SKRIPSI

PERENCANAAN DAN PEMBUATAN PEDOMETER DIGITAL DISERTAI DENGAN PENGHITUNG JARAK DENGAN MENGGUNAKAN IC ADXL202 BERBASIS MIKROKONTROLLER ATMEGA 8

Disusun Oleh :

MUHAMMAD BALI SURYO UTOMO
NIM : 0317034

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

APRIL 2008

LEMBAR PERSETUJUAN

PERENCANAAN DAN PEMBUATAN PEDOMETER DIGITAL DISERTAI DENGAN PENGHITUNG JARAK DENGAN MENGGUNAKAN IC ADXL202 BERBASIS MIKROKONTROLLER ATMEGA8

SKRIPSI

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

Disusun Oleh :

M. Bali Suryo Utomo

NIM : 03.17.034

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
Ir. Teguh Herbasuki, MT	**I Komang Somawirata, ST, MT.**
NIP.Y 1038900209	**NIP. P 1030100361**

Mengetahui

Ketua Jurusan Teknik Elektro S-1

Ir. F. Yudi Limpraptono, MT

NIP.Y 1039500274

KONSENTRASI TEKNIK ELEKTRONIKA
JURUSAN TEKNIK ELEKTRO S-1
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2008

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

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama : M. Bali Suryo Utomo

NIM : 03.17.034

Jurusan : Teknik Elektro S-1

Konsentrasi : Teknik Elektronika

Judul Skripsi : Perencanaan dan Pembuatan pedometer digital disertai dengan penghitung jarak dengan menggunakan ic ADXL202 berbasis mikrokontroller ATMEGA8

Dipertahankan di hadapan majelis penguji Skripsi jenjang Strata satu (S-1) pada :

Hari : Selasa

Tanggal : 18 Maret 2008

Dengan Nilai : 83

Ketua Majelis Penguji

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

Sekretaris Majelis Penguji

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

Penguji I

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

Penguji II

(Dr. Cahyo Crysdian, MSc.)
NIP.Y. 1030400412

ABSTRAKSI

PERENCANAAN DAN PEMBUATAN PEDOMETER DIGITAL DISERTAI DENGAN PENGHITUNG JARAK MENGGUNAKAN IC ADXL202 BERBASIS MIKROKONTROLLER ATMEGA 8

M. Bali Suryo Utomo
03.17.034
Jurusan Teknik Elektronika – Institut Teknologi Nasional Malang
Jln. Raya Karanglo Km 2 Malang
Bali_malang@yahoo.com

Dosen Pembimbing : I. Ir. Teguh Herbasuki,MT.
II. I Komang Somawirata,ST,MT.

Kata Kunci : *ADXL202,Mikrokontroller atmega8,akslerasi sensor*

Salah satu gerak badan yang paling mudah semua orang bisa melakukannya adalah berjalan, Penelitian menunjukkan bahwa orang yang berjalan kira-kira 30-35 mil per minggu atau 10.000 langkah perhari dapat membuat manusia lebih panjang umur beberapa tahun dibanding mereka yang tidak, Jalan kaki juga memberikan manfaat seperti Memperbaiki efektivitas jantung dan paru-paru.

Dari pemikiran tersebut sehingga timbul ide untuk membuat suatu alat pedometer yang dapat memberikan infomasi dengan cepat kepada penggunanya berapa langkah dan jarak yang telah di lakukan sehingga mengetahui berapa besar besar energy yang telah dikeluarkan.

Pedometer memiliki prinsip kerja ketika seorang berjalan maka terdapat pergerakan vertical pada tubuh dalam setiap langkahnya sensor ADXL202 adalah sensor yang sensitive terhadap gerakan vertical sehingga dengan menggunakan rumus tertentu maka langkah dan jarak dapat ditentukan

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat-Mu Ya Allah yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat menyelesaikan skripsi yang berjudul " Perancangan dan pembuatan alat pedometer digital disertai dengan penghitung jarak menggunakan ic ADXL202 berbasis mikrokontroler atmega 8" **ini dengan lancar. Skripsi ini merupakan persyaratan kelulusan Studi di Jurusan Teknik Elektro S-1 Konsentrasi Teknik Elektronika ITN Malang dan untuk mencapai gelar Sarjana Teknik.**

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

1. Bapak Ir. F. Yudi Limpraptono, MT selaku Ketua Jurusan Teknik Elektro S-1.
2. Bapak Ir. Teguh Herbasuki, MT selaku Dosen Pembimbing I dan Ka. Laboratorium pengukuran.
3. Bapak I Komang Somawirata,ST,MT selaku Dosen Pembimbing II dan Ka. Laboratorium elektronika analog.
4. Ayah dan Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
5. Rekan-rekan Instruktur di Laboratorium rangkaian elektrik
6. Semua yang telah membantu dalam penyelesaian penyusunan skripsi ini.

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

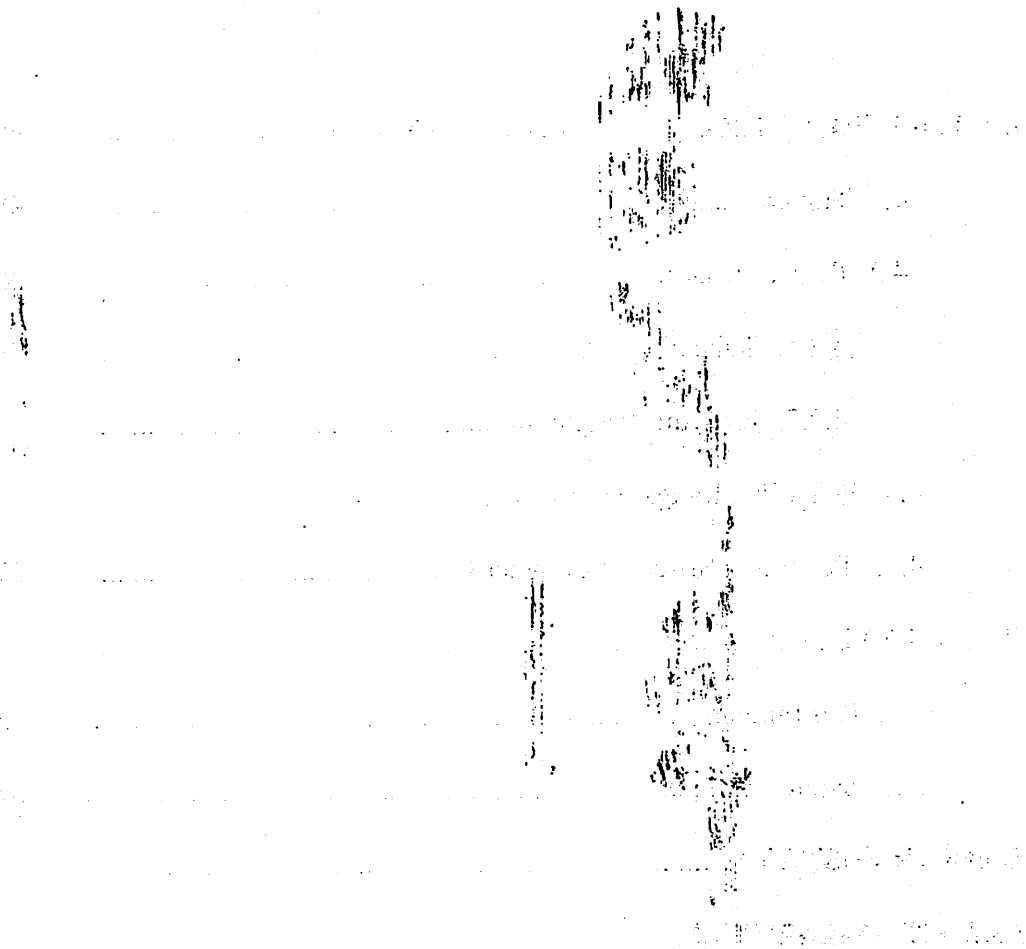
penyusun mengharapkan saran dan kritik yang membangun dari pembaca demi kesempurnaan laporan ini.

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

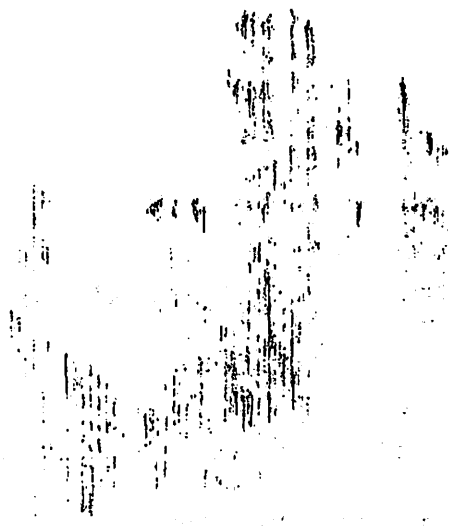
Malang, Maret 2008

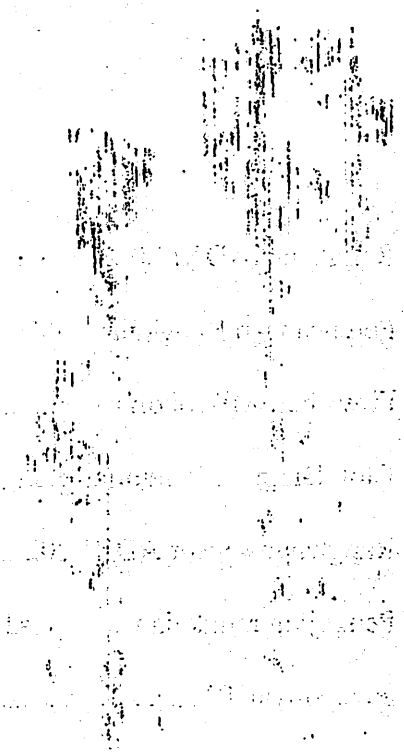
Penyusun

DAFTAR ISI

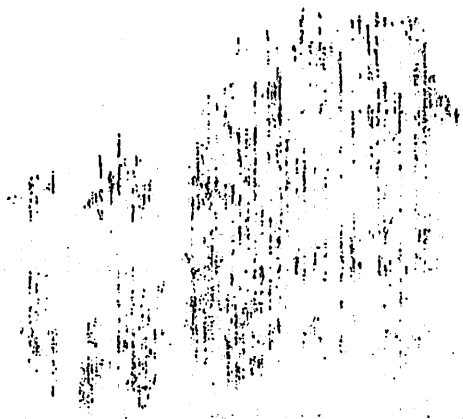


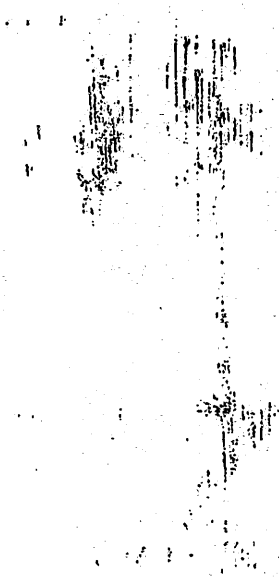
DAFTAR GAMBAR





DAFTAR TABEL





BAB I

PENDAHULUAN

1.1. Latar Belakang

Manusia mempunyai sekitar 650 otot tubuh berarti 650 motor yang memberikan kemampuan untuk bergerak, Otot-otot ini jika tidak digunakan akan kehilangan kemampuannya dan berkurang ukurannya dan jika itu tidak aktif untuk jangka waktu yang cukup lama maka akan berhenti berfungsi semuanya karena itu gerak badan penting salah satu gerak badan yang paling mudah semua orang bisa melakukannya adalah berjalan melakukan jalan kaki secara rutin dan konsisten adalah salah satu faktor terpenting dalam membentuk program aktivitas fisik yang sehat. Penelitian menunjukkan bahwa orang yang berjalan kira-kira 30-35 mil per minggu atau 10.000 langkah perhari dapat membuat manusia lebih panjang umur beberapa tahun dibanding mereka yang tidak. Jalan kaki juga memberikan manfaat seperti memperbaiki efektivitas jantung dan paru-paru,membantu menyembuhkan stress,menurunkan tingkat kolesterol dalam darah,menurunkan tingkat darah tinggi,membantu mengontrol dan mencegah diabetes, dan menurunkan beberapa resiko kanker seperti kanker prostat dan payudara. Sementara gerak badan yang terlalu banyak dapat melemahkan tubuh karena T-sel (antibody) tertekan oleh tingginya kadar adrenalin dan kortikosteroid (hormon stress) yang dihasilkan oleh latihan gerak badan itu.

Dalam mengantisipasi kondisi seperti kasus diatas banyak cara yang dapat dilakukan manusia untuk mengatasi hal-hal tersebut. Dengan makin berkembangnya teknologi pada saat ini berkembang pula suatu sistem teknologi. Pedometer adalah salah satu contoh alat yang dapat memberikan informasi dengan cepat kepada pengguna tentang berapa langkah dan jarak yang telah dia lakukan agar tidak terjadinya 'over trainning' sehingga pengguna dapat mengetahui seberapa besar energi yang telah dia keluarkan.

Demikian sehingga timbul ide untuk membuat suatu alat pedometer yang memiliki keakuratan tinggi . Pedometer memiliki prinsip kerja ketika seorang berjalan maka terdapat pergerakan vertical pada tubuh dalam setiap langkahnya, banyak cara yang digunakan untuk mendeteksi pergerakan vertical seperti salah satu pedometer yang ada dipasaran menggunakan bandul yang terdapat didalam rangkaian sebagai pendeteksi langkah sedangkan pedometer yang penulis menggunakan sensor yang sensitive terhadap kemiringan vertical dan horizontal dan memiliki keluaran percepatan sehingga dapat menentukan jarak dan langkah dengan menggunakan rumus tertentu.

1.2 Tujuan

Penulisan skripsi ini bertujuan merancang dan membuat alat pedometer digital disertai dengan pengukur jarak dengan menggunakan sensor kemiringan yang menggunakan ic ADXl202 berbasis mikrokontroller ATMEGA 8 sebagai pengolah datanya dengan tampilan LCD.

1.3 Rumusan Masalah

Mengacu pada permasalahan yang diuraikan dalam latar belakang, maka rumusan masalah dapat ditekankan pada:

1. Bagaimana mendesain dan membuat suatu sistem berbasis mikrokontroller ATMEGA 8
2. Bagaimana memperantarakan ic ADXL202 sebagai sensor yang yang memiliki keluaran berupa PWM (pulse width modulation).
3. Bagaimana memproses pengiriman dari ic ADXL202 ke mikrokontroller
4. Bagaimana merancang program mikrokontroler agar pembacaan ic ADXL202 dapat diolah datanya kedalam mikrokontroller.
5. Bagaimana membuat instalasi hardware dari ic ADXL202 ke mikrokontroller ATMEGA 8
6. Bagaimana merancang rangkaian push button yang digunakan sebagai masukan.

1.4 Batasan Masalah

Mengingat begitu luasnya bahan kajian mengenai sistem yang terdapat pedometer digital tersebut , maka pokok bahasan dibatasi sebagai berikut :

1. Tidak membahas pedometer digital dalam betuk lain
2. Sensor yang digunakan adalah ic ADXL202 dimana ic tersebut sensitif terhadap pergerakan suatu benda.
3. Karena menggunakan sensor yang sensitive terhadap pergerakan vertical tubuh, tidak memperhitungkan panjang kaki pengguna nya.

4. Tidak membahas system kerja ic ADXl202 sehingga dapat mengukur sudut kemiringan
5. Tidak membahas masalah catu daya.

1.5 Metodologi Penulisan

Metodologi yang dipakai dalam pembuatan skripsi ini adalah:

1. Studi Literatur

   Dengan mencari referensi-referensi yang berhubungan dengan perencanaan dan pembuatan alat yang akan dibuat.

2. *Field Research*

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

3. *Design* dan Pembuatan Alat

   Yaitu meliputi pembuatan PCB, perakitan komponen serta penyolderan dan pembuatan perangkat lunak.

4. Pengujian Alat

   Dengan melakukan pengujian untuk setiap blok rangkaian dan kerja seluruh sistem pada alat tersebut.

1.6 Sistematika Pembahasan

Penulisan skripsi ini terbagi menjadi lima bab dengan sistematika sebagai berikut:

BAB I : PENDAHULUAN

Membahas tentang latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi dan sistematika penulisan pada penulisan skripsi ini.

BAB II : LANDASAN TEORI

Berisikan tentang penjelasan dan teori-teori yang berhubungan denagn komponen-komponen yang digunakan dalan perancangan alat.

BAB III : PERENCANAAN DAN PEMBUATAN ALAT

Membahas tentang perancangan alat yang terdiri dari perancangan perangkat keras dan perancangan perangkat lunak.

BAB IV : PENGUJIAN ALAT

Membahas tentang pengujian peralatan secara keseluruhan dan analisa hasil pengujian.

BAB V : PENUTUP

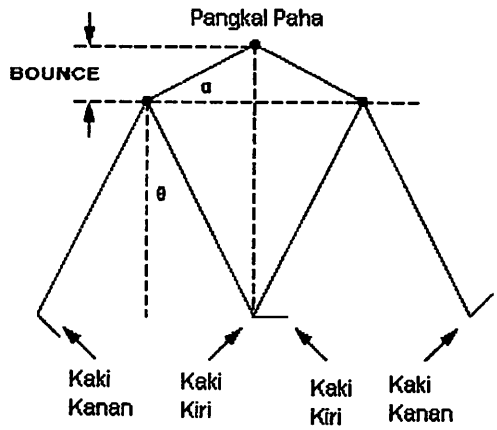
Berisikan kesimpulan yang didapat selama perancangan dan pembuatan alat serta saran-saran.

BAB II

DASAR TEORI

2.1 Umum

Pedometer adalah alat penghitung langkah dan jarak seseorang dimana alat tersebut diletakkan di pangkal paha,sistemnya ketika seseorang berjalan maka tubuh akan mengalami gerakan vertikal dalam setiap langkahnya hal inilah dideteksi oleh ic ADXL202 sebagai langkah.



Gambar 2.1. Gerakan langkah kaki

Dari gambar diatas bounce atau lambungan terjadi ketika kaki terangkat dari tanah. Langkah seseorang memiliki karakter bounce atau pergerakan vertikal yang berbeda tinggi rendahnya seseorang akan mempengaruhi besar kecilnya pegerakan vertikal atau bounce. Berdasarkan gambar diatas kita akan dapatkan cara untuk mencari langkah yaitu dengan cara mengamati nilai percepatan maksimal dan minimal dari sensor ini, sedangkan untuk menghitung

jarak terlebih dahulu kita harus mengetahui panjang setiap langkah, penulis menggunakan rumus yang telah di uji coba oleh analog devices yaitu

$$PL = \sqrt[4]{A_{max} - A_{min}} \times k \text{ [5]}$$

$$\text{Jarak} = PL \times n \text{ [5]}$$

Dimana :

PL = panjang langkah

A_{max} = Percepatan maksimal

A_{min} = Percepatan minimal

n = banyak langkah

k = konstanta (unit konversi dalam meter yang memiliki nilai 0,5)

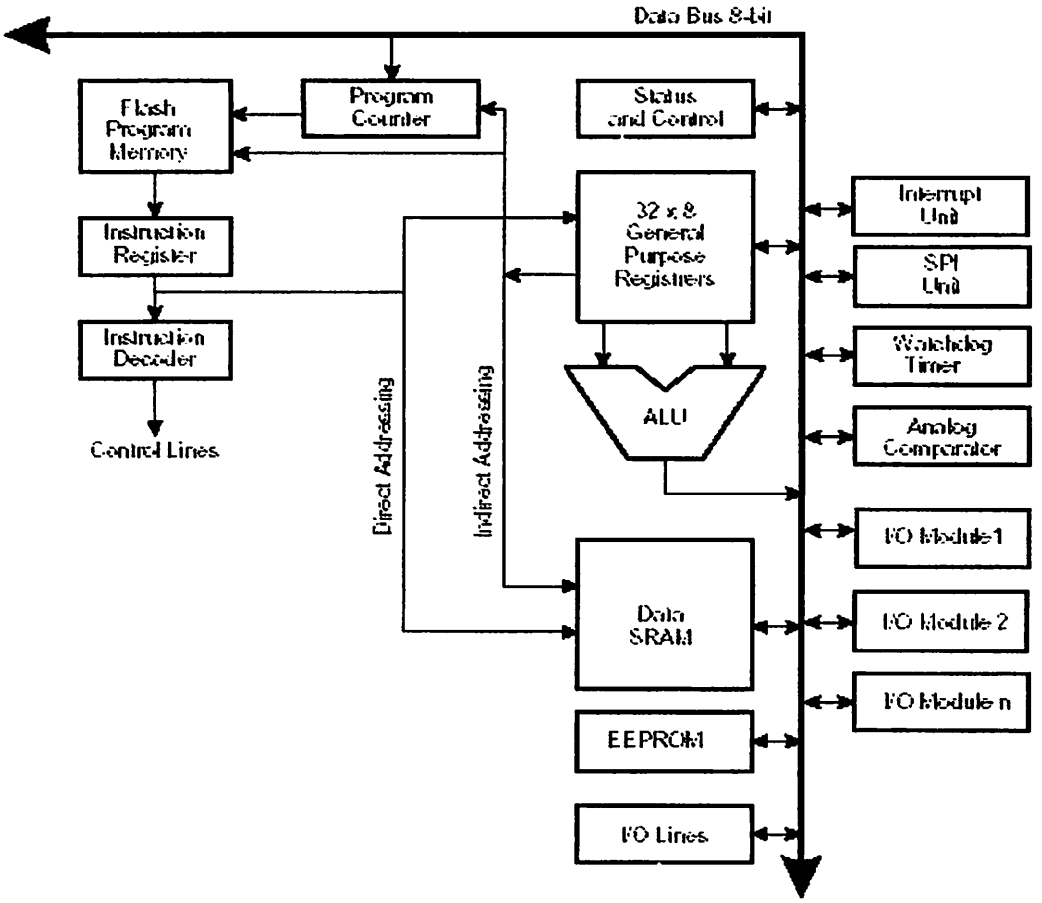
2.2 Mikrokontroller AVR ATMega8

2.2.1. Arsitektur

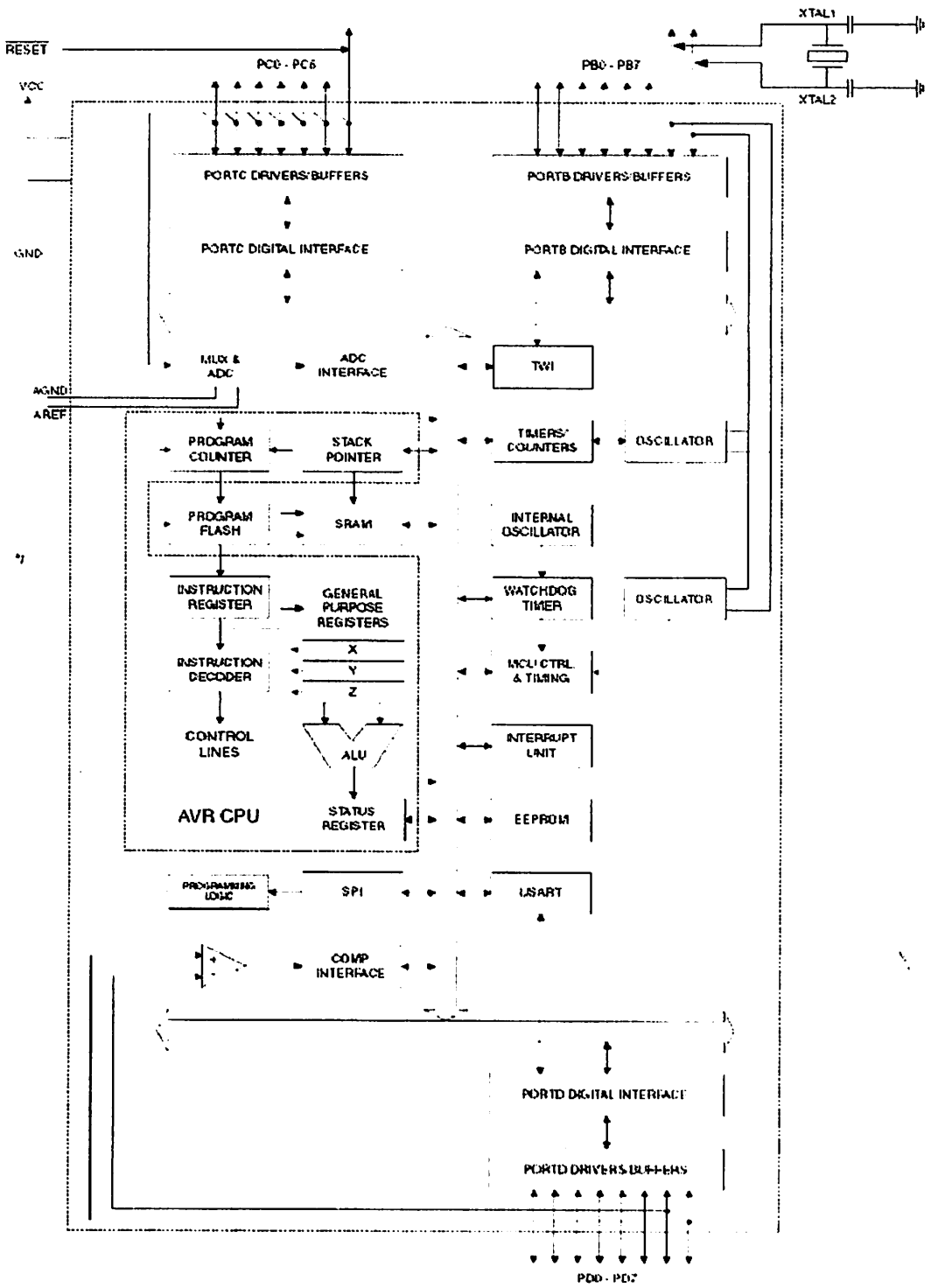
AVR Atmega8 adalah mikrokontroler 8-bit CMOS, *low-power* yang berdasarkan pada bentuk arsitektur AVR RISC (*Reduced Instruction Set Komputer*), yang hampir semua instruksinya selesai dikerjakan dalam satu siklus *clock*. AVR ATmega8 menggunakan instruksi tunggal (*Single Clock Cycle*), yaitu sistem mikrokontroler yang frekuensi kerja dalam chip sama dengan frekuensi kristal untuk osilator tanpa memerlukan rangkaian pembagi frekuensi setelah osilator yang diperlukan untuk memperoleh perbedaan fase dari *clock*, sehingga AVR 12 kali lebih cepat dibanding MCS51.

Berbagai karakteristik yang tersedia dalam IC ATmega8 adalah sebagai berikut:

- 8K *bytes In-Sistem Programable Flash*
- 512 *bytes* EEPROM (*Electrical Erasable Programable Read Only Memory*)
- 512 *bytes* SRAM (*Static Random Access Memory*)
- 23 jalur I/O *general-purpose*
- 32 x 8 *general-purpose working register*
- *Timer/ Counter* yang fleksibel dengan *mode* pembanding
- Interupsi internal dan eksternal
- Pemrograman serial UART (*Universal Asynchronous Receiver and Transmitter)*
- *Serial Port* SPI (*Serial Peripheral Interface*)



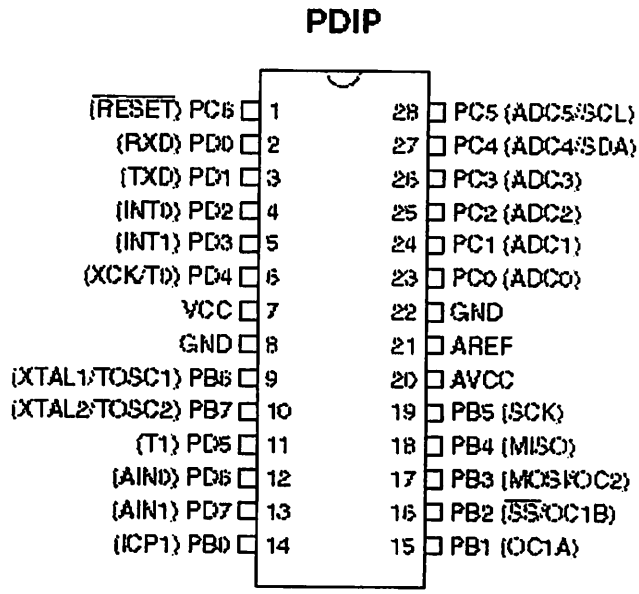
Gambar 2.2. Arsitektur AVR ATmega8[1]



Gambar 2.3. Blok Diagram AVR ATmega8[11]

2.2.2. Konfigurasi Pin-Pin Mikrokontroller ATmega8

Mikrokontroller ATmega8 mempunyai 28 pin seperti pada gambar di bawah ini:



Gambar 2.4. Konfigurasi Pin ATmega8[1]

Fungsi tiap pin-nya adalah sebagai berikut:

a. Vcc: Tegangan *Supply*

b. Gnd: *Ground*

c. *Port* B (PB0-PB7: *Port* dua arah I/O 8-bit dengan *resistor pull-up internal*, digunakan pada fungsi-fungsi khusus dari karakteristik ATmega8.

d. *Port* C (PC0-PC5): *Port* dua arah I/O 6-bit dengan *resistor pull-up internal*.

e. PC6/$\overline{\text{RESET}}$: Jika fuse RSTDISBL sudah diprogram, PC6 digunakan sebagai suatu pin I/O. Jika fuse RSTDISBL belum diprogram, PC6

digunakan sebagai inputan *reset* dimana *level low* dari pin ini lebih panjang dari pulsa minimum yang dihasilkan *reset*.

f. *Port* D (PD0-PD7): *Port* dua arah /O 7-bit dengan *resistor pull-up internal*. Sebagai input, *port* D menggunakan eksternal *pull low* dengan sumber arus jika *pull up resistor* diaktifkan.

g. $\overline{\text{RESET}}$: Input *reset*. *Level low*-nya untuk lebih panjang dari pulsa minimum yang dihasilkan *reset*, meskipun *clock* tidak bekerja.

h. AV_{CC}: sebagai *supplay* tegangan untuk A/D *konverter port* C (3..0), dan ADC (7..6). Pin ini harus dihubungkan dengan V_{CC} melalui *low-pass filter*.

i. AREF: Pin analog referensi untuk A/D *konverter*.

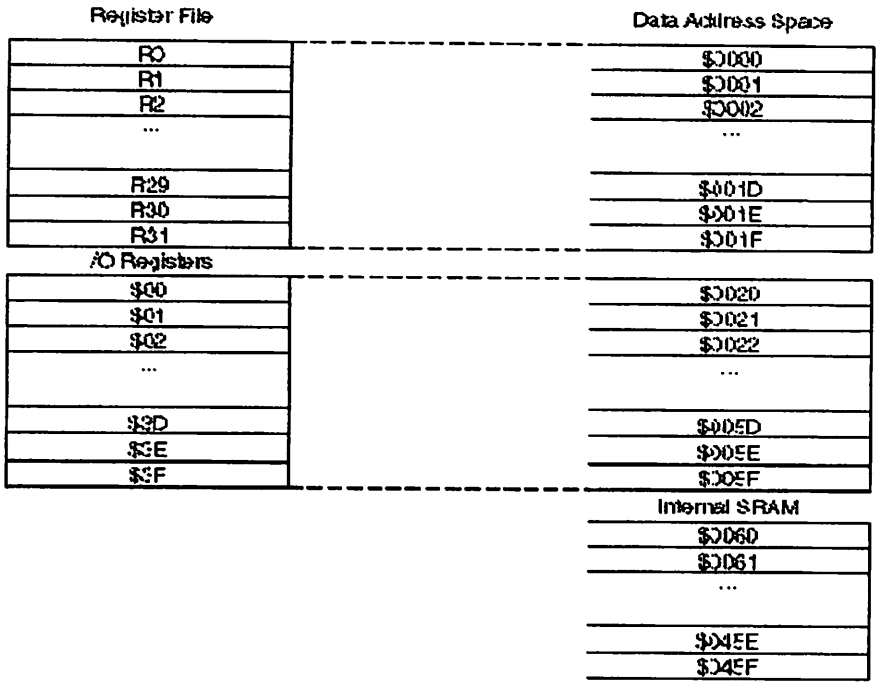
j. ADC 7..6 (TQFT and MLF): Pada TQFP dan MLP, ADC 7..6 bekerja sebagai input *analog* untuk A/D *konverter*. Pin-pinnya mendapat daya dari power *supplay analog* dan dapat melayani 10 bit saluran ADC.

2.2.3 Peta Memori

AVR ATmega8 memiliki ruang pengalamatan memori data dan memori program yang terpisah. Memori data terbagi menjadi 3 bagian, yaitu 32 register umum, 64 buah register I/O, dan 1024 *byte* SRAM *Internal*.

Register keprluan umum menempati *space* data pada alamat terbawah, yaitu $00 sampai$1F. Sementara itu, register khusus unutk menangani I/O dan kontrol terhadap mikrokontroller menempati 64 alamat berikutnya, yaitu mulai dari $20 hingga $5F. register tersebut merupakan register yang khusus digunakan untuk mengatur fungsi terhadap peripheral mikrokontroller, seperti kontrol register, *timer/ counter*, fungsi-fungsi I/O, dan sebagainya. Alamat memori

berikutnya digunakan untuk SRAM 1024 *byte*, yaitu pada lokasi $60 sampai dengan $45F. Konfigurasi memori data ditunjukkan pada gambar dibawah ini.

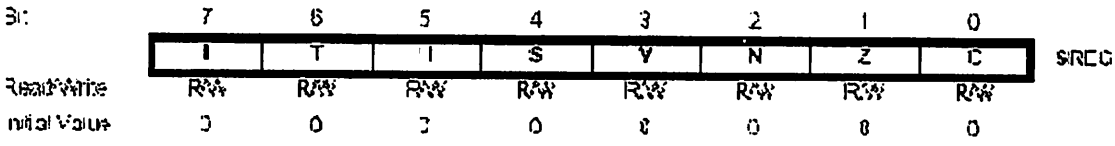


Gambar 2.5. Konfigurasi Memori Data AVR ATmega8[11]

Memori program yang terletak dalam *Flash Sistem Reprogrammable Flash* mempunyai 8K *byte* karena setiep instruksi memiliki lebar 16-bit atau 32-bit. AVR ATmega memiliki 4K *byte* x 16-bit *Flash* dengan alamat mulai dari $00 sampai $FFF. AVR tersebut memiliki 12-bit *Program Counter* sehingga mampu mengalamati isi *Flash*. Selain itu,AVR ATmega8 juga memiliki memori data berupa EEPROM 8-bit sebanyak 512 *byte*. Alamt EEPROM dimulai dari $000 sampai $1FF.

2.2.4 Status Register (SREG)

Status register adalah register berisis status yang dihasilkan pada setiap operasi yang dilakukan ketika suatu instruksi dieksekusi. SREG merupakan bagian dari inti CPU mikrokontroller.



Gambar 2.6. Status Register Atmega8[11]

Keteranga dari bit SREG adalah:

a. Bit7 - I: *Global Interrupt Enable*

Bit harus diset untuk meng-*enable* interupsi. Setelah itu, baru dapat mengaktifkan interupsi mana yang akan digunakan dengan cara meng-*enable* bit kontrol register yang bersangkutan secara individu. Bit akan di-*clear* apabila terjadi suatu interupsi, serta akan diset kembali oleh intruksi RETI.

b. Bit 6 – T: *Bit Copy Storage*

Intruksi BLD dan BST menggunakan bit-T sebagai sumber atau tujuan dalam operasi bit. Suatu bit dalam suatu register GPR dapat disalin ke bit T menggunakan instrusi BST, dan sebaliknya bit T dapat disalin kembali ke suatu bit dengan register GPR menggunakan instruksi BLD.

c. Bit 5 – H: *Half Carry Flag*

d. Bit 4 – S: *Sign* Bit, $S = N \oplus V$

Bit-S merupakan hasil operasi EOR antara *flag*-N (*negative*) dan *flag* V (komplemen dua *overflow*).

e. Bit 3 – V:*Two's Complement Overflow Flag*

Bit berguna untuk mendukung operasi aritmatika.

f. Bit 2 – N: Ne*gative Flag*

Apabila suatu operasi menghasilkan bilangan negative, maka *flag*-N akan diset.

g. Bit 1 – Z: *Zero Flag*

Bit akan diset bila hasil operasi yang diperoleh adalah nol.

h. Bit 0 – C: *Carry Flag*

Apabila suatu operasi menghasilkan *carry*, maka bit akan diset.

2.2.5 Register I/O

Semua *port* pada AVR memiliki kebenaran fungsional *read-modify-write* ketika digunakan sebagai *port* I/O umum. Ini berarti bahwa arah dari satu pin *port* dapat diubah tanpa bermaksud mengubah arah dari pin yang lain. Logika *port* I/O dapat diubah-ubah dalam program secara *byte* atau hanya bit tertentu. Mengubah sebuah keluaran bit I/O dapat dilakukan menggunakan perintah cbi (clear bit I/O) untuk menhasilkan output *low* dan perintah sbi (set bit I/O) untuk menghasilkan output *high*. Pengubahan secara *byte* dilakukan dengan perintah *in* atau *out* yang menggunakan register bantu.

a. *Port* B

Tiga lokasi alamat memori I/O yang dilokasikan pada *port* D, masing-masing adalah register data-*port* B, $18 ($38), register pengarah data-DDRB, $17 ($37), dan pin input *port* B-PINB, $16 ($36). (Blok diagram *port* B dan fungsi *Timer/ Counter 2* pinnya dapat dilihat pada lampiran).

b. *Port* C

Tiga lokasi alamat memori I/O yang dilokasikan pada *port* C, masing-masing adalah register data-*port* C, $15 ($35), register pengarah data-DDRC, $14 ($34), dan pin input *port* C-PINC, $13 ($33). (Blok diagram skematik dapat dilihat pada lampiran).

c. *Port* D

Tiga lokasi alamat memori I/O yang dilokasikan pada *port* D, masing-masing adalah register data-*port* D, $12 ($32), register pengarah data-DDRD, $11 ($31), dan pin input *port* D-PIND, $10 ($30). (Blok diagram skematik *port* D dan fungsi alternatif pinnya dapat dilihat pada lampiran).

2.2.6 Timer/ Counter 2

Timer/ Counter 2 adalah 8 bit *Timer/ Counter* yang multifungsi. Deskripsi *Timer/ counter* 2 pada ATmega8 adalah sebagai berikut:

a. Sebagai *Counter* 1 kanal.

b. *Timer* dinolkan saat *match compare* (*auto reload*).

c. Dapat menghasilakan gelombang PWM dengan *glitch-free*.

d. *Frekuensi generator*.

e. *Prescaler* 10 bit untuk timer.

f. Interupsi *timer* yang disebabkan *timer overflow* dan *match compare* (TOV2 dan OCF2).

g. Dapat menggunakan *clock* dari kristal *independent* luar sebesar 32 kHz pada I/O *clock.*

Pengaturan Timer/*Counter*2 diatur oleh TCCR2 (*Timer/ Counter* Kontrol Register 0) yang dapat dilihat pada gambar berikut:

Bit	7	6	5	4	3	2	1	0	
	FOC2	WGM20	COM21	COM20	WGM21	CS22	CS21	CS20	TCCR2
Read/Write	W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Gambar 2.7. Register TCCR2[1]

Penjelasan untuk setiap bit adalah:

a. bit 7 – FOC2: *Force Output Compare*

b. bit 6,3 – WGM21: WGM21:WGM20: *Waveform Generator Unitl.*

Bit tersebut mengontrol kenaikan *counter*, sumber dari nilai maksimum *counter*, dan tipe dari jenis *Timer/ Counter* yang dihasilkan, yaitu *mode normal, clear timer, mode compare match*, dan dua tipe dari PWM (*Pulse Width Modulation*). Berikut tabel *setting* pada bit untuk menghasilkan *mode* tertentu:

Tabel 2.1. Konfigurasi Bit WGM21 dan WGM20[1]

Mode	WGM21 (CTC2)	WGM20 (PWM2)	Timer/Counter Mode of Operation[1]	TOP	Update of OCR2	TOV2 Flag Set
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR2	Immediate	MAX
3	1	1	Fast PWM	0xFF	TOP	MAX

c. Bit 5,4 – COM01:COM00: *Compare Match Output Mode*

Bit mengontrol pin OC0 (*Output Compare Pin*). Apabila kedua bit tersebut nol atau *clear*, maka pin CO0 berfungsi sebagai pin biasa. Namun, jika salah satu bit *set*, maka fungsi bit tergantung pada *setting* bit pada WGM00 dan WGM01. Berikut daftar tabel *setting* bit pada WGM00 dan WGM01:

Tabel 2.2. Konfigurasi Bit COM21 dan COM20 *Compare Output Mode* non-PWM[1]

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Toggle OC2 on Compare Match
1	0	Clear OC2 on Compare Match
1	1	Set OC2 on Compare Match

Tabel 2.3. Konfigurasi Bit COM21 dan COM20 Compare Output *Mode* Fast PWM[1]

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Reserved
1	0	Clear OC2 on Compare Match, set OC2 at TOP
1	1	Set OC2 on Compare Match, clear OC2 at TOP

Tabel 2.4. Konfigurasi Bit COM21 dan COM20 *Compare Output Phase Correct* PWM[1]

COM21	COM20	Description
0	0	Normal port operation, OC2 disconnected.
0	1	Reserved
1	0	Clear OC2 on Compare Match when up-counting. Set OC2 on Compare Match when downcounting.
1	1	Set OC2 on Compare Match when up-counting. Clear OC2 on Compare Match when downcounting.

d. Bit 2, 1, 0 – CS22, CS21, CS20

Ketiga bit tersebut memilih sumber *clock* yang akan digunakan oleh Timer/*Counter*. Berikut *list* tabelnya:

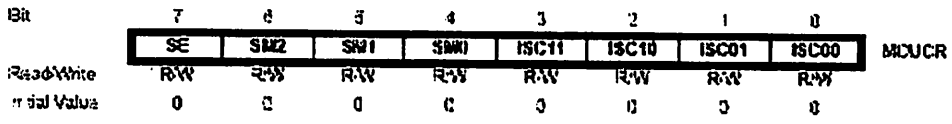
Tabel 2.5. Konfigurasi Bit *Clock Select* Untuk Memilih Sumber *Clock*[1]

CS22	CS21	CS20	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	clk_{T2S}/(No prescaling)
0	1	0	clk_{T2S}/8 (From prescaler)
0	1	1	clk_{T2S}/32 (From prescaler)
1	0	0	clk_{T2S}/64 (From prescaler)
1	0	1	clk_{T2S}/128 (From prescaler)
1	1	0	clk_{T2S}/256 (From prescaler)
1	1	1	clk_{T2S}/1024 (From prescaler)

2.2.7 Interupsi

Interupsi adalah kondisi yang membuat CPU berhenti dari rutinitas yang sedang dikerjakan (rutin utama) untuk mengerjakan rutin lain (rutin intrupsi). AVR ATMega8 memiliki 19 sumber interupsi.

a. Pada AVR terdapat 3 pin untuk interupsi eksternal, yaitu INT0, INT1, INT2. Interupsi eksternal dapat dibangkitkan apabila terdapat perubahan logika atau logika 0 pada pin interupsi. pengaturan kondisi keadaan yang menyebabkan terjadinya interupsi eksternal diatur oleh register MCUCR (MCU Kontrol Register). Yang terlihat pada gambar dibawah ini:



Gambar 2.8. Register MCUCR[1]

b. Bit ISC11 dan ISC10 bersama-sama menentukan kondisi yang dapat menyebabkan interupsi eksternal pada pin INT1. Keadaan selengkapnya dapat dilihat pada tabel dibawah ini:

Tabel 2.6. Beberapa *Setting* Kondisi Yang Menyebabkan Interupsi Eksternal I[1]

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

c. Bit ISC01 dan ISC00 bersama-sama menentukan kondisi yang dapat menyebabkan interupsi eksternal pada pin INT0. keadaan selengkapnya dapat dilihat pada tabel dibawah ini:

Tabel 2.7 Beberapa *Setting* Kondisi Yang Menyebabkan Interupsi Eksternal 0[1]

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

Pemilihan pengaktifan interupsi eksternal diatur oleh register GICR (General Interupsi Kontrol Register) yang terlihat seperti gambar berikut:

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	–	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Gambar 2.9. General Interrupt Kontrol Register[1]

Bit penyusun dapat dijelaskan sebagai berikut:

a. Bit INT1 adalah bit untuk mengaktifkan intrupsi ekternal 1. apabila bit tersebut diberi logika 1 dan bit-I pada SREG (status register) juga satu, maka interupsi eksternal 1 akan aktif.
b. Bit INT0 adalah bit untuk mengaktifkan intrupsi eksternal 0. apabila bit tersebut diberi logika 1 dan bit-I pada SREG (status register) juga satu, maka interupsi eksternal 0 akan aktif.
c. Bit INT2 adalah bit untuk mengaktifkan interupsi ekternal 2 apabila bit tersebut diberi logika 1 dan bit-I pada SREG (status register) juga satu, maka ekternal 2 akan aktif.

Program interupsi dari masing-masing jenis interupsi eksternal akan dimulai dari vektor interupsi pada masing-masing jenis. Alamatnya dapat dilihat pada tabel:

Tabel 2.8. Macam Sumber Interupsi pada AVR Atmega8[11]

Vector No.	Program Address[2]	Source	Interrupt Definition
1	0x000[1]	RESET	External Pin, Power-on Reset, Brown-out Reset, and Watchdog Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	TIMER2 COMP	Timer/Counter2 Compare Match
5	0x004	TIMER2 OVF	Timer/Counter2 Overflow
6	0x005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	0x006	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	0x007	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	0x008	TIMER1 OVF	Timer/Counter1 Overflow
10	0x009	TIMER0 OVF	Timer/Counter0 Overflow
11	0x00A	SPI, STC	Serial Transfer Complete
12	0x00B	USART, RXC	USART, Rx Complete
13	0x00C	USART, UDRE	USART Data Register Empty
14	0x00D	USART, TXC	USART, Tx Complete
15	0x00E	ADC	ADC Conversion Complete
16	0x00F	EE_RDY	EEPROM Ready
17	0x010	ANA_COMP	Analog Comparator
18	0x011	TWI	Two-wire Serial Interface
19	0x012	SPM_RDY	Store Program Memory Ready

Untuk inisialiasasi awal interupsi, perlu dituliskan terlebih dahulu vektor interupsi dari interupsi yang terdapat pada sistem. Vektor interupsi adalah nilai yang disimpan ke *program counter* pada saat terjadi interupsi sehingga program akan menuju ke alamat yang ditunjukkan oleh *program counter*. alamat interupsi eksternal 0 pada alamat 001H dan interupsi terima serial pada alamat 00B masing-masing alamat vektor memililki jarak yang berdekatan sehingga akan timbul masalah jika diperlukan rutin layanan interupsi yang panjang oleh sebab itu

layanan interupsi ekternal 0 akan melompat ke alamat *ext_int0* dan inetrupsi terima serial pada alamat USART_RXC.

Pengaktifan interupsi eksternal dilakukan dengan memberikan logika satu pada register GICR. Dengan demikian, pada pengaktifan interupsi ekternal 0, akan diberikan logika satu pada bit ke 6 register GICR. pengaktifan interupsi terima serial dilakukan dengan memberikan logika 1 pada bit ke 7 register UCSRA. Terakhir, berikan perintah sei untuk menagaktifkan *global interupt*.

Interupsi dapat muncul kapan pun (kecuali jika bit *enable interupsi* dalam SREG *clear*) dengan demikian, interupsi juga dapat mencul ketika program sedang melakukan kalkulasi. Kalkulasi tersebut merubah *flags* dalam status register yang digunakan untuk *next step* dari kalkulasi atau untuk beberapa percabangan program. Jika ISR mengubah *flags* dalam SREG, maka kalkulasi yang sedang ditempatkan dalam program yang berjalan normal dapat di-*corupt*. Oleh sebab itu, perlu pengamanan SREG pada setiap subrutin interupsi.

2.2.8 Register TIMSK

Selain register di atas terdapat pula register TIMSK (*Timer/ Counter* Interrupt Mask Register).

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	-	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Gambar 2.10. Register TIMSK[11]

Penjelasan Untuk setiap bit adalah:

a. Bit 0 – TOIE0: *Timer/ Counter* 0 *Overflow Interrup Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka dilakukan *enable* interupsi Overflow *Timer/ Counter* 0.

b. Bit 1 – OCIE0: *Timer/ Counter* 0, *Output Compare Match Interrupt Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka bias dilakukan *enable* interupsi *Output Compare Match Timer/ Counter* 0.

c. Bit 2 – TOIE1: *Timer/ Counter* 1 *Overflow Interrup Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka dilakukan *enable* interupsi Overflow *Timer/ Counter* 1.

d. Bit 3 – OCIE1B: *Timer/ Counter* 1, *Output Compare* B *Match Interrupt Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka dilakukan *enable* interupsi Overflow Compare Match B *Timer/ Counter* 1.

e. Bit 4 – OCIE1A: *Timer/ Counter* 1, *Output Compare* A *Match Interrupt Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka dilakukan *enable* interupsi *Overflow Compare Match* A *Timer/ Counter* 1.

f. Bit 5 – TICIE1: *Timer/ Counter* 1 *Input Capture Interrupt Enable*

g. Bit 6 – TOIE2: *Timer/ Counter* 2, *Overflow Interrupt Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka dilakukan *enable* interupsi *Overflow Timer/ Counter* 2.

h. Bit 7 – OCIE2: *Timer/ Counter* 2, *Output Compare Match Interurpt Enable*

Jika bit tersebut diberi logika satu dan bit i SREG juga set, maka bias dilakukan *enable* interupsi *Output Compare Match Timer/ Counter* 2.

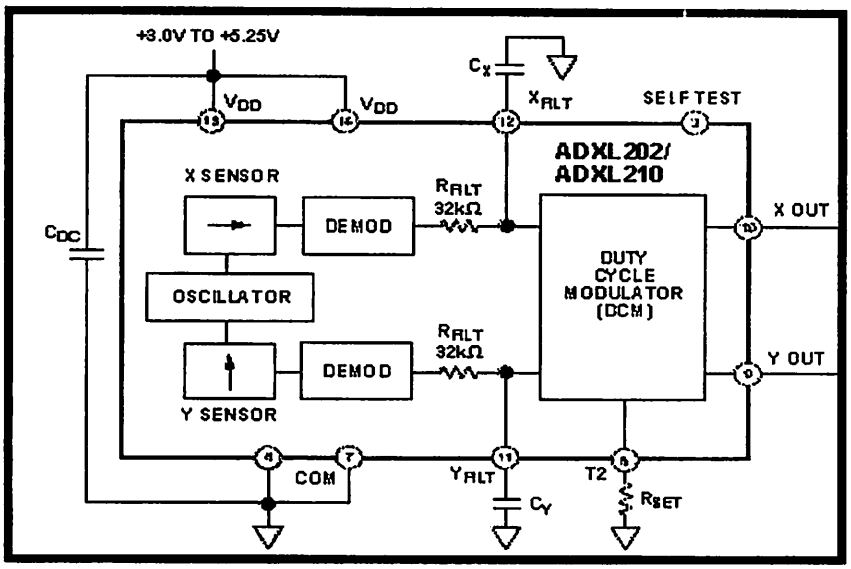
2.3 IC ADXL202 sebagai sensor

2.3.1. Umum

ADXL202 merupakan modul sensor yang berbentuk ic dimana ic tersebut dapat menghitung 2 sudut percepatan suatu benda bergerak atau 2-axis accelerometer yaitu dinamis akselarasi seperti getaran dan statis akselerasi seperti gravitasi. Sinyal keluaran dari ic ini berupa lebar pulsa atau sinyal PWM dan sinyal analog.

2.3.2. Arsitektur

ADXL202 adalah termasuk single monolithic intergrated circuit dimana keluaran sinyal analog dari x dan y sensor diubah menjadi sinyal digital berupa lebar pulsa dengan menggunakan low pass filter yang sudah terintegrasi didalamnya berikut diagram blok ic ini.



Gambar 2.11. Diagram blok IC ADXL202[2]

Sinyal analog dari demodulator adalah low pass filter lalu di ubah menjadi lebar pulsa dengan menggunakan DCM (duty cycle modulator) sehingga menghasilkan sinyal PWM.

konfigirasi pin-pin yang digunakan yaitu

a. Vdd : tegangan supplay sebesar +3,0 volt sampai +5,25 volt
b. X filt : sinyal keluaran analog dari sensor x
c. Y filt : sinyal keluaran analog dari sensor y
d. C_{dc} : decoupling kapasitor yang nilainya 0,1µF
e. Rset : pin ini untuk menentukan lebar sinyal T2
f. X out : sinyal keluaran berupa PWM untuk sensor X
g. Y out : sinyal keluaran berupa PWM untuk sensor Y

Jika ingin menggunkan sinyal analog maka kita harus menentukan bandwidth dari X filt dan Y filt untuk itu kita harus mengganti nilai capacitor berikut tabelnya

Table 2.9. Pengaturan bandwidth[2]

T2	R_{SET}
1 ms	125 kΩ
2 ms	250 kΩ
5 ms	625 kΩ
10 ms	1.25 MΩ

Untuk mengatur niilai dari T2 maka kita harus mengubah nilai resistansi dari pin Rset berikut tabelnya

Tabel 2.10. Pengaturan periode (T2) [2]

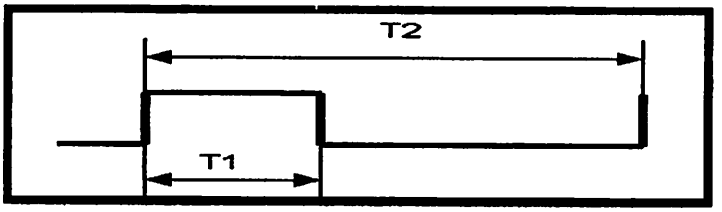
Bandwidth	Capacitor Value
10 Hz	0.47 µF
50 Hz	0.10 µF
100 Hz	0.05 µF
200 Hz	0.027 µF
500 Hz	0.01 µF
5 kHz	0.001 µF

atau dapat juga menggunakan rumus

$$T2 = \frac{R_{SET}\,(\Omega)}{125\,M\Omega}\,[2]$$

2.3.2. Prinsip kerja

Percepatan dapat ditentukan dengan mengukur panjang dari T1 dan T2. Perbandingan dari T1 dan T2 proporsional terhadap percepatan



Gambar 2.12. Sinyal PWM[2]

Untuk menghitung akselerasi atau percepatan dapat digunakan rumus sebagai berikut :

$$a_g = \frac{duty\ cycle - duty\ cycle\ saat\ 0\ g}{duty\ cycle\ per\ g}$$

Dutycycle pada saat 0 gravitasi adalah 50 % sedangkan perubahan dutycycle per gravitasi adalah 12,5% sehingga rumus diatas menjadi :

$$a = \frac{\left(\frac{T_1}{T_2}\right) - 0{,}5}{0{,}125} \times g$$

Dimana g adalah gravitasi yang memiliki konstanta 9,8 m/s^2

2.4 LCD (*Liquid Crystal Display*)

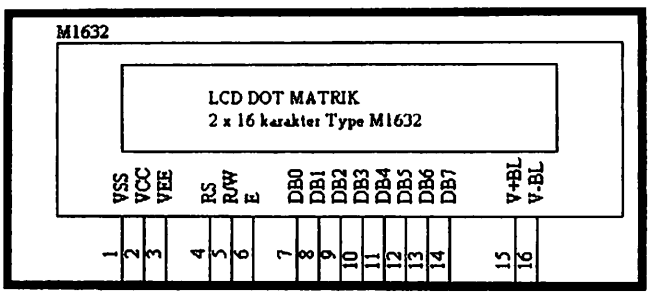
Liquid Crystal Display atau LCD merupakan komponen optoelektronik yaitu komponen yang bekerja atau dipengaruhi oleh sinar (optolistrik), komponen pembangkit cahaya (light emiting) dan komponen-komponen yang akan mengubah sinar. LCD terbuat dari bahan kristal cair yang merupakan suatu komponen organik dan mempunyai sifat optik seperti benda padat meskipun bahan tetap cair.

Sel kristal cair terdiri dari selapis bahan kristal cair yang diapit antara dua kaca tipis yang transparan. Antara dua lembar kaca tersebut diberi bahan kristal cair (*liquid crystal)* yang tembus cahaya. Permukaan luar dari masing-masing keping kaca mempunyai lapisan penghantar tembus cahaya seperti oxida timah

(tin oxide) atau oxida indium *(indium oxide)*. Sel mempunyai ketebalan sekitar 1 x 10^{-5} meter dan diisi dengan kristal cair.

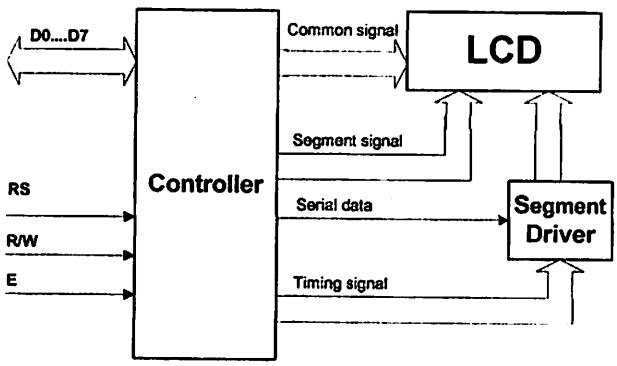
Karena sel-sel kristal cair merefleksikan cahaya dan bukan membangkitkan cahaya maka konsumsi daya yang dibutuhkan relatif rendah. Energi yang dipergunakan hanya untuk mengaktifkan kristal cair. Pada dasarnya LCD bekerja pada tegangan rendah (3 – 15 Vrms), frekuensi rendah (25 – 60 Hz) sinyal AC dan memakai arus listrik yang sangat kecil (25 - 300A). LCD seringkali ditata sebagai tampilan *seven segment* untuk menampilkan angka tetapi juga memiliki keistimewaan lain, yaitu kemampuan untuk menampilkan karakter dan berbagai macam simbol.

Salah satu jenis LCD diantaranya adalah LCD M1632, suatu jenis piranti dengan konsumsi daya yang rendah, disusun dari dot matrik dan dikontrol oleh ROM atau RAM generator karakter dan RAM data display. Pengontrolan utamanya adalah pada ROM generator dan display data RAM yang menghasilkan kode ASCII jika padanya diberikan input ASCII. Untuk dapat difungsikan dengan baik maka perlu diperhatikan proses analisasi yang telah ditentukan oleh pabrik pembuatnya. Timing penganalisasian sangat dipertimbangkan, karena jika meleset sampai ordo *milisecon* maka dapat dipastikan LCD tidak dapat berfungsi.



Gambar 2.13. Konfigurasi Kaki LCD[13]

Masukan yang diperlukan untuk mengendalikan modul berupa bus data yang masih termutiplek dengan bus alamat serta 3 bit sinyal kontrol. Sementara pengendalian LCD dilakukan secara internal oleh kontroler yang sudah terpasang dalam modul LCD. Diagram blok untuk LCD dapat dilihat dalam Gambar 2.8



Gambar 2.14. Blok Diagram LCD[13]

Adapun karakteristik dari LCD M1632 antara lain :

- Dengan 16 karakter – 2 baris dalam bentuk dotmatrik 5x7 dan cursor
- *Duty ratio* 1/16
- Memiliki ROM pembangkitan karakter untuk 192 jenis karakter
- RAM untuk data display sebanyak 80x8 bit
- Dapat dirangkai dengan MPU 8 bit/4 bit
- RAM data display dan RAM pembangkit karakter dapat dibaca oleh MPU

- Memiliki fungsi instruksi antara lain *display on/off, Cursor on/off, display* karakter *blink, cursor shift* dan *display shift*
- Memiliki rangkaian osilator sendiri
- Catu tegangan tunggal yaitu ± 5 V
- Memiliki rangkaian reset otomatis pada catu daya yang dihidupkan
- Temperatur operasi 0^0 - 50^0

LCD memiliki 16 pin yang masing-masing mempunyai fungsi sebagai berikut :

Tabel 2.11. Fungsi Tiap Pin LCD[3]

No. Pin	Simbol	Level	Fungsi	
1	V_{SS}	-	Power Supply	0 V (GND)
2	V_{CC}	-		5 V ± 10%
3	V_{DD}	-		For LCD Drive
4	RS	H/L	Sinyal seleksi register H ; Data Input [register data (write/read)] L ; Instruction Input [register instruksi (write), busy flag dan address counter (read)]	
5	R/W	H/L	H ; Read L ; Write	
6	E	H	Enable Signal [sinyal penanda mulai operasi, aktif saat operasi write atau read]	
7	DB0	H/L	4 bit bus data lower 2 arah, dapat dibaca atau ditulis terhadap mikrokontroler	
8	DB1	H/L		
9	DB2	H/L		
10	DB3	H/L		
11	DB4	H/L	4 bit bus data upper 2 arah, dapat dibaca atau ditulis terhadap mikrokontroler, DB7 juga sebagai busy flag	
12	DB5	H/L		
13	DB6	H/L		
14	DB7	H/L		
15	V+BL	-	Back Light Supply	4 - 4,2 V 50 – 200 mA
16	V-BL	-		0 V (GND)

2.4.1 Instruksi Operasi

Tabel 2.12. Instruksi Pada LCD[11]

Instruksi	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0
Display Clear	0	0	0	0	0	0	0	0	0	1
Cursor Home	0	0	0	0	0	0	0	0	1	*
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
Display On/Off	0	0	0	0	0	0	1	D	C	B
Cursor Display Shift	0	0	0	0	0	1	S/C	R/L	*	*
Function Set	0	0	0	0	1	DL	1	*	*	*
CG RAM Address Set	0	0	0	1	A_{CG}					
DD RAM Address Set	0	0	1	A_{DD}						
BF/Address Read	0	1	BF	AC						
Data Write to CG RAM	1	0	Write Data							
Data Read from CG RAM	1	1	Read Data							

*Invalid Bit

A_{CG} ; CG RAM Address

A_{DD} ; DD RAM Address

2.4.2. Operasi Dasar

- Register

  Kontrol dari LCD memiliki 2 buah register 8 bit yaitu register instruksi (IR) dan register data (DR). IR memiliki instruksi seperti display, clear, cursor shift dan display data (DD RAM) serta karakter (CG RAM). DR menyimpan data untuk ditulis ke DD RAM ataupun membaca data dari DD RAM dan CG RAM. Ketika data ditulis ke DD RAM atau CG RAM maka DR secara otomatis menulis data ke DD RAM atau CG RAM. Ketika data pada CG RAM atau DD RAM akan dibaca maka alamat data ditulis pada IR. Sedangkan data akan dimasukkan melalui DR sehingga dapat dibaca oleh mikrokontroler.

Tabel 2.13. Pemilihan Register Pada LCD[3]

RS	RW	Operasi
0	0	Seleksi IR, IR Write Display Clear
0	1	Busy Flag (DB7), @ Counter (DB0-DB7) Read
1	0	Seleksi DR, DR Write
1	1	Seleksi DR, DR Read

- Busy Flag

  Busy Flag menunjukkan bahwa modul siap untuk menerima instruksi selanjutnya sebagaimana terlihat pada tabel diatas. Register seleksi sinyal akan melalui DB7 jika RS=0 dan R/W=1. Jika bernilai 1 maka sedang melakukan kerja internal dan instruksi tidak akan dapat diterima, oleh karena itu status dari flag harus diperiksa sebelum melaksanakan instruksi selanjutnya.

- Address Counter (AC)

  AC menunjukkan lokasi memori dalam modul LCD. Pemilihan lokasi alamat lewat Ac diberikan lewat register instruksi (IR) ketika data pada A, maka AC secara otomatis menaikkan atau menurunkan alamat tergantung dari Entry Mode Set.

- Display Data RAM

  Pada LCD, masing-masing line memiliki range alamat tersendiri. Alamat itu diekspresikan dengan bilangan hexadesimal. Untuk line 1 range alamat berkisar antara 40_H-$4F_H$.

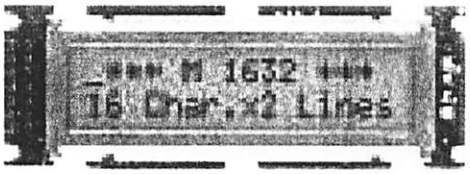
- Character Generator ROM (CG ROM)

  CG ROM memiliki tipe dot matrik 5x7, dimana pada LCD telah tersedian ROM sebagai pembangkit karakter dalam kode ASCII.

- Character Generator RAM (CG RAM)

  CG RAM dipakai untuk pembuatan karakter tersendiri melalui program.

  Adapun bentuk fsik dari LCD M1632 adalah pada gambar berikut :

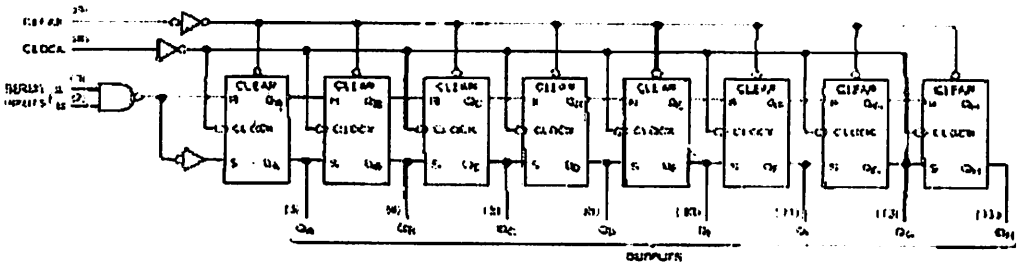


Gambar 2.15. Liquid Crystal Display[3]

2.5 *Shif Register*

Shif register disini menggunakan IC 74LS164 yang merupakan register geser 8 bit yang memiliki jalan masuk deret tergerbang. IC74164 digunakan untuk menampilkan karakter yang dikehendaki oleh mikrokontroler. Ketika terdapat data pada jalur data, data tersebut akan ditahan dengan memberikan *clock* pin E pada LCD. Pin RS menentukan apakah data yang ditahan akan digunakan sebagai instruksi untuk mengatur *setting* tampilan pada LCD atau sebagai kode karakter yang diperlukan LCD untuk menampilkan suatu karakter. Sedangkan untuk pin R/W pada LCD dihubungkan ke *ground* karena dalam hal ini LCD hanya melakukan operasi write atau operasi menampilkan karakter. Kelengkapan IC 74LS164 adalah :

- Gerbang (*enable/disable*) secara serial.
- Jam secara penuh buffered dan masukan serial.
- Tidak *synchronous.*
- Clock frekuensi 36 MHz



Gambar 2.16 Blok diagram IC 74LS164[4]

BAB III

PERANCANGAN DAN PEMBUATAN ALAT

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

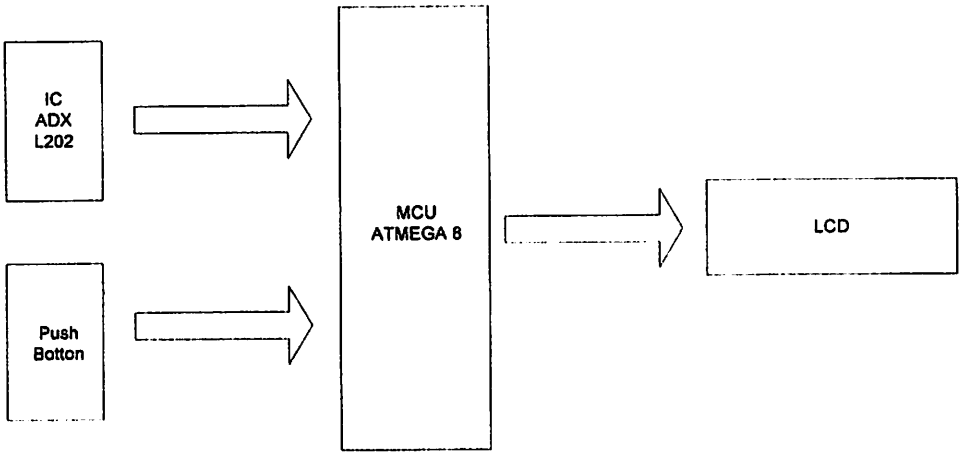
1. Perancangan Perangkat Keras (*Hardware*)
2. Perancangan Perangkat Lunak (*Software*)

Pada perancangan perangkat keras akan meliputi seluruh peripheral yang digunakan pada sistem ini. Pada perancangan perangkat lunak akan meliputi diagram alir dari software secara umum. Kedua perangkat ini dalam kerjanya akan saling menunjang satu sama lain.

3.1. Perancangan Perangkat Keras

3.1.1. Diagram Blok Sistem

Diagram blok sistem secara umum terdiri dari rangkaian sensor yaitu IC ADXL202 dan mikrokontroler unit dengan LCD sebagai tampilannya.



Gambar 3.1. Blok Diagram Sistem

Fungsi dari tiap-tiap blok diagram dijelaskan sebagai berikut :

1. **IC ADXL202**

   Berfungsi sebagai sensor pendeteksi langkah dan jarak yang akan memberikan data masukan digital berupa sinyal PWM ke mikrokontroller untuk diolah lebih lanjut.

2. **Push button**

   Dalam perancangan ini tombol push button memiliki 2 fungsi yaitu

   a. Tombol reset

   mereset mikrokontroller menjadi keadaan seperti semula dan mulai menghitung sinyal yang dikelurakan oleh ic ADXL202 dan tombol

   b. Tombol on/of

   Menghidupkan dan mematikan mikrokontroler

3. **Mikrokontroller ATmega 8-bit**

   Berfungsi sebagai pusat kontrol dan pengolah data lalu menampilakannya ke LCD

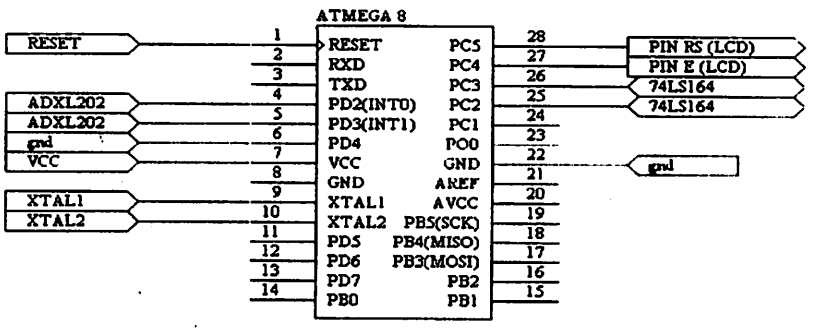
4. LCD (liquid crystal display)

LCD atau liquid crystal display adalah layar cristal cair yang digunakan menampilkan suatu data yang dikirim oleh mikrontroller sehingga langkah dan jarak dapat kita ketahui

3.1.2. Perancangan Rangkaian Mikrokontroler ATmega8

3.1.2.1. Rangkaian ATMEGA 8

Pada rangkaian ini komponen utamanya adalah unit Mikrokontroler ATmega8. Komponen ini merupakan sebuah *chip* tunggal sebagai pengolah data dan pengontrolan alat. Sebagai pengolah data dan pengontrolan sistem, pin-pin mikrokontroler ATmega8 dihubungkan pada rangkaian pendukung membentuk suatu sistem, yang ditunjukkan pada Gambar 3.2 di bawah ini:



Gambar 3.2. Rangkaian Mikrokontroler ATmega8

3.1.2.2.Rangkaian Reset

Rangkaian reset dirancang agar mempunyai kemampuan power on reset yaitu reset yang terjadi pada saat system di nyalakan untuk pertama kalinya. Reset juga dapat dilakukan secara manual dengan menambahkan tombol reset berupa switch button

Untuk me-*reset* mikrokontroler atmega8, maka pin Reset diberi logika tinggi selama sekurangnya dua siklus mesin (24 periode osilator). Untuk membangkitkan sinyal *reset* kapasitor dihubungkan dengan V_{CC} dan sebuah resistor yang dihubungkan ke *ground.*

Rangkaian ini terbentuk oleh komponen resistor dan kapasitor yang sudah ditetapkan oleh atmega. Nilai resistor yang dipakai adalah 10 kΩ dan untuk nilai kapasitornya sebesar 10 μF.

$$f_o = \frac{1}{1.1RC}$$

$$= \frac{1}{1.1.10.10^3.10.10^{-6}}$$

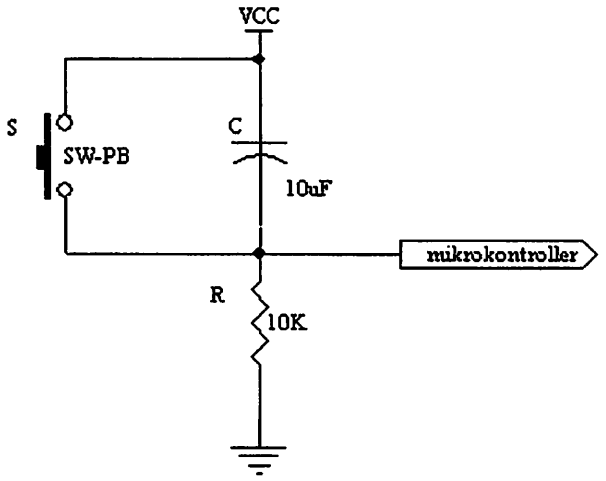
$$= 90{,}9\ Hz$$

$$\text{Maka periode clock} = \frac{1}{f}$$

$$T = \frac{1}{90{,}9}$$

$$T = 0{,}011\ detik$$

Rangkaian reset ditunjukkan dalam gambar di bawah ini :



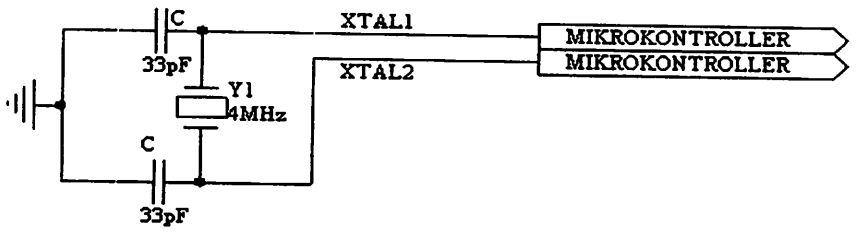
Gambar 3.3 Perencanaan Rangkaian Reset

3.1.2.3 Rangkaian Clock

Kecepatan proses yang diperlukan oleh mikrokontroller atmega8 ditentukan oleh sumber *clock* yang mengendalikan mikrokontroller tersebut. Untuk kristal *clock* dipasang Kristal dan resonator keramik yang berfungsi sebagai pembangkit *clock* osilator yang ada pada mikrokontroller.

Rangkaian ini terdiri dari dua buah kapasitor dan sebuah kristal. Untuk mengendalikan frekuensi osilatornya cukup dengan menghubungkan Kristal pada pin 9 ($P47/X_{out}$) dan pin 10 ($P46/X_{in}$) serta dua buah kapasitor ke *ground*.

Dalam minimum kristal ini, menggunakan kristal 4 Mhz dan $C_1 = C_2$ yaitu sebesar 33 pF. Dengan rangkaian sebagai berikut :



Gambar 3.4 Perencanaan Rangkaian *Clock*

Dengan menggunakan nilai kristal dan kapasitor di atas maka dapat dihitung waktu yang diperlukan untuk 1 siklus mesin yaitu :

Diketahui : F = 4 MHz

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

$$\text{Maka } T = \frac{1}{4\ Mhz} = \frac{1}{4}\ \mu s$$

Maka untuk 1 siklus mesin dari mikrokontroller atmega8 adalah sebesar:

$$T_{si} = 4 \times T$$

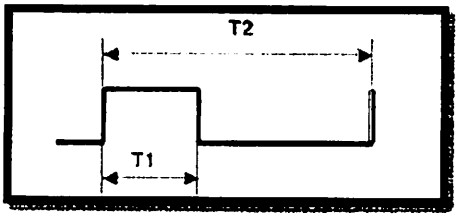
$$= 4 \times \frac{1}{4}\ \mu s$$

$$= 1\ \mu s$$

3.1.3. Perancangan Rangkaian IC ADXL202 (sensor percepatan)

IC ADXL202 ini berfungsi sebagai sensor yang mendeteksi gerakan vertikal seseorang menjadi percepatan dan memiliki output sinyal berupa lebar pulsa . perancangan rangkaian ic ini berfungsi Untuk menentukan periode atau T2 dan bandwidth

- Menentukan periode



Gambar 3.5. Sinyal PWM[2]

Periode atau T2 adalah panjang total dalam satu pulsa, nilainya pun tidak akan berubah sedangkan T1 adalah panjang pulsa ketika dalam kondisi high dan keadaannya akan berubah sesuai dengan keadaan.Dalam data sheet ADXL202 telah diberikan rumus untuk menentukan T2 yaitu

$$T2 = \frac{R_{SET}(\Omega)}{125\ M\Omega}$$ [2]

Untuk menghsilkan 1 ms maka nilai Rset adalah

Rset =T2 x125MΩ

Rset =1ms x125 MΩ

Rset = 125 KΩ

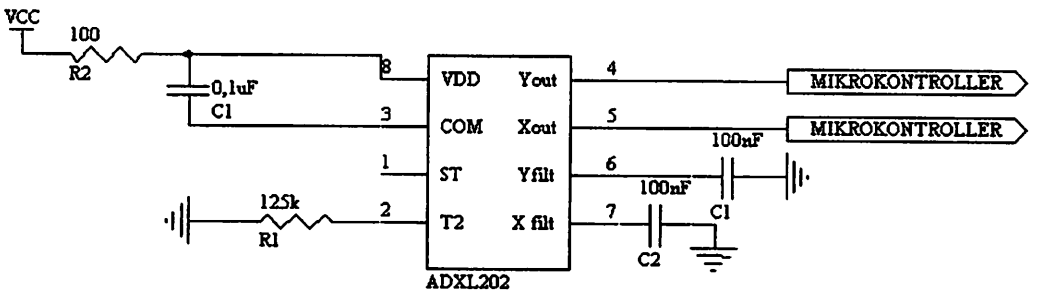
•Menentukan bandwidth

Bandwidth digunakan Jika ingin menggunakan tegangan outputnya adalah analog dan berikut tabel nilai kapasitornya untuk low pass filter yang sudah terintegrasi didalam ic ini.

Tabel 3.1. Pengaturan bandwidth IC ADXL202[2]

Bandwidth	Capacitor Value
10 Hz	0.47 µF
50 Hz	0.10 µF
100 Hz	0.05 µF
200 Hz	0.027 µF
500 Hz	0.01 µF
5 kHz	0.001 µF

Berikut ini adalah gambar rangkain lengkapnya

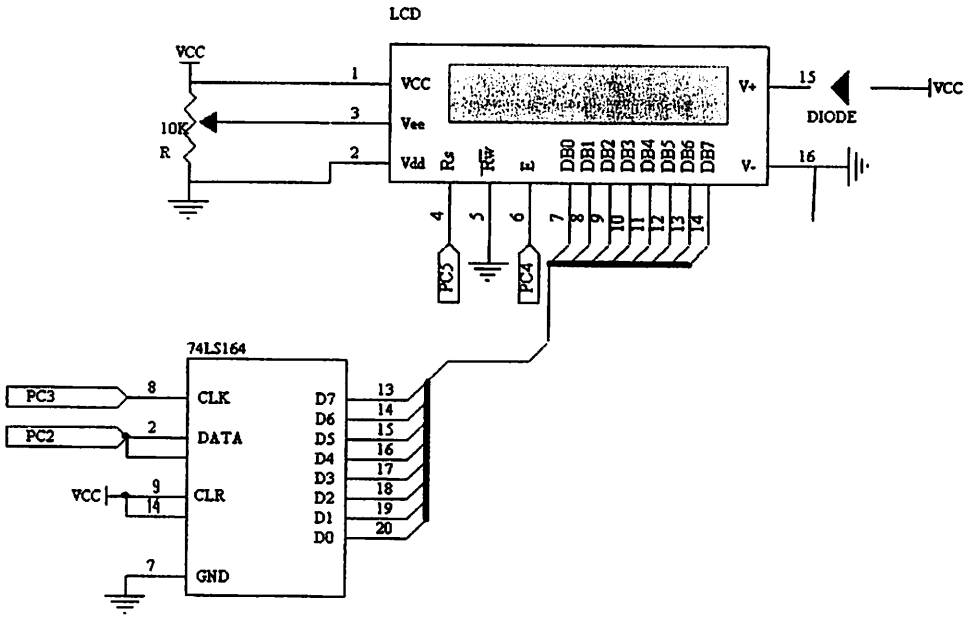


Gambar 3.6. Rangkaian ADXL202

Adapun Rangkaian ADXl202 ditunjukkan dalam Gambar di atas. dengan nilai komponen Cx dan Cy sebesar 0,1µF yang berguna sebagai filter kapasitor dengan bandwidth 50 Hz.

3.1.4 Perencanaan Rangkaian *Liquid Crystal Display* (LCD)

Dalam aplikasi ini menggunakan sebuah layar LCD (*Liquid Crystal Display*) yaitu M1632 yang merupakan LCD dua baris dengan setiap barisnya terdiri 16 karakter dan menggunakan IC 74LS164 yang merupakan register geser 8 bit yang memiliki jalan masuk deret tergerbang. Gambar hubungan antara LCD, IC74LS164 dan mikrokontroler dapat dilihat dalam gambar berikut :



Gambar 3.7. Perencanaan Rangkaian LCD

LCD dot matrik ini membutuhkan sepuluh buah pin masukan/keluaran dari mikrokontroler dan IC 74164 Adapun dua buah pin yakni port PC5 pada penyemat RS yang digunakan sebagai sinyal pemilih register dan port PC4 pada penyemat Enable digunakan sebagai sinyal operasi awal, sinyal enable ini mengaktifkan data tulis atau baca oleh mikrokontroler, pin DB0-DB7 yang dihubungakan ke pin data IC74164 digunakan untuk menampilkan karakter yang dikehendaki oleh mikrokontroler. Ketika terdapat data pada jalur data, data tersebut akan ditahan dengan memberikan *clock* pin E pada LCD. Pin RS menentukan apakah data yang ditahan akan digunakan sebagai instruksi untuk mengatur *setting* tampilan pada LCD atau sebagai kode karakter yang diperlukan LCD untuk menampilkan suatu karakter. Sedangkan untuk pin R/W pada LCD

dihubungkan ke *ground* karena dalam hal ini LCD hanya melakukan operasi write atau operasi menampilkan karakter.

Untuk pin Vcc pada LCD dihubungkan ke supply +Vcc dan Vss dihubungkan ke *ground.* Pin V_{EE} beserta pin Vcc dan Vss dihubungkan ke *trimer potensio* atau kadang disebut dengan *trimpot. Trimpot* ini digunakan untuk mengatur kontras dari tampilan LCD dengan cara mengubah tegangan pada pin V_{EE}. Daftar tabel fungsi penyemat pada LCD dapat dilihat dalam Tabel 3.1.

Tabel 3.2. Fungsi penyemat LCD[3]

Pin	Fungsi
DB0 – DB7	Merupakan saluran data, berisi perintah dan data yang akan ditampilkan di LCD.
Enable	Sinyal operasi awal, sinyal ini mengaktifkan data tulis atau baca.
R/W	Sinyal seleksi tulis atau baca 0: tulis 1: baca
RS	Sinyal pemilih *register* 0: masukan data 1: masukan instruksi

Seperti telah disebutkan sebelumnya bahwa data yang terdapat pada jalur data selain dianggap sebagai kode karakter dapat digunakan sebagai suatu perintah instruksi untuk mengatur setting dari tampilan LCD. Cara pemakaian data antara sebagai instruksi dengan kode karakter berbeda. Perbedaan hanyalah keadaan pin RS ketika data yang ada dijalur data ditahan oleh LCD dengan memberikan *clock* pada pin E.

Pin – pin yang digunakan adalah

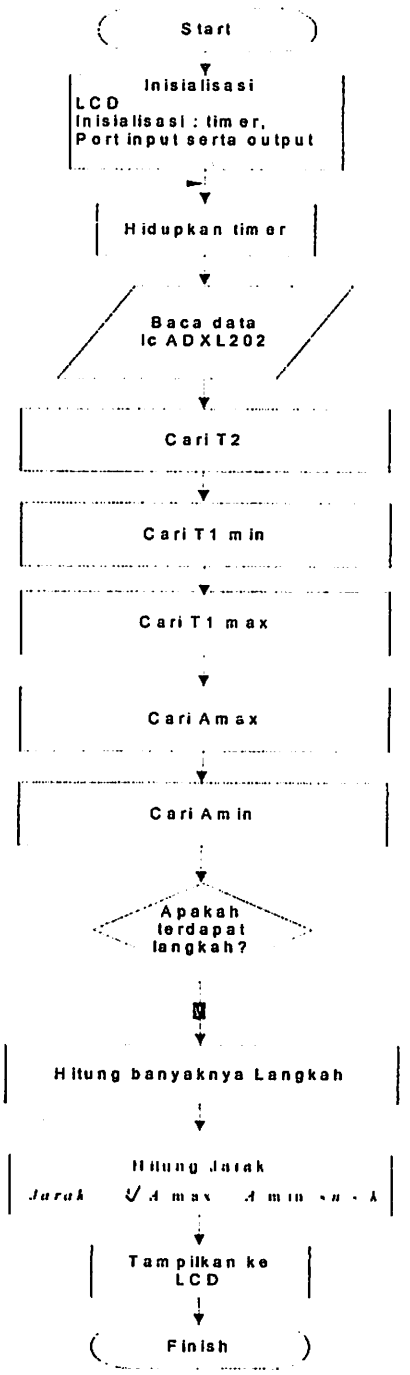
- Pin DB0-DB7 terhubung pada IC 74LS164 yang kemudian pada IC tersebut terhubung pada renesas dengan pin 2 (Data) terhubung pada P 3.0 dan pin 8 (CLK) terhubung pada port 3.1 sedangkan pin 7 (Ground).
- Pin Enable pada LCD terhubung pada renesas yaitu port 3.2.
- Pin RS pada LCD terhubung pada renesas yaitu port 3.3

3.2. Perancangan Perangkat Lunak

Perangkat lunak ini berdasarkan pengendali utama yaitu mikrokontroller Atmega 8. Pembuatan perangkat lunak sistem aplikasi berdasarkan pada semua kejadian yang harus dikerjakan perangkat keras.

Untuk mikrokontroller ATMEGA 8 bahasa yang digunakan adalah bahasa pemrograman *bahasa c* lalu di kompile menjadi bahasa assembly dan di isikan ke mikrokontroller.

3.2.1 Flowchart mikrokontroller



Gambar 3.8. Flowchart Mikrokontroler

BAB IV

PENGUKURAN DAN PENGUJIAN

4.1. Tujuan

Untuk memastikan sistem aplikasi pedometer digital ini dapat bekerja sesuai dengan spesifikasi perencanaan, diperlukan serangkaian pengujian dan pengukuran.

Bab pengujian dan pengukuran ini menguraikan tentang bagian alat yang diuji, tujuan pengujian, langkah-langkah pengujian dan hasil pengujian yang menunjukkan unjuk kerja dari tiap-tiap bagian alat. Pembahasan dalam bab ini dibagi menurut pembagian alat yang diuji untuk mengetahui unjuk kerja sistem secara keseluruhan.

Pengujian sistem tersebut adalah pengujian perangkat keras (hardware) per blok dan alat secara keseluruhan. Berikut ini akan diberikan prosedur pengujian dan hasil pengamatan terhadap pengujian.

4.2 Tujuan Pengujian

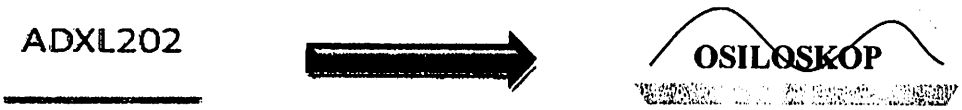
4.2.1 Tujuan Pengujian

Pengujian pada masing-masing blok pada perencanaan dan pembuatan alat pedometer digital ini bertujuan:

- Untuk mengetahui apakah tiap blok rangkaian dapat berfungsi dengan baik.
- Untuk mengetahui sistem kerja pedometer digital dengan menggunakan mikrokontroller ATMEGA 8 sebagai pengendali utama.

4.2.2. Pengujian Rangkaian Sensor

Pengujian rangkaian sensor dimaksudkan untuk mengetahui apakah sensor ADXl202 dapat bekerja dengan baik dalam menentukan atau mengukur percepatan baik static maupun dynamic akselerasi kemiringan sehingga dapat mengetahui berapa tegangan output dari sensor Adxl202 pada setiap perubahan sudutnya .dalam prosedure pengujian ini penulis menggunakan ossiloskop digital sebagai pendeteksi sinyal keluarannya.



Gambar 4.1. Blok Diagram Pengujian Rangkaian Sensor

1. Alat yang digunakan :

- Rangkaian sensor *ADXl202* dan *ossiloskop digital*
- Multimeter Digital SANWA CD800a
- Catu daya 5 volt

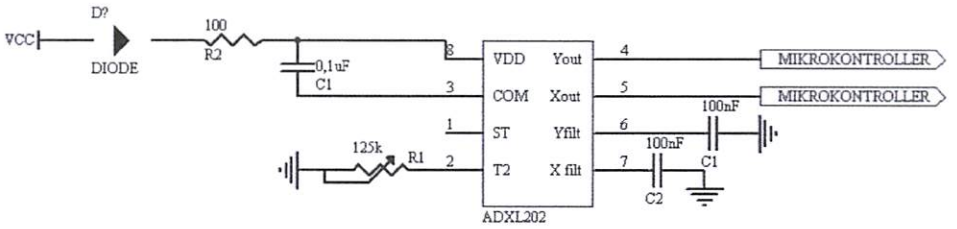
2. Metode Pengujian :

-Gambar 4.2 menunjukkan rangkaian sensor yang terdiri dari sensor *ADXl202* dan *ossiloskop digital*

3. Langkah Pengujian :

- Merangkai rangkaian ADXL202 seperti pada gambar 4.2
- Memasang catu daya DC 5V pada rangkaian.

- Mengukur tegangan keluaran dengan ossiloskop apabila terjadi perubahan gerakan yang terdapat pada rangkaian sensor ADXL202



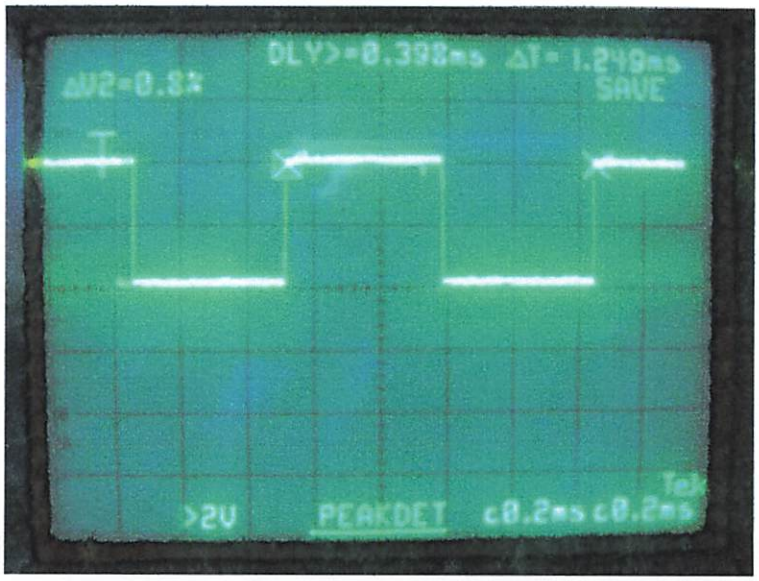
Gambar 4.2 Rangkaian sensor ADXL202

4. Hasil dan Analisa Pengujian sensor

Cara pengujian sensor ditunjukkan pada gambar dibawah ini

Gambar 4.3 Pengujian rangkaian sensor ADXL202 pada ossiloskop

Sebelum mengetehui besarnya perubahan sinyal kita harus mengukur lebar sinyal dalam 1 cycle atau yang disebut T2



Gambar 4.4 Lebar Sinyal T2

Dari hasil percobaan diatas didapatkan nilai T2 sebesar 1.25 ms dan dalam data sheet ADXL202 telah diberikan rumus untuk menentukan T2 yaitu

$$T2 = \frac{R_{SET}(\Omega)}{125\,M\Omega}$$

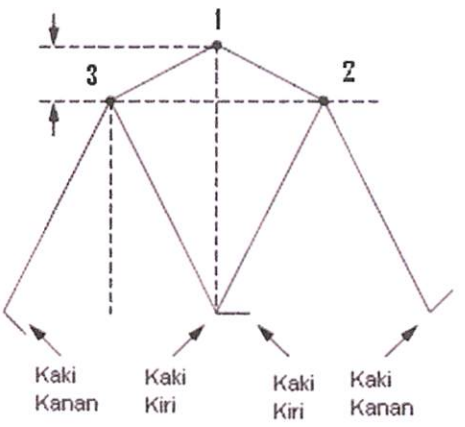
Sehingga dengan Rset sebesar 125kΩ didapat nilai T2

$$T2 = \frac{125k\Omega}{125M\Omega} = 1 \text{ ms}$$

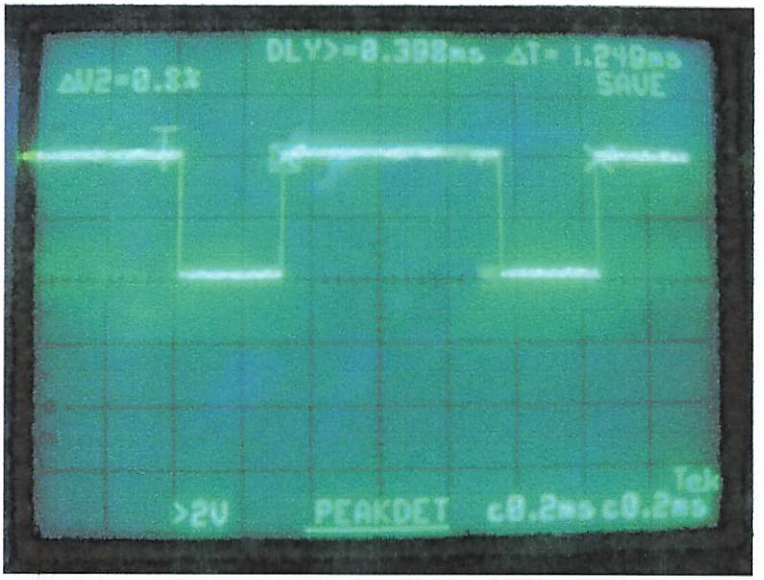
Terdapat 3 pengujian Untuk menguji keluaran dari sensor ini yaitu kondisi :

1. Pada saat kaki sejajar pada atau pada saat 0 gravitasi
2. Pada saat sensor berada didepan
3. Pada saat sensor berada dibelakang

Seperti ditunjukkan pada gambar 4.5



Gambar 4.5. Gerakan langkah kaki



Gambar 4.6 Perubahan sinyal T1 pada saat posisi dirubah

Dari gambar 4.6, dapat dilihat bahwa outputan sensor dapat berubah-ubah dipengaruhi oleh posisi kemiringan sensor, maka dapat dikatakan bahwa sensor telah dapat bekerja dengan baik, dan pada osiloskop ditampilkan nilai T2 sebesar 1,249 ms (pada osiloskop ditunjukkan dengan variabel ΔT).

Pada pengujian ini didapatkan nilai T2, T1, Ax, dan Sudut dalam satuan derajat ditunjukkan pada table 4.1.

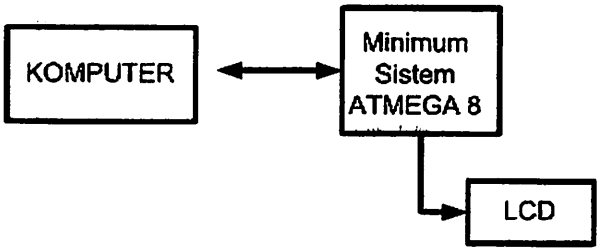
Tabel 4.1 Hasil pengujian output sensor adxl202 terhadap sudut kemiringan

T2(ms)	T1(ms)	Ax(m/s^2)	Sudut(°)			
			Percobaan	Perhitungan	selisih	Error(%)
1.25	0.703	0.503	30	29.34	0.66	2.24
1.25	0.735	0.707	45	44.74	0.26	0.58
1.25	0.761	0.870	60	60.45	0.45	0.74
1.25	0.7814	1.00	90	90	0	0
1.25	0.625	0	0	0	0	0
1.25	0.546	-0.503	-30	-30	0	0
1.25	0.514	-0.707	-45	-45.2	0.2	0.44
1.25	0.489	-0.866	-60	-60.45	0.45	0.744
1.25	0.468	-1	-90	-90	0	0

Sehingga rata-rata kesalahan sudut terdapat sebesar 0.52%

4.3 Pengujian Rangkaian LCD (*Liquid Crystal Display*).

Pengujian ini bertujuan untuk mengetahui apakah rangkaian LCD dapat menampilkan data karakter sesuai dengan program dan mengukur besarnya tegangan yang dibutuhkan oleh LCD.



Gambar 4.7 Diagram Blok Pengujian LCD

1. Alat yang digunakan :

1. Komputer.
2. Minimum sistem *Mikrokontroler* ATMEGA 8

3. Voltmeter.

2. Metode Pengujian :

Terdapat 2 metode pengujian LCD ini yaitu

1. Pengujian secara software yaitu dengan mengisi program di mikrokonterdller lalu di kirimkan ke LCD
2. Pengujian secara hardware yaitu dengan menguji besarnya tegangan yang terdapat pada backlight LCD

3. Langkah Pengujian :

1. Merangkai peralatan seperti pada gambar 4.11.
2. Membuat perangkat lunak pengujian rangkaian LCD, program ini berisi inisialisasi *mikrokontroler* dan LCD.
3. Mengaktifkan catu daya.
4. Mengoperasikan program dan hasil keluaran akan ditunjukkan pada layar penampil kristal cair.
5. Mengukur besarnya tegangan yang pertama pada kaki anoda, yang kedua pada kaki katoda, dan yang ketiga mengukur tegangan jatuh tegangan pada diode silikon yaitu dengan cara mengubungkan jemper posltif pada kaki anoda dan jemper negatif pada katoda.

Blok pengujian LCD ditunjukkan pada gambar 4. 10 dibawah ini.

Berikut ini program pengujian *inisialisasi* LCD:

```
#include <mega8.h>
#include "lcdku.c"
{
cetak(1,1,"  ITN MALANG  ");
 delay_ms(2000);
```

}

maka pada LCD M1632 tampil tulisan sebagai berikut :

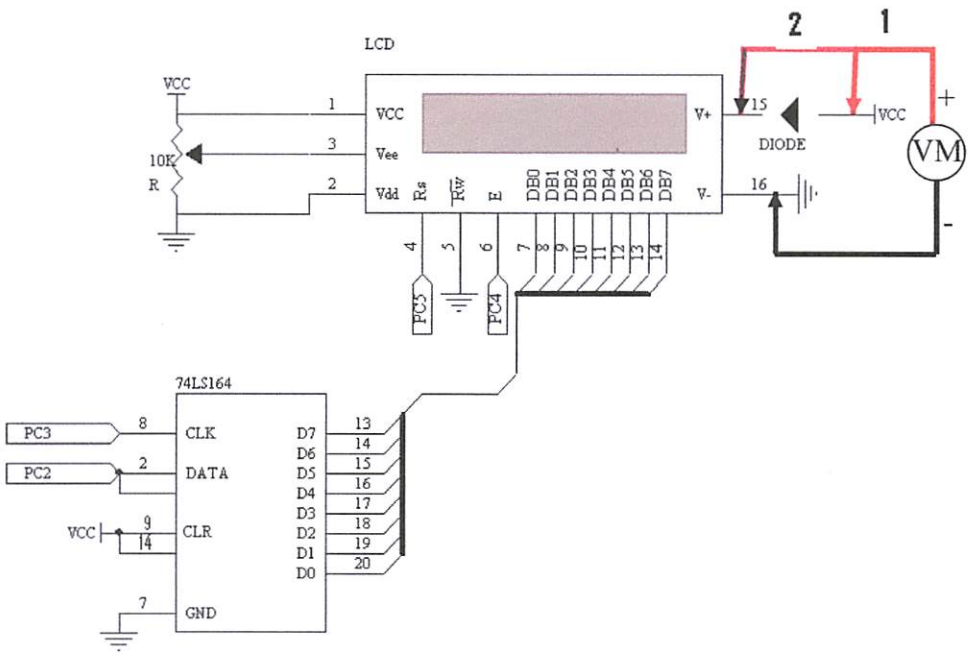
ITN MALANG

4. Hasil dan Analisis pengujian LCD

Dari hasil pengujian didapatkan bahwa rangkaian LCD dapat menampilkan karakter-karakter sesuai dengan data yang dikirimkan oleh MCU. Tampilan penampil kristal cair terdiri atas 2 baris yang masing-masing mempunyai 16 karakter. Hasil pengujian LCD dapat ditunjukkan pada gambar 4.8.

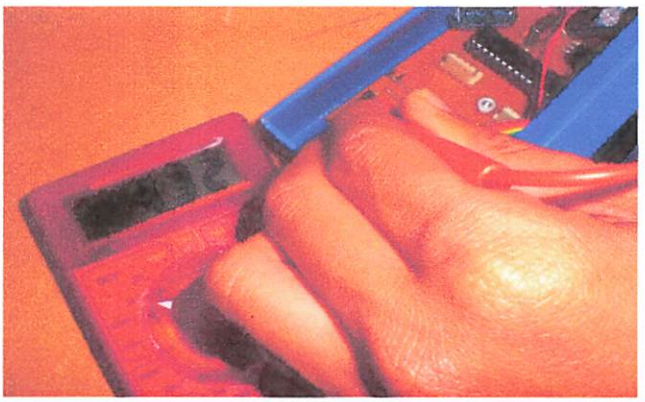
Gambar 4.8 Gambar Hasil Pengujian LCD

Besarnya tegangan yang dibutuhkan untuk mengatur pengendalian kecerahan latar belakang pada LCD harus sesuai dengan datasheet LCD M1632. Gambar rangkaian untuk pengukuran besarnya tegangan pada rangkaian LCD ditunjukkan pada gambar 4.9 di bawah ini:



Gambar 4.9 Pengujian LCD

Gambar 4.10 Pengukuran Vcc



Gambar 4.11 Pengukuran Pada anoda dan katoda

Gambar 4.12 Pengukuran Pada dioda

Tabel 4.2 Hasil Pengukuran Pengujian Rangkaian LCD

No	Tegangan pada kaki Anoda (Volt)	Tegangan pada kaki Katoda (Volt)	Tegangan pada kaki Anoda dan Katoda (Volt)
1	5,03	4,32	0,71

Dari hasil pengujian pengukuran dapat disimpulkan bahwa rangkaian LCD dapat bekerja sesuai dengan perencanaan.

4.4 Pengujian system pedometer digital

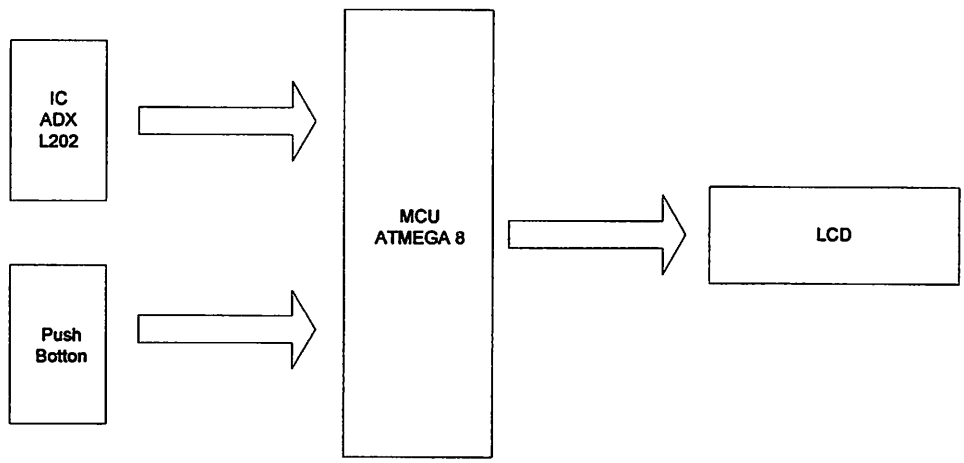
Tujuan dari pengujian ini adalah untuk mengetahui apakah semua sistem berjalan dengan normal dan juga untuk mengetahui *kesalahan* yang terjadi ketika seseorang melangkah.

1. Alat yang digunakan :

- Rangkaian sensor *ADXL202*
- rangakaian mikrokontroller dan LCD
- Catu daya 5 volt

2. Metode Pengujian :

Dalam metode ini sensor ADXL202 di pasang di dikaki lalu melakukan gerakan berjalan, sehingga menghasilkan gerakan langkah kaki yang akan ditampilkan dengan menggunakan lcd dan mikrokontroller sebagai processing unitnya Gambar 4.13 menunjukkan diagram blok dari keseluruhuan system yang akan di uji.



Gambar 4.13 Diagram blok pedometer digital

3. Langkah Pengujian :

- Merangkai rangkaian detektor seperti pada skema yang terdapat pada lampiran
- Memasang catu daya DC 5V pada rangkaian.
- memasang rangkaian sensor adxl202 di kaki

4. Hasil dan Analisa Pengujian rangkaian pedometer digital

4.1 Hasil Pengujian rangkaian pedometer digital

a. Sebelum terjadi koneksi dengan rangkain sensor ADXL202, LCD akan menampilkan tulisan seperti gambar 4.14

Gambar 4.14 Tampilan LCD saat awal

b. Setelah rangkaian ADXL202 di hubungkan ke mikrokontroller maka LCD akan menampilkan tulisan seperti gambar 4.15

Gambar 4.15 Tampilan LCD saat sensor dipasang

c. Pada saat melakukan proses berjalan maka langkah yang akan terjadi akan didentifikasi oleh mikrokontroller lalu akan ditampilkan di LCD dalam bentuk tulisan seperti gambar 4.16.

Gambar 4.16 Tampilan LCD saat langkah terjadi

d. Jika langkah yang dijalankan telah memenuhi jarak dalam 2 langkah maka mikrokontroller akan menampilakan pada lcd jumlah jarak dalam sentimeter seperti terlihat dalam gambar 4.17.

Gambar 4.17 Tampilan LCD saat menmpilkan jarak

4.2 Analisa Pengujian rangkaian pedometer digital

untuk analisa pegujian sistem penulis menggunakan jarak dan langkah sebenarnyadan membandingkan jarak dan langkah yang tertulis dalam lcd sehingga kesalahan dalam persen dapat dicari dengan persamaan di bawah ini:

$$\%\ \text{kesalahan} = \frac{(perhitunga\ \ n - pengukuran\ )}{perhitunga\ \ n} \times 100\ \%$$

Data hasil perhitungan dan pengukuran untuk mendetekasi langkah dari sensor ini terdapat pada table 4.3 di bawah ini.

Tabel 4.3. Data Hasil Perhitungan dan Pengukuran langkah kaki

Percobaan ke	Jumlah langkah kaki		% Kesalahan
	Perhitungan	Tampilan pada alat	
1	5	5	0
2	10	10	0
3	15	15	0
4	20	19	5
5	25	23	8
Σ % kesalahan			13
Rata-rata kesalahan			2,6%

Data hasil perhitungan dan pengukuran untuk mendetekasi jarak dari sensor ini terdapat pada table 4.4 di bawah ini:

Tabel 4.4. Data Hasil Perhitungan dan Pengukuran jarak

Percobaan ke	Jumlah jarak(cm)		% Kesalahan
	Perhitungan	Tampilan pada alat	
1	254	255	0.392
2	510	506	0.784
3	765	727	4.967
4	1019	956	6.182
5	1320	1431	7.756
Σ % kesalahan			20.083
Rata-rata kesalahan			4.016

BAB V

PENUTUP

Berdasarkan perancangan, pembuatan, pengujian dan analisis alat sistem pedometer digital dengan menggunakan mikrokontroler ATMega 8 dapat diambil kesimpulan dan saran sebagai berikut.

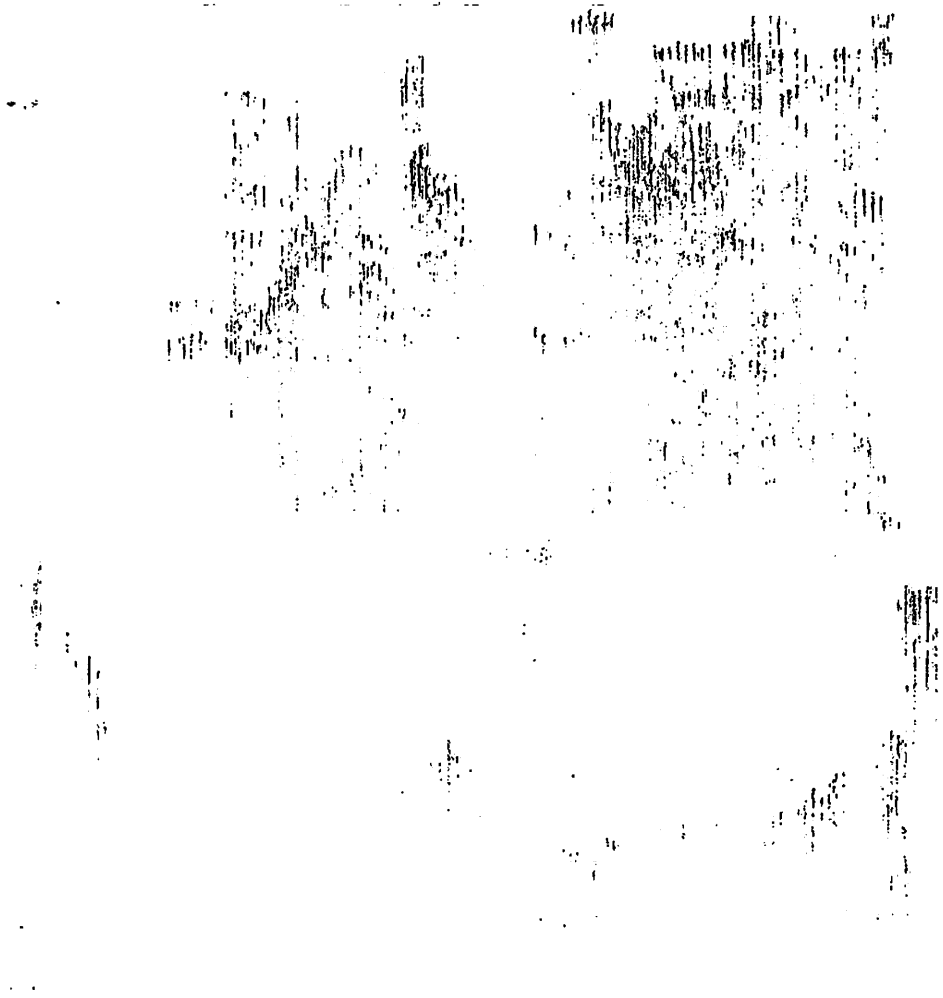
5.1 Kesimpulan

1. Dari hasil percobaan rangkaian pedometer digital terdapat perbedaan T2 hasil pengukuran dan perhitungan,menurut pengukuran T2 di ossiloskop menunjukan 1,249 ms sedangkan menurut perhitungan berdsarakan rangkaian yang di berikan dalam rangkaian sensor ini memiliki T2 sebesarl ms.
2. Mikrokontroler akan menghitung setiap perubahan T1 untuk menghasilkan nilai percepatan maksimal dan minimal yang pada akhirnya dapat menentukan langkah dan jarak berikut percobaannya:

   Pada saat percepatan 0 akan menghasilkan T1 pada dengan AX bernilai 0

   Pada saat percepatan maksimal akan menghasilkan T1 yang bernilai positif

   Pada saat percepatan minimal akan menghasilkan nilai T1 yang bernilai negatif
3. kesalahan yang terjadi pada alat ini dihitung dari Data hasil perhitungan dan pengukuran untuk mendetekasi langkah dari sensor ini dari 5 percobaan terdapat kesalahan dengan jumlah sebesar 13% jadi setiap percobaan memiliki rata-rata kesalahan sebesar 2,6%, sedangkan untuk dari Data hasil perhitungan dan pengukuran untuk mendetekasi jarak dari sensor ini dari 5 percobaan

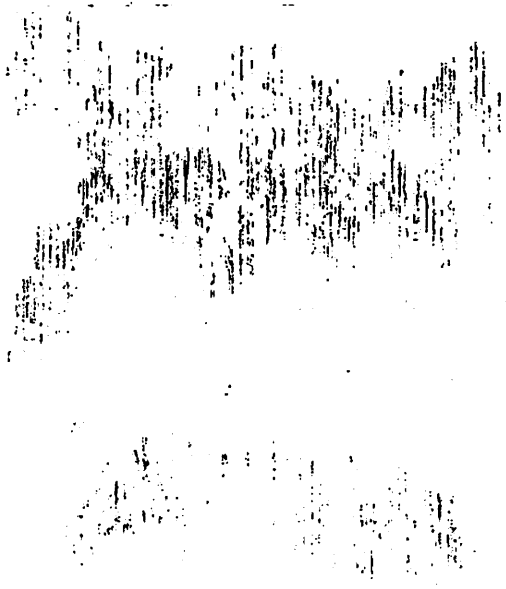


terdapat kesalahan dengan jumlah kesalahan sebesar 20,083% jadi setiap percobaan memiliki rata-rata kesalahan sebesar 4,016%

5.2 Saran

Pada alat hasil perancangan ini masih memhpunyai kekurangan-kekurangan, untuk itu ada beberapa hal yang dapat dilakukan untuk melakukan pengembangan :

1. alat pedometer ini dapat menggunakan LCD yang lebih kecil sehingga dapat mengurangi dimensi agar lebih nyaman digunakan.
2. Hendaknya sistem pedometer ini menggunakan mikrokontroller yang memiliki daya yang lebih kecil sehingga dalam penggunaanya baterai dapat tahan lebih lama.
3. Karena disini hanya menggunakan sudut x atau x –axis untuk lebih presisi alat ini dapat menggunakan menggabungkan sinyal X dan Y sehingga menghailkan Z-axis yang menghasilkan jarak dan langkah lebih akurat.



DAFTAR PUSTAKA

[1] www.atmel.com, Data Sheet ATMEGA8.

[2] www.analog.com, Data Sheet ADXL202.

[3] LCD Module User Manual.

[4] www.alldatasheet.com, Data Sheet 74LS164.

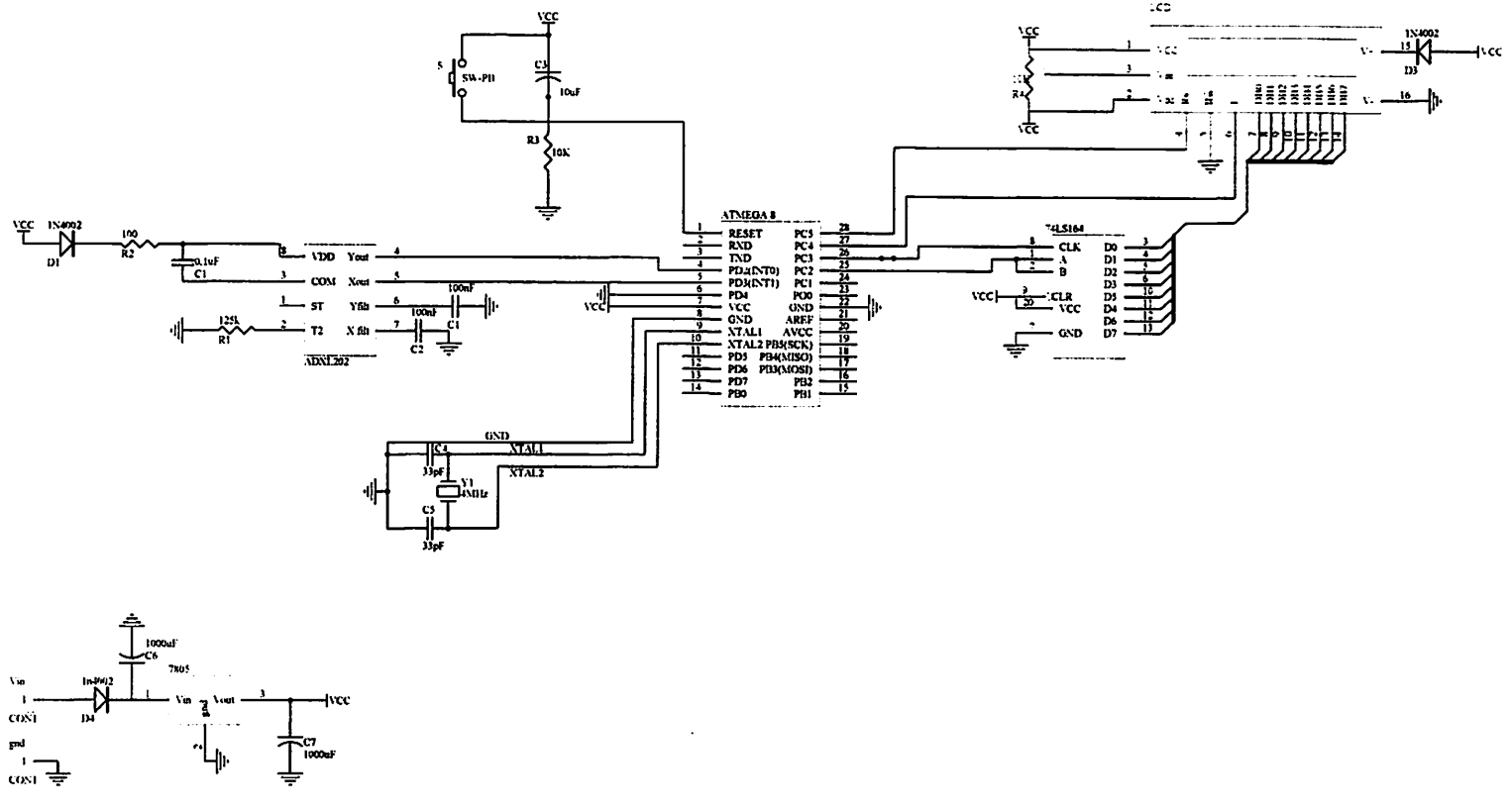
[5] www.analog.com, an-602 application note.

[6] www.analog.com, using the adxl202 duty cycle output.

[7] www.analog.com, Enhancing the Performance of Pedometers Using a Single Accelerometer

[8] Fredrik Brännström, Positioning techniques alternative to GPS, Ericsson AB, Skellefteå,2002

LAMPIRAN



Title		
Size: B	Number:	Revision:
Date: 16-Mar-2008	Sheet of	
File: F:\DATAKU (M)\KULIAH\literatur proposal\Novi\Ngos bab gambar rangkaian ibali.ddb		

Spesifikasi Alat Dan Gambar

Spesifikasi alat

- Dimensi sensor 12,4 x 7,5 x 4 cm
- Dimensi pedometer 5,8 x 3,5 x 3 cm
- Power Supply : 5 volt
- Mikrokontroler ATmegA 8
- LCD M1632
- Sensor ADXL202
- Push button reset
- Saklar on/of
- IC 74HC164

Gambar pedometer digital secara keseluruhan

```
/* File include */
#include <mega8.h>
#include <delay.h>
#include <math.h>
#include "lcdku.c"

/* Pendefinisian */
#define   input           PIND.3
#define   output          PORTC.0

/* Inisialisasi variabel global */
unsigned char maxl,minl,langkah,t2;
float z,x, ;
int menit,pengali;
bit t;

/*Inisialisasi input output */
void init_port()
{
        DDRC=0b00111100;
        DDRB=0b00001100;
        DDRD=0b00000000;
}


void kon(long int n)
{
        int ascii;
        ascii=n/10000+0x30; dataout(ascii,1);
         ascii=n/1000%10+0x30; dataout(ascii,1);
         ascii=n/100%10+48; dataout(ascii,1);
         ascii=n/10%10+48; dataout(ascii,1);
         ascii=n%10+48; dataout(ascii,1);
}

void kon1(long int n)
{
        int ascii;
        ascii=n/10000+0x30; dataout(ascii,1);
         ascii=n/1000%10+0x30; dataout(ascii,1);
         ascii=n/100%10+48; dataout(ascii,1);
         dataout(',',1);
         ascii=n/10%10+48; dataout(ascii,1);
         ascii=n%10+48; dataout(ascii,1);
}

float hitung()
{
amaxl=(maxl/t2)-0.5
aminl=(minl/t2)-0.5
z=(amaxl-aminl)*78.4;
z=sqrt(z);
z=sqrt(z)*0.5;
return z;
}
/* Program Utama */
void main()
{
        int a;
```

```
        /* Inisialisasi */
        init_port();
        initlcd();
        pengali=0;
        TCNT1H=0;
        TCNT1L=0;
        OCR1AH=0;
        OCR1AL=0;
        delay_ms(10);

        #asm("sei")
        cetak(1,1," M.Bali Suryo U ");
        cetak(2,1,"   03.17.034    ");
        delay_ms(2000);
        hapus();
        cetak(1,1,"   ITN MALANG   ");
        delay_ms(2000);
        hapus();
        cetak(1,8,"cm");
        cetak(2,8,"langkah");

 maxl=0;minl=200;langkah=0;t=0;x=0;y=0;
        do
        {
        pengali=0;menit=0;
        while(input==0){menit++;}
        while(input==1){pengali++;}
        t2=menit+pengali;
        if(maxl<pengali) maxl=pengali;
        if(minl>pengali) minl=pengali;
        if((pengali<79)&&(t==0)) {t=1;}
        if((pengali>98)&&(t==1))
{t=0;langkah++;x=hitung()*langkah*100;minl=200;}

        pos(1,1);
        kon(x);
        pos(2,1);
        kon(langkah);
        pos(1,11);
        kon(pengali);

        }while(1);

}
```

DATA SHEET

ADXL202

Low-Cost ±2 *g* Dual-Axis Accelerometer with Duty Cycle Output

ADXL202E*

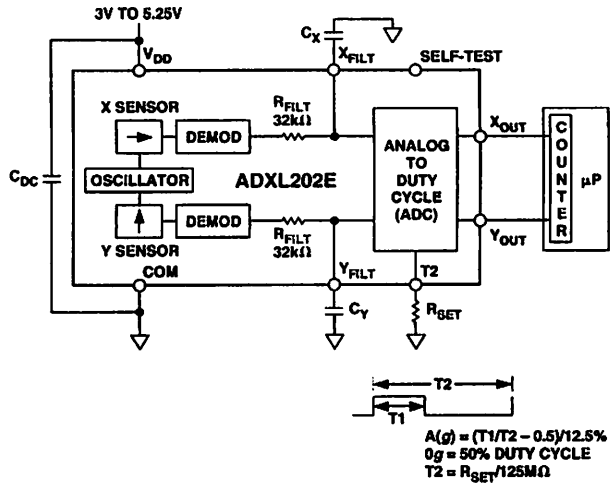
FEATURES

2-Axis Acceleration Sensor on a Single IC Chip
5 mm × 5 mm × 2 mm Ultrasmall Chip Scale Package
2 m*g* Resolution at 60 Hz
Low-Power < 0.6 mA
Direct Interface to Low-Cost Microcontrollers via Duty Cycle Output
BW Adjustment with a Single Capacitor
3 V to 5.25 V Single Supply Operation
1000 *g* Shock Survival

APPLICATIONS

2-Axis Tilt Sensing with Faster Response than Electrolytic, Mercury, or Thermal Sensors
Computer Peripherals
Information Appliances
Alarms and Motion Detectors
Disk Drives
Vehicle Security

FUNCTIONAL BLOCK DIAGRAM



ENERAL DESCRIPTION

ıe ADXL202E is a low-cost, low-power, complete 2-axis acceler-ıeter with a digital output, all on a single monolithic IC. It is an ıproved version of the ADXL202AQC/JQC. The ADXL202E ll measure accelerations with a full-scale range of ±2 *g*. The)XL202E can measure both dynamic acceleration (e.g., vibra-ın) and static acceleration (e.g., gravity).

ıe outputs are analog voltage or digital signals whose duty cycles ıtio of pulsewidth to period) are proportional to acceleration. ıe duty cycle outputs can be directly measured by a micro-ocessor counter, without an A/D converter or glue logic. The ıty cycle period is adjustable from 0.5 ms to 10 ms via a single ıistor (R_{SET}).

The typical noise floor is 200 $\mu g/\sqrt{Hz}$, allowing signals below 2 m*g* (at 60 Hz bandwidth) to be resolved.

The bandwidth of the accelerometer is set with capacitors C_X and C_Y at the X_{FILT} and Y_{FILT} pins. An analog output can be reconstructed by filtering the duty cycle output.

The ADXL202E is available in 5 mm × 5 mm × 2 mm 8-lead hermetic LCC package.

atents Pending

EV. A

One Technology Way, P.O. Box 9106, Norwood, MA 02062-9106, U.S.A.
Tel: 781/329-4700 World Wide Web Site: http://www.analog.com
Fax: 781/326-8703

(L202E–SPECIFICATIONS ($T_A = T_{MIN}$ to T_{MAX}, T_A = 25°C for J Grade only, V_{DD} = 5 V, R_{SET} = 125 kΩ, Acceleration = 0 *g*, unless otherwise noted.)

eter	Conditions	TPC[1] Graph	ADXL202JE Min	ADXL202JE Typ	ADXL202JE Max	ADXL202AE Min	ADXL202AE Typ	ADXL202AE Max	Unit
R INPUT	Each Axis								
urement Range[2]			±2			±2			*g*
inearity	Best Fit Straight Line			0.2			0.2		% of FS
ment Error[3]		X		±1			±1		Degrees
ment Error	X Sensor to Y Sensor			0.01			0.01		Degrees
s-Axis Sensitivity[4]		X		±2			±2		%
TIVITY	Each Axis								
Cycle per *g*	T1/T2, V_{DD} = 5 V	X	10.5	12.5	14.5	10	12.5	15	%/*g*
Cycle per *g*	T1/T2, V_{DD} = 3 V	X	9.0	11	13.0	8.5	11	13.5	%/*g*
ivity X_{FILT}, Y_{FILT}	V_{DD} = 5 V	X	265	312	360	250	312	375	mV/*g*
ivity X_{FILT}, Y_{FILT}	V_{DD} = 3 V	X	140	167	195	140	167	200	mV/*g*
erature Drift[5]	Delta from 25°C	X		±0.5			±0.5		%
BIAS LEVEL	Each Axis								
uty Cycle	T1/T2, V_{DD} = 5 V	X	34	50	66	30	50	70	%
uty Cycle	T1/T2, V_{DD} = 3 V	X	31	50	69	31	50	69	%
oltage X_{FILT}, Y_{FILT}	V_{DD} = 5 V	X	2.1	2.5	2.9	2.0	2.5	3.0	V
oltage X_{FILT}, Y_{FILT}	V_{DD} = 3 V	X	1.2	1.5	1.8	1.2	1.5	1.8	V
uty Cycle vs. Supply		X		1.0	4.0		1.0	4.0	%/V
ffset vs. Temperature[5]	Delta from 25°C	X		2.0			2.0		mg/°C
PERFORMANCE									
Density	@ 25°C	X		200			200	1000	$\mu g\sqrt{Hz}$ rms
ENCY RESPONSE									
Bandwidth	At Pins X_{FILT}, Y_{FILT}			6			6		kHz
r Resonant Frequency				10			10		kHz
Tolerance	32 kΩ Nominal			±15			±15		%
num Capacitance	At Pins X_{FILT}, Y_{FILT}		1000			1000			pF
EST									
Cycle Change	Self-Test "0" to "1"			10			10		%
CYCLE OUTPUT STAGE									
	R_{SET} = 125 kΩ		0.7		1.3	0.7		1.3	kHz
ut High Voltage	I = 25 µA		V_S – 200 mV			V_S – 200 mV			V
ut Low Voltage	I = 25 µA				200			200	mV
rift vs. Temperature				50			50		ppm/°C
all Time				200			200		ns
SUPPLY									
ting Voltage Range			3		5.25	3.0		5.25	V
cent Supply Current				0.6	1.0		0.6	1.0	mA
On Time	C_{FILT} in µF		160 × C_{FILT} + 0.3			160 × C_{FILT} + 0.3			ms
ERATURE RANGE									
ied Performance AE						–40		+85	°C
ting Range			0		70	-40		+85	°C

Performance Characteristics.
eed by measurement of initial offset and sensitivity.
nt error is specified as the angle between the true and indicated axis of sensitivity (see TPC 15).
is sensitivity is the algebraic sum of the alignment and the inherent sensitivity errors.
as the output change from ambient to maximum temperature or ambient to minimum temperature.
tions subject to change without notice.

SOLUTE MAXIMUM RATINGS*

eleration (Any Axis, Unpowered for 0.5 ms) ..... 1000 *g*
eleration (Any Axis, Powered for 0.5 ms) ..... 500 *g*
; ..... –0.3 V to +6.0 V
:put Short Circuit Duration, (Any Pin to Common)
..... Indefinite
:rating Temperature ..... –55°C to +125°C
rage Temperature ..... –65°C to +150°C

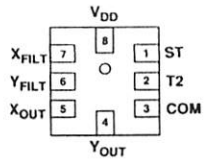
esses above those listed under Absolute Maximum Ratings may cause per ma-
it damage to the device. This is a stress rating only; functional operation of the
ice at these or any other conditions above those indicate in the operational
tions of this specification is not implied. Exposure to absolute maximum rating
ditions for extended periods may affect device reliability.

ops onto hard surfaces can cause shocks of greater than 1000 *g*
exceed the absolute maximum rating of the device. Care
uld be exercised in handling to avoid damage.

Package Characteristics

ckage ight	θ_{JA}	θ_{JC}	Device
ead LCC	120°C/W	tbd°C/W	<1.0 grams

PIN CONFIGURATION



BOTTOM VIEW

PIN FUNCTION DESCRIPTIONS

Pin No.	Mnemonic	Description
1	ST	Self-Test
2	T2	Connect R_{SET} to Set T2 Period
3	COM	Common
4	Y_{OUT}	Y-Channel Duty Cycle Output
5	X_{OUT}	X-Channel Duty Cycle Output
6	Y_{FILT}	Y-Channel Filter Pin
7	X_{FILT}	X-Channel Filter Pin
8	V_{DD}	3 V to 5.25 V

ORDERING GUIDE

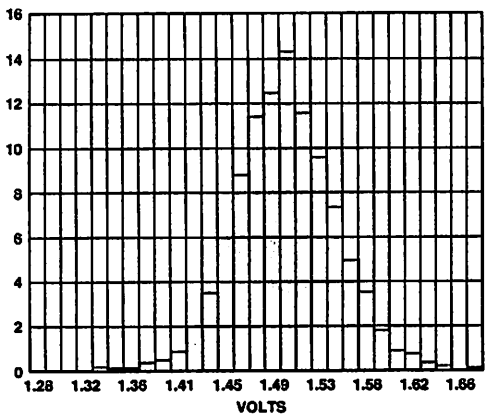
Model	No. of Axes	Specified Voltage	Temperature Range	Package Description	Package Option
ADXL202JE	2	3 V to 5 V	0 to 70°C	8-Lead LCC	E-8
ADXL202AE	2	3 V to 5 V	–40°C to +85°C	8-Lead LCC	E-8

UTION

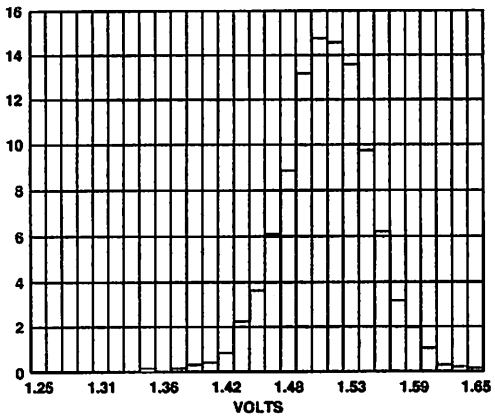
D (electrostatic discharge) sensitive device. Electrostatic charges as high as 4000 V readily umulate on the human body and test equipment and can discharge without detection. Although ADXL202E features proprietary ESD protection circuitry, permanent damage may occur on ices subjected to high-energy electrostatic discharges. Therefore, proper ESD precautions are ommended to avoid performance degradation or loss of functionality.

.202E–Typical Performance Characteristics*

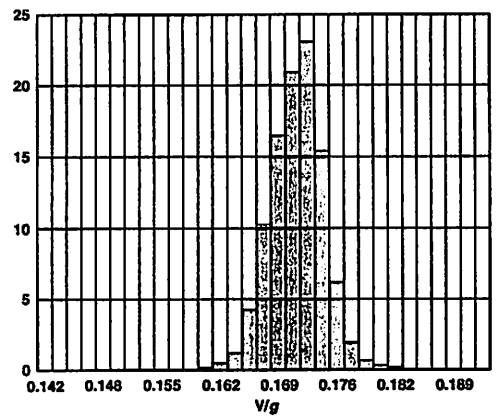
V_{DD} = 3 V



. *X-Axis Zero g Bias Distribution at X_{FILT}, V_{DD} = 3 V*

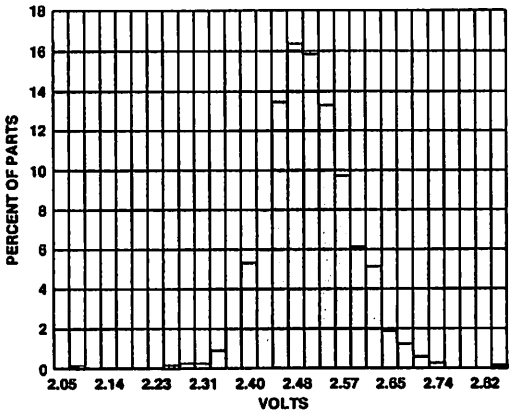


'. *Y-Axis Zero g Bias Distribution at Y_{FILT}, V_{DD} = 3 V*

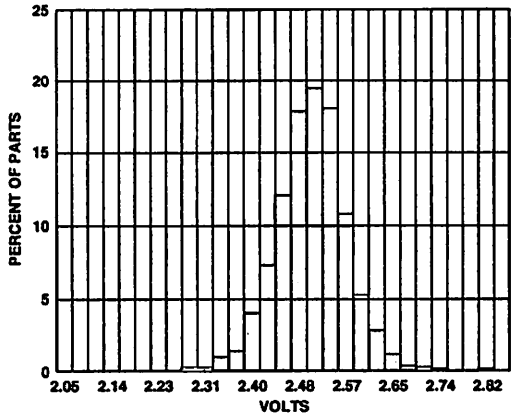


3. *X-Axis Sensitivity Distribution at X_{FILT}, V_{DD} = 3 V*

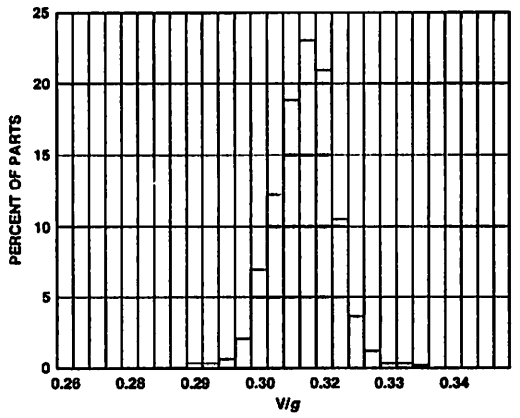
V_{DD} = 5 V



TPC 4. X-Axis Zero g Bias Distribution at X_{FILT}, V_{DD} = 5 V



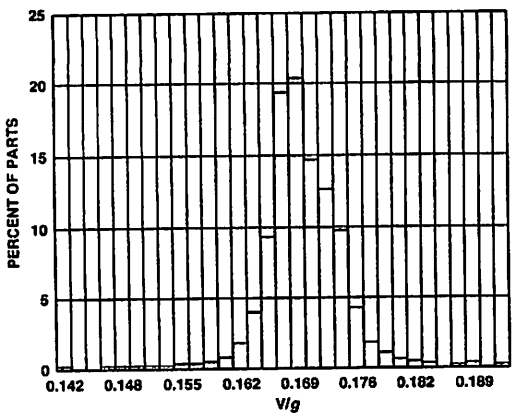
TPC 5. Y-Axis Zero g Bias Distribution at Y_{FILT}, V_{DD} = 5 V



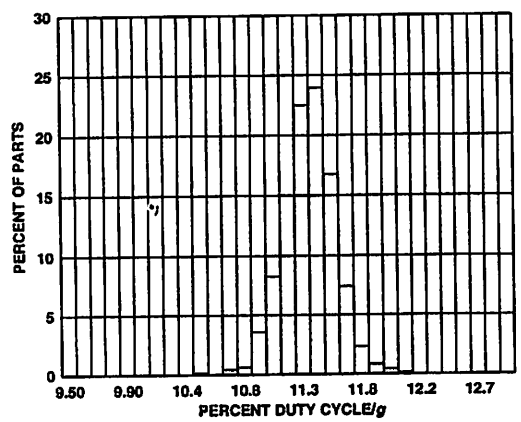
TPC 6. X-Axis Sensitivity Distribution at X_{FILT}, V_{DD} = 5 V

ken from 4500 parts over 3 lots minimum.

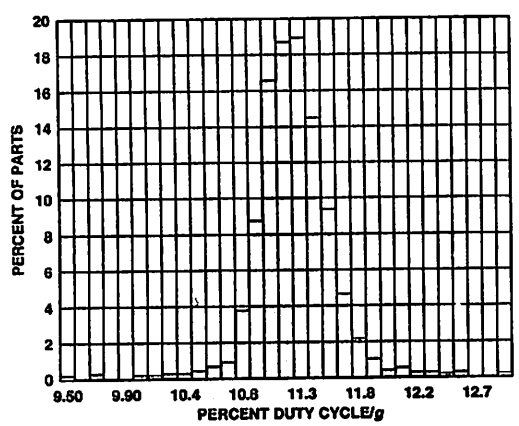
V_{DD} = 3 V



PC 7. Y-Axis Sensitivity Distribution at Y_{FILT}, V_{DD} = 3 V

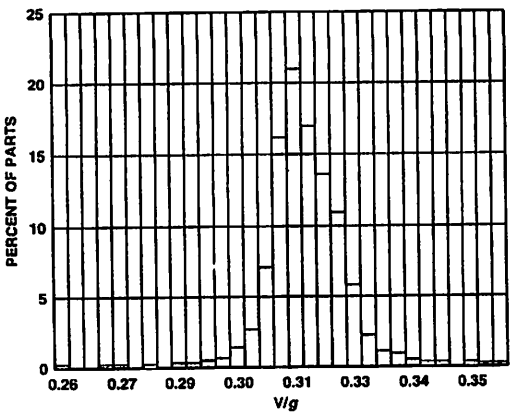


TPC 8. X-Axis Sensitivity at X_{OUT}, V_{DD} = 3 V

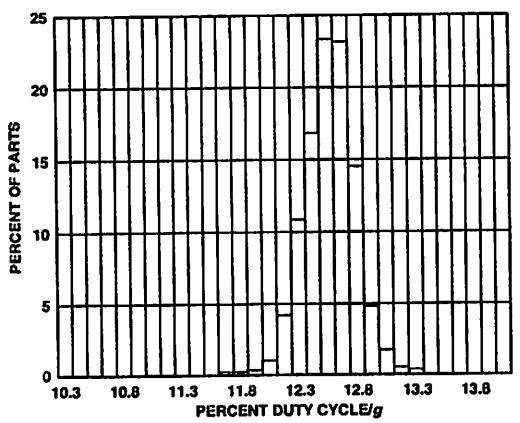


TPC 9. Y-Axis Sensitivity at Y_{OUT}, V_{DD} = 3 V

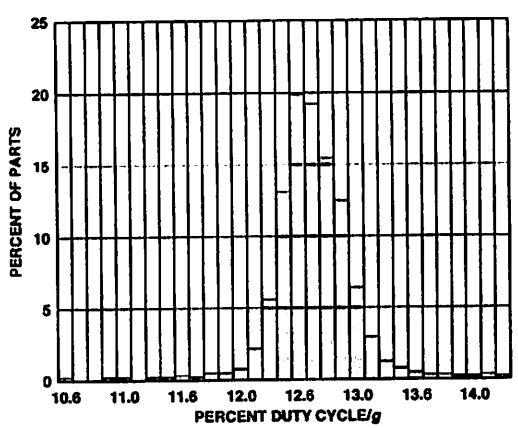
V_{DD} = 5 V



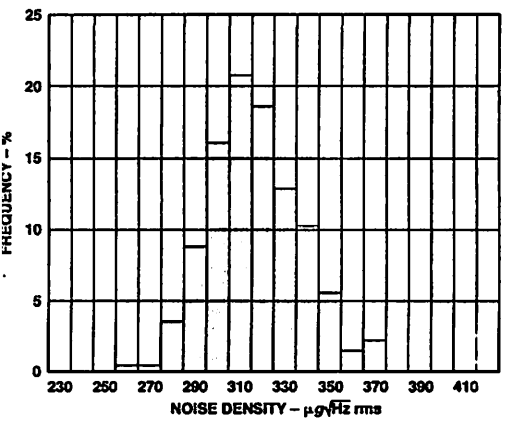
TPC 10. Y-Axis Sensitivity Distribution at Y_{FILT}, V_{DD} = 5 V



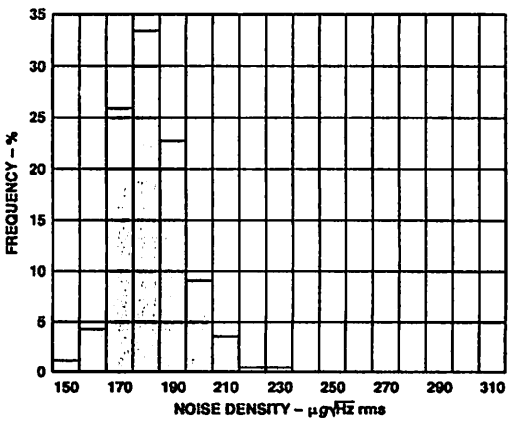
TPC 11. X-Axis Sensitivity at X_{OUT}, V_{DD} = 5 V



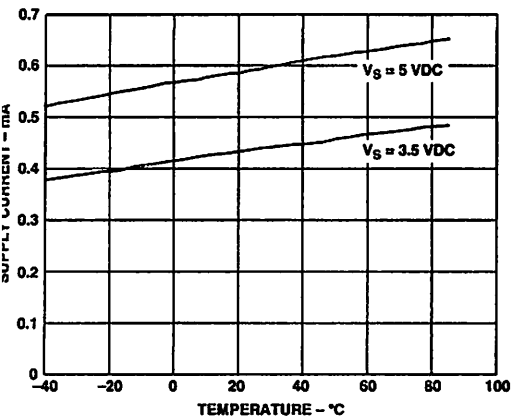
TPC 12. Y-Axis Sensitivity at Y_{OUT}, V_{DD} = 5 V



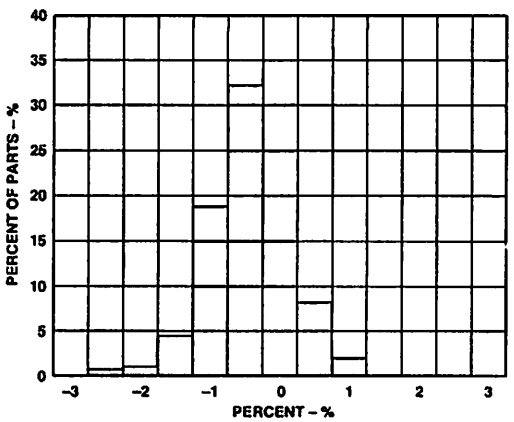
TPC 13. Noise Density Distribution, V_{DD} = 3 V



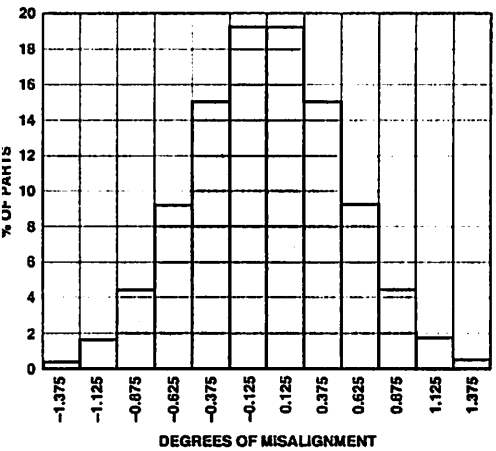
TPC 16. Noise Density Distribution, V_{DD} = 5 V



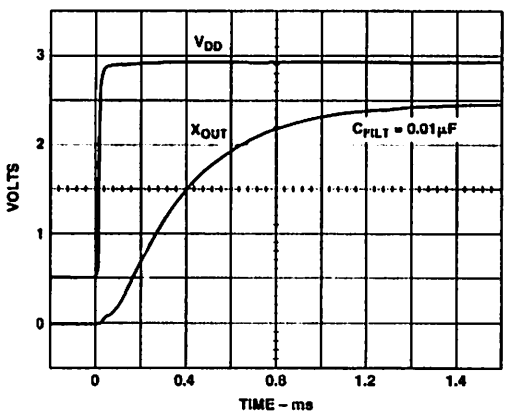
TPC 14. Typical Supply Current vs. Temperature



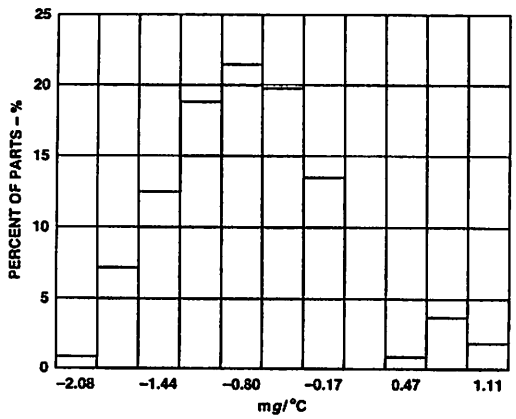
TPC 17. Cross-Axis Sensitivity Distribution



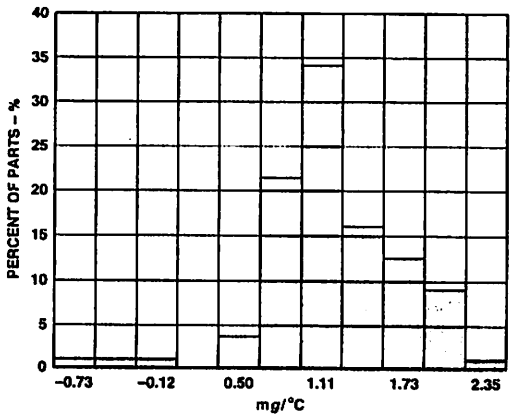
TPC 15. Rotational Die Alignment



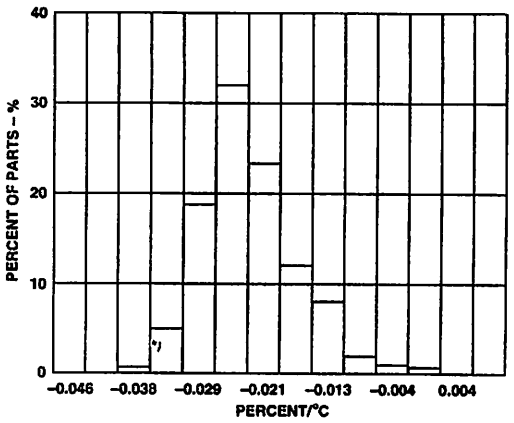
TPC 18. Typical Turn-On Time



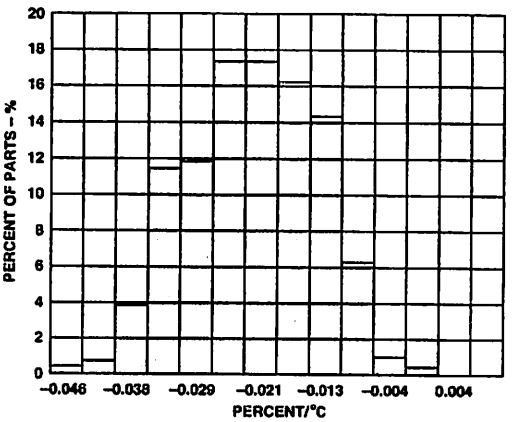
TPC 19. X-Axis Zero g Drift Due to Temperature Distribution, –40°C to +85°C



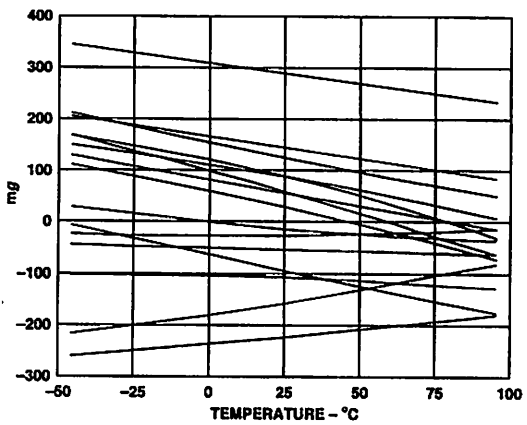
TPC 22. Y-Axis Zero g Drift Due to Temperature Distribution, –40°C to +85°C



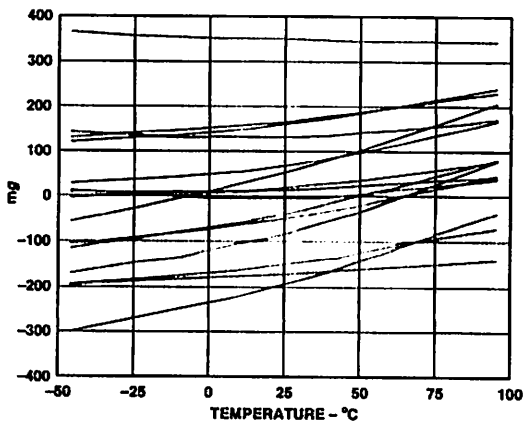
TPC 20. X-Axis Sensitivity Drift at X_{FILT} Due to Temperature Distribution, –40°C to +85°C



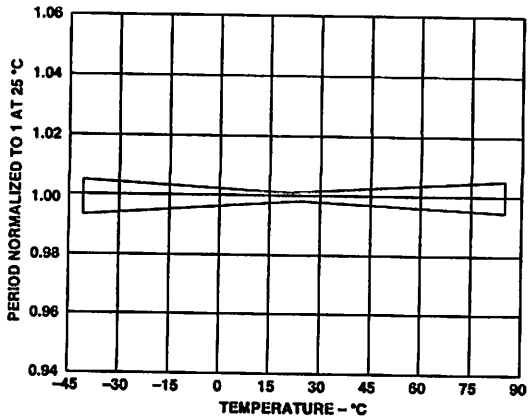
TPC 23. Y-Axis Sensitivity Drift at Y_{FILT} Due to Temperature Distribution, –40°C to +85°C



TPC 21. Typical X-Axis Zero g vs. Output for 16 Parts



TPC 24. Typical Y-Axis Zero g vs. Output for 16 Parts



TPC 25. Normalized DCM Period (T2) vs. Temperature

ITIONS

Length of the "on" portion of the cycle.

Length of the total cycle.

/cle Ratio of the "on" time (T1) of the cycle to the total cycle (T2). Defined as T1/T2 for the ADXL202E/ ADXL210.

ith Time period of the "on" pulse. Defined as T1 for the ADXL202E/ADXL210.

Y OF OPERATION

XL202E is a complete, dual-axis acceleration measurement n a single monolithic IC. It contains a polysilicon surface-chined sensor and signal conditioning circuitry to imple- open loop acceleration measurement architecture. For ;, an output circuit converts the analog signal to a duty dulated (DCM) digital signal that can be decoded with r/timer port on a microprocessor. The ADXL202E is of measuring both positive and negative accelerations to 2 *g*. The accelerometer can measure static acceleration ch as gravity, allowing it to be used as a tilt sensor.

or is a surface micromachined polysilicon structure top of the silicon wafer. Polysilicon springs suspend ure over the surface of the wafer and provide a resistance celeration forces. Deflection of the structure is measured ifferential capacitor that consists of independent fixed d central plates attached to the moving mass. The fixed driven by 180° out of phase square waves. An accelera- leflect the beam and unbalance the differential capacitor, in an output square wave whose amplitude is propor- acceleration. Phase sensitive demodulation techniques are l to rectify the signal and determine the direction of eration.

ut of the demodulator drives a duty cycle modulator tage through a 32 kΩ resistor. At this point a pin is n each channel to allow the user to set the signal band- he device by adding a capacitor. This filtering improves ent resolution and helps prevent aliasing.

g low-pass filtered, the analog signal is converted to a modulated signal by the DCM stage. A single resistor riod for a complete cycle (T2), which can be set between d 10 ms (see Figure 12). A 0 *g* acceleration produces a nominally 50% duty cycle. The acceleration signal can be determined by measuring the length of the T1 and T2 pulses with a counter/timer or with a polling loop using a low cost microcontroller.

An analog output voltage can be obtained either by buffering the signal from the X_{FILT} and Y_{FILT} pin, or by passing the duty cycle signal through an RC filter to reconstruct the dc value.

The ADXL202E will operate with supply voltages as low as 3.0 V or as high as 5.25 V.

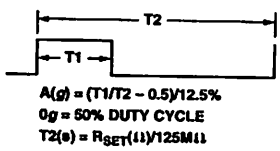


Figure 1. Typical Output Duty Cycle

APPLICATIONS

POWER SUPPLY DECOUPLING

For most applications a single 0.1 µF capacitor, C_{DC}, will adequately decouple the accelerometer from signal and noise on the power supply. However, in some cases, especially where digital devices such as microcontrollers share the same power supply, digital noise on the supply may cause interference on the ADXL202E output. This may be observed as a slowly undulating fluctuation of voltage at X_{FILT} and Y_{FILT}. If additional decoupling is needed, a 100 Ω (or smaller) resistor or ferrite beads, may be inserted in the supply line of the ADXL202E.

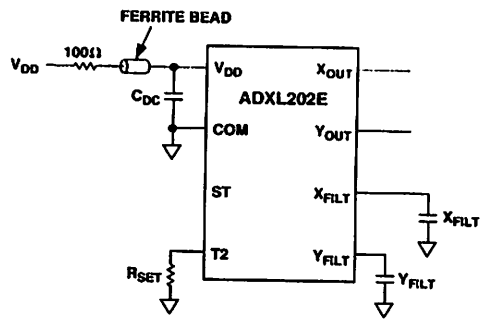


Figure 2.

ESIGN PROCEDURE FOR THE ADXL202E

he design procedure for using the ADXL202E with a duty cycle ıtput involves selecting a duty cycle period and a filter capacitor. proper design will take into account the application requirements r bandwidth, signal resolution and acquisition time, as discussed the following sections.

ecoupling Capacitor C_{DC}

0.1 µF capacitor is recommended from V_{DD} to COM for power ıpply decoupling.

T

he ST pin controls the self-test feature. When this pin is set to DD, an electrostatic force is exerted on the beam of the accelerom-er. The resulting movement of the beam allows the user to test if ıe accelerometer is functional. The typical change in output will : 10% at the duty cycle outputs (corresponding to 800 m*g*). his pin may be left open circuit or connected to common in ɔrmal use.

uty Cycle Decoding

he ADXL202E's digital output is a duty cycle modulator. cceleration is proportional to the ratio T1/T2. The nominal ıtput of the ADXL202E is:

$$0\ g = 50\%\ Duty\ Cycle$$

Scale factor is 12.5% Duty Cycle Change per *g*

hese nominal values are affected by the initial tolerance of the :vice including zero *g* offset error and sensitivity error.

2 does not have to be measured for every measurement cycle. need only be updated to account for changes due to tempera-ıre, (a relatively slow process). Since the T2 time period is shared / both X and Y channels, it is necessary only to measure it on ıe channel of the ADXL202E. Decoding algorithms for various icrocontrollers have been developed. Consult the appropriate pplication Note.

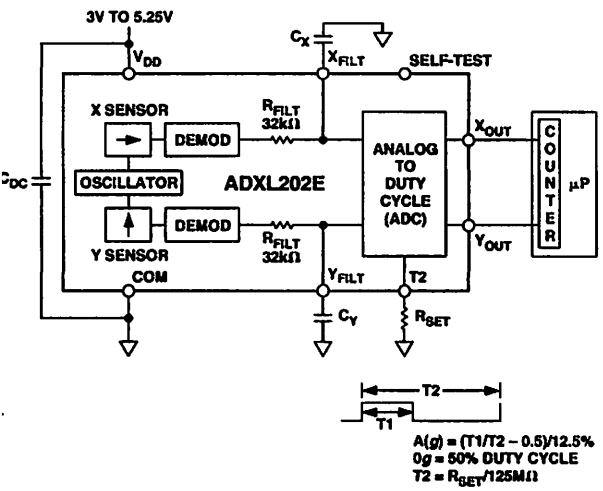


Figure 3. Block Diagram

Setting the Bandwidth Using C_X and C_Y

The ADXL202E has provisions for bandlimiting the X_{FILT} and Y_{FILT} pins. Capacitors must be added at these pins to implement low-pass filtering for antialiasing and noise reduction. The equation for the 3 dB bandwidth is:

$$F_{-3\,dB} = \frac{1}{\left(2\,\pi\,(32\,k\Omega) \times C(x,y)\right)}$$

or, more simply, $F_{-3\,dB} = \frac{5\,\mu F}{C_{(X,Y)}}$

The tolerance of the internal resistor (R_{FILT}), can vary typically as much as ±15% of its nominal value of 32 kΩ; so the bandwidth will vary accordingly. A minimum capacitance of 1000 pF for $C_{(X,Y)}$ is required in all cases.

Table I. Filter Capacitor Selection, C_X and C_Y

Bandwidth	Capacitor Value
10 Hz	0.47 µF
50 Hz	0.10 µF
100 Hz	0.05 µF
200 Hz	0.027 µF
500 Hz	0.01 µF
5 kHz	0.001 µF

Setting the DCM Period with R_{SET}

The period of the DCM output is set for both channels by a single resistor from R_{SET} to ground. The equation for the period is:

$$T2 = \frac{R_{SET}\,(\Omega)}{125\,M\Omega}$$

A 125 kΩ resistor will set the duty cycle repetition rate to approximately 1 kHz, or 1 ms. The device is designed to operate at duty cycle periods between 0.5 ms and 10 ms.

Table II. Resistor Values to Set T2

T2	R_{SET}
1 ms	125 kΩ
2 ms	250 kΩ
5 ms	625 kΩ
10 ms	1.25 MΩ

Note that the R_{SET} should always be included, even if only an analog output is desired. Use an R_{SET} value between 500 kΩ and 2 MΩ when taking the output from X_{FILT} or Y_{FILT}. The R_{SET} resistor should be place close to the T2 Pin to minimize parasitic capacitance at this node.

Selecting the Right Accelerometer

For most tilt sensing applications the ADXL202E is the most appropriate accelerometer. Its higher sensitivity (12.5%/*g*) allows the user to use a lower speed counter for PWM decoding while maintaining high resolution. The ADXL210 should be used in applications where accelerations of greater than ±2 *g* are expected.

)COMPUTER INTERFACES

)XL202E is specifically designed to work with low-cost ıntrollers. Specific code sets, reference designs, and applica- es are available from the factory. This section will outline a design procedure and discuss the various trade-offs that be considered.

ıigner should have some idea of the required performance ystem in terms of:

on: the smallest signal change that needs to be detected.

dth: the highest frequency that needs to be detected.

ion Time: the time that will be available to acquire the signal ı axis.

equirements will help to determine the accelerometer band- he speed of the microcontroller clock and the length of period.

electing a microcontroller it is helpful to have a counter ort available. The microcontroller should have provisions vare calibration. While the ADXL202E is a highly accurate ımeter, it has a wide tolerance for initial offset. The easi- to null this offset is with a calibration factor saved on the ıntroller or by a user calibration for zero *g*. In the case ıe offset is calibrated during manufacture, there are several , including external EEPROM and microcontrollers with ne programmable" features.

N TRADE-OFFS FOR SELECTING FILTER \CTERISTICS: THE NOISE/BW TRADE-OFF

elerometer bandwidth selected will determine the measure- solution (smallest detectable acceleration). Filtering can be lower the noise floor and improve the resolution of the ımeter. Resolution is dependent on both the analog filter ıth at X_{FILT} and Y_{FILT} and on the speed of the micro- er counter.

ılog output of the ADXL202E has a typical bandwidth z, while the duty cycle modulators' bandwidth is 500 Hz. :r must filter the signal at this point to limit aliasing To minimize DCM errors the analog bandwidth should be n 1/10 the DCM frequency. Analog bandwidth may be d to up to 1/2 the DCM frequency in many applications. ll result in greater dynamic error generated at the DCM.

ılog bandwidth may be further decreased to reduce noise ıprove resolution. The ADXL202E noise has the character- : white Gaussian noise that contributes equally at all cies and is described in terms of μ*g* per root Hz; i.e., the proportional to the square root of the bandwidth of the meter. It is recommended that the user limit bandwidth to :st frequency needed by the application, to maximize the ın and dynamic range of the accelerometer.

With the single pole roll-off characteristic, the typical noise of the ADXL202E is determined by the following equation:

$$Noise\,(rms) = \left(200\ \mu g/\sqrt{Hz}\right) \times \left(\sqrt{BW \times 1.6}\right)$$

At 100 Hz the noise will be:

$$Noise\,(rms) = \left(200\ \mu g/\sqrt{Hz}\right) \times \left(\sqrt{100 \times (1.6)}\right) = 2.53\ mg$$

Often the peak value of the noise is desired. Peak-to-peak noise can only be estimated by statistical methods. Table III is useful for estimating the probabilities of exceeding various peak values, given the rms value.

Table III. Estimation of Peak-to-Peak Noise

Nominal Peak-to-Peak Value	% of Time that Noise Will Exceed Nominal Peak-to-Peak Value
2.0 × rms	32%
4.0 × rms	4.6%
6.0 × rms	0.27%
8.0 × rms	0.006%

The peak-to-peak noise value will give the best estimate of the uncertainty in a single measurement.

Table IV gives typical noise output of the ADXL202E for various C_X and C_Y values.

Table IV. Filter Capacitor Selection, C_X and C_Y

Bandwidth	C_X, C_Y	rms Noise	Peak-to-Peak Noise Estimate 95% Probability (rms × 4)
10 Hz	0.47 μF	0.8 m*g*	3.2 m*g*
50 Hz	0.10 μF	1.8 m*g*	7.2 m*g*
100 Hz	0.05 μF	2.5 m*g*	10.1 m*g*
200 Hz	0.027 μF	3.6 m*g*	14.3 m*g*
500 Hz	0.01 μF	5.7 m*g*	22.6 m*g*

CHOOSING T2 AND COUNTER FREQUENCY: DESIGN TRADE-OFFS

The noise level is one determinant of accelerometer resolution. The second relates to the measurement resolution of the counter when decoding the duty cycle output.

The ADXL202E's duty cycle converter has a resolution of approximately 14 bits; better resolution than the accelerometer itself. The actual resolution of the acceleration signal is, however, limited by the time resolution of the counting devices used to decode the duty cycle. The faster the counter clock, the higher the resolution of the duty cycle and the shorter the T2 period can be for a given resolution. The following table shows some of the trade-offs. It is important to note that this is the resolution due to the microprocessors' counter. It is probable that the accelerometer's noise floor may set the lower limit on the resolution, as discussed in the previous section.

able V. Trade-Offs Between Microcontroller Counter Rate, 2 Period, and Resolution of Duty Cycle Modulator

2 (ms)	R_{SET} (kΩ)	ADXL202E Sample Rate	Counter-Clock Rate (MHz)	Counts per T2 Cycle	Counts per *g*	Resolution (mg)
0	124	1000	2.0	2000	250	4.0
0	124	1000	1.0	1000	125	8.0
0	124	1000	0.5	500	62.5	16.0
0	625	200	2.0	10000	1250	0.8
0	625	200	1.0	5000	625	1.6
0	625	200	0.5	2500	312.5	3.2
).0	1250	100	2.0	20000	2500	0.4
).0	1250	100	1.0	10000	1250	0.8
).0	1250	100	0.5	5000	625	1.6

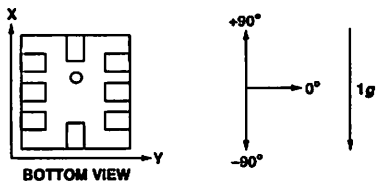
TRATEGIES FOR USING THE DUTY CYCLE OUTPUT 'ITH MICROCONTROLLERS

pplication notes outlining various strategies for using the duty 'cle output with low cost microcontrollers are available from .e factory.

SING THE ADXL202E AS A DUAL-AXIS TILT SENSOR

ne of the most popular applications of the ADXL202E is tilt easurement. An accelerometer uses the force of gravity as an put vector to determine orientation of an object in space.

n accelerometer is most sensitive to tilt when its sensitive axis perpendicular to the force of gravity, i.e., parallel to the earth's rface. At this orientation its sensitivity to changes in tilt is high-t. When the accelerometer is oriented on axis to gravity, i.e., ar its +1 *g* or −1 *g* reading, the change in output acceleration r degree of tilt is negligible. When the accelerometer is perpen-cular to gravity, its output will change nearly 17.5 mg per degree tilt, but at 45° degrees it is changing only at 12.2 mg per gree and resolution declines. The following table illustrates e changes in the X and Y axes as the device is tilted ±90° rough gravity.



Axis rientation Horizon (°)	X Output (g)	Δ per Degree of Tilt (mg)	Y Output (g)	Δ per Degree of Tilt (mg)
−90	−1.000	−0.2	0.000	17.5
−75	−0.966	4.4	0.259	16.9
−60	−0.866	8.6	0.500	15.2
−45	−0.707	12.2	0.707	12.4
−30	−0.500	15.0	0.866	8.9
−15	−0.259	16.8	0.966	4.7
0	0.000	17.5	1.000	0.2
15	0.259	16.9	0.966	−4.4
30	0.500	15.2	0.866	−8.6
45	0.707	12.4	0.707	−12.2
60	0.866	8.9	0.500	−15.0
75	0.966	4.7	0.259	−16.8
90	1.000	0.2	0.000	−17.5

Figure 4. How the X and Y Axes Respond to Changes in Tilt

A DUAL AXIS TILT SENSOR: CONVERTING ACCELERATION TO TILT

When the accelerometer is oriented so both its X and Y axes are parallel to the earth's surface it can be used as a two axis tilt sensor with a roll and a pitch axis. Once the output signal from the accelerometer has been converted to an acceleration that varies between −1 *g* and +1 *g*, the output tilt in degrees is calculated as follows:

$$Pitch = ASIN\ (Ax/1\ g)$$
$$Roll = ASIN\ (Ay/1\ g)$$

Be sure to account for overranges. It is possible for the accelerometers to output a signal greater than ±1 *g* due to vibration, shock or other accelerations.

MEASURING 360° OF TILT

It is possible to measure a full 360° of orientation through gravity by using two accelerometers oriented perpendicular to one another (see Figure 5). When one sensor is reading a maximum change in output per degree, the other is at its minimum.

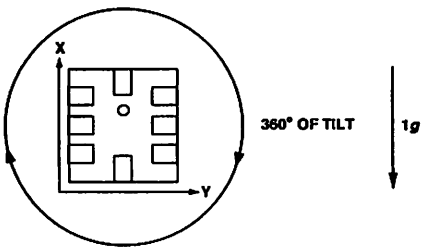


Figure 5. Using a Two-Axis Accelerometer to Measure 360° of Tilt

USING THE ANALOG OUTPUT

The ADXL202E was specifically designed for use with its digital outputs, but has provisions to provide analog outputs as well.

Duty Cycle Filtering

An analog output can be reconstructed by filtering the duty cycle output. This technique requires only passive components. The duty cycle period (T2) should be set to <1 ms. An RC filter with a 3 dB point at least a factor of >10 less than the duty cycle frequency is connected to the duty cycle output. The filter resistor should be no less than 100 kΩ to prevent loading of the output stage. The analog output signal will be ratiometric to the supply voltage. The advantage of this method is an output scale factor of approximately double the analog output. Its disadvantage is that the frequency response will be lower than when using the X_{FILT}, Y_{FILT} output.

X_{FILT}, Y_{FILT} Output

The second method is to use the analog output present at the X_{FILT} and Y_{FILT} pin. Unfortunately, these pins have a 32 kΩ output impedance and are not designed to drive a load directly. An op amp follower may be required to buffer this pin. The advantage of this method is that the full 5 kHz bandwidth of the accelerometer is available to the user. A capacitor still must be added at this point for filtering. The duty cycle converter should be kept running by using R_{SET} <10 MΩ. Note that the accelerometer offset and sensitivity are ratiometric to the supply voltage. The offset and sensitivity are nominally:

0 *g* Offset = $V_{DD}/2$

ADXL202E Sensitivity = $(60\ mV \times V_S)/g$

THE ADXL202E IN VERY LOW POWER ATIONS

cation note outlining low power strategies for the)2E is available. Some key points are presented here. ible to reduce the ADXL202E's average current from o less than 20 µA by using the following techniques:

r Cycle the accelerometer.

he accelerometer at a Lower Voltage, (Down to 3 V).

ycling with an External A/D

ng on the value of the X_{FILT} capacitor, the ADXL202E : of turning on and giving a good reading in 1.6 ms. Most ıtroller based A/Ds can acquire a reading in another 25 µs. s possible to turn on the ADXL202E and take a reading ;. If we assume that a 20 Hz sample rate is sufficient, current required to take 20 samples is 2 ms × 20 samples/s \ = 24 µA average current. Running the part at 3 V will ıe supply current from 0.6 mA to 0.4 mA, bringing the :urrent down to 16 µA.

) should read the analog output of the ADXL202E at r and Y_{FILT} pins. A buffer amplifier is recommended, and equired in any case to amplify the analog output to give :esolution with an 8-bit to 10-bit converter.

ycling When Using the Digital Output

ıative is to run the microcontroller at a higher clock rate it into shutdown between readings, allowing the use of the utput. In this approach the ADXL202E should be set at t sample rate (T2 = 0.5 ms), with a 500 Hz filter at X_{FILT} .T. The concept is to acquire a reading as quickly as pos- . then shut down the ADXL202E and the microcontroller : next sample is needed.

of the above approaches, the ADXL202E can be turned ff directly using a digital port pin on the microcontroller to ıe accelerometer without additional components.

CALIBRATING THE ADXL202E/ADXL210

The initial value of the offset and scale factor for the ADXL202E will require calibration for applications such as tilt measurement. The ADXL202E architecture has been designed so that these calibrations take place in the software of the microcontroller used to decode the duty cycle signal. Calibration factors can be stored in EEPROM or determined at turn-on and saved in dynamic memory.

For low *g* applications, the force of gravity is the most stable, accurate and convenient acceleration reference available. A reading of the 0 *g* point can be determined by orientating the device parallel to the earth's surface and then reading the output.

A more accurate calibration method is to make measurements at +1 *g* and −1 *g*. The sensitivity can be determined by the two measurements.

To calibrate, the accelerometer's measurement axis is pointed directly at the earth. The 1 *g* reading is saved and the sensor is turned 180° to measure −1 *g*. Using the two readings, the sensitivity is:

Let A = Accelerometer output with axis oriented to +1 *g*
Let B = Accelerometer output with axis oriented to −1 *g then:*
$$Sensitivity = [A - B]/2\ g$$

For example, if the +1 *g* reading (A) is 55% duty cycle and the −1 *g* reading (B) is 32% duty cycle, then:

$$Sensitivity = [55\% - 32\%]/2\ g = 11.5\%/g$$

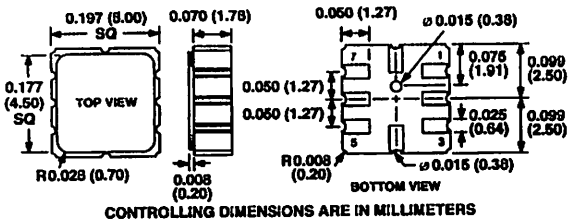
These equations apply whether the output is analog or duty cycle.

Application notes outlining algorithms for calculating acceleration from duty cycle and automated calibration routines are available from the factory.

OUTLINE DIMENSIONS

Dimensions shown in inches and (mm).

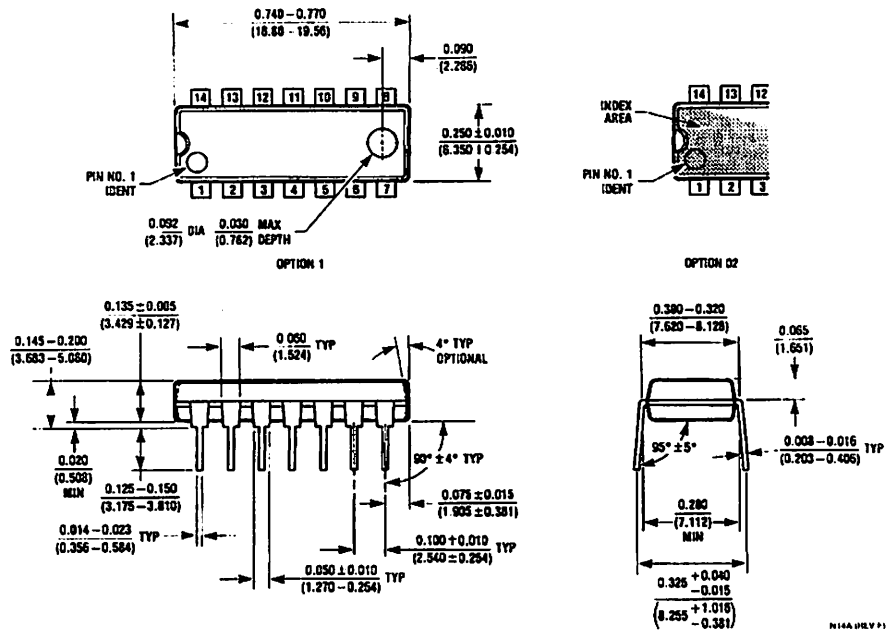
8-Terminal Ceramic Leadless Chip Carrier (E-8)



C02064–2.5–10/00 (rev. A)

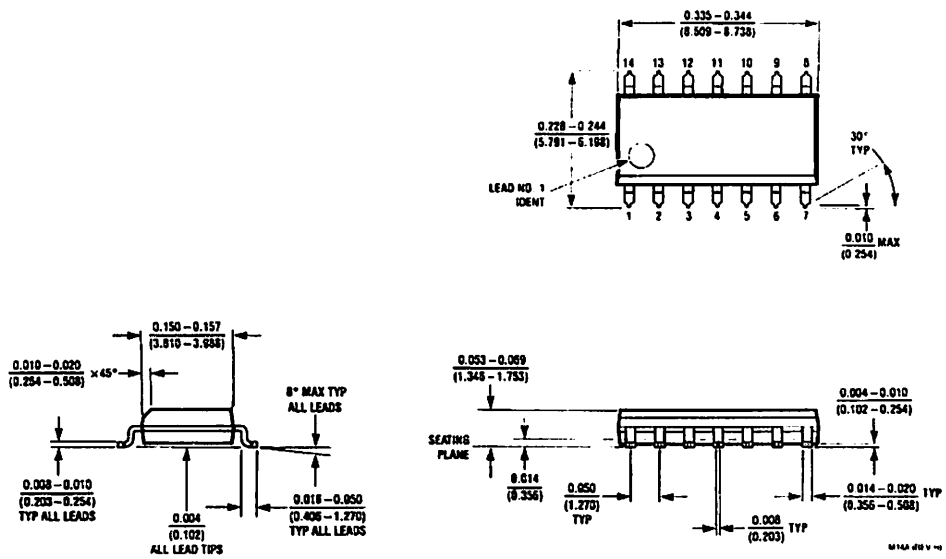
PRINTED IN U.S.A.

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



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

Physical Dimensions inches (millimeters) unless otherwise noted



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

DATA SHEET

74LS164

FAIRCHILD SEMICONDUCTOR™

August 1986
Revised April 2000

DM74LS164
8-Bit Serial In/Parallel Out Shift Register

General Description

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

Features

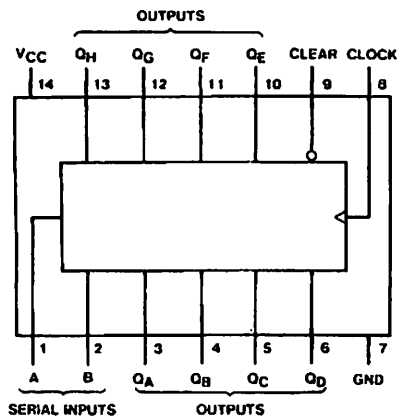
- Gated (enable/disable) serial inputs
- Fully buffered clock and serial inputs
- Asynchronous clear
- Typical clock frequency 36 MHz
- Typical power dissipation 80 mW

Ordering Code:

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

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

Connection Diagram

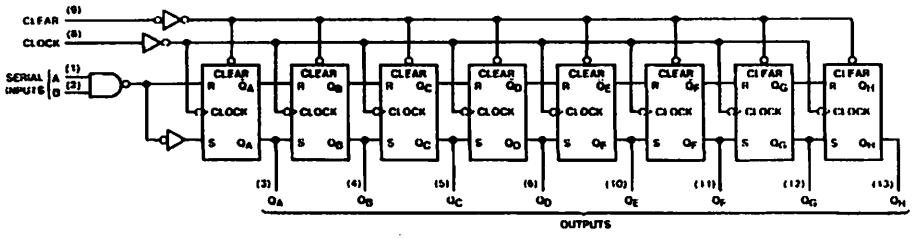


Function Table

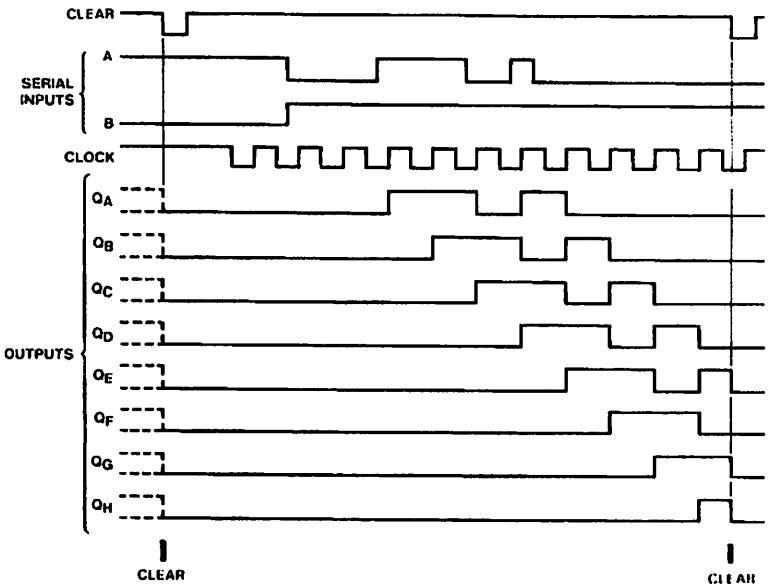
Inputs				Outputs			
Clear	Clock	A	B	Q_A	Q_B	...	Q_H
L	X	X	X	L	L	...	L
H	L	X	X	Q_{A0}	Q_{B0}	...	Q_{H0}
H	↑	H	H	H	Q_{An}	...	Q_{Gn}
H	↑	L	X	L	Q_{An}	...	Q_{Gn}
H	↑	X	L	L	Q_{An}	...	Q_{Gn}

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

Logic Diagram



Timing Diagram



Absolute Maximum Ratings(Note 1)

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

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

Recommended Operating Conditions

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

Note 2: $T_A = 25°C$ and $V_{CC} = 5V$.

Electrical Characteristics

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

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

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

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

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

Switching Characteristics

at $V_{CC} = 5V$ and $T_A = 25°C$

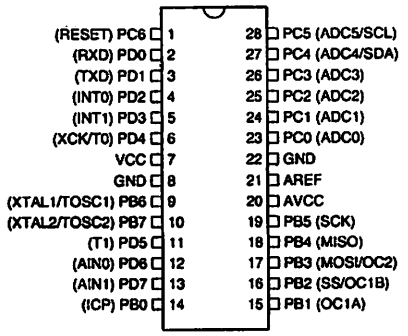
Symbol	Parameter	From (Input) To (Output)	R_L = 2 kΩ				Units
			C_L = 15 pF		C_L = 50 pF		
			Min	Max	Min	Max	
f_{MAX}	Maximum Clock Frequency		25				MHz
t_{PLH}	Propagation Delay Time LOW-to-HIGH Level Output	Clock to Output		27		30	ns
t_{PHL}	Propagation Delay Time HIGH-to-LOW Level Output	Clock to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

DATA SHEET

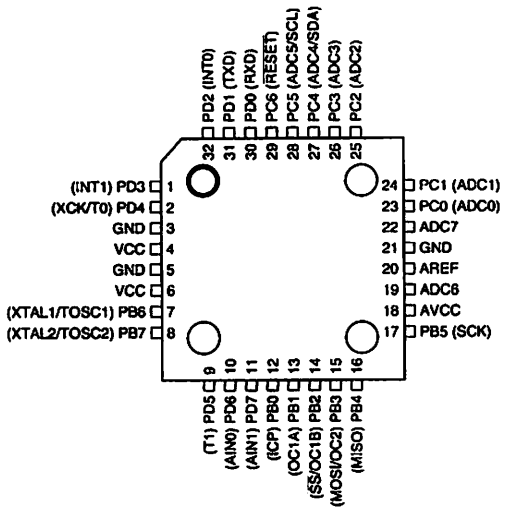
ATMEGA8

onfigurations

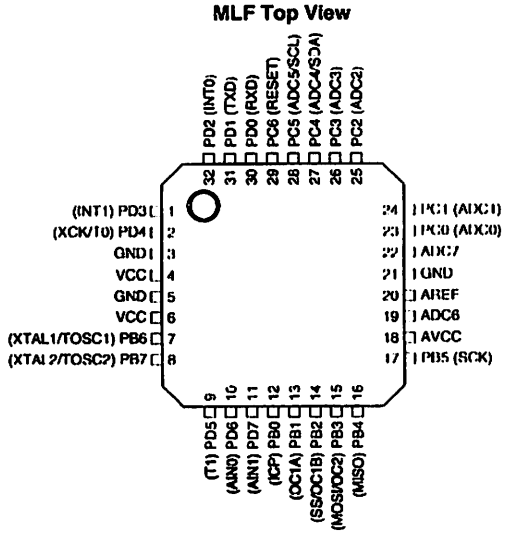
PDIP



TQFP Top View



MLF Top View

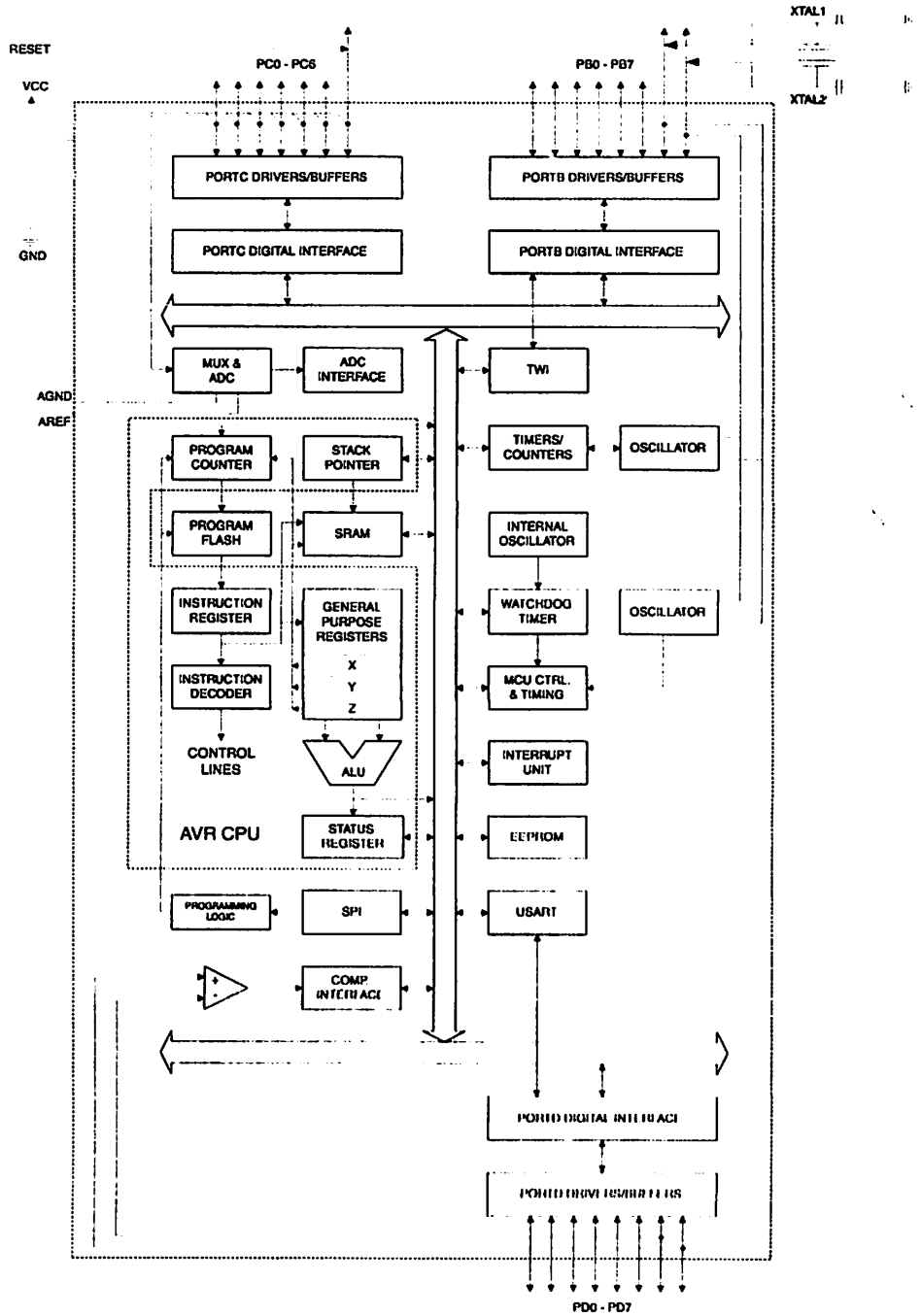


ɪrview

The ATmega8 is a low-power CMOS 8-bit microcontroller based on the AVR RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega8 achieves throughputs approaching 1 MIPS per MHz, allowing the system designer to optimize power consumption versus processing speed.

ɛk Diagram

Figure 1. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All the 32 registers are directly connected to the Arithmetic Logic Unit (ALU), allowing two independent registers to be accessed in one single instruction executed in one clock cycle. The resulting architecture is more code efficient while achieving throughputs up to ten times faster than conventional CISC microcontrollers.

The ATmega8 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes of EEPROM, 1K byte of SRAM, 23 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, a 6-channel ADC (8 channels in TQFP and MLF packages) where 4 (6) channels have 10-bit accuracy and 2 channels have 8-bit accuracy, a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and five software selectable power saving modes. The Idle mode stops the CPU while allowing the SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or hardware reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption.

The device is manufactured using Atmel's high density nonvolatile memory technology. The Flash program memory can be reprogrammed In-System through an SPI serial interface, by a conventional nonvolatile memory programmer, or by an On-chip boot program running on the AVR core. The boot program can use any interface to download the application program in the Application Flash Memory. Software in the Boot Flash Section will continue to run while the Application Flash Section is updated, providing true Read-While-Write operation. By combining an 8-bit RISC CPU with In-System Self-Programmable Flash on a monolithic chip, the Atmel ATmega8 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega8 AVR is supported with a full suite of program and system development tools, including C compilers, macro assemblers, program debugger/simulators, In-circuit emulators, and evaluation kits.

escriptions

Digital supply voltage.

Ground.

(PB7..PB0)/XTAL1/ /TOSC1/TOSC2

Port B is an 8-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port B output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port B pins that are externally pulled low will source current if the pull-up resistors are activated. The Port B pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Depending on the clock selection fuse settings, PB6 can be used as input to the inverting Oscillator amplifier and input to the internal clock operating circuit.

Depending on the clock selection fuse settings, PB7 can be used as output from the inverting oscillator amplifier.

If the Internal Calibrated RC Oscillator is used as chip clock source, PB7..6 is used as TOSC2..1 input for the Asynchronous Timer/Counter2 if the AS2 bit in ASSR is set.

The various special features of Port B are elaborated on page 54.

C (PC5..PC0)

Port C is an 7-bit bi-directional I/O port with internal pull-up resistors (selected for each bit). The Port C output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port C pins that are externally pulled low will source current if the pull-up resistors are activated. The Port C pins are tri-stated when a reset condition becomes active, even if the clock is not running.

RESET

If the RSTDISBL fuse is programmed, PC6 is used as an I/O pin. Note that the electrical characteristics of PC6 differ from those of the other pins of Port C.

If the RSTDISBL fuse is unprogrammed, PC6 is used as a Reset input. A low level on this pin for longer than the minimum pulse length will generate a Reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 34. Shorter pulses are not guaranteed to generate a Reset.

The various special features of Port C are elaborated on page 57.

D (PD7..PD0)

Port D is an 8-bit bidirectional I/O port with internal pull-up resistors (selected for each bit). The Port D output buffers have symmetrical drive characteristics with both high sink and source capability. As inputs, Port D pins that are externally pulled low will source current if the pull-up resistors are activated. The Port D pins are tri-stated when a reset condition becomes active, even if the clock is not running.

Port D also serves the functions of various special features of the ATmega8 as listed on page 59.

ET

Reset input. A low level on this pin for longer than the minimum pulse length will generate a reset, even if the clock is not running. The minimum pulse length is given in Table 15 on page 34. Shorter pulses are not guaranteed to generate a reset.

.1

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

.2

Output from the inverting oscillator amplifier.

:

AVCC is the supply voltage pin for the A/D Converter, Port C (3..0), and ADC (7..6). It should be externally connected to V_{CC}, even if the ADC is not used. If the ADC is used, it should be connected to V_{CC} through a low-pass filter. Note that Port C (5..4) use digital supply voltage, V_{CC}.

=

AREF is the analog reference pin for the A/D Converter.

7..6 (TQFP and MLF age Only)

In the TQFP and MLF package, ADC7..6 serve as analog inputs to the A/D converter. These pins are powered from the analog supply and serve as 10-bit ADC channels.

ıut Code mples

This datasheet contains simple code examples that briefly show how to use various parts of the device. These code examples assume that the part specific header file is included before compilation. Be aware that not all C compiler vendors include bit definitions in the header files and interrupt handling in C is compiler dependent. Please confirm with the C compiler documentation for more details.

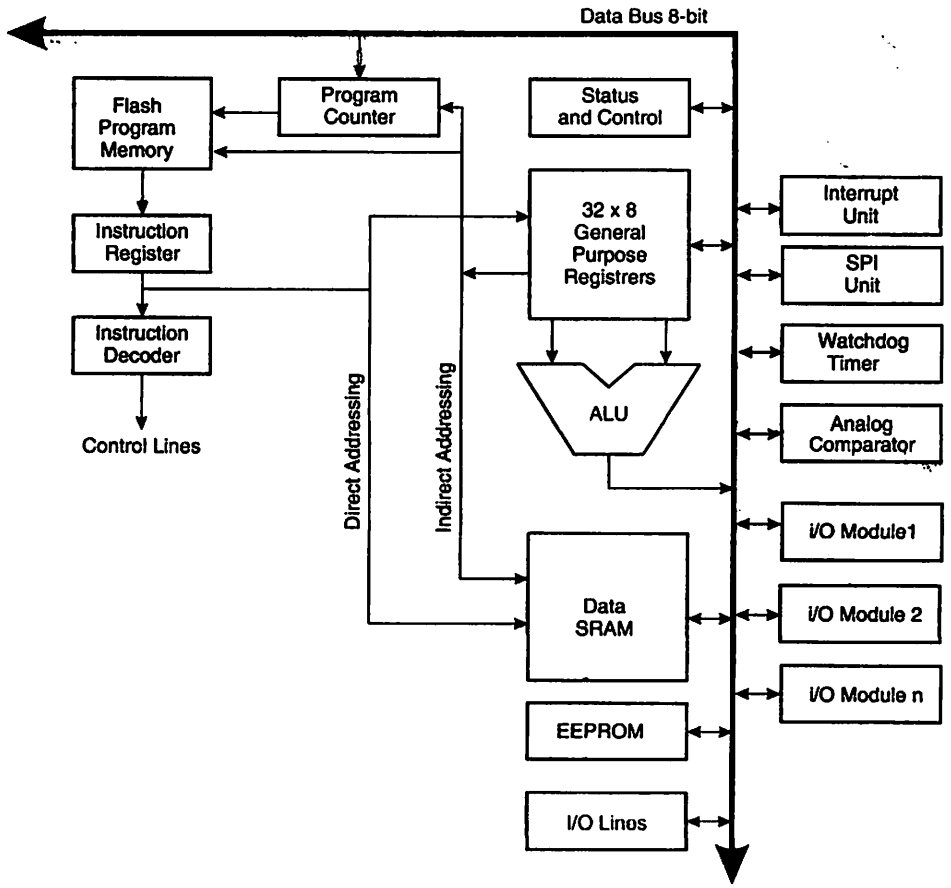
ʀ CPU Core

ɔduction

This section discusses the AVR core architecture in general. The main function of the CPU core is to ensure correct program execution. The CPU must therefore be able to access memories, perform calculations, control peripherals, and handle interrupts.

ıitectural Overview

Figure 2. Block Diagram of the AVR MCU Architecture



In order to maximize performance and parallelism, the AVR uses a Harvard architecture – with separate memories and buses for program and data. Instructions in the program memory are executed with a single level pipelining. While one instruction is being executed, the next instruction is pre-fetched from the program memory. This concept enables instructions to be executed in every clock cycle. The program memory is In-System Reprogrammable Flash memory.

The fast-access Register file contains 32 x 8-bit general purpose working registers with a single clock cycle access time. This allows single-cycle Arithmetic Logic Unit (ALU) operation. In a typical ALU operation, two operands are output from the Register file, the operation is executed, and the result is stored back in the Register file – in one clock cycle.

Six of the 32 registers can be used as three 16-bit indirect address register pointers for Data Space addressing – enabling efficient address calculations. One of the these address pointers can also be used as an address pointer for look up tables in Flash program memory. These added function registers are the 16-bit X-, Y-, and Z-register, described later in this section.

The ALU supports arithmetic and logic operations between registers or between a constant and a register. Single register operations can also be executed in the ALU. After an arithmetic operation, the Status Register is updated to reflect information about the result of the operation.

The Program flow is provided by conditional and unconditional jump and call instructions, able to directly address the whole address space. Most AVR instructions have a single 16-bit word format. Every program memory address contains a 16- or 32-bit instruction.

Program Flash memory space is divided in two sections, the Boot program section and the Application program section. Both sections have dedicated Lock Bits for write and read/write protection. The SPM instruction that writes into the Application Flash memory section must reside in the Boot program section.

During interrupts and subroutine calls, the return address program counter (PC) is stored on the stack. The stack is effectively allocated in the general data SRAM, and consequently the stack size is only limited by the total SRAM size and the usage of the SRAM. All user programs must initialize the SP in the reset routine (before subroutines or interrupts are executed). The stack pointer SP is read/write accessible in the I/O space. The data SRAM can easily be accessed through the five different addressing modes supported in the AVR architecture.

The memory spaces in the AVR architecture are all linear and regular memory maps.

A flexible interrupt module has its control registers in the I/O space with an additional global interrupt enable bit in the status register. All interrupts have a separate interrupt vector in the interrupt vector table. The interrupts have priority in accordance with their interrupt vector position. The lower the interrupt vector address, the higher the priority.

The I/O memory space contains 64 addresses for CPU peripheral functions as Control Registers, SPI, and other I/O functions. The I/O Memory can be accessed directly, or as the Data Space locations following those of the Register file, 0x20 - 0x5F.

hmetic Logic Unit – l

The high-performance AVR ALU operates in direct connection with all the 32 general purpose working registers. Within a single clock cycle, arithmetic operations between general purpose registers or between a register and an immediate are executed. The ALU operations are divided into three main categories – arithmetic, logical, and bit-functions. Some implementations of the architecture also provide a powerful multiplier supporting both signed/unsigned multiplication and fractional format. See the "Instruction Set" section for a detailed description.

us Register

The Status Register contains information about the result of the most recently executed arithmetic instruction. This information can be used for altering program flow in order to perform conditional operations. Note that the status register is updated after all ALU operations, as specified in the Instruction Set Reference. This will in many cases remove the need for using the dedicated compare instructions, resulting in faster and more compact code.

The status register is not automatically stored when entering an interrupt routine and restored when returning from an interrupt. This must be handled by software.

The AVR status register – SREG – is defined as:

Bit	7	6	5	4	3	2	1	0	
	I	T	H	S	V	N	Z	C	SREG
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – I: Global Interrupt Enable**

The global interrupt enable bit must be set for the interrupts to be enabled. The individual interrupt enable control is then performed in separate control registers. If the global interrupt enable register is cleared, none of the interrupts are enabled independent of the individual interrupt enable settings. The I-bit is cleared by hardware after an interrupt has occurred, and is set by the RETI instruction to enable subsequent interrupts. The I-bit can also be set and cleared by the application with the SEI and CLI instructions, as described in the Instruction Set Reference.

- **Bit 6 – T: Bit Copy Storage**

The bit copy instructions BLD (Bit LoaD) and BST (Bit STore) use the T bit as source or destination for the operated bit. A bit from a register in the Register file can be copied into T by the BST instruction, and a bit in T can be copied into a bit in a register in the Register file by the BLD instruction.

- **Bit 5 – H: Half Carry Flag**

The half carry flag H indicates a half carry in some arithmetic operations. Half carry is useful in BCD arithmetic. See the "Instruction Set Description" for detailed information.

- **Bit 4 – S: Sign Bit, S = N ⊕ V**

The S-bit is always an exclusive or between the negative flag N and the two's complement overflow flag V. See the "Instruction Set Description" for detailed information.

- **Bit 3 – V: Two's Complement Overflow Flag**

The two's complement overflow flag V supports two's complement arithmetics. See the "Instruction Set Description" for detailed information.

- **Bit 2 – N: Negative Flag**

The negative flag N indicates a negative result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

- **Bit 1 – Z: Zero Flag**

The zero flag Z indicates a zero result in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

- **Bit 0 – C: Carry Flag**

The carry flag C indicates a carry in an arithmetic or logic operation. See the "Instruction Set Description" for detailed information.

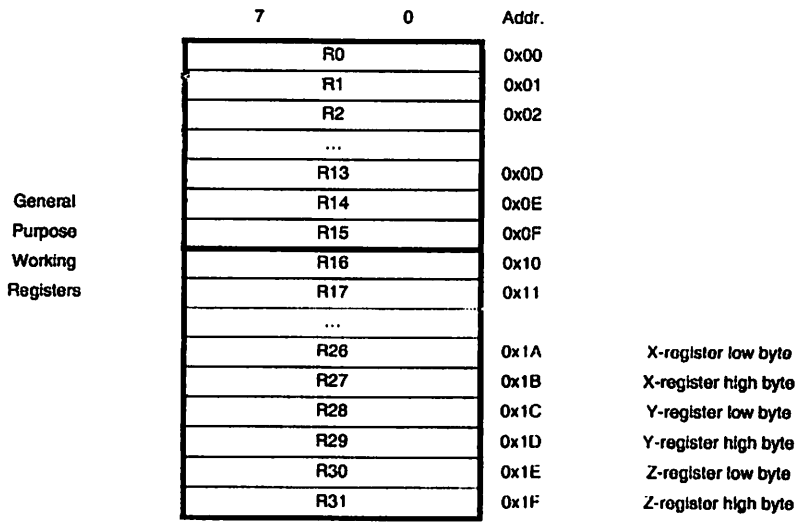
al Purpose ter File

The Register file is optimized for the AVR Enhanced RISC instruction set. In order to achieve the required performance and flexibility, the following input/output schemes are supported by the Register file:

- One 8-bit output operand and one 8-bit result input.
- Two 8-bit output operands and one 8-bit result input.
- Two 8-bit output operands and one 16-bit result input.
- One 16-bit output operand and one 16-bit result input.

Figure 3 shows the structure of the 32 general purpose working registers in the CPU.

Figure 3. AVR CPU General Purpose Working Registers



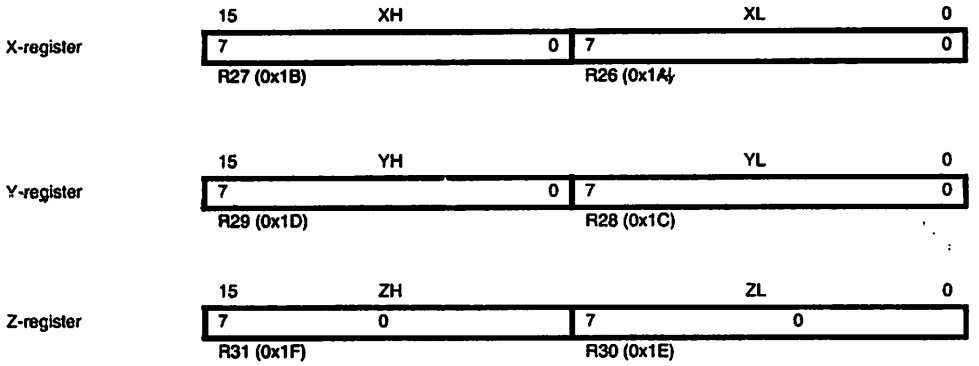
Most of the instructions operating on the Register file have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 3, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-pointer registers can be set to index any register in the file.

X-register, Y-register and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the Data Space. The three indirect address registers X, Y and Z are defined as described in Figure 4.

Figure 4. The X-, Y- and Z-Registers



In the different addressing modes these address registers have functions as fixed displacement, automatic increment, and automatic decrement (see the Instruction Set Reference for details).

Stack Pointer

The stack is mainly used for storing temporary data, for storing local variables and for storing return addresses after interrupts and subroutine calls. The stack pointer register always points to the top of the stack. Note that the stack is implemented as growing from higher memory locations to lower memory locations. This implies that a stack PUSH command decreases the stack pointer.

The Stack Pointer points to the data SRAM stack area where the Subroutine and Interrupt Stacks are located. This Stack space in the data SRAM must be defined by the program before any subroutine calls are executed or interrupts are enabled. The Stack Pointer must be set to point above 0x60. The Stack Pointer is decremented by one when data is pushed onto the Stack with the PUSH instruction, and it is decremented by two when the return address is pushed onto the Stack with subroutine call or interrupt. The Stack Pointer is incremented by one when data is popped from the Stack with the POP instruction, and it is incremented by two when address is popped from the Stack with return from subroutine RET or return from interrupt RETI.

The AVR Stack Pointer is implemented as two 8-bit registers in the I/O space. The number of bits actually used is implementation dependent. Note that the data space in some implementations of the AVR architecture is so small that only SPL is needed. In this case, the SPH register will not be present.

Bit	15	14	13	12	11	10	9	8	
	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SPH
	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0	SPL
	7	6	5	4	3	2	1	0	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

ruction Execution ing

This section describes the general access timing concepts for instruction execution. The AVR CPU is driven by the CPU clock clk_{CPU}, directly generated from the selected clock source for the chip. No internal clock division is used.

Figure 5 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access Register file concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 5. The Parallel Instruction Fetches and Instruction Executions

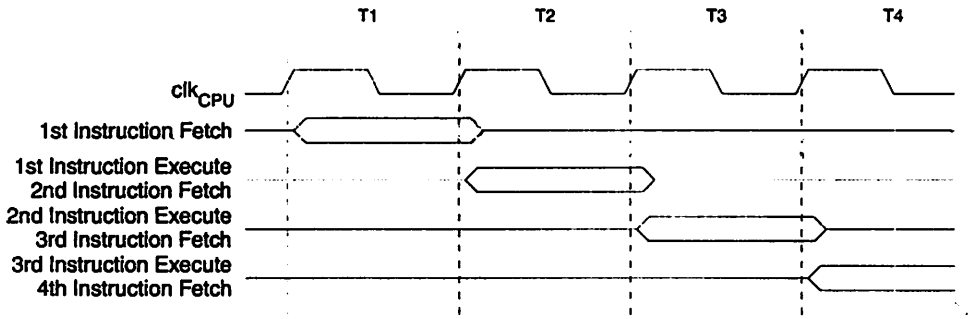
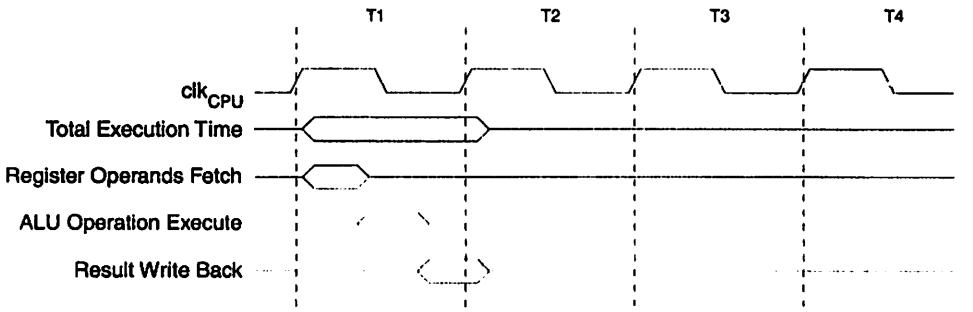


Figure 6 shows the internal timing concept for the Register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 6. Single Cycle ALU Operation



et and Interrupt dling

The AVR provides several different interrupt sources. These interrupts and the separate reset vector each have a separate program vector in the program memory space. All interrupts are assigned individual enable bits which must be written logic one together with the Global Interrupt Enable bit in the Status Register in order to enable the interrupt. Depending on the program counter value, interrupts may be automatically disabled when Boot Lock bits BLB02 or BLB12 are programmed. This feature improves software security. See the section "Memory Programming" on page 214 for details.

The lowest addresses in the program memory space are by default defined as the Reset and Interrupt vectors. The complete list of vectors is shown in "Interrupts" on page 42. The list also determines the priority levels of the different interrupts. The lower the address the higher is the priority level. RESET has the highest priority, and next is INT0 – the External Interrupt Request 0. The interrupt vectors can be moved to the start of the

boot Flash section by setting the Interrupt Vector Select (IVSEL) bit in the General Interrupt Control Register (GICR). Refer to "Interrupts" on page 42 for more information. The Reset vector can also be moved to the start of the boot Flash section by programming the BOOTRST fuse, see "Boot Loader Support – Read-While-Write Self-Programming" on page 202.

When an interrupt occurs, the Global Interrupt Enable I-bit is cleared and all interrupts are disabled. The user software can write logic one to the I-bit to enable nested interrupts. All enabled interrupts can then interrupt the current interrupt routine. The I-bit is automatically set when a Return from Interrupt instruction – RETI – is executed.

There are basically two types of interrupts. The first type is triggered by an event that sets the interrupt flag. For these interrupts, the Program Counter is vectored to the actual interrupt vector in order to execute the interrupt handling routine, and hardware clears the corresponding interrupt flag. Interrupt flags can also be cleared by writing a logic one to the flag bit position(s) to be cleared. If an interrupt condition occurs while the corresponding interrupt enable bit is cleared, the interrupt flag will be set and remembered until the interrupt is enabled, or the flag is cleared by software. Similarly, if one or more interrupt conditions occur while the global interrupt enable bit is cleared, the corresponding interrupt flag(s) will be set and remembered until the global interrupt enable bit is set, and will then be executed by order of priority.

The second type of interrupts will trigger as long as the interrupt condition is present. These interrupts do not necessarily have interrupt flags. If the interrupt condition disappears before the interrupt is enabled, the interrupt will not be triggered.

When the AVR exits from an interrupt, it will always return to the main program and execute one more instruction before any pending interrupt is served.

Note that the status register is not automatically stored when entering an interrupt routine, nor restored when returning from an interrupt routine. This must be handled by software.

When using the CLI instruction to disable interrupts, the interrupts will be immediately disabled. No interrupt will be executed after the CLI instruction, even if it occurs simultaneously with the CLI instruction. The following example shows how this can be used to avoid interrupts during the timed EEPROM write sequence.

Assembly Code Example

```
in  r16, SREG       ; store SREG value
cli     ; disable interrupts during timed sequence
sbi EECR, EEMWE     ; start EEPROM write
sbi EECR, EEWE
out SREG, r16       ; restore SREG value (I-bit)
```

C Code Example

```
char cSREG;
cSREG = SREG;  /* store SREG value */
/* disable interrupts during timed sequence */
_CLI();
EECR |= (1<<EEMWE); /* start EEPROM write */
EECR |= (1<<EEWE);
SREG = cSREG; /* restore SREG value (I-bit) */
```

When using the SEI instruction to enable interrupts, the instruction following SEI will be executed before any pending interrupts, as shown in the following example.

Assembly Code Example

```
sei   ; set global interrupt enable
sleep ; enter sleep, waiting for interrupt
; note: will enter sleep before any pending
; interrupt(s)
```

C Code Example

```
_SEI(); /* set global interrupt enable */
_SLEEP(); /* enter sleep, waiting for interrupt */
/* note: will enter sleep before any pending interrupt(s) */
```

pt Response Time

The interrupt execution response for all the enabled AVR interrupts is four clock cycles minimum. After four clock cycles, the program vector address for the actual interrupt handling routine is executed. During this 4-clock cycle period, the Program Counter is pushed onto the Stack. The vector is normally a jump to the interrupt routine, and this jump takes three clock cycles. If an interrupt occurs during execution of a multi-cycle instruction, this instruction is completed before the interrupt is served. If an interrupt occurs when the MCU is in sleep mode, the interrupt execution response time is increased by four clock cycles. This increase comes in addition to the start-up time from the selected sleep mode.

A return from an interrupt handling routine takes four clock cycles. During these four clock cycles, the Program Counter (2 bytes) is popped back from the Stack, the Stack Pointer is incremented by 2, and the I-bit in SREG is set.

? ATmega8 nories

This section describes the different memories in the ATmega8. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the ATmega8 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

ystem rogrammable Flash ;ram Memory

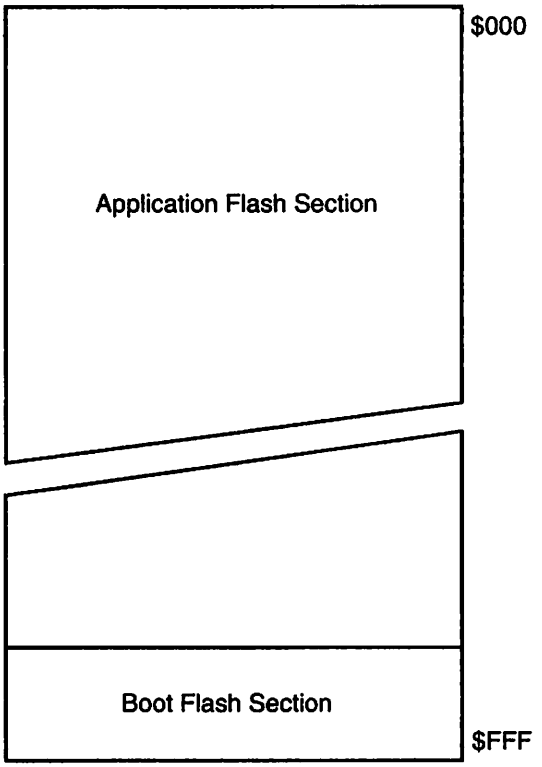
The ATmega8 contains 8K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16- or 32-bits wide, the Flash is organized as 4K x 16 bits. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

The Flash memory has an endurance of at least 1,000 write/erase cycles. The ATmega8 Program Counter (PC) is 12 bits wide, thus addressing the 4K program memory locations. The operation of Boot Program section and associated Boot Lock Bits for software protection are described in detail in "Boot Loader Support – Read-While-Write Self-Programming" on page 202. "Memory Programming" on page 214 contains a detailed description on Flash data serial downloading using the SPI pins.

Constant tables can be allocated within the entire program memory address space (see the LPM – Load Program Memory instruction description).

Timing diagrams for instruction fetch and execution are presented in "Instruction Execution Timing" on page 11.

Figure 7. Program Memory Map



Data Memory

Figure 8 shows how the ATmega8 SRAM Memory is organized.

The lower 1120 Data Memory locations address the Register file, the I/O Memory, and the internal data SRAM. The first 96 locations address the Register file and I/O Memory, and the next 1024 locations address the internal data SRAM.

The five different addressing modes for the data memory cover: Direct, Indirect with Displacement, Indirect, Indirect with Pre-decrement, and Indirect with Post-increment. In the Register file, registers R26 to R31 feature the indirect addressing pointer registers.

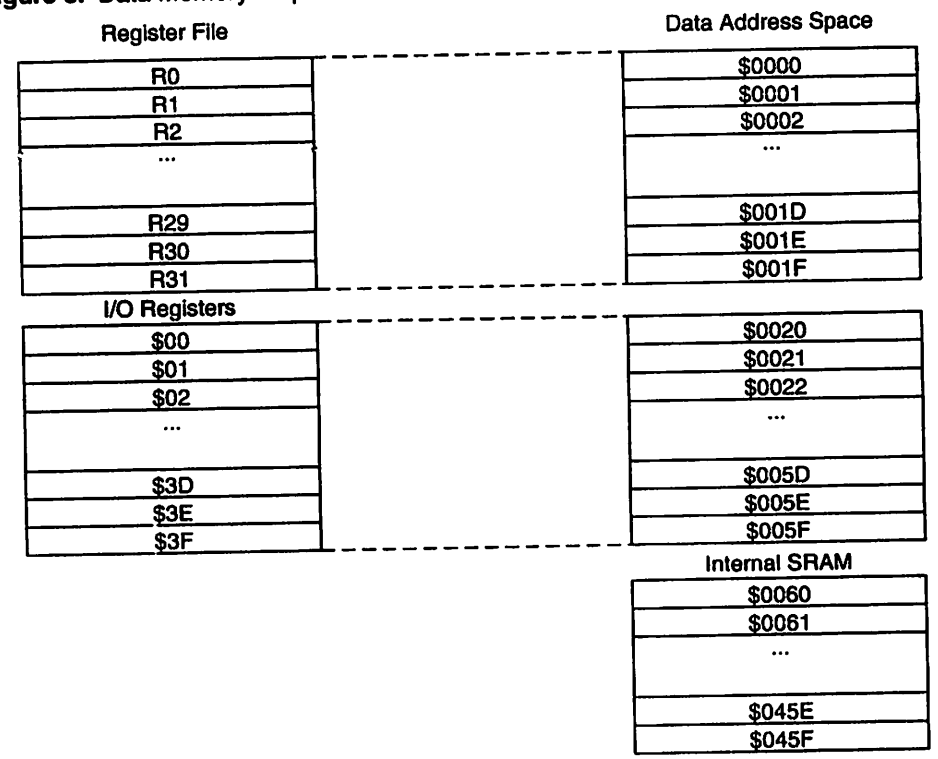
The direct addressing reaches the entire data space.

The Indirect with Displacement mode reaches 63 address locations from the base address given by the Y- or Z-register.

When using register indirect addressing modes with automatic pre-decrement and post-increment, the address registers X, Y and Z are decremented or incremented.

The 32 general purpose working registers, 64 I/O registers, and the 1024 bytes of internal data SRAM in the ATmega8 are all accessible through all these addressing modes. The Register file is described in "General Purpose Register File" on page 9.

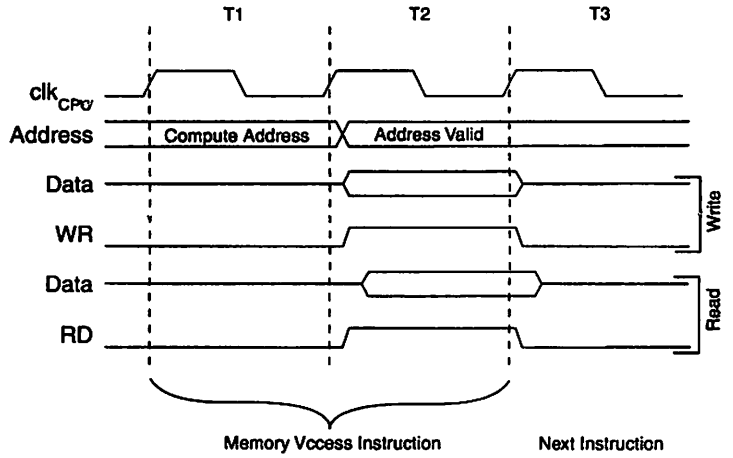
Figure 8. Data Memory Map



a Memory Access es

This section describes the general access timing concepts for internal memory access. The internal data SRAM access is performed in two clk_{CPU} cycles as described in Figure 9.

Figure 9. On-chip Data SRAM Access Cycles



'ROM Data Memory

The ATmega8 contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described bellow, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

For a detailed description of SPI data downloading to the EEPROM, see page 227.

ROM Read/Write Access

The EEPROM access registers are accessible in the I/O space.

The write access time for the EEPROM is given in Table 1 on page 18. A self-timing function, however, lets the user software detect when the next byte can be written. If the user code contains instructions that write the EEPROM, some precautions must be taken. In heavily filtered power supplies, V_{CC} is likely to rise or fall slowly on Power-up/down. This causes the device for some period of time to run at a voltage lower than specified as minimum for the clock frequency used. See "Preventing EEPROM Corruption" on page 20. for details on how to avoid problems in these situations.

In order to prevent unintentional EEPROM writes, a specific write procedure must be followed. Refer to the description of the EEPROM Control Register for details on this.

When the EEPROM is read, the CPU is halted for four clock cycles before the next instruction is executed. When the EEPROM is written, the CPU is halted for two clock cycles before the next instruction is executed.

EPROM Address er – EEARH and EEARL

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	–	EEAR8	EEARH
	EEAR7	EEAR6	EEAR5	EEAR4	EEAR3	EEAR2	EEAR1	EEAR0	EEARL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R/W	
	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	X	
	X	X	X	X	X	X	X	X	

- **Bits 15..9 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8 and will always read as zero.

- **Bits 8..0 – EEAR8..0: EEPROM Address**

The EEPROM Address Registers – EEARH and EEARL – specify the EEPROM address in the 512 bytes EEPROM space. The EEPROM data bytes are addressed linearly between 0 and 511. The initial value of EEAR is undefined. A proper value must be written before the EEPROM may be accessed.

EPROM Data Register –

Bit	7	6	5	4	3	2	1	0	
	MSB							LSB	EEDR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..0 – EEDR7..0: EEPROM Data**

For the EEPROM write operation, the EEDR register contains the data to be written to the EEPROM in the address given by the EEAR register. For the EEPROM read operation, the EEDR contains the data read out from the EEPROM at the address given by EEAR.

EPROM Control Register CR

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	EERIE	EEMWE	EEWE	EERE	EECR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	X	0	

- **Bits 7..4 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8 and will always read as zero.

- **Bit 3 – EERIE: EEPROM Ready Interrupt Enable**

Writing EERIE to one enables the EEPROM Ready Interrupt if the I bit in SREG is set. Writing EERIE to zero disables the interrupt. The EEPROM Ready interrupt generates a constant interrupt when EEWE is cleared.

- **Bit 2 – EEMWE: EEPROM Master Write Enable**

The EEMWE bit determines whether setting EEWE to one causes the EEPROM to be written. When EEMWE is set, setting EEWE within 4 clock cycles will write data to the EEPROM at the selected address If EEMWE is zero, setting EEWE will have no effect. When EEMWE has been written to one by software, hardware clears the bit to zero after four clock cycles. See the description of the EEWE bit for an EEPROM write procedure.

- **Bit 1 – EEWE: EEPROM Write Enable**

The EEPROM Write Enable Signal EEWE is the write strobe to the EEPROM. When address and data are correctly set up, the EEWE bit must be written to one to write the value into the EEPROM. The EEMWE bit must be written to one before a logical one is written to EEWE, otherwise no EEPROM write takes place. The following procedure should be followed when writing the EEPROM (the order of steps 3 and 4 is not essential):

1. Wait until EEWE becomes zero.
2. Wait until SPMEN in SPMCR becomes zero.
3. Write new EEPROM address to EEAR (optional).
4. Write new EEPROM data to EEDR (optional).
5. Write a logical one to the EEMWE bit while writing a zero to EEWE in EECR.
6. Within four clock cycles after setting EEMWE, write a logical one to EEWE.

The EEPROM can not be programmed during a CPU write to the Flash memory. The software must check that the Flash programming is completed before initiating a new EEPROM write. Step 2 is only relevant if the software contains a boot loader allowing the CPU to program the Flash. If the Flash is never being updated by the CPU, step 2 can be omitted. See "Boot Loader Support – Read-While-Write Self-Programming" on page 202 for details about boot programming.

Caution: An interrupt between step 5 and step 6 will make the write cycle fail, since the EEPROM Master Write Enable will time-out. If an interrupt routine accessing the EEPROM is interrupting another EEPROM access, the EEAR or EEDR register will be modified, causing the interrupted EEPROM access to fail. It is recommended to have the global interrupt flag cleared during all the steps to avoid these problems.

When the write access time has elapsed, the EEWE bit is cleared by hardware. The user software can poll this bit and wait for a zero before writing the next byte. When EEWE has been set, the CPU is halted for two cycles before the next instruction is executed.

- **Bit 0 – EERE: EEPROM Read Enable**

The EEPROM Read Enable Signal EERE is the read strobe to the EEPROM. When the correct address is set up in the EEAR register, the EERE bit must be written to a logic one to trigger the EEPROM read. The EEPROM read access takes one instruction, and the requested data is available immediately. When the EEPROM is read, the CPU is halted for four cycles before the next instruction is executed.

The user should poll the EEWE bit before starting the read operation. If a write operation is in progress, it is neither possible to read the EEPROM, nor to change the EEAR register.

The calibrated oscillator is used to time the EEPROM accesses. Table 1 lists the typical programming time for EEPROM access from the CPU.

Table 1. EEPROM Programming Time

Symbol	Number of Calibrated RC Oscillator Cycles[1]	Typ Programming Time
EEPROM Write (from CPU)	8448	8.5 ms

Note: 1. Uses 1 MHz clock, independent of CKSEL-fuse settings.

The following code examples show one assembly and one C function for writing to the EEPROM. The examples assume that interrupts are controlled (for example by disabling interrupts globally) so that no interrupts will occur during execution of these functions. The examples also assume that no Flash boot loader is present in the software. If such code is present, the EEPROM write function must also wait for any ongoing SPM command to finish.

Assembly Code Example

```
EEPROM_write:
  ; Wait for completion of previous write
  sbic EECR,EEWE
  rjmp EEPROM_write
  ; Set up address (r18:r17) in address register
  out  EEARH, r18
  out  EEARL, r17
  ; Write data (r16) to data register
  out  EEDR,r16
  ; Write logical one to EEMWE
  sbi  EECR,EEMWE
  ; Start eeprom write by setting EEWE
  sbi  EECR,EEWE
  ret
```

C Code Example

```
void EEPROM_write(unsigned int uiAddress, unsigned char ucData)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEWE))
    ;
  /* Set up address and data registers */
  EEAR = uiAddress;
  EEDR = ucData;
  /* Write logical one to EEMWE */
  EECR |= (1<<EEMWE);
  /* Start eeprom write by setting EEWE */
  EECR |= (1<<EEWE);
}
```

The next code examples show assembly and C functions for reading the EEPROM. The examples assume that interrupts are controlled so that no interrupts will occur during execution of these functions.

Assembly Code Example

```
EEPROM_read:
  ; Wait for completion of previous write
  sbic EECR.EEWE
  rjmp EEPROM_read
  ; Set up address (r18:r17) in address register
  out  EEARH, r18
  out  EEARL, r17
  ; Start eeprom read by writing EERE
  sbi  EECR,EERE
  ; Read data from data register
  in   r16,EEDR
  ret
```

C Code Example

```
unsigned char EEPROM_read(unsigned int uiAddress)
{
  /* Wait for completion of previous write */
  while(EECR & (1<<EEWE))
    ;
  /* Set up address register */
  EEAR = uiAddress;
  /* Start eeprom read by writing EERE */
  EECR |= (1<<EERE);
  /* Return data from data register */
  return EEDR;
}
```

enting EEPROM uption

During periods of low V_{CC}, the EEPROM data can be corrupted because the supply voltage is too low for the CPU and the EEPROM to operate properly. These issues are the same as for board level systems using EEPROM, and the same design solutions should be applied.

An EEPROM data corruption can be caused by two situations when the voltage is too low. First, a regular write sequence to the EEPROM requires a minimum voltage to operate correctly. Second, the CPU itself can execute instructions incorrectly, if the supply voltage is too low.

EEPROM data corruption can easily be avoided by following this design recommendation:

Keep the AVR RESET active (low) during periods of insufficient power supply voltage. This can be done by enabling the internal Brown-out Detector (BOD). If the detection level of the internal BOD does not match the needed detection level, an external low V_{CC} Reset Protection circuit can be used. If a reset occurs while a write operation is in progress, the write operation will be completed provided that the power supply voltage is sufficient.

Memory

The I/O space definition of the ATmega8 is shown in "" on page 255.

All ATmega8 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O registers as data space using LD and ST instructions, 0x20 must be added to these addresses.

For compatibility with future devices, reserved bits should be written to zero if accessed. Reserved I/O memory addresses should never be written.

Some of the status flags are cleared by writing a logical one to them. Note that the CBI and SBI instructions will operate on all bits in the I/O register, writing a one back into any flag read as set, thus clearing the flag. The CBI and SBI instructions work with registers 0x00 to 0x1F only.

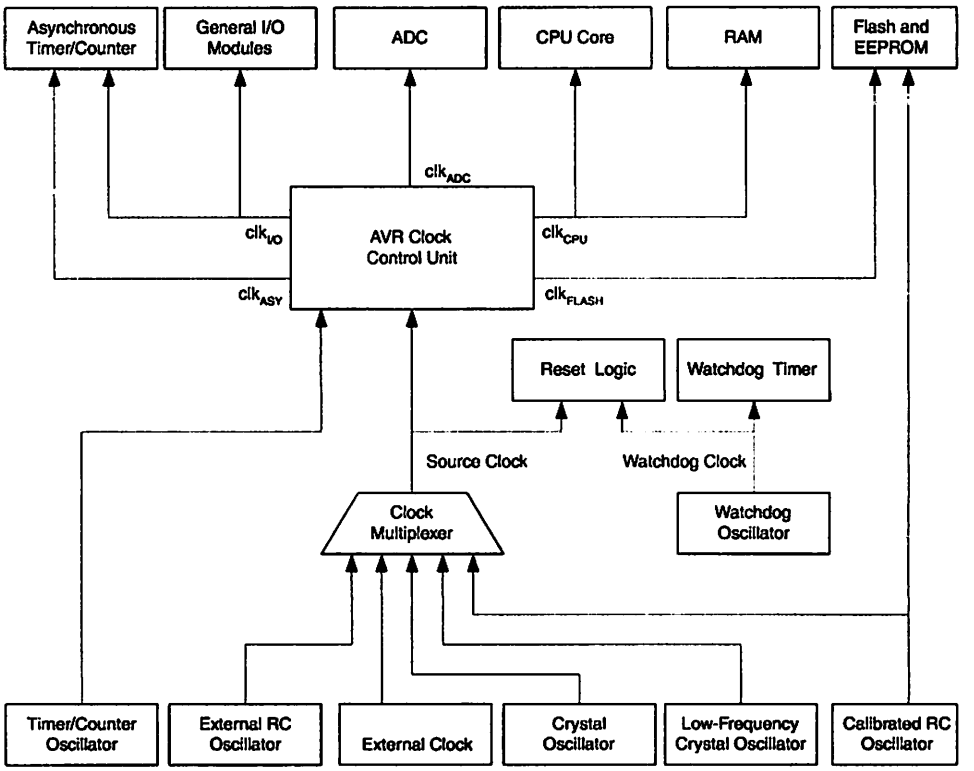
The I/O and peripherals control registers are explained in later sections.

;tem Clock and ck Options

:k Systems and their :ribution

Figure 10 presents the principal clock systems in the AVR and their distribution. All of the clocks need not be active at a given time. In order to reduce power consumption, the clocks to modules not being used can be halted by using different sleep modes, as described in "Power Management and Sleep Modes" on page 29. The clock systems are detailed Figure 10.

Figure 10. Clock Distribution



Clock – clk_{CPU}

The CPU clock is routed to parts of the system concerned with operation of the AVR core. Examples of such modules are the General Purpose Register File, the Status Register and the data memory holding the Stack Pointer. Halting the CPU clock inhibits the core from performing general operations and calculations.

:lock – $clk_{I/O}$

The I/O clock is used by the majority of the I/O modules, like Timer/Counters, SPI, and USART. The I/O clock is also used by the External Interrupt module, but note that some external interrupts are detected by asynchronous logic, allowing such interrupts to be detected even if the I/O clock is halted. Also note that address recognition in the TWI module is carried out asynchronously when $clk_{I/O}$ is halted, enabling TWI address reception in all sleep modes.

h Clock – clk_{FLASH}

The Flash clock controls operation of the Flash interface. The Flash clock is usually active simultaneously with the CPU clock.

ıchronous Timer Clock –
:y

The Asynchronous Timer clock allows the Asynchronous Timer/Counter to be clocked directly from an external 32 kHz clock crystal. The dedicated clock domain allows using this Timer/Counter as a real-time counter even when the device is in sleep mode. The Asynchronous Timer/Counter receives the same 32 kHz Oscillator as the chip clock. Thus, asyncronous operation is only available while the chip is clocked on the Internal Oscillator.

Clock – clk_{ADC}

The ADC is provided with a dedicated clock domain. This allows halting the CPU and I/O clocks in order to reduce noise generated by digital circuitry. This gives more accurate ADC conversion results.

;k Sources

The device has the following clock source options, selectable by Flash fuse bits as shown below. The clock from the selected source is input to the AVR clock generator, and routed to the appropriate modules.

Table 2. Device Clocking Options Select(1)

Device Clocking Option	CKSEL3..0
External Crystal/Ceramic Resonator	1111 - 1010
External Low-frequency Crystal	1001
External RC Oscillator	1000 - 0101
Calibrated Internal RC Oscillator	0100 - 0001
External Clock	0000

Note: 1. For all fuses "1" means unprogrammed while "0" means programmed.

The various choices for each clocking option is given in the following sections. When the CPU wakes up from Power-down or Power-save, the selected clock source is used to time the start-up, ensuring stable Oscillator operation before instruction execution starts. When the CPU starts from reset, there is as an additional delay allowing the power to reach a stable level before commencing normal operation. The Watchdog Oscillator is used for timing this real-time part of the start-up time. The number of WDT Oscillator cycles used for each time-out is shown in Table 3. The frequency of the Watchdog Oscillator is voltage dependent as shown in "ATmega8 Typical Characteristics – Preliminary Data". The device is shipped with CKSEL = "0001" and SUT = "10" (1 MHz Internal RC Oscillator, slowly rising power).

Table 3. Number of Watchdog Oscillator Cycles

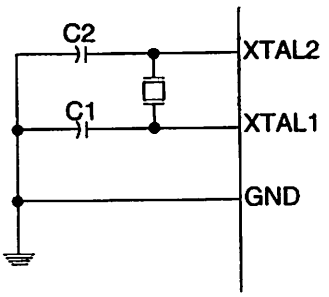
Typical Time-out (V_{CC} = 5.0V)	Typical Time-out (V_{CC} = 3.0V)	Number of Cycles
4.1 ms	4.3 ms	4K (4,096)
65 ms	69 ms	64K (65,536)

;tal Oscillator

XTAL1 and XTAL2 are input and output, respectively, of an inverting amplifier which can be configured for use as an On-chip Oscillator, as shown in Figure 11. Either a quartz crystal or a ceramic resonator may be used. The CKOPT fuse selects between two different oscillator amplifier modes. When CKOPT is programmed, the Oscillator output will oscillate a full rail-to-rail swing on the output. This mode is suitable when operating in a very noisy environment or when the output from XTAL2 drives a second clock buffer. This mode has a wide frequency range. When CKOPT is unprogrammed, the Oscillator has a smaller output swing. This reduces power consumption considerably. This mode has a limited frequency range and it cannot be used to drive other clock buffers.

For resonators, the maximum frequency is 8 MHz with CKOPT unprogrammed and 16 MHz with CKOPT programmed. C1 and C2 should always be equal for both crystals and resonators. The optimal value of the capacitors depends on the crystal or resonator in use, the amount of stray capacitance, and the electromagnetic noise of the environment. Some initial guidelines for choosing capacitors for use with crystals are given in Table 4. For ceramic resonators, the capacitor values given by the manufacturer should be used. For more information on how to choose capacitors and other details on oscillator operation, refer to the Multi-purpose Oscillator application note.

Figure 11. Crystal Oscillator Connections



The Oscillator can operate in three different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..1 as shown in Table 4.

Table 4. Crystal Oscillator Operating Modes

CKOPT	CKSEL3..1	Frequency Range(MHz)[1]	Recommended Range for Capacitors C1 and C2 for Use with Crystals (pF)
1	101[2]	0.4 - 0.9	–
1	110	0.9 - 3.0	12 - 22
1	111	3.0 - 8.0	12 - 22
0	101, 110, 111	1.0 ≤	12 - 22

Notes: 1. The frequency ranges are preliminary values. Actual values are TBD.
2. This option should not be used with crystals, only with ceramic resonators.

The CKSEL0 fuse together with the SUT1..0 fuses select the start-up times as shown in Table 5.

Table 5. Start-up Times for the Crystal Oscillator Clock Selection

CKSEL0	SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
0	00	258 CK[1]	4.1 ms	Ceramic resonator, fast rising power
0	01	258 CK[1]	65 ms	Ceramic resonator, slowly rising power
0	10	1K CK[2]	–	Ceramic resonator, BOD enabled
0	11	1K CK[2]	4.1 ms	Ceramic resonator, fast rising power
1	00	1K CK[2]	65 ms	Ceramic resonator, slowly rising power
1	01	16K CK	–	Crystal Oscillator, BOD enabled
1	10	16K CK	4.1 ms	Crystal Oscillator, fast rising power
1	11	16K CK	65 ms	Crystal Oscillator, slowly rising power

Notes: 1. These options should only be used when not operating close to the maximum frequency of the device, and only if frequency stability at start-up is not important for the application. These options are not suitable for crystals.
2. These options are intended for use with ceramic resonators and will ensure frequency stability at start-up. They can also be used with crystals when not operating close to the maximum frequency of the device, and if frequency stability at start-up is not important for the application.

equency Crystal ator

To use a 32.768 kHz watch crystal as the clock source for the device, the Low-frequency Crystal Oscillator must be selected by setting the CKSEL fuses to "1001". The crystal should be connected as shown in Figure 11. By programming the CKOPT fuse, the user can enable internal capacitors on XTAL1 and XTAL2, thereby removing the need for external capacitors. The internal capacitors have a nominal value of 36 pF. Refer to the 32 kHz Crystal Oscillator application note for details on oscillator operation and how to choose appropriate values for C1 and C2.

When this Oscillator is selected, start-up times are determined by the SUT fuses as shown in Table 6.

Table 6. Start-up Times for the Low-frequency Crystal Oscillator Clock Selection

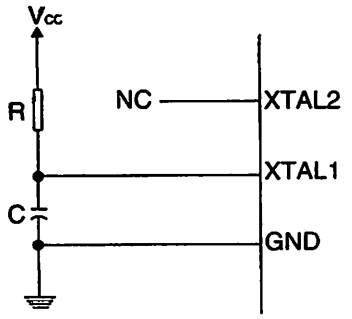
SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	1K CK[1]	4.1 ms	Fast rising power or BOD enabled
01	1K CK[1]	65 ms	Slowly rising power
10	32K CK	65 ms	Stable frequency at start-up
11	Reserved		

Note: 1. These options should only be used if frequency stability at start-up is not important for the application.

ıal RC Oscillator

For timing insensitive applications, the external RC configuration shown in Figure 12 can be used. The frequency is roughly estimated by the equation f = 1/(3RC). C should be at least 22 pF. By programming the CKOPT fuse, the user can enable an internal 36 pF capacitor between XTAL1 and GND, thereby removing the need for an external capacitor. For more information on oscillator operation and details on how to choose R and C, refer to the External RC Oscillator application note.

Figure 12. External RC Configuration



The Oscillator can operate in four different modes, each optimized for a specific frequency range. The operating mode is selected by the fuses CKSEL3..0 as shown in Table 7.

Table 7. External RC Oscillator Operating Modes

CKSEL3..0	Frequency Range (MHz)
0101	≤ 0.9
0110	0.9 - 3.0
0111	3.0 - 8.0
1000	8.0 - 12.0

When this oscillator is selected, start-up times are determined by the SUT fuses as shown in Table 8.

Table 8. Start-up Times for the External RC Oscillator Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	18 CK	–	BOD enabled
01	18 CK	4.1 ms	Fast rising power
10	18 CK	65 ms	Slowly rising power
11	6 CK[1]	4.1 ms	Fast rising power or BOD enabled

Note: 1. This option should not be used when operating close to the maximum frequency of the device.

ırated Internal RC llator

The calibrated internal RC Oscillator provides a fixed 1.0, 2.0, 4.0, or 8.0 MHz clock. All frequencies are nominal values at 5V and 25°C. This clock may be selected as the system clock by programming the CKSEL fuses as shown in Table 9. If selected, it will operate with no external components. The CKOPT fuse should always be unprogrammed when using this clock option. During reset, hardware loads the calibration byte into the OSCCAL register and thereby automatically calibrates the RC Oscillator. At 5V, 25°C and 1.0 MHz Oscillator frequency selected, this calibration gives a frequency within ± 1% of the nominal frequency. When this Oscillator is used as the chip clock, the Watchdog Oscillator will still be used for the Watchdog Timer and for the reset time-out. For more information on the pre-programmed calibration value, see the section "Calibration Byte" on page 216.

Table 9. Internal Calibrated RC Oscillator Operating Modes

CKSEL3..0	Nominal Frequency (MHz)
0001[1]	1.0
0010	2.0
0011	4.0
0100	8.0

Note: 1. The device is shipped with this option selected.

When this oscillator is selected, start-up times are determined by the SUT fuses as shown in Table 10. XTAL1 and XTAL2 should be left unconnected (NC).

Table 10. Start-up Times for the Internal Calibrated RC Oscillator Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	6 CK	–	BOD enabled
01	6 CK	4.1 ms	Fast rising power
10[1]	6 CK	65 ms	Slowly rising power
11	Reserved		

Note: 1. The device is shipped with this option selected.

ator Calibration Register :CAL

Bit	7	6	5	4	3	2	1	0	
	CAL7	CAL6	CAL5	CAL4	CAL3	CAL2	CAL1	CAL0	OSCCAL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	Device Specific Calibration Value								

- **Bits 7..0 – CAL7..0: Oscillator Calibration Value**

Writing the calibration byte to this address will trim the Internal Oscillator to remove process variations from the oscillator frequency. During Reset, the 1 MHz calibration value which is located in the signature row high byte (address 0x00) is automatically loaded into the OSCCAL Register. If the internal RC is used at other frequencies, the calibration values must be loaded manually. This can be done by first reading the signature row by a programmer, and then store the calibration values in the Flash or EEPROM. Then the value can be read by software and loaded into the OSCCAL Register. When OSCCAL is zero, the lowest available frequency is chosen. Writing non-zero values to this register will increase the frequency of the Internal Oscillator. Writing 0xFF to the register gives

the highest available frequency. The calibrated Oscillator is used to time EEPROM and Flash access. If EEPROM or Flash is written, do not calibrate to more than 10% above the nominal frequency. Otherwise, the EEPROM or Flash write may fail. Note that the Oscillator is intended for calibration to 1.0, 2.0, 4.0, or 8.0 MHz. Tuning to other values is not guaranteed, as indicated in Table 11.

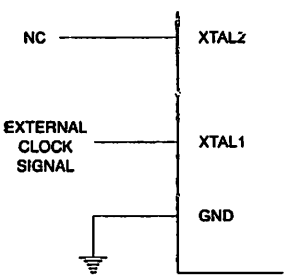
Table 11. Internal RC Oscillator Frequency Range

OSCCAL Value	Min Frequency in Percentage of Nominal Frequency (%)	Max Frequency in Percentage of Nominal Frequency (%)
0x00	50	100
0x7F	75	150
0xFF	100	200

rnal Clock

To drive the device from an external clock source, XTAL1 should be driven as shown in Figure 13. To run the device on an external clock, the CKSEL fuses must be programmed to "0000". By programming the CKOPT fuse, the user can enable an internal 36 pF capacitor between XTAL1 and GND.

Figure 13. External Clock Drive Configuration



When this clock source is selected, start-up times are determined by the SUT fuses as shown in Table 12.

Table 12. Start-up Times for the External Clock Selection

SUT1..0	Start-up Time from Power-down and Power-save	Additional Delay from Reset (V_{CC} = 5.0V)	Recommended Usage
00	6 CK	–	BOD enabled
01	6 CK	4.1 ms	Fast rising power
10	6 CK	65 ms	Slowly rising power
11	Reserved		

r/Counter Oscillator

For AVR microcontrollers with Timer/Counter Oscillator pins (TOSC1 and TOSC2), the crystal is connected directly between the pins. No external capacitors are needed. The Oscillator is optimized for use with a 32.768 kHz watch crystal. Applying an external clock source to TOSC1 is not recommended.

ver Management
l Sleep Modes

Sleep modes enable the application to shut down unused modules in the MCU, thereby saving power. The AVR provides various sleep modes allowing the user to tailor the power consumption to the application's requirements.

To enter any of the five sleep modes, the SE bit in MCUCR must be written to logic one and a SLEEP instruction must be executed. The SM2, SM1, and SM0 bits in the MCUCR register select which sleep mode (Idle, ADC Noise Reduction, Power-down, Power-save, or Standby) will be activated by the SLEEP instruction. See Table 13 for a summary. If an enabled interrupt occurs while the MCU is in a sleep mode, the MCU wakes up. The MCU is then halted for four cycles in addition to the start-up time, it executes the interrupt routine, and resumes execution from the instruction following SLEEP. The contents of the register file and SRAM are unaltered when the device wakes up from sleep. If a reset occurs during sleep mode, the MCU wakes up and executes from the Reset vector.

Note that the Extended Standby mode present in many other AVR MCUs has been removed in the ATmega8, as the TOSC and XTAL inputs share the same physical pins.

Figure 10 on page 22 presents the different clock systems in the ATmega8, and their distribution. The figure is helpful in selecting an appropriate sleep mode.

Control Register – CR

The MCU Control Register contains control bits for power management.

Bit	7	6	5	4	3	2	1	0	
	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – SE: Sleep Enable**

The SE bit must be written to logic one to make the MCU enter the sleep mode when the SLEEP instruction is executed. To avoid the MCU entering the sleep mode unless it is the programmer's purpose, it is recommended to set the Sleep Enable (SE) bit just before the execution of the SLEEP instruction.

- **Bits 6..4 – SM2..0: Sleep Mode Select Bits 2, 1, and 0**

These bits select between the six available sleep modes as shown in Table 13.

Table 13. Sleep Mode Select

SM2	SM1	SM0	Sleep Mode
0	0	0	Idle
0	0	1	ADC Noise Reduction
0	1	0	Power-down
0	1	1	Power-save
1	0	0	Reserved
1	0	1	Reserved
1	1	0	Standby(1)

Note: 1. Standby mode is only available with external crystals or resonators.

ɔde

When the SM2..0 bits are written to 000, the SLEEP instruction makes the MCU enter Idle mode, stopping the CPU but allowing SPI, USART, Analog Comparator, ADC, Two-wire Serial Interface, Timer/Counters, Watchdog, and the interrupt system to continue operating. This sleep mode basically halts clk_{CPU} and clk_{FLASH}, while allowing the other clocks to run.

Idle mode enables the MCU to wake up from external triggered interrupts as well as internal ones like the Timer Overflow and USART Transmit Complete interrupts. If wake-up from the Analog Comparator interrupt is not required, the Analog Comparator can be powered down by setting the ACD bit in the Analog Comparator Control and Status register – ACSR. This will reduce power consumption in Idle mode. If the ADC is enabled, a conversion starts automatically when this mode is entered.

loise Reduction

When the SM2..0 bits are written to 001, the SLEEP instruction makes the MCU enter ADC Noise Reduction mode, stopping the CPU but allowing the ADC, the external interrupts, the Two-wire Serial Interface address watch, Timer/Counter2 and the Watchdog to continue operating (if enabled). This sleep mode basically halts $clk_{I/O}$, clk_{CPU}, and clk_{FLASH}, while allowing the other clocks to run.

This improves the noise environment for the ADC, enabling higher resolution measurements. If the ADC is enabled, a conversion starts automatically when this mode is entered. Apart form the ADC Conversion Complete interrupt, only an External Reset, a Watchdog Reset, a Brown-out Reset, a Two-wire Serial Interface address match interrupt, a Timer/Counter2 interrupt, an SPM/EEPROM ready interrupt, or an external level interrupt on INT0 or INT1, can wake up the MCU from ADC Noise Reduction mode.

'-down Mode

When the SM2..0 bits are written to 010, the SLEEP instruction makes the MCU enter Power-down mode. In this mode, the External Oscillator is stopped, while the external interrupts, the Two-wire Serial Interface address watch, and the Watchdog continue operating (if enabled). Only an External Reset, a Watchdog Reset, a Brown-out Reset, a Two-wire Serial Interface address match interrupt, or an external level interrupt on INT0 or INT1, can wake up the MCU. This sleep mode basically halts all generated clocks, allowing operation of asynchronous modules only.

Note that if a level triggered interrupt is used for wake-up from Power-down mode, the changed level must be held for some time to wake up the MCU. Refer to "External Interrupts" on page 62 for details.

When waking up from Power-down mode, there is a delay from the wake-up condition occurs until the wake-up becomes effective. This allows the clock to restart and become stable after having been stopped. The wake-up period is defined by the same CKSEL fuses that define the reset time-out period, as described in "Clock Sources" on page 23.

r-save Mode

When the SM2..0 bits are written to 011, the SLEEP instruction makes the MCU enter Power-save mode. This mode is identical to Power-down, with one exception:

- If Timer/Counter2 is clocked asynchronously, i.e. the AS2 bit in ASSR is set, Timer/Counter2 will run during sleep. The device can wake up from either Timer Overflow or Output Compare event from Timer/Counter2 if the corresponding Timer/Counter2 interrupt enable bits are set in TIMSK, and the global interrupt enable bit in SREG is set.

If the asynchronous timer is NOT clocked asynchronously, Power-down mode is recommended instead of Power-save mode because the contents of the registers in the asynchronous timer should be considered undefined after wake-up in Power-save mode if AS2 is 0.

This sleep mode basically halts all clocks except clk_{ASY}, allowing operation only of asynchronous modules, including Timer/Counter 2 if clocked asynchronously.

lby Mode

When the SM2..0 bits are 110 and an external crystal/resonator clock option is selected, the SLEEP instruction makes the MCU enter Standby mode. This mode is identical to Power-down with the exception that the Oscillator is kept running. From Standby mode, the device wakes up in 6 clock cycles.

l4. Active Clock Domains and Wake-up Sources in the Different Sleep Modes

	Active Clock Domains					Oscillators		Wake-up Sources					
	clk_{CPU}	clk_{FLASH}	clk_{IO}	clk_{ADC}	clk_{ASY}	Main Clock Source Enabled	Timer Osc. Enabled	INT1 INT0	TWI Address Match	Timer 2	SPM/ EEPROM Ready	ADC	Other I/O
			X	X	X	X	X[2]	X	X	X	X	X	X
loise tion				X	X	X	X[2]	X[3]	X	X	X	X	
								X[3]	X				
					X[2]		X[2]	X[3]	X	X[2]			
y[1]						X		X[3]	X				

1. External Crystal or resonator selected as clock source.
2. If AS2 bit in ASSR is set.
3. Only level interrupt INT1 and INT0.

nizing Power umption

There are several issues to consider when trying to minimize the power consumption in an AVR controlled system. In general, sleep modes should be used as much as possible, and the sleep mode should be selected so that as few as possible of the device's functions are operating. All functions not needed should be disabled. In particular, the following modules may need special consideration when trying to achieve the lowest possible power consumption.

g-to-Digital Converter

If enabled, the ADC will be enabled in all sleep modes. To save power, the ADC should be disabled before entering any sleep mode. When the ADC is turned off and on again, the next conversion will be an extended conversion. Refer to "Analog-to-Digital Converter" on page 189 for details on ADC operation.

g Comparator

When entering Idle mode, the Analog Comparator should be disabled if not used. When entering ADC Noise Reduction mode, the Analog Comparator should be disabled. In the other sleep modes, the Analog Comparator is automatically disabled. However, if the Analog Comparator is set up to use the Internal Voltage Reference as input, the Analog Comparator should be disabled in all sleep modes. Otherwise, the Internal Voltage Reference will be enabled, independent of sleep mode. Refer to "Analog Comparator" on page 186 for details on how to configure the Analog Comparator.

n-out Detector

If the Brown-out Detector is not needed in the application, this module should be turned off. If the Brown-out Detector is enabled by the BODEN fuse, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to "Brown-out Detection" on page 36 for details on how to configure the Brown-out Detector.

l Voltage Reference

The Internal Voltage Reference will be enabled when needed by the Brown-out Detector, the Analog Comparator or the ADC. If these modules are disabled as described in the sections above, the internal voltage reference will be disabled and it will not be consuming power. When turned on again, the user must allow the reference to start up before the output is used. If the reference is kept on in sleep mode, the output can be used immediately. Refer to "Internal Voltage Reference" on page 38 for details on the start-up time.

log Timer

If the Watchdog Timer is not needed in the application, this module should be turned off. If the Watchdog Timer is enabled, it will be enabled in all sleep modes, and hence, always consume power. In the deeper sleep modes, this will contribute significantly to the total current consumption. Refer to "Watchdog Timer" on page 38 for details on how to configure the Watchdog Timer.

ns

When entering a sleep mode, all port pins should be configured to use minimum power. The most important thing is then to ensure that no pins drive resistive loads. In sleep modes where the both the I/O clock ($clk_{I/O}$) and the ADC clock (clk_{ADC}) are stopped, the input buffers of the device will be disabled. This ensures that no power is consumed by the input logic when not needed. In some cases, the input logic is needed for detecting wake-up conditions, and it will then be enabled. Refer to the section "Digital Input Enable and Sleep Modes" on page 51 for details on which pins are enabled. If the input buffer is enabled and the input signal is left floating or have an analog signal level close to $V_{CC}/2$, the input buffer will use excessive power.

stem Control and set

etting the AVR

During reset, all I/O registers are set to their initial values, and the program starts execution from the Reset Vector. If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the interrupt vectors are in the boot section or vice versa. The circuit diagram in Figure 14 shows the reset logic. Table 15 defines the electrical parameters of the reset circuitry.

The I/O ports of the AVR are immediately reset to their initial state when a reset source goes active. This does not require any clock source to be running.

After all reset sources have gone inactive, a delay counter is invoked, stretching the internal reset. This allows the power to reach a stable level before normal operation starts. The time-out period of the delay counter is defined by the user through the CKSEL fuses. The different selections for the delay period are presented in "Clock Sources" on page 23.

et Sources

The ATmega8 has four sources of Reset:

- Power-on Reset. The MCU is reset when the supply voltage is below the Power-on Reset threshold (V_{POT}).
- External Reset. The MCU is reset when a low level is present on the $\overline{\text{RESET}}$ pin for longer than the minimum pulse length.
- Watchdog Reset. The MCU is reset when the Watchdog Timer period expires and the Watchdog is enabled.
- Brown-out Reset. The MCU is reset when the supply voltage V_{CC} is below the Brown-out Reset threshold (V_{BOT}) and the Brown-out Detector is enabled.

Figure 14. Reset Logic

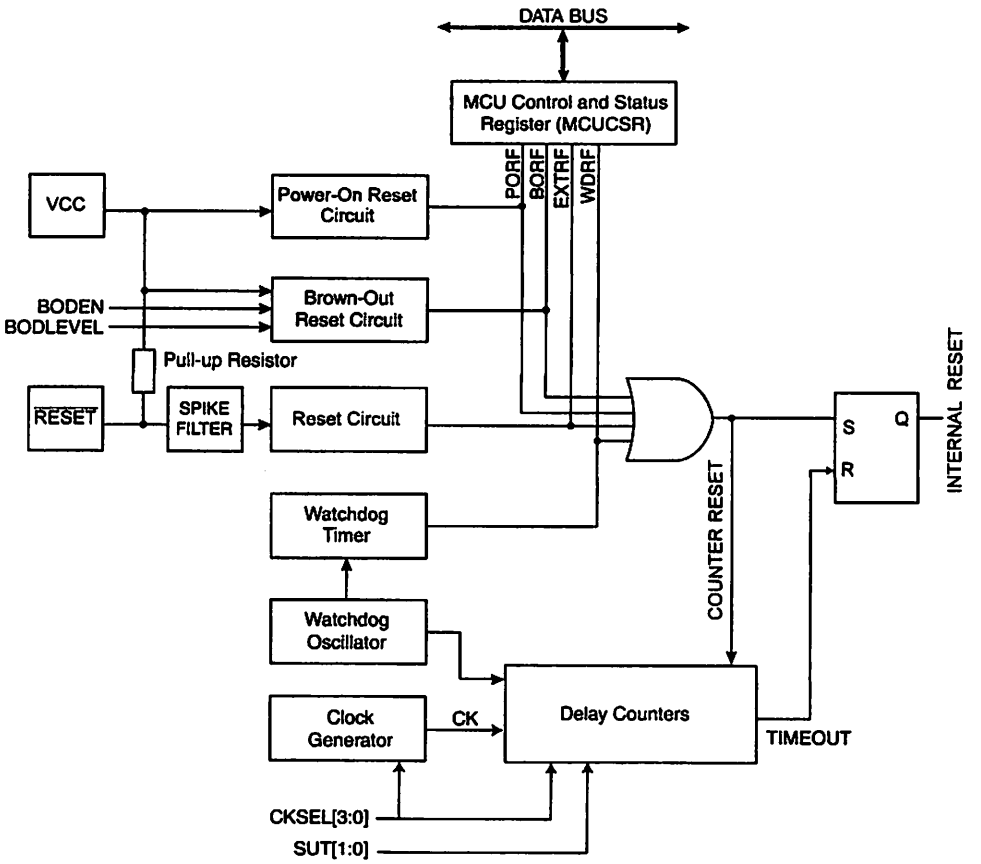


Table 15. Reset Characteristics

Symbol	Parameter	Condition	Min	Typ	Max	Units
V_{POT}	Power-on Reset Threshold Voltage (rising)[1]			1.4	2.3	V
	Power-on Reset Threshold Voltage (falling)			1.3	2.3	V
V_{RST}	$\overline{\text{RESET}}$ Pin Threshold Voltage		0.1		0.9	V_{CC}
t_{RST}	Minimum pulse width on $\overline{\text{RESET}}$ Pin			50		ns
V_{BOT}	Brown-out Reset Threshold Voltage	BODLEVEL = 1	2.5	2.7	3.2	V
		BODLEVEL = 0	3.7	4.0	4.2	
t_{BOD}	Minimum low voltage period for Brown-out Detection	BODLEVEL = 1		2		µs
		BODLEVEL = 0		2		µs
V_{HYST}	Brown-out Detector hysteresis			130		mV

Note: 1. The Power-on Reset will not work unless the supply voltage has been below V_{POT} (falling)

›n Reset

A Power-on Reset (POR) pulse is generated by an on-chip detection circuit. The detection level is defined in Table 15. The POR is activated whenever V_{CC} is below the detection level. The POR circuit can be used to trigger the Start-up Reset, as well as to detect a failure in supply voltage.

A Power-on Reset (POR) circuit ensures that the device is reset from power-on. Reaching the Power-on Reset threshold voltage invokes the delay counter, which determines how long the device is kept in RESET after V_{CC} rise. The RESET signal is activated again, without any delay, when V_{CC} decreases below the detection level.

Figure 15. MCU Start-up, $\overline{\text{RESET}}$ Tied to V_{CC}

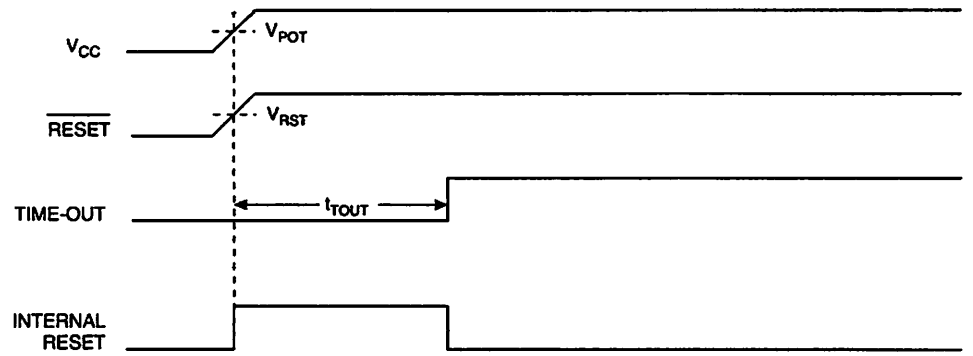
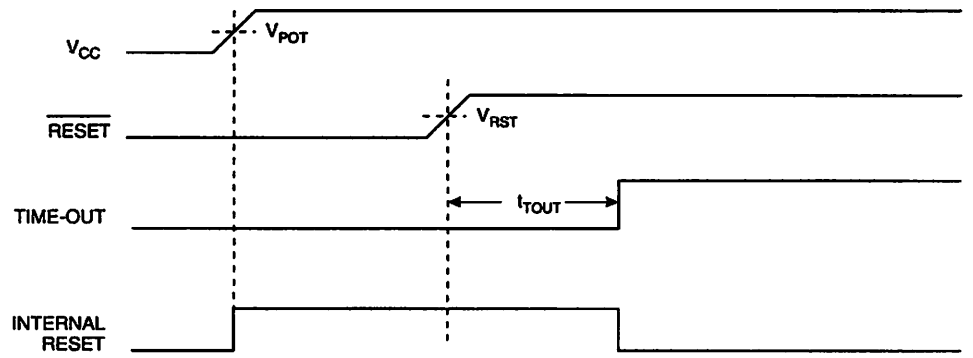


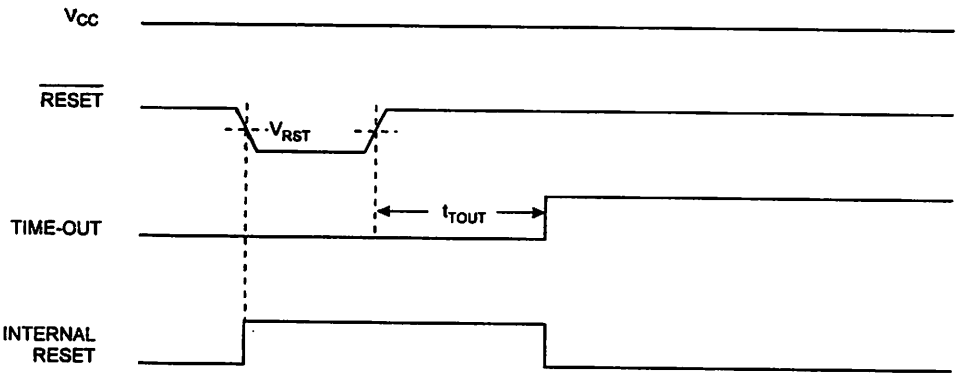
Figure 16. MCU Start-up, $\overline{\text{RESET}}$ Extended Externally



rnal Reset

An External Reset is generated by a low level on the $\overline{\text{RESET}}$ pin. Reset pulses longer than the minimum pulse width (see Table 15) will generate a reset, even if the clock is not running. Shorter pulses are not guaranteed to generate a reset. When the applied signal reaches the Reset Threshold Voltage – V_{RST} on its positive edge, the delay counter starts the MCU after the Time-out period t_{TOUT} has expired.

Figure 17. External Reset During Operation



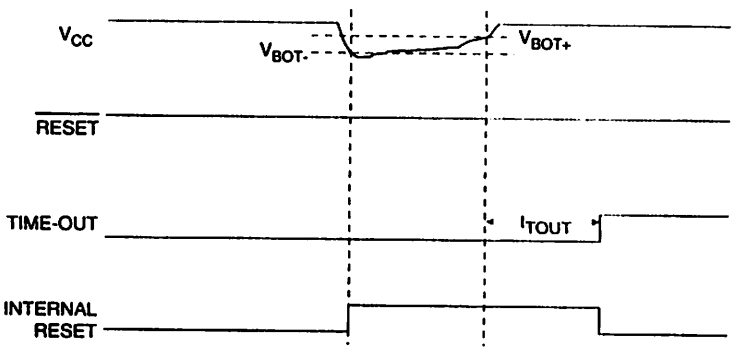
n-out Detection

ATmega8 has an On-chip Brown-out Detection (BOD) circuit for monitoring the V_{CC} level during operation by comparing it to a fixed trigger level. The trigger level for the BOD can be selected by the fuse BODLEVEL to be 2.7V (BODLEVEL unprogrammed), or 4.0V (BODLEVEL programmed). The trigger level has a hysteresis to ensure spike free Brown-out Detection. The hysteresis on the detection level should be interpreted as $V_{BOT+} = V_{BOT} + V_{HYST}/2$ and $V_{BOT-} = V_{BOT} - V_{HYST}/2$.

The BOD circuit can be enabled/disabled by the fuse BODEN. When the BOD is enabled (BODEN programmed), and V_{CC} decreases to a value below the trigger level (V_{BOT-} in Figure 18), the Brown-out Reset is immediately activated. When V_{CC} increases above the trigger level (V_{BOT+} in Figure 18), the delay counter starts the MCU after the time-out period t_{TOUT} has expired.

The BOD circuit will only detect a drop in V_{CC} if the voltage stays below the trigger level for longer than t_{BOD} given in Table 15.

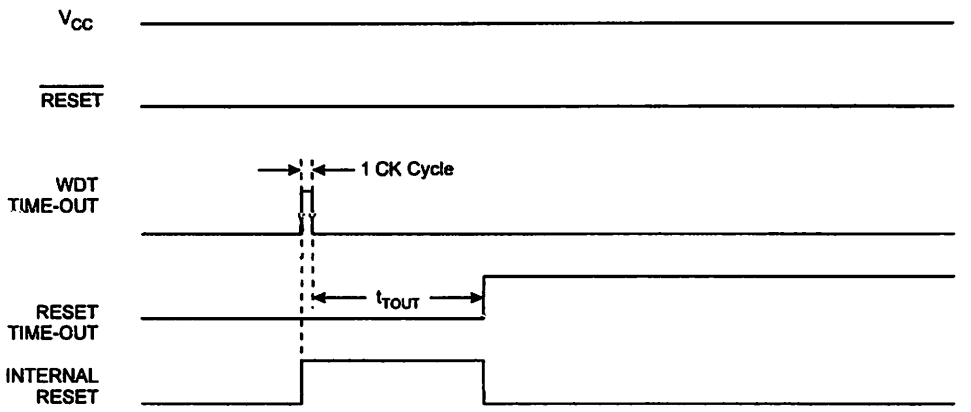
Figure 18. Brown-out Reset During Operation



hdog Reset

When the Watchdog times out, it will generate a short reset pulse of 1 CK cycle duration. On the falling edge of this pulse, the delay timer starts counting the Time-out period t_{TOUT}. Refer to page 38 for details on operation of the Watchdog Timer.

Figure 19. Watchdog Reset During Operation



Control and Status ster – MCUCSR

The MCU Control and Status Register provides information on which reset source caused an MCU Reset.

Bit	7	6	5	4	3	2	1	0	
	–	–	–	–	WDRF	BORF	EXTRF	PORF	MCUCSR
Read/Write	R	R	R	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	See Bit Description				

- **Bit 7..4 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8 and always read as zero.

- **Bit 3 – WDRF: Watchdog Reset Flag**

This bit is set if a Watchdog Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 2 – BORF: Brown-out Reset Flag**

This bit is set if a Brown-out Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 1 – EXTRF: External Reset Flag**

This bit is set if an External Reset occurs. The bit is reset by a Power-on Reset, or by writing a logic zero to the flag.

- **Bit 0 – PORF: Power-on Reset Flag**

This bit is set if a Power-on Reset occurs. The bit is reset only by writing a logic zero to the flag.

To make use of the reset flags to identify a reset condition, the user should read and then reset the MCUCSR as early as possible in the program. If the register is cleared before another reset occurs, the source of the reset can be found by examining the reset flags.

al Voltage ence

ATmega8 features an internal bandgap reference. This reference is used for Brown-out Detection, and it can be used as an input to the Analog Comparator or the ADC. The 2.56V reference to the ADC is generated from the internal bandgap reference.

Reference Enable and Start-up Time

The voltage reference has a start-up time that may influence the way it should be used. The start-up time is given in Table 16. To save power, the reference is not always turned on. The reference is on during the following situations:

1. When the BOD is enabled (by programming the BODEN fuse).
2. When the bandgap reference is connected to the Analog Comparator (by setting the ACBG bit in ACSR).
3. When the ADC is enabled.

Thus, when the BOD is not enabled, after setting the ACBG bit or enabling the ADC, the user must always allow the reference to start up before the output from the Analog Comparator or ADC is used. To reduce power consumption in Power-down mode, the user can avoid the three conditions above to ensure that the reference is turned off before entering Power-down mode.

Table 16. Internal Voltage Reference Characteristics

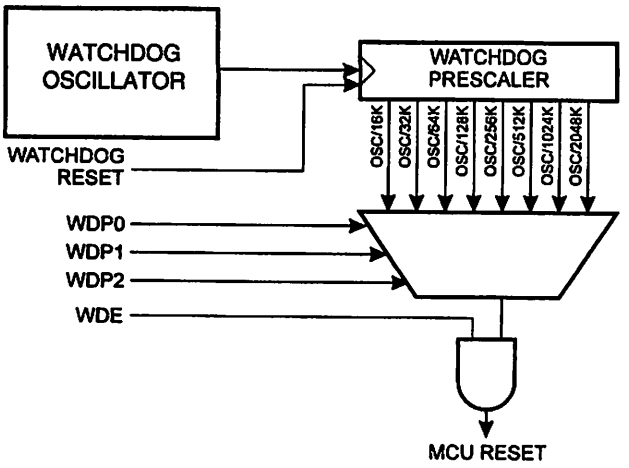
Symbol	Parameter	Min	Typ	Max	Units
V_{BG}	Bandgap reference voltage	1.15	1.23	1.35	V
t_{BG}	Bandgap reference start-up time		40	70	µs
I_{BG}	Bandgap reference current consumption		10		µA

dog Timer

The Watchdog Timer is clocked from a separate On-chip Oscillator which runs at 1 MHz. This is the typical value at V_{CC} = 5V. See characterization data for typical values at other V_{CC} levels. By controlling the Watchdog Timer prescaler, the Watchdog Reset interval can be adjusted as shown in Table 17 on page 40. The WDR – Watchdog Reset – instruction resets the Watchdog Timer. The Watchdog Timer is also reset when it is disabled and when a Chip Reset occurs. Eight different clock cycle periods can be selected to determine the reset period. If the reset period expires without another Watchdog Reset, the ATmega8 resets and executes from the Reset vector. For timing details on the Watchdog Reset, refer to page 37.

To prevent unintentional disabling of the Watchdog, a special turn-off sequence must be followed when the Watchdog is disabled. Refer to the description of the Watchdog Timer Control Register for details.

Figure 20. Watchdog Timer



hdog Timer Control ister – WDTCR

Bit	7	6	5	4	3	2	1	0	
	–	–	–	WDCE	WDE	WDP2	WDP1	WDP0	WDTCR
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bits 7..5 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8 and will always read as zero.

- **Bit 4 – WDCE: Watchdog Change Enable**

This bit must be set when the WDE bit is written to logic zero. Otherwise, the Watchdog will not be disabled. Once written to one, hardware will clear this bit after four clock cycles. Refer to the description of the WDE bit for a Watchdog disable procedure. In Safety Level 1 and 2, this bit must also be set when changing the prescaler bits. See the Code Examples on page 40.

- **Bit 3 – WDE: Watchdog Enable**

When the WDE is written to logic one, the Watchdog Timer is enabled, and if the WDE is written to logic zero, the Watchdog Timer function is disabled. WDE can only be cleared if the WDCE bit has logic level one. To disable an enabled Watchdog Timer, the following procedure must be followed:

1. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE even though it is set to one before the disable operation starts.
2. Within the next four clock cycles, write a logic 0 to WDE. This disables the Watchdog.

- **Bits 2..0 – WDP2, WDP1, WDP0: Watchdog Timer Prescaler 2, 1, and 0**

The WDP2, WDP1, and WDP0 bits determine the Watchdog Timer prescaling when the Watchdog Timer is enabled. The different prescaling values and their corresponding Timeout Periods are shown in Table 17.

Table 17. Watchdog Timer Prescale Select

WDP2	WDP1	WDP0	Number of WDT Oscillator Cycles	Typical Time-out at V_{CC} = 3.0V	Typical Time-out at V_{CC} = 5.0V
0	0	0	16K (16,384)	17.1 ms	16.3 ms
0	0	1	32K (32,768)	34.3 ms	32.5 ms
0	1	0	64K (65,536)	68.5 ms	65 ms
0	1	1	128K (131,072)	0.14 s	0.13 s
1	0	0	256K (262,144)	0.27 s	0.26 s
1	0	1	512K (524,288)	0.55 s	0.52 s
1	1	0	1,024K (1,048,576)	1.1 s	1.0 s
1	1	1	2,048K (2,097,152)	2.2 s	2.1 s

The following code example shows one assembly and one C function for turning off the WDT. The example assumes that interrupts are controlled (for example, by disabling interrupts globally) so that no interrupts will occur during execution of these functions.

ed Sequences for inging the ifiguration of the chdog Timer.

The sequence for changing the Watchdog Timer configuration differs slightly between the safety levels. Separate procedures are described for each level.

Assembly Code Example

```
WDT_off:
  ; Write logical one to WDCE and WDE
  ldi  r16, (1<<WDCE)|(1<<WDE)
  out  WDTCR, r16
  ; Turn off WDT
  ldi  r16, (0<<WDE)
  out  WDTCR, r16
  ret
```

C Code Example

```
void WDT_off(void)
{
  /* Write logical one to WDCE and WDE */
  WDTCR = (1<<WDCE) | (1<<WDE);
  /* Turn off WDT */
  WDTCR = 0x00;
}
```

.evel 1 (WDTON Fuse rammed)

In this mode, the Watchdog Timer is initially disabled, but can be enabled by writing the WDE bit to 1 without any restriction. A timed sequence is needed when changing the Watchdog Time-out period or disabling an enabled watch dog timer. To disable an enabled Watchdog Timer and/or changing the Watchdog Time-out, the following procedure must be followed:

1. In the same operation, write a logic one to WDCE and WDE. A logic one must be written to WDE regardless of the previous value of the WDE bit.
2. Within the next four clock cycles, in the same operation, write the WDE and WDP bits as desired, but with the WDCE bit cleared.

_evel 2 (WDTON Fuse nmed)

In this mode, the Watchdog Timer is always enabled, and the WDE bit will always read as one. A timed sequence is needed when changing the Watchdog Time-out period. To change the Watchdog Time-out, the following procedure must be followed:

1. In the same operation, write a logical one to WDCE and WDE. Even though the WDE always is set, the WDE must be written to one to start the timed sequence.

Within the next four clock cycles, in the same operation, write the WDP bits as desired, but with the WDCE bit cleared. The value written to the WDE bit is irrelevant.

errupts

This section describes the specifics of the interrupt handling performed by the ATmega8. For a general explanation of the AVR interrupt handling, refer to "Reset and Interrupt Handling" on page 11.

rrupt Vectors in nega8

Table 18. Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	0x000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, and Watchdog Reset
2	0x001	INT0	External Interrupt Request 0
3	0x002	INT1	External Interrupt Request 1
4	0x003	TIMER2 COMP	Timer/Counter2 Compare Match
5	0x004	TIMER2 OVF	Timer/Counter2 Overflow
6	0x005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	0x006	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	0x007	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	0x008	TIMER1 OVF	Timer/Counter1 Overflow
10	0x009	TIMER0 OVF	Timer/Counter0 Overflow
11	0x00A	SPI, STC	Serial Transfer Complete
12	0x00B	USART, RXC	USART, Rx Complete
13	0x00C	USART, UDRE	USART Data Register Empty
14	0x00D	USART, TXC	USART, Tx Complete
15	0x00E	ADC	ADC Conversion Complete
16	0x00F	EE_RDY	EEPROM Ready
17	0x010	ANA_COMP	Analog Comparator
18	0x011	TWI	Two-wire Serial Interface
19	0x012	SPM_RDY	Store Program Memory Ready

Notes: 1. When the BOOTRST fuse is programmed, the device will jump to the Boot Loader address at reset, see "Boot Loader Support – Read-While-Write Self-Programming" on page 202.

2. When the IVSEL bit in GICR is set, interrupt vectors will be moved to the start of the boot Flash section. The address of each interrupt vector will then be the address in this table added to the start address of the boot Flash section.

Table 19 shows reset and interrupt vectors placement for the various combinations of BOOTRST and IVSEL settings. If the program never enables an interrupt source, the interrupt vectors are not used, and regular program code can be placed at these locations. This is also the case if the Reset Vector is in the Application section while the interrupt vectors are in the boot section or vice versa.

Table 19. Reset and Interrupt Vectors Placement

BOOTRST[1]	IVSEL	Reset Address	Interrupt Vectors Start Address
0	0	0x000	0x001
0	1	0x000	Boot Reset Address + 0x001
1	0	Boot Reset Address	0x001
1	1	Boot Reset Address	Boot Reset Address + 0x001

Note: 1. The Boot Reset Address is shown in Table 81 on page 213. For the BOOTRST fuse "1" means unprogrammed while "0" means programmed.

The most typical and general program setup for the Reset and Interrupt Vector Addresses in ATmega8 is:

```
address Labels  Code                    Comments
$000            rjmp    RESET           ; Reset Handler
$001            rjmp    EXT_INT0        ; IRQ0 Handler
$002            rjmp    EXT_INT1        ; IRQ1 Handler
$003            rjmp    TIM2_COMP       ; Timer2 Compare Handler
$004            rjmp    TIM2_OVF        ; Timer2 Overflow Handler
$005            rjmp    TIM1_CAPT       ; Timer1 Capture Handler
$006            rjmp    TIM1_COMPA      ; Timer1 CompareA Handler
$007            rjmp    TIM1_COMPB      ; Timer1 CompareB Handler
$008            rjmp    TIM1_OVF        ; Timer1 Overflow Handler
$009            rjmp    TIM0_OVF        ; Timer0 Overflow Handler
$00a            rjmp    SPI_STC         ; SPI Transfer Complete Handler
$00b            rjmp    USART_RXC       ; USART RX Complete Handler
$00c            rjmp    USART_UDRE      ; UDR Empty Handler
$00d            rjmp    USART_TXC       ; USART TX Complete Handler
$00e            rjmp    ADC             ; ADC Conversion Complete Handler
$00f            rjmp    EE_RDY          ; EEPROM Ready Handler
$010            rjmp    ANA_COMP        ; Analog Comparator Handler
$011            rjmp    TWSI            ; Two-wire Serial Interface Handler
$012            rjmp    SPM_RDY         ; Store Program Memory Ready Handler
;
$013    RESET:  ldi         r16,high(RAMEND) ; Main program start
$014            out     SPH,r16         ; Set stack pointer to top of RAM
$015            ldi     r16,low(RAMEND)
$016            out     SPL,r16
$017            sei                     ; Enable interrupts
$018            <instr>  xxx
...             ...      ...
```

When the BOOTRST fuse is unprogrammed, the boot section size set to 2K bytes and the IVSEL bit in the GICR register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address Labels  Code                    Comments
$000            rjmp    RESET           ; Reset handler
;
$001    RESET:  ldi     r16,high(RAMEND) ; Main program start
$002            out     SPH,r16         ; Set stack pointer to top of RAM
$003            ldi     r16,low(RAMEND)
$004            out     SPL,r16
$005            sei                     ; Enable interrupts
$006            <instr>  xxx
;
.org $c01
$c01            rjmp    EXT_INT0        ; IRQ0 Handler
$c02            rjmp    EXT_INT1        ; IRQ1 Handler
...             ...     ... ;
$c12            rjmp    SPM_RDY         ; Store Program Memory Ready Handler
```

When the BOOTRST fuse is programmed and the boot section size set to 2K bytes, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address Labels  Code                    Comments
.org $001
$001            rjmp    EXT_INT0        ; IRQ0 Handler
$002            rjmp    EXT_INT1        ; IRQ1 Handler
...             ...     ...             ;
$012            rjmp    SPM_RDY         ; Store Program Memory Ready Handler
;
.org $c00
$c00            rjmp    RESET           ; Reset handler
;
$c01    RESET:  ldi     r16,high(RAMEND); Main program start
$c02            out     SPH,r16         ; Set stack pointer to top of RAM
$c03            ldi     r16,low(RAMEND)
$c04            out     SPL,r16
$c05            sei                     ; Enable interrupts
$c06            <instr>  xxx
```

When the BOOTRST fuse is programmed, the boot section size set to 2K bytes, and the IVSEL bit in the GICR register is set before any interrupts are enabled, the most typical and general program setup for the Reset and Interrupt Vector Addresses is:

```
Address Labels     Code                 Comments
;
.org $c00
$c00            rjmp    RESET           ; Reset handler
$c01            rjmp    EXT_INT0        ; IRQ0 Handler
$c02            rjmp    EXT_INT1        ; IRQ1 Handler
...             ...     ... ;
$c12            rjmp    SPM_RDY         ; Store Program Memory Ready Handler
$c13    RESET:  ldi         r16,high(RAMEND); Main program start
$c14            out     SPH,r16         ; Set stack pointer to top of RAM
$c15            ldi     r16,low(RAMEND)
$c16            out     SPL,r16
$c17            sei                     ; Enable interrupts
$c18            <instr>  xxx
```

ng Interrupts Between ication and Boot Space

The General Interrupt Control Register controls the placement of the interrupt vector table.

ral Interrupt Control ster – GICR

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	–	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 1 – IVSEL: Interrupt Vector Select**

When the IVSEL bit is cleared (zero), the interrupt vectors are placed at the start of the Flash memory. When this bit is set (one), the interrupt vectors are moved to the beginning of the Boot Loader section of the Flash. The actual address of the start of the boot Flash section is determined by the BOOTSZ fuses. Refer to the section "Boot Loader Support – Read-While-Write Self-Programming" on page 202 for details. To avoid unintentional changes of interrupt vector tables, a special write procedure must be followed to change the IVSEL bit:

1. Write the Interrupt Vector Change Enable (IVCE) bit to one.
2. Within four cycles, write the desired value to IVSEL while writing a zero to IVCE.

Interrupts will automatically be disabled while this sequence is executed. Interrupts are disabled in the cycle IVCE is set, and they remain disabled until after the instruction following the write to IVSEL. If IVSEL is not written, interrupts remain disabled for four cycles. The I-bit in the Status Register is unaffected by the automatic disabling.

Note: If interrupt vectors are placed in the Boot Loader section and Boot Lock bit BLB02 is programmed, interrupts are disabled while executing from the Application section. If interrupt vectors are placed in the Application section and Boot Lock bit BLB12 is programed, interrupts are disabled while executing from the Boot Loader section. Refer to the section "Boot Loader Support – Read-While-Write Self-Programming" on page 202 for details on Boot Lock bits.

- **Bit 0 – IVCE: Interrupt Vector Change Enable**

The IVCE bit must be written to logic one to enable change of the IVSEL bit. IVCE is cleared by hardware four cycles after it is written or when IVSEL is written. Setting the

IVCE bit will disable interrupts, as explained in the IVSEL description above. See Code Example below.

Assembly Code Example

```
Move_interrupts:
  ; Enable change of interrupt vectors
  ldi  r16, (1<<IVCE)
  out  GICR, r16
  ; Move interrupts to boot Flash section
  ldi  r16, (1<<IVSEL)
  out  GICR, r16
  ret
```

C Code Example

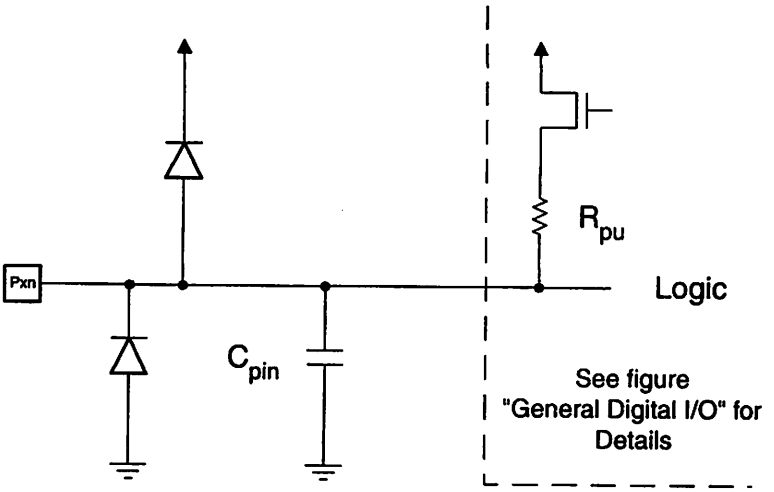
```
void Move_interrupts(void)
{
  /* Enable change of interrupt vectors */
  GICR = (1<<IVCE);
  /* Move interrupts to boot Flash section */
  GICR = (1<<IVSEL);
}
```

Ports

›duction

All AVR ports have true Read-Modify-Write functionality when used as general digital I/O ports. This means that the direction of one port pin can be changed without unintentionally changing the direction of any other pin with the SBI and CBI instructions. The same applies when changing drive value (if configured as output) or enabling/disabling of pull-up resistors (if configured as input). Each output buffer has symmetrical drive characteristics with both high sink and source capability. The pin driver is strong enough to drive LED displays directly. All port pins have individually selectable pull-up resistors with a supply-voltage invariant resistance. All I/O pins have protection diodes to both V_{CC} and Ground as indicated in Figure 21. Refer to "Electrical Characteristics" on page 232 for a complete list of parameters.

Figure 21. I/O Pin Equivalent Schematic



All registers and bit references in this section are written in general form. A lower case "x" represents the numbering letter for the port, and a lower case "n" represents the bit number. However, when using the register or bit defines in a program, the precise form must be used (i.e., PORTB3 for bit 3 in Port B, here documented generally as PORTxn). The physical I/O registers and bit locations are listed in "Register Description for I/O Ports" on page 61.

Three I/O memory address locations are allocated for each port, one each for the Data Register – PORTx, Data Direction Register – DDRx, and the Port Input Pins – PINx. The Port Input Pins I/O location is read only, while the Data Register and the Data Direction Register are read/write. In addition, the Pull-up Disable – PUD bit in SFIOR disables the pull-up function for all pins in all ports when set.

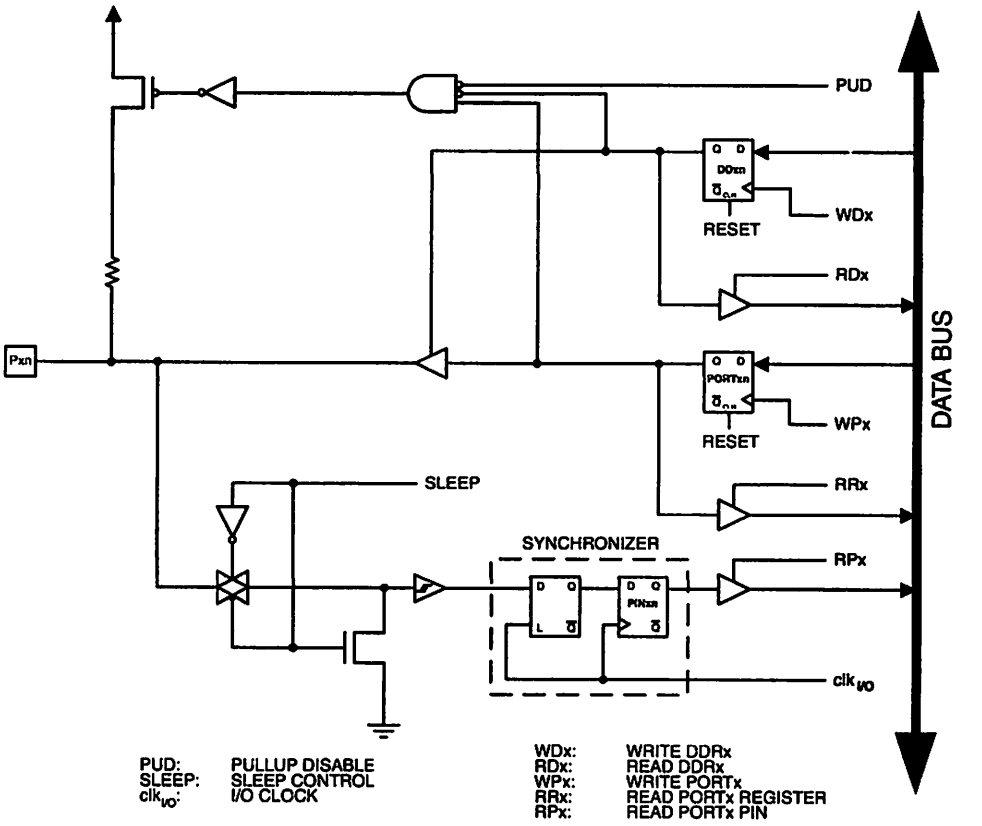
Using the I/O port as General Digital I/O is described in "Ports as General Digital I/O" on page 48. Most port pins are multiplexed with alternate functions for the peripheral features on the device. How each alternate function interferes with the port pin is described in "Alternate Port Functions" on page 52. Refer to the individual module sections for a full description of the alternate functions.

Note that enabling the alternate function of some of the port pins does not affect the use of the other pins in the port as general digital I/O.

ts as General Digital

The ports are bi-directional I/O ports with optional internal pull-ups. Figure 22 shows a functional description of one I/O port pin, here generically called Pxn.

Figure 22. General Digital I/O(1)



Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports.

figuring the Pin

Each port pin consists of 3 Register bits: DDxn, PORTxn, and PINxn. As shown in "Register Description for I/O Ports" on page 61, the DDxn bits are accessed at the DDRx I/O address, the PORTxn bits at the PORTx I/O address, and the PINxn bits at the PINx I/O address.

The DDxn bit in the DDRx register selects the direction of this pin. If DDxn is written logic one, Pxn is configured as an output pin. If DDxn is written logic zero, Pxn is configured as an input pin.

If PORTxn is written logic one when the pin is configured as an input pin, the pull-up resistor is activated. To switch the pull-up resistor off, PORTxn has to be written logic zero or the pin has to be configured as an output pin. The port pins are tri-stated when a reset condition becomes active, even if no clocks are running.

If PORTxn is written logic one when the pin is configured as an output pin, the port pin is driven high (one). If PORTxn is written logic zero when the pin is configured as an output pin, the port pin is driven low (zero).

When switching between tri-state ({DDxn, PORTxn} = 0b00) and output high ({DDxn, PORTxn} = 0b11), an intermediate state with either pull-up enabled ({DDxn, PORTxn} = 0b01) or output low ({DDxn, PORTxn} = 0b10) must occur. Normally, the pull-up enabled state is fully acceptable, as a high-impedant environment will not notice the difference between a strong high driver and a pull-up. If this is not the case, the PUD bit in the SFIOR register can be set to disable all pull-ups in all ports.

Switching between input with pull-up and output low generates the same problem. The user must use either the tri-state ({DDxn, PORTxn} = 0b00) or the output high state ({DDxn, PORTxn} = 0b10) as an intermediate step.

Table 20 summarizes the control signals for the pin value.

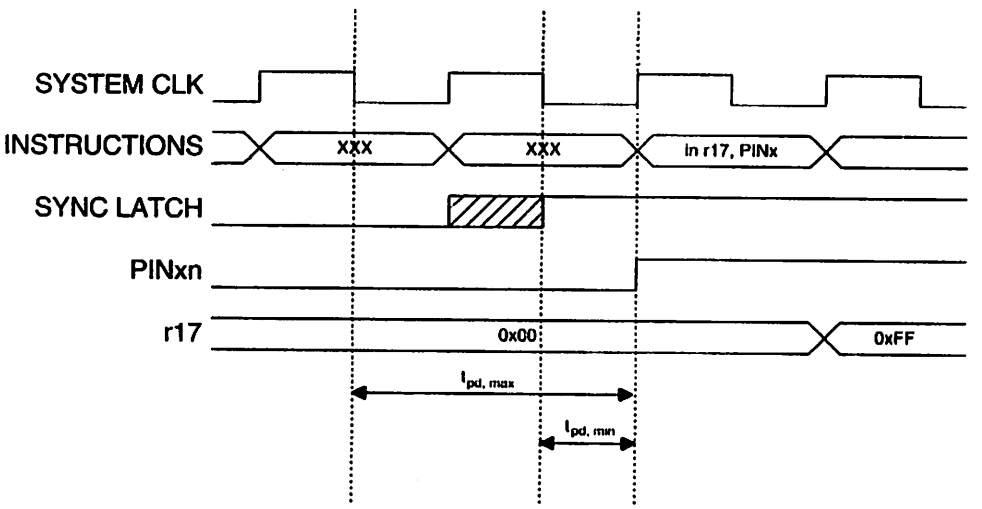
Table 20. Port Pin Configurations

DDxn	PORTxn	PUD (in SFIOR)	I/O	Pull-up	Comment
0	0	X	Input	No	Tri-state (Hi-Z)
0	1	0	Input	Yes	Pxn will source current if external pulled low.
0	1	1	Input	No	Tri-state (Hi-Z)
1	0	X	Output	No	Output Low (Sink)
1	1	X	Output	No	Output High (Source)

ding the Pin Value

Independent of the setting of Data Direction bit DDxn, the port pin can be read through the PINxn Register Bit. As shown in Figure 22, the PINxn Register bit and the preceding latch constitute a synchronizer. This is needed to avoid metastability if the physical pin changes value near the edge of the internal clock, but it also introduces a delay. Figure 23 shows a timing diagram of the synchronization when reading an externally applied pin value. The maximum and minimum propagation delays are denoted $t_{pd,max}$ and $t_{pd,min}$, respectively.

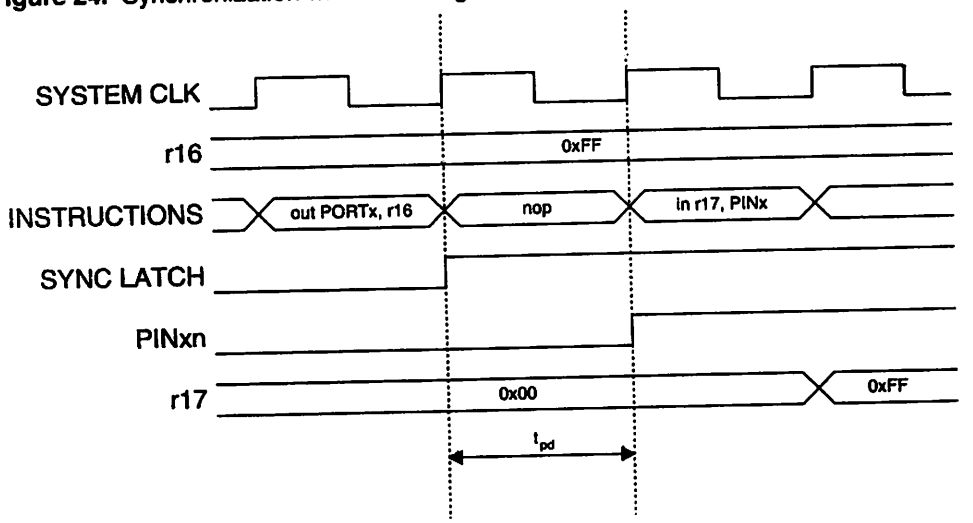
Figure 23. Synchronization when Reading an Externally Applied Pin Value



Consider the clock period starting shortly *after* the first falling edge of the system clock. The latch is closed when the clock is low, and goes transparent when the clock is high, as indicated by the shaded region of the "SYNC LATCH" signal. The signal value is latched when the system clock goes low. It is clocked into the PINxn register at the succeeding positive clock edge. As indicated by the two arrows $t_{pd,max}$ and $t_{pd,min}$, a single signal transition on the pin will be delayed between ½ and 1-½ system clock period depending upon the time of assertion.

When reading back a software assigned pin value, a *nop* instruction must be inserted as indicated in Figure 24. The *out* instruction sets the "SYNC LATCH" signal at the positive edge of the clock. In this case, the delay t_{pd} through the synchronizer is 1 system clock period.

Figure 24. Synchronization when Reading a Software Assigned Pin Value



The following code example shows how to set port B pins 0 and 1 high, 2 and 3 low, and define the port pins from 4 to 7 as input with pull-ups assigned to port pins 6 and 7. The resulting pin values are read back again, but as previously discussed, a *nop* instruction is included to be able to read back the value recently assigned to some of the pins.

Assembly Code Example[1]

```
...
; Define pull-ups and set outputs high
; Define directions for port pins
ldi   r16,(1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0)
ldi   r17,(1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0)
out   PORTB,r16
out   DDRB,r17
; Insert nop for synchronization
nop
; Read port pins
in    r16,PINB
...
```

C Code Example[1]

```
unsigned char i;
...
/* Define pull-ups and set outputs high */
/* Define directions for port pins */
PORTB = (1<<PB7)|(1<<PB6)|(1<<PB1)|(1<<PB0);
DDRB = (1<<DDB3)|(1<<DDB2)|(1<<DDB1)|(1<<DDB0);
/* Insert nop for synchronization*/
_NOP();
/* Read port pins */
i = PINB;
...
```

Note: 1. For the assembly program, two temporary registers are used to minimize the time from pull-ups are set on pins 0, 1, 6, and 7, until the direction bits are correctly set, defining bit 2 and 3 as low and redefining bits 0 and 1 as strong high drivers.

al Input Enable and Sleep es

As shown in Figure 22, the digital input signal can be clamped to ground at the input of the Schmitt-trigger. The signal denoted SLEEP in the figure, is set by the MCU Sleep Controller in Power-down mode, Power-save mode, Standby mode, and Extended Standby mode to avoid high power consumption if some input signals are left floating, or have an analog signal level close to $V_{CC}/2$.

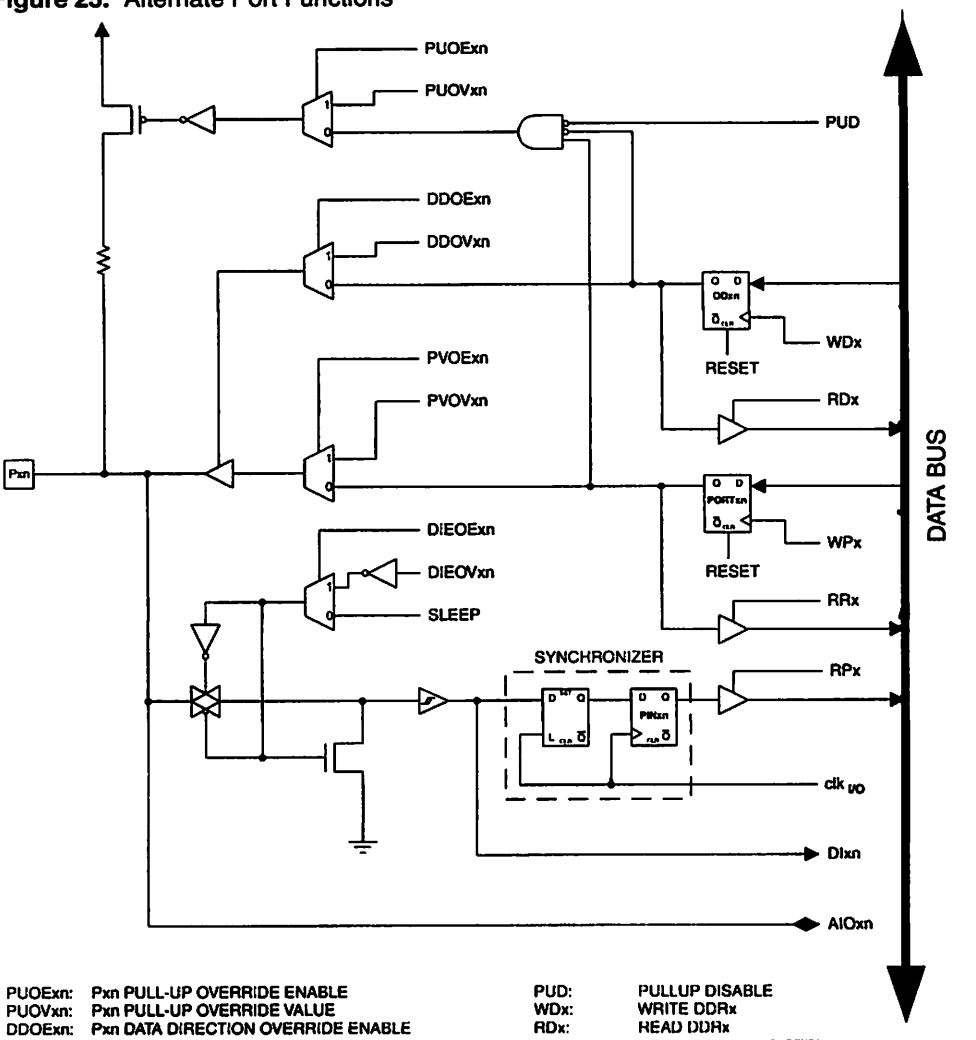
SLEEP is overridden for port pins enabled as External Interrupt pins. If the External Interrupt Request is not enabled, SLEEP is active also for these pins. SLEEP is also overridden by various other alternate functions as described in "Alternate Port Functions" on page 52.

If a logic high level ("one") is present on an Asynchronous External Interrupt pin configured as "Interrupt on Any Logic Change on Pin" while the external interrupt is *not* enabled, the corresponding External Interrupt Flag will be set when resuming from the above mentioned sleep modes, as the clamping in these Sleep modes produces the requested logic change.

rnate Port Functions

Most port pins have alternate functions in addition to being general digital I/Os. Figure 25 shows how the port pin control signals from the simplified Figure 22 can be overridden by alternate functions. The overriding signals may not be present in all port pins, but the figure serves as a generic description applicable to all port pins in the AVR microcontroller family.

Figure 25. Alternate Port Functions[1]



PUOExn:	Pxn PULL-UP OVERRIDE ENABLE	PUD:	PULLUP DISABLE
PUOVxn:	Pxn PULL-UP OVERRIDE VALUE	WDx:	WRITE DDRx
DDOExn:	Pxn DATA DIRECTION OVERRIDE ENABLE	RDx:	READ DDRx
DDOVxn:	Pxn DATA DIRECTION OVERRIDE VALUE	RRx:	HEAD PORTx REGISTER
PVOExn:	Pxn PORT VALUE OVERRIDE ENABLE	WPx:	WRITE PORTx
PVOVxn:	Pxn PORT VALUE OVERRIDE VALUE	RPx:	READ PORTx PIN
DIEOExn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE ENABLE	$clk_{I/O}$:	I/O CLOCK
DIEOVxn:	Pxn DIGITAL INPUT-ENABLE OVERRIDE VALUE	DIxn:	DIGITAL INPUT PIN n ON PORTx
SLEEP:	SLEEP CONTROL	AIOxn:	ANALOG INPUT/OUTPUT PIN n ON PORTx

Note: 1. WPx, WDx, RRx, RPx, and RDx are common to all pins within the same port. $clk_{I/O}$, SLEEP, and PUD are common to all ports. All other signals are unique for each pin.

Table 21 summarizes the function of the overriding signals. The pin and port indexes from Figure 25 are not shown in the succeeding tables. The overriding signals are generated internally in the modules having the alternate function.

Table 21. Generic Description of Overriding Signals for Alternate Functions

Signal Name	Full Name	Description
PUOE	Pull-up Override Enable	If this signal is set, the pull-up enable is controlled by the PUOV signal. If this signal is cleared, the pull-up is enabled when {DDxn, PORTxn, PUD} = 0b010.
PUOV	Pull-up Override Value	If PUOE is set, the pull-up is enabled/disabled when PUOV is set/cleared, regardless of the setting of the DDxn, PORTxn, and PUD register bits.
DDOE	Data Direction Override Enable	If this signal is set, the Output Driver Enable is controlled by the DDOV signal. If this signal is cleared, the Output driver is enabled by the DDxn register bit.
DDOV	Data Direction Override Value	If DDOE is set, the Output Driver is enabled/disabled when DDOV is set/cleared, regardless of the setting of the DDxn register bit.
PVOE	Port Value Override Enable	If this signal is set and the Output Driver is enabled, the port value is controlled by the PVOV signal. If PVOE is cleared, and the Output Driver is enabled, the port Value is controlled by the PORTxn register bit.
PVOV	Port Value Override Value	If PVOE is set, the port value is set to PVOV, regardless of the setting of the PORTxn register bit.
DIEOE	Digital Input Enable Override Enable	If this bit is set, the Digital Input Enable is controlled by the DIEOV signal. If this signal is cleared, the Digital Input Enable is determined by MCU-state (Normal mode, Sleep modes).
DIEOV	Digital Input Enable Override Value	If DIEOE is set, the Digital Input is enabled/disabled when DIEOV is set/cleared, regardless of the MCU state (Normal mode, Sleep modes).
DI	Digital Input	This is the Digital Input to alternate functions. In the figure, the signal is connected to the output of the schmitt trigger but before the synchronizer. Unless the Digital Input is used as a clock source, the module with the alternate function will use its own synchronizer.
AIO	Analog Input/output	This is the Analog Input/output to/from alternate functions. The signal is connected directly to the pad, and can be used bidirectionally.

The following subsections shortly describe the alternate functions for each port, and relate the overriding signals to the alternate function. Refer to the alternate function description for further details.

l Function IO Register –

Bit	7	6	5	4	3	2	1	0	
				ADHSM	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R	R	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 2 – PUD: Pull-up Disable**

When this bit is written to one, the pull-ups in the I/O ports are disabled even if the DDxn and PORTxn registers are configured to enable the pull-ups ({DDxn, PORTxn} = 0b01). See "Configuring the Pin" on page 48 for more details about this feature.

ıte Functions Of Port B

The Port B pins with alternate functions are shown in Table 22.

Table 22. Port B Pins Alternate Functions

Port Pin	Alternate Functions
PB7	XTAL2 (Chip Clock Oscillator pin 2) TOSC2 (Timer Oscillator pin 2)
PB6	XTAL1 (Chip Clock Oscillator pin 1 or External clock input) TOSC1 (Timer Oscillator pin 1)
PB5	SCK (SPI Bus Master clock Input)
PB4	MISO (SPI Bus Master Input/Slave Output)
PB3	MOSI (SPI Bus Master Output/Slave Input) OC2 (Timer/Counter2 Output Compare Match Output)
PB2	$\overline{SS}$ (SPI Bus Master Slave select) OC1B (Timer/Counter1 Output Compare Match B Output)
PB1	OC1A (Timer/Counter1 Output Compare Match A Output)
PB0	ICP (Timer/Counter1 Input Capture Input)

The alternate pin configuration is as follows:

- **XTAL2/TOSC2 – Port B, Bit 7**

XTAL2: Chip clock oscillator pin 2. Used as clock pin for all chip clock sources except internal calibrated RC Oscillator and external clock. When used as a clock pin, the pin can not be used as an I/O pin. When using internal calibrated RC Oscillator or external clock as chip clock sources, PB7 functions as an ordinary I/O pin.

TOSC2: Timer Oscillator pin 2. Used only if internal calibrated RC Oscillator is selected as chip clock source, and the asynchronous timer is enabled by the correct setting in ASSR. When the AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter2, pin PB7 is disconnected from the port, and becomes the inverting output of the oscillator amplifier. In this mode, a crystal oscillator is connected to this pin, and the pin cannot be used as an I/O pin.

If PB7 is used as a clock pin, DDB7, PORTB7 and PINB7 will all read 0.

- **XTAL1/TOSC1 – Port B, Bit 6**

XTAL1: Chip clock oscillator pin 1. Used for all chip clock sources except internal calibrated RC Oscillator. When used as a clock pin, the pin can not be used as an I/O pin. When using internal calibrated RC Oscillator as chip clock source, PB6 functions as an ordinary I/O pin.

TOSC1: Timer Oscillator pin 1. Used only if internal calibrated RC Oscillator is selected as chip clock source, and the asynchronous timer is enabled by the correct setting in ASSR. When the AS2 bit in ASSR is set (one) to enable asynchronous clocking of Timer/Counter1, pin PB6 is disconnected from the port, and becomes the input of the inverting oscillator amplifier. In this mode, a crystal oscillator is connected to this pin, and the pin can not be used as an I/O pin.

If PB6 is used as a clock pin, DDB6, PORTB6 and PINB6 will all read 0.

- **SCK – Port B, Bit 5**

SCK: Master clock output, slave clock input pin for SPI channel. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB5. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB5. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB5 bit.

- **MISO – Port B, Bit 4**

MISO: Master data Input, Slave data Output pin for SPI channel. When the SPI is enabled as a master, this pin is configured as an input regardless of the setting of DDB4. When the SPI is enabled as a slave, the data direction of this pin is controlled by DDB4. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB4 bit.

- **MOSI – Port B, Bit 3**

MOSI: SPI Master data output, slave data input for SPI channel. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB3. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB3. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB3 bit.

- **$\overline{SS}$/OC1B – Port B, Bit 2**

$\overline{SS}$: Slave Select input. When the SPI is enabled as a slave, this pin is configured as an input regardless of the setting of DDB2. As a slave, the SPI is activated when this pin is driven low. When the SPI is enabled as a master, the data direction of this pin is controlled by DDB2. When the pin is forced by the SPI to be an input, the pull-up can still be controlled by the PORTB2 bit.

OC1B, Output compare match output: The PB2 pin can serve as an external output for the Timer/Counter1 compare match B. The PB2 pin has to be configured as an output (DDB2 set (one)) to serve this function. The OC1B pin is also the output pin for the PWM mode timer function.

- **OC1A – Port B, Bit 1**

OC1A, Output compare match output: The PB1 pin can serve as an external output for the Timer/Counter1 compare match A. The PB1 pin has to be configured as an output (DDB1 set (one)) to serve this function. The OC1A pin is also the output pin for the PWM mode timer function.

- **ICP – Port B, Bit 0**

ICP – Input Capture Pin: The PB0 pin can act as an input capture pin for Timer/Counter1.

Table 23 and Table 24 relate the alternate functions of Port B to the overriding signals shown in Figure 25 on page 52. SPI MSTR INPUT and SPI SLAVE OUTPUT constitute the MISO signal, while MOSI is divided into SPI MSTR OUTPUT and SPI SLAVE INPUT.

Table 23. Overriding Signals for Alternate Functions in PB7..PB4

Signal Name	PB7/XTAL2/ TOSC2	PB6/XTAL1/ TOSC1[1]	PB5/SCK	PB4/MISO
PUOE	INTR[1]C&~AS2	INTRC&~AS2	SPE • $\overline{\text{MSTR}}$	SPE • MSTR
PUO	0	0	PORTB5 • $\overline{\text{PUD}}$	PORTB4 • $\overline{\text{PUD}}$
DDOE	INTRC&AS2	INTRC&~AS2	SPE • $\overline{\text{MSTR}}$	SPE • MSTR
OO	0	0	0	0
PVOE	0	0	SPE • MSTR	SPE • $\overline{\text{MSTR}}$
PVOV	0	0	SCK OUTPUT	SPI SLAVE OUTPUT
DIEOE	INTRC&~AS2	INTRC&~AS2	0	0
DIEOV	0	0	0	0
DI	–	–	SCK INPUT	SPI MSTR INPUT
AIO	Oscillator Output	Oscillator/Clock Input	–	–

Note: 1. INTRC means that the internal RC Oscillator is selected (by the CKSEL fuse).

Table 24. Overriding Signals for Alternate Functions in PB3..PB0

Signal Name	PB3/MOSI/OC2	PB2/$\overline{\text{SS}}$/OC1B	PB1/OC1A	PB0/ICP
PUOE	SPE • $\overline{\text{MSTR}}$	SPE • $\overline{\text{MSTR}}$	0	0
PUO	PORTB3 • $\overline{\text{PUD}}$	PORTB2 • $\overline{\text{PUD}}$	0	0
DDOE	SPE • $\overline{\text{MSTR}}$	SPE • $\overline{\text{MSTR}}$	0	0
DDOV	0	0	0	0
PVOE	SPE • MSTR + OC2 ENABLE	OC1B ENABLE	OC1A ENABLE	0
PVOV	SPI MSTR OUTPUT + OC2	OC1B	OC1A	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	SPI SLAVE INPUT	SPI $\overline{\text{SS}}$	–	ICP INPUT
AIO	–	–	–	–

nate Functions of Port C The Port C pins with alternate functions are shown in Table 25.

Table 25. Port C Pins Alternate Functions

Port Pin	Alternate Function
PC6	$\overline{\text{RESET}}$ (Reset pin)
PC5	ADC5 (ADC Input Channel 5) SCL (Two-wire Serial Bus Clock Line)
PC4	ADC4 (ADC Input Channel 4) SDA (Two-wire Serial Bus Data Input/Output Line)
PC3	ADC3 (ADC Input Channel 3)
PC2	ADC2 (ADC Input Channel 2)
PC1	ADC1 (ADC Input Channel 1)
PC0	ADC0 (ADC Input Channel 0)

The alternate pin configuration is as follows:

- **$\overline{\text{RESET}}$ – Port C, Bit 6**

$\overline{\text{RESET}}$, Reset pin: When the RSTDISBL fuse is set, this pin functions as a normal I/O pin, and the part will have to rely on Power-on Reset and Brown-out Reset as its reset sources. When the RSTDISBL fuse is cleared, the reset circuitry is connected to the pin, and the pin can not be used as an I/O pin.

If PC6 is used as a reset pin, DDC6, PORTC6 and PINC6 will all read 0.

- **SCL/ADC5 – Port C, Bit 5**

SCL, Two-wire Serial Interface Clock: When the TWEN bit in TWCR is set (one) to enable the Two-wire Serial Interface, pin PC1 is disconnected from the port and becomes the Serial Clock I/O pin for the Two-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC5 can also be used as ADC input Channel 5. Note that ADC input channel 5 uses digital power.

- **SDA/ADC4 – Port C, Bit 4**

SDA, Two-wire Serial Interface Data: When the TWEN bit in TWCR is set (one) to enable the Two-wire Serial Interface, pin PC1 is disconnected from the port and becomes the Serial Data I/O pin for the Two-wire Serial Interface. In this mode, there is a spike filter on the pin to suppress spikes shorter than 50 ns on the input signal, and the pin is driven by an open drain driver with slew-rate limitation.

PC4 can also be used as ADC input Channel 4. Note that ADC input channel 4 uses digital power.

- **ADC3 – Port C, Bit 3**

PC3 can also be used as ADC input Channel 3. Note that ADC input channel 3 uses analog power.

- **ADC2 – Port C, Bit 2**

PC2 can also be used as ADC input Channel 2. Note that ADC input channel 2 uses analog power.

- **ADC1 – Port C, Bit 1**

PC1 can also be used as ADC input Channel 1. Note that ADC input channel 1 uses analog power.

- **ADC0 – Port C, Bit 0**

PC0 can also be used as ADC input Channel 0. Note that ADC input channel 0 uses analog power.

Table 26 and Table 27 relate the alternate functions of Port C to the overriding signals shown in Figure 25 on page 52.

Table 26. Overriding Signals for Alternate Functions in PC6..PC4

Signal Name	PC6/$\overline{\text{RESET}}$	PC5/SCL/ADC5	PC4/SDA/ADC4
PUOE	RSTDISBL	TWEN	TWEN
PUOV	0	PORTC5 • $\overline{\text{PUD}}$	PORTC4 • $\overline{\text{PUD}}$
DDOE	RSTDISBL	TWEN	TWEN
DDOV	0	SCL_OUT	SDA_OUT
PVOE	0	TWEN	TWEN
PVOV	0	0	0
DIEOE	RSTDISBL	0	0
DIEOV	0	0	0
DI	–	–	–
AIO	RESET INPUT	ADC5 INPUT / SCL INPUT	ADC4 INPUT / SDA INPUT

Table 27. Overriding Signals for Alternate Functions in PC3..PC0[1]

Signal Name	PC3/ADC3	PC2/ADC2	PC1/ADC1	PC0/ADC0
PUOE	0	0	0	0
PUOV	0	0	0	0
DDOE	0	0	0	0
DDOV	0	0	0	0
PVOE	0	0	0	0
PVOV	0	0	0	0
DIEOE	0	0	0	0
DIEOV	0	0	0	0
DI	–	–	–	–
AIO	ADC3 INPUT	ADC2 INPUT	ADC1 INPUT	ADC0 INPUT

Note: 1. When enabled, the Two-wire Serial Interface enables slew-rate controls on the output pins PC4 and PC5. This is not shown in the figure. In addition, spike filters are con-

nected between the AIO outputs shown in the port figure and the digital logic of the TWI module.

rnate Functions of Port D

The Port D pins with alternate functions are shown in Table 28.

Table 28. Port D Pins Alternate Functions

Port Pin	Alternate Function
PD7	AIN1 (Analog Comparator Negative Input)
PD6	AIN0 (Analog Comparator Positive Input)
PD5	T1 (Timer/Counter 1 External Counter Input)
PD4	XCK (USART External Clock Input/Output) T0 (Timer/Counter 0 External Counter Input)
PD3	INT1 (External Interrupt 1 Input)
PD2	INT0 (External Interrupt 0 Input)
PD1	TXD (USART Output Pin)
PD0	RXD (USART Input Pin)

The alternate pin configuration is as follows:

- **AIN1 – Port D, Bit 7**

AIN1, Analog Comparator Negative Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the analog comparator.

- **AIN0 – Port D, Bit 6**

AIN0, Analog Comparator Positive Input. Configure the port pin as input with the internal pull-up switched off to avoid the digital port function from interfering with the function of the analog comparator.

- **T1 – Port D, Bit 5**

T1, Timer/Counter1 counter source.

- **XCK/T0 – Port D, Bit 4**

XCK, USART external clock.

T0, Timer/Counter0 counter source.

- **INT1 – Port D, Bit 3**

INT1, External Interrupt source 1: The PD3 pin can serve as an external interrupt source.

- **INT0 – Port D, Bit 2**

INT0, External Interrupt source 0: The PD2 pin can serve as an external interrupt source.

- **TXD – Port D, Bit 1**

TXD, Transmit Data (Data output pin for the USART). When the USART transmitter is enabled, this pin is configured as an output regardless of the value of DDD1.

- **RXD – Port D, Bit 0**

RXD, Receive Data (Data input pin for the USART). When the USART receiver is enabled this pin is configured as an input regardless of the value of DDD0. When the USART forces this pin to be an input, the pull-up can still be controlled by the PORTD0 bit.

Table 29 and Table 30 relate the alternate functions of Port D to the overriding signals shown in Figure 25 on page 52.

Table 29. Overriding Signals for Alternate Functions PD7..PD4

Signal Name	PD7/AIN1	PD6/AIN0	PD5/T1	PD4/XCK/T0
PUOE	0	0	0	0
PUO	0	0	0	0
OOE	0	0	0	0
OO	0	0	0	0
PVOE	0	0	0	UMSEL
PVO	0	0	0	XCK OUTPUT
DIEOE	0	0	0	0
DIEO	0	0	0	0
DI	–	–	T1 INPUT	XCK INPUT / T0 INPUT
AIO	AIN1 INPUT	AIN0 INPUT	–	–

Table 30. Overriding Signals for Alternate Functions in PD3..PD0

Signal Name	PD3/INT1	PD2/INT0	PD1/TXD	PD0/RXD
PUOE	0	0	TXEN	RXEN
PUO	0	0	0	PORTD0 • $\overline{PUD}$
OOE	0	0	TXEN	RXEN
OO	0	0	1	0
PVOE	0	0	TXEN	0
PVO	0	0	TXD	0
DIEOE	INT1 ENABLE	INT0 ENABLE	0	0
DIEO	1	1	0	0
DI	INT1 INPUT	INT0 INPUT	–	RXD
AIO	–	–	–	–

gister Description for I/O Ports

Port B Data Register – RTB

Bit	7	6	5	4	3	2	1	0	
	PORTB7	PORTB6	PORTB5	PORTB4	PORTB3	PORTB2	PORTB1	PORTB0	PORTB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port B Data Direction ister – DDRB

Bit	7	6	5	4	3	2	1	0	
	DDB7	DDB6	DDB5	DDB4	DDB3	DDB2	DDB1	DDB0	DDRB
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port B Input Pins Address NB

Bit	7	6	5	4	3	2	1	0	
	PINB7	PINB6	PINB5	PINB4	PINB3	PINB2	PINB1	PINB0	PINB
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Port C Data Register – RTC

Bit	7	6	5	4	3	2	1	0	
	–	PORTC6	PORTC5	PORTC4	PORTC3	PORTC2	PORTC1	PORTC0	PORTC
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port C Data Direction ister – DDRC

Bit	7	6	5	4	3	2	1	0	
	–	DDC6	DDC5	DDC4	DDC3	DDC2	DDC1	DDC0	DDRC
Read/Write	R	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port C Input Pins Address NC

Bit	7	6	5	4	3	2	1	0	
	–	PINC6	PINC5	PINC4	PINC3	PINC2	PINC1	PINC0	PINC
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

Port D Data Register – RTD

Bit	7	6	5	4	3	2	1	0	
	PORTD7	PORTD6	PORTD5	PORTD4	PORTD3	PORTD2	PORTD1	PORTD0	PORTD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port D Data Direction ister – DDRD

Bit	7	6	5	4	3	2	1	0	
	DDD7	DDD6	DDD5	DDD4	DDD3	DDD2	DDD1	DDD0	DDRD
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Port D Input Pins Address ND

Bit	7	6	5	4	3	2	1	0	
	PIND7	PIND6	PIND5	PIND4	PIND3	PIND2	PIND1	PIND0	PIND
Read/Write	R	R	R	R	R	R	R	R	
Initial Value	N/A	N/A	N/A	N/A	N/A	N/A	N/A	N/A	

ternal Interrupts

The external interrupts are triggered by the INT0, and INT1 pins. Observe that, if enabled, the interrupts will trigger even if the INT0..1 pins are configured as outputs. This feature provides a way of generating a software interrupt. The external interrupts can be triggered by a falling or rising edge or a low level. This is set up as indicated in the specification for the MCU Control Register – MCUCR and MCU Control and Status Register – MCUCSR. When the external interrupt is enabled and is configured as level triggered, the interrupt will trigger as long as the pin is held low. Note that recognition of falling or rising edge interrupts on INT0 and INT1 requires the presence of an I/O clock, described in "Clock Systems and their Distribution" on page 22. Low level interrupts on INT0/INT1 are detected asynchronously. This implies that these interrupts can be used for waking the part also from sleep modes other than Idle mode. The I/O clock is halted in all sleep modes except Idle mode.

Note that if a level triggered interrupt is used for wake-up from Power-down mode, the changed level must be held for some time to wake up the MCU. This makes the MCU less sensitive to noise. The changed level is sampled twice by the Watchdog Oscillator clock. The period of the Watchdog Oscillator is 1 µs (nominal) at 5.0V and 25°C. The frequency of the Watchdog Oscillator is voltage dependent as shown in "Electrical Characteristics" on page 232. The MCU will wake up if the input has the required level during this sampling or if it is held until the end of the start-up time. The start-up time is defined by the SUT fuses as described in "System Clock and Clock Options" on page 22. If the level is sampled twice by the Watchdog Oscillator clock but disappears before the end of the start-up time, the MCU will still wake up, but no interrupt will be generated. The required level must be held long enough for the MCU to complete the wake up to trigger the level interrupt.

J Control Register – JCR

The MCU Control Register contains control bits for interrupt sense control and general MCU functions.

Bit	7	6	5	4	3	2	1	0	
	SE	SM2	SM1	SM0	ISC11	ISC10	ISC01	ISC00	MCUCR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 3, 2 – ISC11, ISC10: Interrupt Sense Control 1 Bit 1 and Bit 0**

The External Interrupt 1 is activated by the external pin INT1 if the SREG I-bit and the corresponding interrupt mask in the GICR are set. The level and edges on the external INT1 pin that activate the interrupt are defined in Table 31. The value on the INT1 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 31. Interrupt 1 Sense Control

ISC11	ISC10	Description
0	0	The low level of INT1 generates an interrupt request.
0	1	Any logical change on INT1 generates an interrupt request.
1	0	The falling edge of INT1 generates an interrupt request.
1	1	The rising edge of INT1 generates an interrupt request.

- **Bit 1, 0 – ISC01, ISC00: Interrupt Sense Control 0 Bit 1 and Bit 0**

The External Interrupt 0 is activated by the external pin INT0 if the SREG I-flag and the corresponding interrupt mask are set. The level and edges on the external INT0 pin that activate the interrupt are defined in Table 32. The value on the INT0 pin is sampled before detecting edges. If edge or toggle interrupt is selected, pulses that last longer than one clock period will generate an interrupt. Shorter pulses are not guaranteed to generate an interrupt. If low level interrupt is selected, the low level must be held until the completion of the currently executing instruction to generate an interrupt.

Table 32. Interrupt 0 Sense Control

ISC01	ISC00	Description
0	0	The low level of INT0 generates an interrupt request.
0	1	Any logical change on INT0 generates an interrupt request.
1	0	The falling edge of INT0 generates an interrupt request.
1	1	The rising edge of INT0 generates an interrupt request.

eral Interrupt Control ister – GICR

Bit	7	6	5	4	3	2	1	0	
	INT1	INT0	–	–	–	–	IVSEL	IVCE	GICR
Read/Write	R/W	R/W	R	R	R	R	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – INT1: External Interrupt Request 1 Enable**

When the INT1 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control1 bits 1/0 (ISC11 and ISC10) in the MCU general Control Register (MCUCR) define whether the external interrupt is activated on rising and/or falling edge of the INT1 pin or level sensed. Activity on the pin will cause an interrupt request even if INT1 is configured as an output. The corresponding interrupt of External Interrupt Request 1 is executed from the INT1 interrupt vector.

- **Bit 6 – INT0: External Interrupt Request 0 Enable**

When the INT0 bit is set (one) and the I-bit in the Status Register (SREG) is set (one), the external pin interrupt is enabled. The Interrupt Sense Control0 bits 1/0 (ISC01 and ISC00) in the MCU general Control Register (MCUCR) define whether the external interrupt is activated on rising and/or falling edge of the INT0 pin or level sensed. Activity on the pin will cause an interrupt request even if INT0 is configured as an output. The corresponding interrupt of External Interrupt Request 0 is executed from the INT0 interrupt vector.

eral Interrupt Flag ister – GIFR

Bit	7	6	5	4	3	2	1	0	
	INTF1	INTF0	–	–	–	–	–	–	GIFR
Read/Write	R/W	R/W	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – INTF1: External Interrupt Flag 1**

When an event on the INT1 pin triggers an interrupt request, INTF1 becomes set (one). If the I-bit in SREG and the INT1 bit in GICR are set (one), the MCU will jump to the corresponding interrupt vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it. This flag is always cleared when INT1 is configured as a level interrupt.

- **Bit 6 – INTF0: External Interrupt Flag 0**

When an event on the INT0 pin triggers an interrupt request, INTF0 becomes set (one). If the I-bit in SREG and the INT0 bit in GICR are set (one), the MCU will jump to the corresponding interrupt vector. The flag is cleared when the interrupt routine is executed. Alternatively, the flag can be cleared by writing a logical one to it. This flag is always cleared when INT0 is configured as a level interrupt.

›it er/Counter1

The 16-bit Timer/Counter unit allows accurate program execution timing (event management), wave generation, and signal timing measurement. The main features are:

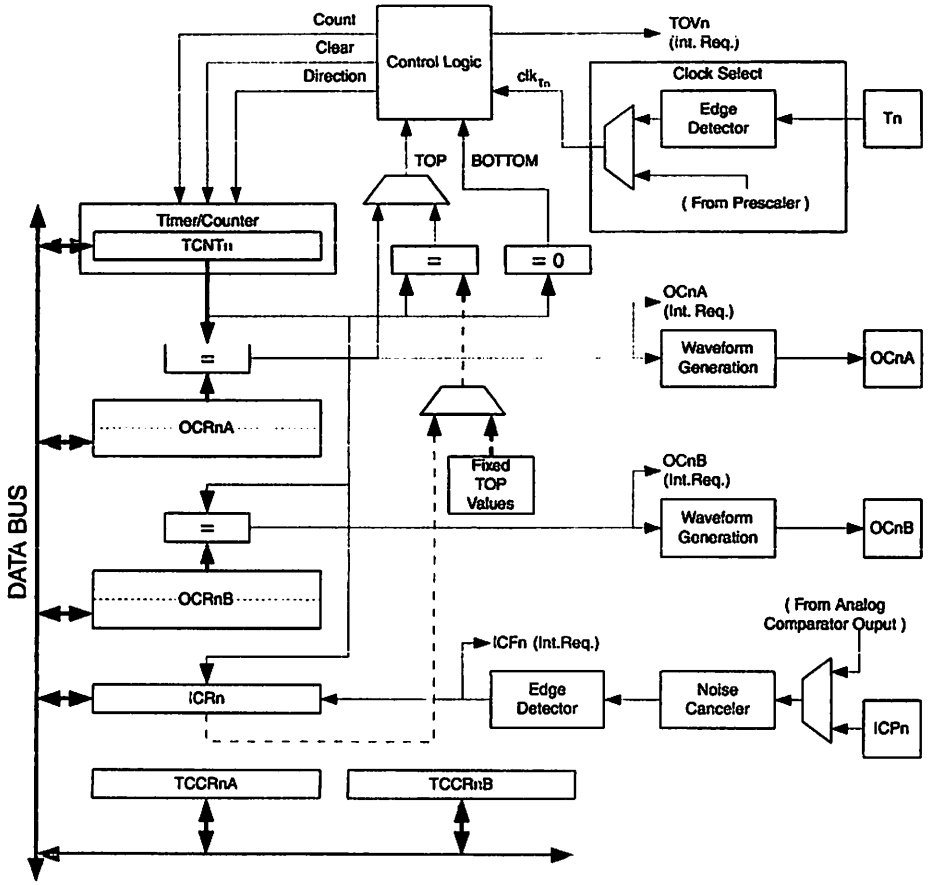
- **True 16-bit Design (i.e., allows 16-bit PWM)**
- **Two Independent Output Compare Units**
- **Double Buffered Output Compare Registers**
- **One Input Capture Unit**
- **Input Capture Noise Canceler**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch-free, Phase Correct Pulse Width Modulator (PWM)**
- **Variable PWM Period**
- **Frequency Generator**
- **External Event Counter**
- **Four Independent Interrupt Sources (TOV1, OCF1A, OCF1B, and ICF1)**

'view

Most register and bit references in this document are written in general form. A lower case "n" replaces the Timer/Counter number, and a lower case "x" replaces the output compare unit channel. However, when using the register or bit defines in a program, the precise form must be used (i.e., TCNT1 for accessing Timer/Counter1 counter value and so on). The physical I/O register and bit locations for ATmega8 are listed in the "16-bit Timer/Counter Register Description" on page 93.

A simplified block diagram of the 16-bit Timer/Counter is shown in Figure 32. CPU accessible I/O registers, including I/O bits and I/O pins, are shown in bold.

Figure 32. 16-bit Timer/Counter Block Diagram[1]



Note: 1. Refer to "Pin Configurations" on page 2, Table 22 on page 54, and Table 28 on page 59 for Timer/Counter1 pin placement and description.

isters

The *Timer/Counter* (TCNT1), *Output Compare Registers* (OCR1A/B), and *Input Capture Register* (ICR1) are all 16-bit registers. Special procedures must be followed when accessing the 16-bit registers. These procedures are described in the section "Accessing 16-bit Registers" on page 75. The *Timer/Counter Control Registers* (TCCR1A/B) are 8-bit registers and have no CPU access restrictions. Interrupt requests (abbreviated to Int.Req. in the figure) signals are all visible in the *Timer Interrupt Flag Register* (TIFR). All interrupts are individually masked with the *Timer Interrupt Mask register* (TIMSK). TIFR and TIMSK are not shown in the figure since these registers are shared by other timer units.

The Timer/Counter can be clocked internally, via the prescaler, or by an external clock source on the T1 pin. The Clock Select logic block controls which clock source and edge the Timer/Counter uses to increment (or decrement) its value. The Timer/Counter is inactive when no clock source is selected. The output from the clock select logic is referred to as the timer clock (clk_{T1}).

The double buffered Output Compare Registers (OCR1A/B) are compared with the Timer/Counter value at all time. The result of the compare can be used by the waveform generator to generate a PWM or variable frequency output on the Output Compare Pin (OC1A/B). See "Output Compare Units" on page 81. The compare match event will also

set the compare match flag (OCF1A/B) which can be used to generate an output compare interrupt request.

The Input Capture Register can capture the Timer/Counter value at a given external (edge triggered) event on either the Input Capture Pin (ICP1) or on the analog comparator pins (see "Analog Comparator" on page 186). The input capture unit includes a digital filtering unit (Noise Canceler) for reducing the chance of capturing noise spikes.

The TOP value, or maximum Timer/Counter value, can in some modes of operation be defined by either the OCR1A register, the ICR1 register, or by a set of fixed values. When using OCR1A as TOP value in a PWM mode, the OCR1A register can not be used for generating a PWM output. However, the TOP value will in this case be double buffered allowing the TOP value to be changed in run time. If a fixed TOP value is required, the ICR1 register can be used as an alternative, freeing the OCR1A to be used as PWM output.

initions

The following definitions are used extensively throughout the document:

Table 35. Definitions

BOTTOM	The counter reaches the *BOTTOM* when it becomes 0x0000.
MAX	The counter reaches its *MAX*imum when it becomes 0xFFFF (decimal 65535).
TOP	The counter reaches the *TOP* when it becomes equal to the highest value in the count sequence. The TOP value can be assigned to be one of the fixed values: 0x00FF, 0x01FF, or 0x03FF, or to the value stored in the OCR1A or ICR1 register. The assignment is dependent of the mode of operation.

ıpatibility

The 16-bit Timer/Counter has been updated and improved from previous versions of the 16-bit AVR Timer/Counter. This 16-bit Timer/Counter is fully compatible with the earlier version regarding:

- All 16-bit Timer/Counter related I/O register address locations, including timer interrupt registers.
- Bit locations inside all 16-bit Timer/Counter registers, including timer interrupt registers.
- Interrupt vectors.

The following control bits have changed name, but have same functionality and register location:

- PWM10 is changed to WGM10.
- PWM11 is changed to WGM11.
- CTC1 is changed to WGM12.

The following bits are added to the 16-bit Timer/Counter control registers:

- FOC1A and FOC1B are added to TCCR1A.
- WGM13 is added to TCCR1B.

The 16-bit Timer/Counter has improvements that will affect the compatibility in some special cases.

cessing 16-bit gisters

The TCNT1, OCR1A/B, and ICR1 are 16-bit registers that can be accessed by the AVR CPU via the 8-bit data bus. The 16-bit register must be byte accessed using two read or write operations. Each 16-bit timer has a single 8-bit register for temporary storing of the high byte of the 16-bit access. The same temporary register is shared between all 16-bit registers within each 16-bit timer. Accessing the low byte triggers the 16-bit read or write operation. When the low byte of a 16-bit register is written by the CPU, the high byte stored in the temporary register, and the low byte written are both copied into the 16-bit register in the same clock cycle. When the low byte of a 16-bit register is read by the CPU, the high byte of the 16-bit register is copied into the temporary register in the same clock cycle as the low byte is read.

Not all 16-bit accesses uses the temporary register for the high byte. Reading the OCR1A/B 16-bit registers does not involve using the temporary register.

To do a 16-bit write, the high byte must be written before the low byte. For a 16-bit read, the low byte must be read before the high byte.

The following code examples show how to access the 16-bit timer registers assuming that no interrupts updates the temporary register. The same principle can be used directly for accessing the OCR1A/B and ICR1 registers. Note that when using "C", the compiler handles the 16-bit access.

Assembly Code Example[1]

```
...
; Set TCNT1 to 0x01FF
ldi r17,0x01
ldi r16,0xFF
out TCNT1H,r17
out TCNT1L,r16
; Read TCNT1 into r17:r16
in  r16,TCNT1L
in  r17,TCNT1H
...
```

C Code Example[1]

```
unsigned int i;
...
/* Set TCNT1 to 0x01FF */
TCNT1 = 0x1FF;
/* Read TCNT1 into i */
i = TCNT1;
...
```

Note: 1. The example code assumes that the part specific header file is included.

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

It is important to notice that accessing 16-bit registers are atomic operations. If an interrupt occurs between the two instructions accessing the 16-bit register, and the interrupt code updates the temporary register by accessing the same or any other of the 16-bit timer registers, then the result of the access outside the interrupt will be corrupted. Therefore, when both the main code and the interrupt code update the temporary register, the main code must disable the interrupts during the 16-bit access.

The following code examples show how to do an atomic read of the TCNT1 register contents. Reading any of the OCR1A/B or ICR1 registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_ReadTCNT1:
  ; Save global interrupt flag
  in  r18,SREG
  ; Disable interrupts
  cli
  ; Read TCNT1 into r17:r16
  in  r16,TCNT1L
  in  r17,TCNT1H
  ; Restore global interrupt flag
  out SREG,r18
  ret
```

C Code Example[1]

```
unsigned int TIM16_ReadTCNT1( void )
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Read TCNT1 into i */
  i = TCNT1;
  /* Restore global interrupt flag */
  SREG = sreg;
  return i;
}
```

Note: 1. The example code assumes that the part specific header file is included.

The assembly code example returns the TCNT1 value in the r17:r16 register pair.

The following code examples show how to do an atomic write of the TCNT1 register contents. Writing any of the OCR1A/B or ICR1 registers can be done by using the same principle.

Assembly Code Example[1]

```
TIM16_WriteTCNT1:
  ; Save global interrupt flag
  in  r18,SREG
  ; Disable interrupts
  cli
  ; Set TCNT1 to r17:r16
  out TCNT1H,r17
  out TCNT1L,r16
  ; Restore global interrupt flag
  out SREG,r18
  ret
```

C Code Example[1]

```
void TIM16_WriteTCNT1( unsigned int i )
{
  unsigned char sreg;
  unsigned int i;
  /* Save global interrupt flag */
  sreg = SREG;
  /* Disable interrupts */
  _CLI();
  /* Set TCNT1 to i */
  TCNT1 = i;
  /* Restore global interrupt flag */
  SREG = sreg;
}
```

Note: 1. The example code assumes that the part specific header file is included.

The assembly code example requires that the r17:r16 register pair contains the value to be written to TCNT1.

sing the Temporary High e Register

If writing to more than one 16-bit register where the high byte is the same for all registers written, then the high byte only needs to be written once. However, note that the same rule of atomic operation described previously also applies in this case.

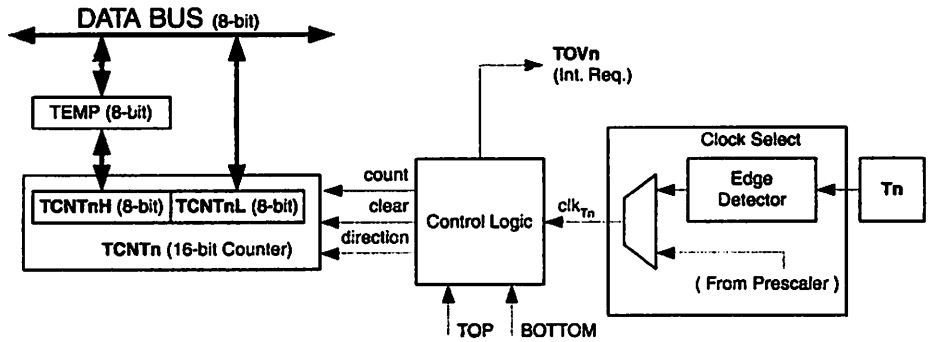
ner/Counter Clock urces

The Timer/Counter can be clocked by an internal or an external clock source. The clock source is selected by the clock select logic which is controlled by the *clock select* (CS12:0) bits located in the *Timer/Counter control register B* (TCCR1B). For details on clock sources and prescaler, see "Timer/Counter0 and Timer/Counter1 Prescalers" on page 70.

unter Unit

The main part of the 16-bit Timer/Counter is the programmable 16-bit bidirectional counter unit. Figure 33 shows a block diagram of the counter and its surroundings.

Figure 33. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT1 by 1.
direction	Select between increment and decrement.
clear	Clear TCNT1 (set all bits to zero).
clk$_{T1}$	Timer/counter clock.
TOP	Signalize that TCNT1 has reached maximum value.
BOTTOM	Signalize that TCNT1 has reached minimum value (zero).

The 16-bit counter is mapped into two 8-bit I/O memory locations: *counter high* (TCNT1H) containing the upper 8 bits of the counter, and *counter low* (TCNT1L) containing the lower 8 bits. The TCNT1H register can only be indirectly accessed by the CPU. When the CPU does an access to the TCNT1H I/O location, the CPU accesses the high byte temporary register (TEMP). The temporary register is updated with the TCNT1H value when the TCNT1L is read, and TCNT1H is updated with the temporary register value when TCNT1L is written. This allows the CPU to read or write the entire 16-bit counter value within one clock cycle via the 8-bit data bus. It is important to notice that there are special cases of writing to the TCNT1 register when the counter is counting that will give unpredictable results. The special cases are described in the sections where they are of importance.

Depending on the mode of operation used, the counter is cleared, incremented, or decremented at each *timer clock* (clk$_{T1}$). The clk$_{T1}$ can be generated from an external or internal clock source, selected by the *clock select* bits (CS12:0). When no clock source is selected (CS12:0 = 0) the timer is stopped. However, the TCNT1 value can be accessed by the CPU, independent of whether clk$_{T1}$ is present or not. A CPU write overrides (has priority over) all counter clear or count operations.

The counting sequence is determined by the setting of the *Waveform Generation mode* bits (WGM13:0) located in the *Timer/Counter control registers* A and B (TCCR1A and TCCR1B). There are close connections between how the counter behaves (counts) and how waveforms are generated on the output compare outputs OC1x. For more details about advanced counting sequences and waveform generation, see "Modes of Operation" on page 84.

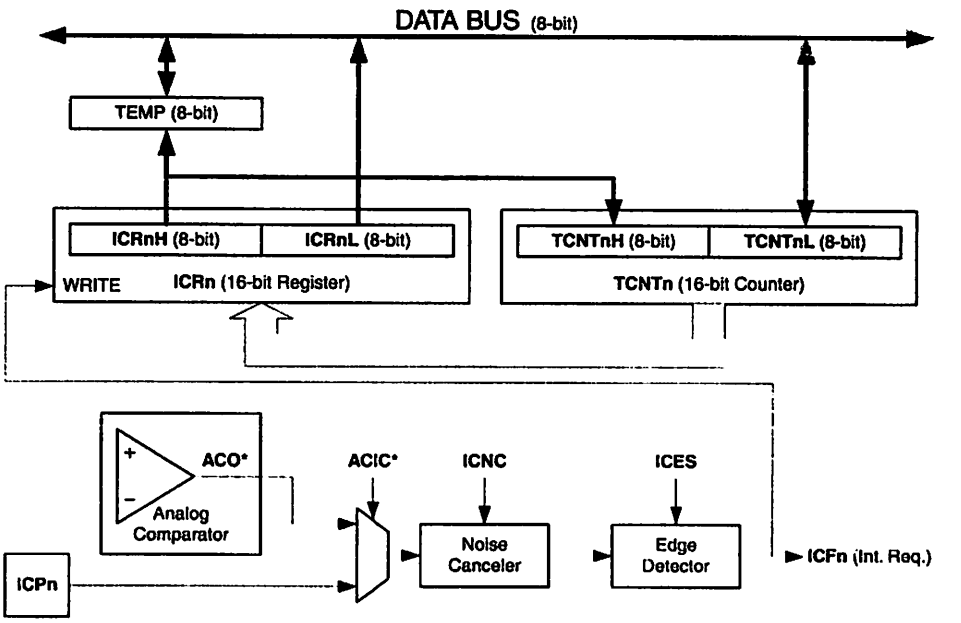
The *Timer/Counter Overflow* (TOV1) flag is set according to the mode of operation selected by the WGM13:0 bits. TOV1 can be used for generating a CPU interrupt.

it Capture Unit

The Timer/Counter incorporates an input capture unit that can capture external events and give them a time-stamp indicating time of occurrence. The external signal indicating an event, or multiple events, can be applied via the ICP1 pin or alternatively, via the analog-comparator unit. The time-stamps can then be used to calculate frequency, duty-cycle, and other features of the signal applied. Alternatively the time-stamps can be used for creating a log of the events.

The input capture unit is illustrated by the block diagram shown in Figure 34. The elements of the block diagram that are not directly a part of the input capture unit are gray shaded. The small "n" in register and bit names indicates the Timer/Counter number.

Figure 34. Input Capture Unit Block Diagram



When a change of the logic level (an event) occurs on the *input capture pin* (ICP1), alternatively on the *analog comparator output* (ACO), and this change confirms to the setting of the edge detector, a capture will be triggered. When a capture is triggered, the 16-bit value of the counter (TCNT1) is written to the *input capture register* (ICR1). The *input capture flag* (ICF1) is set at the same system clock as the TCNT1 value is copied into ICR1 register. If enabled (TICIE1 = 1), the input capture flag generates an input capture interrupt. The ICF1 flag is automatically cleared when the interrupt is executed. Alternatively the ICF1 flag can be cleared by software by writing a logical one to its I/O bit location.

Reading the 16-bit value in the *Input Capture Register* (ICR1) is done by first reading the low byte (ICR1L) and then the high byte (ICR1H). When the low byte is read the high byte is copied into the high byte temporary register (TEMP). When the CPU reads the ICR1H I/O location it will access the TEMP register.

The ICR1 register can only be written when using a waveform generation mode that utilizes the ICR1 register for defining the counter's TOP value. In these cases the *Waveform Generation mode* (WGM13:0) bits must be set before the TOP value can be written to the ICR1 register. When writing the ICR1 register the high byte must be written to the ICR1H I/O location before the low byte is written to ICR1L.

For more information on how to access the 16-bit registers refer to "Accessing 16-bit Registers" on page 75.

ut Capture Trigger Source

The main trigger source for the input capture unit is the *Input Capture Pin* (ICP1). Timer/counter 1 can alternatively use the analog comparator output as trigger source for the input capture unit. The analog comparator is selected as trigger source by setting the *Analog Comparator Input Capture* (ACIC) bit in the *Analog Comparator Control and Status Register* (ACSR). Be aware that changing trigger source can trigger a capture. The input capture flag must therefore be cleared after the change.

Both the *Input Capture Pin* (ICP1) and the *Analog Comparator Output* (ACO) inputs are sampled using the same technique as for the T1 pin (Figure 30 on page 70). The edge detector is also identical. However, when the noise canceler is enabled, additional logic is inserted before the edge detector, which increases the delay by 4 system clock cycles. Note that the input of the noise canceler and edge detector is always enabled unless the Timer/Counter is set in a waveform generation mode that uses ICR1 to define TOP.

An input capture can be triggered by software by controlling the port of the ICP1 pin.

se Canceler

The noise canceler improves noise immunity by using a simple digital filtering scheme. The noise canceler input is monitored over 4 samples, and all 4 must be equal for changing the output that in turn is used by the edge detector.

The noise canceler is enabled by setting the *Input Capture Noise Canceler* (ICNC1) bit in *Timer/Counter Control Register B* (TCCR1B). When enabled the noise canceler introduces additional four system clock cycles of delay from a change applied to the input, to the update of the ICR1 register. The noise canceler uses the system clock and is therefore not affected by the prescaler.

ng the Input Capture Unit

The main challenge when using the input capture unit is to assign enough processor capacity for handling the incoming events. The time between two events is critical. If the processor has not read the captured value in the ICR1 register before the next event occurs, the ICR1 will be overwritten with a new value. In this case the result of the capture will be incorrect.

When using the input capture interrupt, the ICR1 register should be read as early in the interrupt handler routine as possible. Even though the Input capture interrupt has relatively high priority, the maximum interrupt response time is dependent on the maximum number of clock cycles it takes to handle any of the other interrupt requests.

Using the input capture unit in any mode of operation when the TOP value (resolution) is actively changed during operation, is not recommended.

Measurement of an external signal's duty cycle requires that the trigger edge is changed after each capture. Changing the edge sensing must be done as early as possible after the ICR1 register has been read. After a change of the edge, the input capture flag

(ICF1) must be cleared by software (writing a logical one to the I/O bit location). For measuring frequency only, the clearing of the ICF1 flag is not required (if an interrupt handler is used).

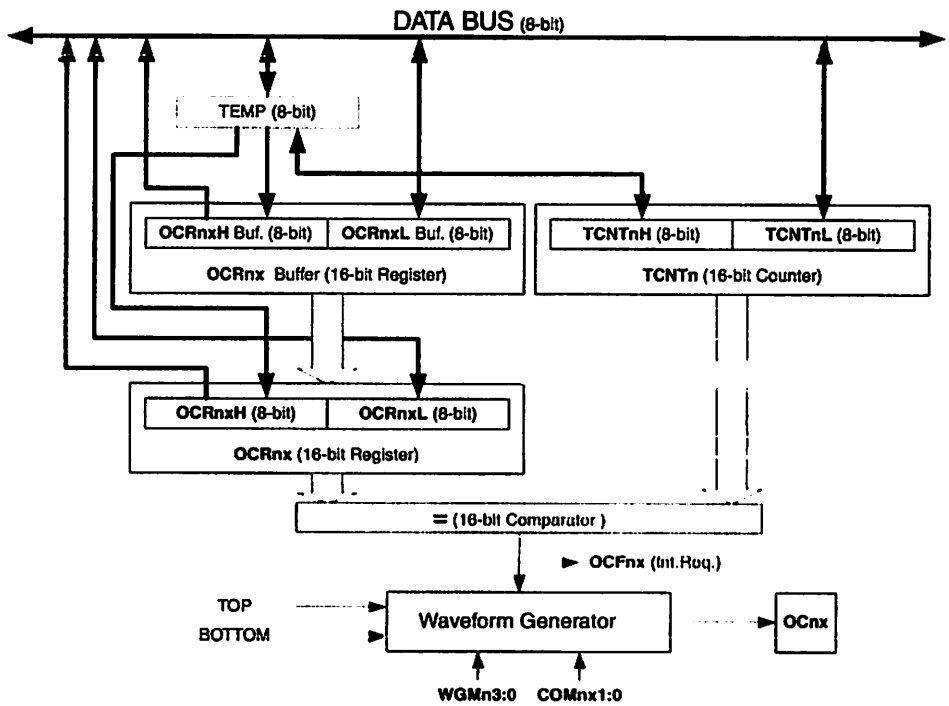
tput Compare Units

The 16-bit comparator continuously compares TCNT1 with the *Output Compare Register* (OCR1x). If TCNT equals OCR1x the comparator signals a match. A match will set the *Output Compare Flag* (OCF1x) at the next timer clock cycle. If enabled (OCIE1x = 1), the output compare flag generates an output compare interrupt. The OCF1x flag is automatically cleared when the interrupt is executed. Alternatively the OCF1x flag can be cleared by software by writing a logical one to its I/O bit location. The waveform generator uses the match signal to generate an output according to operating mode set by the *Waveform Generation mode* (WGM13:0) bits and *Compare Output mode* (COM1x1:0) bits. The TOP and BOTTOM signals are used by the waveform generator for handling the special cases of the extreme values in some modes of operation (See "Modes of Operation" on page 84.)

A special feature of output compare unit A allows it to define the Timer/Counter TOP value (i.e. counter resolution). In addition to the counter resolution, the TOP value defines the period time for waveforms generated by the waveform generator.

Figure 35 shows a block diagram of the output compare unit. The small "n" in the register and bit names indicates the device number (n = 1 for Timer/Counter 1), and the "x" indicates output compare unit (A/B). The elements of the block diagram that are not directly a part of the output compare unit are gray shaded.

Figure 35. Output Compare Unit, Block Diagram



The OCR1x register is double buffered when using any of the twelve *Pulse Width Modulation* (PWM) modes. For the normal and *Clear Timer on Compare* (CTC) modes of operation, the double buffering is disabled. The double buffering synchronizes the

update of the OCR1x compare register to either TOP or BOTTOM of the counting sequence. The synchronization prevents the occurrence of odd-length, non-symmetrical PWM pulses, thereby making the output glitch-free.

The OCR1x register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR1x buffer register, and if double buffering is disabled the CPU will access the OCR1x directly. The content of the OCR1x (buffer or compare) register is only changed by a write operation (the Timer/Counter does not update this register automatically as the TCNT1- and ICR1 register). Therefore OCR1x is not read via the high byte temporary register (TEMP). However, it is a good practice to read the low byte first as when accessing other 16-bit registers. Writing the OCR1x registers must be done via the TEMP register since the compare of all 16-bit is done continuously. The high byte (OCR1xH) has to be written first. When the high byte I/O location is written by the CPU, the TEMP register will be updated by the value written. Then when the low byte (OCR1xL) is written to the lower 8 bits, the high byte will be copied into the upper 8-bits of either the OCR1x buffer or OCR1x compare register in the same system clock cycle.

For more information of how to access the 16-bit registers refer to "Accessing 16-bit Registers" on page 75.

ce Output Compare

In non-PWM Waveform Generation modes, the match output of the comparator can be forced by writing a one to the *Force Output Compare* (FOC1x) bit. Forcing compare match will not set the OCF1x flag or reload/clear the timer, but the OC1x pin will be updated as if a real compare match had occurred (the COM11:0 bits settings define whether the OC1x pin is set, cleared or toggled).

npare Match Blocking by NT1 Write

All CPU writes to the TCNT1 register will block any compare match that occurs in the next timer clock cycle, even when the timer is stopped. This feature allows OCR1x to be initialized to the same value as TCNT1 without triggering an interrupt when the Timer/Counter clock is enabled.

ng the Output Compare t

Since writing TCNT1 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT1 when using any of the output compare channels, independent of whether the Timer/Counter is running or not. If the value written to TCNT1 equals the OCR1x value, the compare match will be missed, resulting in incorrect waveform generation. Do not write the TCNT1 equal to TOP in PWM modes with variable TOP values. The compare match for the TOP will be ignored and the counter will continue to 0xFFFF. Similarly, do not write the TCNT1 value equal to BOTTOM when the counter is downcounting.

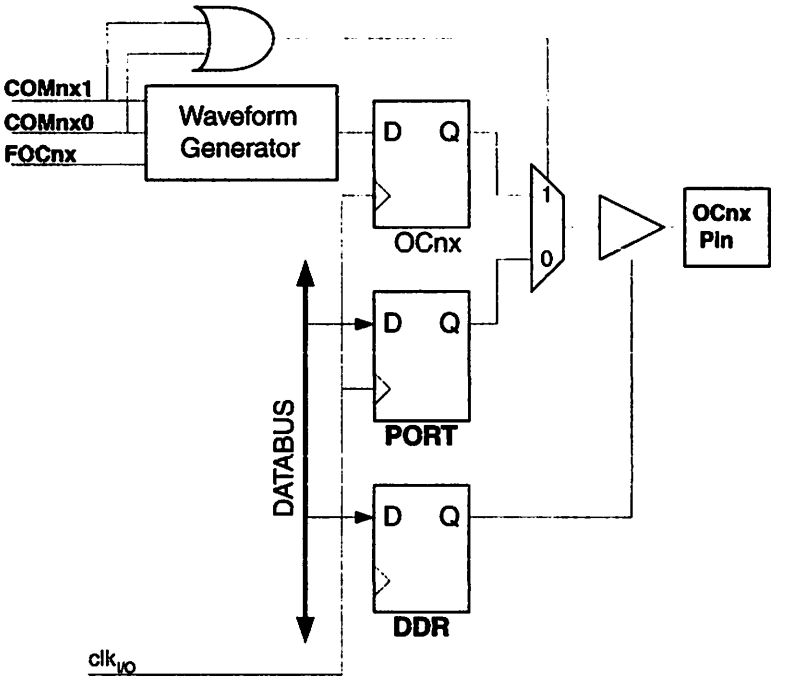
The setup of the OC1x should be performed before setting the data direction register for the port pin to output. The easiest way of setting the OC1x value is to use the force output compare (FOC1x) strobe bits in normal mode. The OC1x register keeps its value even when changing between Waveform Generation modes.

Be aware that the COM1x1:0 bits are not double buffered together with the compare value. Changing the COM1x1:0 bits will take effect immediately.

mpare Match Output it

The *Compare Output mode* (COM1x1:0) bits have two functions. The waveform generator uses the COM1x1:0 bits for defining the output compare (OC1x) state at the next compare match. Secondly the COM1x1:0 bits control the OC1x pin output source. Figure 36 shows a simplified schematic of the logic affected by the COM1x1:0 bit setting. The I/O registers, I/O bits, and I/O pins in the figure are shown in bold. Only the parts of the general I/O port control registers (DDR and PORT) that are affected by the COM1x1:0 bits are shown. When referring to the OC1x state, the reference is for the internal OC1x register, not the OC1x pin. If a System Reset occur, the OC1x register is reset to "0".

Figure 36. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the output compare (OC1x) from the waveform generator if either of the COM1x1:0 bits are set. However, the OC1x pin direction (input or output) is still controlled by the *Data Direction Register* (DDR) for the port pin. The data direction register bit for the OC1x pin (DDR_OC1x) must be set as output before the OC1x value is visible on the pin. The port override function is generally independent of the Waveform Generation mode, but there are some exceptions. Refer to Table 36, Table 37 and Table 38 for details.

The design of the output compare pin logic allows initialization of the OC1x state before the output is enabled. Note that some COM1x1:0 bit settings are reserved for certain modes of operation. See "16-bit Timer/Counter Register Description" on page 93.

The COM1x1:0 bits have no effect on the input capture unit.

npare Output Mode and veform Generation

The waveform generator uses the COM1x1:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM1x1:0 = 0 tells the waveform generator that no action on the OC1x register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 36 on page 93. For fast PWM mode refer to Table 37 on page 94, and for phase correct and phase and frequency correct PWM refer to Table 38 on page 94.

A change of the COM1x1:0 bits state will have effect at the first compare match after the bits are written. For non-PWM modes, the action can be forced to have immediate effect by using the FOC1x strobe bits.

des of Operation

The mode of operation (i.e., the behavior of the Timer/Counter and the output compare pins) is defined by the combination of the *Waveform Generation mode* (WGM13:0) and *Compare Output mode* (COM1x1:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM1x1:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM1x1:0 bits control whether the output should be set, cleared or toggle at a compare match. See "Compare Match Output Unit" on page 83.

For detailed timing information refer to "Timer/Counter Timing Diagrams" on page 91.

mal Mode

The simplest mode of operation is the *Normal* mode (WGM13:0 = 0). In this mode the counting direction is always up (incrementing), and no counter clear is performed. The counter simply overruns when it passes its maximum 16-bit value (MAX = 0xFFFF) and then restarts from the BOTTOM (0x0000). In normal operation the *Timer/Counter Overflow Flag* (TOV1) will be set in the same timer clock cycle as the TCNT1 becomes zero. The TOV1 flag in this case behaves like a 17th bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV1 flag, the timer resolution can be increased by software. There are no special cases to consider in the Normal mode, a new counter value can be written anytime.

The input capture unit is easy to use in Normal mode. However, observe that the maximum interval between the external events must not exceed the resolution of the counter. If the interval between events are too long, the timer overflow interrupt or the prescaler must be used to extend the resolution for the capture unit.

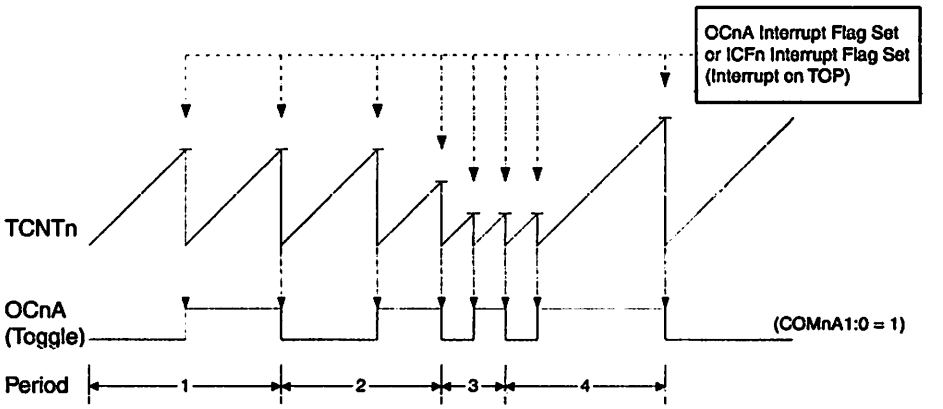
The output compare units can be used to generate interrupts at some given time. Using the output compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

ar Timer on Compare ch (CTC) Mode

In *Clear Timer on Compare* or CTC mode (WGM13:0 = 4 or 12), the OCR1A or ICR1 register are used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT1) matches either the OCR1A (WGM13:0 = 4) or the ICR1 (WGM13:0 = 12). The OCR1A or ICR1 define the top value for the counter, hence also its resolution. This mode allows greater control of the compare match output frequency. It also simplifies the operation of counting external events.

The timing diagram for the CTC mode is shown in Figure 37. The counter value (TCNT1) increases until a compare match occurs with either OCR1A or ICR1, and then counter (TCNT1) is cleared.

Figure 37. CTC Mode, Timing Diagram



An interrupt can be generated at each time the counter value reaches the TOP value by either using the OCF1A or ICF1 flag according to the register used to define the TOP value. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing the TOP to a value close to BOTTOM when the counter is running with none or a low prescaler value must be done with care since the CTC mode does not have the double buffering feature. If the new value written to OCR1A or ICR1 is lower than the current value of TCNT1, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. In many cases this feature is not desirable. An alternative will then be to use the fast PWM mode using OCR1A for defining TOP (WGM13:0 = 15) since the OCR1A then will be double buffered.

For generating a waveform output in CTC mode, the OC1A output can be set to toggle its logical level on each compare match by setting the compare output mode bits to toggle mode (COM1A1:0 = 1). The OC1A value will not be visible on the port pin unless the data direction for the pin is set to output (DDR_OC1A = 1). The waveform generated will have a maximum frequency of $f_{OC1A} = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). The waveform frequency is defined by the following equation:

$$f_{OCnA} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRnA)}$$

The N variable represents the prescaler factor (1, 8, 64, 256, or 1024).

As for the Normal mode of operation, the TOV1 flag is set in the same timer clock cycle that the counter counts from MAX to 0x0000.

t PWM mode

The *fast Pulse Width Modulation* or fast PWM mode (WGM13:0 = 5, 6, 7, 14, or 15) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM options by its single-slope operation. The counter counts from BOTTOM to TOP then restarts from BOTTOM. In non-inverting Compare Output mode, the output compare (OC1x) is set on the compare match between TCNT1 and OCR1x, and cleared at TOP. In inverting Compare Output mode output is cleared on compare match and set at TOP. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as high as the phase correct and phase and frequency correct PWM modes that use dual-slope operation. This high frequency makes the fast PWM mode well suited for power regulation, rectification, and DAC applications. High frequency

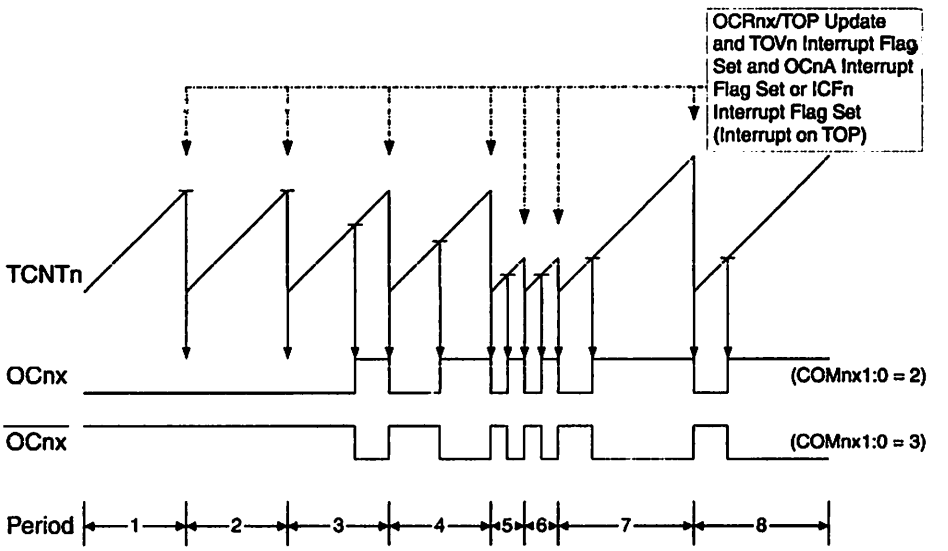
allows physically small sized external components (coils, capacitors), hence reduces total system cost.

The PWM resolution for fast PWM can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{FPWM} = \frac{\log(TOP+1)}{\log(2)}$$

In fast PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 5, 6, or 7), the value in ICR1 (WGM13:0 = 14), or the value in OCR1A (WGM13:0 = 15). The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 38. The figure shows fast PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the single-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 38. Fast PWM Mode, Timing Diagram



The Timer/Counter overflow flag (TOV1) is set each time the counter reaches TOP. In addition the OC1A or ICF1 flag is set at the same timer clock cycle as TOV1 is set when either OCR1A or ICR1 is used for defining the TOP value. If one of the interrupts are enabled, the interrupt handler routine can be used for updating the TOP and compare values.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the compare registers. If the TOP value is lower than any of the compare registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values the unused bits are masked to zero when any of the OCR1x registers are written.

The procedure for updating ICR1 differs from updating OCR1A when used for defining the TOP value. The ICR1 register is not double buffered. This means that if ICR1 is

changed to a low value when the counter is running with none or a low prescaler value, there is a risk that the new ICR1 value written is lower than the current value of TCNT1. The result will then be that the counter will miss the compare match at the TOP value. The counter will then have to count to the MAX value (0xFFFF) and wrap around starting at 0x0000 before the compare match can occur. The OCR1A register, however, is double buffered. This feature allows the OCR1A I/O location to be written anytime. When the OCR1A I/O location is written the value written will be put into the OCR1A buffer register. The OCR1A compare register will then be updated with the value in the buffer register at the next timer clock cycle the TCNT1 matches TOP. The update is done at the same timer clock cycle as the TCNT1 is cleared and the TOV1 flag is set.

Using the ICR1 register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A register is free to be used for generating a PWM output on OC1A. However, if the base PWM frequency is actively changed (by changing the TOP value), using the OCR1A as TOP is clearly a better choice due to its double buffer feature.

In fast PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to 3. See Table 37 on page 94. The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x register at the compare match between OCR1x and TCNT1, and clearing (or setting) the OC1x register at the timer clock cycle the counter is cleared (changes from TOP to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnxPWM} = \frac{f_{clk\_I/O}}{N \cdot (1 + TOP)}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR1x is set equal to BOTTOM (0x0000) the output will be a narrow spike for each TOP+1 timer clock cycle. Setting the OCR1x equal to TOP will result in a constant high or low output (depending on the polarity of the output set by the COM1x1:0 bits.)

A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC1A to toggle its logical level on each compare match (COM1A1:0 = 1). The waveform generated will have a maximum frequency of $f_{OC1A} = f_{clk\_I/O}/2$ when OCR1A is set to zero (0x0000). This feature is similar to the OC1A toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

ıse Correct PWM Mode

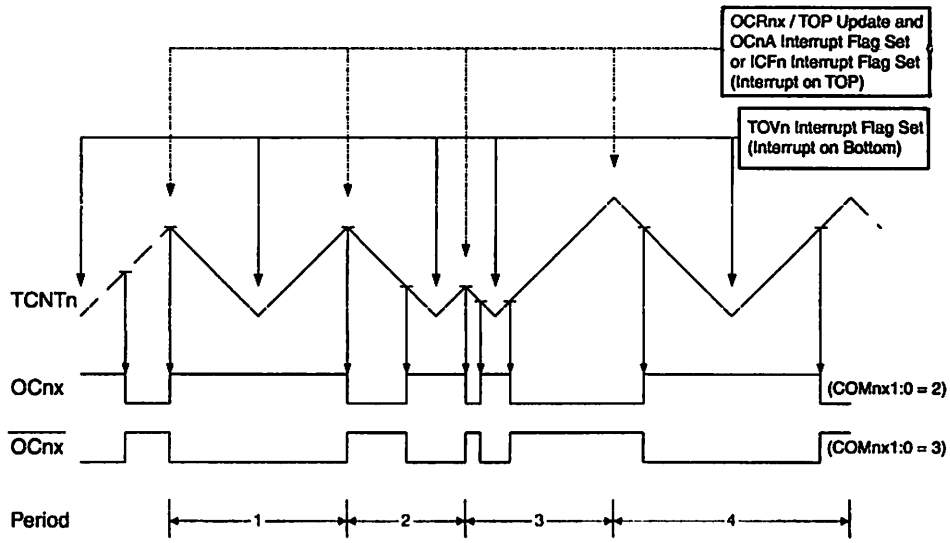
The *phase correct Pulse Width Modulation* or phase correct PWM mode (WGM13:0 = 1, 2, 3, 10, or 11) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is, like the phase and frequency correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the output compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting Output Compare mode, the operation is inverted. The dual-slope operation has lower maximum operation frequency than single slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

The PWM resolution for the phase correct PWM mode can be fixed to 8-, 9-, or 10-bit, or defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated by using the following equation:

$$R_{PCPWM} = \frac{\log(TOP+1)}{\log(2)}$$

In phase correct PWM mode the counter is incremented until the counter value matches either one of the fixed values 0x00FF, 0x01FF, or 0x03FF (WGM13:0 = 1, 2, or 3), the value in ICR1 (WGM13:0 = 10), or the value in OCR1A (WGM13:0 = 11). The counter has then reached the TOP and changes the count direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 39. The figure shows phase correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 39. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter overflow flag (TOV1) is set each time the counter reaches BOTTOM. When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 flag is set accordingly at the same timer clock cycle as the OCR1x registers are updated with the double buffer value (at TOP). The interrupt flags can be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the compare registers. If the TOP value is lower than any of the compare registers, a compare match will never occur between the TCNT1 and the OCR1x. Note that when using fixed TOP values, the unused bits are masked to zero when any of the OCR1x registers are written. As the third period shown in Figure 39 illustrates, changing the TOP actively while the Timer/Counter is running in the Phase Correct mode can result in an unsymmetrical output. The reason for this can be found in the time of update of the OCR1x register. Since the OCR1x update occurs at TOP, the PWM period starts and ends at TOP. This implies that the length of the falling

slope is determined by the previous TOP value, while the length of the rising slope is determined by the new TOP value. When these two values differ the two slopes of the period will differ in length. The difference in length gives the unsymmetrical result on the output.

It is recommended to use the phase and frequency correct mode instead of the phase correct mode when changing the TOP value while the Timer/Counter is running. When using a static TOP value there are practically no differences between the two modes of operation.

In phase correct PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to 3. See Table 38 on page 94. The actual OC1x value will only be visible on the port pin if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x register at the compare match between OCR1x and TCNT1 when the counter increments, and clearing (or setting) the OC1x register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnxPCPWM} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot TOP}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

se and Frequency Correct M Mode

The *phase and frequency correct Pulse Width Modulation,* or phase and frequency correct PWM mode (WGM13:0 = 8 or 9) provides a high resolution phase and frequency correct PWM waveform generation option. The phase and frequency correct PWM mode is, like the phase correct PWM mode, based on a dual-slope operation. The counter counts repeatedly from BOTTOM (0x0000) to TOP and then from TOP to BOTTOM. In non-inverting Compare Output mode, the output compare (OC1x) is cleared on the compare match between TCNT1 and OCR1x while upcounting, and set on the compare match while downcounting. In inverting Compare Output mode, the operation is inverted. The dual-slope operation gives a lower maximum operation frequency compared to the single-slope operation. However, due to the symmetric feature of the dual-slope PWM modes, these modes are preferred for motor control applications.

The main difference between the phase correct, and the phase and frequency correct PWM mode is the time the OCR1x register is updated by the OCR1x buffer register, (see Figure 39 and Figure 40).

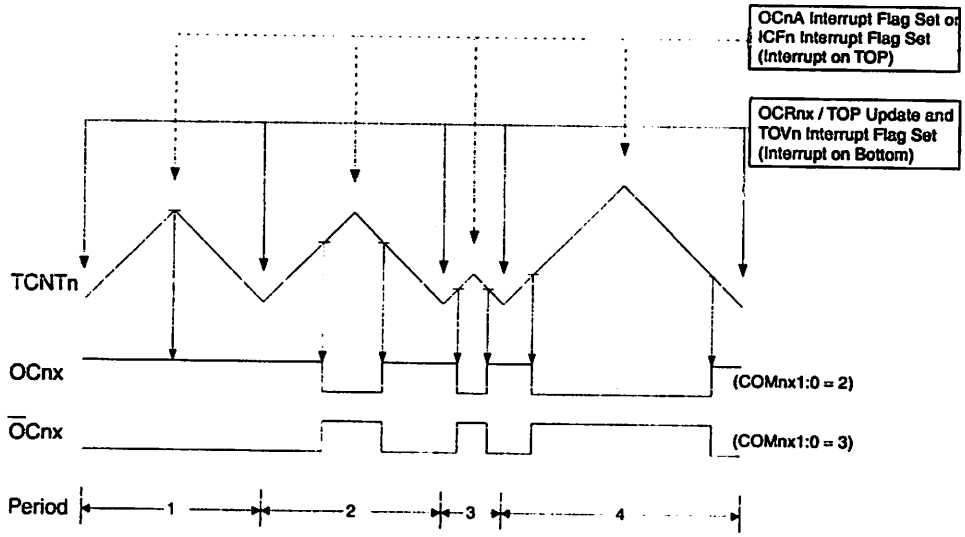
The PWM resolution for the phase and frequency correct PWM mode can be defined by either ICR1 or OCR1A. The minimum resolution allowed is 2-bit (ICR1 or OCR1A set to 0x0003), and the maximum resolution is 16-bit (ICR1 or OCR1A set to MAX). The PWM resolution in bits can be calculated using the following equation:

$$R_{PFCPWM} = \frac{\log(TOP+1)}{\log(2)}$$

In phase and frequency correct PWM mode the counter is incremented until the counter value matches either the value in ICR1 (WGM13:0 = 8), or the value in OCR1A (WGM13:0 = 9). The counter has then reached the TOP and changes the count

direction. The TCNT1 value will be equal to TOP for one timer clock cycle. The timing diagram for the phase correct and frequency correct PWM mode is shown on Figure 40. The figure shows phase and frequency correct PWM mode when OCR1A or ICR1 is used to define TOP. The TCNT1 value is in the timing diagram shown as a histogram for illustrating the dual-slope operation. The diagram includes non-inverted and inverted PWM outputs. The small horizontal line marks on the TCNT1 slopes represent compare matches between OCR1x and TCNT1. The OC1x interrupt flag will be set when a compare match occurs.

Figure 40. Phase and Frequency Correct PWM Mode, Timing Diagram



The Timer/Counter overflow flag (TOV1) is set at the same timer clock cycle as the OCR1x registers are updated with the double buffer value (at BOTTOM). When either OCR1A or ICR1 is used for defining the TOP value, the OC1A or ICF1 flag set when TCNT1 has reached TOP. The interrupt flags can then be used to generate an interrupt each time the counter reaches the TOP or BOTTOM value.

When changing the TOP value the program must ensure that the new TOP value is higher or equal to the value of all of the compare registers. If the TOP value is lower than any of the compare registers, a compare match will never occur between the TCNT1 and the OCR1x.

As Figure 40 shows the output generated is, in contrast to the phase correct mode, symmetrical in all periods. Since the OCR1x registers are updated at BOTTOM, the length of the rising and the falling slopes will always be equal. This gives symmetrical output pulses and is therefore frequency correct.

Using the ICR1 register for defining TOP works well when using fixed TOP values. By using ICR1, the OCR1A register is free to be used for generating a PWM output on OC1A. However, if the base PWM frequency is actively changed by changing the TOP value, using the OCR1A as TOP is clearly a better choice due to its double buffer feature.

In phase and frequency correct PWM mode, the compare units allow generation of PWM waveforms on the OC1x pins. Setting the COM1x1:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM1x1:0 to 3. See Table 38 on page 94. The actual OC1x value will only be visible on the port pin

if the data direction for the port pin is set as output (DDR_OC1x). The PWM waveform is generated by setting (or clearing) the OC1x register at the compare match between OCR1x and TCNT1 when the counter increments, and clearing (or setting) the OC1x register at compare match between OCR1x and TCNT1 when the counter decrements. The PWM frequency for the output when using phase and frequency correct PWM can be calculated by the following equation:

$$f_{OCnxPFCPWM} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot TOP}$$

The N variable represents the prescaler divider (1, 8, 64, 256, or 1024).

The extreme values for the OCR1x register represents special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR1x is set equal to BOTTOM the output will be continuously low and if set equal to TOP the output will be set to high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

ıer/Counter Timing ıgrams

The Timer/Counter is a synchronous design and the timer clock (clk_{T1}) is therefore shown as a clock enable signal in the following figures. The figures include information on when interrupt flags are set, and when the OCR1x register is updated with the OCR1x buffer value (only for modes utilizing double buffering). Figure 41 shows a timing diagram for the setting of OCF1x.

Figure 41. Timer/Counter Timing Diagram, Setting of OCF1x, no Prescaling

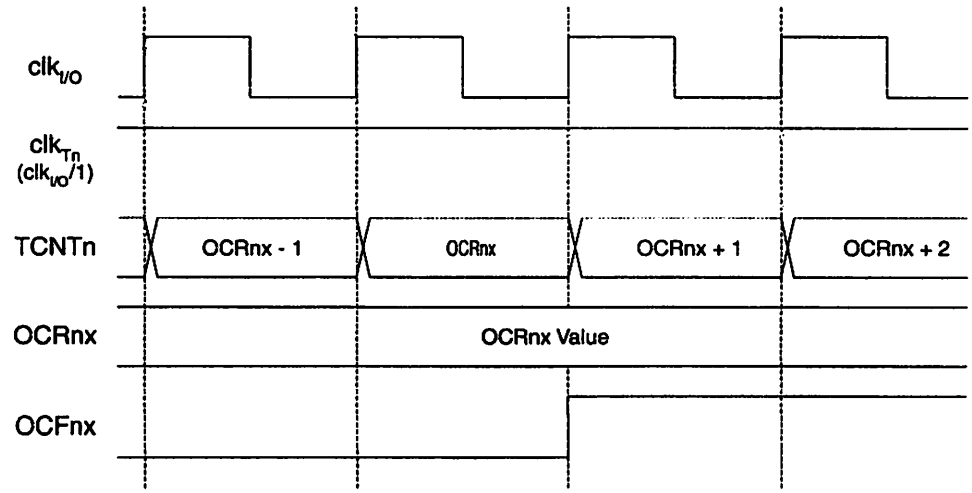


Figure 42 shows the same timing data, but with the prescaler enabled.

Figure 42. Timer/Counter Timing Diagram, Setting of OCF1x, with Prescaler ($f_{clk\_I/O}/8$)

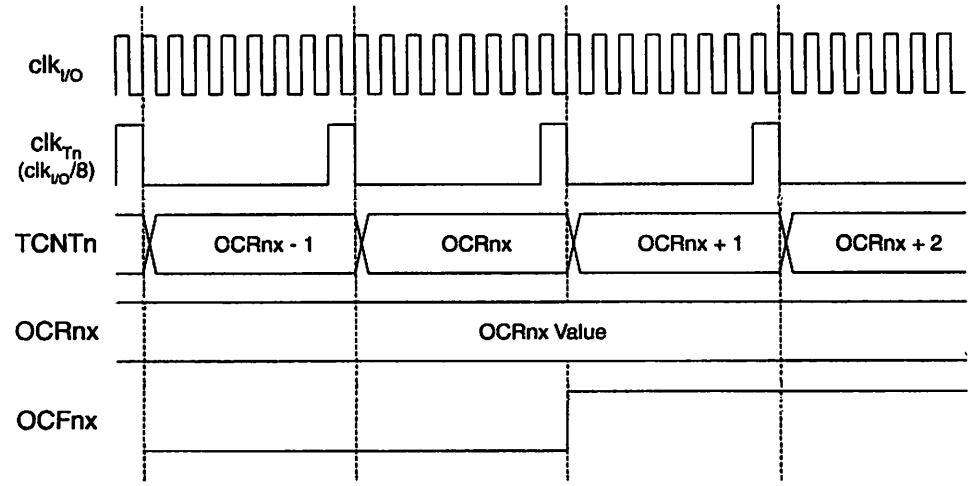


Figure 43 shows the count sequence close to TOP in various modes. When using phase and frequency correct PWM mode the OCR1x register is updated at BOTTOM. The timing diagrams will be the same, but TOP should be replaced by BOTTOM, TOP-1 by BOTTOM+1 and so on. The same renaming applies for modes that set the TOV1 flag at BOTTOM.

Figure 43. Timer/Counter Timing Diagram, no Prescaling

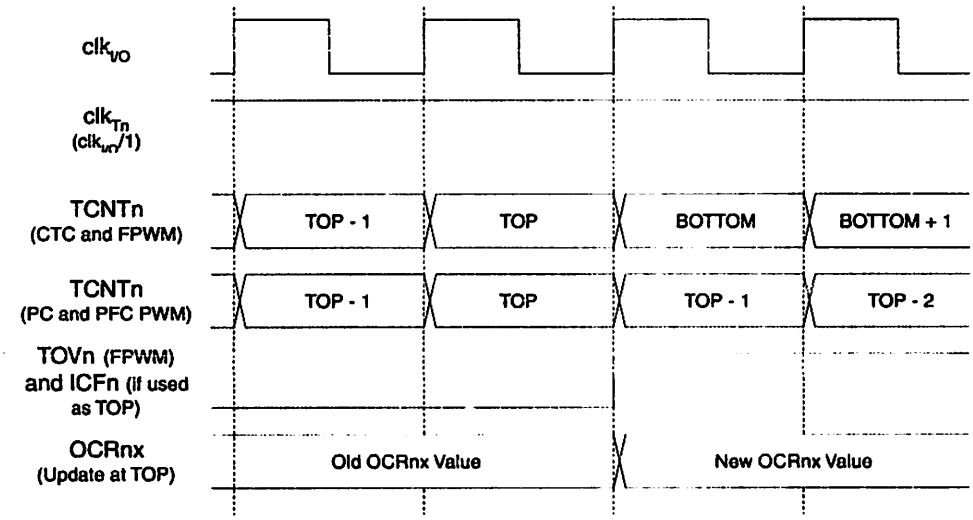
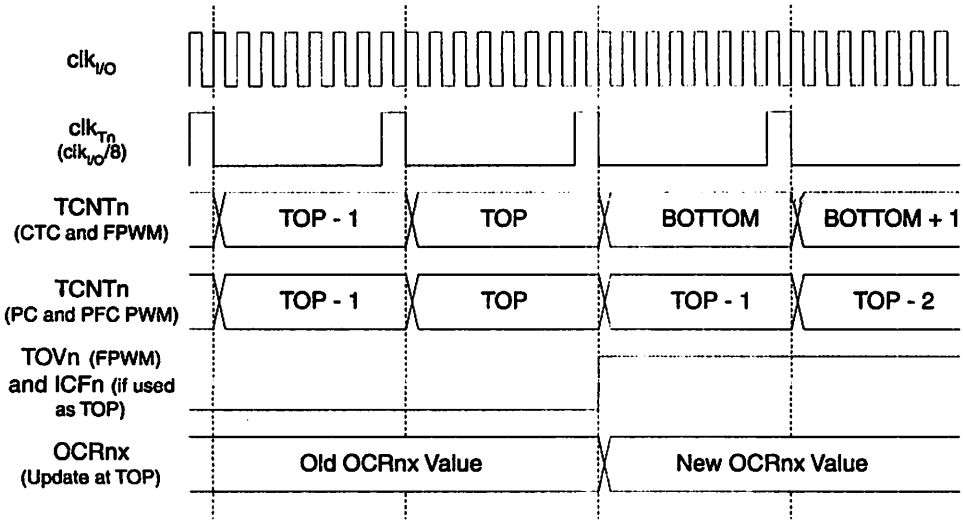


Figure 44 shows the same timing data, but with the prescaler enabled.

Figure 44. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)



·bit Timer/Counter gister Description

ıer/Counter 1 Control ǧister A – TCCR1A

Bit	7	6	5	4	3	2	1	0	
	COM1A1	COM1A0	COM1B1	COM1B0	FOC1A	FOC1B	WGM11	WGM10	TCCR1A
Read/Write	R/W	R/W	R/W	R/W	W	W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:6 – COM1A1:0: Compare Output Mode for channel A**

- **Bit 5:4 – COM1B1:0: Compare Output Mode for channel B**

The COM1A1:0 and COM1B1:0 control the output compare pins (OC1A and OC1B respectively) behavior. If one or both of the COM1A1:0 bits are written to one, the OC1A output overrides the normal port functionality of the I/O pin it is connected to. If one or both of the COM1B1:0 bit are written to one, the OC1B output overrides the normal port functionality of the I/O pin it is connected to. However, note that the *Data Direction Register* (DDR) bit corresponding to the OC1A or OC1B pin must be set in order to enable the output driver.

When the OC1A or OC1B is connected to the pin, the function of the COM1x1:0 bits is dependent of the WGM13:0 bits setting. Table 36 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to a normal or a CTC mode (non-PWM).

Table 36. Compare Output Mode, Non-PWM

COM1A1/ COM1B1	COM1A0/ COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	Toggle OC1A/OC1B on compare match
1	0	Clear OC1A/OC1B on compare match (Set output to low level)
1	1	Set OC1A/OC1B on compare match (Set output to high level)

Table 37 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the fast PWM mode.

Table 37. Compare Output Mode, Fast PWM[1]

COM1A/ COM1B1	COM1A0/ COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13 = 0: Normal port operation, OC1A/OC1B disconnected. WGM13 = 1: Toggle OC1A on compare match, OC1B reserved.
1	0	Clear OC1A/OC1B on compare match, set OC1A/OC1B at TOP
1	1	Set OC1A/OC1B on compare match, clear OC1A/OC1B at TOP

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. In this case the compare match is ignored, but the set or clear is done at TOP. See "Fast PWM mode" on page 85. for more details.

Table 38 shows the COM1x1:0 bit functionality when the WGM13:0 bits are set to the phase correct or the phase and frequency correct, PWM mode.

Table 38. Compare Output Mode, Phase Correct and Phase and Frequency Correct PWM[1]

COM1A1/ COM1B1	COM1A0/ COM1B0	Description
0	0	Normal port operation, OC1A/OC1B disconnected.
0	1	WGM13 = 0: Normal port operation, OC1A/OC1B disconnected. WGM13 = 1: Toggle OC1A on compare match, OC1B reserved.
1	0	Clear OC1A/OC1B on compare match when up-counting. Set OC1A/OC1B on compare match when downcounting.
1	1	Set OC1A/OC1B on compare match when up-counting. Clear OC1A/OC1B on compare match when downcounting.

Note: 1. A special case occurs when OCR1A/OCR1B equals TOP and COM1A1/COM1B1 is set. See "Phase Correct PWM Mode" on page 87. for more details.

- **Bit 3 – FOC1A: Force Output Compare for channel A**

- **Bit 2 – FOC1B: Force Output Compare for channel B**

The FOC1A/FOC1B bits are only active when the WGM13:0 bits specifies a non-PWM mode. However, for ensuring compatibility with future devices, these bits must be set to zero when TCCR1A is written when operating in a PWM mode. When writing a logical one to the FOC1A/FOC1B bit, an immediate compare match is forced on the waveform generation unit. The OC1A/OC1B output is changed according to its COM1x1:0 bits setting. Note that the FOC1A/FOC1B bits are implemented as strobes. Therefore it is the value present in the COM1x1:0 bits that determine the effect of the forced compare.

A FOC1A/FOC1B strobe will not generate any interrupt nor will it clear the timer in Clear Timer on Compare Match (CTC) mode using OCR1A as TOP.

The FOC1A/FOC1B bits are always read as zero.

- **Bit 1:0 – WGM11:0: Waveform Generation Mode**

Combined with the WGM13:2 bits found in the TCCR1B register, these bits control the counting sequence of the counter, the source for maximum (TOP) counter value, and what type of waveform generation to be used, see Table 39. Modes of operation supported by the Timer/Counter unit are: Normal mode (counter), Clear Timer on Compare match (CTC) mode, and three types of Pulse Width Modulation (PWM) modes. (See "Modes of Operation" on page 84.)

ıle 39. Waveform Generation Mode Bit Description

ɔde	WGM13	WGM12 (CTC1)	WGM11 (PWM11)	WGM10 (PWM10)	Timer/Counter Mode of Operation[1]	TOP	Update of OCR1x	TOV1 Flag Set on
0	0	0	0	0	Normal	0xFFFF	Immediate	MAX
1	0	0	0	1	PWM, Phase Correct, 8-bit	0x00FF	TOP	BOTTOM
2	0	0	1	0	PWM, Phase Correct, 9-bit	0x01FF	TOP	BOTTOM
3	0	0	1	1	PWM, Phase Correct, 10-bit	0x03FF	TOP	BOTTOM
4	0	1	0	0	CTC	OCR1A	Immediate	MAX
5	0	1	0	1	Fast PWM, 8-bit	0x00FF	TOP	TOP
6	0	1	1	0	Fast PWM, 9-bit	0x01FF	TOP	TOP
7	0	1	1	1	Fast PWM, 10-bit	0x03FF	TOP	TOP
8	1	0	0	0	PWM, Phase and Frequency Correct	ICR1	BOTTOM	BOTTOM
9	1	0	0	1	PWM, Phase and Frequency Correct	OCR1A	BOTTOM	BOTTOM
10	1	0	1	0	PWM, Phase Correct	ICR1	TOP	BOTTOM
11	1	0	1	1	PWM, Phase Correct	OCR1A	TOP	BOTTOM
12	1	1	0	0	CTC	ICR1	Immediate	MAX
13	1	1	0	1	(Reserved)	–	–	–
14	1	1	1	0	Fast PWM	ICR1	TOP	TOP
15	1	1	1	1	Fast PWM	OCR1A	TOP	TOP

e: 1. The CTC1 and PWM11:0 bit definition names are obsolete. Use the WGM12:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

ıer/Counter 1 Control gister B – TCCR1B

Bit	7	6	5	4	3	2	1	0	
	ICNC1	ICES1	–	WGM13	WGM12	CS12	CS11	CS10	TCCR1B
Read/Write	R/W	R/W	R	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7 – ICNC1: Input Capture Noise Canceler**

Setting this bit (to one) activates the input capture noise canceler. When the noise canceler is activated, the input from the Input Capture Pin (ICP1) is filtered. The filter function requires four successive equal valued samples of the ICP1 pin for changing its output. The input capture is therefore delayed by four oscillator cycles when the noise canceler is enabled.

- **Bit 6 – ICES1: Input Capture Edge Select**

This bit selects which edge on the Input Capture Pin (ICP1) that is used to trigger a capture event. When the ICES1 bit is written to zero, a falling (negative) edge is used as trigger, and when the ICES1 bit is written to one, a rising (positive) edge will trigger the capture.

When a capture is triggered according to the ICES1 setting, the counter value is copied into the Input Capture Register (ICR1). The event will also set the Input Capture Flag (ICF1), and this can be used to cause an Input Capture Interrupt, if this interrupt is enabled.

When the ICR1 is used as TOP value (see description of the WGM13:0 bits located in the TCCR1A and the TCCR1B register), the ICP1 is disconnected and consequently the input capture function is disabled.

- **Bit 5 – Reserved Bit**

This bit is reserved for future use. For ensuring compatibility with future devices, this bit must be written to zero when TCCR1B is written.

- **Bit 4:3 – WGM13:2: Waveform Generation Mode**

See TCCR1A register description.

- **Bit 2:0 – CS12:0: Clock Select**

The three clock select bits select the clock source to be used by the Timer/Counter, see Figure 41 and Figure 42.

Table 40. Clock Select Bit Description

CS12	CS11	CS10	Description
0	0	0	No clock source. (Timer/counter stopped)
0	0	1	$clk_{I/O}$/1 (No prescaling)
0	1	0	$clk_{I/O}$/8 (From prescaler)
0	1	1	$clk_{I/O}$/64 (From prescaler)
1	0	0	$clk_{I/O}$/256 (From prescaler)
1	0	1	$clk_{I/O}$/1024 (From prescaler)
1	1	0	External clock source on T1 pin. Clock on falling edge.
1	1	1	External clock source on T1 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter1, transitions on the T1 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

ıer/Counter 1 – TCNT1H ı TCNT1L

Bit	7	6	5	4	3	2	1	0	
	TCNT1[15:8]								TCNT1H
	TCNT1[7:0]								TCNT1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The two *Timer/Counter* I/O locations (TCNT1H and TCNT1L, combined TCNT1) give direct access, both for read and for write operations, to the Timer/Counter unit 16-bit counter. To ensure that both the high and low bytes are read and written simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary high byte register (TEMP). This temporary register is shared by all the other 16-bit registers. See "Accessing 16-bit Registers" on page 75.

Modifying the counter (TCNT1) while the counter is running introduces a risk of missing a compare match between TCNT1 and one of the OCR1x registers.

Writing to the TCNT1 register blocks (removes) the compare match on the following timer clock for all compare units.

tput Compare Register 1 A ıCR1AH and OCR1AL

Bit	7	6	5	4	3	2	1	0	
	OCR1A[15:8]								OCR1AH
	OCR1A[7:0]								OCR1AL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

tput Compare Register 1 B ıCR1BH and OCR1BL

Bit	7	6	5	4	3	2	1	0	
	OCR1B[15:8]								OCR1BH
	OCR1B[7:0]								OCR1BL
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The output compare registers contain a 16-bit value that is continuously compared with the counter value (TCNT1). A match can be used to generate an output compare interrupt, or to generate a waveform output on the OC1x pin.

The output compare registers are 16-bit in size. To ensure that both the high and low bytes are written simultaneously when the CPU writes to these registers, the access is performed using an 8-bit temporary high byte register (TEMP). This temporary register is shared by all the other 16-bit registers. See "Accessing 16-bit Registers" on page 75.

ıt Capture Register 1 – 1H and ICR1L

Bit	7	6	5	4	3	2	1	0	
	ICR1[15:8]								ICR1H
	ICR1[7:0]								ICR1L
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

The input capture is updated with the counter (TCNT1) value each time an event occurs on the ICP1 pin (or optionally on the analog comparator output for Timer/Counter1). The input capture can be used for defining the counter TOP value.

The input capture register is 16-bit in size. To ensure that both the high and low bytes are read simultaneously when the CPU accesses these registers, the access is performed using an 8-bit temporary high byte register (TEMP). This temporary register is shared by all the other 16-bit registers. See "Accessing 16-bit Registers" on page 75.

ıer/Counter Interrupt Mask ıister – TIMSK[1]

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	–	TOIE0	TIMSK
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Note: 1. This register contains interrupt control bits for several Timer/Counters, but only timer 1 bits are described in this section. The remaining bits are described in their respective timer sections.

- **Bit 5 – TICIE1: Timer/Counter1, Input Capture Interrupt Enable**

When this bit is written to one, and the I-flag in the status register is set (interrupts globally enabled), the Timer/Counter1 input capture interrupt is enabled. The corresponding interrupt vector (see "Interrupts" on page 42) is executed when the ICF1 flag, located in TIFR, is set.

- **Bit 4 – OCIE1A: Timer/Counter1, Output Compare A Match Interrupt Enable**

When this bit is written to one, and the I-flag in the status register is set (interrupts globally enabled), the Timer/Counter1 output compare A match interrupt is enabled. The corresponding interrupt vector (see "Interrupts" on page 42) is executed when the OCF1A flag, located in TIFR, is set.

- **Bit 3 – OCIE1B: Timer/Counter1, Output Compare B Match Interrupt Enable**

When this bit is written to one, and the I-flag in the status register is set (interrupts globally enabled), the Timer/Counter1 output compare B match interrupt is enabled. The corresponding interrupt vector (see "Interrupts" on page 42) is executed when the OCF1B flag, located in TIFR, is set.

- **Bit 2 – TOIE1: Timer/Counter1, Overflow Interrupt Enable**

When this bit is written to one, and the I-flag in the status register is set (interrupts globally enabled), the Timer/Counter1 overflow interrupt is enabled. The corresponding interrupt vector (see "Interrupts" on page 42) is executed when the TOV1 flag, located in TIFR, is set.

ıer/Counter Interrupt Flag gister – TIFR(1)

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	–	TOV0	TIFR
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W	
Initial Value	0	0	0	0	0	0	0	0	

Note: 1. This register contains flag bits for several Timer/Counters, but only timer 1 bits are described in this section. The remaining bits are described in their respective timer sections.

- **Bit 5 – ICF1: Timer/Counter1, Input Capture Flag**

This flag is set when a capture event occurs on the ICP1 pin. When the Input Capture Register (ICR1) is set by the WGM13:0 to be used as the TOP value, the ICF1 flag is set when the counter reaches the TOP value.

ICF1 is automatically cleared when the Input Capture Interrupt vector is executed. Alternatively, ICF1 can be cleared by writing a logic one to its bit location.

- **Bit 4 - OCF1A: Timer/Counter1, Output Compare A Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the Output Compare Register A (OCR1A).

Note that a forced output compare (FOC1A) strobe will not set the OCF1A flag.

OCF1A is automatically cleared when the Output Compare Match A interrupt vector is executed. Alternatively, OCF1A can be cleared by writing a logic one to its bit location.

- **Bit 3 – OCF1B: Timer/Counter1, Output Compare B Match Flag**

This flag is set in the timer clock cycle after the counter (TCNT1) value matches the Output Compare Register B (OCR1B).

Note that a forced output compare (FOC1B) strobe will not set the OCF1B flag.

OCF1B is automatically cleared when the Output Compare Match B interrupt vector is executed. Alternatively, OCF1B can be cleared by writing a logic one to its bit location.

- **Bit 2 – TOV1: Timer/Counter1, Overflow Flag**

The setting of this flag is dependent of the WGM13:0 bits setting. In normal and CTC modes, the TOV1 flag is set when the timer overflows. Refer to Table 39 on page 95 for the TOV1 flag behavior when using another WGM13:0 bit setting.

TOV1 is automatically cleared when the Timer/Counter1 Overflow interrupt vector is executed. Alternatively, TOV1 can be cleared by writing a logic one to its bit location.

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No. 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : M. Bali Suryo Utomo
Nim : 0317034
Masa Bimbingan : 03-November 2007 s/d 03-Mei-2008
Judul Skripsi : Perencanaan Dan Pembuatan pedometer digital disertai dengan penghitung jarak menggunakan ic ADXL202 berbasis mikrokontroller ATMEGA8

NO	Tanggal	Uraian	Paraf Pembimbing
1	05 Januari 2008	Konsultasi perencanaan alat	
2	11 Januari 2008	Konsultasi BAB I,II,II	
3	19 Januari 2008	Revisi tujuan,latar belakang	
4	25 Januari 2008	Konsultasi pembuatan alat	
5	4 Februari 2008	Konsultasi pengujian	
6	11 Februari 2008	Revisi hasil analisa pengujian	
7	18 Februari 2008	Konsultasi BAB V	
8	20 Februari 2008	Acc, buat makalah seminar hasil	
9	6 maret 2008	Acc ujian skripsi	

Malang, Maret 2008
Dosen Pembimbing

Ir.Teguh Herbasuki,MT
NIP. Y 1038900209

Form S-4b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura-gura No. 2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : M. Bali Suryo Utomo
Nim : 0317034
Masa Bimbingan : 03-November 2007 s/d 03- MEI -2008
Judul Skripsi : Perencanaan Dan Pembuatan pedometer digital disertai dengan penghitung jarak menggunakan ic ADXL202 berbasis mikrokontroller ATMEGA8

NO	Tanggal	Uraian	Paraf Pembimbing
1	31 januari 2008	Konsultasi BAB 1 sampai dengan BAB III	
2	14 Februari 2008	Revisi tata tulis BAB 1 sampai dengan BAB III, BAB IV revisi	
3	15 Februari 2008	Revisi BAB IV dan BAB V,buat makalah seminar proposal	
4	16 Februari 2008	Konsultasi makalah seminar	
5	6 Maret 2008	Acc Kompre	

Malang,
Dosen Pembimbing

I Komang Somawirata, ST,MT
NIP.P. 1030100361

Form S-4b

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

FORMULIR PERBAIKAN SKRIPSI

Dari hasil ujian skripsi jurusan Teknik Elektro jenjang strata satu (S-1), yang diselenggarakan pada:

Hari : Selasa
Tanggal : 18 Maret 2008

Telah dilakukan perbaikan oleh:

Nama : M. Bali Suryo Utomo
N.I.M : 03.17.034
Jurusan : TEKNIK ELEKTRO S-1
Konsentrasi : ELEKTRONIKA
Judul Skripsi : **Perencanaan dan Pembuatan pedometer digital disertai dengan penghitung jarak dengan menggunakan ic ADXL202 berbasis mikrokontroller ATMEGA8**

Penguji/Tanggal	Uraian	Paraf
Penguji II 18 Maret 2008	Perbaikan kalimat dalam buku skripsi sesuai dengan bahasa indonesia	
	Buat pengujian tentang nilai T1 dan nilai T2 untuk berbagai sudut	

Dosen Penguji,

Penguji II

(Dr. Cahyo Crysdian, MSc.)
NIP.Y. 1030400412

Mengetahui,

Dosen Pembimbing I

(Ir. Teguh Herbasuki, MT.)
NIP.Y. 1038900209

Dosen Pembimbing II

(I Komang Somawirata, ST, MT.)
NIP.Y. 1030100361