SKRIPSI

PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA

Disusun Oleh :

Mukhlish Andi Kristiyawan
02.17.043

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

2008

LEMBAR PERSETUJUAN

PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA

SKRIPSI

Disusun dan diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik Elektronika Strata Satu (S-1)

Disusun Oleh :

Mukhlish Andi Kristiyawan
02.17.043

Diperiksa dan disetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Yusuf Ismail Nakhoda, MT.	**Sotyohadi, ST**
NIP.Y. 1018800189	

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
2008

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

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama Mahasiswa : Mukhlish Andi Kristiyawan
NIM : 02.17.043
Jurusan : Teknik Elektro S-1
Konsentrasi : Teknik Elektronika
Judul Skripsi : "PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA"

Dipertahankan dihadapan Tim Penguji Skripsi Jenjang Strata Satu (S-1) pada:
Hari : Sabtu
Tanggal : 15 Maret 2008
Dengan Nilai : 80,5 (A)

Panitia Ujian Skripsi

Ketua

(Ir. Mochtar Asroni, MSME)
NIP.Y. 1018100036

Sekretaris

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

Anggota Penguji

Penguji Pertama

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

Penguji Kedua

(Ir. Eko Nurcahyo)
NIP. Y. 1028700172

ABSTRAKSI

PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA

(Mukhlish Andi Kristiyawan, 02.17.043, Teknik Elektro S.1/Elektronika)
(Dosen Pembimbing: (1) Ir. Yusuf Ismail Nakhoda, MT. dan (2) Sotyohadi, ST.)

Kata Kunci : Mikrokontroler, *Keypad*, LCD, ISD

Umumnya sudah banyak kalkulator yang beredar di masyarakat yang hanya menggunakan tampilan LCD saja. Sehingga bagi tuna netra tidak mungkin bisa menggunakan kalkulator yang beredar saat ini. Sehingga dibuat kalkulator yang dilengkapi dengan keypad braille dan dapat memberikan *output* berupa suara.

Dalam perancangan dan pembuatan alat ini menggunakan mikrokontroller AT89S8253 sebagai pemroses data dan ISD 2590 sebagai penyimpan suara. Setiap ada *input* dari *keypad*, maka mikrokontroler akan menyimpan sebagai data *input* dan langsung mengirim data ini ke LCD untuk ditampilkan, dan memgaktifkan ISD untuk memberikan *output* suara sesuai dengan data *input*, demikian juga dengan hasil dari operasi aritmatika.

Pada alat ini hanya mampu memroses *input* bilangan bulat tak bertanda, tetapi sekalipun demikian alat ini mampu memberikan *output* bilangan pecahan dan bilangan bertanda. Disamping itu alat ini bisa digunakan oleh tuna netra dengan mudah karena telah dilengkapi dengan *keypad braille*.

KATA PENGANTAR

Puji dan syukur kehadirat Tuhan Yang Maha Esa yang telah memberikan rahmat dan hidayahNya, sehingga penyusun dapat menyelesaikan skripsi dengan judul: **"Perancangan Dan Pembuatan Kalkulator dengan *Output* LCD dan *Speaker* sebagai Alat Hitung Elektronik bagi Tuna Netra"**. Penyusunan skripsi merupakan syarat yang harus ditempuh mahasiswa jurusan Teknik Elektro S-1 Institut Teknologi Nasional Malang untuk memperoleh gelar Sarjana Teknik. Penulis mengucapkan terima kasih kepada:

1. Kedua orang tuaku yang telah memberikan do'a restunya.
2. Bapak Prof.Dr.Ir. Abraham Lomi, MSEE. selaku Rektor Institut Teknologi Nasional Malang.
3. Bapak Ir. Mochtar Asroni, MSME. selaku Dekan Fakultas Teknologi Industri di Institut Teknologi Nasional Malang.
4. Bapak Ir.F. Yudi Limpraptono, MT, selaku Kajur Teknik Elektro S1 dan juga sebagai penguji I.
5. Bapak Ir. Yusuf Ismail Nakhoda, MT. selaku dosen pembimbing I.
6. Bapak Sotyohadi, ST. selaku dosen pembimbing II.
7. Bapak Ir. Eko Nurcahyo, selaku Penguji II.
8. Rekan-rekan semua yang telah banyak membantu dan memberikan saran kepada saya dalam pembuatan alat ini.

Penulis mengharapkan adanya pengembangan dari alat ini, agar dapat digunakan sesuai dengan harapan yang diinginkan terutama bagi tuna netra.

Malang, April 2008

Penyusun

DAFTAR ISI

HALAMAN JUDUL i

LEMBAR PERSETUJUAN ii

BERITA ACARA UJIAN SKRIPSI iii

ABSTRAKSI iv

KATA PENGANTAR v

DAFTAR ISI vi

DAFTAR GAMBAR ix

DAFTAR TABEL xi

BAB I PENDAHULUAN

1.1. Latar Belakang 1

1.2. Tujuan 2

1.3. Rumusan Masalah 2

1.4. Batasan Masalah 2

1.5. Metodologi 3

1.6. Sistematika Penulisan 4

BAB II LANDASAN TEORI

2.1. Mikrokontroler AT89S8253 5

2.1.1. Pendahuluan 5

2.1.2. Arsitektur AT89S8253 6

2.1.3. SFR *(Special Functional Register)* 8

2.1.4. Proses Pengolahan Data dan Instruksi Aritmatika pada Mikrokontroler 8

2.2. LCD (*Liquid Crystal Display*) 15

2.3. IC *Information Storage Device* (ISD) 2590 18

2.4. *Keypad* 22

2.5. Huruf/Angka *Braille* 22

BAB III PERANCANGAN DAN PEMBUATAN ALAT

3.1. Perancangan Perangkat Keras (*Hardware*) 23

3.1.1. Sistem Mikrokontroler AT89S8253 24

3.1.1.1 Perancangan Rangkaian *Clock* 25

3.1.1.2 Rangkaian Reset 26

3.1.2. LCD M1632 27

3.1.3. ISD2590 30

3.1.4. *Keypad* 4 x 4 33

3.2. Perancangan Perangkat Lunak 35

3.2.1 Diagram Alir (*Flowchart*) 36

BAB IV PENGUJIAN DAN PERCOBAAN ALAT

4.1. Pengujian *Keypad* 38

4.2. Pengujian LCD M1632 40

4.3. Pengujian Rangkaian *Information Storage Device (ISD)* 2590 . 42

4.4. Pengujian Alat Keseluruhan 44

BAB V PENUTUP

5.1. Kesimpulan 50

5.2. Saran 51

DAFTAR PUSTAKA

LAMPIRAN

DAFTAR GAMBAR

Gambar 2.1 Blok Diagram AT89S8253 7

Gambar 2.2 Proses Perkalian pada Accumulator 11

Gambar 2.3 Contoh Proses Pembagian Bilangan Biner 12

Gambar 2.4 Proses Pengurangan yang Terjadi pada Sistem Pembagian Berekor 13

Gambar 2.5 Penjumlahan Hasil Bagi 14

Gambar 2.6 Modul LCD 2 ×16 karakter 16

Gambar 2.7 Mengirim/Mengambil Data Ke/Dari M1632 17

Gambar 2.8 Susunan kaki ISD2590 19

Gambar 2.9 Rangkaian *Keypad* 4x4 22

Gambar 2.10 Huruf *Braille* 22

Gambar 3.1 Diagram Blok Keseluruhan Sistem 23

Gambar 3.2 Perancangan Sistem Mikrokontroler AT89S8253 24

Gambar 3.3 Rangkaian *Clock* untuk MCU AT89S8253 25

Gambar 3.4 Rangkaian Reset 26

Gambar 3.5 Rangkaian LCD 28

Gambar 3.6. Proses Pengiriman Hasil Aritmatika ke LCD 30

Gambar 3.7 Rangkaian ISD2590 31

Gambar 3.8 Rangkaian *Keypad* 34

Gambar 3.9. *Flowchart* Proses Aritmatika 37

Gambar 4.1 Rangkaian *Keypad* 39

Gambar 4.2 Pengujian LCD 40

Gambar 4.3 Hasil Pengujian LCD 41

Gambar 4.4 Rangkaian Pengujian Tegangan LCD 41

Gambar 4.5 Rangkaian Pengujian ISD 2590 43

Gambar 4.6. Tampilan LCD pada saat saklar di ON-kan 45

Gambar 4.7. Menuliskan Angka yang Ingin Dijumlahkan 45

Gambar 4.8 Hasil Penjumlahan 46

Gambar 4.9 Tampilan Lebih dari 8 Digit 46

Gambar 4.10 Menuliskan Angka yang Akan Dikurangi 46

Gambar 4.11 Hasil Penjumlahan 47

Gambar 4.12 Hasil Pengurangan dengan Tampilan Tanda Minus 47

Gambar 4.13 Menuliskan Angka yang Akan Dikalikan 47

Gambar 4.14 Hasil Perkalian 48

Gambar 4.15 Menuliskan Angka yang Akan Dibagi 48

Gambar 4.16 Hasil Pembagian 48

Gambar 4.17 Hasil Pembagian Disertai Tanda Koma 49

DAFTAR TABEL

Tabel 2.1 Instruksi/Operator Aritmatika 8

Tabel 2.2 Pemilihan Register Pada LCD M16321 5

Tabel 2.3 Fungsi Pin – Pin LCD 16

Tabel 3.1 Data Hasil Perekaman dan Alamatnya Pada ISD 2590 33

Tabel 4.1 Pengujian data *Output* Baris dan Kolom untuk setiap tombol 39

Tabel 4.2 Pengujian Tegangan VCC Pada LCD 42

Tabel 4.3 Hasil Pengujian ISD 2590 dengan pemanggilan alamat 44

Tabel 4.4 Hasil Pengujian Pengucapan 49

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan teknologi yang sangat pesat dalam beberapa tahun terakhir ini mengakibatkan semakin cepat perubahan yang ada di dunia. Salah satu teknologi yang mengalami perkembangan pesat adalah elektronika. Dari tahun ke tahun akan selalu ditemukan alat untuk memudahkan atau mempercepat pekerjaan manusia.

Sebagai contoh dengan ditemukannya alat hitung elektronik atau yang biasa disebut dengan kalkulator. Kalkulator ini telah mempermudah manusia dalam melakukan perhitungan. Hanya dengan menekan tombol angka dan operasi aritmatika sederhana yang tersedia di *keypad* maka kita dapat melihat angka dan operasi aritmatika yang kita inginkan tampil di LCD, demikian juga dengan hasil dari operasi aritmatika tersebut.

Tetapi lain halnya dengan seorang yang buta/tuna netra, dia tidak bisa melihat angka dan operasi aritmatika yang tersedia di *keypad*, serta apa yang tampil di LCD.

Oleh karena itu pada tugas akhir ini dikembangkan sistem kalkulator yang tidak hanya saja dapat dilihat dengan tampilan LCD tetapi juga dilengkapi dengan *keypad braille* dan dapat di*output*kan melalui *speaker* dengan menggunakan fasilitas ISD. Pada sistem kalkulator ini akan memudahkan bagi tuna netra untuk melakukan penghitungan, karena hasil dari perhitungan akan di*output*kan melalui

speaker, serta setiap ada penekanan pada *keypad* dalam kalkulator ini juga dapat mengeluarkan suara sesuai dengan tombol yang ditekan.

Dengan demikian diharapkan bagi para tuna netra dapat melakukan penghitungan menggunakan kalkulator secara mudah karena telah dilengkapi fasilitas suara.

1.2. Tujuan

Penulisan skripsi ini bertujuan merancang dan membuat kalkulator yang dilengkapi *keypad braille* dan *speaker* yang dapat memberikan *output* berupa suara, sehingga memudahkan bagi tuna netra untuk menggunakannya.

1.3. Rumusan Masalah

Berdasarkan latar belakang yang telah dikemukakan sebelumnya, maka beberapa hal yang perlu dirumuskan dalam perancangan dan pembuatan kalkulator yang dilengkapi dengan *keypad braille* dan *output speaker* ini adalah:

1. Bagaimana membuat perangkat keras (*hardware*) kalkulator yang dilengkapi dengan *keypad braille* dengan *output speaker*.
2. Bagaimana membuat perangkat lunak (*software*) pada kalkulator yang dilengkapi dengan *keypad braille* dengan *output speaker*.

1.4. Batasan Masalah

Agar permasalahan yang ada dapat di jelaskan secara tepat dan terhindar dari pembahasan yang tidak sesuai dengan topik yang dibahas maka dianggap perlu adanya batasan masalah. Adapun batasan masalah pada tugas akhir ini antara lain:

1. Pengontrol utama menggunakan mikrokontroler AT89S8253.
2. *Output* pada alat hitung elektronik ini maksimal 8 digit.
3. Operasi aritmatika yang digunakan adalah penjumlahan (+), pengurangan (-), perkalian (x), dan pembagian (/) serta *equal* (=).
4. Bilangan yang diproses adalah bilangan bulat tak bertanda.
5. Tidak membahas *Power Supply*.

1.5. Metodologi

Untuk mencapai tujuan yang direncanakan, maka pada tugas akhir ini menggunakan metodologi sebagai berikut:

☐ Studi Literatur

Yaitu dengan melakukan studi kepustakaan untuk memperoleh teori serta gambaran tentang masalah yang akan dibahas serta mempelajari teori serta aplikasi sistem kontrol menggunakan mikrokontroler AT89S52.

☐ Perencanaan Dan Pembuatan Alat

Dalam pembuatan alat ini menggunakan konsep sebagai berikut :

- Perencanaan sistem secara keseluruhan (pembuatan diagram blok sistem).
- Mendeskripsikan fungsi dari masing-masing blok diagram.
- Membuat perangkat keras *(hardware)* dan perangkat lunaknya *(software)*.

1.6. Sistematika Penulisan

Adapun sistematika dari penyusunan laporan tugas akhir ini adalah sebagai berikut:

BAB I Berisi latar belakang, rumusan masalah, batasan masalah, tujuan, metodologi dan sistematika penyusunan laporan tugas akhir.

BAB II Membahas tentang dasar teori Mikrokontroler AT89S52, ISD, *Keypad*, LCD dan teori-teori lain yang menunjang dalam perancangan tugas akhir ini.

BAB III Membahas tentang perencanaan *hardware* dan *software* dari sistem yang akan dibuat.

BAB IV Membahas tentang pengujian *hardware* dan *software* dari sistem yang telah dibuat.

BAB V Penutup berisi kesimpulan, saran dan daftar pustaka.

BAB II

LANDASAN TEORI

Landasan teori sangat membantu untuk dapat memahami suatu sistem. Selain dari pada itu dapat juga dijadikan sebagai bahan acuan didalam merencanakan suatu sistem. Dengan pertimbangan hal-hal tersebut, maka landasan teori merupakan bagian yang harus dipahami untuk pembahasan selanjutnya.

2.1 Mikrokontroler AT89S8253

2.1.1 Pendahuluan

Mikrokontroler AT89S8253 membutuhkan daya yang rendah, memiliki *performance* yang tinggi dan merupakan mikrokomputer 8 *bit* yang dilengkapi 12K *byte flash* PEROM (*Programmable and eraseable Read Only Memory*) yaitu ROM yang dapat ditulis menggunakan programmer. Serta 2 Kbyte EEPROM (*Electrical Eraseable Programmable Read Only Memory*) internal.

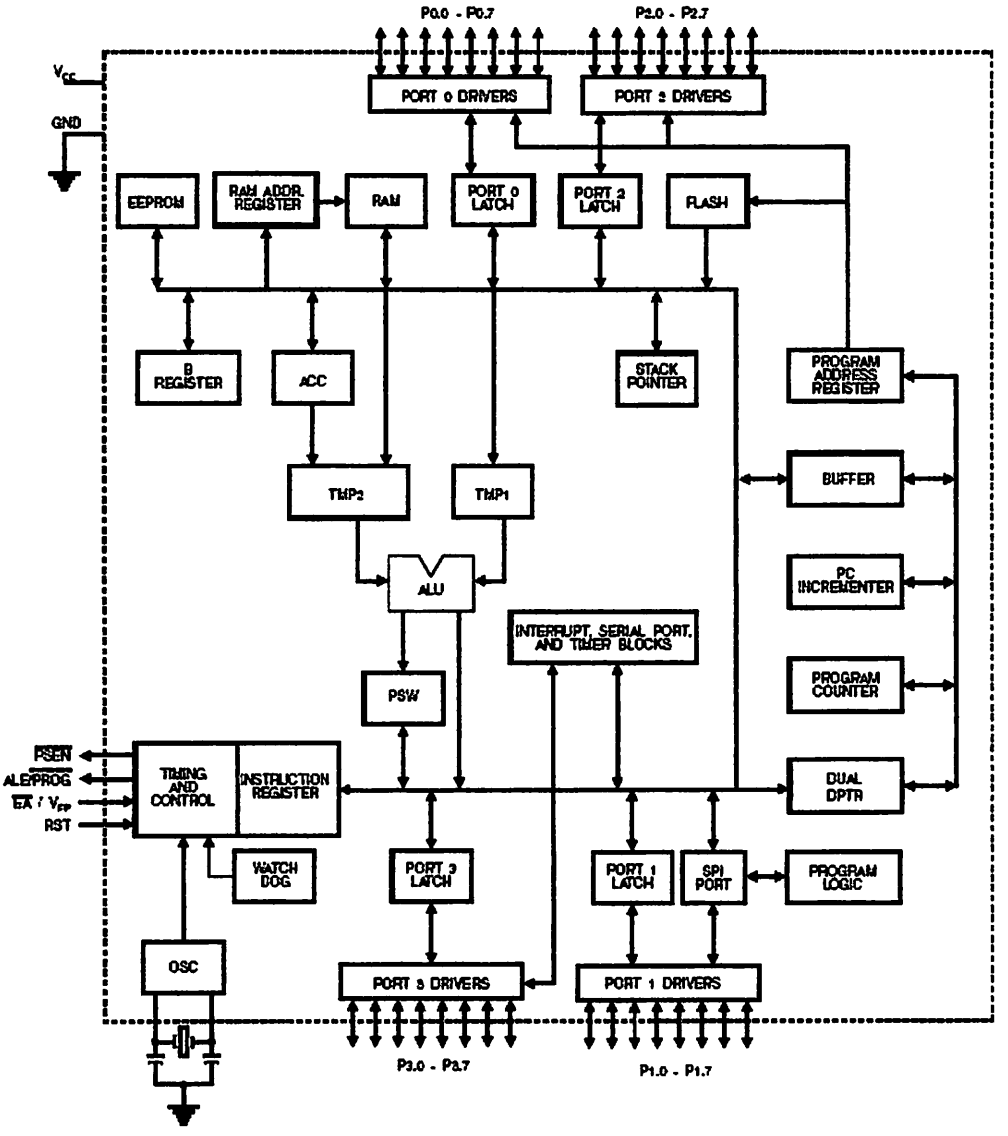
Flash PEROM dalam AT89S8253 menggunakan *Atmel 's High-Density Non Volatile Technology* yang mempunyai kemampuan untuk ditulis ulang hingga 1000 kali dan oleh karena mikrokontroler AT89S8253 merupakan mikrokontroler ATMEL yang kompatibel dengan mikrokontroler keluarga MCS-51, maka *flash* PEROM berisikan perintah *standard* MCS-51.

2.1.2 Arsitektur AT89S8253

Sebagai *single chip* yaitu suatu sistem mikroprosesor yang terintegrasi, mikrokontroler AT89S8253 mempunyai konfigurasi sebagai berikut:

1. CPU *(Central Processing Unit)* 8 bit dengan *register* A *(Accumulator)* dan *register* B.
2. 16-bit *Program Counter* (PC) dan *(Data Pointer)* DTPR.
3. 8-bit *Program Status Word (PSW).*
4. 8-bit *Stack Pointer (SP)*
5. 12 *kbyte* Flash PEROM *internal*
6. 256 *byte internal* RAM.
7. 32 pin *input-output* tersusun atas PO-P3. masing-masing 8-bit.
8. 3 buah *timer (TO & T1)* dengan masing-masing 16-bit *timer/ counter.*
9. *Receiver/transmiter data* secara *serial full duplex: Serial buffer* (SBUF).
10. *Control register : TCON,* TMOD, SCON, PCON, IP & IE.
11. 9 buah *sumber* interupsi (4 buah sumber interupsi *external dan 5* buah sumber interupsi *internal).*
12. *Oscillator* dan *clock internal.*

Arsitektur dasar dari mikrokontroler AT89S8253 seperti blok diagram berikut:



Gambar 2.1 Blok Diagram AT89S8253[1]

2.1.3 SFR *(Special Functional Register)*

Untuk operasi AT89S8253 yang menggunakan alamat internal RAM (00H-FFH), tetapi tidak semua alamat tersebut digunakan SFR_s. Berikut ini adalah contoh dari vektor alamat pada SFR_s.

- *Accumulator* (ACC) atau *register* A dan *register* B, kedua *register* ini digunakan untuk operasi aritmatika.
- *Program Status Word, register* ini meliputi: CY (*Carry*), AC (*Auxcilliary carry*), FO(*Flag*), RS0 dan RS1 (untuk pemilihan *register bank*), OV (*overflow*) dan *Parity* (*parity flag*).
- *Stack Pointer* merupakan *register* yang digunakan untuk menunjuk alamat, *register* ini digunakan bila terdapat suatu *routine* pada program utama.

2.1.4 Proses Pengolahan Data dan Instruksi Aritmatika pada Mikrokontroler

Instruksi-instruksi yang diberikan pada mikrokontroler sangatlah penting karena mikrokontroler dapat memroses suatu data sesuai dengan instruksi tersebut. Maka dari itu, disini akan dijelaskan instruksi aritmatika yang mendukung dengan menggunakan bahasa C beserta proses pengolahan datanya sesuai dengan instruksi yang diberikan.

Tabel 2.1 Instruksi/Operator Aritmatika[2]

No	Instruksi/Operator	Keterangan
1.	+	Operasi penjumlahan
2.	-	Operasi pengurangan
3.	*	Operasi perkalian
4.	/	Operasi pembagian
5.	%	Operasi sisa pembagian

Dengan menggunakan instruksi/operator aritmatika diatas, maka mikrokontroler akan dengan mudah melakukan operasi aritmatika. Sedangkan proses pengolahan data yang terdapat pada accumulator mikrokontroler dijelaskan sebagai berikut:

- Penjumlahan (+)

  Operasi ini akan menambahkan nilai variabel satu dengan nilai variabel lainnya (menambahkan nilai register A dengan register lainnya) sesuai dengan instruksi yang diberikan. Dan hasilnya disimpan pada register A.

  Sebagai contoh:

  bil1=8 atau 1000
  bil2=9 atau 1001
  Hasil=bil1+bil2

  Dengan adanya instruksi Hasil=bil1+bil2, maka *accumulator* akan memroses data tersebut dalam bentuk bilangan biner. Proses penjumlahan bilangan biner sama halnya dengan penjumlahan bilangan desimal. Dalam hal ini penjumlahan bilangan biner menggunakan 4 prinsip, yaitu:

  0 + 0 = 0

  0 + 1 = 1

  1 + 0 = 1

  1 + 1 = 10, ditulis 0 dan 1 ditambahkan pada bit sebelah kirinya

- Pengurangan (-)

  Operasi ini akan mengurangi nilai variabel satu dengan nilai variabel lainnya (mengurangi nilai register A dengan register lainnya) sesuai dengan instruksi yang diberikan. Dan hasilnya disimpan pada register A.

  Sebagai contoh:

  bil1=5 atau 0101
  bil2=3 atau 0011
  Hasil=bil1-bil2

  Dengan adanya instruksi Hasil=bil1+bil2, maka *accumulator* akan memroses data tersebut dalam bentuk bilangan biner. Proses pengurangan bilangan biner sama halnya dengan pengurangan bilangan desimal. Dalam hal ini pengurangan bilangan biner juga menggunakan 4 prinsip, yaitu:

  0 - 0 = 0

  1 - 0 = 1

  1 - 1 = 0

  0 - 1 = 1 → dengan meminjam '1' dari digit sebelahnya

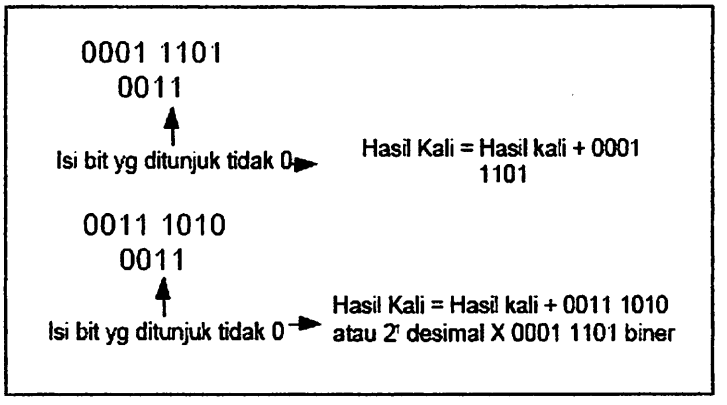
- Perkalian (*)

  Operasi ini akan melakukan perkalian antara nilai variabel satu dengan nilai variabel yang lain (melakukan perkalian antara nilai yang tersimpan di dalam register A dan register B). Dan hasil perkalian akan disimpan di register A.

Sebagai contoh:

bil1=29 atau 00011101
bil2=3 atau 0011
Hasil=bill*bil2

Dengan adanya instruksi Hasil=bil1*bil2, maka *accumulator* akan memroses data tersebut dalam bentuk bilangan biner. Proses perkalian yang dilakukan *accumulator* dijelaskan sebagai berikut:



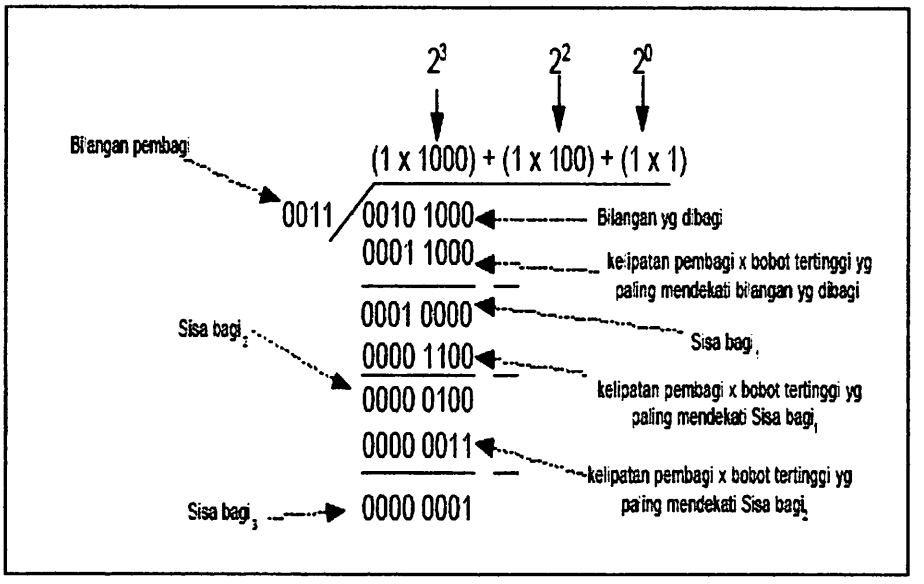
Gambar 2.2 Proses Perkalian pada Accumulator[4]

- Menggeser ke kanan bilangan pengali beserta *carry*.
- Setiap kali *carry flag* = 1, dimana ekivalen dengan pointer menunjuk ke bit yang bukan 0, maka bilangan yang dikali dijumlahkan dengan hasil kali.
- Menggeser ke kiri bilangan yang akan dikali baik terjadi atau tidak terjadi *carry*.
- Proses dilakukan sebanyak jumlah bit dari bilangan yang diproses.

- Pembagian (/)

Operasi ini akan melakukan pembagian antara nilai variabel satu dengan nilai variabel yang lain (melakukan pembagian antara nilai yang tersimpan di dalam register A dan register B). Dan hasil bagi akan

tersimpan pada register A dan sisa hasil bagi tersimpan pada register B. sebagai contoh dapat dilihat pada gambar 2.3 dibawah ini:



Gambar 2.3 Contoh Proses Pembagian Bilangan Biner[4]

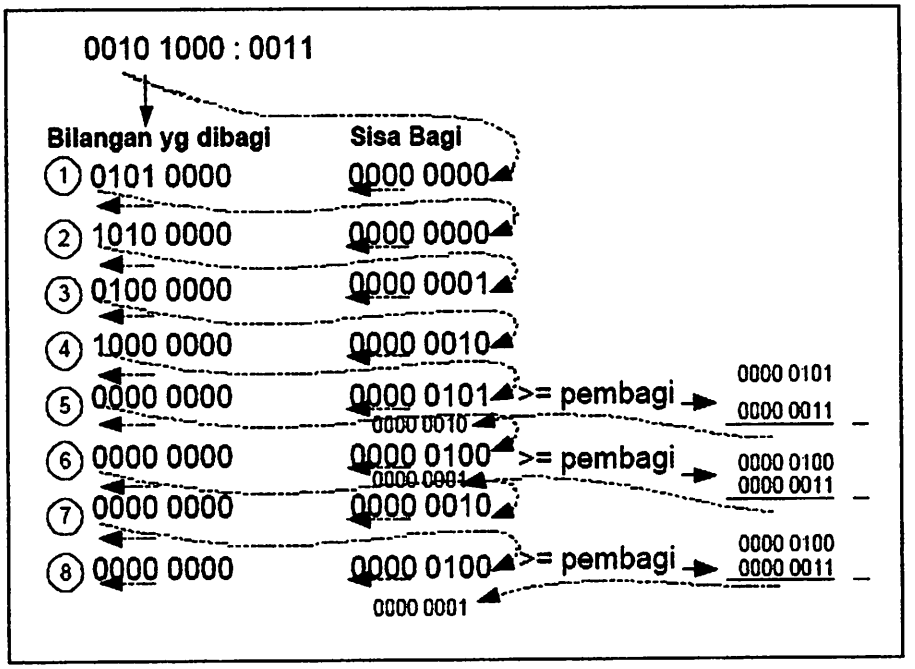
Pada proses pembagian di atas terdapat dua buah proses yaitu:

- Pengurangan terhadap kelipatan pembagi berbobot tertinggi.
- Penjumlahan hasil bagi dengan hasil bagi sebelumnya setiap kali terjadi pengurangan dengan terhadap kelipatan pembagi berbobot tertinggi terjadi.

Untuk melakukan proses pengurangan antara bilangan yang dibagi dengan kelipatan pembagi berbobot tertinggi pada mikrokontroler, maka hal ini dapat dilakukan dengan:

- Menggeser ke kiri bilangan yang dibagi ke suatu memori tertentu yang selanjutnya digunakan untuk menyimpan sisa bagi sebanyak 8 kali bila bilangan yang diproses hanya menempati area 8 bit.

- Mengurangi isi memori sisa bagi dengan bilangan pembagi saat ditemukan isi memori sisa bagi >= bilangan pembagi
- Isi memori sisa bagi adalah merupakan pengurangan dari bilangan yang dibagi dengan kelipatan pembagi berbobot tertinggi
- Contoh:

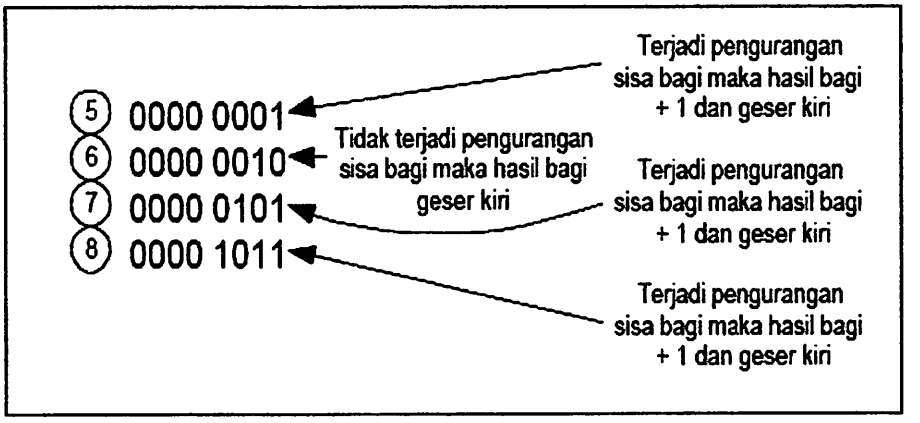


Gambar 2.4 Proses Pengurangan yang Terjadi pada Sistem Pembagian Berekor[4]

Penjumlahan hasil bagi dengan hasil bagi sebelumnya setiap kali terjadi pengurangan dengan terhadap kelipatan pembagi berbobot tertinggi terjadi. Dalam mikrokontroler, hal ini dapat dilakukan dengan:

- Menggeser ke kiri isi memori hasil bagi sebanyak 8 x bila bilangan yang diproses hanya menempati area 8 bit.
- Menambah satu (increment) memori hasil bagi setiap kali ditemukan isi memori sisa bagi >= bilangan pembagi.

Seperti yang terlihat pada gambar 2.4, proses pengurangan sisa bagi (karena memori sisa bagi >= bilangan pembagi) terjadi pada step 5, step 6 dan step 8, maka hanya pada step-step tersebut saja terjadi increment dan pergeseran ke kiri sedangkan pada step 7 hanya dilakukan pergeseran ke kiri seperti pada gambar 2.5.



Gambar 2.5 Penjumlahan Hasil Bagi[4]

Pada step 1 hingga 4 sebetulnya juga terdapat proses pergeseran ke kiri, namun pada saat itu isi memori hasil bagi hanyalah 0000 0000 saja sehingga proses pergeseran tidak mengubah isi memori tersebut.

Pada step 5 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 3x hingga akhir step, hal ini ekivalen dengan penambahan 1 x 1000 atau 1 x 23.

Pada step 6 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 2x hingga akhir step, hal ini ekivalen dengan penambahan 1 x 100 atau 1 x 22.

Pada step 5 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 0x atau tidak bergeser hingga akhir step, hal ini ekivalen dengan penambahan 1 x 1 atau 1 x 20.

Maka total penjumlahan hasil bagi adalah 1 x 1000 + 1 x 100 + 1 x 1 = 1101 biner

2.2 LCD (*Liquid Crystal Display*)

LCD (*Liquid Crystal Display*) adalah suatu jenis piranti output yang menggunakan daya rendah dengan pengontrol kontras dan kecerahan. Pengontrol utamanya dan karakter ada pada ROM (*Read Only Memory*) generator dan *display* data RAM (*Random Access Memory*) yang akan menghasilkan *extended key codes* (kode tombol/*keyboard* standart internasional dalam *Hexsa*) jika padanya diberikan sebuah *input*. Untuk mendapatkan fungsi dengan baik maka perlu diperhatikan proses inisialisasi yang telah ditentukan oleh pabrik pembuatnya. *Timming* penginisialisasian sangat perlu dipertimbangkan, karena jika meleset sampai orde *millisecond*, maka dapat dipastikan LCD itu tidak dapat berfungsi.

Ada dua jenis register yang terdapat dalam LCD M1632 ini, yaitu data register dan *instruction* register. Dengan menggunakan pin RS (*Register Select*) pada LCD, pemakaian kedua register dapat dipilih. Pemilihan. Pemilihan register pada LCDditunjukkan dalam tabel berikut ini :

Tabel 2.2 Pemilihan Register Pada LCD M1632[8]

Nama Sinyal	No. Terminal	I/O	Tujuan	Keterangan Sinyal
RS	4	Input	MPU	0 : *Instruction* Register 1 : Data Register

Jika bagian yang dipilih adalah *instruction* register maka output yang dihasilkan adalah meliputi operasional dari LCD, misalnya fungsi *display clear, cursor home, entry mode set, display on/off, cursor shift*, dan sejenisnya.

Maka total penjumlahan hasil bagi adalah 1 x 1000 + 4 x 100 + 4 x 10 + 1 x 1 =

1441 biner

2.2 LCD (*Liquid Crystal Display*)

LCD (*Liquid Crystal Display*) adalah suatu jenis piranti output yang menggunakan daya rendah dengan pengontrol kontras dan kecerahan. Pengontrol utamanya dan karakter ada pada ROM (*Read Only Memory*) generator dan *display* data RAM (*Random Access Memory*) yang akan menghasilkan cara *asied key codes* (kode tombol *keyboard* standar internasional dalam ASCII) jika padanya diberikan sebuah *input*. Untuk mendapatkan fungsi dengan baik maka perlu diperhatikan proses inisialisasi yang telah ditentukan oleh pabrik pembuatnya. *Timing* penginisialisasian sangat perlu diperhatikan, karena jika meleset sampai orde *millisecond*, maka dapat dipastikan LCD itu tidak dapat berfungsi.

Ada dua jenis register yang terdapat dalam LCD M1632 ini, yaitu data register dan *instruction* register. Dengan menggunakan pin RS (*Register Select*) pada LCD, pemakaian kedua register dapat dipilih. Pemilihan. Pemilihan register pada LCD ditunjukkan dalam tabel berikut ini :

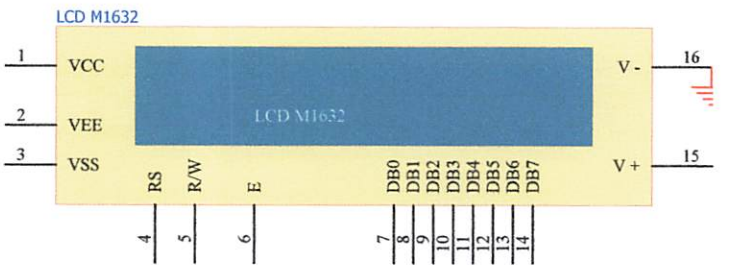
Tabel 2.2 Pemilihan Register Pada LCD M1632[8]

Nama Signal	*No. Terminal*	*I/O*	*Tujuan*	*Keterangan Signal*
RS	4	input	MPU	0 : *Instruction* Register 1 : Data Register

Jika bagian yang dipilih adalah *instruction* register maka output yang dihasilkan adalah instruksi operasional dari LCD, misalnya fungsi *display clear*, *cursor home*, *entry mode set*, *display on/off*, *cursor shift*, dan sejenisnya.

Sebaliknya, jika bagian yang dipilih adalah data register, output yang dihasilkan adalah meliputi karakter yang tabelnya terdapat pada lampiran *datasheet* LCD.

Berikut adalah gambar dari LCD dengan pin- pin yang terhubung dengan mikrokontroler AT89S8253 :



Gambar 2.6 Modul LCD 2 ×16 karakter[8]

LCD modul M1632 mempunyai 16 pin dengan fungsi sebagai berikut :

Tabel 2.3 Fungsi Pin – Pin LCD[8]

No. PIN	Nama PIN	Fungsi
1	Vss	Terminal Ground
2	Vcc	Tegangan Catu + 5 volt
3	Vee	Mengendalikan kecerahan LCD
4	RS	Sinyal pemilihan register 0 = Tulis 1 = Baca
5	R/W	Sinyal seleksi tulis atau baca 0 = Tulis 1 = Baca
6	E	Sinyal operasi awal yang mengaktifkan data tulis atau baca
7 – 14	DB0 – DB7	Merupakan saluran data berisi perintah data yang akan ditampilkan
15	V + BL	*Back Light Supply* 4 - 4,2 (Volt)
16	V – BL	*Back Ligth Supply* 0 (*Ground*)

Untuk berhubungan dengan mikrokontroler pemakai, M1632 dilengkapi dengan 8 jalur data (DB0..DB7) yang dipakai untuk menyalurkan kode ASCII

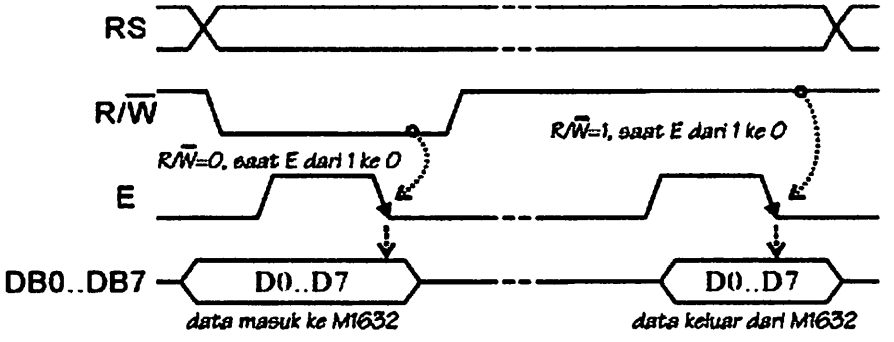
maupun perintah pengatur kerjanya M1632. Selain itu dilengkapi pula dengan E, R/W dan RS seperti layaknya komponen yang kompatibel dengan mikroprosesor.

Kombinasi lainya E dan R/W merupakan sinyal standar pada komponen buatan Motorola. Sebaliknya sinyal-sinyal dari MCS51 merupakan sinyal khas Intel dengan kombinasi sinyal WR dan RS..

RS, singkatan dari *Register Select,* dipakai untuk membedakan jenis data yang dikirim ke M1632, kalau RS=0 data yang dikirim adalah perintah untuk mengatur kerja M1632, sebaliknya kalau RS=1 data yang dikirim adalah kode ASCII yang ditampilkan.

Demikian pula saat pengambilan data, saat RS=0 data yang diambil dari M1632 merupakan data status yang mewakili aktivitas M1632, dan saat RS=1 maka data yang diambil merupakan kode ASCII dari data yang ditampilkan.

Proses mengirim/mengambil data ke/dari M1632 digambarkan dalam gambar 2.7 bisa dijabarkan sebagai berikut :



Gambar 2.7 Mengirim/Mengambil Data Ke/Dari M1632[8]

1. RS harus dipersiapkan dulu, untuk menentukan jenis data seperti yang telah dibicarakan di atas.

2. R/W di-nol-kan untuk menandakan akan diadakan pengiriman data ke M1632. Data yang akan dikirim disiapkan di DB0..DB7, sesaat kemudian sinyal E di-satu-kan dan di-nol-kan kembali. Sinyal E merupakan sinyal sinkronisasi, saat E berubah dari 1 menjadi 0 data di DB0 .. DB7 diterima oleh M1632.
3. Untuk mengambil data dari M1632 sinyal R/W di-satu-kan, menyusul sinyal E di-satu-kan. Pada saatu E menjadi 1, M1632 akan meletakkan datanya di DB0 .. DB7, data ini harus diambil sebelum sinyal E di-nol-kan kembali.

2.3 IC *Information Storage Device* (ISD) 2590

IC penyimpan suara yang digunakan merupakan jenis EEPROM *(Electrically Erasable Programable Read Only Memory)* yaitu ROM yang dapat diprogram, dihapus dan diprogram ulang secara elektrik dengan arus listrik, bukan sinar *ultraviolet*. IC ISD *(Information Strorage Device),* yang dipakai yaitu ISD 2590. IC ini dapat merekam pesan maksimal 90 detik dan dapat dikaskade sehingga pesan yang disimpan dapat diperpanjang sesuai dengan keinginan kita dengan alamat yang berbeda.

Didalam ISD 2590 dilengkapi dengan internal *amplifier*, *internal automatic gain control (AGC), filter antialiasing* (perata) dan *speaker amplifier* (penguat *speaker*). Secara keseluruhan seri ISD 2590 dapat melakukan sebuah perekaman atau pemutaran ulang pesan dengan komponen sederhana seperti mikropon, *speaker*, beberapa komponen penunjang, duah buah saklar dan sumber tegangan.

2. R/W di-nol-kan untuk memastikan akan diadakan pengiriman data ke M1632. Data yang akan dikirim disiapkan di DB0..DB7, sesaat kemudian sinyal E di-satu-kan dan di-nol-kan kembali. Sinyal E merupakan sinyal sinkronisasi, saat E berubah dari 1 menjadi 0 data di DB0 .. DB7 diterima oleh M1632.
3. Untuk mengambil data dari M1632 sinyal R/W di-satu-kan, menyusul sinyal E di-satu-kan. Pada saat E menjadi 1 M1632 akan meletakkan datanya di DB0 .. DB7, data ini harus diambil sebelum sinyal E di-nol-kan kembali.

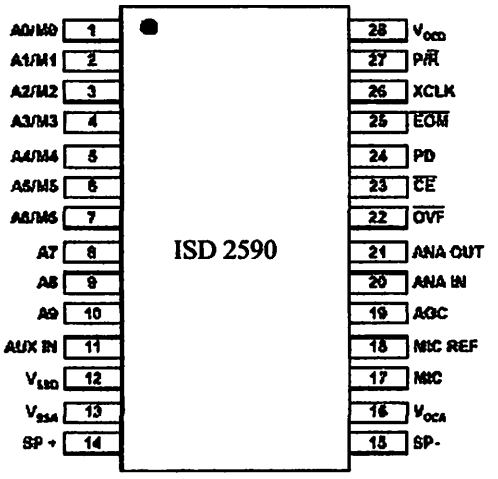
2.3 IC *Information Storage Device* (ISD) 2590

IC perekaman suara yang digunakan merupakan jenis EEPROM (*Electrically Erasable Programmable Read Only Memory*) yaitu ROM yang dapat diprogram, dihapus dan diprogram ulang secara elektrik dengan arus listrik bukan sinar *ultraviolet*. IC ISD (*Information Storage Device*) yang dipakai yaitu ISD 2590. IC ini dapat merekam pesan maksimal 90 detik dan dapat dikaskade sehingga pesan yang disimpan dapat diperpanjang sesuai dengan keinginan kita dengan alamat yang berbeda.

Didalam ISD 2590 dilengkapi dengan internal *amplifier*, internal *automatic gain control* (AGC), *filter antialiasing*, *preamp* dan *speaker amplifier* (penguat *speaker*). Secara keseluruhan seri ISD 2590 dapat melakukan sebuah perekaman atau pemutaran ulang pesan dengan komponen tambahan seperti mikropon, *speaker*, beberapa komponen penunjang, buah-buah saklar dan sumber tegangan.

Rekaman akan disimpan dalam sel memori yang tidak mudah hilang *(non volztile),* memberikan tempat penyimpanan yang masih kosong. Cara unik ini yang membuat ISD disebut *Direct Analog Stroge Technology* (DAST) atau teknik penyimpanan analog langsung, dengan jalan sinyal suara (*voice*) dan bunyi disimpan secara langung dalam bentuk analog, kedalam memori EPROM. Penyimpanan analog langsung memungkinkan reproduksi suara secara alami dalam satu *chip* tunggal.

Susunan ISD 2590 DAST adalah memiliki alamat A0 sampai A7 yang menunjukkan akses tips segmen dalam kesatuan untuk alamat pesan. Kemampuan pemberian atau penyediaan alamat yang berupa pesan yang disimpan dalam bentuk kalimat dan suara.



Gambar 2.8 Susunan kaki ISD2590[3]

Penjelasan dari fungsi spesifik masing-masing kaki dari ISD2590 adalah sebagai berikut:

- ***Address* Input (A0-A7) Pin 1-6 Dan 9-10**

  Input alamat ini mempunyai dua fungsi, tergantung dari level dari dua *Most Significant Bits* (MSB) dari alamat. Jika dua MSB ini keduanya *low*, maka semua *input* digunakan sebagai bit pengalamatan *(Adress Bits)* dan digunakan sebagai alamat untuk memulai *(Start Address)* dari perekaman atau pemutaran ulang *(Play Back)*. Kaki-kaki dari pengalamat hanya merupakan masukan dan bukan merupakan informasi keluaran pengalamatan *internal (Output Internal Address Information)*. Ketika proses operasional sedang berjalan dan pada saat kedua MSB ini *high*, maka sinyal *input* pengalamatan digunakan sebagai bit mode *(Mode Bit)* yang membuat mode operasi normal dan pengalamatan secara tidak langsung *(Simultanously)*.

- **Vssd dan Vssa (*Ground*) Pin 12 dan 13**

  Sama seperti Vccd dan Vcca, analog *input* dan digital sirkuit di dalam ISD 2590 menggunakan *bus ground* yang terpisah untuk meminimalisasi *noise*. Pin ini harus dihubungkan sedekat mungkin dengan *ground*.

- **Speaker *Output* (SP +, SP -) Pin 14 dan 15**

  Pin SP + dan SP – digunakan untuk mengeluarkan suara yang telah direkam ke *speaker* atau ke *device* lainnya. *Output* ini mempunyai impedansi sebesar 16 Ohm.

- ***Chip Enable* (CE) Pin 23**

  Ketika sinyal ini berpindah dari *high* ke *low*, maka pin P/R akan aktif/berjalan. pin P/R akan aktif sampai *input* ini kembali ke *high* yang

• *Address Input* (A0-A7) Pin 1-6 Dan 9-10

Input alamat ini mempunyai dua fungsi, tergantung dari level dari dua *Most Significant Bit* (MSB) dari alamat. Jika dua MSB ini keduanya *Low*, maka semua *input* digunakan sebagai bit pengalamatan (*address bits*) dan digunakan sebagai alamat untuk memulai (*Start Address*) dari perekaman atau pemutaran ulang (*Play Back*). Kaki-kaki dari pengalamat hanya merupakan masukan dan tidak mengeluarkan informasi keluaran pengalamatan internal (*Output Internal Address Information*). Ketika proses operasional sedang berjalan dan pada saat kedua MSB ini *High*, maka sinyal *input* pengalamatan digunakan sebagai bit mode (*Mode Bit*) yang membuat mode operasi normal dan pengalamatan secara tidak langsung (*Nonaddressable*).

• *Vccd* dan *Vssa* (*Ground*) Pin 12 dan 13

Sama seperti Vccd dan Vcca, analog *input* dan digital sirkuit di dalam ISD 2590 menggunakan *bus ground* yang terpisah untuk meminimalisasi *noise*. Pin ini harus dihubungkan sedekat mungkin dengan *ground*.

• *Speaker Output* (SP+, SP-) Pin 14 dan 15

Pin SP+ dan SP- digunakan untuk mengeluarkan suara yang telah direkam ke *speaker* atau ke *device* lainnya. *Output* ini mempunyai impedansi sebesar 16 Ohm

• *Chip Enable* (CE) Pin 23

Ketika sinyal ini berpindah dari *high* ke *low*, maka pin P/R akan aktif/berjalan, pin P/R akan aktif sampai *input* ini kembali ke *high* yang

berarti tanda akhir dari pesan atau ruang memori sudah habis. ISD akan kembali ke mode *standby* setelah pin P/R tidak aktif.

- ***Power Down* (PD) Pin 24**

  Ketika sinyal akan bepindah menuju *low (low-going transition)* terdeteksi di *input* ini, maka pin P/R akan berjalan. Pin P/R berjalan sampai tanda akhir dari pesan tercapai (Akhir dari ruang memori tercapai). Setelah selesai pemutaran ulang/perekaman, ISD secara otomatis akan kembali ke mode *stanby,* menekan pin PD ke *high* pada waktu pin P/R berjalan tidak akan menghentikan operasi dari pin P/R. Jadi proses pemutaran ulang/perekaman akan berhenti bila mencapai akhir dari pesan atau ruang memori habis.

- ***Playback/Record* (P/R) Pin 27**

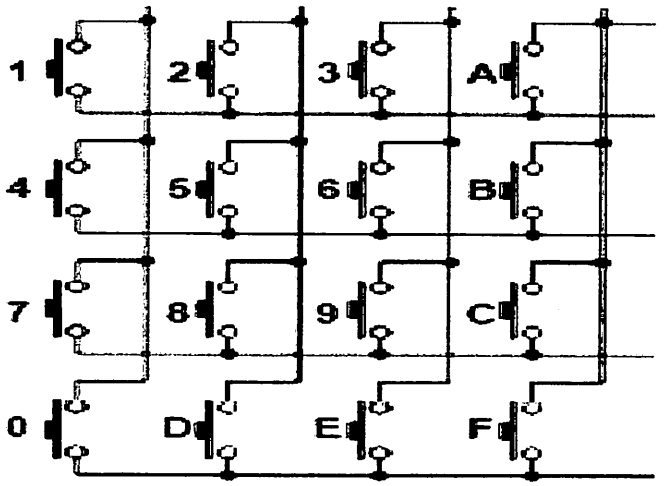
  Jika *Input* sinyal dari pin ini dalam kondisi *low*, ISD 2590 akan merekam suara, dan sinyal ini harus terus dalam keadaan *low* bila ingin terus merekam. Jika *input* sinyal dari pin P/R *high* maka ISD 2590 akan melakukan pemutaran ulang dari suara yang telah direkam sesuai dengan alamat yang diinginkan.

- **VCCA Dan VCCD Pin 16 Dan 28**

  Analog dan digital sirkuit yang terdapat didalam *chip* ISD 2590 menggunakan *bus power* yang terpisah untuk meminimalisasi *noise*. Pin power ini harus dihubungkan sedekat mungkin dengan sumber tegangan.

2.4 *Keypad*

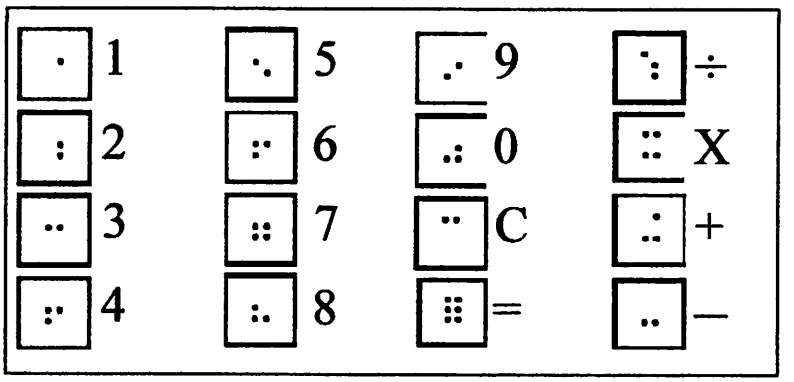
Keypad ini terdiri dari beberapa *switch* yang disusun secara matrik dimana jumlah dari *switch* tersebut adalah perkalian antara jumlah baris dan kolom. Rangkaian susunan *keypad* matrik 4×4 dapat dilihat pada gambar dibawah ini.



Gambar 2.9 Rangkaian *Keypad* 4x4[8]

2.5 Huruf/Angka *Braille*

Braille adalah suatu kode timbul yang digunakan untuk tuna netra dan para tuna netra dapat membaca huruf berdasarkan kode ini dengan cara meraba menggunakan jari. Adapun contoh dari *Braille* itu sendiri adalah sebagai berikut:



Gambar 2.10 Huruf *Braille*[10]

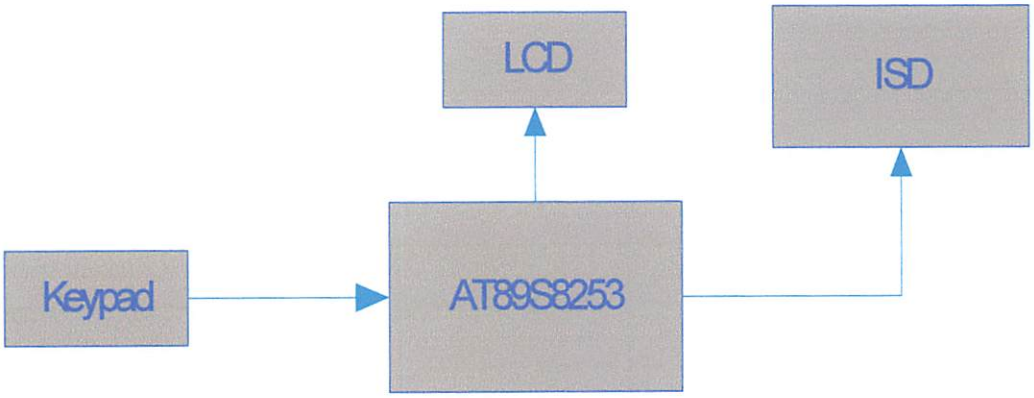
BAB III

PERANCANGAN DAN PEMBUATAN ALAT

Dalam bab ini akan dibahas mengenai perancangan dan pembuatan alat yang meliputi perangkat keras (*hardware*), perangkat lunak (*software*). Pembahasan dalam bab ini akan dilakukan perblok.

3.1. Perancangan Perangkat Keras (*Hardware*)

Perancangan *hardware* utamanya adalah berdasarkan blok sistem dari seluruh proses seperti yang ditunjukkan oleh gambar 3.1 berikut ini:



Gambar 3.1 Diagram Blok Keseluruhan Sistem

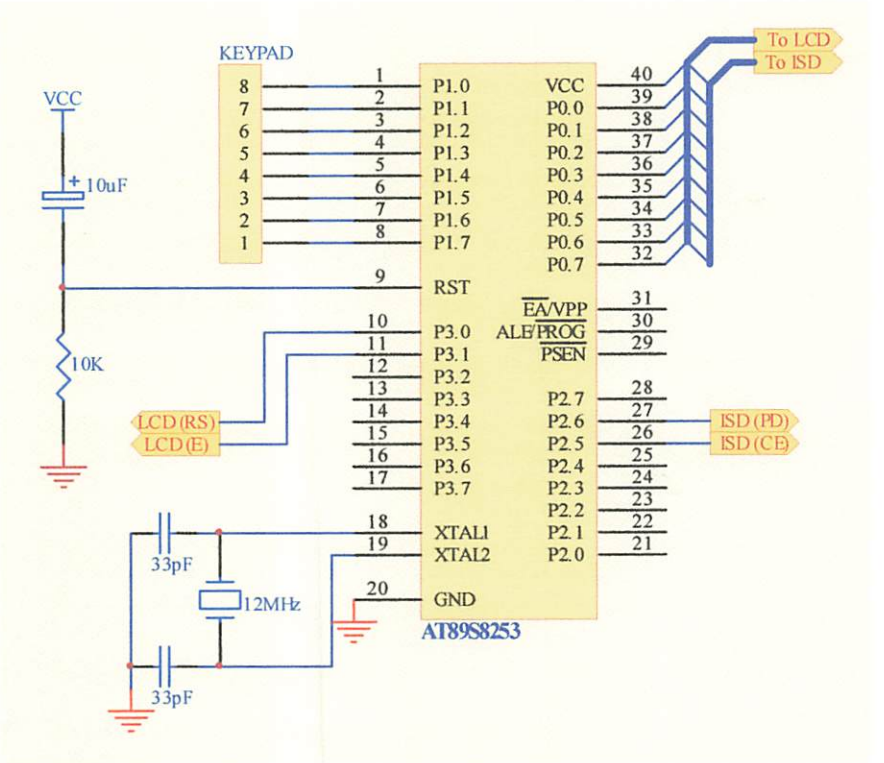
Fungsi dari tiap-tiap blok diagram dijelaskan sebagai berikut:

1. Unit sistem MC 89S8253 merupakan unit pengolah data.
2. Unit LCD berfungsi untuk menampilkan data/*output* data.
3. Unit *keypad* berfungsi untuk memasukkan data/*input* data.
4. Unit ISD sebagai memori suara terprogram yang dapat menyimpan suara dalam bentuk data-data digital dan dapat dipanggil dan mengubahnya kembali menjadi data suara *audio*.

3.1.1. Sistem Mikrokontroler AT89S8253

Mikrokontroler AT89S8253 berfungsi sebagai pengolah data yang dihasilkan oleh penekanan *keypad* yang berfungsi sebagai *input* dan menampilkan data tersebut ke dalam LCD. Mikrokontroler juga akan menampilkan data identitas penulis ke dalam LCD pada awal penyalaan sistem.

Pengaturan jalur *input* dan *output* pada rangkaian mikrokontroler untuk sebuah rancangan terprogram, sangat berkaitan erat dengan program yang kita buat. Agar tidak terjadi kesalahan saat pembacaan data, mikrokontroler menyediakan jalur-jalur 32 *input-output* yang dapat digunakan secara berkelompok atau bersamaan untuk tiap kelompok terisi 8 bit. Untuk lebih jelasnya, hubungan mikrokontroler dengan perangkat pendukungnya dapat dilihat dalam gambar 3.2 di bawah ini:



Gambar 3.2 Perancangan Sistem Mikrokontroler AT89S8253

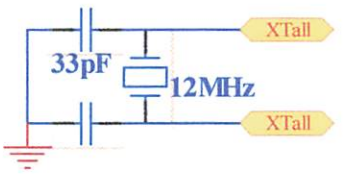
Berikut adalah fungsi dari pin-pin mikrokontroler AT89S8253:

- Port 0 digunakan sebagai *output* (memberikan) data ke LCD dan ISD.
- Port 1 digunakan untuk menerima *input* dari *keypad* 4 x 4.
- Port 2.6 digunakan untuk mengaktifkan ISD yaitu pada pin CE.
- Port 2.7 digunakan untuk mereset ISD yaitu pada pin PD.
- Port 3.0 digunakan untuk memberikan pulsa ke RS pada LCD.
- Port 3.1 digunakan untuk memberikan pulsa ke E pada LCD.

3.1.1.1 Perancangan Rangkaian Clock

Kecepatan proses yang dilakukan oleh mikrokontroler ditentukan oleh sumber *clock* yang mengendalikan mikrokontroler tersebut. Sistem yang dirancang ini menggunakan osilator internal yang telah tersedia dalam *chip* AT89S8253. Untuk menentukan frekuensi osilatornya cukup dengan menghubungkan kristal dalam pin 19 (X1) dan pin 18 (X2) serta dua buah kapasitor ke *ground*.

Besarnya kapasitansinya disesuaikan dengan spesifikasi dalam lembar data AT89S8253 yaitu 33pF. Kristal yang digunakan adalah 12 MHz. gambar 3.3 memperlihatkan rangkaian *clock* yang dirancang.

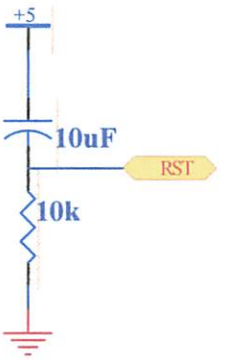


Gambar 3.3 Rangkaian *Clock* untuk MCU AT89S8253[1]

Dalam sistem ini, rangkaian terdiri dari dua buah kapasitor sebesar 33 pF dan sebuah kristal sebesar 12 MHz.

3.1.1.2 Rangkaian Reset

Mikrokontroler AT89S8253 dapat bekerja jika ada rangkaian resetnya. AT89S8253 memakai reset aktif *high* sehingga *input* reset harus tinggi minimal selama 2 siklus mesin (24 periode osilator) saat pertama kali mikrokontroler AT89S8253 dijalankan. Rangkaian reset terdiri atas resistor dan kapasitor yang dihubungkan ke kaki 9 pada mikrokontroler AT89S8253.



Gambar 3.4 Rangkaian Reset[1]

Karena kristal yang digunakan mempunyai frekuensi sebesar 12 MHz maka satu periode membutuhkan waktu sebesar :

$$T = \frac{1}{f_{XTALL}} = \frac{1}{12\,MHz}\,s = 8.333 \cdot 10^{-8}\,s$$

Sehingga waktu minimal logika tinggi yang dibutuhkan untuk me*reset* mikrokontroler adalah :

$$\begin{aligned} t_{reset(\min)} &= T \times \text{ periode yang dibutuhkan} \\ &= 8{,}333 \cdot 10^{-8} \times 24 \\ &= 2\,\mu S \end{aligned}$$

Jadi mikrokontroler membutuhkan waktu minimal 2 µs untuk me*reset*. Waktu minimal inilah yang dijadikan pedoman untuk menentukan nilai R dan C.

Dengan menentukan nilai R = 10 KΩ dan *t* sebesar 0,035detik. Nilai *t* lebih besar daripada 2 × 10^{-6} detik sehingga rangkaian tersebut dapat digunakan untuk me-*reset* mikrokontroler.

$$t = 0{,}357 \cdot R \cdot C$$
$$35 \cdot 10^{-3} = 0{,}357 \cdot 10 \cdot 10^{3} C$$
$$C = 9{,}8 \cdot 10^{-6} F$$

Jadi dengan nilai komponen R = 10 KΩ nilai kapasitor yang dapat memenuhi syarat untuk me*reset* mikrokontroler harus diatas 9,8 μF. Untuk kemudahan perancangan dipilih C = 10 μF.

3.1.2. LCD M1632

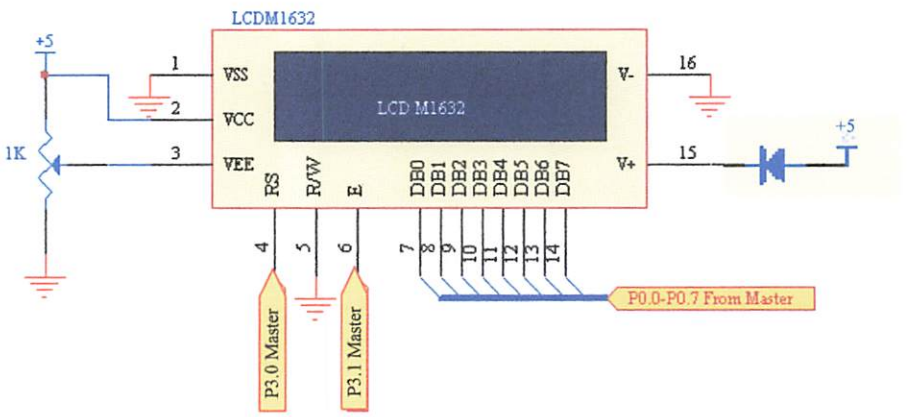
LCD M1632 merupakan panel LCD sebagai media penampil informasi dalam bentuk huruf/angka dua baris, masing-masing baris bisa menampung 16 huruf/angka.

Untuk berhubungan dengan mikrokontroler, pemakai LCD M1632 dilengkapi dengan 8 jalur data (DB0..DB7) yang dipakai untuk menyalurkan kode ASCII maupun perintah pengatur kerjanya M1632. Selain itu dilengkapi pula dengan E, R/W* dan RS seperti layaknya komponen yang kompatibel dengan mikroprosesor.

RS singkatan dari *Register Select*, dipakai untuk membedakan jenis data yang dikirim ke M1632, kalau RS=0 data yang dikirim adalah perintah untuk mengatur kerja M1632, sebaliknya kalau RS=1 data yang dikirim adalah kode ASCII yang ditampilkan.

Demikian pula saat pengambilan data, saat RS=0 data yang diambil dari M1632 merupakan data status yang mewakili aktivitas M1632, dan saat RS=1 maka data yang diambil merupakan kode ASCII dari data yang ditampilkan.

Untuk tampilan dipergunakan LCD Dot Matrik 2 x 16 karakter. Sinyal-sinyal yang diperlukan oleh LCD adalah RS dan Enable, sinyal RS dan Enable dipergunakan sebagai *input* yang *output*nya dipakai untuk mengaktifkan LCD. LCD akan aktif apabila mikrokontroler memberikan instruksi tulis pada LCD. Saat kondisi RS *don't care* dan *Enable* 0 maka LCD tetap pada kondisi semula, pengiriman data ke LCD dilakukan saat RS berlogika 1 dan *Enable* berlogika 1. Instruksi dikirim pada LCD bila keadaan RS 0 dan Enable 1. Pin LCD ini untuk data terkoneksi pada *Port 0* mikrokontoller. Kemudian untuk RS dihubungkan pada *Port 3.0*, tulis/baca *(Read/Write)* diberikan logika *low* karena disini LCD bersifat menulis data, dan yang terakhir *Enable* (E) dikendalikan dengan *Port 3.1*. Gambar rangkaian LCD ditunjukkan pada gambar 3.5 sebagai berikut:



Gambar 3.5 Rangkaian LCD

Oleh karena pada LCD ini hanya menggunakan 8 jalur data yang berarti hanya 8 bit data saja yang dapat dikirim ke LCD. Maka pada perancangan ini dijelaskan pengiriman data ke LCD yang melebihi 8 bit adalah sebagai berikut.

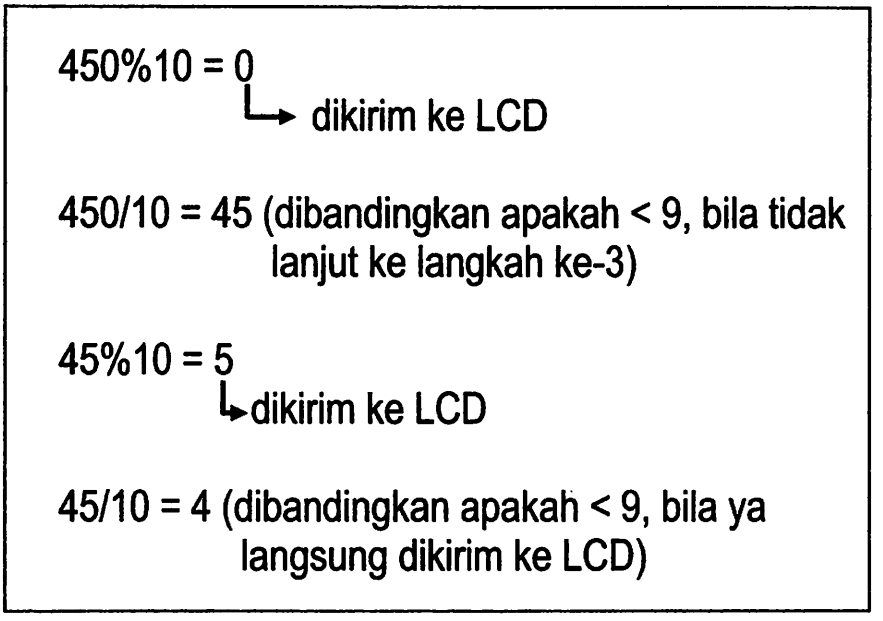
Contoh:

18 x 25 = 450 atau dalam biner 111000010

Disini akan dijelaskan proses pengiriman hasil dari sebuah perkalian dari 18 x 25, yaitu 450. Angka 450 ini tidak dapat langsung di kirim ke LCD karena datanya melebihi 8 bit. Maka disini perlu dilakukan pemecahan data tahap demi tahap agar bisa mendapatkan data 8 bit dan dikirim ke LCD. Adapun langkah-langkahnya adalah sebagai berikut:

1. Melakukan perintah sisa bagi terhadap angka 450 dengan 10 (450 di sisa bagi 10), dari sini akan didapat hasil sisa bagi dari 450 yaitu 0. Dengan demikian telah didapatkan data 8 bit, yaitu 0 (0000 0000), maka angka 0 dapat dikirim ke LCD lebih dahulu.
2. Selanjutnya adalah melakukan perintah pembagian, yaitu 450 dibagi dengan 10 dan hasilnya adalah 45. Angkah 45 ini tidak dikirim ke LCD karena lebih dari 8 bit, tetapi dilakukan perbandingan yaitu apakah 45 > 9, kalau ternyata setelah adanya pembagian, hasil bagi lebih besar dari 9 maka dilakukan proses seperti langkah pertama.
3. Kedua langkah/proses tersebut akan dilakukan secara terus-menerus hingga mendapatkan hasil bagi tidak lebih besar dari 9. Dan data yang terakhir dikirim ke LCD tentunya <= 9.

Untuk lebih jelasnya bisa dilihat gambar berikut ini:

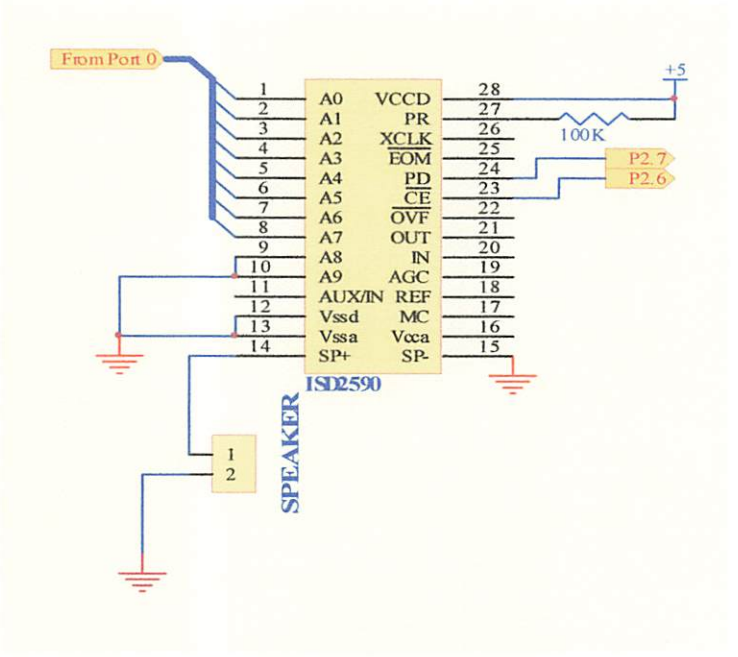


Gambar 3.6. Proses Pengiriman Hasil Aritmatika ke LCD

3.1.3. ISD2590

Pada rangkaian ini suara yang di rekam menggunakan *Information Storage Device* ISD 2590 yang disimpan dalam EEPROM. Berdasarkan data sheet, ISD 2590 ini mampu merekam suara dengan lama perekaman 90 detik dengan alamat yang berbeda. Rangkaian ISD 2590 ditunjukkan pada gambar 3.6. Pin *address* data menerima masukan 8 bit dari mikrokontroler pada Port 0. Alamat-alamat ini akan memilih data suara yang mana yang akan dipanggil. Kemudian untuk pin *Chip Enable* (CE) berfungsi sebagai pengaktifan (*Enabled),* secara aktif *low* yang terkoneksi dengan *Port* 2.6. Untuk *Play/Record* (P/R) pin 27 berfungsi untuk merekam dan memainkan lagi dengan memberikan aktif *low,* karena ISD pada alat ini tidak digunakan untuk merekam suara tetapi hanya untuk memanggil suara yang telah tersimpan maka pin 27 dari ISD ini langsung dihubungkan dengan

ground. Sedangkan *Power Down* (PD) pin 23 berfungsi untuk mereset ISD, pin ini terhubung dengan mikrokontroler pada *Port 2.7*.



Gambar 3.7 Rangkaian ISD2590

Pemutar/perekam suara mempunyai 10 jalur alamat dan 2 buah pin kontrol. Pada perancangan ini pin-pin yang digunakan adalah:

A0-A7 : Alamat ISD yang dihubungkan ke mikrokontroler AT89S8253 port 0.

CE : digunakan untuk mengaktifkan ISD2590 dan dihubungkan ke port 2.6.

PD : digunakan untuk mereset ISD2590 dan dihubungkan ke port 2.7.

Untuk memberikan informasi suara dari tombol yang ditekan pada *keypad* serta hasil dari aritmatika dengan menggunakan IC ISD 2590. Sinyal suara merupakan bentuk sinyal analog kemudian diubah menjadi bentuk digital untuk disimpan ke dalam *memory*. Data-data digital yang sudah tersimpan yang berasal dari data analog (suara) dapat dipanggil kembali dengan memanggil alamat penyimpanan datanya. Karena telah dijelaskan bahwa pada alat ini tidak

digunakan sebagai fungsi perekaman tetapi hanya untuk memanggil suara yang telah terekam maka hanya dijelaskan prosedur pemanggilan data-data suara yang telah direkam adalah sebagai berikut:

1. Pada perancangan alat ini, pin P/R langsung diberikan logika *high* karena pada alat ini ISD hanya sebagai *play* (pemanggilan suara), maka disini hanya perlu mengaktifkan ISD saja, yaitu dengan memberikan pin CE logika *low,* yang berarti ISD aktif dan mulai pemanggilan suara pada alamat memori yang diinginkan, pada waktu perekaman tadi.
2. Bila alamat suara yang telah dipanggil selesai, maka pin CE diberikan logika *high*, dan pin PD berlogika *high* yang akan mereset alamat ISD kembali ke awal (0).
3. Demikian seterusnya pada alamat-alamat selanjutnya, sesuai dengan data suara yang diinginkan.

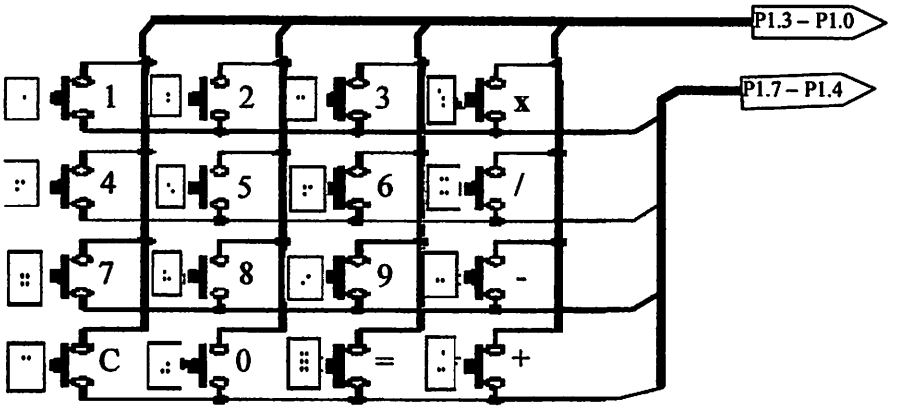
Untuk memudahkan dalam mencari alamat-alamat pada waktu *Playback* menggunakan tabel logika biner 8 bit, dimana tabel ini berasal dari hasil perekaman sebelumnya. Alamat ISD disini hanya digunakan sebanyak 8 bit saja, karena sudah mencukupi untuk kebutuhan. Berikut ini adalah data-data suara dan alamat-alamatnya yang telah direkam dengan lebar data 8 bit.

Tabel 3.1. Data Hasil Perekaman dan Alamatnya Pada ISD 2590

Data Suara yang Direkam	Alamat Data Suara dalam Biner 8 bit	Alamat Data Suara dalam Hexa
Satu	0000 0000	00H
Dua	0000 1001	09H
Tiga	0001 0001	11H
Empat	0001 1001	19H
Lima	0010 0001	21H
Enam	0010 0111	27H
Tujuh	0010 1110	2EH
Delapan	0011 0101	35H
Sembilan	0011 1101	3DH
Sepuluh	0100 1011	4BH
Sebelas	0101 0111	57H
Seratus	0110 0001	61H
Seribu	0110 1010	6AH
Juta	0111 0011	73H
Bagi	1000 0001	81H
Tambah	1000 1010	8AH
Kurang	1001 0010	92H
Kali	1001 1000	98H
Sama dengan	1010 0010	A2H
Nol	1010 1101	ADH
Data error	1011 0101	B5H
Koma	1100 0001	C1H
Minus	1100 1000	C8H

3.1.4. *Keypad* 4 x 4

Rangkaian *keypad* dalam perencanaan ini menggunakan *keypad matrik push button*. Dari tombol-tombol *keypad* tersebut apabila ditekan akan terbentuk angka dan perintah aritmatika, yang merupakan inputan dari alat ini. Teknik pembacaan dari *keypad* ini, yaitu model *scanning* pada jalur baris dan jalur kolom. Bila baris dan kolom disilangkan maka akan terbentuk titik-titik potong antara keduanya seperti pada gambar 3.8 di bawah ini:



Gambar 3.8 Rangkaian *Keypad*

Proses pembacaan *keypad* dengan model *scanning* dapat dijelaskan sebagai berikut :

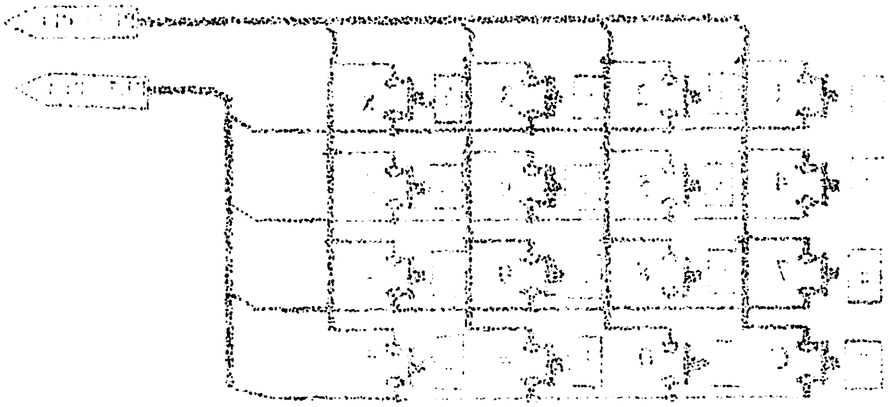
a) Menentukan urutan data (karakter) pada *keypad*, misal (1, 2, 3.....+). Dimana (1) yang pertama dan seterusnya sampai (+) yang terakhir.

b) Selanjutnya menentukan apakah baris atau kolom yang akan kita *scanning*, dalam perancangan ini yang di-*scanning* adalah baris yang berarti diberikan logika *low* (0) untuk kolom 1 – 4 secara bergantian. Dimana dari salah satu kolom itu *low* maka yang lainnya *high*.

c) Setelah itu mikrokontroler akan mengecek pada baris 1 – 4 yang berlogika *low*, kemudian mikrokontroler akan membandingkan dengan data yang sudah ditentukan dalam *software* dan mendefinisikan data tersebut.

d) Demikian seterusnya mikrokontroler akan memulai pengecekan dari kolom 1 – 4 dan men-*scanning* baris 1 – 4 apakah ada yang *low*.

Pada perancangan *keypad* ini digunakan *keypad braille* dimana tombol yang terdapat pada *keypad* ini adalah tombol yang dapat diraba oleh tuna netra, sehingga bagi tuna netra bisa menggunakan keypad ini dengan mudah, karena tiap tombol yang diraba bisa dirasakan dan diartikan sendiri oleh tuna netra.



Gambar 3.8 Rangkaian Keypad

Proses pembacaan keypad dengan model scanning dapat dijelaskan sebagai berikut :

a) Menentukan nomor baris (karakter) pada keypad, misal 1, 2, 3, ..., 4. Dimana 1 yang pertama dan seterusnya sampai 4 yang terakhir.

b) Selanjutnya menentukan apakah baris atau kolom yang akan kita scanning, dalam perancangan ini yang discanning adalah baris yang berarti diberikan logika low (0) untuk kolom 1 – 4 secara bergantian. Dimana dari salah satu kolom itu low maka yang lainnya high.

c) Setelah itu mikrokontroler akan mengecek pada baris 1 – 4 yang berlogika low. Kemudian mikrokontroler akan membandingkan dengan data yang sudah disediakan dalam software dan mendefinisikan data tersebut.

d) Demikian seterusnya mikrokontroler akan memulai pengecekan dari kolom 1 – 4 dan men-scanning baris 1 – 4 apakah ada yang low.

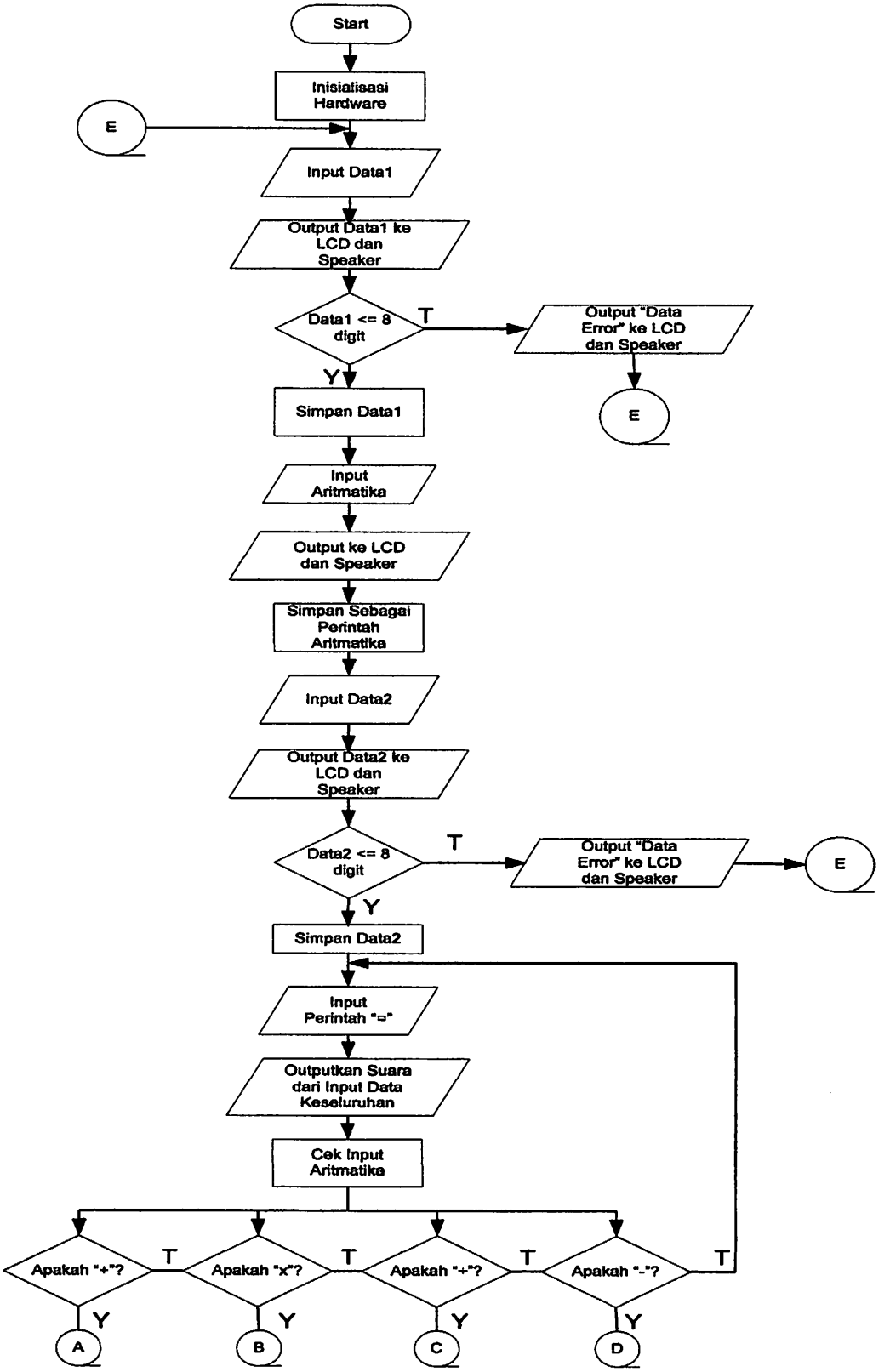
Pada perancangan keypad ini digunakan keypad tactile dimana tombol yang terdapat pada keypad ini adalah tombol yang dapat diraba oleh tuna netra sehingga bagi tuna netra bisa menggunakan keypad ini dengan mudah, karena tiap tombol yang ditekan bisa dirasakan dan diartikan sendiri oleh tuna netra.

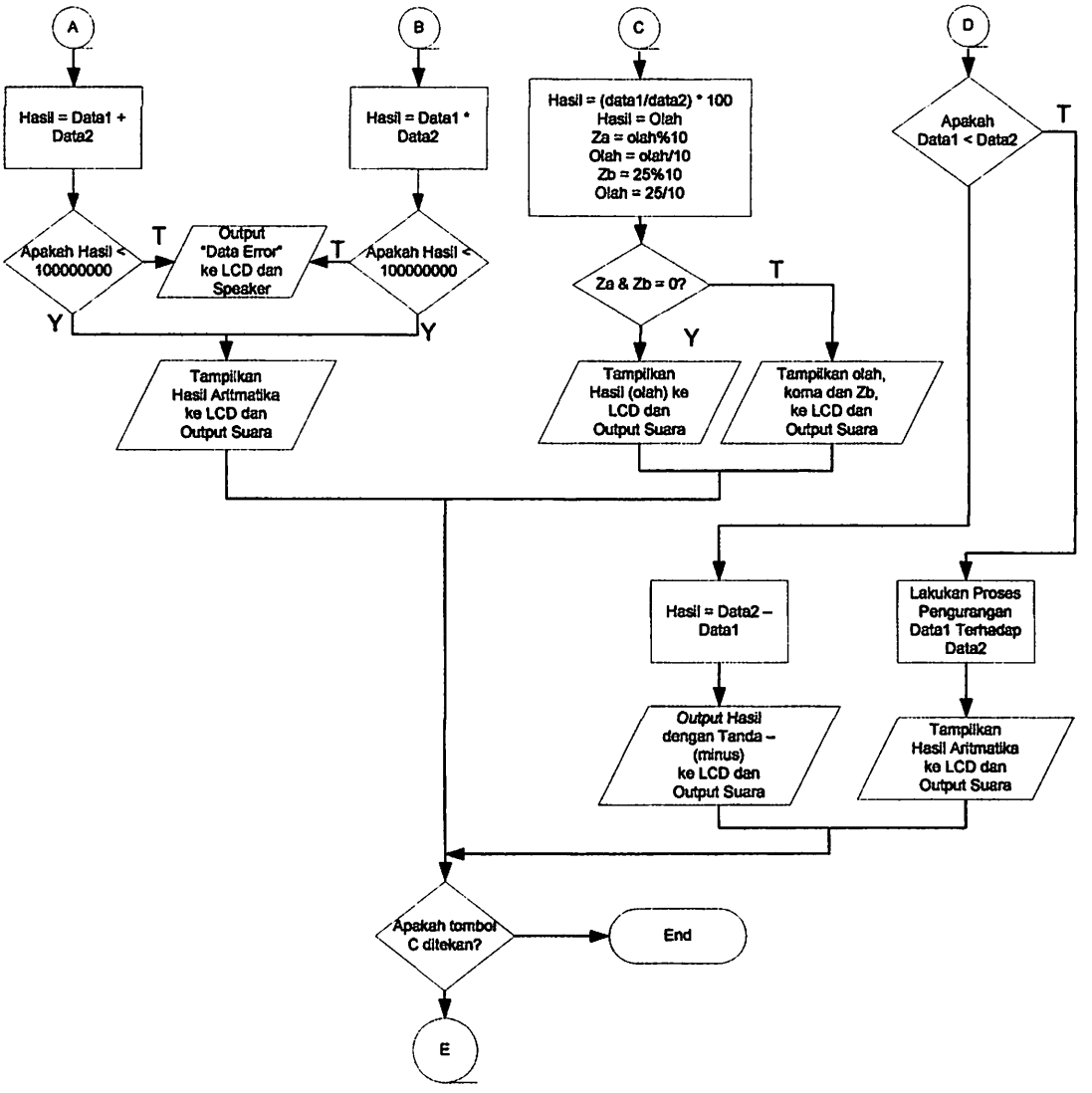
3.2. Perancangan Perangkat Lunak

Untuk mendukung agar perangkat keras berfungsi sesuai dengan perencanaan, maka diperlukan perangkat lunak sebagai penunjangnya. Perangkat lunak ini sendiri maksudnya adalah suatu program yang kita buat yang nantinya akan ditanam kedalam mikrokontroler AT89S8253. setelah mikrokontroler tersebut diprogram, maka akan diketahui apakah program yang telah kita buat bekerja sesuai dengan yang kita rencanakan ataukah masih memiliki kesalahan.

Sistem aplikasi mikrokontroler AT89S8253 ini dapat mengatur dan mengendalikan keseluruhan sistem apabila ada urutan instruksi yang mendefinisikan secara jelas urutan tugas yang harus dikerjakan. Urutan instruksi ini sangat penting untuk didefinisikan, karena mikrokontroler dapat bekerja secara pasti sesuai dengan instruksi yang telah dibuat.

3.2.1 Diagram Alir (*Flowchart*)





Gambar 3.9 *Flowchart* Proses Aritmatika

BAB IV

PENGUJIAN DAN PERCOBAAN ALAT

Dalam bab ini akan dibahas mengenai pengujian dan analisis alat yang telah dibuat. Secara umum, pengujian ini bertujuan untuk mengetahui piranti yang direlisasikan dapat bekerja sesuai dengan spesifikasi perancangan yang telah ditetapkan. Pengujian ini dilakukan pada sistem secara keseluruhan dengan mengintegrasikan perangkat keras (*hardware*) dan perangkat lunak (*software*) untuk mengetahui unjuk kerja sistem.

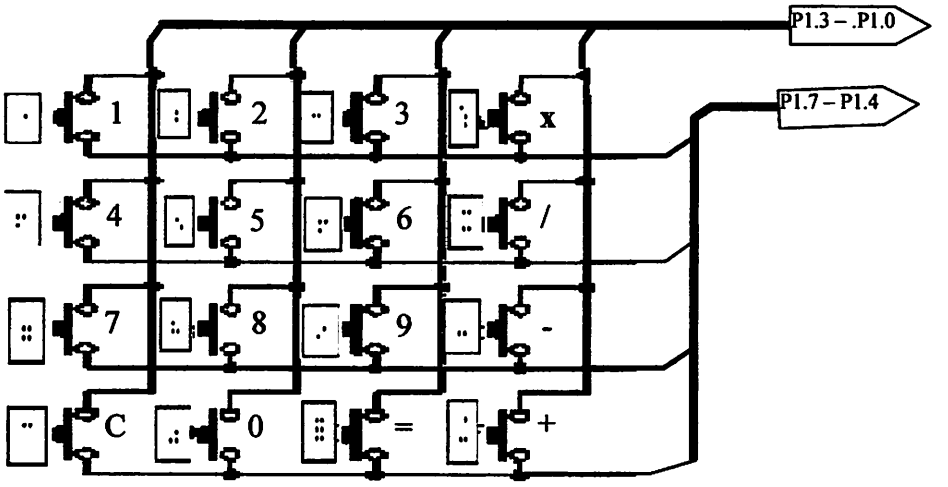
4.1. Pengujian *Keypad*

4.1.1. Tujuan

Untuk mengetahui kebenaran data yang dikirim melalui penekanan tombol.

4.1.2. Pelaksanaan Pengujian

Keypad yang digunakan adalah *keypad* matrik 4x4. Untuk bagian baris dihubungkan pada port 1.4 – 1.7, sedangkan pada bagian kolom dihubungkan pada port 1.0 – 1.3 mikrokontroler AT89S8253. Adapun rangkaian *keypad* seperti gambar di bawah ini :



Gambar 4.1 Rangkaian *Keypad*

Pengujian data *output* baris dan kolom untuk setiap tombol adalah sebagai berikut:

Tabel 4.1 Pengujian data *Output* Baris dan Kolom untuk setiap tombol

No	Baris				Kolom				Keterangan
	P1.7	P1.6	P1.5	P1.4	P1.3	P1.2	P1.1	P1.0	
1	0	1	1	1	0	1	1	1	1
2	0	1	1	1	1	0	1	1	2
3	0	1	1	1	1	1	0	1	3
4	0	1	1	1	1	1	1	0	X
5	1	0	1	1	0	1	1	1	4
6	1	0	1	1	1	0	1	1	5
7	1	0	1	1	1	1	0	1	6
8	1	0	1	1	1	1	1	0	÷
9	1	1	0	1	0	1	1	1	7
10	1	1	0	1	1	0	1	1	8
11	1	1	0	1	1	1	0	1	9
12	1	1	0	1	1	1	1	0	+
13	1	1	1	0	0	1	1	1	C
14	1	1	1	0	1	0	1	1	0
15	1	1	1	0	1	1	0	1	E
16	1	1	1	0	1	1	1	0	-

Data-data dari tabel di atas akan diolah oleh mikrokontroler sesuai dengan fungsi yang diinginkan seperti yang tercantum pada "keterangan" di tabel.

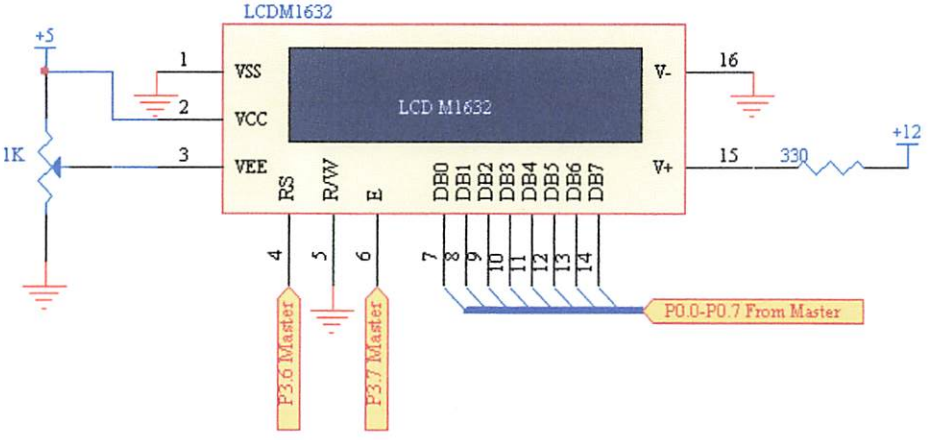
4.2. Pengujian LCD M1632

4.2.1. Tujuan

Untuk mengetahui apakah unit tampilan (LCD) dapat bekerja dengan baik dalam menampilkan huruf pada layar LCD.

4.2.2. Peralatan Yang Digunakan

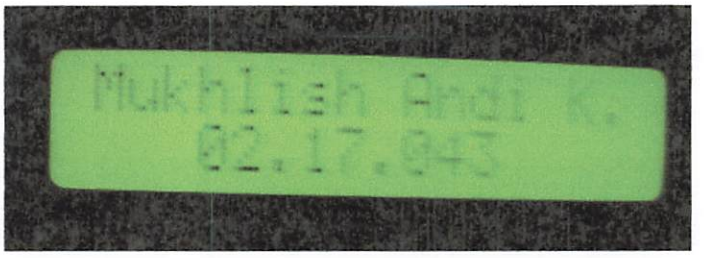
1. Minimum sistem AT89S51.
2. LCD M1632.
3. Catu daya 5 Volt.



Gambar 4.2 Pengujian LCD

4.2.3. Hasil Pengujian

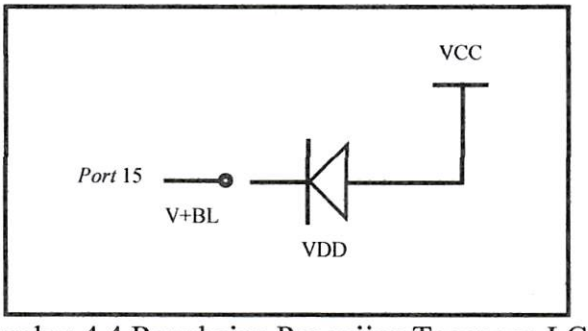
Dalam pengujian rangkaian unit tampilan (LCD) ini terlihat bahwa LCD dapat menampilkan tulisan dan angka dengan baik, hal ini seperti terlihat pada gambar berikut ini:



Gambar 4.3 Hasil Pengujian LCD

4.2.4. Pengujian Tegangan LCD

Selain hal di atas, pengujian LCD juga dapat dilakukan dengan mengukur besar tegangan yang masuk melalui *port* 15 pada LCD. Pengujian dapat dilakukan dengan rangkaian seperti gambar 4.4 di bawah ini :



Gambar 4.4 Rangkaian Pengujian Tegangan LCD

4.2.5. Langkah-langkah Pengujian

1. Dalam keadaan menyala, ukur power supply sebagai VCC.
2. Dalam keadaan menyala, ukur dioda sebagai VDD.
3. Kurangi nilai VCC dengan nilai VDD, hasilnya sebagai V+BL.
4. Nilai V+BL inilah yang masuk ke *port* 15 pada LCD, yang akan menyalakan lampu pada LCD.

4.2.6. Analisa Hasil Pengujian

Tabel 4.2 Pengujian Tegangan VCC Pada LCD

Perhitungan	Pengukuran	Error
5 V	4,9 V	2,04 %
5 V	4,89 V	2,24 %
5 V	4,91 V	1,83 %
5 V	4,87 V	2,66 %
5 V	4,9 V	2,04 %
	Error Rata-Rata	2,1%

Dari hasil pengujian didapatkan rata-rata nilai VCC adalah 4,89 volt dan nilai VDD adalah 0,7 volt. Dengan mengurangi nilai VCC dengan nilai VDD, maka didapatkan nilai V+BL sebesar 4,2 volt. Maka dengan nilai V+BL sebesar 4,2 volt LCD sudah dapat menyala. (*Module M1632 User Manual*)

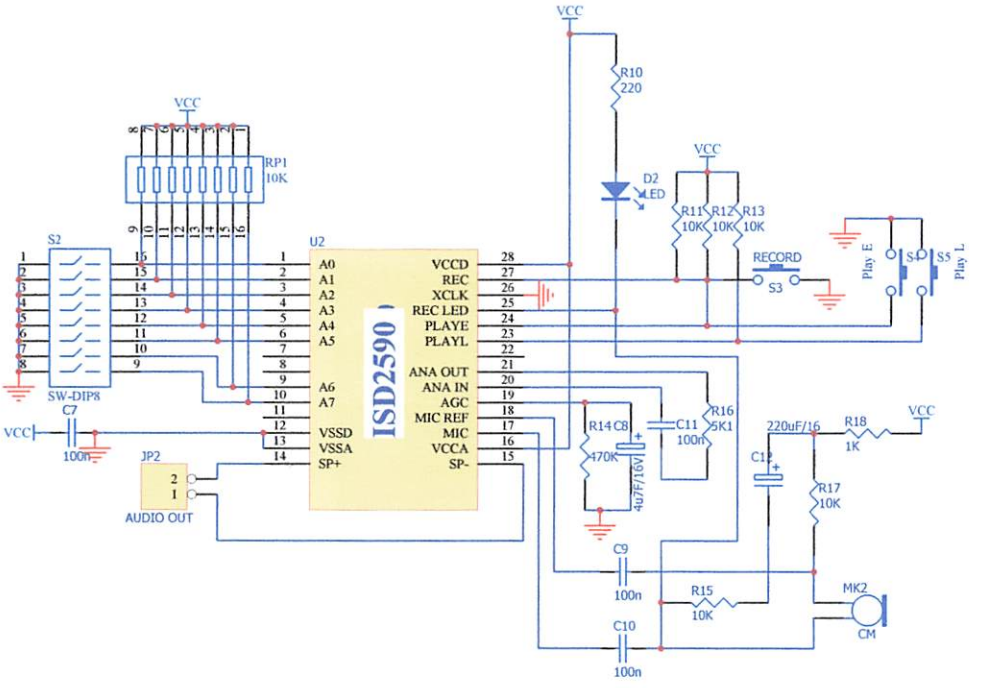
4.3. Pengujian Rangkaian *Information Storage Device (ISD)* 2590.

4.3.1. Tujuan.

Untuk mengetahui data biner 8 bit dari mikrokontroler dapat memanggil data suara yang disimpan pada memori ISD 2590.

4.3.2. Peralatan Yang Digunakan.

- DIP Switch 8 bit.
- Modul ISD 2590 (*Datasheet* ISD 2590)
- Catu Daya 5 Volt.
- *Loudspeaker*
- *Microphone*



Gambar 4.5 Modul ISD 2590

4.3.3. Langkah Pengujian

- Memberikan alamat dengan memasukkan data pada *DIP-Switch.*
- Menekan tombol *record*, untuk memulai merekam melalui *microphone.*
- Mencatat alamat awal dan alamat akhirnya dan ditabelkan.
- Memanggil alamat yang sudah direkam.
- Menekan tombol *play* dan mendengarkan hasil suaranya.

4.3.4. Hasil Pengujian

Tabel 4.3. Hasil Pengujian ISD 2590 dengan pemanggilan alamat.

Data Suara yang Direkam	Alamat Data Suara dalam Biner 8 bit	Alamat Data Suara dalam Hexa
Satu	0000 0000	00H
Dua	0000 1001	09H
Tiga	0001 0001	11H
Empat	0001 1001	19H
Lima	0010 0001	21H
Enam	0010 0111	27H
Tujuh	0010 1110	2EH
Delapan	0011 0101	35H
Sembilan	0011 1101	3DH
Sepuluh	0100 1011	4BH
Sebelas	0101 0111	57H
Seratus	0110 0001	61H
Seribu	0110 1010	6AH
Juta	0111 0011	73H
Bagi	1000 0001	81H
Tambah	1000 1010	8AH
Kurang	1001 0010	92H
Kali	1001 1000	98H
Sama dengan	1010 0010	A2H
Nol	1010 1101	ADH
Data error	1011 0101	B5H
Koma	1100 0001	C1H
Minus	1100 1000	C8H

Dari tabel hasil pengujian dapat dilihat bahwa rangkaian ISD2590 telah dapat bekerja sesuai dengan yang diharapkan pada perancangan.

4.4. Pengujian Alat Keseluruhan

4.4.1. Tujuan

Untuk mengetahui alat yang dibuat sesuai dengan yang telah dirancangkan.

4.4.2 Cara Pengoperasian Alat :

1. Hidupkan saklar ke posisi ON.

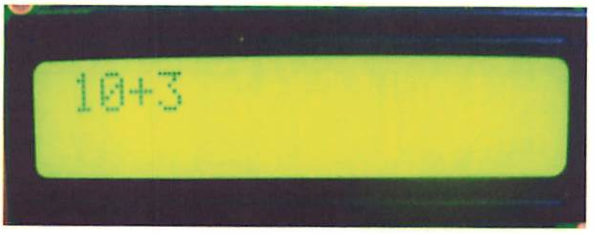
2. Setelah alat menyala, maka muncul nama pembuat alat dan layar LCD terlihat *blank*. Seperti gambar dibawah ini:

Gambar 4.6. Tampilan LCD pada saat saklar di ON-kan

3. Kemudian menekan tombol angka pada *keypad* sesuai dengan proses aritmatika yang kita inginkan.

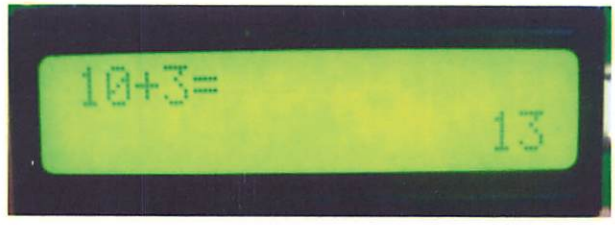
4.4.2.1 Proses Penjumlahan

1. Pada saat ON, tampilan awal adalah LCD blank (tanpa kursor).
2. Kemudian langsung tuliskan angka yang ingin kita jumlahkan seperti gambar dibawah ini:



Gambar 4.7. Menuliskan Angka yang Ingin Dijumlahkan

3. Tekan "="



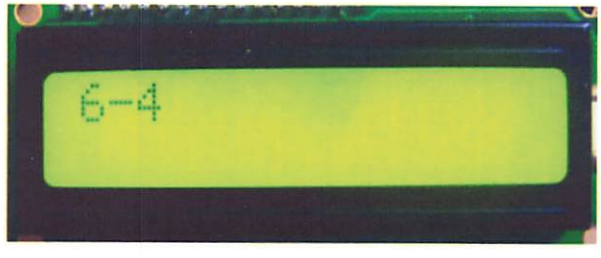
Gambar 4.8 Hasil Penjumlahan

4. Jika angka yang ditulis lebih dari 8 digit maka akan tampil pesan "Data Error".

Gambar 4.9 Tampilan Lebih dari 8 Digit

4.4.2.2 Proses Pengurangan

1. Pada saat ON, tampilan awal adalah LCD blank (tanpa kursor).
2. Kemudian langsung tuliskan angka yang diinginkan seperti gambar dibawah ini:

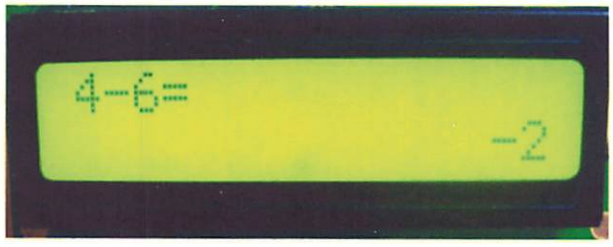


Gambar 4.10 Menuliskan Angka yang Akan Dikurangi

3. Tekan "="

Gambar 4.11 Hasil Penjumlahan

4. Jika dilakukan pengurangan bilangan yang lebih kecil terhadap bilangan yang lebih besar maka akan muncul tanda minus.



Gambar 4.12 Hasil Pengurangan dengan Tampilan Tanda Minus

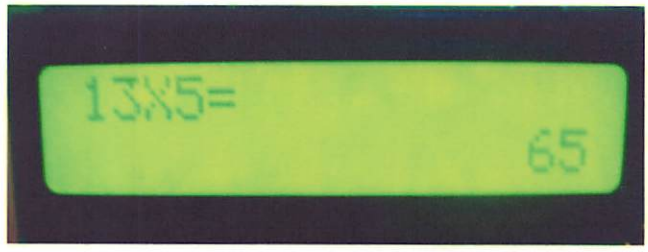
5. Jika angka yang ditulis lebih dari 8 digit maka akan tampil pesan "Data Error".

4.4.2.3 Proses Perkalian

1. Pada saat ON, tampilan awal adalah LCD blank (tanpa kursor).
2. Kemudian langsung tuliskan angka yang diinginkan seperti gambar dibawah ini:

Gambar 4.13 Menuliskan Angka yang Akan Dikalikan

3. Tekan "="

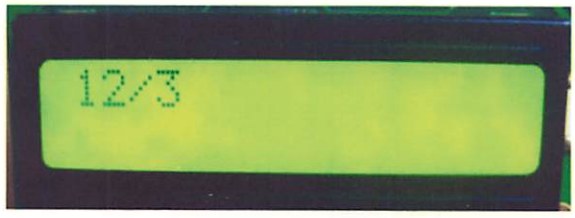


Gambar 4.14 Hasil Perkalian

4. Jika angka yang ditulis lebih dari 8 digit maka akan tampil pesan "Data Error".

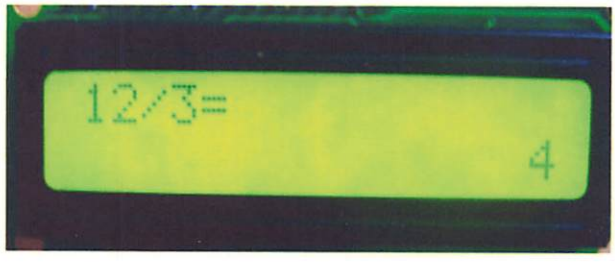
4.4.2.4 Proses Pembagian

1. Pada saat ON, tampilan awal adalah LCD blank (tanpa kursor).
2. Kemudian langsung tuliskan angka yang diinginkan seperti gambar dibawah ini:



Gambar 4.15 Menuliskan Angka yang Akan Dibagi

3. Tekan "="



Gambar 4.16 Hasil Pembagian

Gambar 4.17 Hasil Pembagian Disertai Tanda Koma

4. Jika angka yang ditulis lebih dari 8 digit maka akan tampil pesan "Data Error".

4.4.2.5 Cara Pengucapan

1. Setelah alat dihidupkan atau dalam posisi ON.
2. Suara akan keluar melalui *speaker* pada setiap tombol *keypad* dan yang tampil pada LCD.

Tabel 4.4. Hasil Pengujian Pengucapan

***Keypad* yang Ditekan / Karakter yang Tampil pada LCD**	**Pengucapan**
1	Satu
1 + 12 = 13	Satu tambah dua belas sama dengan tiga belas
4 – 6 = -2	Empat kurangi enam sama dengan minus dua
78 / 5 = 15,6	Tujuh puluh delapan bagi lima sama dengan lima belas koma enam
1 / 3 = 0,33	Satu bagi tiga sama dengan nol koma tiga tiga

BAB V

PENUTUP

5.1. Kesimpulan

Dari perancangan dan pembuatan kalkulator yang dilengkapi dengan *keypad braille* dan *output* suara maka dapat diambil kesimpulan sebagai berikut:

1. Alat ini mempunyai kemampuan penghitungan maksimal 8 digit.
2. Pada alat ini hanya mampu memroses *input* bilangan bulat tak bertanda, tetapi alat ini mampu memberikan *output* bilangan pecahan dan bilangan bulat bertanda.
3. Alat ini akan memberikan *output* suara setiap ada *input* dari *keypad* dan ketika ditekan tombol sama dengan ("=") maka akan mengeluarkan suara sesuai data *input* dari *keypad* secara keseluruhan sesuai dengan pembacaan urutan bilangan yang sebenarnya, demikian juga dengan hasil dari proses aritmatika.
4. Adanya error pada suara yaitu ketika ada *input/output* 1.000.000 suara yang keluar menjadi "satu juta ribu" bukan "satu juta". Demikian juga pada kelipatannya ada penambahan suara "ribu".
5. Penggunaan alat ini sangat mudah bagi tuna netra karena dilengkapi dengan *keypad Braille* yang bisa diraba dan diartikan langsung oleh tuna netra.

5.2. Saran

Dalam perancangan dan pembuatan kalkulator dengan *output* suara ini, masih didapatkan hal-hal yang bisa menjadikan fungsi alat lebih optimal dengan adanya penambahan atau pengembangan. Hal tersebut antara lain:

1. Menambah kemampuan penghitungan yang lebih besar dari 8 digit.
2. Menambah kemampuan mengolah bilangan pecahan dan bilangan bertanda.

DAFTAR PUSTAKA

[1] Atmel Corp. "*Datasheet AT89S8253*". (http://www.atmel.com)

[2] Budioko, Totok. 2005. *Belajar dengan Mudah dan Cepat Pemrograman Bahasa C dengan SDCC pada Mikrokontroler AT89x051/AT89251/52 Teori, Simulasi dan Aplikasi*. Jogjakarta: Gava Media

[3] Datasheet Archieve. "*Datasheet IC ISD 2590*". (http://datasheet_archieve.com)

[4] *Logika Aritmatika pada Sistem Biner*. (http://www.logika.co.id)

[5] Munir, Rinaldi. 2001, "*Algoritma dan Pemrograman dalam Bahasa Pascal dan C*". Bandung: Informatika

[6] Mytutorial. "*Tutorial Pelatihan Mikrokontroller*". (http://www.mytutorialcafe.com)

[7] *Operasi Aritmatika pada Mikrokontroler*. (www.innovative_electronic.com)

[8] Seiko Instrument Inc, *liquid Crystal Display* Modul M1632 *User Manual*, Japan: jan. 1987

[9] Tokheim, Roger L. 1990. "*Elektronika Digital*". Edisi Kedua. Jakarta: Erlangga

[10] US Braille. "*US Computer Braille*". (www.us_braille.com)

LAMPIRAN

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

LEMBAR PERBAIKAN SKRIPSI

Nama : Mukhlish Andi Kristiyawan
Nim : 02.17.043
Jurusan : Teknik Elektro S1
Konsentrasi : Teknik Elektronika
Judul : " PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA"

Hari/Tgl Skripsi : Sabtu, 15 Maret 2008

No.	Materi Perbaikan	Paraf
1.	Ada error suara untuk menyebutkan 1.000.000, suara yang keluar "satu juta ribu" bukan "satu juta"	
2.	Siapkan jawaban untuk proses konversi dari tulisan ke suara	

Mengetahui

Dosen Pembimbing I

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y. 1018800189

Dosen Pembimbing II

Sotyohadi, ST.

Diperiksa / Disetujui

Penguji I

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

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

NAMA : MUKHLISH ANDI KRISTIYAWAN
NIM : 02.17.043
Masa Bimbingan : 23 Januari 2008 – 23 Juli 2008
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA.**

No.	Tanggal	Uraian	Paraf Pembimbing
1.	28 Januari 2008	Demo alat	
2.	30 Januari 2008	Revisi Bab I (Tujuan dan Rumusan Masalah)	
3.	1 Februari 2008	Revisi Bab II (Teori Huruf *Braille*)	
4.	4 Februari 2008	ACC Bab I dan II	
5.	4 Februari 2008	Revisi Bab III (*Flowchart*)	
6.	5 Februari 2008	ACC Bab III	
7.	6 Februari 2008	Revisi Bab IV (Tabel dan Gambar Pengujian)	
8.	11 Februari 2008	ACC Bab IV dan V	
9.	20 Februari 2008	ACC Seminar Hasil	
10.	13 Maret 2008	ACC Ujian Skripsi	

Malang, ...
Dosen Pembimbing I

Ir. Yusuf Ismail Nakhoda, MT.
NIP.Y. 1018800189

Form S- 4a

INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
JURUSAN TEKNIK ELEKTRO

FORMULIR BIMBINGAN SKRIPSI

NAMA : MUKHLISH ANDI KRISTIYAWAN
NIM : 02.17.043
Masa Bimbingan : 23 Januari 2008 – 23 Juli 2008
Judul Skripsi : **PERANCANGAN DAN PEMBUATAN KALKULATOR DENGAN *OUTPUT* LCD DAN *SPEAKER* SEBAGAI ALAT HITUNG ELEKTRONIK BAGI TUNA NETRA.**

No.	Tanggal	Uraian	Paraf Pembimbing
1.	7 Januari 2008	Konsultasi Program (*Output* Pecahan)	
2.	12 Januari 2008	Konsultasi Teori Aritmatika Biner	
3.	14 Januari 2008	Revisi Bab II (Teori Aritmatika Biner)	
4.	18 Januari 2008	ACC Bab II	
5.	21 Januari 2008	Revisi Bab III (*Flowchart*)	
6.	24 Januari 2008	ACC Bab III	
7.	28 Januari 2008	Demo Alat dan Revisi Bab IV (Tabel dan Gambar Pengujian)	
8.	11 Februari 2008	ACC Bab IV dan V	
9.	18 Februari 2008	ACC Seminar Hasil	

Malang, ...
Dosen Pembimbing II

21/4 '08

Sotyohadi, ST.

Form S- 4a

Listing Program

```
#include     <at89s8253.h>      //P0 untuk data LCD  P2_6, P2_7 untuk LCD
#include     "lcd.h"            //rs P3.0, en P3.1, data P0
#include     "isd.h"            //ce P2.6, stop P2.7, data P0

#define      baris1  P1_0
#define      baris2  P1_1
#define      baris3  P1_2
#define      baris4  P1_3

unsigned char keydata = 0xFF;
unsigned char tekan1 = 0;
unsigned char tekan2 = 0;
unsigned char hit = 0;
unsigned char a1,aa,ab,ac,ad,ae,af,ag,ah,ai,aj,d1,d2,za,zb;
unsigned long rest;
float data1;
float data2;
float hasil;
unsigned long olah;
char faktor = 0;
char bilang = 0;
char bite = 0;

void tampilangka(char lokasi,unsigned long nilai)
{
        unsigned char angka;
        hit = 0;
        LCDdata(0,0x04);
        LCDdata(0,lokasi);
        while(nilai > 9)
        {
                angka = nilai % 10;
                LCDdata(1,angka + 0x30);
                nilai = nilai / 10;
                hit++;
        }
        LCDdata(1,nilai + 0x30);
        LCDdata(0,0x06);hit++;
}

void tampilangka1(char lokasi,unsigned long nilai)
{
        unsigned char koma = 0;
        unsigned char angka;
        hit = 0;
        LCDdata(0,0x04);
        LCDdata(0,lokasi);
        while(nilai > 9)
        {
                angka = nilai % 10;
                LCDdata(1,angka + 0x30);
```

```
            nilai = nilai / 10;
            while ( koma == 1 )
            {
                  LCDdata(1,0x2E);
                  koma++;
            }
            koma++;
            hit++;
      }
      LCDdata(1,nilai + 0x30);
      koma++;
      hit++;
      while ( koma == 1)
      {     LCDdata(1,0x30);
            koma++;
            hit++;
      }
      delay(100);
      while( koma == 2 )
            {
                  LCDdata(1,0x2E);
                  koma++;
                  LCDdata(1,0x30);
                  hit++;
            }
      //LCDdata(1,0x20);
      LCDdata(0,0x06); //auto incremant
      hit = hit - 2;
}

void tampilangka2(char lokasi,unsigned long nilai)
{
      unsigned char angka;
      hit = 0;
      LCDdata(0,0x04);     //auto decremant
      LCDdata(0,lokasi);
      while(nilai > 9)
      {
            angka = nilai % 10;
            LCDdata(1,angka + 0x30);
            nilai = nilai / 10;
            hit++;
      }
      LCDdata(1,nilai + 0x30);
      LCDdata(1,0x2D);
      LCDdata(0,0x06);hit++;
}

void keypad()
{
      P1 = 0xEF;
      if(!baris1)keydata = 1;
```

```
        if(!baris2)keydata = 4;
        if(!baris3)keydata = 7;
        if(!baris4)keydata = 15;        //clear
        P1 = 0xDF;
        if(!baris1)keydata = 2;
        if(!baris2)keydata = 5;
        if(!baris3)keydata = 8;
        if(!baris4)keydata = 0;
        P1 = 0xBF;
        if(!baris1)keydata = 3;
        if(!baris2)keydata = 6;
        if(!baris3)keydata = 9;
        if(!baris4)keydata = 14;        //sama dengan
        P1 = 0x7F;
        if(!baris1)keydata = 10;        //kali
        if(!baris2)keydata = 11;        //bagi
        if(!baris3)keydata = 12;        //tambah
        if(!baris4)keydata = 13;        //kurang
}

void say1()
{
        if(a1 == 0)nol();
        if(a1 == 1)satu();
        if(a1 == 2)dua();
        if(a1 == 3)tiga();
        if(a1 == 4)empat();
        if(a1 == 5)lima();
        if(a1 == 6)enam();
        if(a1 == 7)tujuh();
        if(a1 == 8)delapan();
        if(a1 == 9)sembilan();
}

void satuan()
{
        if(aa == 1)satu();
        if(aa == 2)dua();
        if(aa == 3)tiga();
        if(aa == 4)empat();
        if(aa == 5)lima();
        if(aa == 6)enam();
        if(aa == 7)tujuh();
        if(aa == 8)delapan();
        if(aa == 9)sembilan();
}

void puluhan()
{
        if ( ab == 1 && aa == 0 ){se();puluh();}
        if ( ab == 1 && aa == 1 ){se();belas();}
        if ( ab == 1 && aa == 2 ){dua();belas();}
```

```
        if ( ab == 1 && aa == 3 ){tiga();belas();}
        if ( ab == 1 && aa == 4 ){empat();belas();}
        if ( ab == 1 && aa == 5 ){lima();belas();}
        if ( ab == 1 && aa == 6 ){enam();belas();}
        if ( ab == 1 && aa == 7 ){tujuh();belas();}
        if ( ab == 1 && aa == 8 ){delapan();belas();}
        if ( ab == 1 && aa == 9 ){sembilan();belas();}
        if ( ab == 0 ){satuan();}
        if ( ab == 2 ){dua();puluh();satuan();}
        if ( ab == 3 ){tiga();puluh();satuan();}
        if ( ab == 4 ){empat();puluh();satuan();}
        if ( ab == 5 ){lima();puluh();satuan();}
        if ( ab == 6 ){enam();puluh();satuan();}
        if ( ab == 7 ){tujuh();puluh();satuan();}
        if ( ab == 8 ){delapan();puluh();satuan();}
        if ( ab == 9 ){sembilan();puluh();satuan();}
}

void ratusan()
{
        if ( ac == 0 ) {puluhan();}
        if ( ac == 1){se();ratus();puluhan();}
        if ( ac == 2){dua();ratus();puluhan();}
        if ( ac == 3){tiga();ratus();puluhan();}
        if ( ac == 4){empat();ratus();puluhan();}
        if ( ac == 5){lima();ratus();puluhan();}
        if ( ac == 6){enam();ratus();puluhan();}
        if ( ac == 7){tujuh();ratus();puluhan();}
        if ( ac == 8){delapan();ratus();puluhan();}
        if ( ac == 9){sembilan();ratus();puluhan();}
}

void ribuan()
{
        if ( ad == 1 ){se();ribu();ratusan();}
        if ( ad == 2 ){dua();ribu();ratusan();}
        if ( ad == 3 ){tiga();ribu();ratusan();}
        if ( ad == 4 ){empat();ribu();ratusan();}
        if ( ad == 5 ){lima();ribu();ratusan();}
        if ( ad == 6 ){enam();ribu();ratusan();}
        if ( ad == 7 ){tujuh();ribu();ratusan();}
        if ( ad == 8 ){delapan();ribu();ratusan();}
        if ( ad == 9 ){sembilan();ribu();ratusan();}
}

//void prsatuan()
{
        if(ad == 1)satu();
        if(ad == 2)dua();
        if(ad == 3)tiga();
        if(ad == 4)empat();
        if(ad == 5)lima();
```

```
        if(ad == 6)enam();
        if(ad == 7)tujuh();
        if(ad == 8)delapan();
        if(ad == 9)sembilan();
}//

void pribu()
{
        if ( ae == 0 && ad == 0 ){ribu();ratusan();}
        if ( ae == 1 && ad == 0 ){se();puluh();ribu();ratusan();}
        if ( ae == 1 && ad == 1 ){se();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 2 ){dua();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 3 ){tiga();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 4 ){empat();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 5 ){lima();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 6 ){enam();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 7 ){tujuh();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 8 ){delapan();belas();ribu();ratusan();}
        if ( ae == 1 && ad == 9 ){sembilan();belas();ribu();ratusan();}
        if ( ae == 2 ){dua();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 3 ){tiga();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 4 ){empat();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 5 ){lima();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 6 ){enam();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 7 ){tujuh();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 8 ){delapan();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 9 ){sembilan();puluh();prsatuan();ribu();ratusan();}
        if ( ae == 0 && ad != 0 ){ribuan();}
}

void rribu()
{
        if ( af == 0 ){pribu();}
        if ( af == 1 ){se();ratus();pribu();}
        if ( af == 2 ){dua();ratus();pribu();}
        if ( af == 3 ){tiga();ratus();pribu();}
        if ( af == 4 ){empat();ratus();pribu();}
        if ( af == 5 ){lima();ratus();pribu();}
        if ( af == 6 ){enam();ratus();pribu();}
        if ( af == 7 ){tujuh();ratus();pribu();}
        if ( af == 8 ){delapan();ratus();pribu();}
        if ( af == 9 ){sembilan();ratus();pribu();}
}

void jutaan()
{
        if ( ag == 1 ){satu();juta();rribu();}
        if ( ag == 2 ){dua();juta();rribu();}
        if ( ag == 3 ){tiga();juta();rribu();}
        if ( ag == 4 ){empat();juta();rribu();}
        if ( ag == 5 ){lima();juta();rribu();}
        if ( ag == 6 ){enam();juta();rribu();}
```

```
        if( ai == 7 ){tujuh();ratus();pjuta();}
        if( ai == 8 ){delapan();ratus();pjuta();}
        if( ai == 9 ){sembilan();ratus();pjuta();}
}

void milyaran()
{
        if ( aj == 1 ){satu();milyar();rjuta();}
        if ( aj == 2 ){dua();milyar();rjuta();}
        if ( aj == 3 ){tiga();milyar();rjuta();}
        if ( aj == 4 ){empat();milyar();rjuta();}
        if ( aj == 5 ){lima();milyar();rjuta();}
        if ( aj == 6 ){enam();milyar();rjuta();}
        if ( aj == 7 ){tujuh();milyar();rjuta();}
        if ( aj == 8 ){delapan();milyar();rjuta();}
        if ( aj == 9 ){sembilan();milyar();rjuta();}
}

void says()
{
        if ( hit == 1 )
                {a1 = rest;say1();}
        if ( hit == 2 )
                {aa = rest % 10;rest = rest / 10;
                ab = rest;puluhan();}
        if ( hit == 3 )
                {aa = rest % 10;rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest; ratusan();}
        if ( hit == 4 )
                {aa = rest % 10;rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest % 10;rest = rest / 10;
                ad = rest; ribuan();}
        if ( hit == 5 )
                {aa = rest % 10; rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest % 10;rest = rest / 10;
                ad = rest % 10; rest = rest / 10; ae = rest; pribu();}
        if ( hit == 6 )
                {aa = rest % 10; rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest % 10; rest = rest / 10;
                ad = rest % 10; rest = rest / 10; ae = rest % 10;
                rest = rest / 10; af = rest; rribu();}
        if ( hit == 7 )
                {aa = rest % 10; rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest % 10; rest = rest / 10;
                ad = rest % 10; rest = rest / 10; ae = rest % 10;
                rest = rest / 10; af = rest % 10; rest = rest / 10;
                ag = rest; jutaan();}
        if ( hit == 8 )
                {aa = rest % 10; rest = rest / 10; ab = rest % 10;
                rest = rest / 10; ac = rest % 10; rest = rest / 10;
                ad = rest % 10; rest = rest / 10; ae = rest % 10;
                rest = rest / 10; af = rest % 10; rest = rest / 10;
```

```
            ag = rest % 10; rest = rest / 10; ah = rest; pjuta();}
        if ( hit == 9 )
            {aa = rest % 10; rest = rest / 10; ab = rest % 10;
            rest = rest / 10; ac = rest % 10; rest = rest / 10;
            ad = rest % 10; rest = rest / 10; ae = rest % 10;
            rest = rest / 10; af = rest % 10; rest = rest / 10;
            ag = rest % 10; rest = rest / 10; ah = rest % 10;
            rest = rest / 10; ai = rest; rjuta();}
        if ( hit == 10 )
            {aa = rest % 10; rest = rest / 10; ab = rest % 10;
            rest = rest / 10; ac = rest % 10; rest = rest / 10;
            ad = rest % 10; rest = rest / 10; ae = rest % 10;
            rest = rest / 10; af = rest % 10; rest = rest / 10;
            ag = rest % 10; rest = rest / 10; ah = rest % 10;
            rest = rest / 10; ai = rest % 10; rest = rest / 10;
            aj = rest; milyaran();}
}

void memdata1()
{
        if ( bilang == 0 && bite < 8)
        {
            if ( tekan1 == 0 )
            {
                data1 = keydata;a1 = keydata;
                tampilangka(0x80 + bite,data1);
                tekan1++;bite++;say1();
            }
            else
            {
                data1 = data1 * 10;
                data1 = data1 + keydata;a1 = keydata;
                tampilangka(0x80 + bite,data1);
                tekan1++;bite++;say1();
            }
        }
        else
        {
            delay(20000);
            TulisLCD(0xC0,"  Data Error  ");
            dataerror();
        }
}

void memdata2()
{
        if ( bilang == 1 && bite < 8 )
        {
            if ( tekan2 == 0 )
            {
                data2 = keydata;a1 = keydata;
                tampilangka(0x80 + bite ,data2);
```

```
                tekan2++;bite++;say1();
            }
            else
            {
                data2 = data2 * 10;
                data2 = data2 + keydata;a1 = keydata;
                tampilangka(0x80 + bite ,data2);
                tekan2++;bite++;say1();
            }
        }
        else
        {
            delay(20000);
            TulisLCD(0xC0,"  Data Error  ");
            dataerror();
        }
}

void check()
{
        if (hasil >= 100000000)
        {
            delay(100000);
            TulisLCD(0xC0,"  Data Error  ");
            dataerror();
        }
        else
        {
            tampilangka(0xCF,hasil);rest = hasil;says();
        }
}

void check1()
{
        if (hasil >= 100000000)
        {
            delay(100000);
            TulisLCD(0xC0,"  Data Error  ");
            dataerror();
        }
        else
        {
            tampilangka1(0xCF,hasil);hasil = olah;delay(100);rest = hasil;says();
        }
}

void check2()
{
        if (hasil >= 100000000)
        {
            delay(100000);
            TulisLCD(0xC0,"  Data Error  ");
```

```
                dataerror();
        }
        else
        {
                tampilangka2(0xCF,hasil);minus();delay(1000);rest = hasil;says();
        }
}

void says2()
{
        if (faktor == 10 ) {kali();}
        if (faktor == 11 ) {bagi();}
        if (faktor == 12 ) {tambah();}
        if (faktor == 13 ) {kurang();}
}

void says1()
{
        data1=d1;
        hit = d1;
        rest = data1;
        says();
        delay(100);
        says2();
        data2=d2;
        hit = d2;
        rest = data2;
        says();
        delay(1000);
        samadengan();
        delay(1000);
}

void keluaran()
{
        says1();
        if (faktor == 10)
        {
                hasil = data1 * data2;hit = 0;check();
        }
        if (faktor == 11)                    // Bagi
        {
                hasil = (data1 / data2) * 100; olah = hasil; za = olah % 10;
                olah = olah / 10; zb = olah % 10; olah = olah / 10;

                if ( za == 0 && zb == 0 )
                        {hasil = olah;check();}
                else
                {
                        check1();delay(100);delay(1000);saykoma();delay(1000);a1 = zb;
say1();delay(100);a1 = za; say1();
```

```
            }
        }
        if (faktor == 12)
        {
            hasil = data1 + data2;hit = 0;check();
        }
        if (faktor == 13)               // Kurang
        {
            if ( data1 >= data2 )
            {
                hasil = data1 - data2;hit = 0;check();
            }
            if ( data2 > data1 )
            {
                hasil = data2 - data1;hit = 0;check2();
            }
        }
}

void main()
{
        //char hit = 0x00;
        //P1 = 0xFF;
        initLCD();
        TulisLCD(0x80,"Mukhlish Andi K.");
        TulisLCD(0xC0,"   02.17.043   ");
        delay(40000);
        clear();
        while(1)
        {
            keypad();
            if( keydata != 0xFF )
            {
                if ( keydata <=9 )
                {
                    if (bilang == 0)
                    {
                        memdata1();
                    }
                    if (bilang == 1)
                    {
                        memdata2();
                    }
                }
                else
                {
                    if (keydata == 10)              //kali
                    {
                        faktor = 10;
                        TulisLCD(0x80 + bite,"X");
                        d1 = bite;bilang++;bite++;kali();
                    }
```

```
                    if (keydata == 11)              //bagi
                    {
                        faktor = 11;
                        TulisLCD(0x80 + bite,"/");
                        d1 = bite;bilang++;bite++;bagi();
                    }
                    if (keydata == 12)              //tambah
                    {
                        faktor = 12;
                        TulisLCD(0x80 + bite,"+");
                        d1 = bite;bilang++;bite++;tambah();
                    }
                    if (keydata == 13)              //kurang
                    {
                        faktor = 13;
                        TulisLCD(0x80 + bite,"-");
                        d1 = bite;bilang++;bite++;kurang();
                    }
                    if (keydata == 14)              //sama dengan
                    {
                        TulisLCD(0x80 + bite,"=");
                        keluaran();delay(25000);
                    }
                    if (keydata == 15)              // Clear
                    {
                        faktor = 0;
                        bilang = 0;
                        bite = 0;
                        keydata = 0xFF;
                        data1 = 0;
                        data2 = 0;
                        tekan1 = 0;
                        tekan2 = 0;
                        clear();
                    }
                }
                keydata = 0xFF;
                delay(1000);
            }
        }
}
```

ıtures

- ıpatible with MCS®-51 Products
- K Bytes of In-System Programmable (ISP) Flash Program Memory
  - SPI Serial Interface for Program Downloading
  - Endurance: 10,000 Write/Erase Cycles
- Bytes EEPROM Data Memory
  - Endurance: 100,000 Write/Erase Cycles
- byte User Signature Array
- V to 5.5V Operating Range
- lly Static Operation: 0 Hz to 24 MHz
- ree-level Program Memory Lock
- 5 x 8-bit Internal RAM
- Programmable I/O Lines
- ree 16-bit Timer/Counters
- ıe Interrupt Sources
- hanced UART Serial Port with Framing Error Detection and Automatic Address cognition
- hanced SPI (Double Write/Read Buffered) Serial Interface
- w-power Idle and Power-down Modes
- errupt Recovery from Power-down Mode
- ›grammable Watchdog Timer
- al Data Pointer
- wer-off Flag
- xible ISP Programming (Byte and Page Modes)
  - Page Mode: 64 Bytes/Page for Code Memory, 32 Bytes/Page for Data Memory
- ur-level Enhanced Interrupt Controller
- ›grammable and Fuseable x2 Clock Option
- ernal Power-on Reset
- pin PDIP Package Option for Reduced EMC Emission

ɔcription

AT89S8253 is a low-power, high-performance CMOS 8-bit microcontroller with bytes of In-System Programmable (ISP) Flash program memory and 2K bytes of ROM data memory. The device is manufactured using Atmel's high-density non- ile memory technology and is compatible with the industry-standard MCS-51 ıction set and pinout. The on-chip downloadable Flash allows the program mem- › be reprogrammed in-system through an SPI serial interface or by a conventional olatile memory programmer. By combining a versatile 8-bit CPU with download- Flash on a monolithic chip, the Atmel AT89S8253 is a powerful microcontroller ı provides a highly-flexible and cost-effective solution to many embedded control cations.

AT89S8253 provides the following standard features: 12K bytes of In-System Pro- ımable Flash, 2K bytes of EEPROM, 256 bytes of RAM, 32 I/O lines, ammable watchdog timer, two data pointers, three 16-bit timer/counters, a six- ır, four-level interrupt architecture, a full duplex serial port, on-chip oscillator, and circuitry. In addition, the AT89S8253 is designed with static logic for operation ı to zero frequency and supports two software selectable power saving modes. dle Mode stops the CPU while allowing the RAM, timer/counters, serial port, and upt system to continue functioning. The Power-down mode saves the RAM con- but freezes the oscillator, disabling all other chip functions until the next external upt or hardware reset.

›n-board Flash/EEPROM is accessible through the SPI serial interface. Holding ET active forces the SPI bus into a serial programming interface and allows the am memory to be written to or read from, unless one or more lock bits have been ıted.

8-bit Microcontroller with 12K Bytes Flash and 2K Bytes EEPROM

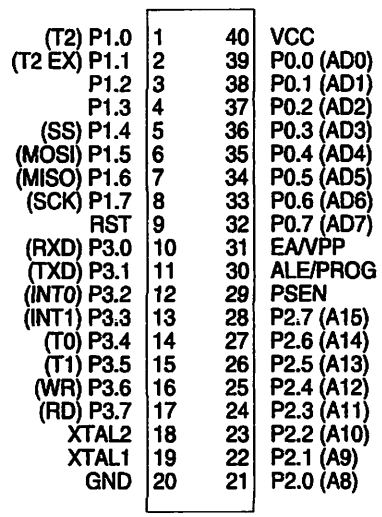
AT89S8253

Preliminary

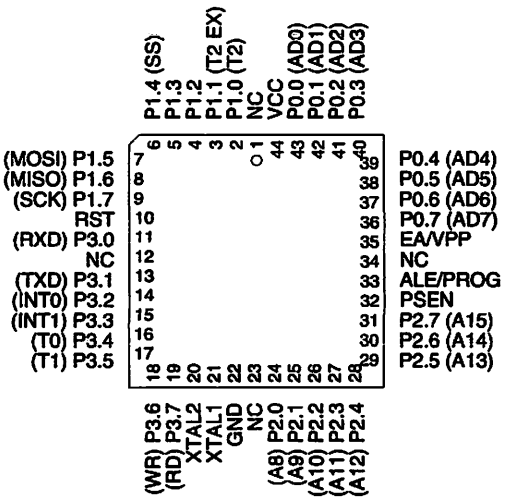
Rev. 3286A–MICRO–7/04

Configurations

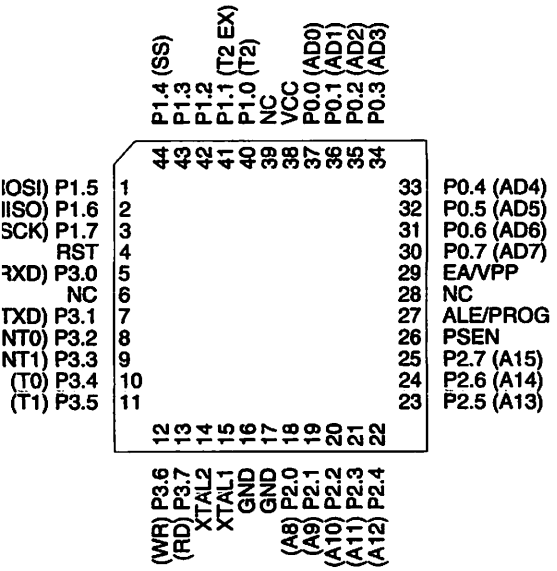
PDIP



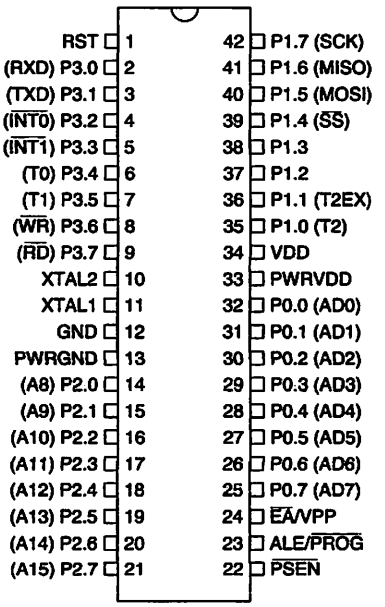
PLCC



TQFP



PDIP



Description

C

Supply voltage (all packages except 42-PDIP).

D

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

)

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

RVDD

Supply voltage for the 42-PDIP which connects only the I/O Pad Drivers.

The application board **must** connect both VDD and PWRVDD to the board supply voltage.

RGND

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

t 0

Port 0 is an 8-bit open drain bi-directional I/O port. As an output port, each pin can sink six 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.**

t 1

Port 1 is an 8-bit bi-directional I/O port with internal pull-ups. The Port 1 output buffers can sink/source six TTL inputs. When 1s are written to Port 1 pins, they are pulled high by the weak 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}, 150 µA typical) because of the weak internal pull-ups.

Some Port 1 pins provide additional functions. P1.0 and P1.1 can be configured to be the timer/counter 2 external count input (P1.0/T2) and the timer/counter 2 trigger input (P1.1/T2EX), respectively.

Furthermore, P1.4, P1.5, P1.6, and P1.7 can be configured as the SPI slave port select, data input/output and shift clock input/output pins as shown in the following table.

Port Pin	Alternate Functions
P1.0	T2 (external count input to Timer/Counter 2), clock-out
P1.1	T2EX (Timer/Counter 2 capture/reload trigger and direction control)
P1.4	$\overline{SS}$ (Slave port select input)
P1.5	MOSI (Master data output, slave data input pin for SPI channel)
P1.6	MISO (Master data input, slave data output pin for SPI channel)
P1.7	SCK (Master clock output, slave clock input pin for SPI channel)

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

ort 2

Port 2 is an 8-bit bi-directional I/O port with internal pull-ups. The Port 2 output buffers can sink/source six TTL inputs. When 1s are written to Port 2 pins, they are pulled high by the weak 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}, 150 µA typical) because of the weak 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.

ort 3

Port 3 is an 8-bit bi-directional I/O port with internal pull-ups. The Port 3 output buffers can sink/source six TTL inputs. When 1s are written to Port 3 pins, they are pulled high by the weak 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}, 150 µA typical) because of the weak internal 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 AT89S8253, 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)(1)
P3.3	$\overline{INT1}$ (external interrupt 1)(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)

Note: 1. All pins in ports 1 and 2 and almost all pins in port 3 (the exceptions are P3.2 $\overline{INT0}$ and P3.3 $\overline{INT1}$) have their weak internal pull-ups disabled in the Power-down mode. Port pins P3.2 ($\overline{INT0}$) and P3.1 ($\overline{INT1}$) are active even in Power-down mode (to be able to sense an interrupt request to exit the Power-down mode) and as such still have their weak internal pull-ups turned on.

ST

Reset input. A high on this pin for at least two machine cycles while the oscillator is running resets the device.

LE/$\overline{PROG}$

Address Latch Enable. ALE/$\overline{PROG}$ is an output pulse for latching the low byte of the address (on its falling edge) 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 the AUXR SFR at location 8EH. With the bit set, ALE is active only during a MOVX or MOVC instruction. Otherwise, the pin is weakly pulled high. Setting the ALE-disable bit has no effect if the microcontroller is in external execution mode.

SEN

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

When the AT89S8253 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.

A/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 when 12-volt programming is selected.

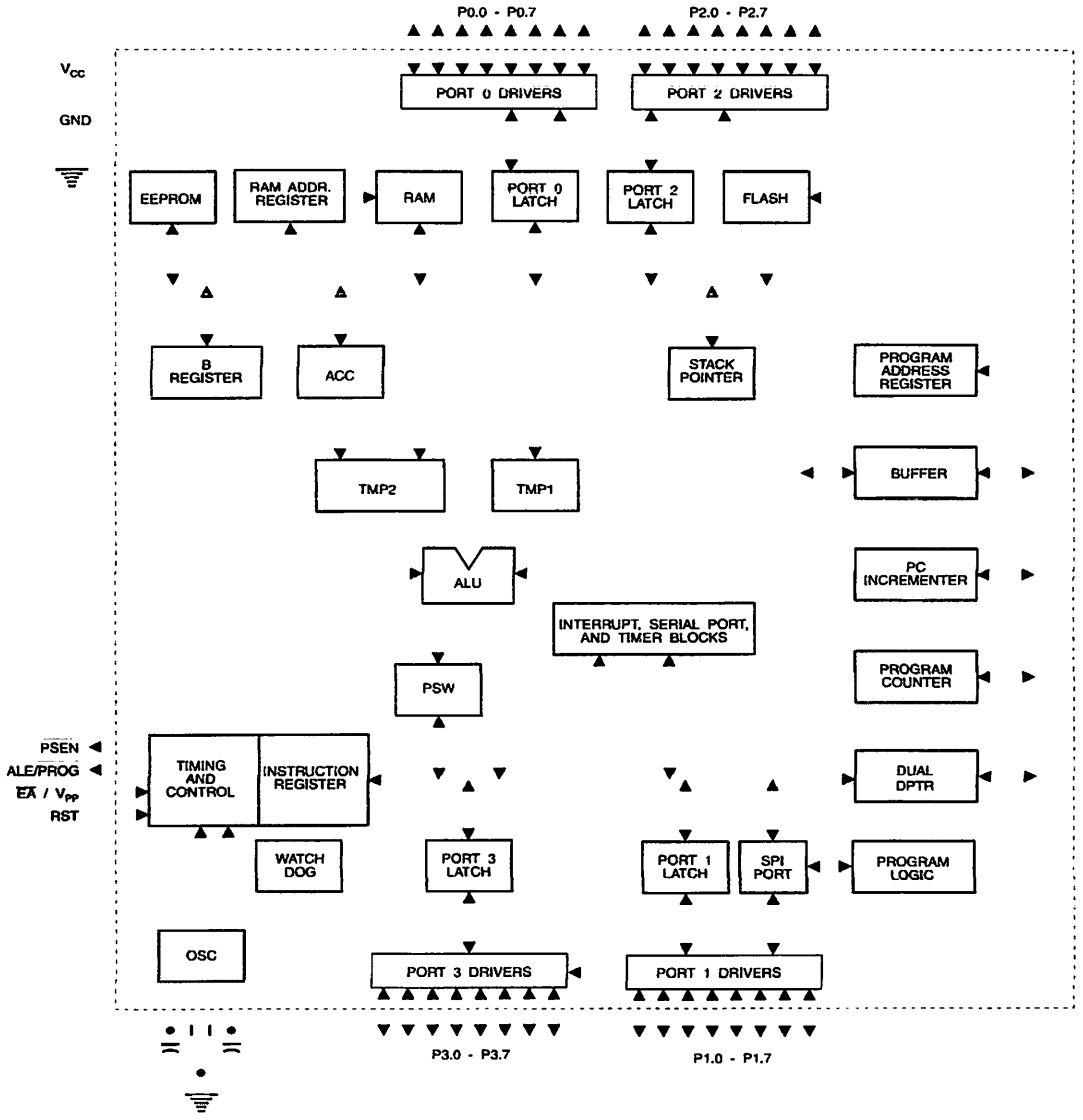
TAL1

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

TAL2

Output from the inverting oscillator amplifier.

lock Diagram



pecial unction 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 generally return random data, and write accesses will have an indeterminate effect.

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

ble 1. AT89S8253 SFR Map and Reset Values

0F8H								0FFH
0F0H	B 00000000							0F7H
0E8H								0EFH
0E0H	ACC 00000000							0E7H
0D8H								0DFH
0D0H	PSW 00000000					SPCR 00000100		0D7H
0C8H	T2CON 00000000	T2MOD XXXXXX00	RCAP2L 00000000	RCAP2H 00000000	TL2 00000000	TH2 00000000		0CFH
0C0H								0C7H
0B8H	IP XX000000	SADEN 00000000						0BFH
0B0H	P3 11111111							IPH XX000000 0B7H
0A8H	IE 0X000000	SADDR 00000000	SPSR 000XXX00					0AFH
0A0H	P2 11111111						WDTRST (Write Only)	WDTCON 0000 0000 0A7H
98H	SCON 00000000	SBUF XXXXXXXX						9FH
90H	P1 11111111						EECON XX000011	97H
88H	TCON 00000000	TMOD 00000000	TL0 00000000	TL1 00000000	TH0 00000000	TH1 00000000	AUXR XXXXXXX0	CLKREG XXXXXXX0 8FH
80H	P0 11111111	SP 00000111	DP0L 00000000	DP0H 00000000	DP1L 00000000	DP1H 00000000	SPDR ########	PCON 0XXX0000 87H

te: # means: 0 after cold reset and unchanged after warm reset.

uxiliary Register

The AUXR Register contains a single active bit called DISALE.

ble 2. AUXR – Auxiliary Register

UXR Address = 8EH — Reset Value = XXXX XXX0B

lot Bit Addressable

	–	–	–	–	–	–	–	DISALE
it	7	6	5	4	3	2	1	0

ymbol	Function
ISALE	When DISALE = 0, ALE is emitted at a constant rate of 1/6 the oscillator frequency (except during MOVX when 1 ALE pulse is missing). When DISALE = 1, ALE is active only during a MOVX or MOVC instruction.

lock Register

The CLKREG register contains a single active bit called X2.

ble 3. CLKREG – Clock Register

LKREG Address = 8FH — Reset Value = XXXX XXX0B

lot Bit Addressable

	–	–	–	–	–	–	–	X2
it	7	6	5	4	3	2	1	0

ymbol	Function
2	When X2 = 0, the oscillator frequency (at XTAL1 pin) is internally divided by 2 before it is used as the device system frequency. When X2 = 1, the divider by 2 is no longer used and the XTAL1 frequency becomes the device system frequency. This enables the user to choose a 6 MHz crystal instead of a 12 MHz crystal, for example, in order to reduce EMI.

PI Registers

Control and status bits for the Serial Peripheral Interface are contained in registers SPCR (see Table 11 on page 25) and SPSR (see Table 12 on page 26). The SPI data bits are contained in the SPDR register. In normal SPI mode, writing the SPI data register during serial data transfer sets the Write Collision bit (WCOL) in the SPSR register. In enhanced SPI mode, the SPDR is also write double-buffered because WCOL works as a Write Buffer Full Flag instead of being a collision flag. The values in SPDR are not changed by Reset.

terrupt Registers

The global interrupt enable bit and the individual interrupt enable bits are in the IE register. In addition, the individual interrupt enable bit for the SPI is in the SPCR register. Four priorities can be set for each of the six interrupt sources in the IP and IPH registers.

IPH bits have the same functions as IP bits, except IPH has higher priority than IP. By using IPH in conjunction with IP, a priority level of 0, 1, 2, or 3 may be set for each interrupt.

ual Data Pointer egisters

To facilitate accessing both internal EEPROM 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 WMCON 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.

›wer Off Flag

The Power Off Flag (POF), located at bit_4 (PCON.4) in the PCON SFR. POF, is set to "1" during power up. It can be set and reset under software control and is not affected by RESET.

ata Memory – EPROM and AM

The AT89S8253 implements 2K bytes of on-chip EEPROM for data storage and 256 bytes of RAM. The upper 128 bytes of RAM occupy a parallel space to the Special Function Registers. That means the upper 128 bytes have the same addresses as the SFR space but are physically separate from SFR space.

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

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

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

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

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

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

The on-chip EEPROM data memory is selected by setting the EEMEN bit in the EECON register at SFR address location 96H. The EEPROM address range is from 000H to 7FFH. MOVX instructions are used to access the EEPROM. To access off-chip data memory with the MOVX instructions, the EEMEN bit needs to be set to "0".

The EEMWE bit in the EECON register needs to be set to "1" before any byte location in the EEPROM can be written. User software should reset EEMWE bit to "0" if no further EEPROM write is required. EEPROM write cycles in the serial programming mode are self-timed and typically take 4 ms. The progress of EEPROM write can be monitored by reading the RDY/$\overline{BSY}$ bit (read-only) in SFR EECON. RDY/$\overline{BSY}$ = 0 means programming is still in progress and RDY/$\overline{BSY}$ = 1 means an EEPROM write cycle is completed and another write cycle can be initiated. Bit EELD in EECON controls whether the next MOVX instruction will only load the write buffer of the EEPROM or will actually start the programming cycle. By setting EELD, only load will occur. Before the last MOVX in a given page of 32 bytes, EELD should be cleared so that after the last MOVX the entire page will be programmed at the same time. This way, 32 bytes will only require 4 ms of programming time instead of 128 ms required in single byte programming.

In addition, during EEPROM programming, an attempted read from the EEPROM will fetch the byte being written with the MSB complemented. Once the write cycle is completed, true data are valid at all bit locations.

emory Control Register

ie EECON register contains control bits for the 2K bytes of on-chip data EEPROM. It also contains the control bit for the al data pointer.

ble 4. EECON – Data EEPROM Control Register

:ECON Address = 96H — Reset Value = XX00 0011B

lot Bit Addressable

Bit	–	–	EELD	EEMWE	EEMEN	DPS	RDY/$\overline{BSY}$	$\overline{WRTINH}$
	7	6	5	4	3	2	1	0

ymbol	Function
:ELD	EEPROM data memory load enable bit. Used to implement Page Mode Write. A MOVX instruction writing into the data EEPROM will not initiate the programming cycle if this bit is set, rather it will just load data into the volatile data buffer of the data EEPROM memory. Before the last MOVX, reset this bit and the data EEPROM will program all the bytes previously loaded on the same page of the address given by the last MOVX instruction.
:EMWE	EEPROM data memory write enable bit. Set this bit to 1 before initiating byte write to on-chip EEPROM with the MOVX instruction. User software should set this bit to 0 after EEPROM write is completed.
:EMEN	Internal EEPROM access enable. When EEMEN = 1, the MOVX instruction with DPTR will access on-chip EEPROM instead of external data memory if the address used is less than 2K. When EEMEN = 0 or the address used is ≥ 2K, MOVX with DPTR accesses external data memory.
)PS	Data pointer register select. DPS = 0 selects the first bank of data pointer register, DP0, and DPS = 1 selects the second bank, DP1.
:DY/$\overline{BSY}$	RDY/$\overline{BSY}$ (Ready/$\overline{Busy}$) flag for the data EEPROM memory. This is a read-only bit which is cleared by hardware during the programming cycle of the on-chip EEPROM. It is also set by hardware when the programming is completed. Note that RDY/$\overline{BSY}$ will be cleared long after the completion of the MOVX instruction which has initiated the programming cycle.
$\overline{VRTINH}$	$\overline{WRTINH}$ (Write Inhibit) is a READ-ONLY bit which is cleared by hardware when V_{cc} is too low for the programming cycle of the on-chip EEPROM to be executed. When this bit is cleared, an ongoing programming cycle will be aborted or a new programming cycle will not start.

rogrammable 'atchdog imer

The programmable Watchdog Timer (WDT) counts instruction cycles. The prescaler bits, PS0, PS1 and PS2 in SFR WDTCON are used to set the period of the Watchdog Timer from 16K to 2048K instruction cycles. The available timer periods are shown in Table 5 and the actual timer periods (at V_{CC} = 5V) are within ±30% of the nominal value.

The WDT is disabled by Power-on Reset and during Power-down mode. When WDT times out without being serviced or disabled, an internal RST pulse is generated to reset the CPU. See Table 5 for the WDT period selections.

Table 5. Watchdog Timer Time-out Period Selection

WDT Prescaler Bits			Period* (Nominal for F_{CLK} = 24 MHz)
PS2	PS1	PS0	
0	0	0	16 ms
0	0	1	32 ms
0	1	0	64 ms
0	1	1	128 ms
1	0	0	256 ms
1	0	1	512 ms
1	1	0	1024 ms
1	1	1	2048 ms

Note: *The WDT time-out period is dependent upon the external clock frequency.

atchdog Control egister

The WDTCON register contains control bits for the Watchdog Timer (shown in Table 6).

ble 6. WDTCON – Watchdog Control Register

VDTCON Address = A7H — Reset Value = 0000 0000B

lot Bit Addressable

	PS2	PS1	PS0	WDIDLE	DISRTO	HWDT	WSWRST	WDTEN
it	7	6	5	4	3	2	1	0

ymbol	Function
S2 S1 S0	Prescaler bits for the watchdog timer (WDT). When all three bits are cleared to 0, the watchdog timer has a nominal period of 16K machine cycles, (i.e. 16 ms at a XTAL frequency of 24 MHz in normal mode). When all three bits are set to 1, the nominal period is 2048K machine cycles, (i.e. 2048 ms at 24 MHz clock frequency).
VDIDLE	Disable/enable the Watchdog Timer in IDLE mode. When WDIDLE = 0, WDT continues to count in IDLE mode. When WDIDLE = 1, WDT freezes while the device is in IDLE mode.
ISRTO	Disable/enable the WDT-driven Reset Out (WDT drives the RST pin). When DISRTO = 0, the RST pin is driven high after WDT times out and the entire board is reset. When DISRTO = 1, the RST pin remains only as an input and the WDT resets only the microcontroller internally after WDT times out.
IWDT	Hardware mode select for the WDT. When HWDT = 0, the WDT can be turned on/off by simply setting or clearing WDTEN in the same register (this is the software mode for WDT). When HWDT = 1, the WDT has to be set by writing the sequence 1EH/E1H to the WDTRST register (with address 0A6H) and after being set in this way, WDT cannot be turned off except by reset, warm or cold (this is the hardware mode for WDT). To prevent the hardware WDT from resetting the entire device, the same sequence 1EH/E1H must be written to the same WDTRST SFR before the timeout interval.
VSWRST	Watchdog software reset bit. When HWDT = 0 (i.e. WDT is in software controlled mode), this bit resets WDT. After being set by software, WSWRST is reset by hardware during the next machine cycle. If HWDT = 1, this bit has no effect, and if set by software, it will not be cleared by hardware.
VDTEN	Watchdog software enable bit. When HWDT = 0 (i.e. WDT is in software-controlled mode), this bit enables WDT when set to 1 and disables WDT when cleared to 0 (it does not reset WDT in this case, but just freezes the existing counter state). If HWDT = 1, this bit is READ-ONLY and reflects the status of the WDT (whether it is running or not).

mer 0 and 1

Timer 0 and Timer 1 in the AT89S8253 operate the same way as Timer 0 and Timer 1 in the AT89S51 and AT89S52. For more detailed information on the Timers' operation, refer to the Atmel Web site (www.atmel.com). From the home page, select "Products," then "Microcontrollers", then "8051-Architecture", then "Documentation", and "Other Documents". Open the Adobe Acrobat file "AT89 Series Hardware Description."

mer 2

Timer 2 is a 16-bit Timer/Counter that can operate as either a timer or an event counter. The type of operation is selected by bit C/$\overline{T2}$ in the SFR T2CON (see Table 8 on page 14). Timer 2 has three operating modes: capture, auto-reload (up or down counting), and baud rate generator. The modes are selected by bits in T2CON, as shown in Table 8.

Timer 2 consists of two 8-bit registers, TH2 and TL2. In the Timer function, the TL2 register is incremented every machine cycle. Since a machine cycle consists of 12 oscillator periods, the count rate is 1/12 of the oscillator frequency.

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

Table 7. Timer 2 Operating Modes

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

ble 8. T2CON – Timer/Counter 2 Control Register

2CON Address = 0C8H — Reset Value = 0000 0000B

it Addressable

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

ymbol	Function
F2	Timer 2 overflow flag set by a Timer 2 overflow and must be cleared by software. TF2 will not be set when either RCLK = 1 or TCLK = 1.
XF2	Timer 2 external flag set when either a capture or reload is caused by a negative transition on T2EX and EXEN2 = 1. When Timer 2 interrupt is enabled, EXF2 = 1 will cause the CPU to vector to the Timer 2 interrupt routine. EXF2 must be cleared by software. EXF2 does not cause an interrupt in up/down counter mode (DCEN = 1).
CLK	Receive clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its receive clock in serial port Modes 1 and 3. RCLK = 0 causes Timer 1 overflows to be used for the receive clock.
CLK	Transmit clock enable. When set, causes the serial port to use Timer 2 overflow pulses for its transmit clock in serial port Modes 1 and 3. TCLK = 0 causes Timer 1 overflows to be used for the transmit clock.
XEN2	Timer 2 external enable. When set, allows a capture or reload to occur as a result of a negative transition on T2EX if Timer 2 is not being used to clock the serial port. EXEN2 = 0 causes Timer 2 to ignore events at T2EX.
R2	Start/Stop control for Timer 2. TR2 = 1 starts the timer.
/$\overline{T2}$	Timer or counter select for Timer 2. C/$\overline{T2}$ = 0 for timer function. C/$\overline{T2}$ = 1 for external event counter (falling edge triggered).
P/$\overline{RL2}$	Capture/Reload select. CP/$\overline{RL2}$ = 1 causes captures to occur on negative transitions at T2EX if EXEN2 = 1. CP/$\overline{RL2}$ = 0 causes automatic reloads to occur when Timer 2 overflows or negative transitions occur at T2EX when EXEN2 = 1. When either RCLK or TCLK = 1, this bit is ignored and the timer is forced to auto-reload on Timer 2 overflow.

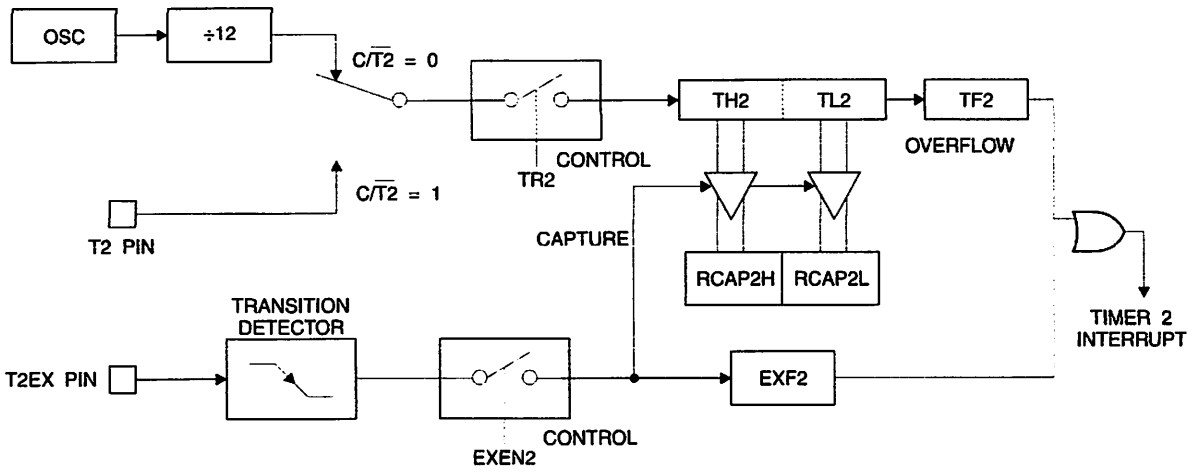
mer 2 Registers

Control and status bits are contained in registers T2CON (see Table 8) and T2MOD (see Table 9) for Timer 2. The register pair (RCAP2H, RCAP2L) are the Capture/Reload registers for Timer 2 in 16-bit capture mode or 16-bit auto-reload mode.

apture Mode

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

gure 1. Timer 2 in Capture Mode



uto-reload (Up or own Counter)

Timer 2 can be programmed to count up or down when configured in its 16-bit auto-reload mode. This feature is invoked by the DCEN (Down Counter Enable) bit located in the SFR T2MOD (see Table 9). Upon reset, the DCEN bit is set to 0 so that timer 2 will default to count up. When DCEN is set, Timer 2 can count up or down, depending on the value of the T2EX pin.

ble 9. T2MOD – Timer 2 Mode Control Register

'2MOD Address = 0C9H — Reset Value = XXXX XX00B

lot Bit Addressable

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

ymbol	Function
-	Not implemented, reserved for future use.
'2OE	Timer 2 Output Enable bit.
)CEN	When set, this bit allows Timer 2 to be configured as an up/down counter.

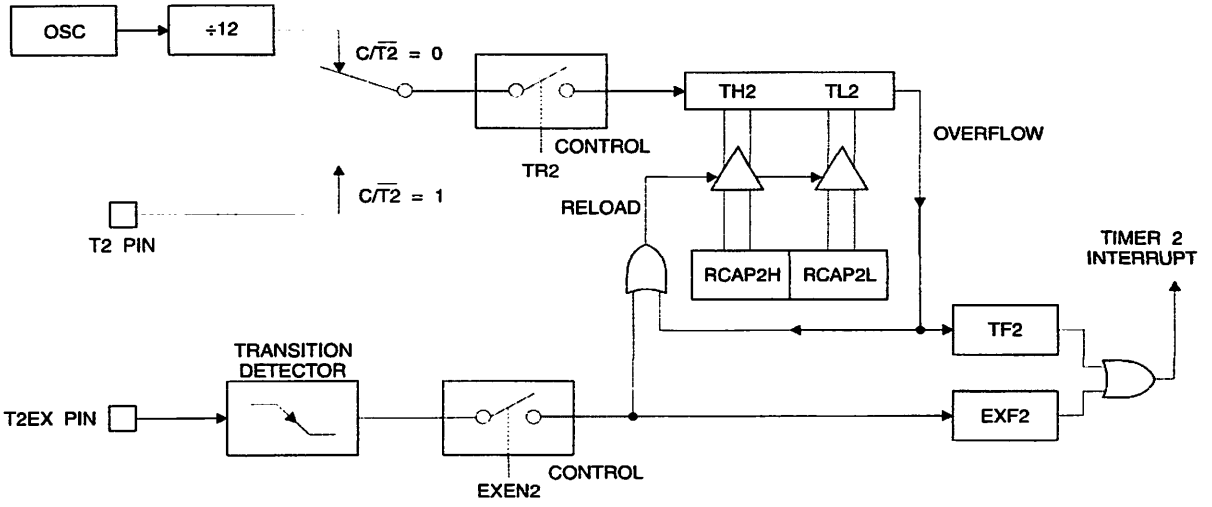
Figure 2 shows Timer 2 automatically counting up when DCEN = 0. In this mode, two options are selected by bit EXEN2 in T2CON. If EXEN2 = 0, Timer 2 counts up to 0FFFFH and then sets the TF2 bit upon overflow. The overflow also causes the timer registers to be reloaded with the 16-bit value in RCAP2H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

Setting the DCEN bit enables Timer 2 to count up or down, as shown in Figure 3. In this mode, the T2EX pin controls the direction of the count. A logic 1 at T2EX makes Timer 2 count up. The timer will overflow at 0FFFFH and set the TF2 bit. This overflow also causes the 16-bit value in RCAP2H and RCAP2L to be reloaded into the timer registers, TH2 and TL2, respectively.

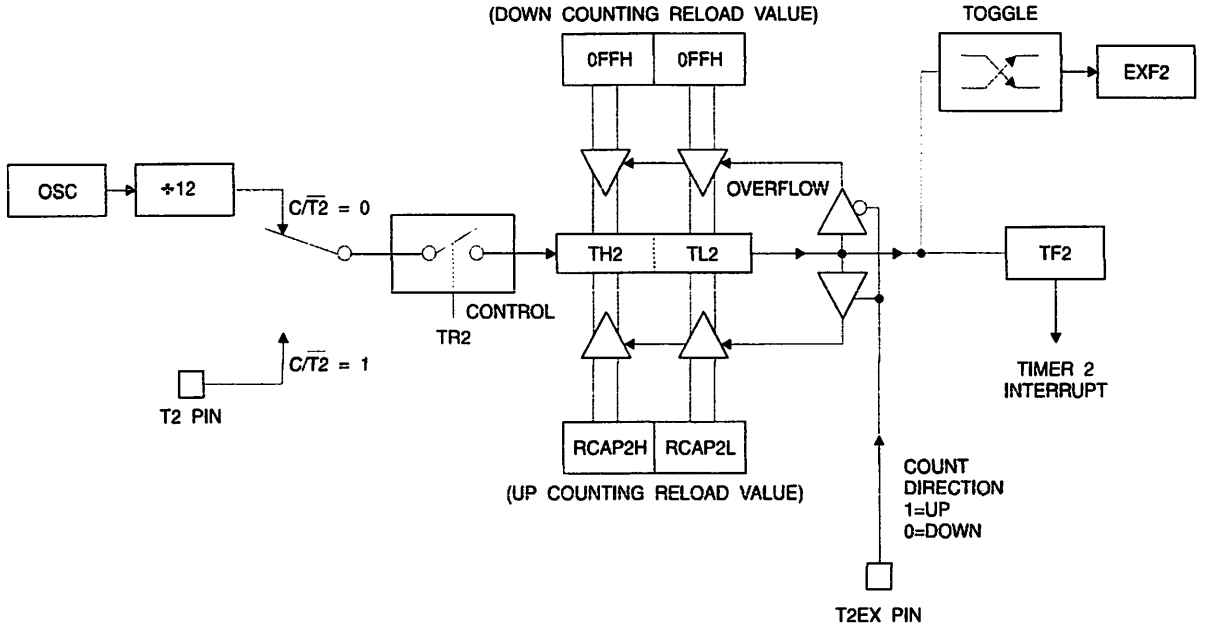
A logic 0 at T2EX makes Timer 2 count down. The timer underflows when TH2 and TL2 equal the values stored in RCAP2H and RCAP2L. The underflow sets the TF2 bit and causes 0FFFFH to be reloaded into the timer registers.

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

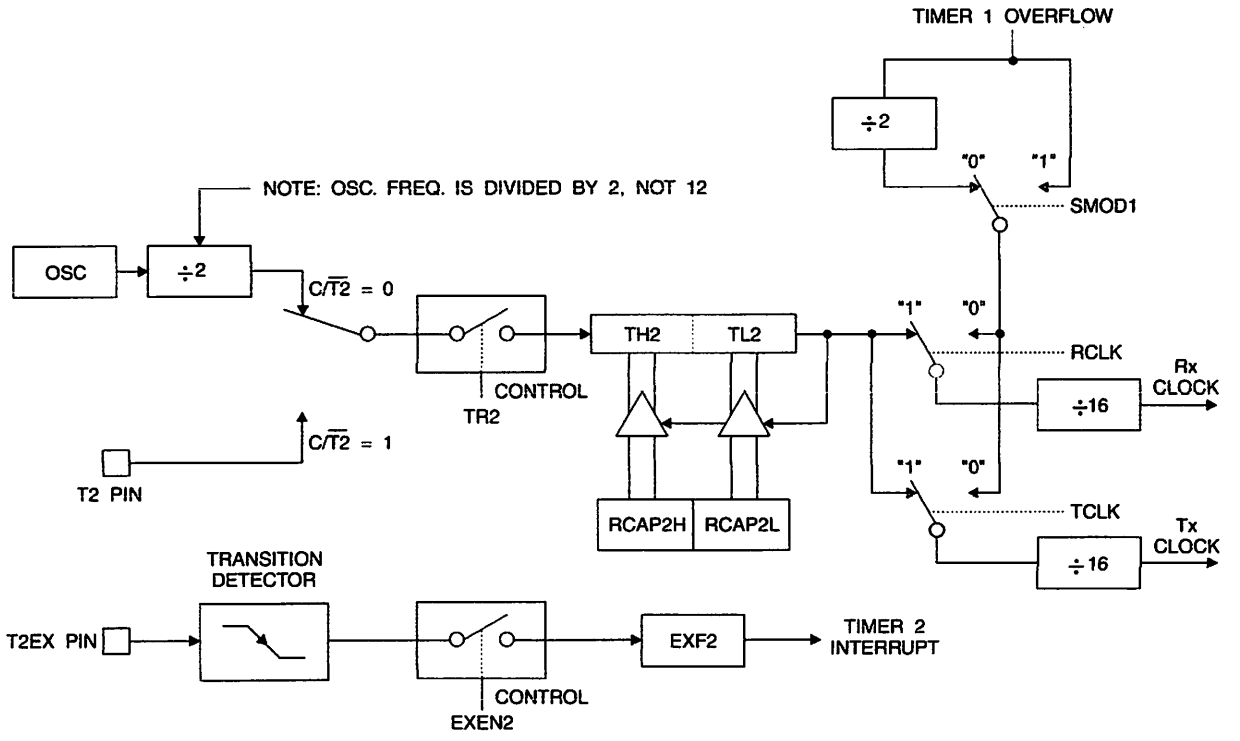
gure 2. Timer 2 in Auto Reload Mode (DCEN = 0)



gure 3. Timer 2 Auto Reload Mode (DCEN = 1 Timer 2 Auto Reload Mode (DCEN = 1)



gure 4. Timer 2 in Baud Rate Generator Mode



aud Rate enerator

Timer 2 is selected as the baud rate generator by setting TCLK and/or RCLK in T2CON (Table 8). Note that the baud rates for transmit and receive can be different if Timer 2 is used for the receiver or transmitter and Timer 1 is used for the other function. Setting RCLK and/or TCLK puts Timer 2 into its baud rate generator mode, as shown in Figure 4.

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

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

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

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

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

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

Timer 2 as a baud rate generator is shown in Figure 4. This figure is valid only if RCLK or TCLK = 1 in T2CON. Note that a rollover in TH2 does not set TF2 and will not generate an interrupt. Note too, that if EXEN2 is set, a 1-to-0 transition in T2EX will set EXF2 but will not cause a reload from (RCAP2H, RCAP2L) to (TH2, TL2). Thus when Timer 2 is in use as a baud rate generator, T2EX can be used as an extra external interrupt.

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

rogrammable lock Out

A 50% duty cycle clock can be programmed to come out on P1.0, as shown in Figure 5. This pin, besides being a regular I/O pin, has two alternate functions. It can be programmed to input the external clock for Timer/Counter 2 or to output a 50% duty cycle clock ranging from 61 Hz to 4 MHz (for a 16 MHz operating frequency).

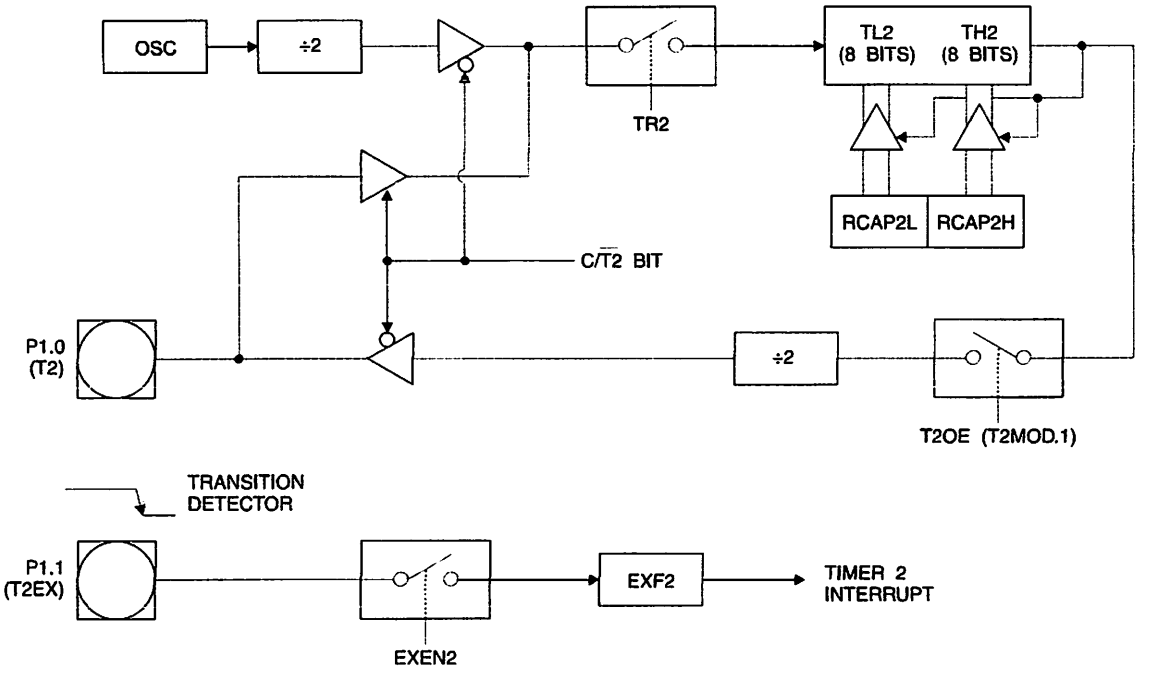
To configure the Timer/Counter 2 as a clock generator, bit C/$\overline{T2}$ (T2CON.1) must be cleared and bit T2OE (T2MOD.1) must be set. Bit TR2 (T2CON.2) starts and stops the timer.

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

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

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

gure 5. Timer 2 in Clock-out Mode



ART

The UART in the AT89S8253 operates the same way as the UART in the AT89S51 and AT89S52. For more detailed information on the UART operation refer to the Atmel Web site (www.atmel.com). From the home page, select "Products," then "Microcontrollers", then "8051-Architecture," then "Documentation", and "Other Documents". Open the Adobe Acrobat file "AT89 Series Hardware Description."

nhanced UART

In addition to all of its usual modes, the UART can perform framing error detection by looking for missing stop bits, and automatic address recognition. The UART also fully supports multi-processor communication as does the standard 80C51 UART.

When used for framing error detect, the UART looks for missing stop bits in the communication. A missing bit will set the FE bit in the SCON register. The FE bit shares the SCON.7 bit with SM0 and the function of SCON.7 is determined by PCON.6 (SMOD0). If SMOD0 is set then SCON.7 functions as FE. SCON.7 functions as SM0 when SMOD0 is cleared. When used as FE, SCON.7 can only be cleared by software.

ıtomatic Address ecognition

Automatic Address Recognition is a feature which allows the UART to recognize certain addresses in the serial bit stream by using hardware to make the comparisons. This feature saves a great deal of software overhead by eliminating the need for the software to examine every serial address which passes by the serial port. This feature is enabled by setting the SM2 bit in SCON. In the 9-bit UART modes, mode 2 and mode 3, the Receive Interrupt flag (RI) will be automatically set when the received byte contains either the "Given" address or the "Broadcast" address. The 9-bit mode requires that the 9th information bit is a 1 to indicate that the received information is an address and not data.

The 8-bit mode is called mode 1. In this mode the RI flag will be set if SM2 is enabled and the information received has a valid stop bit following the 8 address bits and the information is either a Given or Broadcast address.

Mode 0 is the Shift Register mode and SM2 is ignored.

Using the Automatic Address Recognition feature allows a master to selectively communicate with one or more slaves by invoking the given slave address or addresses. All of the slaves may be contacted by using the Broadcast address. Two special Function Registers are used to define the slave's address, SADDR, and the address mask, SADEN. SADEN is used to define which bits in the SADDR are to be used and which bits are "don't care". The SADEN mask can be logically ANDed with the SADDR to create the "Given" address which the master will use for addressing each of the slaves. Use of the Given address allows multiple slaves to be recognized while excluding others. The following examples will help to show the versatility of this scheme:

Slave 0 SADDR = 1100 0000
SADEN = 1111 1101
Given = 1100 00X0

Slave 1 SADDR = 1100 0000
SADEN = 1111 1110
Given = 1100 000X

In the previous example SADDR is the same and the SADEN data is used to differentiate between the two slaves. Slave 0 requires a 0 in bit 0 and it ignores bit 1. Slave 1 requires a 0 in bit 1 and bit 0 is ignored. A unique address for slave 0 would be 1100 0010 since slave 1 requires a 0 in bit 1. A unique address for slave 1 would be 1100 0001 since a 1 in bit 0 will exclude slave 0. Both slaves can be selected at the same time by an address which has bit 0 = 0 (for slave 0) and bit 1 = 0 (for slave 1). Thus, both could be addressed with 1100 0000.

In a more complex system the following could be used to select slaves 1 and 2 while excluding slave 0:

Slave 0 SADDR = 1100 0000
SADEN = 1111 1001
Given = 1100 0XX0

Slave 1 SADDR = 1110 0000
SADEN = 1111 1010
Given = 1110 0X0X

Slave 2 SADDR = 1110 0000
SADEN = 1111 1100
Given = 1110 00XX

In the above example the differentiation among the 3 slaves is in the lower 3 address bits. Slave 0 requires that bit 0 = 0 and it can be uniquely addressed by 1110 0110. Slave 1 requires that bit 1 = 0 and it can be uniquely addressed by 1110 and 0101. Slave 2 requires that bit 2 = 0 and its unique address is 1110 0011. To select Slaves 0 and 1 and exclude Slave 2, use address 1110 0100, since it is necessary to make bit 2 = 1 to exclude slave 2.

The Broadcast Address for each slave is created by taking the logical OR of SADDR and SADEN. Zeros in this result are trended as don't-cares. In most cases, interpreting the don't-cares as ones, the broadcast address will be FF hexadecimal.

Upon reset SADDR (SFR address 0A9H) and SADEN (SFR address 0B9H) are leaded with 0s. This produces a given address of all "don't cares" as well as a Broadcast address of all "don't cares". This effectively disables the Automatic Addressing mode and allows the microcontroller to use standard 80C51-type UART drivers which do not make use of this feature.

ıble 10. SCON – Serial Port Control Register

SCON Address = 98H — Reset Value = 0000 0000B

Bit Addressable

	SM0/FE	SM1	SM2	REN	TB8	RB8	T1	RI
Bit	7	6	5	4	3	2	1	0

(SMOD = 0/1)[1]

Symbol	Function
FE	Framing error bit. This bit is set by the receiver when an invalid stop bit is detected. The FE bit is not cleared by valid frames but should be cleared by software. The SMOD0 bit must be set to enable access to the FE bit. FE will be set regardless of the state of SMOD.
SM0	Serial Port Mode Bit 0, (SMOD must = 0 to access bit SM0)
SM1	Serial Port Mode Bit 1 SM0 / SM1 / Mode / Description / Baud Rate[2] 0 / 0 / 0 / shift register / $f_{osc}/12$ 0 / 1 / 1 / 8-bit UART / variable 1 / 0 / 2 / 9-bit UART / $f_{osc}/64$ or $f_{osc}/32$ 1 / 1 / 3 / 9-bit UART / variable
SM2	Enables the Automatic Address Recognition feature in modes 2 or 3. If SM2 = 1 then RI will not be set unless the received 9th data bit (RB8) is 1, indicating an address, and the received byte is a Given or Broadcast Address. In mode 1, if SM2 = 1 then RI will not be activated unless a valid stop bit was received, and the received byte is a Given or Broadcast Address. In Mode 0, SM2 should be 0.
REN	Enables serial reception. Set by software to enable reception. Clear by software to disable reception.
TB8	The 9th data bit that will be transmitted in modes 2 and 3. Set or clear by software as desired.
RB8	In modes 2 and 3, the 9th data bit that was received. In mode 1, if SM2 = 0, RB8 is the stop bit that was received. In mode 0, RB8 is not used.
TI	Transmit interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or at the beginning of the stop bit in the other modes, in any serial transmission. Must be cleared by software.
RI	Receive interrupt flag. Set by hardware at the end of the 8th bit time in mode 0, or halfway through the stop bit time in the other modes, in any serial reception (except see SM2). Must be cleared by software.

ɔtes: 1. SMOD is located at PCON.7.
2. f_{osc} = oscillator frequency.

erial eripheral nterface

The serial peripheral interface (SPI) allows high-speed synchronous data transfer between the AT89S8253 and peripheral devices or between multiple AT89S8253 devices. The AT89S8253 SPI features include the following:

- Full-Duplex, 3-Wire Synchronous Data Transfer
- Master or Slave Operation
- Maximum Bit Frequency = f/4 (f/2 if in x2 Clock Mode)
- LSB First or MSB First Data Transfer
- Four Programmable Bit Rates in Master Mode
- End of Transmission Interrupt Flag
- Write Collision Flag Protection
- Double-Buffered Receive
- Double-Buffered Transmit (Enhanced Mode only)
- Wakeup from Idle Mode (Slave Mode only)

The interconnection between master and slave CPUs with SPI is shown in Figure 6. The four pins in the interface are Master-In/Slave-Out (MISO), Master-Out/Slave-In (MOSI), Shift Clock (SCK), and Slave Select ($\overline{SS}$). The SCK pin is the clock output in master mode, but is the clock input in slave mode. The MSTR bit in SPCR determines the directions of MISO and MOSI. Also notice that MOSI connects to MOSI and MISO to MISO. In master mode, $\overline{SS}$/P1.4 is ignored and may be used as a general-purpose input or output. In slave mode, $\overline{SS}$ must be driven low to select an individual device as a slave. When $\overline{SS}$ is driven high, the slave's SPI port is deactivated and the MOSI/P1.5 pin can be used as a general-purpose input.

Figure 6. SPI Master-Slave Interconnection

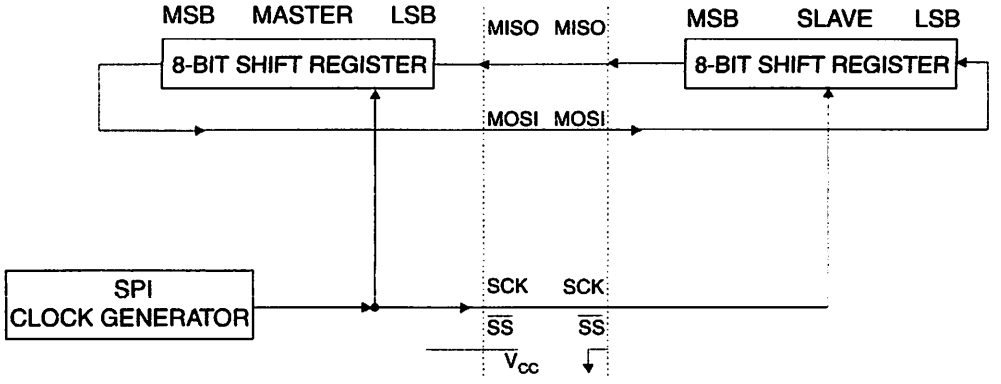
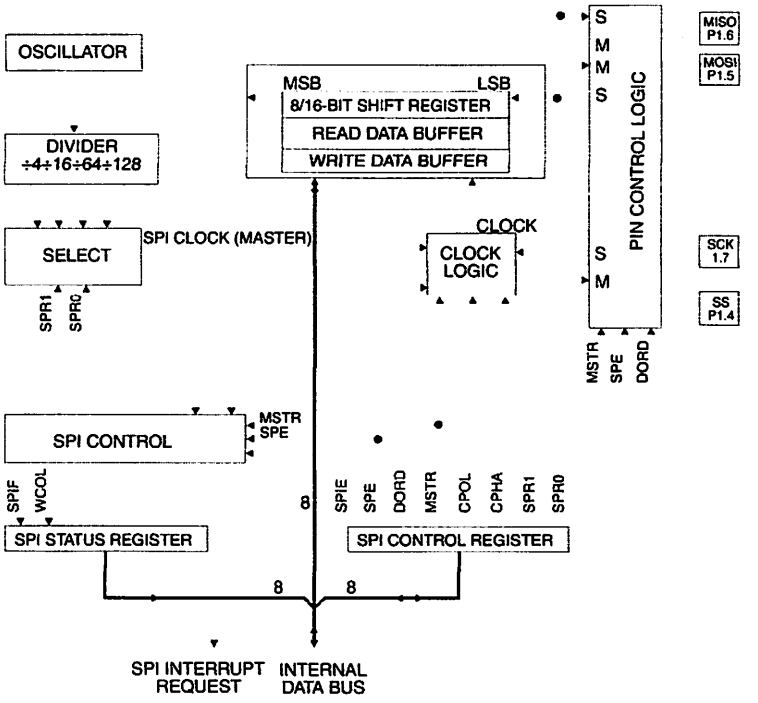


Figure 7. SPI Block Diagram



The SPI has two modes of operation: normal (non-buffered write) and enhanced (buffered write). In normal mode, writing to the SPI data register (SPDR) of the master CPU starts the SPI clock generator and the data written shifts out of the MOSI pin and into the MOSI pin of the slave CPU. Transmission may start after an initial delay while the clock generator waits for the next full bit slot of the specified baud rate. After shifting one byte, the SPI clock generator stops, setting the end of transmission flag (SPIF) and transferring the received byte to the read buffer (SPDR). If both the SPI interrupt enable bit (SPIE) and the serial port interrupt enable bit (ES) are set, an interrupt is requested. Note that SPDR refers to either the write data buffer or the read data buffer, depending on whether the access is a write or read. In normal mode, because the write buffer is transparent (and a write access to SPDR will be directed to the shift buffer), any attempt to write to SPDR while a transmission is in progress will result in a write collision with WCOL set. However, the transmission will still complete normally, but the new byte will be ignored and a new write access to SPDR will be necessary.

Enhanced mode is similar to normal mode except that the write buffer holds the next byte to be transmitted. Writing to SPDR loads the write buffer and sets WCOL to signify that the buffer is full and any further writes will overwrite the buffer. WCOL is cleared by hardware when the buffered byte is loaded into the shift register and transmission begins. If the master SPI is currently idle, i.e. if this is the first byte, then after loading SPDR, transmission of the byte starts and WCOL is cleared immediately. While this byte is transmitting, the next byte may be written to SPDR. The Load Enable flag (LDEN) in SPSR can be used to determine when transmission has started. LDEN is asserted during the first four bit slots of a SPI transfer. The master CPU should first check that LDEN is set and that WCOL is cleared before loading the next byte. In enhanced mode, if WCOL is set when a transfer completes, i.e. the next byte is available, then the SPI immediately loads the buffered byte into the shift register, resets WCOL, and continues transmission without stopping and restarting the clock generator. As long as the CPU can keep the write buffer full in this manner, multiple bytes may be transferred with minimal latency between bytes.

ible 11. SPCR – SPI Control Register

SPCR Address = D5H — Reset Value = 0000 0100B

Not Bit Addressable

Bit	SPIE	SPE	DORD	MSTR	CPOL	CPHA	SPR1	SPR0
	7	6	5	4	3	2	1	0

Symbol	Function
SPIE	SPI interrupt enable. This bit, in conjunction with the ES bit in the IE register, enables SPI interrupts: SPIE = 1 and ES = 1 enable SPI interrupts. SPIE = 0 disables SPI interrupts.
SPE	SPI enable. SPI = 1 enables the SPI channel and connects $\overline{SS}$, MOSI, MISO and SCK to pins P1.4, P1.5, P1.6, and P1.7. SPI = 0 disables the SPI channel.
DORD	Data order. DORD = 1 selects LSB first data transmission. DORD = 0 selects MSB first data transmission.
MSTR	Master/slave select. MSTR = 1 selects Master SPI mode. MSTR = 0 selects slave SPI mode.
CPOL	Clock polarity. When CPOL = 1, SCK is high when idle. When CPOL = 0, SCK of the master device is low when not transmitting. Please refer to figure on SPI clock phase and polarity control.
CPHA	Clock phase. The CPHA bit together with the CPOL bit controls the clock and data relationship between master and slave. Please refer to figure on SPI clock phase and polarity control.
SPR0 SPR1	SPI clock rate select. These two bits control the SCK rate of the device configured as master. SPR1 and SPR0 have no effect on the slave. The relationship between SCK and the oscillator frequency, $F_{osc.}$, is as follows: SPR1SPR0SCK = $F_{osc.}$ divided by: 0 0 4 0 1 16 1 0 64 1 1 128

Notes:
1. Set up the clock mode before enabling the SPI: set all bits needed in SPCR except the SPE bit, then set SPE.
2. Enable the master SPI prior to the slave device.
3. Slave echoes master on next Tx if not loaded with new data.

ıble 12. SPSR – SPI Status Register

SPSR Address = AAH — Reset Value = 000X XX00B

Not Bit Addressable

Bit	7	6	5	4	3	2	1	0
	SPIF	WCOL	LDEN	–	–	–	DISSO	ENH

Symbol	Function
SPIF	SP interrupt flag. When a serial transfer is complete, the SPIF bit is set and an interrupt is generated if SPIE = 1 and ES = 1. The SPIF bit is cleared by reading the SPI status register with SPIF and WCOL bits set, and then reading/writing the SPI data register.
WCOL	When ENH = 0: Write collision flag. The WCOL bit is set if the SPI data register is written during a data transfer. During data transfer, the result of reading the SPDR register may be incorrect, and writing to it has no effect. The WCOL bit (and the SPIF bit) are cleared by reading the SPI status register with SPIF and WCOL set, and then reading/writing the SPI data register. When ENH = 1: WCOL works in Enhanced mode as Tx Buffer Full. Writing during WCOL = 1 in enhanced mode will overwrite the waiting data already present in the Tx Buffer. In this mode, WCOL is no longer reset by the SPIF reset but *is reset when the write buffer has been unloaded into the serial shift register.*
LDEN	Load enable for the Tx buffer in enhanced SPI mode. When ENH is set, it is safe to load the Tx Buffer while LDEN = 1 and WCOL = 0. LDEN is high during bits 0 - 3 and is low during bits 4 - 7 of the SPI serial byte transmission time frame.
DISSO	Disable slave output bit. When set, this bit causes the MISO pin to be tri-stated so more than one slave device can share the same interface with a single master. Normally, the first byte in a transmission could be the slave address and only the selected slave should clear its DISSO bit.
ENH	Enhanced SPI mode select bit. When ENH = 0, SPI is in normal mode, i.e. without write double buffering. When ENH = 1, SPI is in enhanced mode with write double buffering. The Tx buffer shares the same address with the SPDR register.

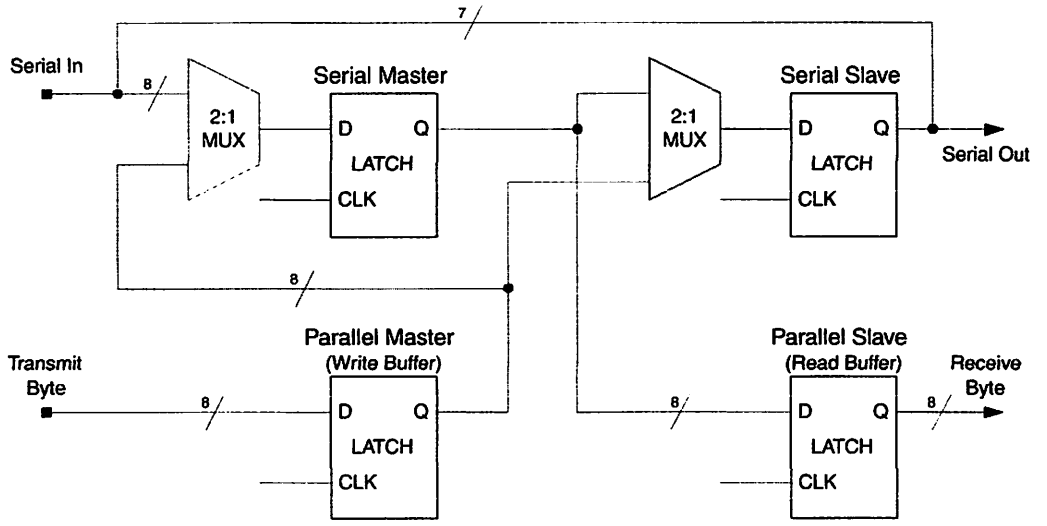
ıble 13. SPDR – SPI Data Register

SPDR Address = 86H — Reset Value = 00H (after cold reset)
unchanged (after warm reset)

Not Bit Addressable

Bit	7	6	5	4	3	2	1	0
	SPD7	*SPD6*	*SPD5*	*SPD4*	*SPD3*	*SPD2*	*SPD1*	*SPD0*

Figure 8. SPI Shift Register Diagram



The CPHA (Clock PHAse), CPOL (Clock POLarity), and SPR (Serial Peripheral clock Rate = baud rate) bits in SPCR control the shape and rate of SCK. The two SPR bits provide four possible clock rates when the SPI is in master mode. In slave mode, the SPI will operate at the rate of the incoming SCK as long as it does not exceed the maximum bit rate. There are also four possible combinations of SCK phase and polarity with respect to the serial data. CPHA and CPOL determine which format is used for transmission. The SPI data transfer formats are shown in Figures 9 and 10. To prevent glitches on SCK from disrupting the interface, CPHA, CPOL, and SPR should be set up before the interface is enabled, and the master device should be enabled before the slave device(s).

Table 14. SPI Master Characteristics

Symbol	Parameter	Min	Max	Units
t_{SCK}	Serial Clock Cycle Time	TBD	TBD	TBD
t_{SHSL}	Clock High Time	TBD	TBD	TBD
t_{SLSH}	Clock Low Time	TBD	TBD	TBD
t_{SR}	Rise Time	TBD	TBD	TBD
t_{SF}	Fall Time	TBD	TBD	TBD
t_{SIS}	Serial Input Setup Time	TBD	TBD	TBD
t_{SIH}	Serial Input Hold Time	TBD	TBD	TBD
t_{SOH}	Serial Output Hold Time	TBD	TBD	TBD
t_{SOV}	Serial Output Valid Time	TBD	TBD	TBD

Table 15. SPI Slave Characteristics

Symbol	Parameter	Min	Max	Units
t_{SCK}	Serial Clock Cycle Time	TBD	TBD	TBD
t_{SHSL}	Clock High Time	TBD	TBD	TBD
t_{SLSH}	Clock Low Time	TBD	TBD	TBD
t_{SR}	Rise Time	TBD	TBD	TBD
t_{SF}	Fall Time	TBD	TBD	TBD
t_{SIS}	Serial Input Setup Time	TBD	TBD	TBD
t_{SIH}	Serial Input Hold Time	TBD	TBD	TBD
t_{SOH}	Serial Output Hold Time	TBD	TBD	TBD
t_{SOV}	Serial Output Valid Time	TBD	TBD	TBD
t_{SOE}	Output Enable Time	TBD	TBD	TBD
t_{SOX}	Output Disable Time	TBD	TBD	TBD
t_{SSE}	Slave Enable Lead Time	TBD	TBD	TBD
t_{SSD}	Slave Disable Lag Time	TBD	TBD	TBD

Figure 9. SPI Master Timing (CPHA = 0)

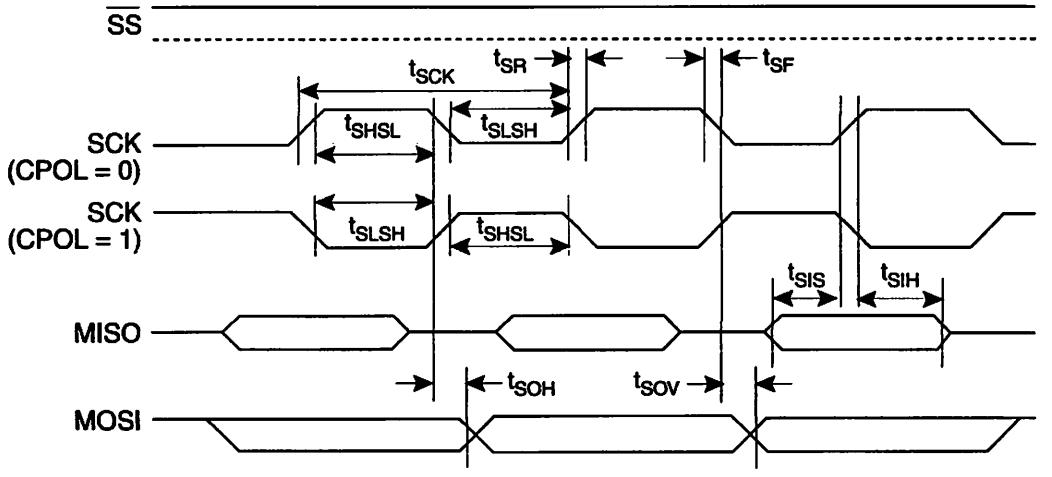


Figure 10. SPI Slave Timing (CPHA = 0)

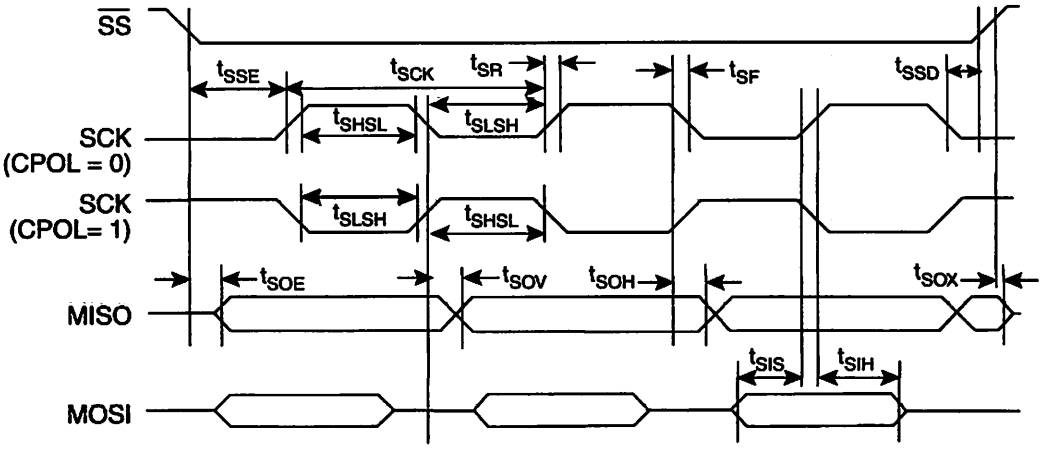


Figure 11. SPI Master Timing (CPHA = 1)

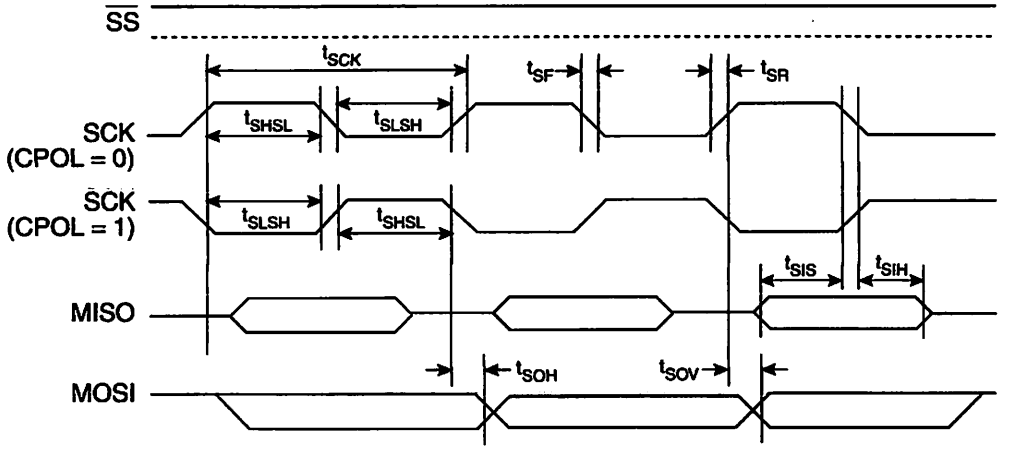
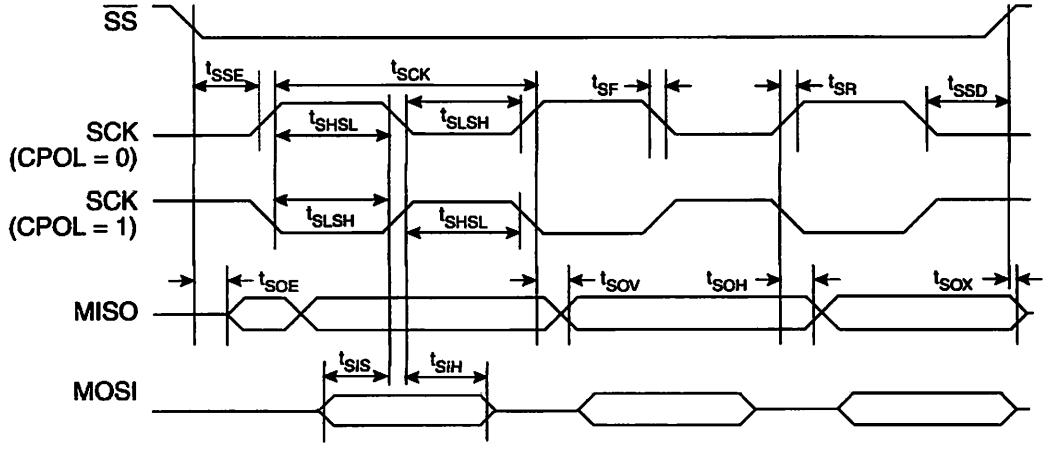
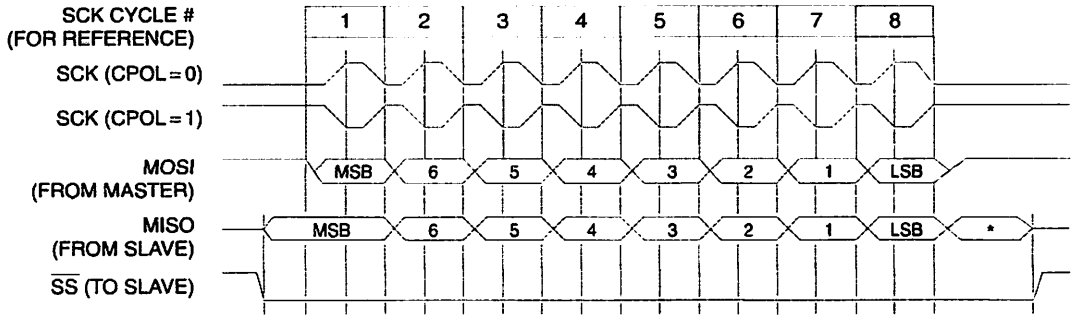


Figure 12. SPI Slave Timing (CPHA = 1)

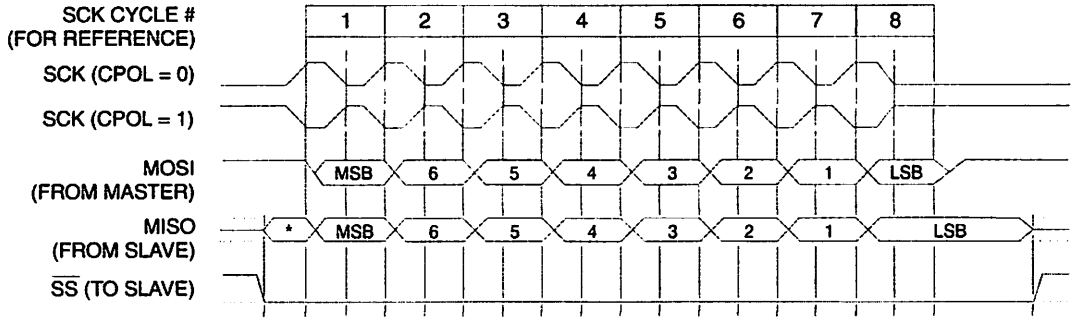


igure 13. SPI Transfer Format with CPHA = 0



ote: *Not defined but normally MSB of character just received

igure 14. SPI Transfer Format with CPHA = 1



ote: *Not defined but normally LSB of previously transmitted character

nterrupts

The AT89S8253 has a total of six interrupt vectors: two external interrupts ($\overline{INT0}$ and $\overline{INT1}$), three timer interrupts (Timers 0, 1, and 2), and the serial port interrupt. These interrupts are all shown in Figure 15.

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

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

The serial interrupt is the logical OR of bits RI and TI in register SCON and also bit SPIF in SPSR (if SPIE in SPCR is set). None of these flags is cleared by hardware when the service routine is vectored to. The service routine may have to determine whether the UART or SPI generated the interrupt.

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

ıble 16. Interrupt Enable (IE) Register

E Address = A8H — Reset Value = 0X00 0000B

3it Addressable

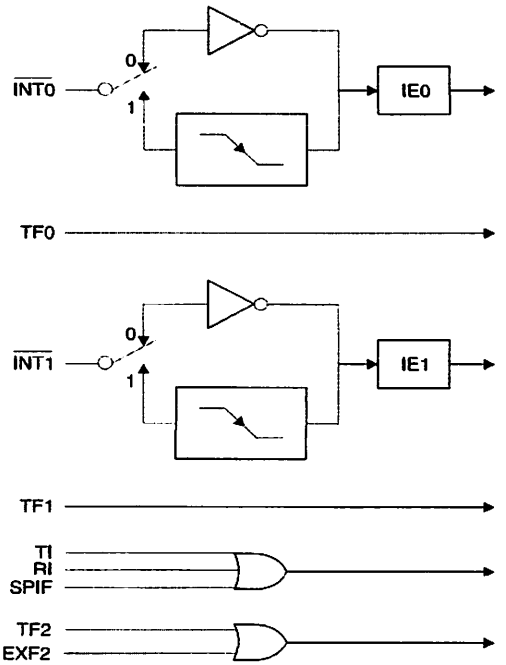
EA	–	ET2	ES	ET1	EX1	ET0	EX0

Enable Bit = 1 enables the interrupt.

Enable Bit = 0 disables the interrupt.

Symbol	Position	Function
EA	IE.7	Disables all interrupts. If EA = 0, no interrupt is acknowledged. If EA = 1, each interrupt source is individually enabled or disabled by setting or clearing its enable bit.
-	IE.6	Reserved.
ET2	IE.5	Timer 2 interrupt enable bit.
ES	IE.4	SPI and UART 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.
Jser software should never write 1s to reserved bits, because they may be used in future AT89 products.		

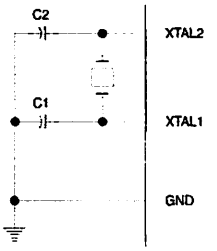
gure 15. Interrupt Sources



scillator haracteristics

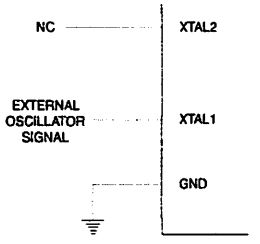
XTAL1 and XTAL2 are the input and output, respectively, of an inverting amplifier that can be configured for use as an on-chip oscillator, as shown in Figure 16. Either a quartz crystal or ceramic resonator may be used. To drive the device from an external clock source, XTAL2 should be left unconnected while XTAL1 is driven, as shown in Figure 17. 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 16. Oscillator Connections



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

Figure 17. External Clock Drive Configuration



Idle Mode

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

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

Table 17. Status of External Pins During Idle and Power-down Modes

Mode	Program Memory	ALE	$\overline{\text{PSEN}}$	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

ower-down lode

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

To exit power-down via an interrupt, external interrupt pin P3.2 or P3.3 must be kept low for at least the specified required crystal oscillator start up time. Afterwards, the interrupt service routine starts at the **rising edge** of the external interrupt pin.

rogram lemory Lock its

The AT89S8253 has three lock bits that can be left unprogrammed (U) or can be programmed (P) to obtain the additional features listed in Table 18.

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.

Once programmed, the lock bits can only be unprogrammed with the Chip Erase operation in either the parallel or serial modes.

Table 18. Lock Bit Protection Modes[1]

Program Lock Bits				
	LB1	LB2	LB3	Protection Type
1	U	U	U	No internal memory lock feature.
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 (parallel or serial mode) is disabled.
3	P	P	U	Same as Mode 2, but parallel or serial verify are also disabled.
4	P	P	P	Same as Mode 3, but external execution is also disabled.

Note: 1. U = Unprogrammed
P = Programmed

rogramming
ıe Flash and
EPROM

Atmel's AT89S8253 Flash microcontroller offers 12K bytes of In-System reprogrammable Flash code memory and 2K bytes of EEPROM data memory.

The AT89S8253 is normally shipped with the on-chip Flash code and EEPROM data memory arrays in the erased state (i.e. contents = FFH) and ready to be programmed. This device supports a parallel programming mode and a serial programming mode. The serial programming mode provides a convenient way to reprogram the AT89S8253 inside the user's system. The parallel programming mode is compatible with conventional third-party Flash or EPROM programmers.

The code and data memory arrays are mapped via separate address spaces in the parallel and serial programming modes: 0000H to 2FFFH for code memory and 000H to 7FFH for data memory.

The code and data memory arrays in the AT89S8253 are programmed byte-by-byte or by page in either programming mode. To reprogram any non-blank byte in the parallel or serial mode, the user needs to invoke the Chip Erase operation first to erase both arrays.

Parallel Programming Algorithm: To program and verify the AT89S8253 in the parallel programming mode, the following sequence is recommended (see Figure 21):

1. Power-up sequence:

   Apply power between V_{CC} and GND pins.

   Set RST pin to "H".

   Apply a 3 MHz to 24 MHz clock to XTAL1 pin and wait for at least 10 ms.
2. Set $\overline{PSEN}$ pin to "L"

   ALE pin to "H"

   $\overline{EA}$ pin to "H" and all other pins to "H".
3. Raise $\overline{EA}$/VPP to 12V to enable Flash programming, erase or verification. Enable the P3.0 pull-up (10 KΩ typical) for RDY/$\overline{BSY}$ operation.
4. Apply the appropriate combination of "H" or "L" logic levels to pins P3.3, P3.4, P3.5, P3.6, P3.7 to select one of the programming operations shown in the Flash Programming Modes table.
5. Apply the desired byte address to pins P1.0 to P1.7 and P2.0 to P2.5.

   Apply data to pins P0.0 to P0.7 for write code operation.
6. Pulse ALE/$\overline{PROG}$ once to load a byte in the code memory array, the data memory array, or the lock bits.
7. Repeat steps 5 and 6, changing the address and data for up to 64 bytes in the code memory page or 32 bytes in the data memory (EEPROM) page. When loading a page with individual bytes, the interval between consecutive byte loads should be no longer than 150 µs. Otherwise the device internally times out and assumes that the page load sequence is completed, rejecting any further loads before the page programming sequence has finished. This timing restriction also applies to Page Write of the 64-byte User Row.
8. After the last byte of the current page has been loaded, wait for 5 ms or monitor the RDY/$\overline{BUSY}$ pin until it transitions high. The page write cycle is self-timed and typically takes less than 5 ms.
9. To verify the last byte of the page just programmed, bring pin P3.4 to "L" and read the programmed data at pins P0.0 to P0.7.
10. Repeat steps 4 through 7 changing the address and data for the entire array or until the end of the object file is reached.

11. Power-off sequence:

    Tri-state the address and data inputs.

    Disable the P3.0 pullup used for RDY/$\overline{\text{BUSY}}$ operation.

    Set XTAL1 to "L".

    Set RST and $\overline{\text{EA}}$ pins to "L".

    Turn V_{CC} power off.

$\overline{\text{Data}}$ Polling: The AT89S8253 features $\overline{\text{DATA}}$ Polling to indicate the end of a byte write cycle. During a write cycle in the parallel or serial programming mode, an attempted read of the last loaded byte will result in the complement of the written datum on P0.7 (parallel mode), and on the MSB of the serial output byte on MISO (serial mode). Once the write cycle has been completed, true data are valid on all outputs, and the next cycle may begin. $\overline{\text{DATA}}$ Polling may begin any time after a write cycle has been initiated.

Ready/$\overline{\text{Busy}}$: The progress of byte programming in the parallel programming mode can also be monitored by the RDY/$\overline{\text{BSY}}$ output signal. Pin 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. P3.0 needs an external pullup (typical 10 KΩ) when functioning as RDY/$\overline{\text{BSY}}$.

Program Verify: If lock bits LB1 and LB2 have not been programmed, the programmed Code or Data byte can be read back via the address and data lines for verification. The state of the lock bits can also be verified directly in the parallel and serial programming modes.

Chip Erase: Both Flash and EEPROM arrays are erased electrically at the same time. In the parallel programming mode, Chip Erase is initiated by using the proper combination of control signals. The code and data arrays are written with all "1"s during the Chip Erase operation. The User Row will also be erased if the UsrRowProEn fuse (Fuse3) = 0 (enabled state).

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 also takes about 8 ms.

During Chip Erase, a serial read from any address location will return 00H at the data outputs.

Serial Programming Fuse: A programmable fuse is available to disable Serial Programming if the user needs maximum system security. The Serial Programming Fuse can be enabled/disabled in both the Parallel/Serial Programming Modes.

The AT89S8253 is shipped with the Serial Programming Mode enabled.

Reading the Signature Bytes: The signature bytes are read by the same procedure as a normal verification of locations 030H and 031H, except that P3.6 and P3.7 must be pulled to a logic low. The values returned are as follows:

(030H) = 1EH indicates manufactured by Atmel
(031H) = 73H indicates AT89S8253

rogramming
ıterface

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

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

erial ownloading

Both the code and data memory arrays can be programmed using the serial SPI bus 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 must be executed first before other operations can be executed.

The Chip Erase operation turns the content of every memory location in both the Code and Data arrays into FFH.

The code and data memory arrays have separate address spaces:

0000H to 2FFFH for code memory and 000H to 7FFH for data memory.

Either an external system clock is 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 24 MHz oscillator clock, the maximum SCK frequency is 1.5 MHz.

erial rogramming lgorithm

To program and verify the AT89S8253 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 24 MHz clock to XTAL1 pin and wait for at least 10 ms with RST pin high and P1.7 (SCK) low.
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 or data array is programmed one byte or one page at a time by supplying the address and data together with the appropriate Write instruction. The write cycle is self-timed and typically takes less than 4.0 ms at 5V.
4. Any memory location can be verified by using the Read instruction which returns the content at the selected address at serial output MISO/P1.6.
5. At the end of a programming session, RST can be set low to commence normal 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.

erial Programming Instruction

ıe Instruction Set for Serial Programming follows a 4-byte protocol and is shown in Table 19.

ıble 19. Serial Programming Instruction Set

nstruction	Instruction Format					Operation
	Byte 1	Byte 2	Byte 3	Byte 4	Byte n	
Programming Enable	1010 1100	0101 0011	xxxx xxxx	xxxx xxxx		Enable Serial Programming while RST is high
Chip Erase	1010 1100	100x xxxx	xxxx xxxx	xxxx xxxx		Chip Erase both the 12K and 2K memory arrays
Write Program Memory (Byte Mode)	0100 0000	xx A13 A12 A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0		Write data to Program Memory – Byte Mode
Read Program Memory (Byte Mode)	0010 0000	xx A13 A12 A11 A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0		Read data from Program Memory – Byte Mode
Write Program Memory (Page Mode)	0101 0000	xx A13 A12 A11 A10 A9 A8	A7 A6 00 0000	Byte 0 ... Byte 63		Write data to Program Memory – Page Mode (64 bytes)
Read Program Memory (Page Mode)	0011 0000	xx A13 A12 A11 A10 A9 A8	A7 A6 00 0000	Byte 0 ... Byte 63		Read data from Program Memory – Page Mode (64 bytes)
Write Data Memory (Byte Mode)	1100 0000	xxxx x A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0		Write data to Data Memory – Byte Mode
Read Data Memory (Byte Mode)	1010 0000	xxxx x A10 A9 A8	A7 A6 A5 A4 A3 A2 A1 A0	D7 D6 D5 D4 D3 D2 D1 D0		Read data from Data Memory – Byte Mode
Write Data Memory (Page Mode)	1101 0000	xxxx x A10 A9 A8	A7 A6 A5 0 0000	Byte 0 ... Byte 31		Write data to Data Memory – Page Mode (32 bytes)
Read Data Memory (Page Mode)	1011 0000	xxxx x A10 A9 A8	A7 A6 A5 0 0000	Byte 0 ... Byte 31		Read data from Data Memory – Page Mode (32 bytes)
Write User Fuses	1010 1100	0001 1 FUSE3 FUSE2 FUSE1	xxxx xxxx	xxxx xxxx		Write user fuse bits
Read User Fuses	0010 0001	xxxx xxxx	xxxx xxxx	xxxx x FUSE3 FUSE2 FUSE1		Read back status of user fuse bits
Write Lock Bits	1010 1100	1110 0 LB3 LB2 LB1	xxxx xxxx	xxxx xxxx		Write the lock bits
Read Lock Bits	0010 0100	xxxx xxxx	xxxx xxxx	xxxx x LB3 LB2 LB1		Read back current status of the lock bits (a programmed lock bit reads back as a "1")
Write User Sgn. Byte	0100 0010	xxxx xxxx	xx A13 A12 A11 A10 A9 A8	D7 D6 D5 D4 D3 D2 D1 D0		
Read User Sgn. Byte	0010 0010	xxxx xxxx	xx A13 A12 A11 A10 A9 A8	D7 D6 D5 D4 D3 D2 D1 D0		
Write User Sgn. Page	0101 0010	xxxx xxxx	xxxx xxxx	Byte 0 ... Byte 63		
Read User Sgn. Page	0011 0010	xxxx xxxx	xxxx xxxx	Byte 0 ... Byte 63		
Read ATMEL Sgn. Byte	0010 1000	xxxx xxxx	xx A13 A12 A11 A10 A9 A8	D7 D6 D5 D4 D3 D2 D1 D0		Read Signature Byte

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

ır Page Read/Write, the data always starts from byte 0 to 31 or 63. After the command byte and upper address byte are tched, each byte thereafter is treated as data until all 32 or 64 bytes are shifted in/out. Then the next instruction will be ady to be decoded.

lash and EEPROM Parallel Programming Modes

Mode		RST	PSEN	ALE	EA	P3.3	P3.4	P3.5	P3.6	P3.7	Data I/O P0.7:0	Address P2.5:0, P1.7:0
Serial Prog. Modes[1]		H	h	h								
Chip Erase[2]		H	L	1.0 µs	12V	H	L	H	L	L	X	X
Page Write[3][4][5]	12K Code	H	L	1.0 µs	12V	L	H	H	H	H	DI	ADDR
Read	12K Code	H	L	H	12V	L	L	H	H	H	DO	ADDR
Page Write[3][4][6]	2K Data	H	L	1.0 µs	12V	L	H	L	H	H	DI	ADDR
Read	2K Data	H	L	H	12V	L	L	L	H	H	DO	ADDR
Write Lock Bits[2][4]	Bit - 1	H	L	1.0 µs	12V	H	L	H	H	L	D0 = 0	X
	Bit - 2										D1 = 0	X
	Bit - 3										D2 = 0	X
Read Lock Bits	Bit - 1	H	L	H	12V	H	H	H	L	L	D0	X
	Bit - 2										D1	X
	Bit - 3										D2	X
Page Write[3][4][5]	User Row	H	L	1.0 µs	12V	H	L	H	H	H	DI	0 - 3FH
Read	User Row	H	L	H	12V	L	L	H	L	H	DO	0 - 3FH
Read	Sig. Row	H	L	H	12V	L	L	H	L	L	DO	0 - 3FH
Write Fuse[2][4]	Fuse1: SerialPrgEn	H	L	1.0 µs	12V	L	H	H	L	H	D0 = 0	X
	Fuse1: SerialPrgDis										D0 = 1	X
	Fuse2: x2 ClockEn										D1 = 0	X
	Fuse2: x2 ClockDis										D1 = 1	X
	Fuse3: UsrRowPrgEn										D2 = 0	X
	Fuse3: UsrRowPrgDis										D2 = 1	X
Read Fuse	SerialPrg (Fuse1)	H	L	H	12V	H	H	H	L	H	D0	X
	x2 Clock (Fuse2)										D1	X
	UsrRow Prg (Fuse3)										D2	X

otes:
1. See detailed timing for Serial Programming Mode.
2. Internally timed for 8.0 ms.
3. Internally timed for 8.0 ms. Programming begins 150 µs (minimum) after the last write pulse.
4. P3.0 is pulled low during programming to indicate RDY/$\overline{BSY}$
5. 1 to 64 bytes can be programmed at a time per page.
6. 1 to 32 bytes can be programmed at a time per page.

Figure 18. Programming the Flash/EEPROM Memory

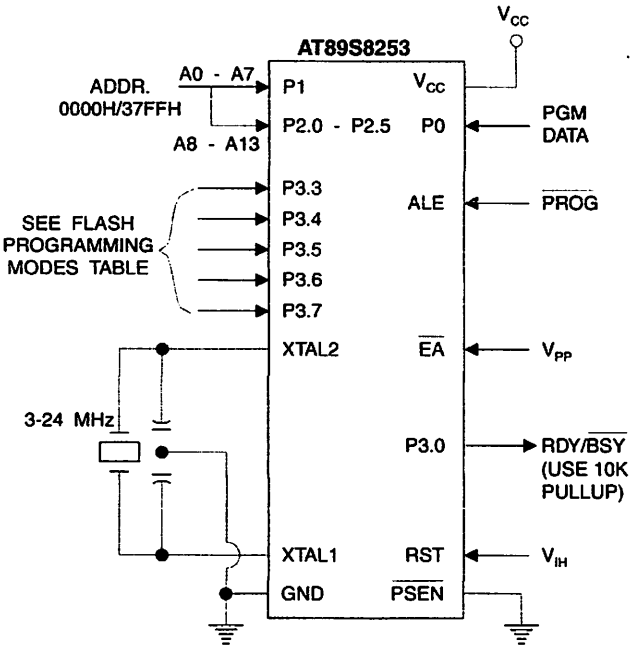


Figure 19. Verifying the Flash/EEPROM Memory

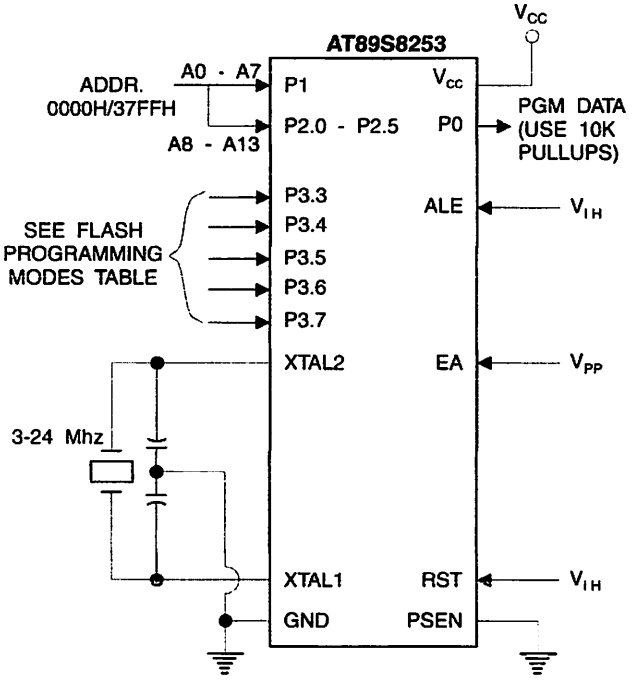
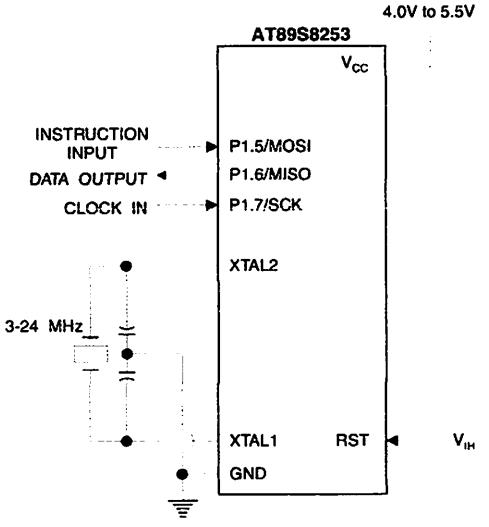


Figure 20. Flash/EEPROM Serial Downloading



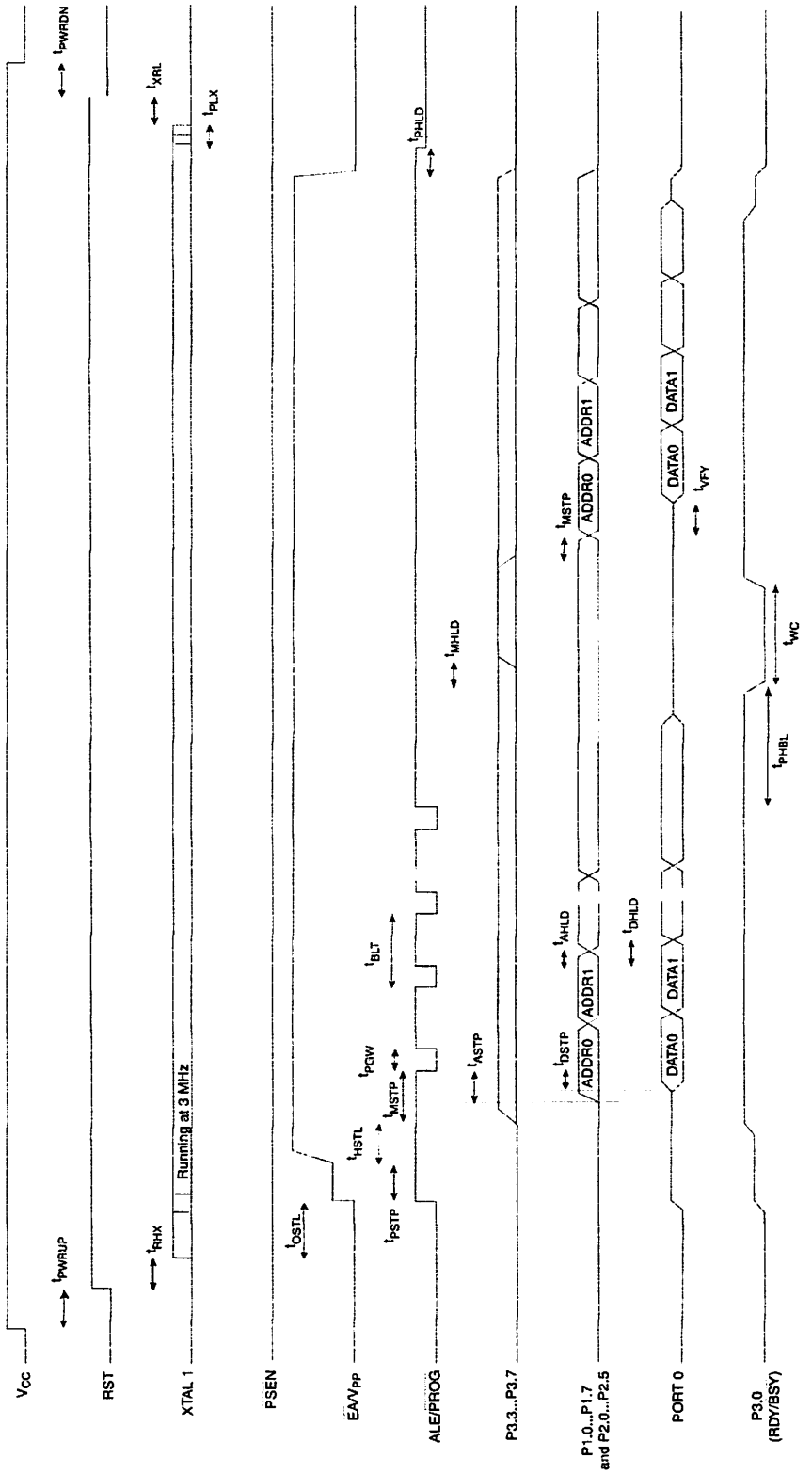
Flash Programming and Verification Characteristics – Parallel Mode

T_A = 20°C to 30°C, V_{CC} = 4.0V to 5.5V

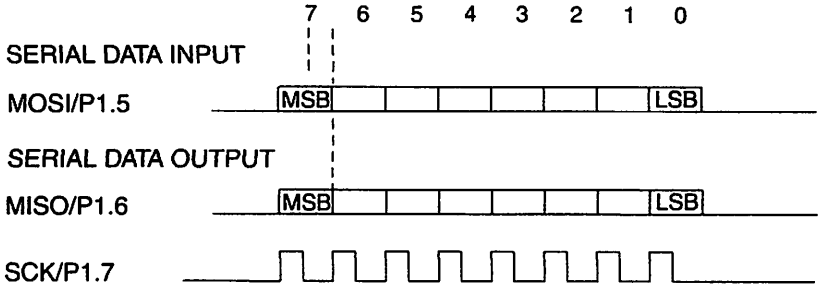
Symbol	Parameter	Min	Max	Units
V_{PP}	Programming Enable Voltage	11.5	12.5	V
I_{PP}	Programming Enable Current		1.0	mA
$1/t_{CLCL}$	Oscillator Frequency	3	24	MHz
t_{PWRUP}	Power On to RST High (1)	10		µs
t_{RHX}	RST High to XTAL Start	10		µs
t_{OSTL}	Oscillator Settling Time	10		ms
t_{HSTL}	High Voltage Settling Time	10		µs
t_{MSTP}	Mode Setup to $\overline{PROG}$ Low	1		µs
t_{ASTP}	Address Setup to $\overline{PROG}$ Low	1		µs
t_{DSTP}	Data Setup to $\overline{PROG}$ Low	1		µs
t_{PGW}	$\overline{PROG}$ Width	1		µs
t_{AHLD}	Address Hold after $\overline{PROG}$	1		µs
t_{DHLD}	Data Hold after $\overline{PROG}$	1		µs
t_{BLT}	Byte Load Period		150	µs
t_{PHBL}	$\overline{PROG}$ High to $\overline{BUSY}$ Low		256	µs
t_{WC}	Write Cycle Time(2)		4.5	ms
t_{MHLD}	Mode Hold After $\overline{BUSY}$ Low	10		µs
t_{VFY}	Address to Data Verify Valid		1	µs
t_{PSTP}	$\overline{PROG}$ Setup to V_{PP} High	10		µs
t_{PHLD}	$\overline{PROG}$ Hold after V_{PP} Low	10		µs
t_{PLX}	$\overline{PROG}$ Low to XTAL Halt	1		µs
t_{XRL}	XTAL Halt to RST Low	1		µs
t_{PWRDN}	RST Low to Power Off	1		µs

Notes: 1. Power On occurs once V_{CC} reaches 2.4V.
2. 9 ms if Chip Erase.

gure 21. Flash/EEPROM Programming and Verification Waveforms – Parallel Mode

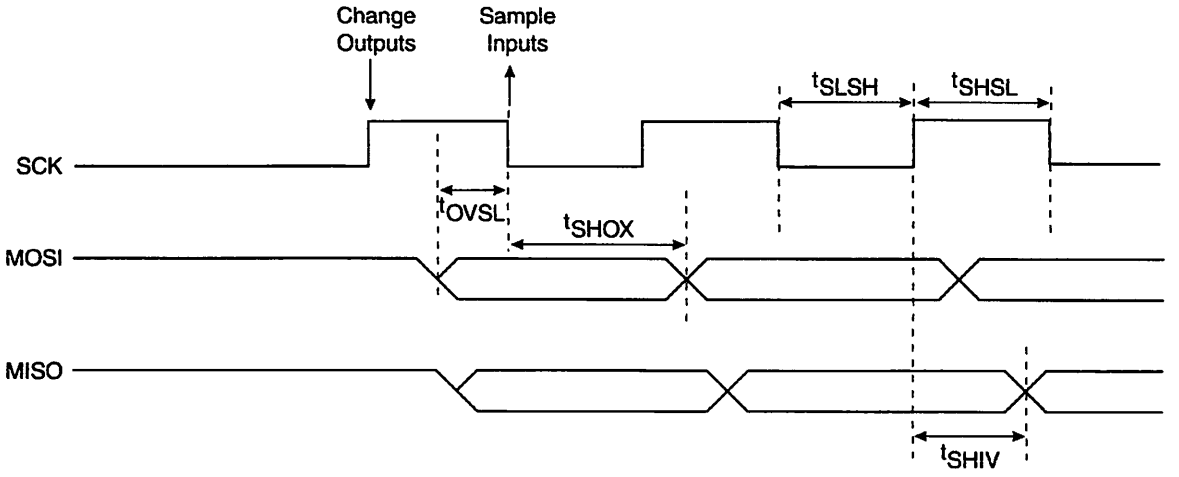


erial Downloading Waveforms (SPI Mode 1 —> CPOL = 0, CPHA = 1)



erial Programming Characteristics

gure 22. Serial Programming Timing



ıble 20. Serial Programming Characteristics, T_A = -40°C to 85°C, V_{CC} = 4.0V - 5.5V (Unless Otherwise Noted)

Symbol	Parameter	Min	Typ	Max	Units
I/t_{CLCL}	Oscillator Frequency	0		24	MHz
CLCL	Oscillator Period	41.6			ns
SHSL	SCK Pulse Width High	8 t_{CLCL}			ns
SLSH	SCK Pulse Width Low	8 t_{CLCL}			ns
OVSL	MOSI Setup to SCK Low	t_{CLCL}			ns
SHOX	MOSI Hold after SCK Low	2 t_{CLCL}			ns
SHIV	SCK High to MISO Valid	10	16	32	ns
ERASE	Chip Erase Instruction Cycle Time			9	ms
SWC	Serial Page Write Cycle Time			4.5	ms

ıbsolute 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.

ıC Characteristics

ıe values shown in this table are valid for T_A = -40°C to 85°C and V_{CC} = 4.0 to 5.5V, Unless Otherwise Noted.

Symbol	Parameter	Condition	Min	Max
V_{IL}	Input Low-voltage	(Except $\overline{EA}$)	-0.5V	0.2 V_{CC} - 0.1V
V_{IL1}	Input Low-voltage ($\overline{EA}$)		-0.5V	0.2 V_{CC} - 0.3V
V_{IH}	Input High-voltage	(Except XTAL1, RST)	0.2 V_{CC} + 0.9V	V_{CC} + 0.5V
V_{IH1}	Input High-voltage	(XTAL1, RST)	0.7 V_{CC}	V_{CC} + 0.5V
V_{OL}	Output Low-voltage[1]	I_{OL} = 10 mA, V_{CC} = 4.0V, T_A = 85°C		0.4V
V_{OH}	Output High-voltage When Weak Pull Ups are Enabled (Ports 1, 2, 3, ALE, $\overline{PSEN}$)	I_{OH} = -60 µA, T_A = 85°C	2.4V	
		I_{OH} = -25 µA, T_A = 85°C	0.75 V_{CC}	
		I_{OH} = -10 µA, T_A = 85°C	0.9 V_{CC}	
V_{OH1}	Output High-voltage When Strong Pull Ups are Enabled (Port 0 in External Bus Mode, P1, 2, 3, ALE, $\overline{PSEN}$)	I_{OH} = -40 mA, T_A = 85°C	2.4V	
		I_{OH} = -25 mA, T_A = 85°C	0.75 V_{CC}	
		I_{OH} = -10 mA, T_A = 85°C	0.9 V_{CC}	
I_{IL}	Logical 0 Input Current (Ports 1, 2, 3)	V_{IN} = 0.45V, V_{CC} = 5.5V, T_A = -40°C		-50 µA
I_{TL}	Logical 1 to 0 Transition Current (Ports 1, 2, 3)	V_{IN} = 2V, V_{CC} = 5.5V, T_A = -40°C		-352 µA
I_{LI}	Input Leakage Current (Port 0, $\overline{EA}$)	0.45V< V_{IN} < V_{CC}		±10 µA
R_{RST}	Reset Pull-down Resistor		50 KΩ	150 KΩ
C_{IO}	Pin Capacitance	Test Freq. = 1 MHz, T_A = 25°C		10 pF
I_{CC}	Power Supply Current	Active Mode, 12 MHz, V_{CC} = 5.5V, T_A = -40°C		25 mA
		Idle Mode, 12 MHz, V_{CC} = 5.5V, T_A = -40°C		6.5 mA
	Power-down Mode[2]	V_{CC} = 5.5V, T_A = -40°C		100 µA
		V_{CC} = 4.0V, T_A = -40°C		40 µA

ɔtes:
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: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.

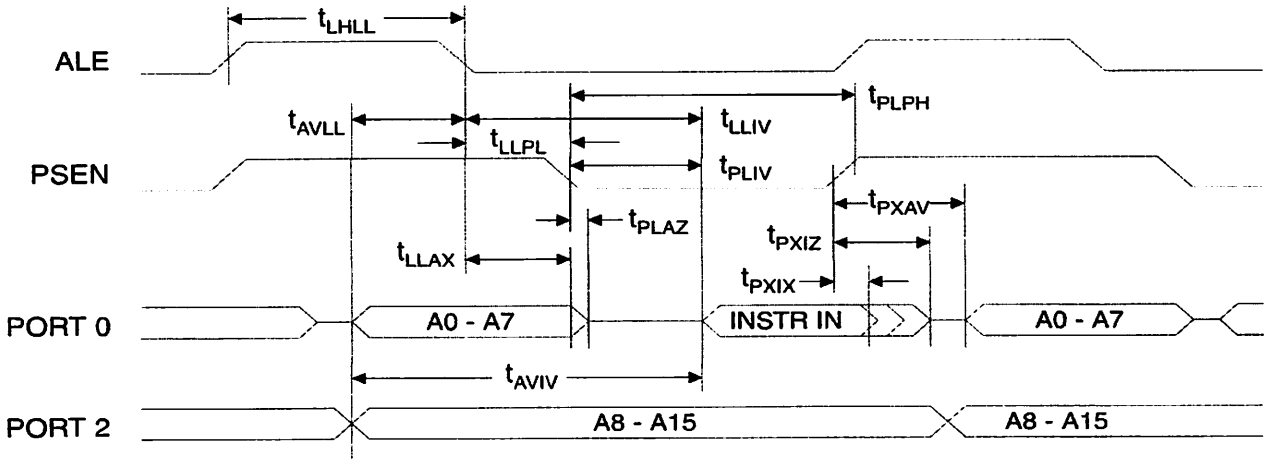
ıC Characteristics

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

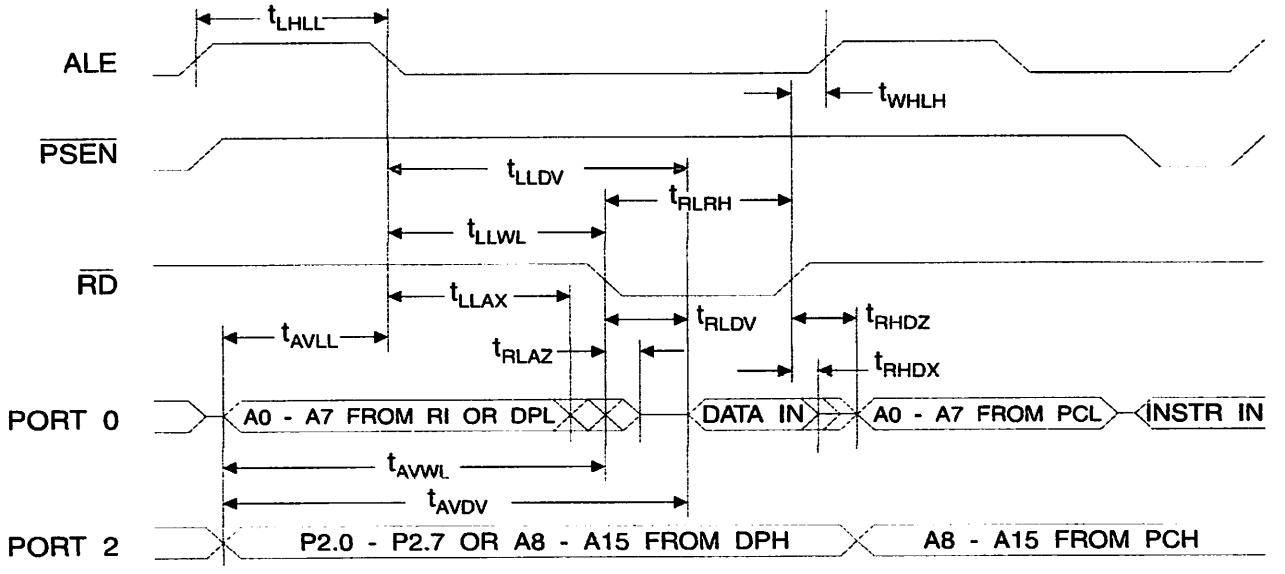
xternal Program and Data Memory Characteristics

Symbol	Parameter	Variable Oscillator		Units
		Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{LHLL}	ALE Pulse Width	$2t_{CLCL}$ - 12		ns
t_{AVLL}	Address Valid to ALE Low	t_{CLCL} - 12		ns
t_{LLAX}	Address Hold after ALE Low	t_{CLCL} - 16		ns
t_{LLIV}	ALE Low to Valid Instruction In		$4t_{CLCL}$ - 50	ns
t_{LLPL}	ALE Low to $\overline{PSEN}$ Low	t_{CLCL} - 12		ns
t_{PLPH}	$\overline{PSEN}$ Pulse Width	t_{CLCL}		ns
t_{PLIV}	$\overline{PSEN}$ Low to Valid Instruction In		$3t_{CLCL}$ - 50	ns
t_{PXIX}	Input Instruction Hold after $\overline{PSEN}$	-10		ns
t_{PXIZ}	Input Instruction Float after $\overline{PSEN}$		t_{CLCL} - 20	ns
t_{PXAV}	$\overline{PSEN}$ to Address Valid	t_{CLCL} - 4		ns
t_{AVIV}	Address to Valid Instruction In		$5t_{CLCL}$ - 50	ns
t_{PLAZ}	$\overline{PSEN}$ Low to Address Float		20	ns
t_{RLRH}	$\overline{RD}$ Pulse Width	$6t_{CLCL}$		ns
t_{WLWH}	$\overline{WR}$ Pulse Width	$6t_{CLCL}$		ns
t_{RLDV}	$\overline{RD}$ Low to Valid Data In		$5t_{CLCL}$ - 50	ns
t_{RHDX}	Data Hold after $\overline{RD}$	t_{CLCL} + 20		ns
t_{RHDZ}	Data Float after $\overline{RD}$		$2t_{CLCL}$ - 20	ns
t_{LLDV}	ALE Low to Valid Data In		$8t_{CLCL}$ - 50	ns
t_{AVDV}	Address to Valid Data In		$9t_{CLCL}$ - 50	ns
t_{LLWL}	ALE Low to $\overline{RD}$ or $\overline{WR}$ Low	$3t_{CLCL}$ - 24	$3t_{CLCL}$	ns
t_{AVWL}	Address to $\overline{RD}$ or $\overline{WR}$ Low	$4t_{CLCL}$ - 12		ns
t_{QVWX}	Data Valid to $\overline{WR}$ Transition	$2t_{CLCL}$ - 24		ns
t_{QVWH}	Data Valid to $\overline{WR}$ High	$8t_{CLCL}$ - 24		ns
t_{WHQX}	Data Hold after $\overline{WR}$	$2t_{CLCL}$ - 24		ns
t_{RLAZ}	$\overline{RD}$ Low to Address Float		$2t_{CLCL}$ + 10	ns
t_{WHLH}	$\overline{RD}$ or $\overline{WR}$ High to ALE High	t_{CLCL} - 10	t_{CLCL} + 20	ns

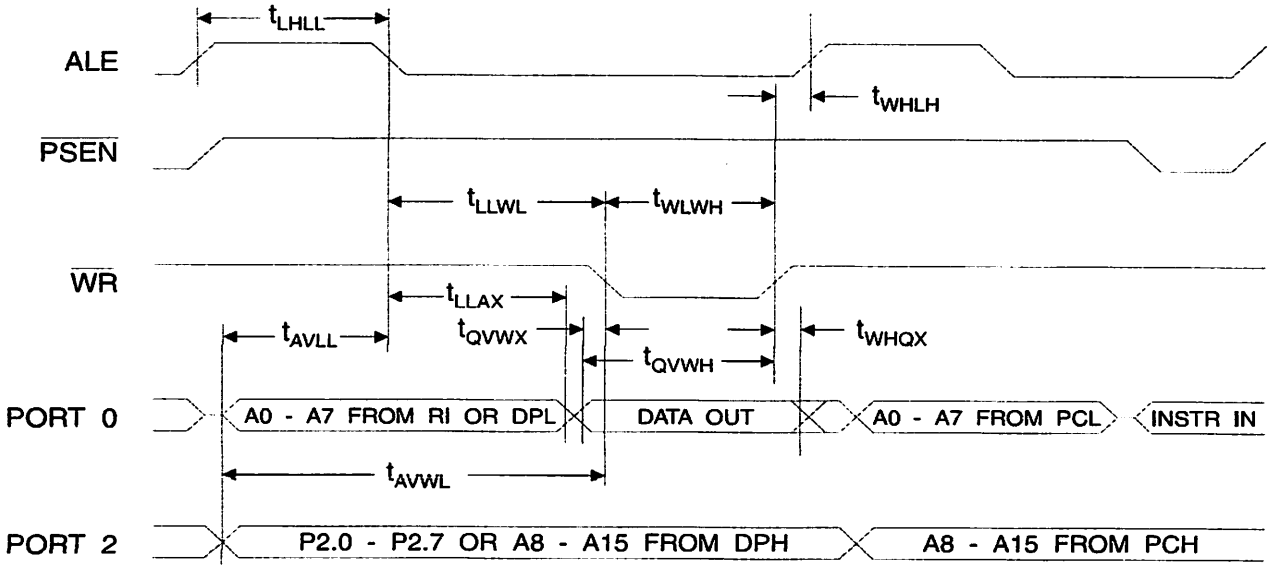
xternal Program Memory Read Cycle



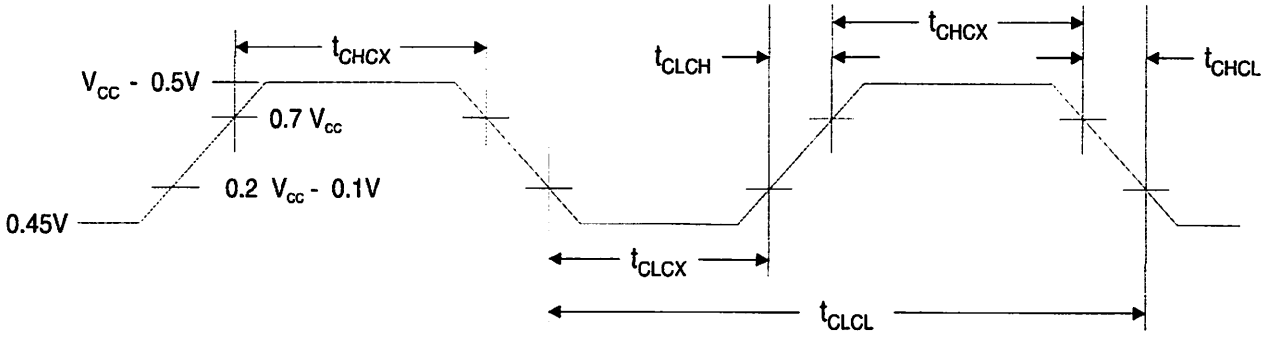
xternal Data Memory Read Cycle



External Data Memory Write Cycle



External Clock Drive Waveforms



External Clock Drive

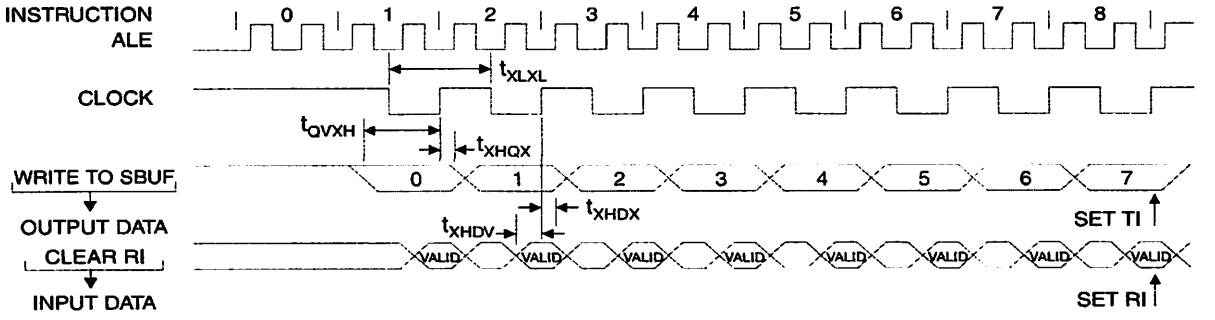
Symbol	Parameter	V_{CC} = 4.0V to 5.5V		Units
		Min	Max	
$1/t_{CLCL}$	Oscillator Frequency	0	24	MHz
t_{CLCL}	Clock Period	41.6		ns
t_{CHCX}	High Time	12		ns
t_{CLCX}	Low Time	12		ns
t_{CLCH}	Rise Time		5	ns
t_{CHCL}	Fall Time		5	ns

erial Port Timing: Shift Register Mode Test Conditions

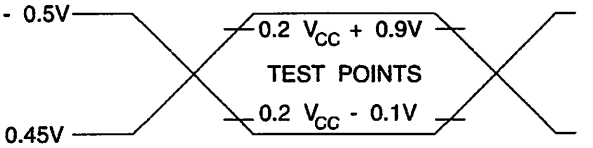
ıe values in this table are valid for V_{CC} = 4.0V to 5.5V and Load Capacitance = 80 pF.

Symbol	Parameter	Variable Oscillator		Units
		Min	Max	
XLXL	Serial Port Clock Cycle Time	$12t_{CLCL}$ -15		µs
QVXH	Output Data Setup to Clock Rising Edge	$10t_{CLCL}$ -15		ns
XHQX	Output Data Hold after Clock Rising Edge	$2t_{CLCL}$ -15		ns
XHDX	Input Data Hold after Clock Rising Edge	t_{CLCL}		ns
XHDV	Input Data Valid to Clock Rising Edge	0		ns

hift Register Mode Timing Waveforms

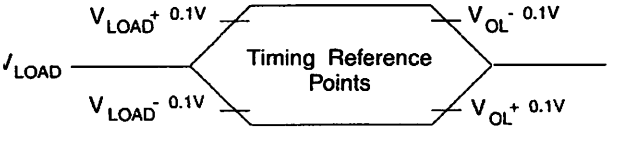


C Testing Input/Output Waveforms[1]



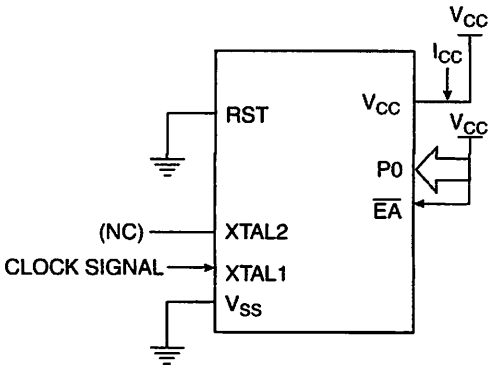
ote: 1. AC Inputs during testing are driven at V_{CC} - 0.5V for a logic 1 and 0.45V for a logic 0. Timing measurements are made at V_{IH} min. for a logic 1 and V_{IL} max. for a logic 0.

loat Waveforms[1]

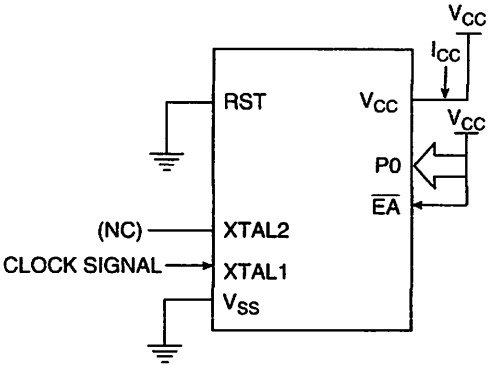


ote: 1. For timing purposes, a port pin is no longer floating when a 100 mV change from load voltage occurs. A port pin begins to float when a 100 mV change from the loaded V_{OH}/V_{OL} level occurs.

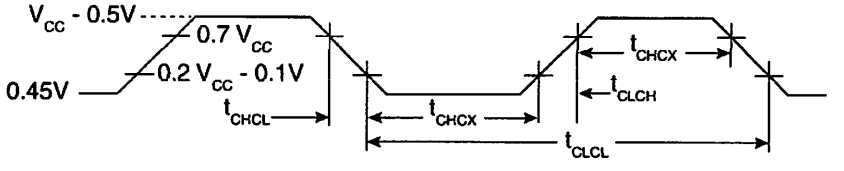
I_{CC} Test Condition, Active Mode, All Other Pins are Disconnected



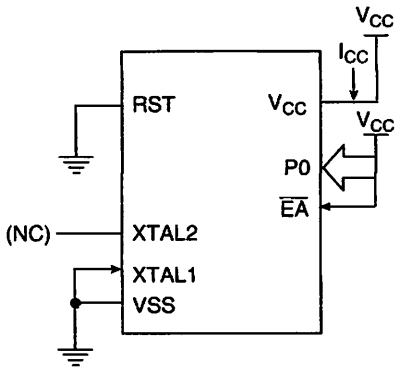
I_{CC} Test Condition, Idle Mode, All Other Pins are Disconnected



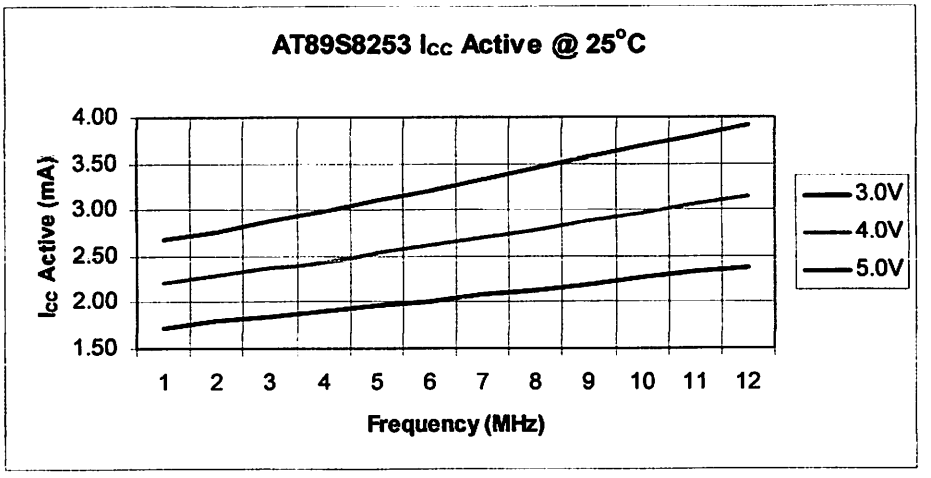
Clock Signal Waveform for I_{CC} Tests in Active and Idle Modes, $t_{CLCH} = t_{CHCL} = 5$ ns

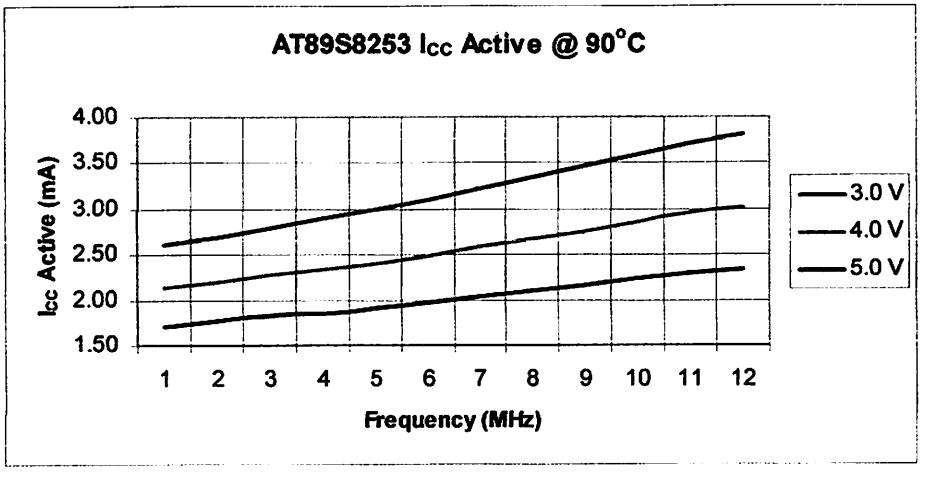


I_{CC} Test Condition, Power-down Mode, All Other Pins are Disconnected, V_{CC} = 2V to 5.5V

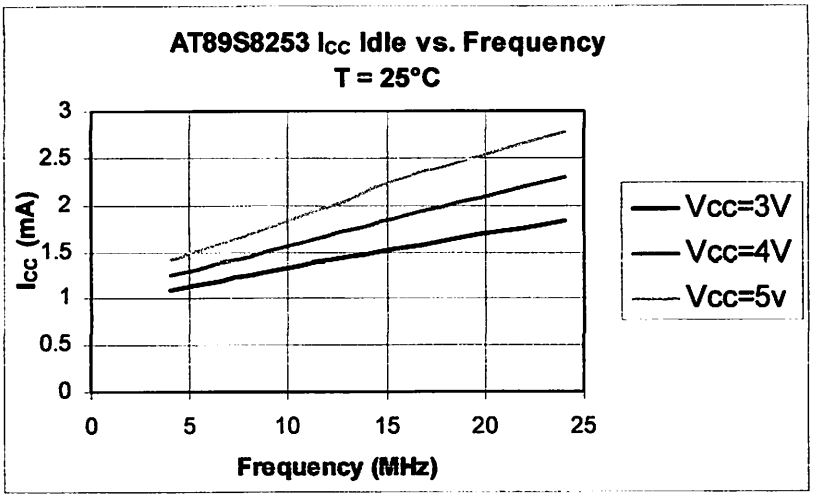


I_{CC} (Active Mode) Measurements

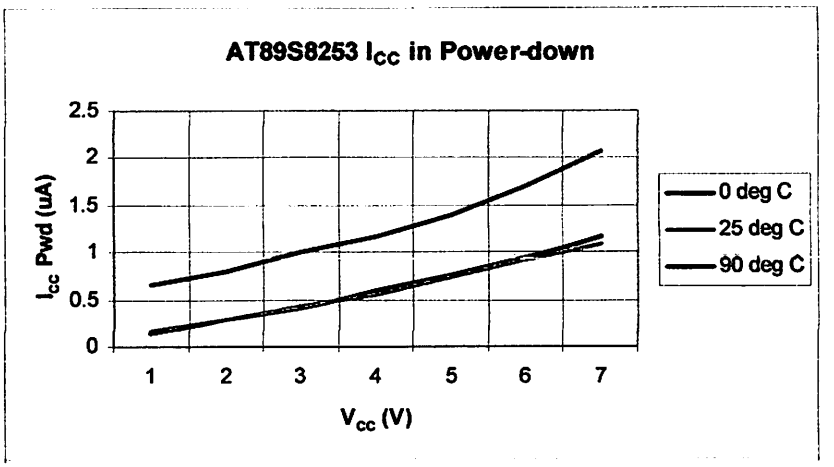




I_{CC} (Idle Mode) Measurements



I_{CC} (Power Down Mode) Measurements



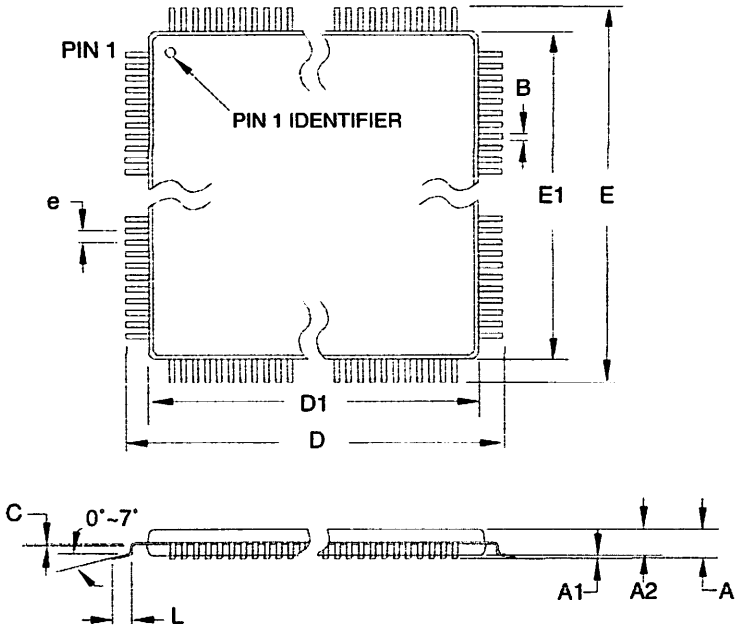
)rdering Information

Speed (MHz)	Power Supply	Ordering Code	Package	Operation Range
24	2.7V to 5.5V	AT89S8253-24AC AT89S8253-24JC AT89S8253-24PC AT89S8253-24SC	44A 44J 40P6 42PS6	Commercial (0°C to 70°C)
	2.7V to 5.5V	AT89S8253-24AI AT89S8253-24JI AT89S8253-24PI AT89S8253-24SI	44A 44J 40P6 42PS6	Industrial (-40°C to 85°C)

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

Package Information

44A – TQFP



COMMON DIMENSIONS
(Unit of Measure = mm)

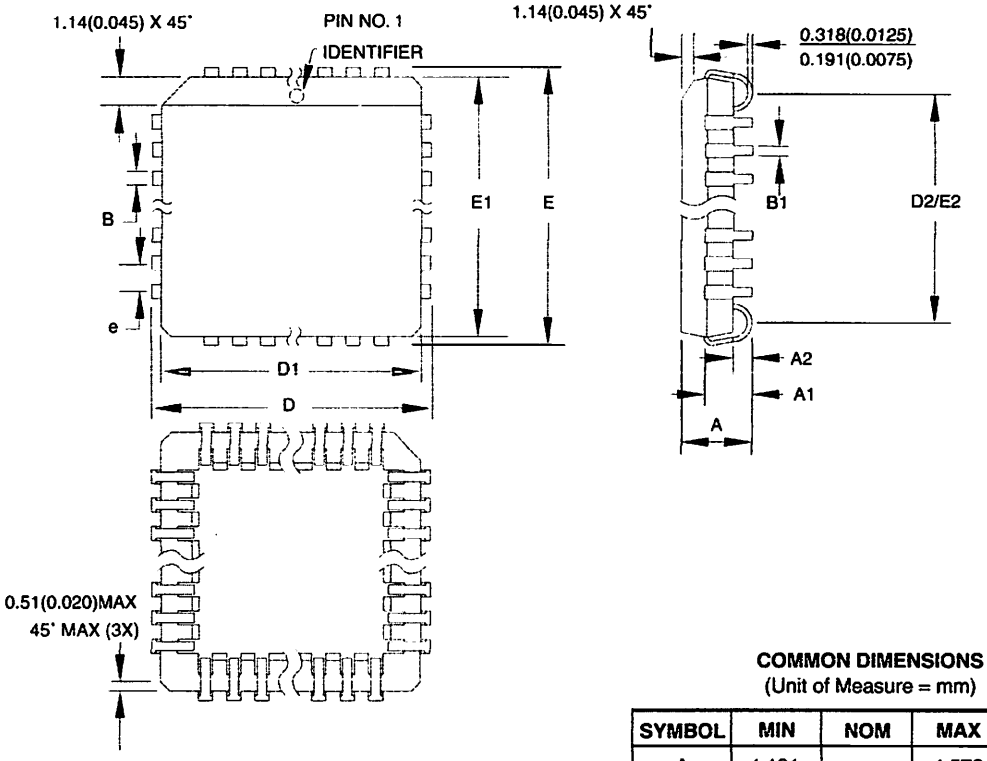
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	1.20	
A1	0.05	–	0.15	
A2	0.95	1.00	1.05	
D	11.75	12.00	12.25	
D1	9.90	10.00	10.10	Note 2
E	11.75	12.00	12.25	
E1	9.90	10.00	10.10	Note 2
B	0.30	–	0.45	
C	0.09	–	0.20	
L	0.45	–	0.75	
e	0.80 TYP			

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

10/5/2001

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

4J – PLCC



COMMON DIMENSIONS
(Unit of Measure = mm)

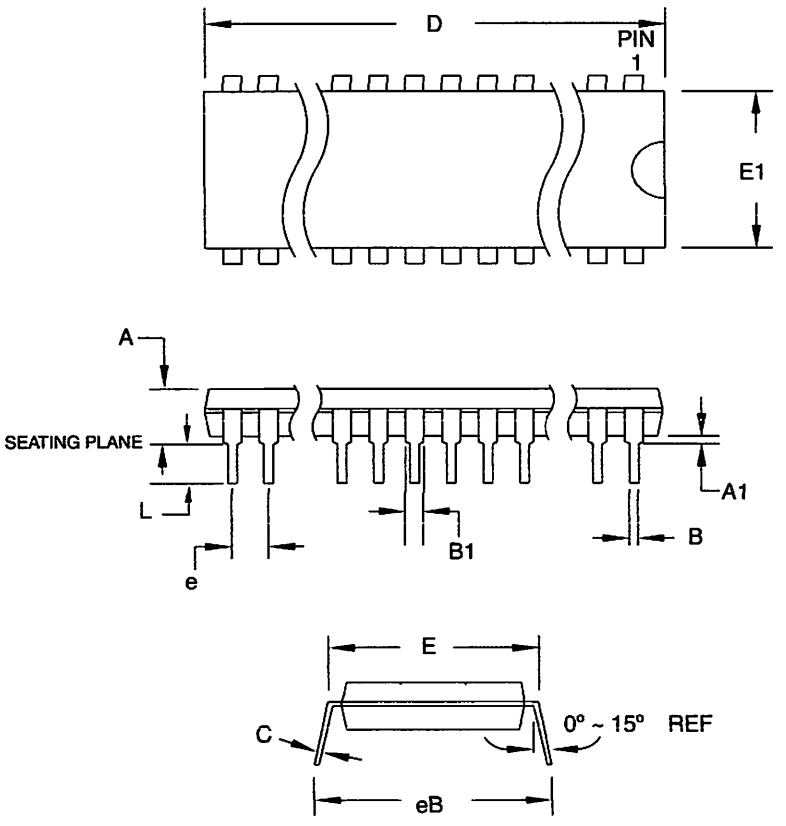
SYMBOL	MIN	NOM	MAX	NOTE
A	4.191	–	4.572	
A1	2.286	–	3.048	
A2	0.508	–	–	
D	17.399	–	17.653	
D1	16.510	–	16.662	Note 2
E	17.399	–	17.653	
E1	16.510	–	16.662	Note 2
D2/E2	14.986	–	16.002	
B	0.660	–	0.813	
B1	0.330	–	0.533	
e	1.270 TYP			

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

10/04/01

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

0P6 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

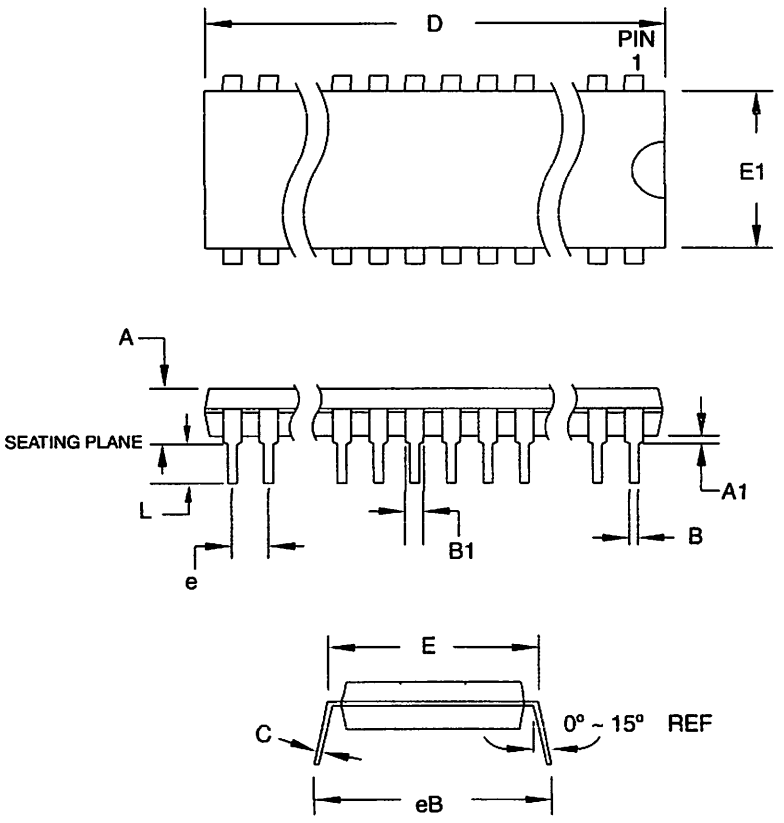
SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

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

09/28/01

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

2PS6 – PDIP



COMMON DIMENSIONS
(Unit of Measure = mm)

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

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

11/6/03

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

Atmel Corporation

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

Regional Headquarters

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

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

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

Atmel Operations

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

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

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

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

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

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

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

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

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

Literature Requests
www.atmel.com/literature

3286A–MICRO–7/04 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.

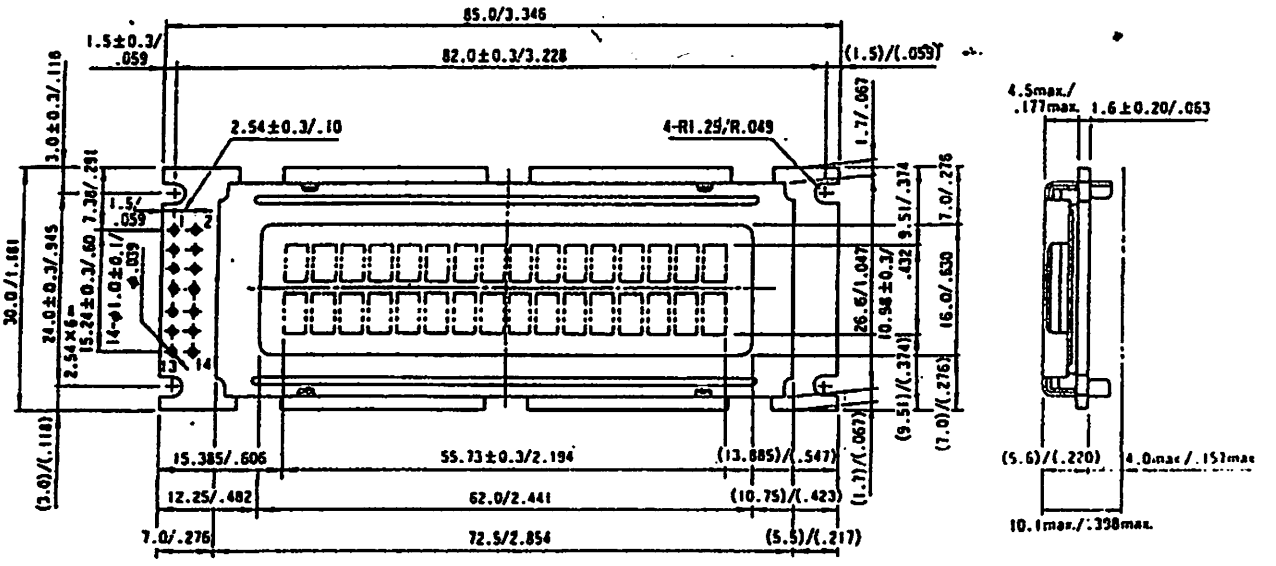
1.2 Features

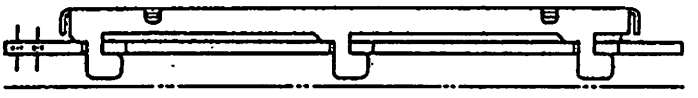
- 16-character, two-line TN liquid crystal display of 5 x 7 dot matrix + cursor
- Duty ratio: 1/16
- Character generator ROM for 192 character types (character font: 5 x 7 dot matrix)
- Character generator RAM for eight character types (program write) (character font: 5 x 7 dot matrix)
- 80 x 8 bit display 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

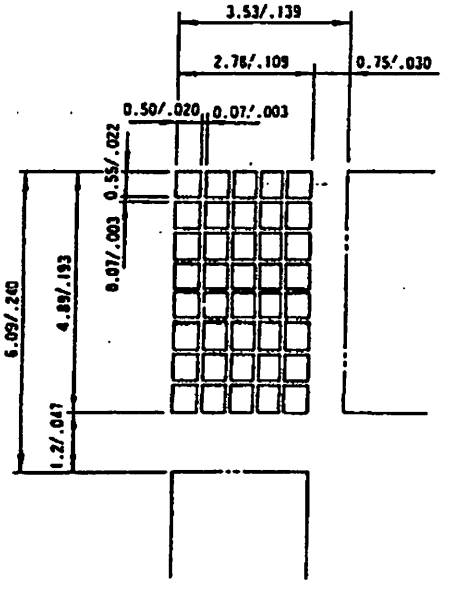
  Display Clear, Cursor Home, Display ON/OFF, Cursor ON/OFF, Display Character Blink, Cursor Shift, and Display Shift

- 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





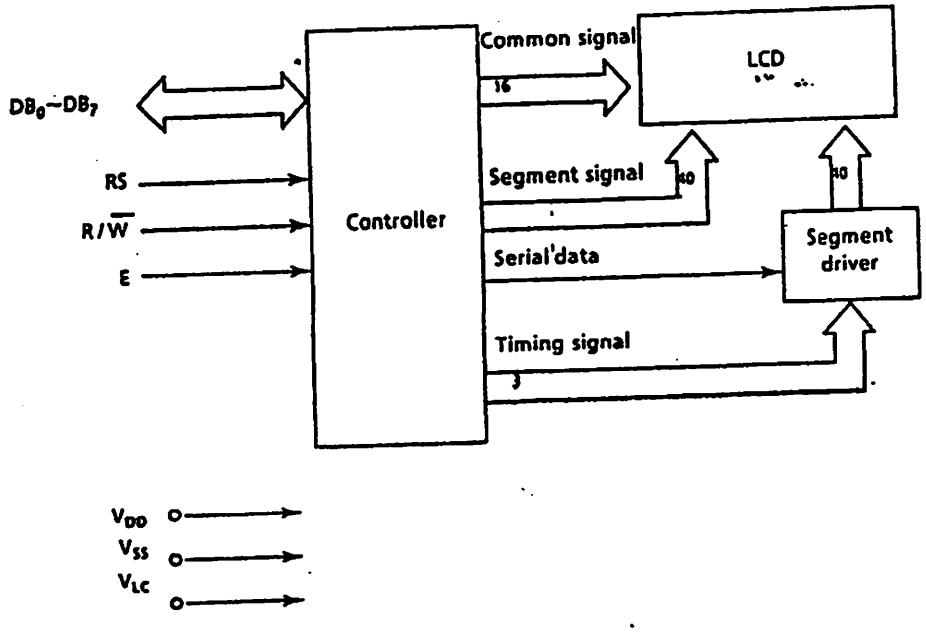


Unit : mm/inch
General tolerance : ± 0.5 mm

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 diagram

1.4 Block Diagram



1.5 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

1.6 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

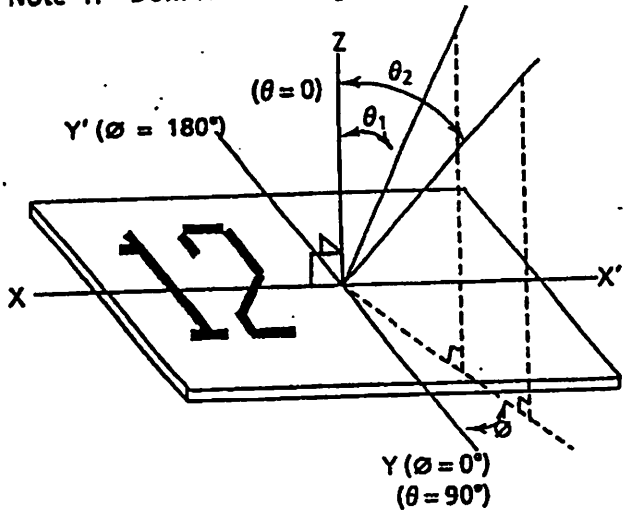
1.7 Optical Characteristics

1.7.1 Optical characteristics

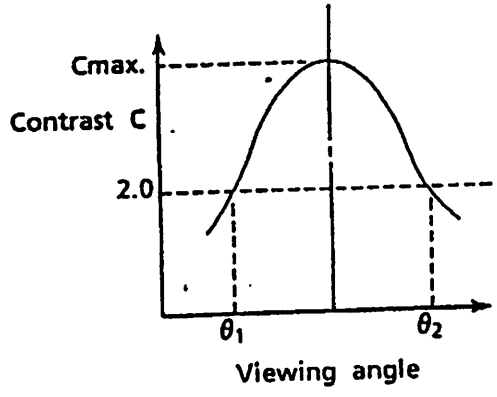
Maximum viewing angle: 6 o'clock ($\varnothing$ = 0°)
T_A = 25°C, V_{opr} = 4.75 V

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

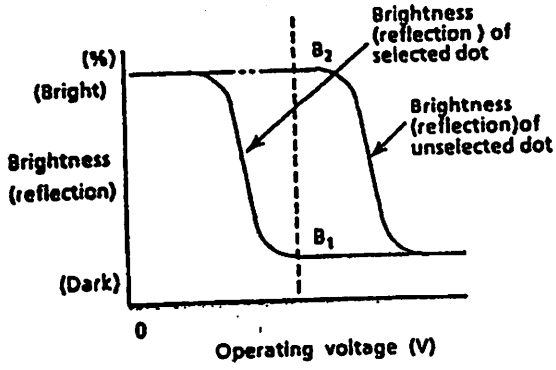


Note 2: Definition of viewing angles θ_1 and θ_2

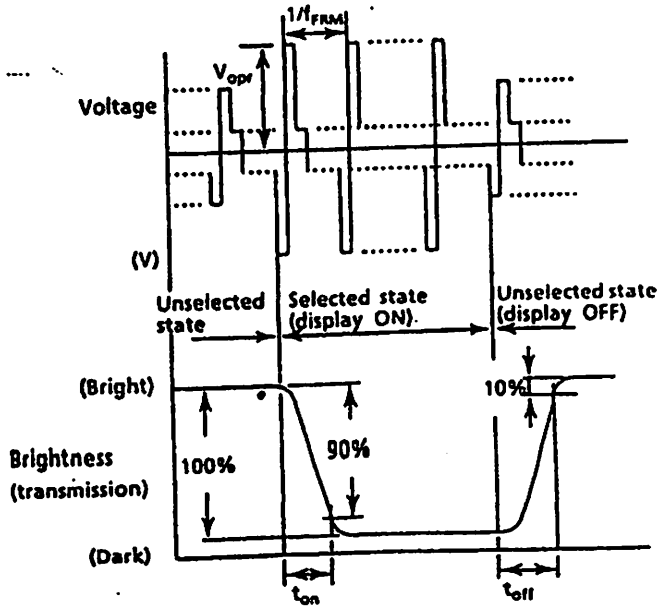


Note 3: Definition of contrast C

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



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

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

2. 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)

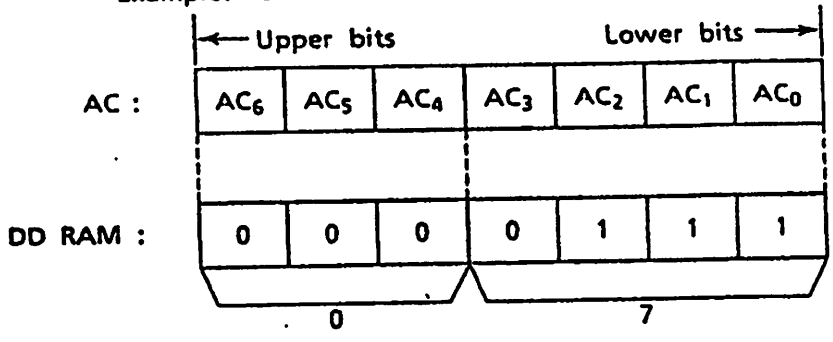
The counter specifies an address when data is written into DD RAM or CG RAM and the data stored in DD RAM or CG RAM is read out. If an Address Set instruction (for DD RAM or CG RAM) is written in the IR, the address information is transferred from the IR to the AC. When display data is written into or 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.

Example: DD RAM address = 07



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 bit 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 2 1 0 ←Upper bit 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 1 0 / 1 0 1 0 0 / 1 0 0 1 0 / 1 0 0 0 1 / 0 0 0 0 0 ← Cursor position / * * *
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 1 0 : 0 0 0 / 0 0 1	* * *
0 0 0 0 * 1 1 1	1 1 1 : 1 0 0 / 1 0 1 / 1 1 0 / 1 1 1	* * *

Example of character pattern (R)

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.

2.3 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

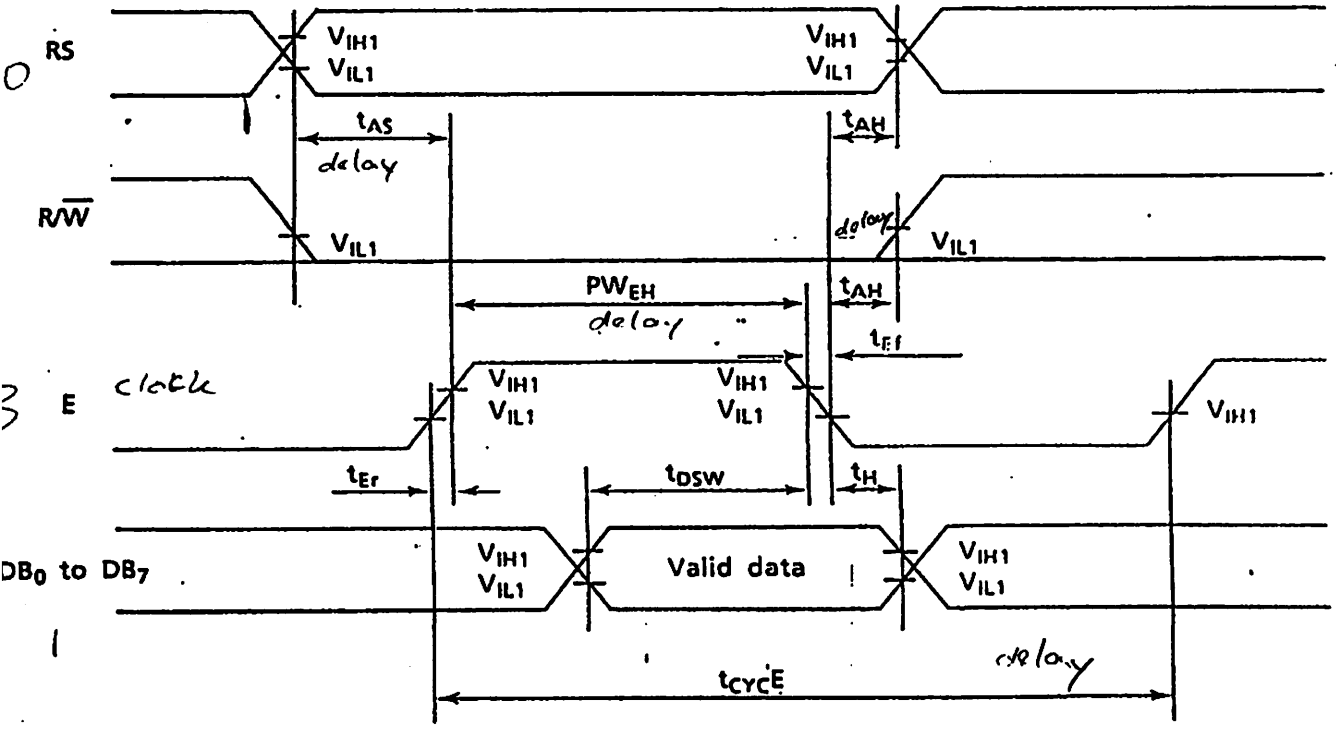


Figure 3 Data write from MPU to module

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_{CYCE}	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/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

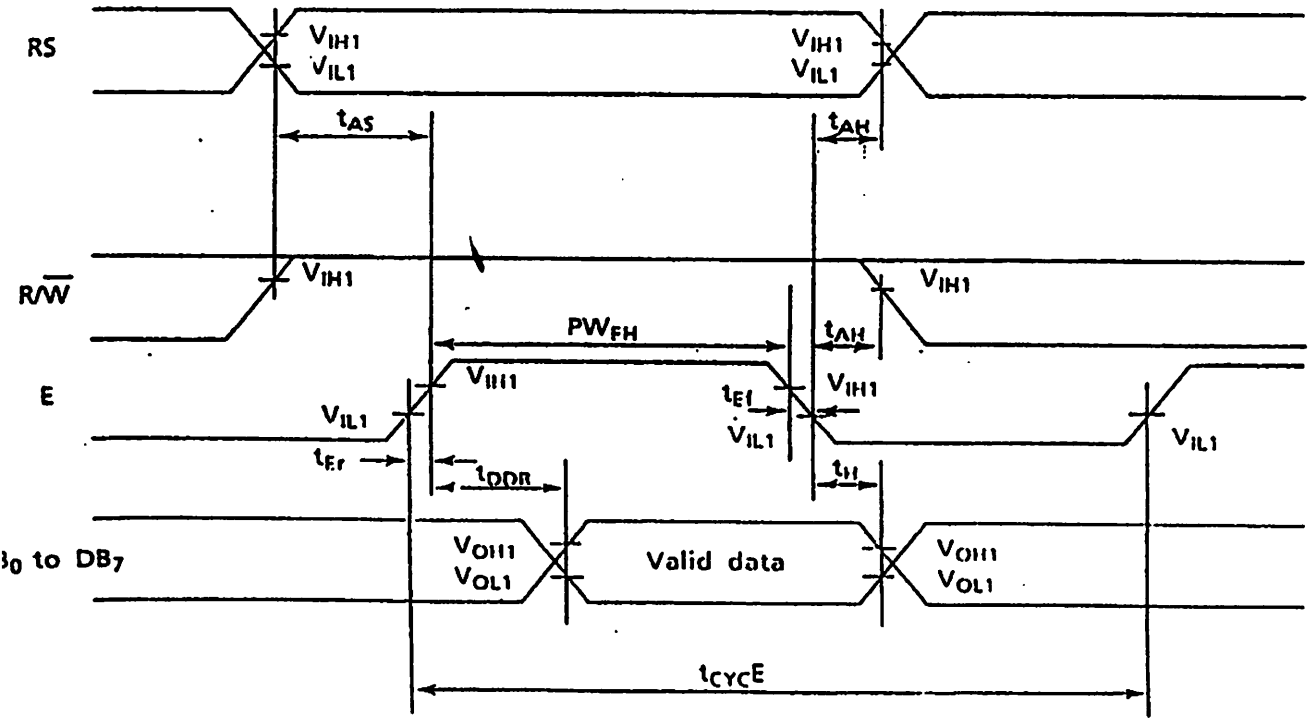


Figure 4 Data read from module to MPU

4 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		
1) 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
2) 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
3) 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
4) 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
5) 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
6) Function Set	0	0	0	0	1	DL	1	•	•	•	Sets interface data length (DL)	40 µs
7) CG RAM Address Set	0	0	0	1	A_{CG}						Sets CG RAM address to start transmitting or receiving CG RAM data	40 µs
8) DD RAM Address Set	0	0	1	A_{DD}							Sets DD RAM address to start transmitting or receiving DD RAM data	40 µs
9) 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
10) Data Write to CG RAM or DD RAM	1	0	Write Data								Writes data into DD RAM or CG RAM	40 µs
11) Data Read from CG RAM or DD RAM	1	1	Read Data								Reads data from DD RAM or CG RAM	40 µs

• : Invalid bit
A_{CG} : CG RAM address
A_{DD} : 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

1) Display Clear

	RS	R/$\overline{W}$	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.

(2) Cursor Home

	RS	R/$\overline{W}$	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.

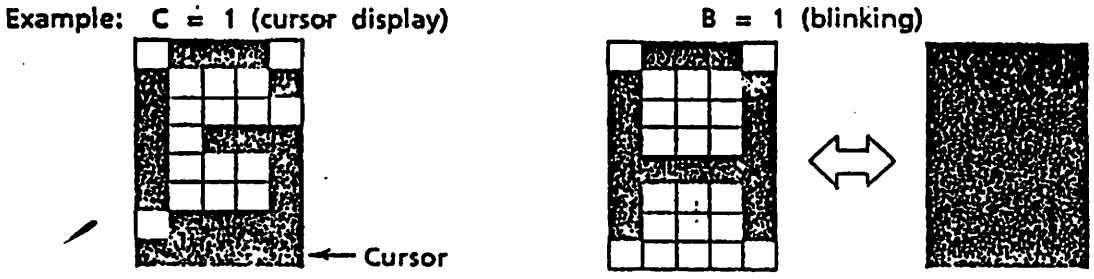
D : When D = 1, the display is turned ON.
When D = 0, the display is turned OFF.
If D = 0 is used, display data remains in DD RAM. Change 0 to 1 to display data.

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.



(5) Cursor/Display Shift

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

* :Invalid bit

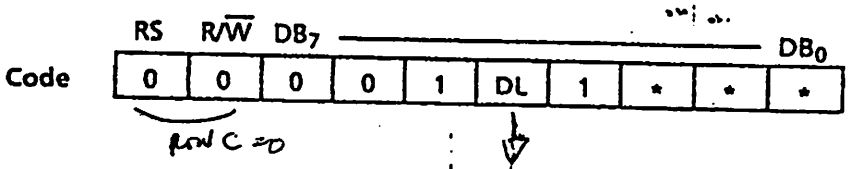
Cursor/Display Shift moves the cursor and shifts the display without changing the DD RAM contents.

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.

(6) Function Set



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).

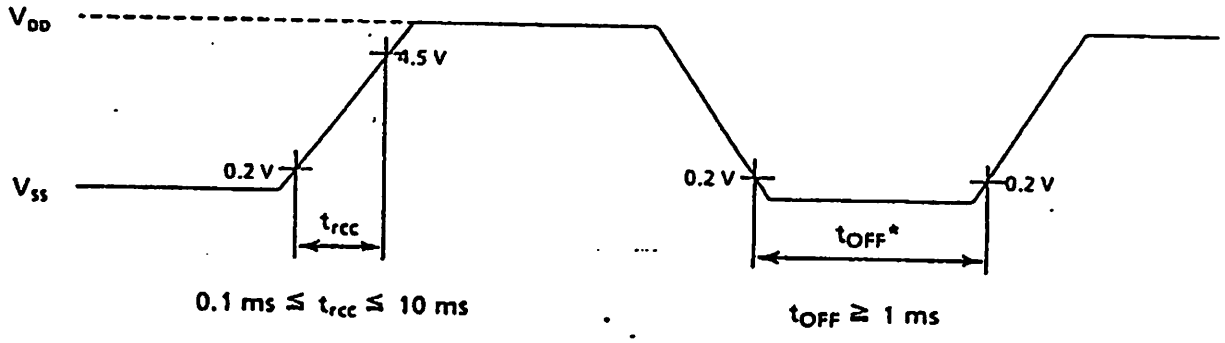
When DL = 0, the data length is set at four bits (DB_7 to DB_4).

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

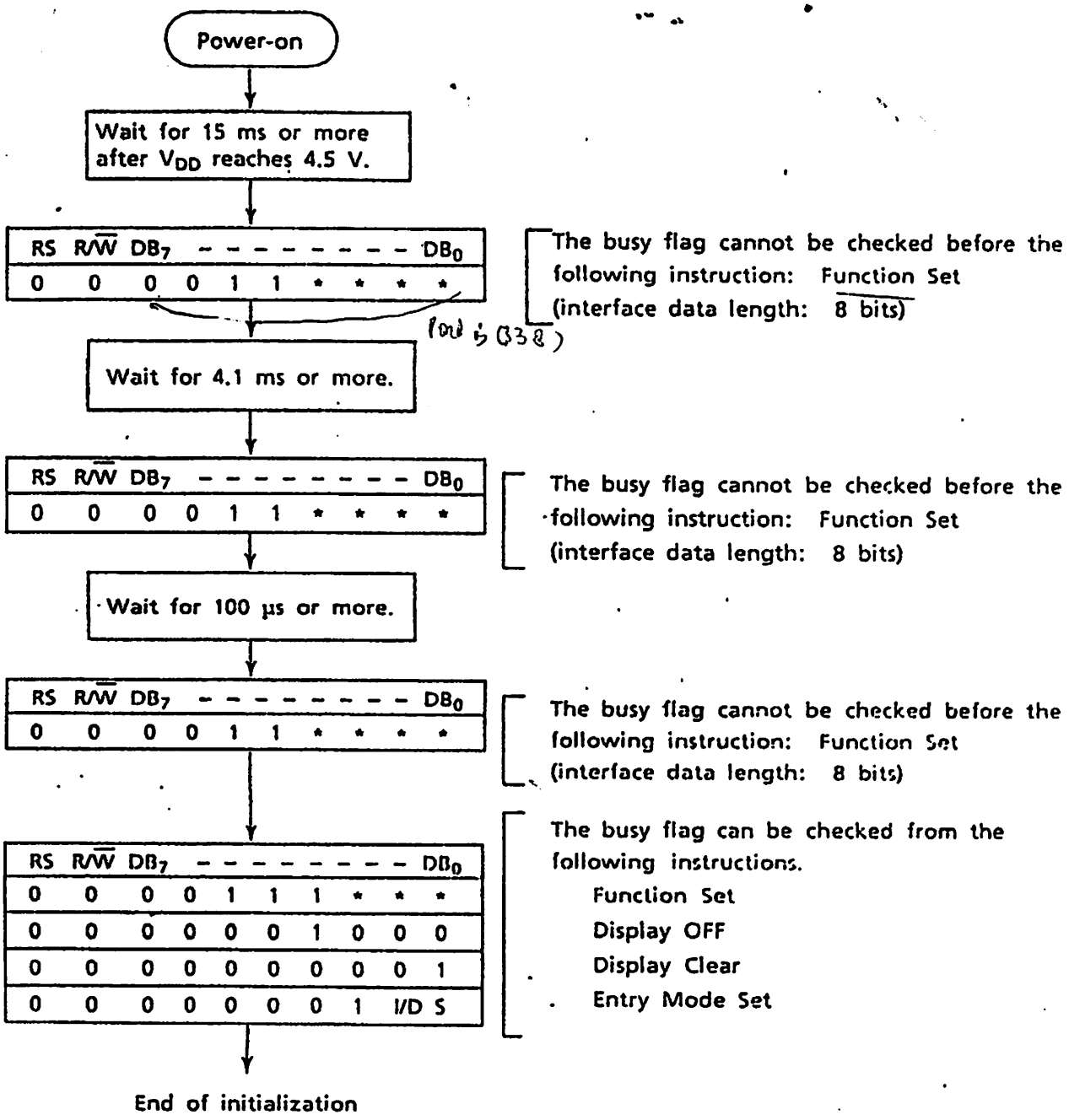
The following instructions are executed for initialization.

- 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

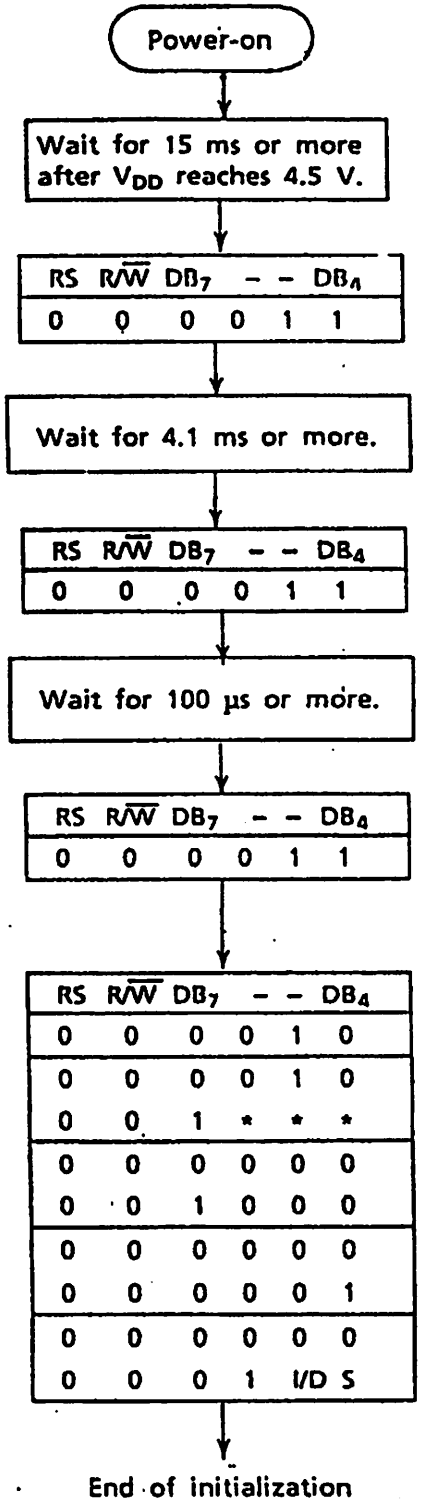
Since the condition is not suitable for the M1632, further function setting is necessary.

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

(a) **Interface data length : Eight bits**



b) **Interface data length: Four bits**



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

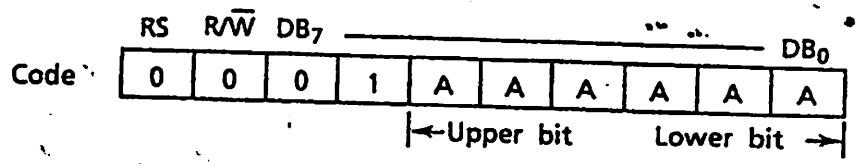
The busy flag cannot be checked before the following instruction: Function Set (interface data length: 8 bits)

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.

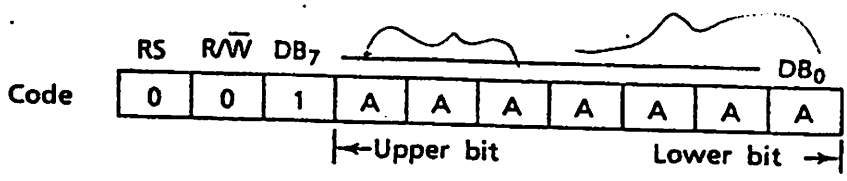
- Function Set (interface data length: 4 bits)
- Function Set
- Display OFF
- Display Clear
- Entry Mode Set

(7) CG RAM Address Set



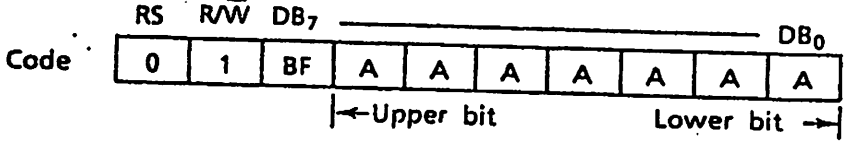
CG RAM addresses expressed as binary AAAAAA are set to the AC. Then data in CG RAM is written from or read to the MPU.

(8) 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.

(9) Busy Flag/Address Read



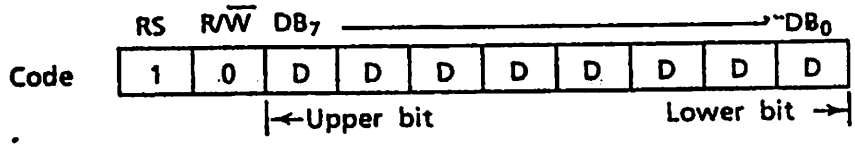
The BF signal is read out, indicating that the module is working internally because of the previous instruction.

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.

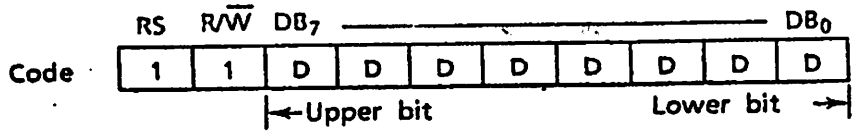
Therefore, make sure that BF = 0 before writing the next instruction. The AC values of binary AAAAAAA are read out at the same time as reading the busy flag. The AC addresses are used for both CG RAM and DD RAM but the address set before execution of the instruction determines which address is to be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

(10) 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.

(11) 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.

.6 Examples of Instruction Use

(1) Interface data length: Eight bits

No.	Instruction	Display	Operation
1	Power-on RS: —, R/W: —, DB_7 —— DB_0: —		The built-in reset circuit initializes the module.
2	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.
3	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.
4	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.
5	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.
6	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.
7	⋮	⋮	
8	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.

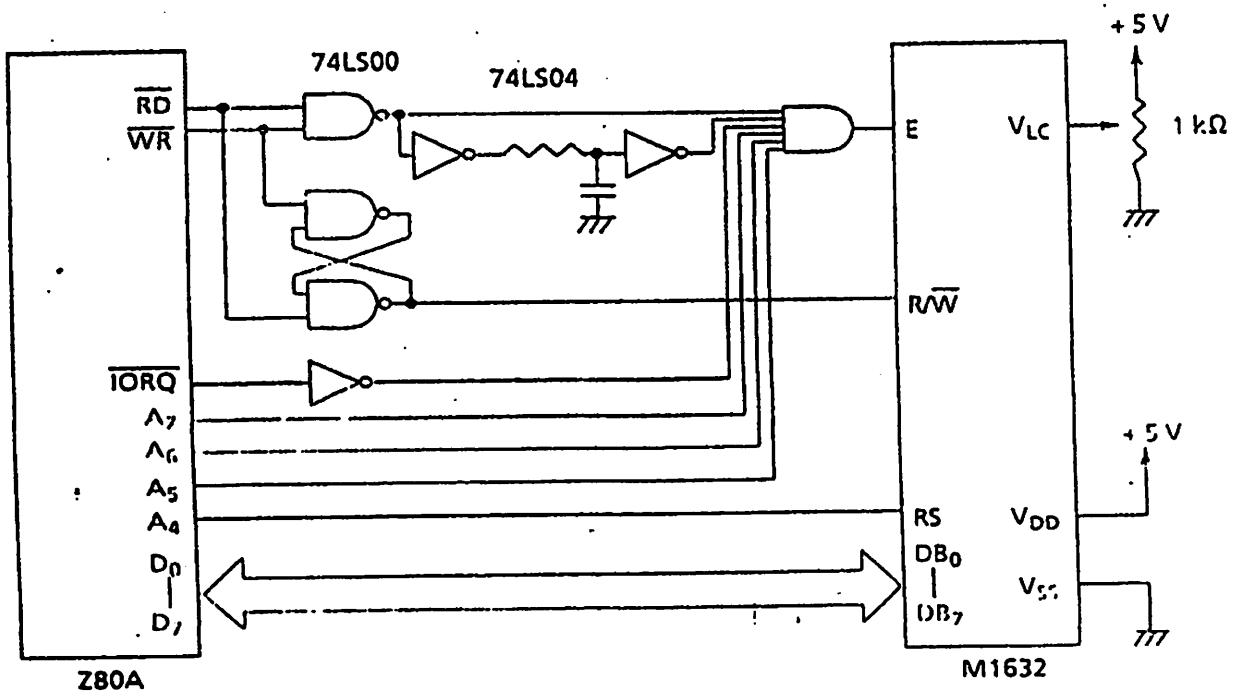
No.	Instruction	Display	Operation
9	DD RAM address set RS: 0, R/W: 0, DB_7 —— DB_0: 1 1 0 0 0 0 0 0	LCD MODULE M1632 _	The DD RAM address is set so that the cursor appears at digit 1 of line 2.
10	Write to CG RAM or DD RAM RS: 1, R/W: 0, DB_7 —— DB_0: 0 0 1 1 0 0 0 1	LCD MODULE M1632 1_	1 is written.
11	Write to CG RAM or DD RAM RS: 1, R/W: 0, DB_7 —— DB_0: 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, DB_7 —— DB_0: 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, DB_7 —— DB_0: 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, DB_7 —— DB_0: 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

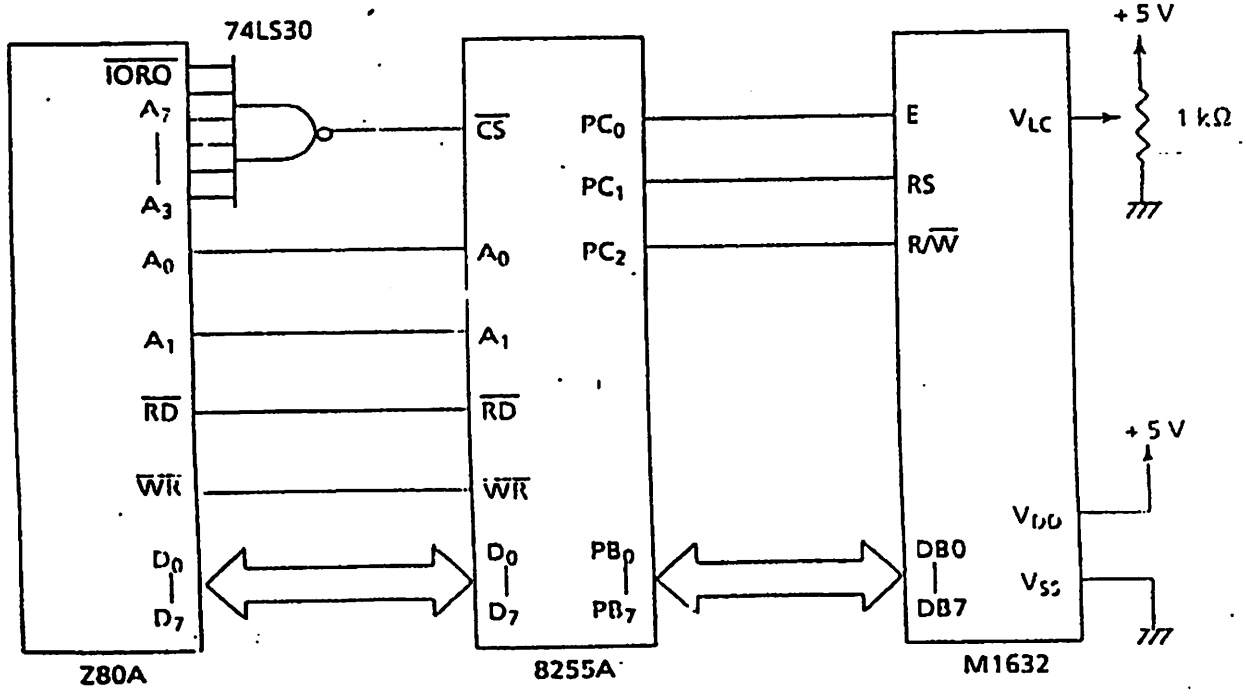
No.	Instruction	Display	Operation
1	Power-on RS: ⟋ R/W: ⟋ DB7 — DB4: ⟋		The built-in reset circuit initializes the module.
2	Function Set RS: 0 R/W: 0 DB7 — DB4: 0 0 1 0 RS: ⟋ R/W: ⟋ DB7 — DB4: ⟋		Four-bit operation mode is set. *Eight-bit operation mode is set by initialization, and the instruction 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.

.7 MPU Connection Diagrams

2.7.1 Z80A



2.7.2 Z80A and 8255A



Winbond Electronics Corp.

ISD2560/75/90/120

SINGLE-CHIP, MULTIPLE-MESSAGES, VOICE RECORD/PLAYBACK DEVICE 60-, 75-, 90-, AND 120-SECOND DURATION

Publication Release Date: May 2003
Revision 1.0

Winbond Electronics Corp.

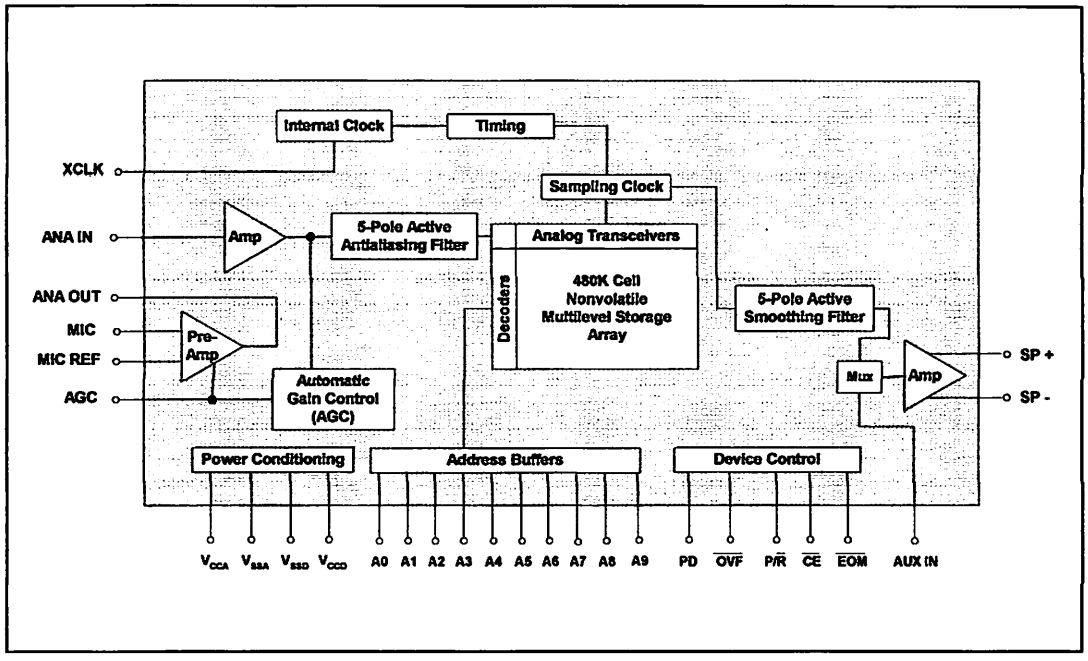
1. GENERAL DESCRIPTION

Winbond's ISD2500 ChipCorder® Series provide high-quality, single-chip, Record/Playback solutions for 60- to 120-second messaging applications. The CMOS devices include an on-chip oscillator, microphone preamplifier, automatic gain control, antialiasing filter, smoothing filter, speaker amplifier, and high density multi-level storage array. In addition, the ISD2500 is microcontroller compatible, allowing complex messaging and addressing to be achieved. Recordings are stored into on-chip nonvolatile memory cells, providing zero-power message storage. This unique, single-chip solution is made possible through Winbond's patented multilevel storage technology. Voice and audio signals are stored directly into memory in their natural form, providing high-quality, solid-state voice reproduction.

2. FEATURES

- Easy-to-use single-chip, voice record/playback solution
- High-quality, natural voice/audio reproduction
- Single-chip with duration of 60, 75, 90, or 120 seconds.
- Manual switch or microcontroller compatible
- Playback can be edge- or level-activated
- Directly cascadable for longer durations
- Automatic power-down (push-button mode)
  - Standby current 1 µA (typical)
- Zero-power message storage
  - Eliminates battery backup circuits
- Fully addressable to handle multiple messages
- 100-year message retention (typical)
- 100,000 record cycles (typical)
- On-chip clock source
- Programmer support for play-only applications
- Single +5 volt power supply
- Available in die form, PDIP, SOIC and TSOP packaging
- Temperature = die (0°C to +50°C) and package (0°C to +70°C)

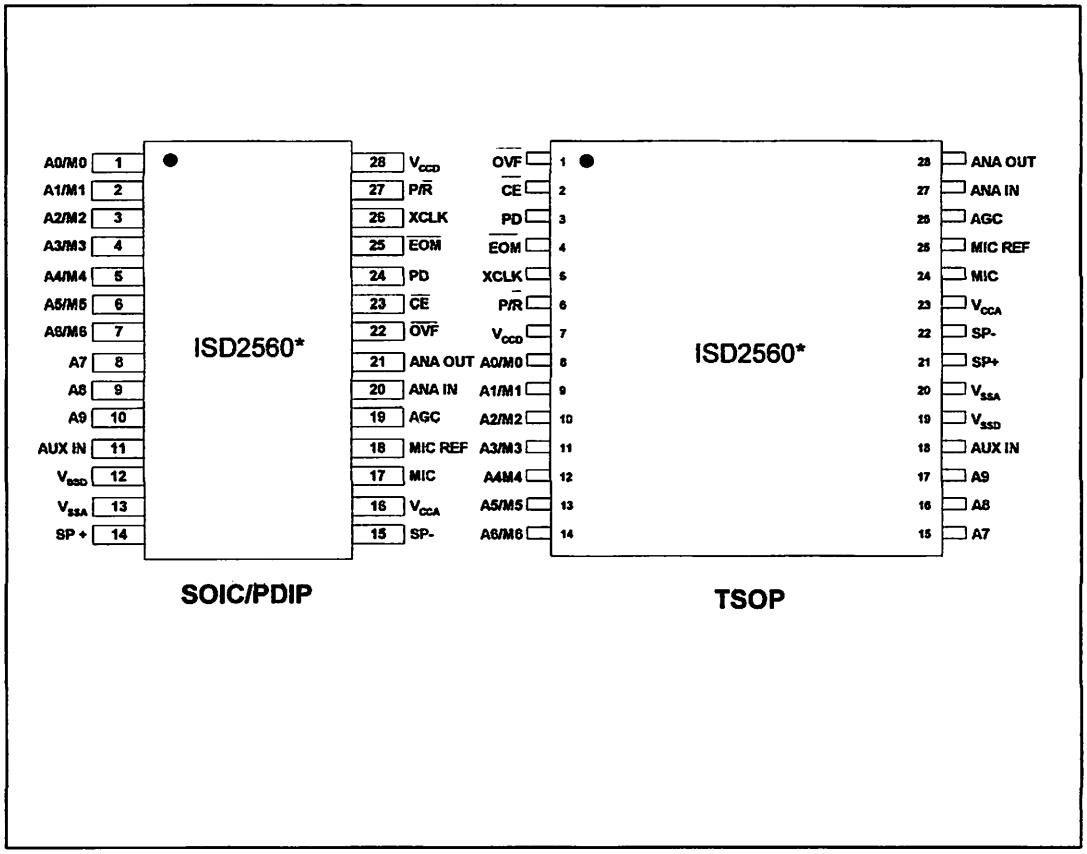
3. BLOCK DIAGRAM



4. TABLE OF CONTENTS

1. GENERAL DESCRIPTION 2
2. FEATURES 2
3. BLOCK DIAGRAM 3
4. TABLE OF CONTENTS 4
5. PIN CONFIGURATION 5
6. PIN DESCRIPTION 6
7. FUNCTIONAL DESCRIPTION 10
7.1. Detailed Description 10
7.2. Operational Modes 11
7.2.1. Operational Modes Description 12
8. TIMING DIAGRAMS 16
9. ABSOLUTE MAXIMUM RATINGS 19
9.1 Operating Conditions 20
10. ELECTRICAL CHARACTERISTICS 21
10.1. Parameters For Packaged Parts 21
10.1.1. Typical Parameter Variation with Voltage and Temperature 24
10.2. Parameters For Die 25
10.2.1. Typical Parameter Variation with Voltage and Temperature 28
10.3. Parameters For Push-Button Mode 29
11. TYPICAL APPLICATION CIRCUIT 30
12. PACKAGE DRAWING AND DIMENSIONS 35
12.1. 28-Lead 300-Mil Plastic Small Outline IC (SOIC) 35
12.2. 28-Lead 600-Mil Plastic Dual Inline Package (PDIP) 36
12.3. 28-Lead 8x13.4MM Plastic Thin Small Outline Package (TSOP) Type 1 37
12.4. ISD2560/75/95/120 Product Bonding Physical Layout (Die) [1] 38
14. VERSION HISTORY 41

5. PIN CONFIGURATION



* Same pinouts for ISD2575 / 2590 / 25120 products

6. PIN DESCRIPTION

PIN NAME	PIN NO.		FUNCTION
	SOIC/ PDIP	TSOP	
Ax/Mx	1-10/ 1-7	8-17/ 8-14	**Address/Mode Inputs**: The Address/Mode Inputs have two functions depending on the level of the two Most Significant Bits (MSB) of the address pins (A8 and A9). If either or both of the two MSBs are LOW, the inputs are all interpreted as address bits and are used as the start address for the current record or playback cycle. The address pins are inputs only and do not output any internal address information during the operation. Address inputs are latched by the falling edge of $\overline{CE}$. If both MSBs are HIGH, the Address/Mode inputs are interpreted as Mode bits according to the Operational Mode table on page 12. There are six operational modes (M0...M6) available as indicated in the table. It is possible to use multiple operational modes simultaneously. Operational Modes are sampled on each falling edge of $\overline{CE}$, and thus Operational Modes and direct addressing are mutually exclusive.
AUX IN	11	18	**Auxiliary Input:** The Auxiliary Input is multiplexed through to the output amplifier and speaker output pins when $\overline{CE}$ is HIGH, $\overline{P/R}$ is HIGH, and playback is currently not active or if the device is in playback overflow. When cascading multiple ISD2500 devices, the AUX IN pin is used to connect a playback signal from a following device to the previous output speaker drivers. For noise considerations, it is suggested that the auxiliary input not be driven when the storage array is active.
V_{SSA}, V_{SSD}	13, 12	20, 19	**Ground**: The ISD2500 series of devices utilizes separate analog and digital ground busses. These pins should be connected separately through a low-impedance path to power supply ground.
SP+/SP-	14/15	21/22	**Speaker Outputs**: All devices in the ISD2500 series include an on-chip differential speaker driver, capable of driving 50 mW into 16 Ω from AUX IN (12.2mW from memory). [1] The speaker outputs are held at V_{SSA} levels during record and power down. It is therefore not possible to parallel speaker outputs of multiple ISD2500 devices or the outputs of other speaker drivers. [2] A single-end output may be used (including a coupling capacitor between the SP pin and the speaker). These outputs may be used individually with the output signal taken from either pin. However, the use of single-end output results in a 1 to 4 reduction in its output power.

[1] Connection of speaker outputs in parallel may cause damage to the device.
[2] Never ground or drive an unused speaker output.

PIN NAME	PIN NO.		FUNCTION
	SOIC/ PDIP	TSOP	
V_{CCA}, V_{CCD}	16, 28	23, 7	**Supply Voltage**: To minimize noise, the analog and digital circuits in the ISD2500 series devices use separate power busses. These voltage busses are brought out to separate pins and should be tied together as close to the supply as possible. In addition, these supplies should be decoupled as close to the package as possible.
MIC	17	24	**Microphone**: The microphone pin transfers input signal to the on-chip preamplifier. A built-in Automatic Gain Control (AGC) circuit controls the gain of this preamplifier from –15 to 24dB. An external microphone should be AC coupled to this pin via a series capacitor. The capacitor value, together with the internal 10 KΩ resistance on this pin, determines the low-frequency cutoff for the ISD2500 series passband. See Winbond's Application Information for additional information on low-frequency cutoff calculation.
MIC REF	18	25	**Microphone Reference**: The MIC REF input is the inverting input to the microphone preamplifier. This provides a noise-canceling or common-mode rejection input to the device when connected to a differential microphone.
AGC	19	26	**Automatic Gain Control**: The AGC dynamically adjusts the gain of the preamplifier to compensate for the wide range of microphone input levels. The AGC allows the full range of whispers to loud sounds to be recorded with minimal distortion. The "attack" time is determined by the time constant of a 5 KΩ internal resistance and an external capacitor (C2 on the schematic of Figure 5 in section 11) connected from the AGC pin to V_{SSA} analog ground. The "release" time is determined by the time constant of an external resistor (R2) and an external capacitor (C2) connected in parallel between the AGC pin and V_{SSA} analog ground. Nominal values of 470 KΩ and 4.7 µF give satisfactory results in most cases.
ANA IN	20	27	**Analog Input**: The analog input transfers analog signal to the chip for recording. For microphone inputs, the ANA OUT pin should be connected via an external capacitor to the ANA IN pin. This capacitor value, together with the 3.0 KΩ input impedance of ANA IN, is selected to give additional cutoff at the low-frequency end of the voice passband. If the desired input is derived from a source other than a microphone, the signal can be fed, capacitively coupled, into the ANA IN pin directly.
ANA OUT	21	28	**Analog Output**: This pin provides the preamplifier output to the user. The voltage gain of the preamplifier is determined by the voltage level at the AGC pin.

PIN NAME	PIN NO.		FUNCTION
	SOIC/ PDIP	TSOP	
$\overline{OVF}$	22	1	**Overflow**: This signal pulses LOW at the end of memory array, indicating the device has been filled and the message has overflowed. The $\overline{OVF}$ output then follows the $\overline{CE}$ input until a PD pulse has reset the device. This pin can be used to cascade several ISD2500 devices together to increase record/playback durations.
$\overline{CE}$	23	2	**Chip Enable**: The $\overline{CE}$ input pin is taken LOW to enable all playback and record operations. The address pins and playback/record pin ($P/\overline{R}$) are latched by the falling edge of $\overline{CE}$. $\overline{CE}$ has additional functionality in the M6 (Push-Button) Operational Mode as described in the Operational Mode section.
PD	24	3	**Power Down**: When neither record nor playback operation, the PD pin should be pulled HIGH to place the part in standby mode (see I_{SB} specification). When overflow ($\overline{OVF}$) pulses LOW for an overflow condition, PD should be brought HIGH to reset the address pointer back to the beginning of the memory array. The PD pin has additional functionality in the M6 (Push-Button) Operation Mode as described in the Operational Mode section.
$\overline{EOM}$	25	4	**End-Of-Message**: A nonvolatile marker is automatically inserted at the end of each recorded message. It remains there until the message is recorded over. The $\overline{EOM}$ output pulses LOW for a period of T_{EOM} at the end of each message. In addition, the ISD2500 series has an internal V_{CC} detect circuit to maintain message integrity should V_{CC} fall below 3.5V. In this case, $\overline{EOM}$ goes LOW and the device is fixed in Playback-only mode. When the device is configured in Operational Mode M6 (Push-Button Mode), this pin provides an active-HIGH signal, indicating the device is currently recording or playing. This signal can conveniently drive an LED for visual indicator of a record or playback operation in process.

<table>
<tr><th rowspan="2">PIN NAME</th><th colspan="2">PIN NO.</th><th rowspan="2">FUNCTION</th></tr>
<tr><th>SOIC/ PDIP</th><th>TSOP</th></tr>
<tr><td>XCLK</td><td>26</td><td>5</td><td>External Clock: The external clock input has an internal pull-down device. The device is configured at the factory with an internal sampling clock frequency centered to ±1 percent of specification. The frequency is then maintained to a variation of ±2.25 percent over the entire commercial temperature and operating voltage ranges. If greater precision is required, the device can be clocked through the XCLK pin as follows:
<table>
<tr><th>Part Number</th><th>Sample Rate</th><th>Required Clock</th></tr>
<tr><td>ISD2560</td><td>8.0 kHz</td><td>1024 kHz</td></tr>
<tr><td>ISD2575</td><td>6.4 kHz</td><td>819.2 kHz</td></tr>
<tr><td>ISD2590</td><td>5.3 kHz</td><td>682.7 kHz</td></tr>
<tr><td>ISD25120</td><td>4.0 kHz</td><td>512 kHz</td></tr>
</table>
These recommended clock rates should not be varied because the antialiasing and smoothing filters are fixed, and aliasing problems can occur if the sample rate differs from the one recommended. The duty cycle on the input clock is not critical, as the clock is immediately divided by two. If the XCLK is not used, this input must be connected to ground.</td></tr>
<tr><td>P/$\overline{R}$</td><td>27</td><td>6</td><td>Playback/Record: The P/$\overline{R}$ input pin is latched by the falling edge of the $\overline{CE}$ pin. A HIGH level selects a playback cycle while a LOW level selects a record cycle. For a record cycle, the address pins provide the starting address and recording continues until PD or $\overline{CE}$ is pulled HIGH or an overflow is detected (i.e. the chip is full). When a record cycle is terminated by pulling PD or $\overline{CE}$ HIGH, then End-Of-Message ($\overline{EOM}$) marker is stored at the current address in memory. For a playback cycle, the address inputs provide the starting address and the device will play until an $\overline{EOM}$ marker is encountered. The device can continue to pass an $\overline{EOM}$ marker if $\overline{CE}$ is held LOW in address mode, or in an Operational Mode. (See Operational Modes section)</td></tr>
</table>

7. FUNCTIONAL DESCRIPTION

7.1. DETAILED DESCRIPTION

Speech/Sound Quality

The Winbond's ISD2500 series includes devices offered at 4.0, 5.3, 6.4, and 8.0 kHz sampling frequencies, allowing the user a choice of speech quality options. Increasing the duration within a product series decreases the sampling frequency and bandwidth, which affects the sound quality. Please refer to the ISD2560/75/90/120 Product Summary table below to compare the duration, sampling frequency and filter pass band.

The speech samples are stored directly into the on-chip nonvolatile memory without any digitization and compression associated like other solutions. Direct analog storage provides a very true, natural sounding reproduction of voice, music, tones, and sound effects not available with most solid state digital solutions.

Duration

To meet various system requirements, the ISD2560/75/90/120 products offer single-chip solutions at 60, 75, 90, and 120 seconds. Parts may also be cascaded together for longer durations.

TABLE 1: ISD2560/75/90/120 PRODUCT SUMMARY

Part Number	Duration (Seconds)	Input Sample Rate (kHz)	Typical Filter Pass Band * (kHz)
ISD2560	60	8.0	3.4
ISD2575	75	6.4	2.7
ISD2590	90	5.3	2.3
ISD25120	120	4.0	1.7

* 3db roll-off point

EEPROM Storage

One of the benefits of Winbond's ChipCorder® technology is the use of on-chip nonvolatile memory, providing zero-power message storage. The message is retained for up to 100 years typically without power. In addition, the device can be re-recorded typically over 100,000 times.

Microcontroller Interface

In addition to its simplicity and ease of use, the ISD2500 series includes all the interfaces necessary for microcontroller-driven applications. The address and control lines can be interfaced to a microcontroller and manipulated to perform a variety of tasks, including message assembly, message concatenation, predefined fixed message segmentation, and message management.

Programming

The ISD2500 series is also ideal for playback-only applications, where single or multiple messages are referenced through buttons, switches, or a microcontroller. Once the desired message configuration is created, duplicates can easily be generated via a gang programmer.

7.2. OPERATIONAL MODES

The ISD2500 series is designed with several built-in Operational Modes that provide maximum functionality with minimum external components. These modes are described in details as below. The Operational Modes are accessed via the address pins and mapped beyond the normal message address range. When the two Most Significant Bits (MSB), A8 and A9, are HIGH, the remaining address signals are interpreted as mode bits and not as address bits. Therefore, Operational Modes and direct addressing are not compatible and cannot be used simultaneously.

There are two important considerations for using Operational Modes. First, all operations begin initially at address 0 of its memory. Later operations can begin at other address locations, depending on the Operational Mode(s) chosen. In addition, the address pointer is reset to 0 when the device is changed from record to playback, playback to record (except M6 mode), or when a Power-Down cycle is executed.

Second, Operational Modes are executed when $\overline{CE}$ goes LOW. This Operational Mode remains in effect until the next LOW-going $\overline{CE}$ signal, at which point the current mode(s) are sampled and executed.

TABLE 2: OPERATIONAL MODES

Mode [1]	Function	Typical Use	Jointly Compatible [2]
M0	Message cueing	Fast-forward through messages	M4, M5, M6
M1	Delete $\overline{EOM}$ markers	Position $\overline{EOM}$ marker at the end of the last message	M3, M4, M5, M6
M2	Not applicable	Reserved	N/A
M3	Looping	Continuous playback from Address 0	M1, M5, M6
M4	Consecutive addressing	Record/playback multiple consecutive messages	M0, M1, M5
M5	$\overline{CE}$ level-activated	Allows message pausing	M0, M1, M3, M4
M6	Push-button control	Simplified device interface	M0, M1, M3

[1] Besides mode pin needed to be "1", A8 and A9 pin are also required to be "1" in order to enter into the related operational mode.

[2] Indicates additional Operational Modes which can be used simultaneously with the given mode.

7.2.1. Operational Modes Description

The Operational Modes can be used in conjunction with a microcontroller, or they can be hardwired to provide the desired system operation.

M0 – Message Cueing

Message Cueing allows the user to skip through messages, without knowing the actual physical addresses of each message. Each $\overline{CE}$ LOW pulse causes the internal address pointer to skip to the next message. This mode is used for playback only, and is typically used with the M4 Operational Mode.

M1 – Delete $\overline{EOM}$ Markers

The M1 Operational Mode allows sequentially recorded messages to be combined into a single message with only one $\overline{EOM}$ marker set at the end of the final message. When this Operational Mode is configured, messages recorded sequentially are played back as one continuous message.

M2 – Unused

When Operational Modes are selected, the M2 pin should be LOW.

M3 – Message Looping

The M3 Operational Mode allows for the automatic, continuously repeated playback of the message located at the beginning of the address space. A message can completely fill the ISD2500 device and will loop from beginning to end without $\overline{OVF}$ going LOW.

M4 – Consecutive Addressing

During normal operation, the address pointer will reset when a message is played through an $\overline{EOM}$ marker. The M4 Operational Mode inhibits the address pointer reset on $\overline{EOM}$, allowing messages to be played back consecutively.

M5 - $\overline{CE}$-Level Activated

The default mode for ISD2500 devices is for $\overline{CE}$ to be edge-activated on playback and level-activated on record. The M5 Operational Mode causes the $\overline{CE}$ pin to be interpreted as level-activated as opposed to edge-activated during playback. This is especially useful for terminating playback operations using the $\overline{CE}$ signal. In this mode, $\overline{CE}$ LOW begins a playback cycle, at the beginning of the device memory. The playback cycle continues as long as $\overline{CE}$ is held LOW. When $\overline{CE}$ goes HIGH, playback will immediately end. A new $\overline{CE}$ LOW will restart the message from the beginning unless M4 is also HIGH.

M6 – Push-Button Mode

The ISD2500 series contain a Push-Button Operational Mode. The Push-Button Mode is used primarily in very low-cost applications and is designed to minimize external circuitry and components, thereby reducing system cost. In order to configure the device in Push-Button Operational Mode, the two most significant address bits must be HIGH, and the M6 mode pin must also be HIGH. A device in this mode always powers down at the end of each playback or record cycle after $\overline{CE}$ goes HIGH.

When this operational mode is implemented, three of the pins on the device have alternate functionality as described in the table below.

TABLE 3: ALTERNATE FUNCTIONALITY IN PINS

Pin Name	Alternate Functionality in Push-Button Mode
$\overline{CE}$	Start/Pause Push-Button (LOW pulse-activated)
PD	Stop/Reset Push-Button (HIGH pulse-activated)
$\overline{EOM}$	Active-HIGH Run Indicator

$\overline{CE}$ (START/PAUSE)

In Push-Button Operational Mode, $\overline{CE}$ acts as a LOW-going pulse-activated START/PAUSE signal. If no operation is currently in progress, a LOW-going pulse on this signal will initiate a playback or record cycle according to the level on the P/$\overline{R}$ pin. A subsequent pulse on the $\overline{CE}$ pin, before an $\overline{EOM}$ is reached in playback or an overflow condition occurs, will pause the current operation, and the address counter is not reset. Another $\overline{CE}$ pulse will cause the device to continue the operation from the place where it is paused.

PD (STOP/RESET)

In Push-Button Operational Mode, PD acts as a HIGH-going pulse-activated STOP/RESET signal. When a playback or record cycle is in progress and a HIGH-going pulse is observed on PD, the current cycle is terminated and the address pointer is reset to address 0, the beginning of the message space.

$\overline{EOM}$ (RUN)

In Push-Button Operational Mode, $\overline{EOM}$ becomes an active-HIGH RUN signal which can be used to drive an LED or other external device. It is HIGH whenever a record or playback operation is in progress.

Recording in Push-Button Mode

1. The PD pin should be LOW, usually using a pull-down resistor.

2. The P/$\overline{R}$ pin is taken LOW.
3. The $\overline{CE}$ pin is pulsed LOW. Recording starts, $\overline{EOM}$ goes HIGH to indicate an operation in progress.
4. When the $\overline{CE}$ pin is pulsed LOW. Recording pauses, $\overline{EOM}$ goes back LOW. The internal address pointers are not cleared, but the $\overline{EOM}$ marker is stored in memory to indicate as the message end. The P/$\overline{R}$ pin may be taken HIGH at this time. Any subsequent $\overline{CE}$ would start a playback at address 0.
5. The $\overline{CE}$ pin is pulsed LOW. Recording starts at the next address after the previous set $\overline{EOM}$ marker. $\overline{EOM}$ goes back HIGH.[3]
6. When the recording sequences are finished, the final $\overline{CE}$ pulse LOW will end the last record cycle, leaving a set $\overline{EOM}$ marker at the message end. Recording may also be terminated by a HIGH level on PD, which will leave a set $\overline{EOM}$ marker.

Playback in Push-Button Mode

1. The PD pin should be LOW.
2. The P/$\overline{R}$ pin is taken HIGH.
3. The $\overline{CE}$ pin is pulsed LOW. Playback starts, $\overline{EOM}$ goes HIGH to indicate an operation in progress.
4. If the $\overline{CE}$ pin is pulsed LOW or an $\overline{EOM}$ marker is encountered during an operation, the part will pause. The internal address pointers are not cleared, and $\overline{EOM}$ goes back LOW. The P/$\overline{R}$ pin may be changed at this time. A subsequent record operation would not reset the address pointers and the recording would begin where playback ended.
5. $\overline{CE}$ is again pulsed LOW. Playback starts where it left off, with $\overline{EOM}$ going HIGH to indicate an operation in progress.
6. Playback continues as in steps 4 and 5 until PD is pulsed HIGH or overflow occurs.
7. If in overflow, pulling $\overline{CE}$ LOW will reset the address pointer and start playback from the beginning. After a PD pulse, the part is reset to address 0.

Note: Push-Button Mode can be used in conjunction with modes M0, M1, and M3.

[3] If the M1 Operational Mode pin is also HIGH, the just previously written $\overline{EOM}$ bit is erased, and recording starts at that address.

Good Audio Design Practices

Winbond products are very high-quality single-chip voice recording and playback systems. To ensure the highest quality voice reproduction, it is important that good audio design practices on layout and power supply decoupling be followed. See Application Information or below links for details.

Good Audio Design Practices

http://www.winbond-usa.com/products/isd_products/chipcorder/applicationinfo/apin11.pdf

Single-Chip Board Layout Diagrams

http://www.winbond-usa.com/products/isd_products/chipcorder/applicationinfo/apin12.pdf

8. TIMING DIAGRAMS

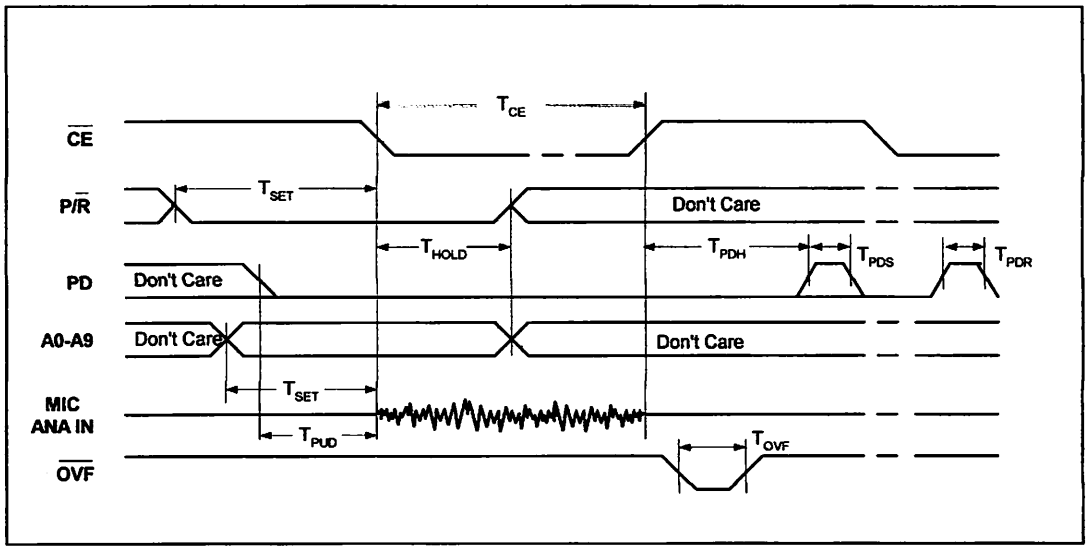


FIGURE 1: RECORD

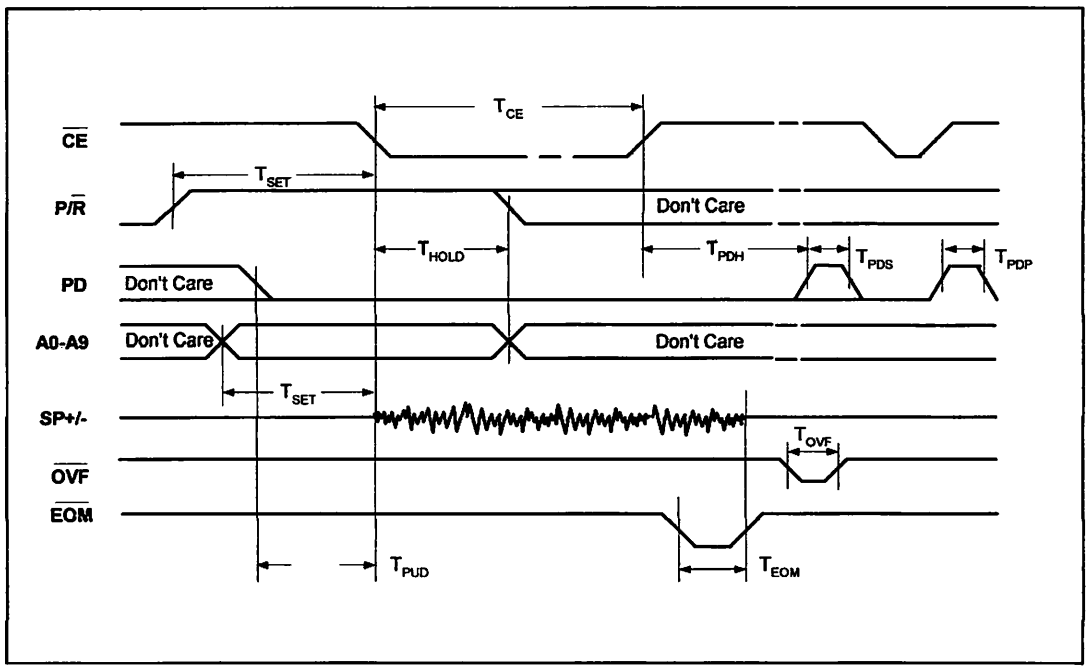


FIGURE 2: PLAYBACK

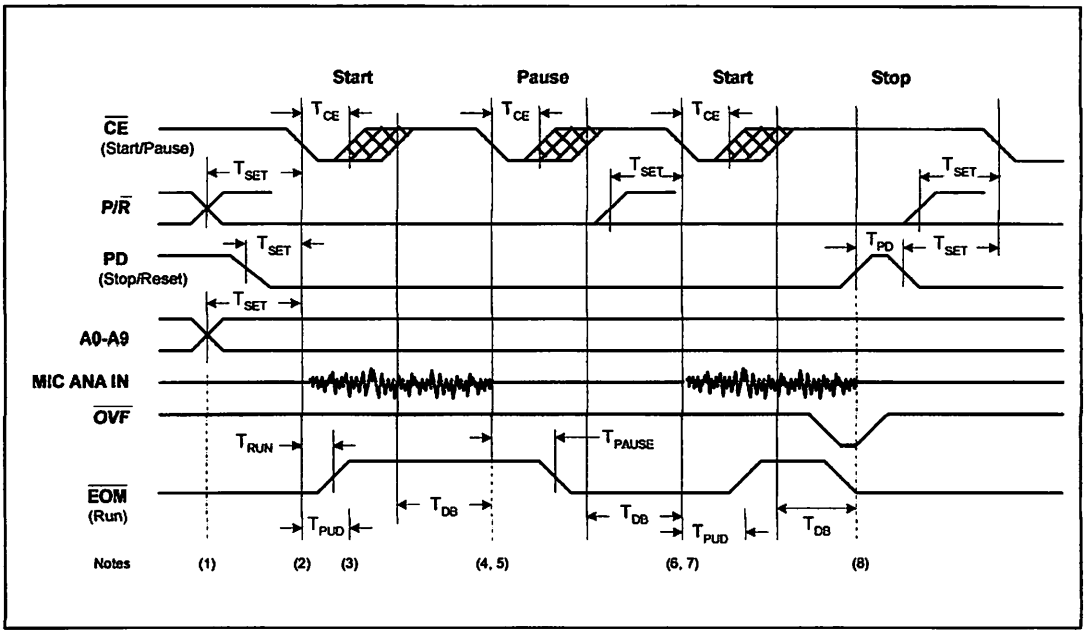


FIGURE 3: PUSH-BUTTON MODE RECORD

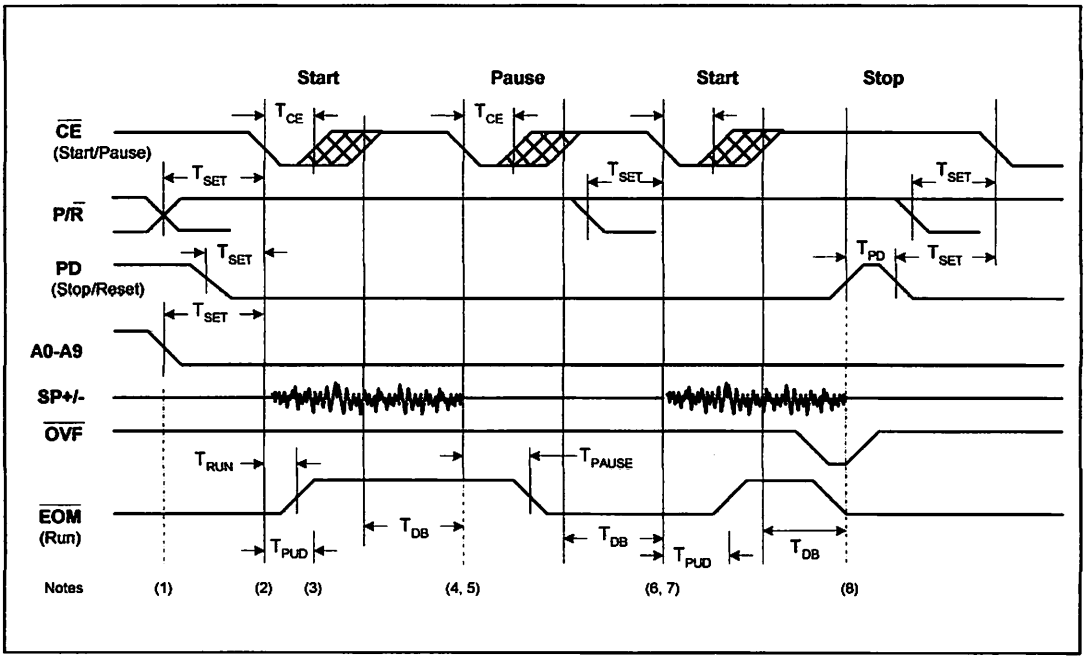


FIGURE 4: PUSH-BUTTON MODE PLAYBACK

Notes for Push-Button modes:

1. A9, A8, and A6 = 1 for push-button operation.
2. The first $\overline{CE}$ LOW pulse performs a start function.
3. The part will begin to play or record after a power-up delay T_{PUD}.
4. The part must have $\overline{CE}$ HIGH for a debounce period T_{DB} before it will recognize another falling edge of $\overline{CE}$ and pause.
5. The second $\overline{CE}$ LOW pulse, and every even pulse thereafter, performs a Pause function.
6. Again, the part must have $\overline{CE}$ HIGH for a debounce period T_{DB} before it will recognize another falling edge of $\overline{CE}$, which would restart an operation. In addition, the part will not do an internal power down until $\overline{CE}$ is HIGH for the T_{DB} time.
7. The third $\overline{CE}$ LOW pulse, and every odd pulse thereafter, performs a Resume function.
8. At any time, a HIGH level on PD will stop the current function, reset the address counter, and power down the device.

9. ABSOLUTE MAXIMUM RATINGS

TABLE 4: ABSOLUTE MAXIMUM RATINGS (DIE)

CONDITION	VALUE
Junction temperature	150°C
Storage temperature range	-65°C to +150°C
Voltage applied to any pad	(V_{SS} –0.3V) to (V_{CC} +0.3V)
Voltage applied to any pad (Input current limited to ±20mA)	(V_{SS} –1.0V) to (V_{CC} +1.0V)
$V_{CC} - V_{SS}$	-0.3V to +7.0V

TABLE 5: ABSOLUTE MAXIMUM RATINGS (PACKAGED PARTS)

CONDITION	VALUE
Junction temperature	150°C
Storage temperature range	-65°C to +150°C
Voltage applied to any pin	(V_{SS} –0.3V) to (V_{CC} +0.3V)
Voltage applied to any pin (Input current limited to ±20 mA)	(V_{SS} –1.0V) to (V_{CC} +1.0V)
Lead temperature (Soldering – 10sec)	300°C
$V_{CC} - V_{SS}$	-0.3V to +7.0V

Note: Stresses above those listed may cause permanent damage to the device. Exposure to the absolute maximum ratings may affect device reliability and performance. Functional operation is not implied at these conditions.

9.1 OPERATING CONDITIONS

TABLE 6: OPERATING CONDITIONS (DIE)

CONDITION	VALUE
Commercial operating temperature range	0°C to +50°C
Supply voltage (V_{CC}) [1]	+4.5V to +6.5V
Ground voltage (V_{SS}) [2]	0V

TABLE 7: OPERATING CONDITIONS (PACKAGED PARTS)

CONDITION	VALUE
Commercial operating temperature range [3]	0°C to +70°C
Supply voltage (V_{CC}) [1]	+4.5V to +5.5V
Ground voltage (V_{SS}) [2]	0V

[1] $V_{CC} = V_{CCA} = V_{CCD}$

[2] $V_{SS} = V_{SSA} = V_{SSD}$

[3] Case Temperature

10. ELECTRICAL CHARACTERISTICS

10.1. PARAMETERS FOR PACKAGED PARTS

TABLE 8: DC PARAMETERS – Packaged Parts

PARAMETER	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
Input Low Voltage	V_{IL}			0.8	V	
Input High Voltage	V_{IH}	2.0			V	
Output Low Voltage	V_{OL}			0.4	V	I_{OL} = 4.0 mA
Output High Voltage	V_{OH}	V_{CC} - 0.4			V	I_{OH} = -10 µA
$\overline{OVF}$ Output High Voltage	V_{OH1}	2.4			V	I_{OH} = -1.6 mA
$\overline{EOM}$ Output High Voltage	V_{OH2}	V_{CC} – 1.0	V_{CC} - 0.8		V	I_{OH} = -3.2 mA
V_{CC} Current (Operating)	I_{CC}		25	30	mA	R_{EXT} = ∞ [3]
V_{CC} Current (Standby)	I_{SB}		1	10	µA	[3]
Input Leakage Current	I_{IL}			±1	µA	
Input Current HIGH w/Pull Down	I_{ILPD}			130	µA	Force V_{CC} [4]
Output Load Impedance	R_{EXT}	16			Ω	Speaker Load
Preamp Input Resistance	R_{MIC}	4	9	15	KΩ	MIC and MIC REF Pins
AUX IN Input Resistance	R_{AUX}	5	11	20	KΩ	
ANA IN Input Resistance	$R_{ANA\ IN}$	2.3	3	5	KΩ	
Preamp Gain 1	A_{PRE1}	21	24	26	dB	AGC = 0.0V
Preamp Gain 2	A_{PRE2}		-15	5	dB	AGC = 2.5V
AUX IN/SP+ Gain	A_{AUX}		0.98	1.0	V/V	
ANA IN to SP+/- Gain	A_{ARP}	21	23	26	dB	
AGC Output Resistance	R_{AGC}	2.5	5	9.5	KΩ	

Notes:

[1] Typical values @ T_A = 25° and V_{CC} = 5.0V.

[2] All Min/Max limits are guaranteed by Winbond via electrical testing or characterization. Not all specifications are 100 percent tested.

[3] V_{CCA} and V_{CCD} connected together.

[4] XCLK pin only.

TABLE 9: AC PARAMETERS – Packaged Parts

CHARACTERISTIC	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
Sampling Frequency	F_S					
ISD2560			8.0		kHz	[7]
ISD2575			6.4		kHz	[7]
ISD2590			5.3		kHz	[7]
ISD25120			4.0		kHz	[7]
Filter Pass Band	F_{CF}					
ISD2560			3.4		kHz	3 dB Roll-Off Point[3][8]
ISD2575			2.7		kHz	3 dB Roll-Off Point[3][8]
ISD2590			2.3		kHz	3 dB Roll-Off Point[3][8]
ISD25120			1.7		kHz	3 dB Roll-Off Point[3][8]
Record Duration	T_{REC}					
ISD2560		58.1	60.0	62.0	sec	Commercial Operation[7]
ISD2575		72.6	75.0	77.5	sec	Commercial Operation[7]
ISD2590		87.1	90.0	93.0	sec	Commercial Operation[7]
ISD25120		116.1	120.0	123.9	sec	Commercial Operation[7]
Playback Duration	T_{PLAY}					
ISD2560		58.1	60.0	62.0	sec	Commercial Operation
ISD2575		72.6	75.0	77.5	sec	Commercial Operation
ISD2590		87.1	90.0	93.0	sec	Commercial Operation
ISD25120		116.1	120.0	123.9	sec	Commercial Operation
$\overline{CE}$ Pulse Width	T_{CE}		100		nsec	
Control/Address Setup Time	T_{SET}		300		nsec	
Control/Address Hold Time	T_{HOLD}		0		nsec	
Power-Up Delay	T_{PUD}					
ISD2560		24.1	25.0	27.8	msec	Commercial Operation
ISD2575		30.2	31.3	34.3	msec	Commercial Operation
ISD2590		36.2	37.5	40.8	msec	Commercial Operation
ISD25120		48.2	50.0	53.6	msec	Commercial Operation
PD Pulse Width (record)	T_{PDR}					
ISD2560			25.0		msec	
ISD2575			31.25		msec	
ISD2590			37.5		msec	
ISD25120			50.0		msec	

TABLE 9: AC PARAMETERS – Packaged Parts (Cont'd)

CHARACTERISTIC	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
PD Pulse Width (Play)	T_{PDP}					
ISD2560			12.5		msec	
ISD2575			15.625		msec	
ISD2590			18.75		msec	
ISD25120			25.0		msec	
PD Pulse Width (Static)	T_{PDS}		100		nsec	[6]
Power Down Hold	T_{PDH}		0		nsec	
$\overline{EOM}$ Pulse Width	T_{EOM}					
ISD2560			12.5		msec	
ISD2575			15.625		msec	
ISD2590			18.75		msec	
ISD25120			25.0		msec	
Overflow Pulse Width	T_{OVF}		6.5		µsec	
Total Harmonic Distortion	THD		1	2	%	@ 1 kHz
Speaker Output Power	P_{OUT}		12.2	50	mW	R_{EXT} = 16 Ω[4]
Voltage Across Speaker Pins	V_{OUT}			2.5	V p-p	R_{EXT} = 600 Ω
MIC Input Voltage	V_{IN1}			20	mV	Peak-to-Peak[5]
ANA IN Input Voltage	V_{IN2}			50	mV	Peak-to-Peak
AUX Input Voltage	V_{IN3}			1.25	V	Peak-to-Peak; R_{EXT} = 16 Ω

Notes:

[1] Typical values @ T_A = 25°C and V_{CC} = 5.0V.

[2] All Min/Max limits are guaranteed by Winbond via electrical testing or characterization. Not all specifications are 100 percent tested.

[3] Low-frequency cutoff depends upon the value of external capacitors (see Pin Descriptions)

[4] From AUX IN; if ANA IN is driven at 50 mV p-p, the P_{OUT} = 12.2 mW, typical.

[5] With 5.1 K Ω series resistor at ANA IN.

[6] T_{PDS} is required during a static condition, typically overflow.

[7] Sampling Frequency and playback Duration can vary as much as ±2.25 percent over the commercial temperature range. For greater stability, an external clock can be utilized (see Pin Descriptions)

[8] *Filter specification applies to the antialiasing filter and the smoothing filter. Therefore, from input to output, expect a 6 dB drop by nature of passing through both filters.*

10.1.1. Typical Parameter Variation with Voltage and Temperature (Packaged Parts)

Chart 1: Record Mode Operating Current (I_{CC})

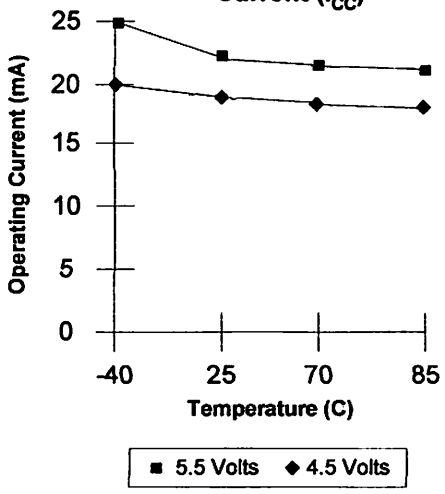


Chart 3: Standby Current (I_{SB})

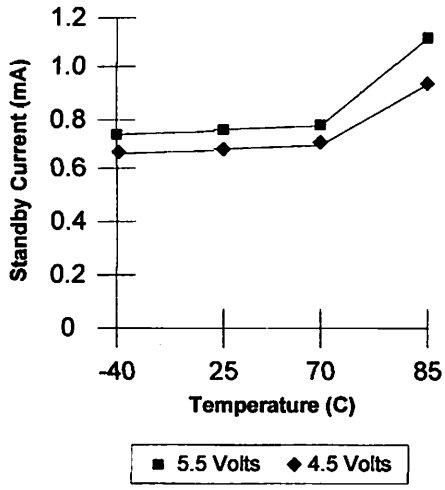


Chart 2: Total Harmonic Distortion

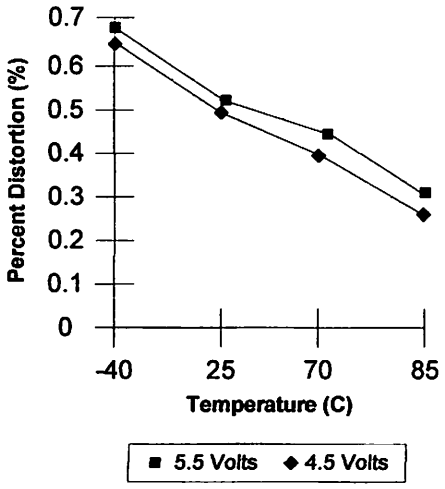
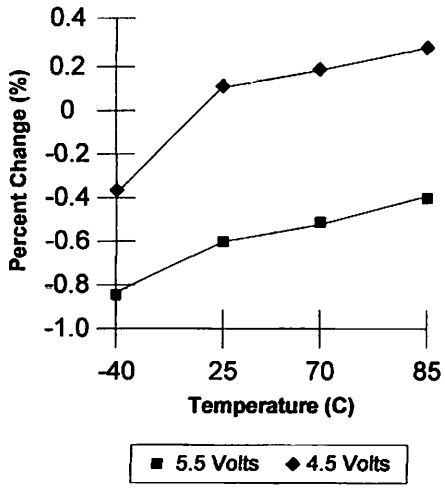


Chart 4: Oscillator Stability



10.2. PARAMETERS FOR DIE

TABLE 10: DC PARAMETERS – Die

PARAMETER	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
Input Low Voltage	V_{IL}			0.8	V	
Input High Voltage	V_{IH}	2.0			V	
Output Low Voltage	V_{OL}			0.4	V	I_{OL} = 4.0 mA
Output High Voltage	V_{OH}	V_{CC} - 0.4			V	I_{OH} = -10 µA
$\overline{OVF}$ Output High Voltage	V_{OH1}	2.4			V	I_{OH} = -1.6 mA
$\overline{EOM}$ Output High Voltage	V_{OH2}	V_{CC} – 1.0	V_{CC} - 0.8		V	I_{OH} = -3.2 mA
V_{CC} Current (Operating)	I_{CC}		25	30	mA	R_{EXT} = ∞ [3]
V_{CC} Current (Standby)	I_{SB}		1	10	µA	[2]
Input Leakage Current	I_{IL}			±1	µA	
Input Current HIGH w/Pull Down	I_{ILPD}			130	µA	Force V_{CC} [4]
Output Load Impedance	R_{EXT}	16			Ω	Speaker Load
Preamp IN Input Resistance	R_{MIC}	4	9	15	KΩ	MIC and MIC REF Pads
AUX IN Input Resistance	R_{AUX}	5	11	20	KΩ	
ANA IN Input Resistance	$R_{ANA\ IN}$	2.3	3	5	KΩ	
Preamp Gain 1	A_{PRE1}	21	24	26	dB	AGC = 0.0V
Preamp Gain 2	A_{PRE2}		-15	5	dB	AGC = 2.5V
AUX IN/SP+ Gain	A_{AUX}		0.98	1.0	V/V	
ANA IN to SP+/- Gain	A_{ARP}	21	23	26	dB	
AGC Output Resistance	R_{AGC}	2.5	5	9.5	KΩ	

Notes:

[1] Typical values @ T_A = 25°C and V_{CC} = 5.0V.

[2] All Min/Max limits are guaranteed by Winbond via electrical testing or characterization. Not all specifications are 100 percent tested.

[3] V_{CCA} and V_{CCD} connected together.

[4] XCLK pad only.

TABLE 11: AC PARAMETERS – Die

CHARACTERISTIC	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
Sampling Frequency	F_S					
ISD2560			8.0		kHz	[7]
ISD2575			6.4		kHz	[7]
ISD2590			5.3		kHz	[7]
ISD25120			4.0		kHz	[7]
Filter Pass Band	F_{CF}					
ISD2560			3.4		kHz	3 dB Roll-Off Point[3][8]
ISD2575			2.7		kHz	3 dB Roll-Off Point[3][8]
ISD2590			2.3		kHz	3 dB Roll-Off Point[3][8]
ISD25120			1.7		kHz	3 dB Roll-Off Point[3][8]
Record Duration	T_{REC}					
ISD2560		58.1	60.0	62.0	sec	Commercial Operation[7]
ISD2575		72.6	75.0	77.5	sec	Commercial Operation[7]
ISD2590		87.1	90.0	93.0	sec	Commercial Operation[7]
ISD25120		116.1	120.0	123.9	sec	Commercial Operation[7]
Playback Duration	T_{PLAY}					
ISD2560		58.1	60.0	62.0	sec	Commercial Operation[7]
ISD2575		72.6	75.0	77.5	sec	Commercial Operation[7]
ISD2590		87.1	90.0	93.0	sec	Commercial Operation[7]
ISD25120		116.1	120.0	123.9	sec	Commercial Operation[7]
$\overline{CE}$ Pulse Width	T_{CE}		100		nsec	
Control/Address Setup Time	T_{SET}		300		nsec	
Control/Address Hold Time	T_{HOLD}		0		nsec	
Power-Up Delay	T_{PUD}					
ISD2560		24.1	25.0	27.8	msec	Commercial Operation
ISD2575		30.2	31.3	34.3	msec	Commercial Operation
ISD2590		36.2	37.5	40.8	msec	Commercial Operation
ISD25120		48.2	50.0	53.6	msec	Commercial Operation
PD Pulse Width (Record)	T_{PDR}					
ISD2560			25.0		msec	
ISD2575			31.25		msec	
ISD2590			37.5		msec	
ISD25120			50.0		msec	

TABLE 11: AC PARAMETERS – Die (Cont'd)

CHARACTERISTIC	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
PD Pulse Width (Play)	T_{PDP}					
ISD2560			12.5		msec	
ISD2575			15.625		msec	
ISD2590			18.75		msec	
ISD25120			25.0		msec	
PD Pulse Width (Static)	T_{PDS}		100		nsec	[6]
Power Down Hold	T_{PDH}		0		nsec	
$\overline{EOM}$ Pulse Width	T_{EOM}					
ISD2560			12.5		msec	
ISD2575			15.625		msec	
ISD2590			18.75		msec	
ISD25120			25.0		msec	
Overflow Pulse Width	T_{OVF}		6.5		µsec	
Total Harmonic Distortion	THD		1	3	%	@ 1 kHz
Speaker Output Power	P_{OUT}		12.2	50	mW	R_{EXT} = 16 Ω[4]
Voltage Across Speaker Pins	V_{OUT}			2.5	V p-p	R_{EXT} = 600 Ω
MIC Input Voltage	V_{IN1}			20	mV	Peak-to-Peak[5]
ANA IN Input Voltage	V_{IN2}			50	mV	Peak-to-Peak
AUX Input Voltage	V_{IN3}			1.25	V	Peak-to-Peak; R_{EXT} = 16 Ω

Notes:

[1] Typical values @ T_A = 25°C and V_{CC} = 5.0V.

[2] All Min/Max limits are guaranteed by Winbond via electrical testing or characterization. Not all specifications are 100 percent tested.

[3] Low-frequency cutoff depends upon the value of external capacitors (see Pin Descriptions)

[4] From AUX IN; if ANA IN is driven at 50 mV p-p, the P_{OUT} = 12.2 mW, typical.

[5] With 5.1 K Ω series resistor at ANA IN.

[6] T_{PDS} is required during a static condition, typically overflow.

[7] Sampling Frequency and playback Duration can vary as much as ±2.25 percent over the commercial temperature range. For greater stability, an external clock can be utilized (see Pin Descriptions)

[8] *Filter specification applies to the antialiasing filter and the smoothing filter. Therefore, from input to output, expect a 6* dB drop by nature of passing through both filters.

10.2.1. Typical Parameter Variation with Voltage and Temperature (Die)

Chart 5: Record Mode Operating Current (I_{CC})

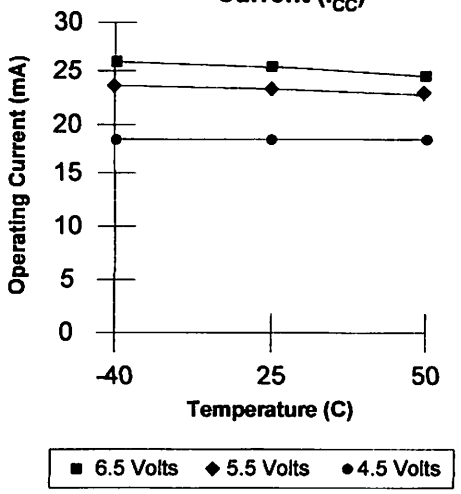


Chart 7: Standby Current (I_{SB})

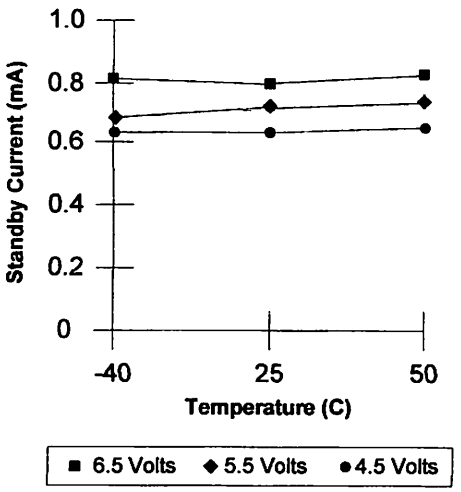


Chart 6: Total Harmonic Distortion

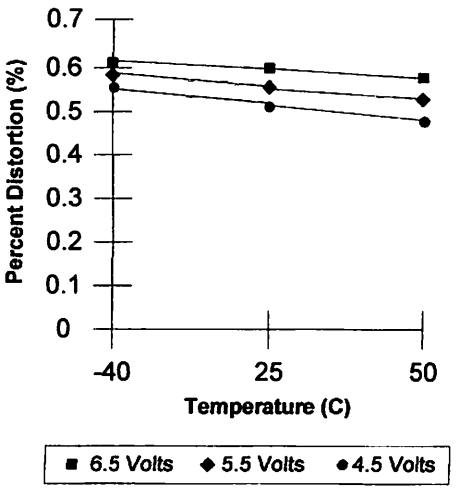
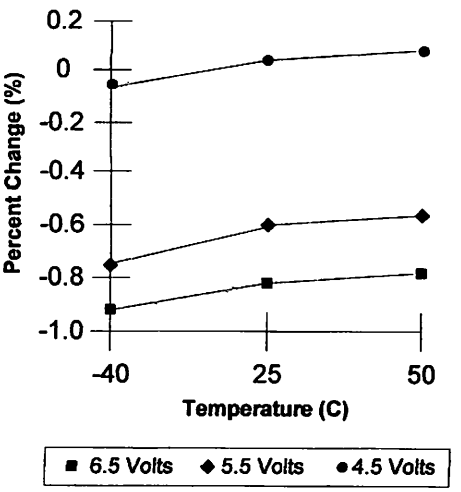


Chart 8: Oscillator Stability



10.3. PARAMETERS FOR PUSH-BUTTON MODE

TABLE 12: PARAMETERS FOR PUSH-BUTTON MODE

PARAMETER	SYMBOL	MIN[2]	TYP[1]	MAX[2]	UNITS	CONDITIONS
$\overline{CE}$ Pulse Width (Start/Pause)	T_{CE}		300		nsec	
Control/Address Setup Time	T_{SET}		300		nsec	
Power-Up Delay	T_{PUD}					
ISD2560			25.0		msec	
ISD2575			31.25		msec	
ISD2590			37.25		msec	
ISD25120			50.0		msec	
PD Pulse Width (Stop/Restart)	T_{PD}		300		nsec	
$\overline{CE}$ to $\overline{EOM}$ HIGH	T_{RUN}	25		400	nsec	
$\overline{CE}$ to $\overline{EOM}$ LOW	T_{PAUSE}	50		400	nsec	
$\overline{CE}$ HIGH Debounce	T_{DB}					
ISD2560		70		105	msec	
ISD2575		85		135	msec	
ISD2590		105		160	msec	
ISD25120		135		215	msec	

Notes:

[1] Typical values @ T_A = 25°C and V_{CC} = 5.0V.

[2] All Min/Max limits are guaranteed by Winbond via electrical testing or characterization. Not all specifications are 100 percent tested.

11. TYPICAL APPLICATION CIRCUIT

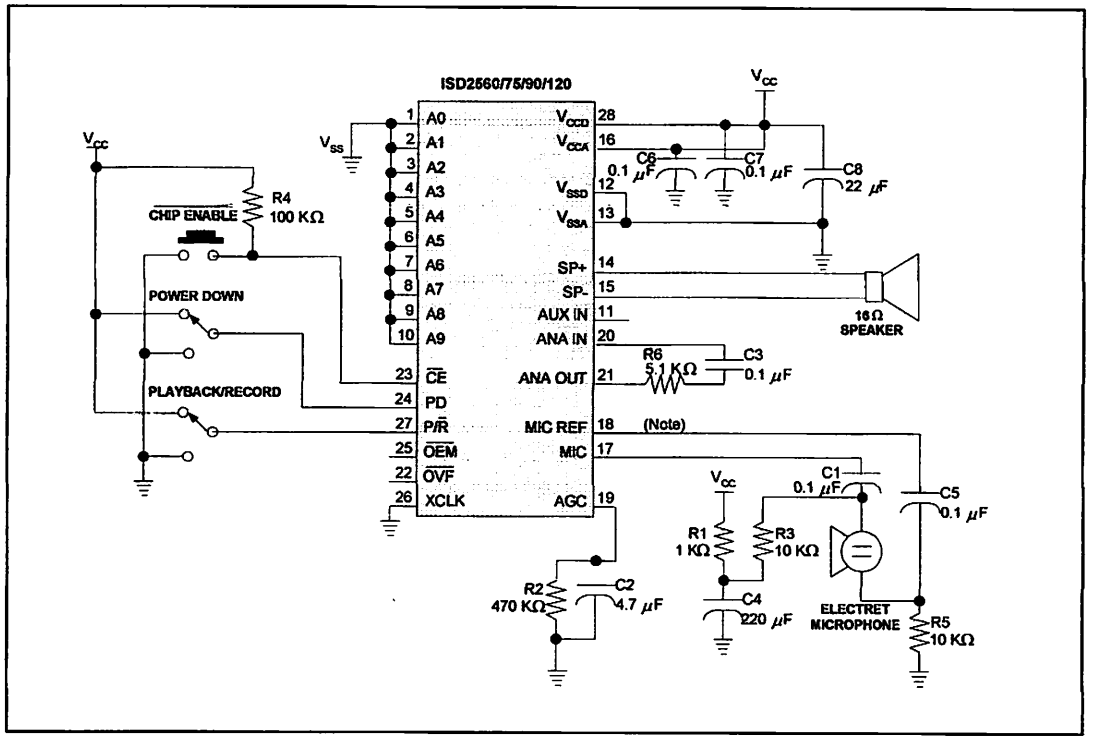


FIGURE 5: DESIGN SCHEMATIC

Note: If desired, pin 18 (PDIP package) may be left unconnected (microphone preamplifier noise will be higher). In this case, pin 18 must not be tied to any other signal or voltage. Additional design example schematics are provided below.

TABLE 13: APPLICATION EXAMPLE – BASIC DEVICE CONTROL

Control Step	Function	Action
1	Power up chip and select Record/Playback Mode	1. PD = LOW, 2. P/$\overline{R}$ = As desired
2	Set message address for record/playback	Set addresses A0-A9
3A	Begin playback	P/$\overline{R}$ = HIGH, $\overline{CE}$ = Pulse LOW
3B	Begin record	P/$\overline{R}$ = LOW, $\overline{CE}$ = LOW
4A	End playback	Automatic
4B	End record	PD or $\overline{CE}$ = HIGH

TABLE 14: APPLICATION EXAMPLE – PASSIVE COMPONENT FUNCTIONS

Part	Function	Comments
R1	Microphone power supply decoupling	Reduces power supply noise
R2	Release time constant	Sets release time for AGC
R3, R5	Microphone biasing resistors	Provides biasing for microphone operation
R4	Series limiting resistor	Reduces level to prevent distortion at higher supply voltages
R6	Series limiting resistor	Reduces level to high supply voltages
C1, C5	Microphone DC-blocking capacitor Low-frequency cutoff	Decouples microphone bias from chip. Provides single-pole low-frequency cutoff and command mode noise rejection.
C2	Attack/Release time constant	Sets attack/release time for AGC
C3	Low-frequency cutoff capacitor	Provides additional pole for low-frequency cutoff
C4	Microphone power supply decoupling	Reduces power supply noise
C6, C7, C8	Power supply capacitors	Filter and bypass of power supply

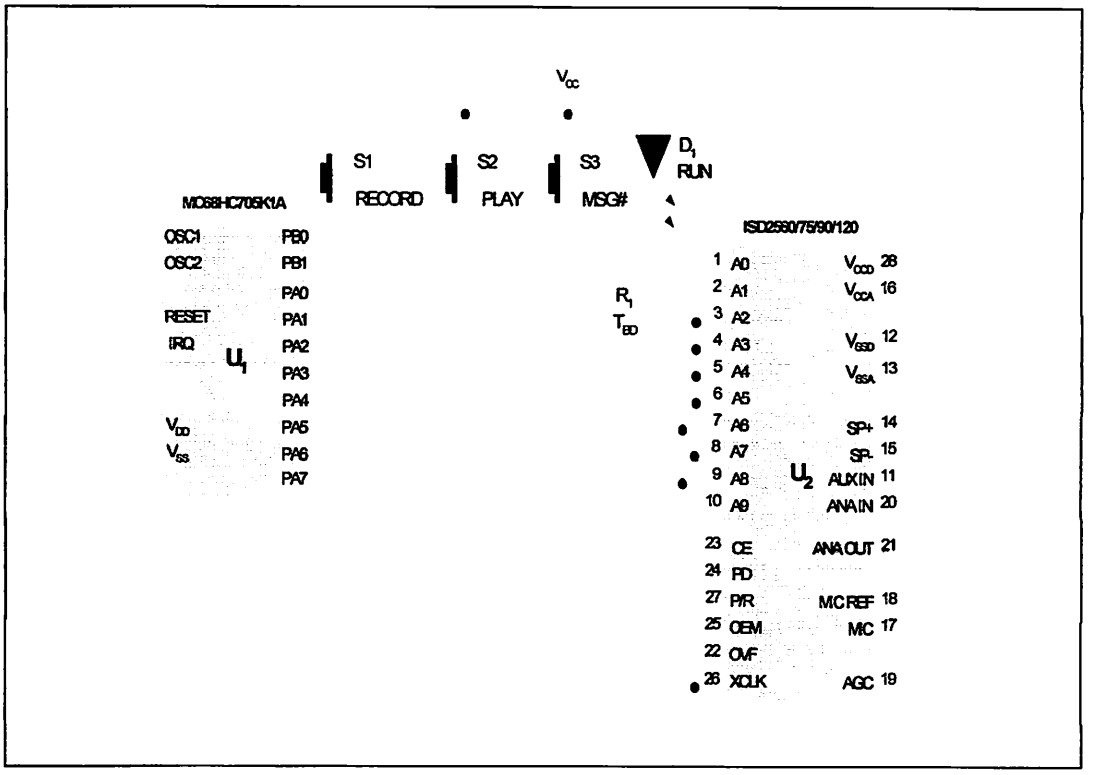


FIGURE 6: ISD2560/75/90/120 APPLICATION EXAMPLE – MICROCONTROLLER/ISD2500 INTERFACE

In this simplified block diagram of a microcontroller application, the Push-Button Mode and message cueing are used. The microcontroller is a 16-pin version with enough port pins for buttons, an LED, and the ISD2500 series device. The software can be written to use three buttons: one each for play and record, and one for message selection. Because the microcontroller is interpreting the buttons and commanding the ISD2500 device, software can be written for any function desired in a particular application.

Note: Winbond does not recommend connecting address lines directly to a microprocessor bus. Address lines should be externally latched.

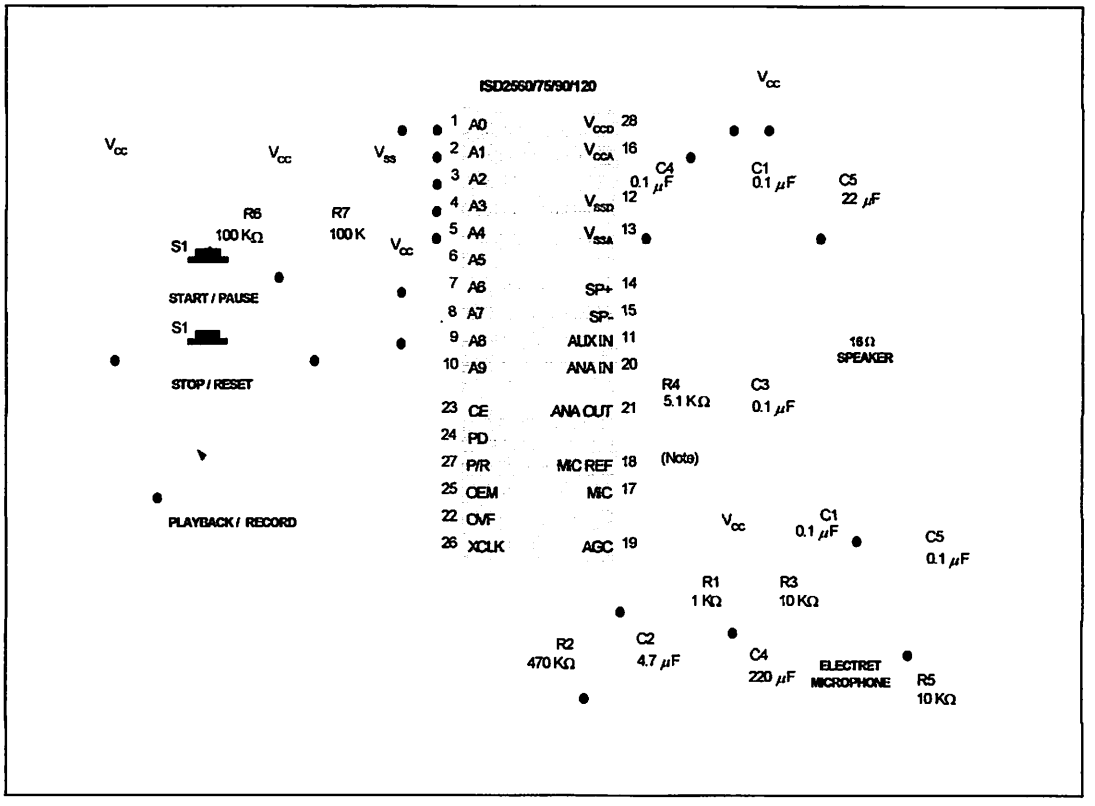


FIGURE 7: ISD2560/75/90/120 APPLICATION EXAMPLE – PUSH-BUTTON

Note: Please refer to page 13 for more details.

TABLE 15: APPLICATION EXAMPLE – PUSH-BUTTON CONTROL

Control Step	Function	Action
1	Select Record/Playback Mode	$P/\overline{R}$ = As desired
2A	Begin playback	$P/\overline{R}$ = HIGH, $\overline{CE}$ = Pulse LOW
2B	Begin record	$P/\overline{R}$ = LOW, $\overline{CE}$ = Pulse LOW
3	Pause record or playback	$\overline{CE}$ = Pulsed LOW
4A	End playback	Automatic at $\overline{EOM}$ marker or PD = Pulsed HIGH
4B	End record	PD = Pulsed HIGH

TABLE 16: APPLICATION EXAMPLE – PASSIVE COMPONENT FUNCTIONS

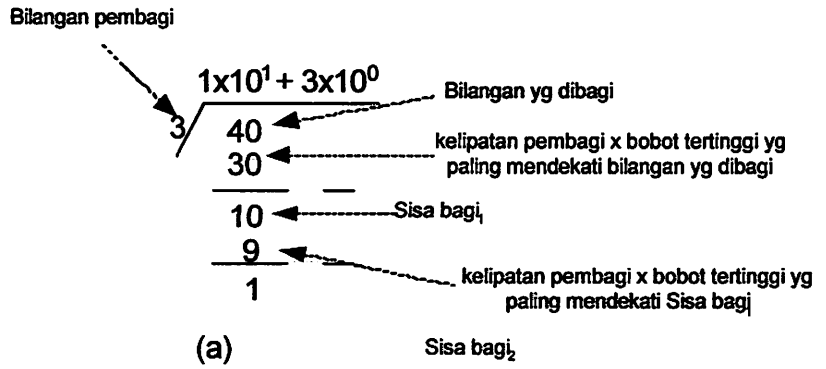
Part	Function	Comments
R2	Release time constant	Sets release time for AGC
R4	Series limiting resistor	Reduces level to prevent distortion at higher supply voltages
R6, R7	Pull-up and pull-down resistors	Defines static state of inputs
C1, C4, C5	Power supply capacitors	Filters and bypass of power supply
C2	Attack/Release time constant	Sets attack/release time for AGC
C3	Low-frequency cutoff capacitor	Provides additional pole for low-frequency cutoff

LOGIKA ARITMATIKA PADA SISTEM BINER

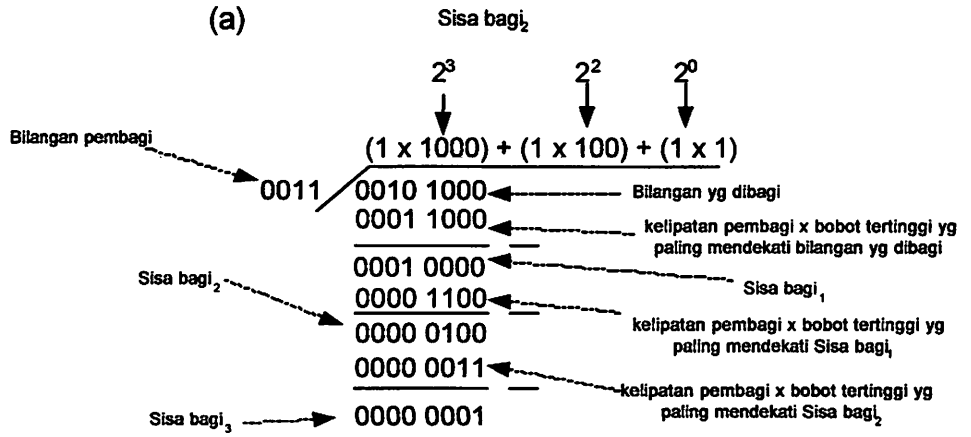
1. Pembagian

Proses pembagian pada sistem biner menggunakan proses pembagian berekor yaitu sebagai berikut:

- Kurangi bilangan yang dibagi dengan kelipatan pembagi yang berbobot tertinggi
- Jumlah kelipatan x bobot yang merupakan hasil $bagi_n$ dijumlahkan dengan hasil $bagi_{n-1}$
- Bila sisa bagi (bilangan yang dibagi – kelipatan pembagi berbobot tertinggi) >= pembagi, lakukan step 1 dan 2 hingga diperoleh kelipatan pembagi berbobot tertinggi < pembagi
- Hasil Bagi dan sisa bagi telah ditemukan



(a)



(b)

Perbedaan antara proses desimal dengan biner adalah bobot bilangannya, bila pada proses desimal bobot bilangan adalah merupakan kepangkatan dari 10, maka pada proses biner, bobot bilangan adalah merupakan kepangkatan dari 2.

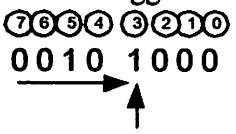
Pada proses pembagian di atas terdapat dua buah proses yaitu

- Pengurangan terhadap kelipatan pembagi berbobot tertinggi
- Penjumlahan hasil bagi dengan hasil bagi sebelumnya setiap kali terjadi pengurangan dengan terhadap kelipatan pembagi berbobot tertinggi terjadi

Pengurangan terhadap kelipatan pembagi berbobot tertinggi

Kelipatan pembagi berbobot tertinggi adalah sebagai berikut:

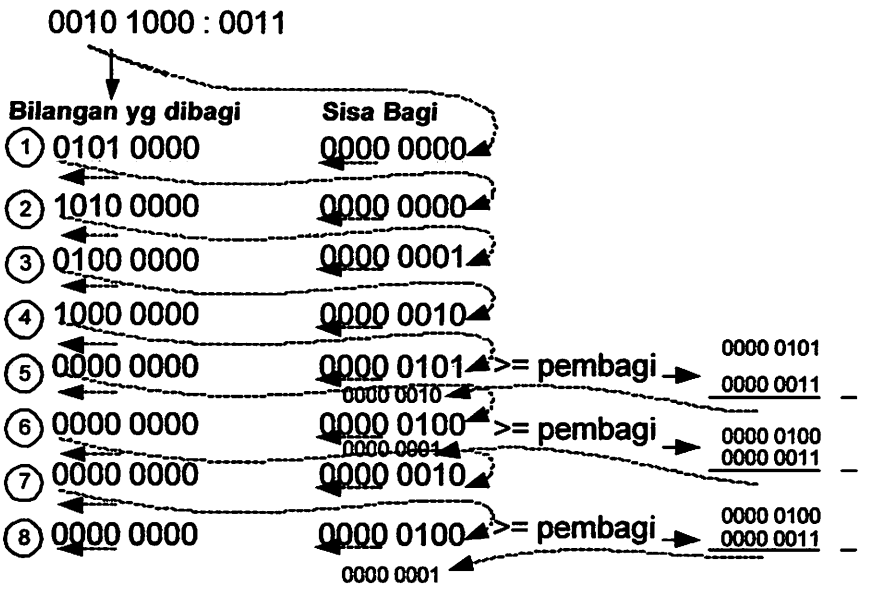
- Pada bilangan yang akan dibagi 40 atau 28H atau 0010 1000 biner dengan pembagi 0011 biner maka kelipatan pembagi
- Apabila sebuah pointer digeser satu per satu dari bagian MSB ke kanan, maka akan ditemukan sebuah bilangan yang >= pembagi saat pointer menunjuk bit ke 3 yaitu 00101 biner yang merupakan bilangan terhitung dari MSB hingga bit 3.



- Bilangan biner 0101 dapat dikurangi dengan bilangan pembagi sebanyak sekali, oleh karena itu kelipatan pembagi adalah 1 x pembagi atau 0011.
- Di belakang bit ketiga masih ada 3 buah bit lagi yaitu bit 2, bit 1 dan bit 0 oleh karena itu bobot dari bilangan adalah kepangkatan 3 dari dasar bilangan atau 2^3
- Di sini telah ditemukan bahwa kelipatan pembagi berbobot tertinggi untuk bilangan 0010 1000 biner dengan pembagi 0011 biner adalah 0011 x 1000 = 0011000 atau 3 x 2^3 = 24

Untuk melakukan proses pengurangan antara bilangan yang dibagi dengan kelipatan pembagi berbobot tertinggi pada mikrokontroler, maka hal ini dapat dilakukan dengan:

- Menggeser ke kiri bilangan yang dibagi ke suatu memori tertentu yang selanjutnya digunakan untuk menyimpan **sisa bagi** sebanyak 8 kali bila bilangan yang diproses hanya menempati **area 8 bit.**
- Mengurangi isi memori sisa bagi dengan bilangan pembagi saat ditemukan isi memori sisa bagi >= bilangan pembagi
- Isi memori sisa bagi adalah merupakan pengurangan dari bilangan yang dibagi dengan kelipatan pembagi berbobot tertinggi
- Contoh:



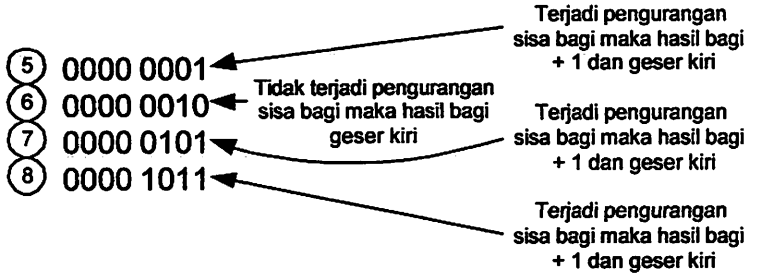
Gambar 1
Proses pengurangan yang terjadi pada sistem pembagian berekor

- Pada proses di atas, bila setelah ditemukan isi memori sisa bagi >= bilangan pembagi (0011) yang pertama kali pergeseran terus dilakukan tanpa melakukan pengurangan dengan pembagi pada step-step berikutnya maka, proses ini sudah merupakan pengurangan **bilangan yang dibagi** dengan **kelipatan pembagi berbobot tertinggi**, hal ini disebabkan karena:
  - **i.** Pada saat isi memori sisa bagi dikurangi dengan pembagi maka proses pengurangan antara bilangan yang dibagi dengan kelipatan pembagi telah dilakukan.
  - **ii.** Pergeseran yang dilakukan pada step-step berikutnya adalah merupakan perkalian hasil pada step i dengan bobot tertinggi yaitu dalam contoh di atas adalah 2^3
- Namun pada sistem pembagian berekor, proses pengurangan terhadap kelipatan pembagi berbobot tertinggi terus diulang hingga ditemukan sisa dari pembagian < bilangan pembagi. Saat sisa dari pembagian masih >= bilangan pembagi, maka Sisa Pembagian tersebut harus selalu dikurangi dengan kelipatan pembagi berbobot tertinggi. Hal ini dapat dilakukan dengan melakukan pengurangan terhadap bilangan pembagi setiap kali ditemukan isi memori sisa bagi >= bilangan pembagi.
- Seperti yang tampak pada contoh di atas, proses pengurangan pada memori sisa bagi juga berlangsung pada step 6 dan step 8. Pada step-step tersebut isi memori sisa bagi dikurangi dengan kelipatan pembagi berbobot tertinggi saat itu. Contohnya pada step 6, isi memori sisa bagi dikurangi dengan 0011 x 0100 atau 3 x 2^2 di mana 3 adalah kelipatan pembagi dan 2^2 adalah bobot tertinggi dari kelipatan pembagi yang diperoleh dengan sisa bagi yang ada saat itu.

Penjumlahan hasil bagi dengan hasil bagi sebelumnya setiap kali terjadi pengurangan dengan terhadap kelipatan pembagi berbobot tertinggi terjadi
Dalam mikrokontroler, hal ini dapat dilakukan dengan:

- Menggeser ke kiri isi memori hasil bagi sebanyak 8 x bila bilangan yang diproses hanya menempati area **8 bit**.
- Menambah satu (increment) memori hasil bagi setiap kali ditemukan isi memori sisa bagi >= bilangan pembagi.

Seperti yang terlihat pada gambar 1, proses pengurangan sisa bagi (karena memori sisa bagi >= bilangan pembagi) terjadi pada step 5, step 6 dan step 8, maka hanya pada step-step tersebut saja terjadi increment dan pergeseran ke kiri sedangkan pada step 7 hanya dilakukan pergeseran ke kiri seperti pada gambar 2.



Gambar 2
Penjumlahan Hasil Bagi

Pada step 1 hingga 4 sebetulnya juga terdapat proses pergeseran ke kiri, namun pada saat itu isi memori hasil bagi hanyalah 0000 0000 saja sehingga proses pergeseran tidak mengubah isi memori tersebut.

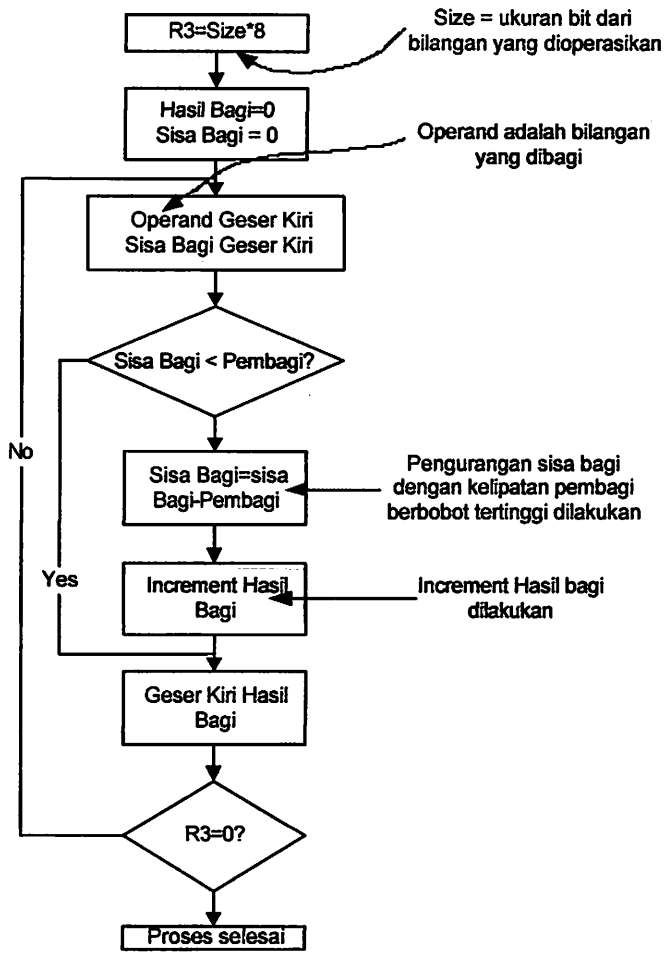
Pada step 5 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 3x hingga akhir step, hal ini ekivalen dengan penambahan 1 x 1000 atau 1 x 2^3

Pada step 6 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 2x hingga akhir step, hal ini ekivalen dengan penambahan 1 x 100 atau 1 x 2^2

Pada step 5 terlihat terdapat penambahan angka 1 dan selanjutnya angka tersebut terus bergeser ke kiri sebanyak 0x atau tidak bergeser hingga akhir step, hal ini ekivalen dengan penambahan 1 x 1 atau 1 x 2^0

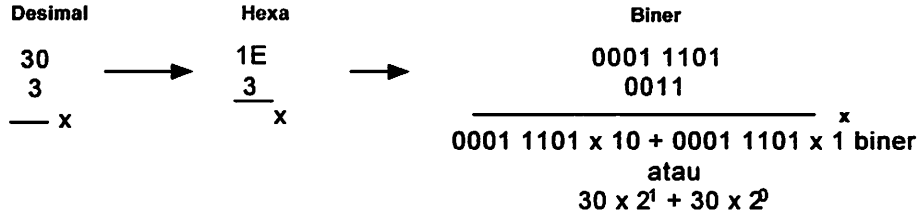
Maka total penjumlahan hasil bagi adalah 1 x 1000 + 1 x 100 + 1 x 1 = 1011 biner

Alur Program dalam Mikrokontroler

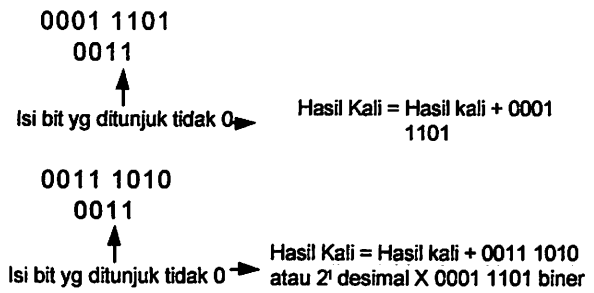


2. Perkalian

Proses perkalian dilakukan dengan menggunakan mengalikan bilangan yang akan dikali dengan masing-masing bobot dari pengali.



Gambar 3
Proses perkalian

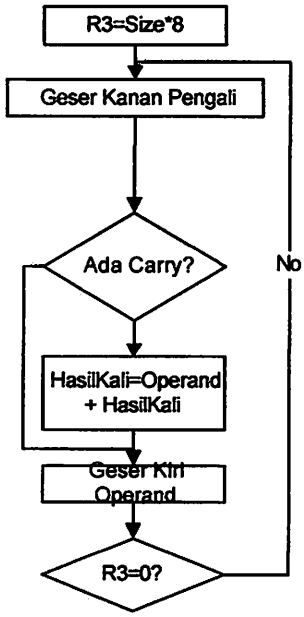


Gambar 4
Proses logika perkalian

Pada gambar di atas tampak sebuah pointer pada bilangan pengali bergeser ke kiri sedangkan bilangan yang akan dikali bergeser ke kiri saat step kedua (untuk mengali dengan 2^1)

Pada mikrokontroler pergeseran ke kiri pointer pada bilangan pengali dapat dilakukan dengan menggeser ke kanan bilangan pengali beserta *carry flag*, sedangkan pergeseran ke kiri bilangan yang dikali tetap dilakukan dengan menggeser ke kiri isi memori bilangan yang akan dikali.

Oleh karena itu, proses perkalian dapat dilakukan dengan:
- Menggeser ke kanan bilangan pengali beserta carry
- Setiap kali carry flag = 1, di mana ekivalen dengan pointer menunjuk ke bit yang bukan 0 (gambar 4), maka jumlahkan bilangan yang dikali dengan hasil kali
- Geser kiri bilangan yang akan dikali baik terjadi atau tidak terjadi carry.
- Proses dilakukan sebanyak jumlah bit dari bilangan yang diproses.
- Hasil perkalian akan tersimpan di memori hasil kali



Five-Minute Introduction to Braille

Braille is a system of transcribing print so it can be read by touch. Braille is now mainly used by blind people but the original idea was for soldiers to be able to read at night without putting themselves in danger by using any light. You can learn about braille by reading this page and following the links.

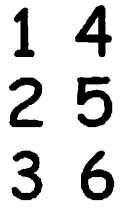


Fig. 1. *Dot positions.*

Cells.

The characters of the standard braille alphabet are called cells and consist of dots placed on a two-column grid with three positions in each column; there are 63 different cells not counting the space. The positions are normally numbered starting at the top of the left-hand column as shown in Figure 1.

The two main forms of tactile braille are embossed paper braille and refreshable braille displays (RBDs) in which an electronic signal results in pins moving up and down to make a row of cells. Braille readers use RBDs as computer monitors.

Codes.

A natural question is what the braille cells mean. However, the cells have no intrinsic meanings; since there is only one standard braille alphabet, the cells mean different things depending on which braille code is in use: math, music, Japanese, etc. The most common code in the United States is literary braille or, more accurately, standard English Braille American Edition (EBAE).

Braille is not just one-to-one with print and—even within a single code—the same cells are re-used to have different meanings in different contexts. For example, the cell for the letter "p" in literary braille means "people" when it stands alone as a word. (The use of context-dependent shorthand is one reason why transcribing from print to braille isn't straightforward; much transcribing today is done with computer applications, cf. Duxbury Systems, Inc.)

The braille shorthand symbols are called contractions; in addition to whole-word contractions, there are also numerous part-word contractions. The rules for using contractions aren't simple; one needs to either look them up in a reference or learn them from a source like Braille through Remote Learning. In addition to contractions, the other unique aspect of braille codes are their use of HTML-like embedded markup or composition indicators; for example, when the cell with a single dot in position 6 is placed before a letter cell in literary braille, it means that the letter is capitalized.

Memorizing the dots.

One way to learn the alphabet in literary braille is to memorize the dot patterns for the first ten letters, a-j, shown by the simulated or inkprint braille cells in Figure 2.

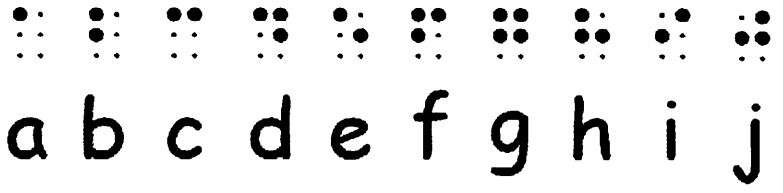


Fig. 2. *Braille cells a-j in alphabetical order.*

(The shadow dots in empty positions are for sighted persons and are not used in embossed braille.)

The dot patterns for the next ten letters, k-t, are the same as the first ten but with an additional dot in position 3. The dot patterns for the letters u,v,x,y, and z are the same as the letters a-e with additional dots in positions 3 and 6. The letter "w", dot pattern 2-4-5-6, is out of alphabetical order because the French alphabet did not have that letter when Louis Braille invented the braille alphabet in 1829.

All 63 braille cells are shown with their literary braille identifications on our Find-A-Cell chart. The Find-A-Cell chart can be used to quickly locate the meaning of an unknown dot pattern. (Note that a more common way of displaying the braille cells is the so-called *standard order* with five rows of ten cells followed by one row of six and one of seven as illustrated by the Braille ASCII chart.)

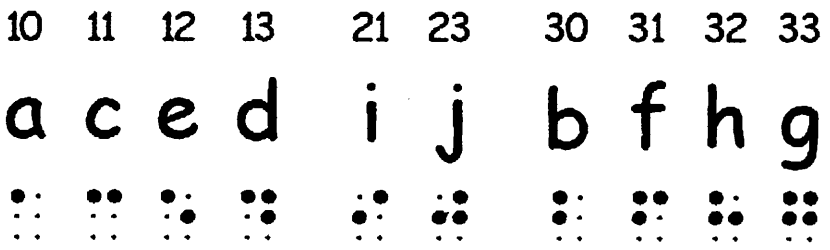


Fig. 3. *Braille cells a-j in numerical order by their NUMBRL codes.*

A new method.

Figure 3 shows a different arrangement of the first ten cells that you may find makes it easier to memorize their dot patterns.

Dotless Braille Tip! Note that the dot patterns of the braille cells in the sequence "aced" all have a dot in dot position one and a simple progression of zero, one, or two dots in dot positions four and five of the right-hand column of the cells. The dot patterns of the cells in the sequence "bfhg" are the same as the corresponding cells in the "aced" sequence except for the addition of a dot in dot position two while the dot patterns for "ij" are the same as for "cd" except for a dot in position two instead of position one.

If you are familiar with the binary number system, you may have realized that the order in Figure 3 is based on interpreting the braille cells as binary-coded numbers that use filled positions for ones and empty positions for zeroes. This means that you can use the numbers to reconstruct the dot patterns. The octal numerical values used in Figure 3 are based on our particularly useful NUMBRL assignment of place values to the dot positions.

A (Slightly) Different Introduction to Braille

One purpose of this site is to present a systematic, top-down description of the structure of braille as an aid to understanding for persons who learn better with a top-down approach, rather than with the more common bottom-up or detail-oriented approach. However, there are also lots of **tips** that would be useful for bottom-up learning.

The site is designed to be self-contained but does omit detail in areas that are well-covered by other sources; if you want to learn to read and write braille dots, this is a good place to start but, as clearly indicated, you will eventually need to go to other sources. The annotated links are a good starting point.

We hope that our unique summaries will demonstrate that there is more than one way to learn braille and will provide a useful guide to choosing the way that is likely to work best for you. All our pages are carefully cross-linked so you should be able to start wherever you choose.

- Five-Minute Introduction to Braille Go.
- Links to Official Braille Standards and Courses
- Cells versus Codes: Braille Basics
    - Cells.
        - Dot patterns.
        - Lower cell dot patterns.

        - Upper cell dot patterns.
        - Extended character sets.
    - Codes.
        - Literary braille.
        - Nemeth.
- Braille Contractions
- Braille Indicators

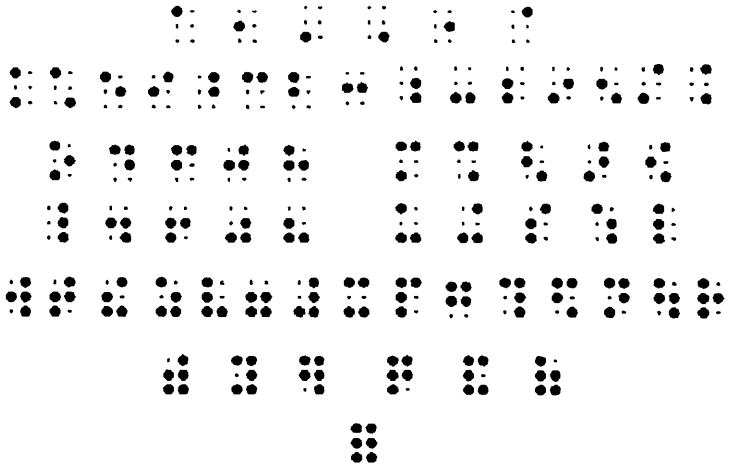
Cells versus Codes: Braille Basics

What is braille? The word *braille* has two different meanings; it can refer either to a braille alphabet or to one of the many braille codes.

Cells

What is a braille alphabet? A braille alphabet is a set of characters or cells designed for reading by touch. The standard or six-dot braille alphabet--which is the one described on this site--is the one usually intended unless explicitly indicated otherwise. (There are also extended character sets.)

Standard cells have one to six dots. The dot positions in a cell are arranged in two parallel columns of three positions each. The alphabet is composed of all 63 possible braille dot patterns that can be created from the different arrangements of one to six filled positions.



The adjacent picture shows the cells arranged in rows according to the number of filled positions. There are six different cells with one dot and six with five dots, fifteen each with two dots and four dots, and twenty cells with three dots--which are displayed in two rows. There is, of course, only one cell with six dots.

A braille cell or sequence of braille cells that has a single meaning, such as two cells that create an uppercase letter, is generally called a braille *symbol*; the print character or sequence of characters that are transcribed by a braille symbol is called a *sign*. Although this useage is standard in braille, it is somewhat at odds with the terminology of semiotics and can be confusing to the novice.

What is a dot pattern? A dot pattern is the arrangement of dots that make up a particular braille cell. The most widely-used method of specifying a dot pattern is to list the position numbers of the dots. The standard numbering system for dot positions numbers the positions for the left-hand column from 1-3 starting at the top of the column and from 4-6 for the right-hand column, again starting at the top of the column. For example, the cell with all six positions filled is called dot pattern 1-2-3-4-5-6. The ALT parameters displayed when hovering over the dots in the logo at the top of our main page use this standard system.

What is the literary braille code? The literary or English braille code, sometimes called Braille 2 or contracted braille, is the most widely-used code in the United States for transcribing printed material using the standard, six-dot braille alphabet and is the one assumed here unless otherwise specified. (There is also an uncontracted form of English braille that is sometimes called Braille 1.)

The literary code uses single cells as symbols for letters, part-words such as 'th' and 'ing', whole words such as 'and', punctuation marks, and indicators. Most of the single cells have additional context-dependent semantics including use as full-word and part-word contractions. The literary code also makes use of multiple-cell sequences for contractions.

What is the Nemeth braille code? The Nemeth code, originally developed by Dr. Abraham Nemeth around 1950, is used for transcribing mathematical expressions. Nemeth is used in the United States for elementary math textbooks as well as highly technical material. (There are other braille mathematics codes in use in other countries.)

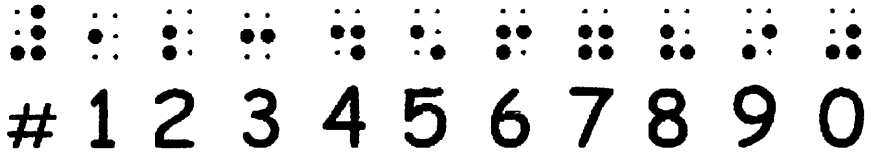


Figure. Nemeth numerals.

Nemeth is essentially a superset of a slightly modified literary braille. The most notable modification to literary braille is for numerals, which use the corresponding lower-cell dot patterns rather than the upper-cell ones. Since the lower-cell dot patterns are the same as the punctuation marks, Nemeth requires the use of a Punctuation Indicator for changing back to that symbol set as well as a Numeric Indicator.

The Nemeth code is intended to reflect the appearance, presentation, or syntax of mathematics more than its semantics. This allows a braille transcriber to transcribe mathematical texts with little knowledge of the underlying mathematics. Further discussion of the relationship between this characteristic of the Nemeth code and transcribing issues is on the page on transcribing.

Lots of additional information about the Nemeth code has just (February 2002) been posted. Start here to learn more about Nemeth.

This page was last changed March 2, 2002.

US extended computer Braille

Later an extended version was introduced with the use of dot 7, the standard only encodes the printable characters from ASCII. The symbols that show an ellipsis where undefined.