             ;\
admn11: cjne    R3,#1,tdmn11        ; | cek status menu 7
        mov     DPTR,#tsmn11        ; | ada / tidak
tdmn11: cjne    R3,#2,sbmn12        ; |
        mov     DPTR,#mnkosg        ;/
sbmn12: cjne    R2,#3,sbmn13
        mov     R3,Sm07             ;\
admn12: cjne    R3,#1,tdmn12        ; | cek status menu 8
        mov     DPTR,#tsmn12        ; | ada / tidak
tdmn12: cjne    R3,#2,sbmn13        ; |
        mov     DPTR,#mnkosg        ;/
sbmn13: cjne    R2,#4,sbmn14
        mov     R3,Sm08             ;\
admn13: cjne    R3,#1,tdmn13        ; | cek status menu 9
        mov     DPTR,#tsmn13        ; | ada / tidak
tdmn13: cjne    R3,#2,sbmn14        ; |
        mov     DPTR,#mnkosg        ;/
sbmn14: cjne    R2,#5,sbmn15
        mov     R3,Sm09             ;\
admn14: cjne    R3,#1,tdmn14        ; | cek status menu 10
        mov     DPTR,#tsmn14        ; | ada / tidak
tdmn14: cjne    R3,#2,sbmn15        ; |
        mov     DPTR,#mnkosg        ;/
sbmn15: ljmp    sbmenB
;
sbmen8: cjne    R1,#3,sbmen9
sbmn20: cjne    R2,#1,sbmn21
        mov     R3,Sm10             ;\
admn20: cjne    R3,#1,tdmn20        ; | cek status menu 11
        mov     DPTR,#tsmn20        ; | ada / tidak
tdmn20: cjne    R3,#2,sbmn21        ; |
```

```
        mov     DPTR,#mnkosg        ;/
sbmn21: cjne    R2,#2,sbmn22
        mov     R3,Sm11             ;\
admn21: cjne    R3,#1,tdmn21        ; | cek status menu 12
        mov     DPTR,#tsmn21        ; | ada / tidak
tdmn21: cjne    R3,#2,sbmn22        ; |
        mov     DPTR,#mnkosg        ;/
sbmn22: cjne    R2,#3,sbmn23
        mov     R3,Sm12             ;\
admn22: cjne    R3,#1,tdmn22        ; | cek status menu 13
        mov     DPTR,#tsmn22        ; | ada / tidak
tdmn22: cjne    R3,#2,sbmn23        ; |
        mov     DPTR,#mnkosg        ;/
sbmn23: cjne    R2,#4,sbmn24
        mov     R3,Sm13             ;\
admn23: cjne    R3,#1,tdmn23        ; | cek status menu 14
        mov     DPTR,#tsmn23        ; | ada / tidak
tdmn23: cjne    R3,#2,sbmn24        ; |
        mov     DPTR,#mnkosg        ;/
sbmn24: cjne    R2,#5,sbmn25
        mov     R3,Sm14             ;\
admn24: cjne    R3,#1,tdmn24        ; | cek status menu 15
        mov     DPTR,#tsmn24        ; | ada / tidak
tdmn24: cjne    R3,#2,sbmn25        ; |
        mov     DPTR,#mnkosg        ;/
sbmn25: ljmp    sbmenB
;
sbmen9: cjne    R1,#4,sbmenA
sbmn30: cjne    R2,#1,sbmn31
        mov     R3,Sm15             ;\
admn30: cjne    R3,#1,tdmn30        ; | cek status menu 16
        mov     DPTR,#tsmn30        ; | ada / tidak
tdmn30: cjne    R3,#2,sbmn31        ; |
        mov     DPTR,#mnkosg        ;/
sbmn31: cjne    R2,#2,sbmn32
        mov     R3,Sm16             ;\
admn31: cjne    R3,#1,tdmn31        ; | cek status menu 17
        mov     DPTR,#tsmn31        ; | ada / tidak
tdmn31: cjne    R3,#2,sbmn32        ; |
        mov     DPTR,#mnkosg        ;/
sbmn32: cjne    R2,#3,sbmn33
        mov     R3,Sm17             ;\
admn32: cjne    R3,#1,tdmn32        ; | cek status menu 18
        mov     DPTR,#tsmn32        ; | ada / tidak
tdmn32: cjne    R3,#2,sbmn33        ; |
        mov     DPTR,#mnkosg        ;/
sbmn33: cjne    R2,#4,sbmn34
        mov     R3,Sm18             ;\
admn33: cjne    R3,#1,tdmn33        ; | cek status menu 19
        mov     DPTR,#tsmn33        ; | ada / tidak
tdmn33: cjne    R3,#2,sbmn34        ; |
        mov     DPTR,#mnkosg        ;/
sbmn34: cjne    R2,#5,sbmn35
        mov     R3,Sm19             ;\
admn34: cjne    R3,#1,tdmn34        ; | cek status menu 20
        mov     DPTR,#tsmn34        ; | ada / tidak
tdmn34: cjne    R3,#2,sbmn35        ; |
```

```
        mov     DPTR,#mnkosg        ;/
sbmn35: ljmp    sbmenB
;
sbmenA: cjne    R1,#5,sbmenB
sbmn40: cjne    R2,#1,sbmn41
        mov     R3,Sm20             ;\
admn40: cjne    R3,#1,tdmn40        ; | cek status menu 21
        mov     DPTR,#tsmn40        ; | ada / tidak
tdmn40: cjne    R3,#2,sbmn41        ; |
        mov     DPTR,#mnkosg        ;/
sbmn41: cjne    R2,#2,sbmn42
        mov     R3,Sm21             ;\
admn41: cjne    R3,#1,tdmn41        ; | cek status menu 22
        mov     DPTR,#tsmn41        ; | ada / tidak
tdmn41: cjne    R3,#2,sbmn42        ; |
        mov     DPTR,#mnkosg        ;/
sbmn42: cjne    R2,#3,sbmn43
        mov     R3,Sm22             ;\
admn42: cjne    R3,#1,tdmn42        ; | cek status menu 23
        mov     DPTR,#tsmn42        ; | ada / tidak
tdmn42: cjne    R3,#2,sbmn43        ; |
        mov     DPTR,#mnkosg        ;/
sbmn43: cjne    R2,#4,sbmn44
        mov     R3,Sm23             ;\
admn43: cjne    R3,#1,tdmn43        ; | cek status menu 24
        mov     DPTR,#tsmn43        ; | ada / tidak
tdmn43: cjne    R3,#2,sbmn44        ; |
        mov     DPTR,#mnkosg        ;/
sbmn44: cjne    R2,#5,sbmn45
        mov     R3,Sm24             ;\
admn44: cjne    R3,#1,tdmn44        ; | cek status menu 25
        mov     DPTR,#tsmn44        ; | ada / tidak
tdmn44: cjne    R3,#2,sbmn45        ; |
        mov     DPTR,#mnkosg        ;/
sbmn45: ljmp    sbmenB
;
sbmenB: lcall   line2
        mov     Hurf,#16
        lcall   tulis
        lcall   tg_lps
sbmenC: lcall   scnkpd
        cjne    R0,#15,sbmenE
        dec     R2
        cjne    R2,#0,sbmenD
        mov     R2,#1
sbmenD: ljmp    sbmen6
sbmenE: cjne    R0,#16,sbmenG
        inc     R2
        cjne    R2,#6,sbmenF
        mov     R2,#5
sbmenF: ljmp    sbmen6
sbmenG: cjne    R0,#11,sbmenH
        ljmp    mamenu
sbmenH: cjne    R0,#12,sbmenC
        ljmp    pesan
;
pesan:  mov     DPTR,#maspsn
```

```
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        lcall   line2
        mov     Hurf,#16
        lcall   tulis
;
        mov     DPTR,#tangka
        lcall   line2
        mov     A,#10
        lcall   wr_chr
        mov     A,#10
        lcall   wr_chr
        lcall   tg_tkn
        mov     A,R0
        mov     R4,A                    ; jumlah pesanan
        lcall   wr_chr
        cjne    R4,#0,pesan0            ; cek 0
        ljmp    pesan                   ; ya -> ulang
;
pesan0: mov     DPTR,#dtabnr
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
;
pesan1: lcall   scnkpd
        cjne    R0,#11,pesan2
        ljmp    pesan
pesan2: cjne    R0,#12,pesan1
        inc     R5
        mov     A,R5
        mov     B,#6                    ; max jumlah pesanan
        div     AB
        cjne    A,#0,pesan3
        mov     DPTR,#spnpsn
        lcall   line1
        mov     Hurf,#15
        lcall   tulis
        mov     DPTR,#tangka
        mov     A,R5
        lcall   wr_chr
        lcall   delay2
        ljmp    pesan4
;
pesan3: mov     DPTR,#qtotod
        lcall   line1
        mov     Hurf,#16
        lcall   tulis
        lcall   delay2
        ljmp    mamenu
;
pesan4: cjne    R5,#1,pesan5
        mov     Ps0m,R1
        mov     Ps0s,R2
        mov     Ps0j,R4
pesan5: cjne    R5,#2,pesan6
        mov     Ps1m,R1
```

```
        mov     Ps1s,R2
        mov     Ps1j,R4
pesan6: cjne    R5,#3,pesan7
        mov     Ps2m,R1
        mov     Ps2s,R2
        mov     Ps2j,R4
pesan7: cjne    R5,#4,pesan8
        mov     Ps3m,R1
        mov     Ps3s,R2
        mov     Ps3j,R4
pesan8: cjne    R5,#5,pesan9
        mov     Ps4m,R1
        mov     Ps4s,R2
        mov     Ps4j,R4
pesan9: ljmp    mamenu
;
krmpsn: mov     R7,#0FFh            ;\
        cjne    R5,#0,krpsn1        ; | cek ada quota
        ljmp    krpsn2              ; | ya -> kirim data
krpsn1: lcall   kr_dta              ; | tidak -> return
        lcall   rstpsn              ; | reset menu pesan & jumlah quota
        mov     R5,#0               ; | return
krpsn2: ret                     ;/
;
kr_dta: mov     A,Ps0m
        lcall   kr_srl
        mov     A,Ps0s
        lcall   kr_srl
        mov     A,Ps0j
        lcall   kr_srl
;
        mov     A,Ps1m
        lcall   kr_srl
        mov     A,Ps1s
        lcall   kr_srl
        mov     A,Ps1j
        lcall   kr_srl
;
        mov     A,Ps2m
        lcall   kr_srl
        mov     A,Ps2s
        lcall   kr_srl
        mov     A,Ps2j
        lcall   kr_srl
;
        mov     A,Ps3m
        lcall   kr_srl
        mov     A,Ps3s
        lcall   kr_srl
        mov     A,Ps3j
        lcall   kr_srl
;
        mov     A,Ps4m
        lcall   kr_srl
        mov     A,Ps4s
        lcall   kr_srl
        mov     A,Ps4j
```

```
        lcall   kr_srl
        ret
;
stsmnu: mov     Sm00,#1             ; reset status menu
        mov     Sm01,#1
        mov     Sm02,#1
        mov     Sm03,#1
        mov     Sm04,#1
        mov     Sm05,#1
        mov     Sm06,#1
        mov     Sm07,#1
        mov     Sm08,#1
        mov     Sm09,#1
        mov     Sm10,#1
        mov     Sm11,#1
        mov     Sm12,#1
        mov     Sm13,#1
        mov     Sm14,#1
        mov     Sm15,#1
        mov     Sm16,#1
        mov     Sm17,#1
        mov     Sm18,#1
        mov     Sm19,#1
        mov     Sm20,#1
        mov     Sm21,#1
        mov     Sm22,#1
        mov     Sm23,#1
        mov     Sm24,#1
        ret
;
rstpsn: mov     Ps0m,#00
        mov     Ps0s,#00
        mov     Ps0j,#00
        mov     Ps1m,#00
        mov     Ps1s,#00
        mov     Ps1j,#00
        mov     Ps2m,#00
        mov     Ps2s,#00
        mov     Ps2j,#00
        mov     Ps3m,#00
        mov     Ps3s,#00
        mov     Ps3j,#00
        mov     Ps4m,#00
        mov     Ps4s,#00
        mov     Ps4j,#00
        ret
;
mnupdt: mov     R7,#0FFh            ; update menu
        lcall   respon
        lcall   tgudta
        mov     Sm00,R7
        lcall   respon
        lcall   tgudta
        mov     Sm01,R7
        lcall   respon
        lcall   tgudta
        mov     Sm02,R7
```

```
lcall   respon
lcall   tgudta
mov     Sm03,R7
lcall   respon
lcall   tgudta
mov     Sm04,R7
lcall   respon
lcall   tgudta
mov     Sm05,R7
lcall   respon
lcall   tgudta
mov     Sm06,R7
lcall   respon
lcall   tgudta
mov     Sm07,R7
lcall   respon
lcall   tgudta
mov     Sm08,R7
lcall   respon
lcall   tgudta
mov     Sm09,R7
lcall   respon
lcall   tgudta
mov     Sm10,R7
lcall   respon
lcall   tgudta
mov     Sm11,R7
lcall   respon
lcall   tgudta
mov     Sm12,R7
lcall   respon
lcall   tgudta
mov     Sm13,R7
lcall   respon
lcall   tgudta
mov     Sm14,R7
lcall   respon
lcall   tgudta
mov     Sm15,R7
lcall   respon
lcall   tgudta
mov     Sm16,R7
lcall   respon
lcall   tgudta
mov     Sm17,R7
lcall   respon
lcall   tgudta
mov     Sm18,R7
lcall   respon
lcall   tgudta
mov     Sm19,R7
lcall   respon
lcall   tgudta
mov     Sm20,R7
lcall   respon
lcall   tgudta
mov     Sm21,R7
```

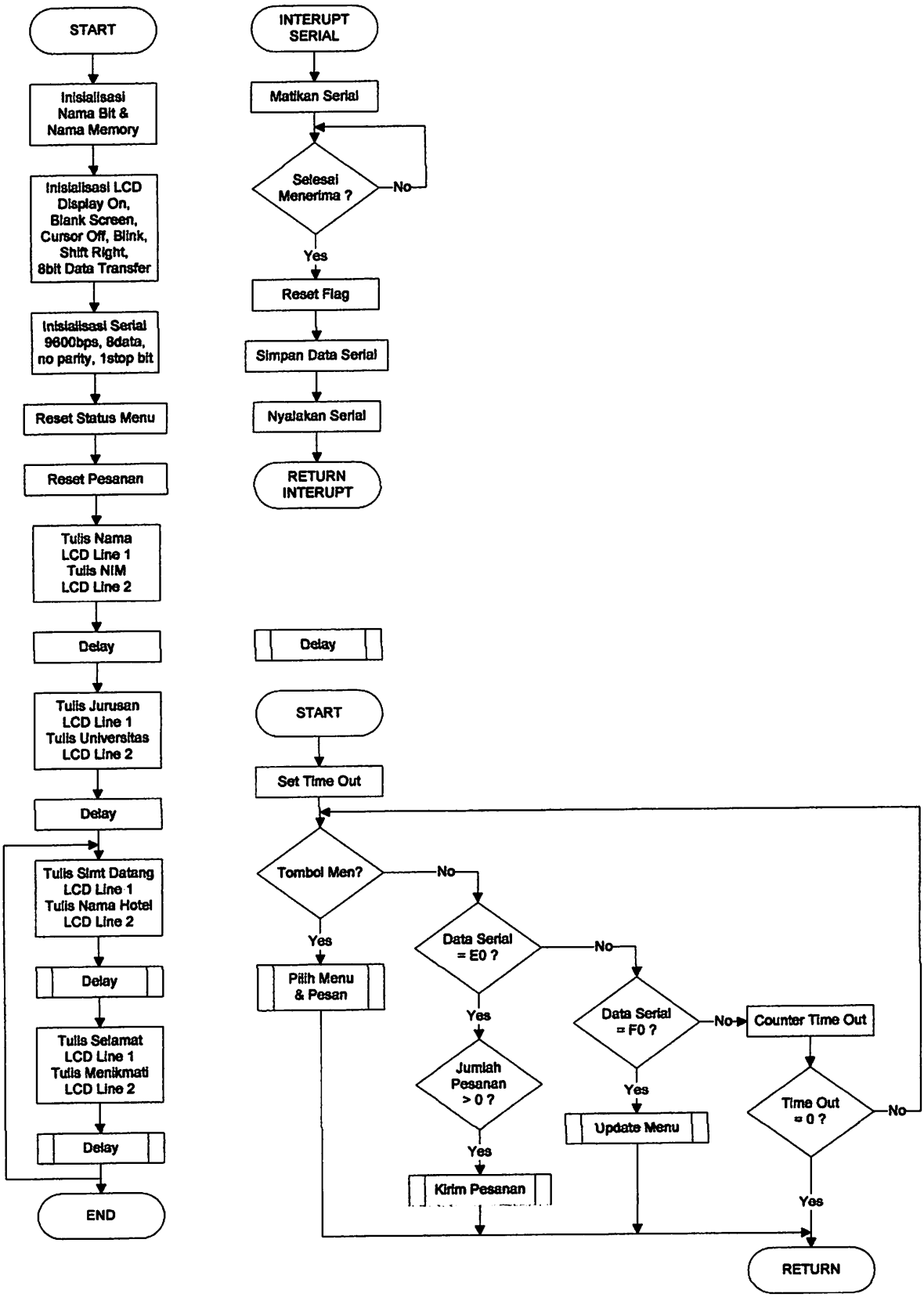
```
        lcall   respon
        lcall   tgudta
        mov     Sm22,R7
        lcall   respon
        lcall   tgudta
        mov     Sm23,R7
        lcall   respon
        lcall   tgudta
        mov     Sm24,R7
        lcall   respon
        ret
;
srl_in: lcall   delay1
        mov     TMOD,#20h
        mov     TH1,#0F4h
        mov     SCON,#50h
        setb    ES
        setb    EA
        setb    TR1
        clr     Slct                    ; RS485 receive
        ret
;
kr_srl: setb    Slct                    ; RS485 transmit
        lcall   delay0
        clr     ES
        mov     SBUF,A
        jnb     TI,$
        clr     TI
        setb    ES
        clr     Slct                    ; RS485 receive
        ret
;
tgudta: mov     R7,#0FFh
tdkada: cjne    R7,#0FFh,adadta
        ljmp    tdkada
adadta: ret
;
respon: lcall   delay0
        mov     A,#0FFh
        lcall   kr_srl
        ret
;
scnkpd: mov     R0,#10
        lcall   delay0
col1:   mov     P2,#11111110b
        mov     A,P2
c1b1:   cjne    A,#11101110b,c1b2
        mov     R0,#1
c1b2:   cjne    A,#11011110b,c1b3
        mov     R0,#2
c1b3:   cjne    A,#10111110b,c1b4
        mov     R0,#3
c1b4:   cjne    A,#01111110b,col2
        mov     R0,#13
;
col2:   mov     P2,#11111101b
        mov     A,P2
```

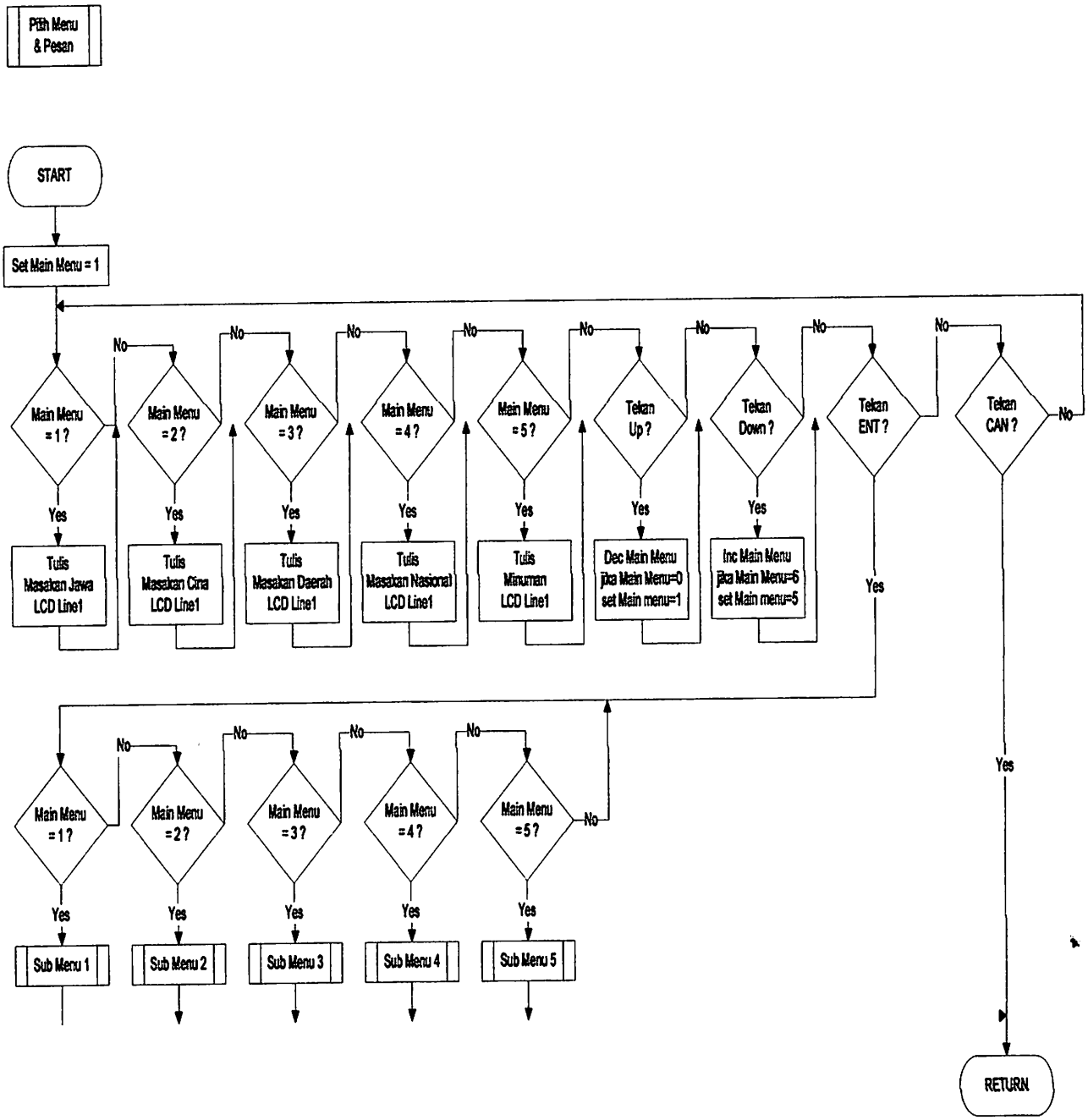
```
c2b1:   cjne    A,#11101101b,c2b2
        mov     R0,#4
c2b2:   cjne    A,#11011101b,c2b3
        mov     R0,#5
c2b3:   cjne    A,#10111101b,c2b4
        mov     R0,#6
c2b4:   cjne    A,#01111101b,col3
        mov     R0,#14
;
col3:   mov     P2,#11111011b
        mov     A,P2
c3b1:   cjne    A,#11101011b,c3b2
        mov     R0,#7
c3b2:   cjne    A,#11011011b,c3b3
        mov     R0,#8
c3b3:   cjne    A,#10111011b,c3b4
        mov     R0,#9
c3b4:   cjne    A,#01111011b,col4
        mov     R0,#15
;
col4:   mov     P2,#11110111b
        mov     A,P2
c4b1:   cjne    A,#11100111b,c4b2
        mov     R0,#11
c4b2:   cjne    A,#11010111b,c4b3
        mov     R0,#0
c4b3:   cjne    A,#10110111b,c4b4
        mov     R0,#12
c4b4:   cjne    A,#01110111b,back
        mov     R0,#16
back:   ret
;
tg_tkn: lcall   scnkpd
tg_tk0: cjne    R0,#16,tg_tk1
        ljmp    tg_tkn
tg_tk1: cjne    R0,#15,tg_tk2
        ljmp    tg_tkn
tg_tk2: cjne    R0,#14,tg_tk3
        ljmp    tg_tkn
tg_tk3: cjne    R0,#13,tg_tk4
        ljmp    tg_tkn
tg_tk4: cjne    R0,#12,tg_tk5
        ljmp    tg_tkn
tg_tk5: cjne    R0,#11,tg_tk6
        ljmp    mamenu
tg_tk6: cjne    R0,#10,tg_tk7
        ljmp    tg_tkn
tg_tk7: ret
;
tg_lps: lcall   scnkpd
        cjne    R0,#10,tg_lps
        ret
;
line1:      mov   P0,#080h
        lcall   w_ins
        ret
;
```

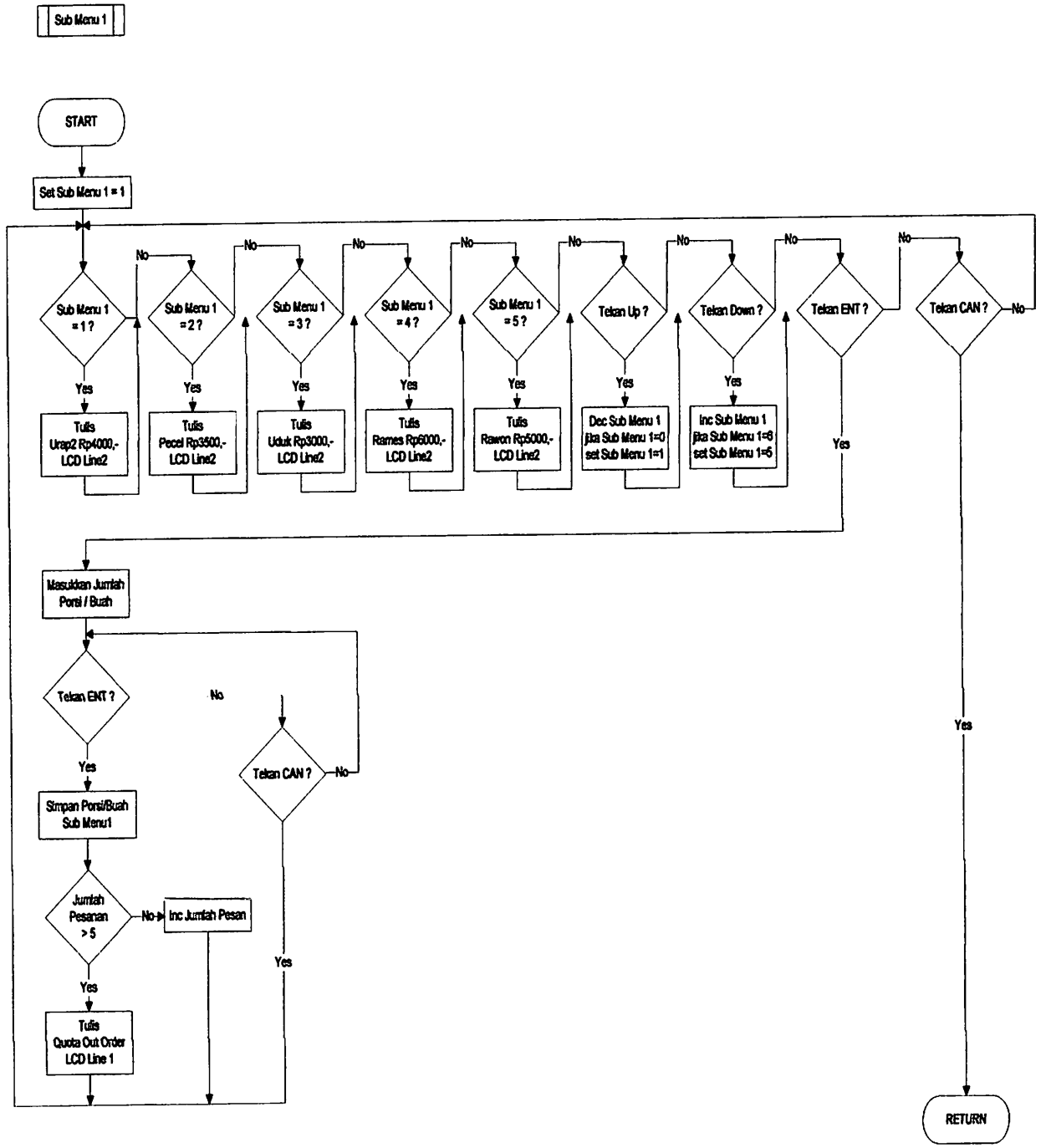
```
line2:      mov   P0,#0C0h
        lcall   w_ins
        ret
;
tulis:  clr A
      movc  A,@A+DPTR
      mov   P0,A
        lcall   w_chr
      inc   DPTR
      djnz  Hurf,tulis
        ret
;
wr_chr: movc    A,@A+DPTR
        mov     P0,A
        lcall   w_chr
        ret
;
w_ins:  clr     Enbl
        clr     Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret
;
w_chr:  clr     Enbl
        setb    Rest
        setb    Enbl
        clr     Enbl
        lcall   delay0
        ret
;
lcd_in: acall   delay1
        mov     P0,#01h             ; Display Clear
        acall   w_ins
        mov     P0,#38h             ; Function Set
        acall   w_ins
        mov     P0,#0Dh             ; Display On, Cursor, Blink
        acall   w_ins
        mov     P0,#06h             ; Entry Mode
        acall   w_ins
        mov     P0,#02h             ; Cursor Home
        acall   w_ins
        ret
;
lcdclr: mov     P0,#01h             ; Display Clear
        acall   w_ins
        acall   delay0
        acall   delay0
        ret
;
delay0: djnz    Dly0,delay0
        ret
;
delay1: lcall   delay0
        djnz    Dly1,delay1
        ret
;
```

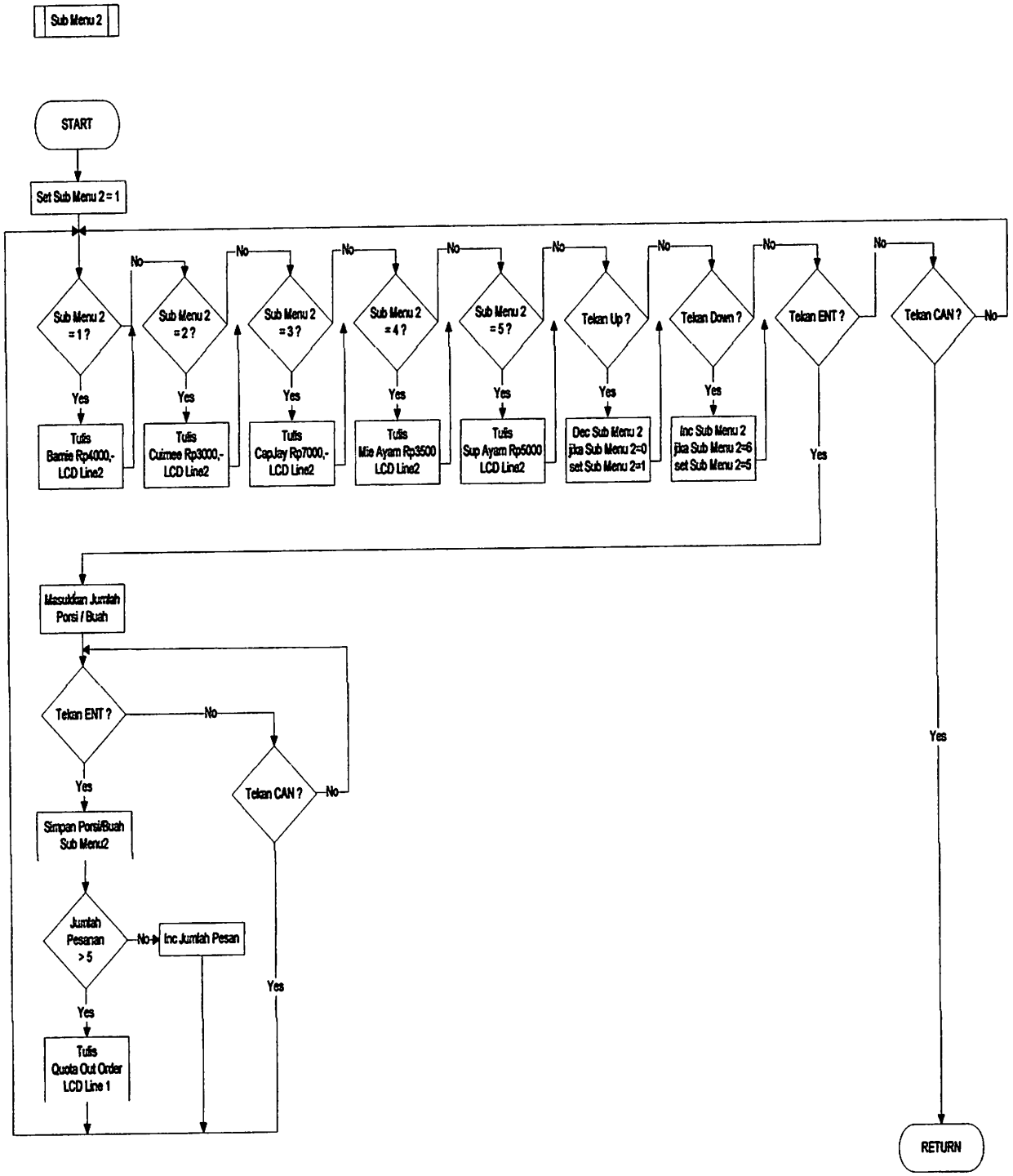
```
delay2: mov     Dly2,#20
dely20: lcall   delay1
        djnz    Dly2,dely20
        ret
;
delay3: mov     Dly2,#20
dely30: acall   delay0
        cjne    R7,#0E0h,dely31          ; cek request E0,E1,E2,E3,...
        ljmp    krmpsn             ; ya -> cek pesanan, ada -> kirim
dely31: cjne    R7,#0F0h,dely32          ; cek request F0,F1,F2,F3,...
        ljmp    mnupdt             ; ya -> update menu
dely32: lcall   scnkpd
        cjne    R0,#14,dely33
        mov     R1,#1
        ljmp    mamenu
dely33: djnz    Dly1,dely30
        djnz    Dly2,dely30
        ret
;
nama:   DB      '   Sepvian  Y   '
nim:    DB      ' Nim: 03.17.019 '
jurs:   DB      ' Teknik Elektro '
univ:   DB      '   ITN Malang   '
judul1: DB      ' Selamat Datang '
judul2: DB      '   RM Digital   '
judul3: DB      '    Selamat     '
judul4: DB      '   Menikmati    '
dtabnr: DB      '  Data Benar ?  '
maspsn: DB      'Masukkan Pesanan'
prsbua: DB      '    Porsi/Buah  '
spnpsn: DB      'Simpan Pesanan  '
qtotod: DB      'Quota Out Order!'
tangka: DB      '0123456789      '
mnkosg: DB      ' ---        --- '
tmamnu: DB      ' >>   Menu   << '
tmamn0: DB      '  Masakan Jawa  '
tmamn1: DB      '  Masakan Cina  '
tmamn2: DB      ' Masakan Daerah '
tmamn3: DB      'Masakan Nasional'
tmamn4: DB      '    Minuman     '
tsmn00: DB      'Urap2    Rp4.000'
tsmn01: DB      'Pecel    Rp3.500'
tsmn02: DB      'Uduk     Rp3.000'
tsmn03: DB      'Rames    Rp6.000'
tsmn04: DB      'Rawon    Rp5.000'
tsmn10: DB      'Bamie    Rp4.000'
tsmn11: DB      'Cuimee   Rp3.000'
tsmn12: DB      'CapJay   Rp7.000'
tsmn13: DB      'Mie Ayam Rp3.500'
tsmn14: DB      'Sup Ayam Rp5.000'
tsmn20: DB      'Soto     Rp4.500'
tsmn21: DB      'Rawon    Rp4.500'
tsmn22: DB      'Pepes    Rp3.000'
tsmn23: DB      'Gado2    Rp3.500'
tsmn24: DB      'Oseng2   Rp2.500'
tsmn30: DB      'Gule     Rp5.000'
tsmn31: DB      'Rendang  Rp5.000'
```

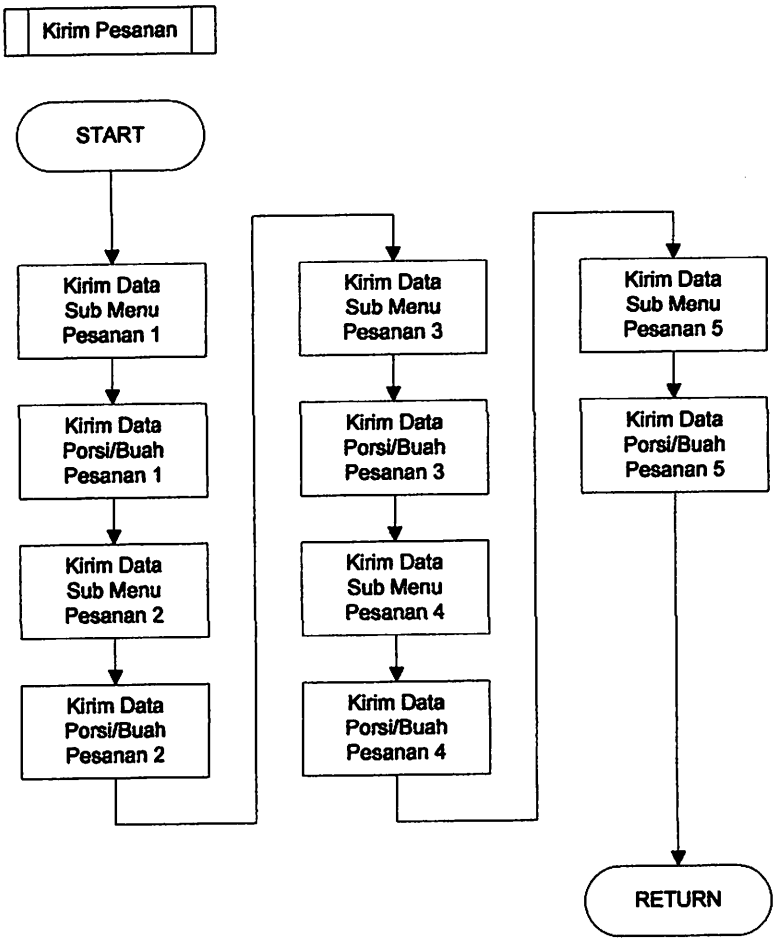
```
tsmn32: DB      'Opor     Rp6.000'
tsmn33: DB      'Kare     Rp4.000'
tsmn34: DB      'Sate     Rp7.000'
tsmn40: DB      'Es Teh   Rp1.500'
tsmn41: DB      'Es Teler Rp3.000'
tsmn42: DB      'Es Moka  Rp2.500'
tsmn43: DB      'Es Jeruk Rp2.000'
tsmn44: DB      'Es Buah  Rp3.500'
;
        end
```

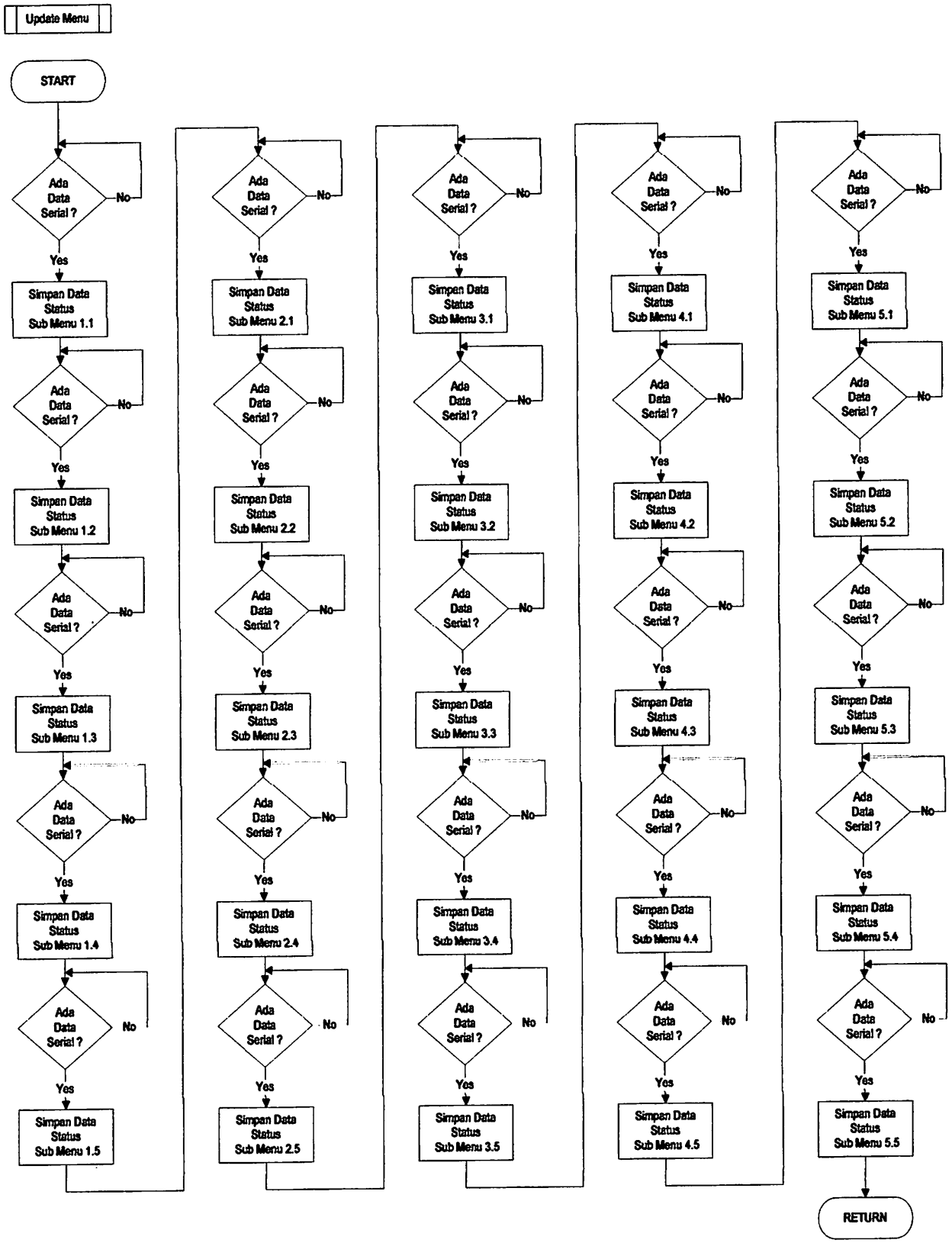

START
Inisialisasi Nama Bit & Nama Memory
Inisialisasi LCD Display On, Blank Screen, Cursor Off, Blink, Shift Right, 8bit Data Transfer
Inisialisasi Serial 9600bps, 8data, no parity, 1stop bit
Reset Status Menu
Reset Pesanan
Tulis Nama LCD Line 1 Tulis NIM LCD Line 2
Delay
Tulis Jurusan LCD Line 1 Tulis Universitas LCD Line 2
Delay
Tulis Simt Datang LCD Line 1 Tulis Nama Hotel LCD Line 2
Delay
Tulis Selamat LCD Line 1 Tulis Menikmati LCD Line 2
Delay
END
INTERUPT SERIAL
Matikan Serial
Selesai Menerima ?
No
Yes
Reset Flag
Simpan Data Serial
Nyalakan Serial
RETURN INTERUPT
Delay
START
Set Time Out
Tombol Men?
No
Yes
Pilih Menu & Pesan
Data Serial = E0 ?
No
Yes
Jumlah Pesanan > 0 ?
Yes
Kirim Pesanan
Data Serial = F0 ?
No
Yes
Update Menu
Counter Time Out
Time Out = 0 ?
No
Yes
RETURN


Pilih Menu & Pesan
START
Set Main Menu = 1
Main Menu =1?
Main Menu =2?
Main Menu =3?
Main Menu =4?
Main Menu =5?
Tekan Up?
Tekan Down?
Tekan ENT?
Tekan CAN?
No
Yes
Tulis Masakan Jawa LCD Line1
Tulis Masakan Cina LCD Line1
Tulis Masakan Daerah LCD Line1
Tulis Masakan Nasional LCD Line1
Tulis Minuman LCD Line1
Dec Main Menu jika Main Menu=0 set Main menu=1
Inc Main Menu jika Main Menu=6 set Main menu=5
Main Menu =1?
Main Menu =2?
Main Menu =3?
Main Menu =4?
Main Menu =5?
Sub Menu 1
Sub Menu 2
Sub Menu 3
Sub Menu 4
Sub Menu 5
RETURN


Sub Menu 1
START
Set Sub Menu 1 = 1
Sub Menu 1 =1 ?
Yes
No
Tulis Urap2 Rp4000,- LCD Line2
Sub Menu 1 =2 ?
Yes
No
Tulis Pecel Rp3500,- LCD Line2
Sub Menu 1 =3 ?
Yes
No
Tulis Uduk Rp3000,- LCD Line2
Sub Menu 1 =4 ?
Yes
No
Tulis Rames Rp6000,- LCD Line2
Sub Menu 1 =5 ?
Yes
No
Tulis Rawon Rp5000,- LCD Line2
Tekan Up ?
Yes
No
Dec Sub Menu 1 jika Sub Menu 1=0 set Sub Menu 1=1
Tekan Down ?
Yes
No
Inc Sub Menu 1 jika Sub Menu 1=6 set Sub Menu 1=5
Tekan ENT ?
Yes
No
Tekan CAN ?
No
Yes
Masukkan Jumlah Porsi / Buah
Tekan ENT ?
Yes
No
Tekan CAN ?
No
Yes
Simpan Porsi/Buah Sub Menu1
Jumlah Pesanan > 5
No
Inc Jumlah Pesan
Yes
Tulis Quota Out Order LCD Line 1
RETURN


Sub Menu 2
START
Set Sub Menu 2 = 1
Sub Menu 2 = 1 ?
Yes
No
Tulis Bamie Rp4000,- LCD Line2
Sub Menu 2 = 2 ?
Yes
No
Tulis Cuimee Rp3000,- LCD Line2
Sub Menu 2 = 3 ?
Yes
No
Tulis Capjay Rp7000,- LCD Line2
Sub Menu 2 = 4 ?
Yes
No
Tulis Mie Ayam Rp3500 LCD Line2
Sub Menu 2 = 5 ?
Yes
No
Tulis Sup Ayam Rp5000 LCD Line2
Tekan Up ?
Yes
No
Dec Sub Menu 2 jika Sub Menu 2=0 set Sub Menu 2=1
Tekan Down ?
Yes
No
Inc Sub Menu 2 jika Sub Menu 2=6 set Sub Menu 2=5
Tekan ENT ?
Yes
No
Tekan CAN ?
No
Yes
Masukkan Jumlah Porsi / Buah
Tekan ENT ?
Yes
No
Simpan Porsi/Buah Sub Menu2
Jumlah Pesanan > 5
No
Inc Jumlah Pesan
Yes
Tulis Quota Out Order LCD Line 1
Tekan CAN ?
No
Yes
RETURN


Kirim Pesanan
START
Kirim Data Sub Menu Pesanan 1
Kirim Data Porsi/Buah Pesanan 1
Kirim Data Sub Menu Pesanan 2
Kirim Data Porsi/Buah Pesanan 2
Kirim Data Sub Menu Pesanan 3
Kirim Data Porsi/Buah Pesanan 3
Kirim Data Sub Menu Pesanan 4
Kirim Data Porsi/Buah Pesanan 4
Kirim Data Sub Menu Pesanan 5
Kirim Data Porsi/Buah Pesanan 5
RETURN


Update Menu
START
Ada Data Serial ?
No
Yes
Simpan Data Status Sub Menu 1.1
Simpan Data Status Sub Menu 1.2
Simpan Data Status Sub Menu 1.3
Simpan Data Status Sub Menu 1.4
Simpan Data Status Sub Menu 1.5
Simpan Data Status Sub Menu 2.1
Simpan Data Status Sub Menu 2.2
Simpan Data Status Sub Menu 2.3
Simpan Data Status Sub Menu 2.4
Simpan Data Status Sub Menu 2.5
Simpan Data Status Sub Menu 3.1
Simpan Data Status Sub Menu 3.2
Simpan Data Status Sub Menu 3.3
Simpan Data Status Sub Menu 3.4
Simpan Data Status Sub Menu 3.5
Simpan Data Status Sub Menu 4.1
Simpan Data Status Sub Menu 4.2
Simpan Data Status Sub Menu 4.3
Simpan Data Status Sub Menu 4.4
Simpan Data Status Sub Menu 4.5
Simpan Data Status Sub Menu 5.1
Simpan Data Status Sub Menu 5.2
Simpan Data Status Sub Menu 5.3
Simpan Data Status Sub Menu 5.4
Simpan Data Status Sub Menu 5.5
RETURN