SKRIPSI

PERANCANGAN SISTEM PENGENDALIAN MOTOR DC PADA ROBOT MANUAL MENGGUNAKAN PID KONTROL BERBASIS MIKROKONTROLER ATMega 16

Disusun Oleh :

ERIK BUDHIAWAN

NIM 04.12.232

JURUSAN TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK ELEKTRONIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

SEPTEMBER 2008

LEMBAR PERSETUJUAN

PERANCANGAN SISTEM PENGENDALIAN MOTOR DC PADA ROBOT MANUAL MENGGUNAKAN PID KONTROL BERBASIS MIKROKONTROLER ATMega 16

SKRIPSI

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

Disusun Oleh :

ERIK BUDHIAWAN
NIM : 04.12.232

Diperiksa dan Disetujui

Dosen Pembimbing I	Dosen Pembimbing II
(Ir. Widodo Pudji M, MT.)	(I Komang Somawirata, ST, MT.)
NIP.Y. 1028700171	NIP.Y. 1030100361

Mengetahui
Ketua Jurusan Teknik Elektro S-I

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

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

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

BERITA ACARA UJIAN SKRIPSI

FAKULTAS TEKNOLOGI INDUSTRI

Nama	:	Erik Budhiawan
NIM	:	04.12.232
Jurusan	:	Teknik Elektro S-I
Konsentrasi	:	Teknik Elektronika
Judul Skripsi	:	**Perancangan Sistem Pengendalian Motor DC Pada Robot Manual Menggunakan PID Kontrol Berbasis Mikrokontroler ATMega16**

Dipertahankan dihadapan team penguji Skripsi jenjang Sarjana (S-I) pada :

Hari	:	Rabu
Tanggal	:	24 September 2008
Nilai	:	82,6 (A)

PANITIA MAJELIS PENGUJI

KETUA

Ir. Mochtar Asroni, MSME
NIP.Y. 1018100036

SEKRETARIS

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

ANGGOTA PENGUJI

PENGUJI I

Sotyohadi, ST. MSc
NIP.Y. 1039700309

PENGUJI II

Joseph Dedy Irawan, ST. MT
NIP.Y. 132315178

ABSTRAKSI

PERANCANGAN SISTEM PENGENDALIAN MOTOR DC PADA ROBOT MANUAL MENGGUNAKAN PID KONTROL BERBASIS MIKROKONTROLER ATmega 16

Erik Budhiawan
04.12.232
Fakultas Teknologi Industri, Jurusan Teknik Elektro S1,
Konsentrasi Teknik Elektronika, Institut Teknologi Nasional Malang
Jln. Raya Karanglo Km 2 Malang
Email: erick_inside@yahoo.com

Dosen Pembimbing : I. Ir. Widodo Pudji M, MT
II. I Komang Somawirata, ST, MT

Kata Kunci : PID Kontrol, Mikrokontroler ATMega16, Robot manual

Pada Laporan Tugas Akhir ini telah diimplementasikan sebuah robot yang dilengakapi dengan sistem pengendalian kecepatan motor DC menggunakan PID kontrol. Robot ini memiliki dua buah sensor kecepatan berupa sensor *optocoupler* model U yang terdiri dari *infra red* sebagai *transmitter* dan *photodiode* sebagai *receiver*. Output dari sensor kecapatan yang berupa pulsa merupakan umpan balik dari sistem yang kemudian diproses oleh PID kontrol agar tidak terdapat nilai *error* (*error* = 0). Robot ini dikendalikan dengan *joystick* yang berupa potensio. Data analog yang dikeluarkan oleh potensio diterima oleh ADC *internal* mikrokontroler untuk diubah menjadi data digital sehingga dapat diproses oleh mikrokontroler. Sistem ini juga dilengkapi dengan LCD dan *keypad matrik* 4x4 yang berfungsi untuk memasukkan parameter-parameter yang diperlukan PID kontrol dalam melakukan proses.

KATA PENGANTAR

Alhamdulillah, puji syukur kehadirat-Mu Ya Allah yang telah memberikan rahmat dan hidayah-Nya, sehingga dapat menyelesaikan skripsi yang berjudul "Perancangan sistem pengendalian motor DC pada robot manual menggunakan PID kontrol berbasis mikrokontroler ATMega 16" 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. Widodo Pudji M, MT selaku Dosen Pembimbing I dan Ka. Laboratorium Kendali Industri.
3. Bapak I Komang Somawirata, ST,MT selaku Dosen Pembimbing II dan Ka. Laboratorium Elektronika Analog dan Instrumentasi.
4. Ayah dan Ibu serta saudara-saudara kami yang telah memberikan do'a restu, dorongan, semangat, dan biaya.
5. Rekan-rekan Instruktur di Laboratorium Perancangan Elektronika.
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, Agustus 2008

Penyusun

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I

PENDAHULUAN

1.1. Latar Belakang

Sehubungan dengan perkembangan ilmu pengetahuan dan teknologi yang selalu mengalami kemajuan terus menerus. Perbaikan terhadap teknologi yang sudah ada terus dilakukan agar menjadi lebih baik dan lebih mudah. Salah satu bidang yang mengalami perkembangan lebih pesat adalah teknologi elektronika yang tidak terlepas dari tuntunan masyarakat yang terus menerus berkembang sesuai dengan kondisi dan situasi yang dihadapi. Seiring dengan perkembangan teknologi ini , maka pemanfaatannya dapat diterapkan dalam berbagai bidang. Disini penulis memberikan sebuah solusi khususnya di bidang robotik yaitu sebuah alat pengendali kecepatan motor DC dengan menggunakan PID kontrol.

Pada pengendalian kecepatan motor DC dengan metode umpan balik (PID Kontrol), masukan dari sistem adalah *setting point* / referensi. Masukan ini kemudian dibandingkan dengan kecepatan motor DC yang sebenarnya. Selisih dari masukan dan kecepatan sebenarnya menghasilkan kesalahan (*error*). Kesalahan inilah yang akan dikompensasi oleh pengendali. Untuk mengetahui kecepatan motor DC yang sebenarnya, perlu ditambahkan sensor kecepatan. Sensor ini dapat berupa *rotary encoder* yang menghasilkan pulsa-pulsa yang frekuensinya sebanding dengan kecepatan putar motor DC.

Dalam pengoperasiannya, motor listrik membutuhkan arus yang cukup besar. Sedangkan pengendali hanya dapat menyediakan arus yang tidak terlalu besar. Oleh karena itu dibutuhkan piranti penguat arus yang akan menguatkan sinyal keluaran pengendali. Jadi pengendali hanya menghasilkan variasi tegangan dengan arus kecil yang kemudian akan dikuatkan oleh penguat arus. Keluaran dari penguat arus inilah yang akan masuk untuk memutar motor DC. Pengendali akan mempertahankan kecepatan putar motor DC agar sesuai dengan masukan kecepatan yang diberikan.

1.2. Rumusan Masalah

Dalam perencanaan dan pembuatan sistem pengedalian motor DC menggunakan PID kontrol berbasis mikrokontroler ATmega16 maka dapat dirumuskan beberapa masalah yang akan dibahas yaitu:

1. Bagaimana merancang algoritma PID pada Mikrokontroler untuk pengendalian motor DC (*software*) ?
2. Bagaimana merancang dan membuat perangkat keras atau *hardware* untuk sistem pengendalian PID menggunakan mikrokontroler?

1.3. Batasan Masalah

Agar pembahasan dari perancangan dan pembuatan sistem pengendalian motor DC ini tidak terlalu meluas maka penyusun perlu membuat batasan-batasan masalah yang meliputi :

1. Sensor yang dirancang memiliki spesifikasi 1 putaran = 20 pulsa.
2. PWM dihasilkan dari Mikrokontroller.
3. Driver motor yang didesain memiliki tegangan maksimum = 24 volt dan memiliki arus maksimum = 15 ampere.
4. Duty Cycle PWM (DC PWM) = 0 % s/d 100 %.

1.4. Tujuan

Tujuan dari perancangan dan pembuatan sistem pengendalian motor DC ini adalah :

Merencanakan dan membuat sistem untuk mengendalikan kecepatan motor DC menggunakan PID kontrol yang diaplikasikan pada robot.

1.5. Metodologi

Metodologi yang dipakai dalam pembuatan skripsi ini adalah:

a. Studi Literatur

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

b. Penelitian Lapangan

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

c. Pengolahan Data

Mengolah Data dengan jalan membuat analisa dan menarik kesimpulan dari hasil pengujian yang ada.

1.6. Sistematika Pembahasan

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

BAB I Pendahuluan

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

BAB II Landasan Teori

Pada bab ini dibahas tentang teori-teori yang mendukung dalam perencanaan dan pembuatan alat ini yang meliputi PID kontrol, rangkaian sensor kecepatan, dan mikrokontroler ATmega16.

BAB III Perencanaan Dan Pembuatan Alat

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

BAB IV Pengujian Alat

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

BAB V Penutup

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

BAB II

TEORI DASAR

2.1. Pendahuluan

Pada bab ini akan dibahas mengenai teori penunjang dari peralatan yang direncanakan. Teori penunjang ini akan membahas tentang komponen dan peralatan pendukung pada alat yang dibuat. Pokok pembahasan pada bab ini adalah :

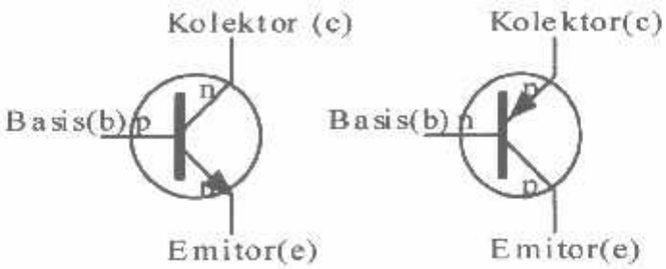
1. Transistor.
2. LED *Infra* merah.
3. *Photodiode.*
4. Schmitt Trigger.
5. Motor DC.
6. *Key pad.*
7. LCD (*Liquid Crystal Display*) M1632.
8. Mikrokontroler ATMEGA 16.
9. Kontroler PID

2.2. Transistor

2.2.1. Pendahuluan

Transistor merupakan devais dengan tiga terminal seperti yang diperlihatkan oleh simbol sirkit pada gambar 2.1. Setelah bahan semikonduktor

dasar diolah, terbentuklah bahan semikonduktor jenis P dan N. Walaupun proses pembuatannya banyak, pada dasarnya transistor merupakan tiga jenis lapis gabungan kedua jenis bahan tadi, yaitu NPN atau PNP.



Gambar 2.1. Simbol Sirkit untuk Transistor
Sumber: www.innovativeelectronics.com

Simbol sirkit kedua jenis transistor itu hampir sama. Perbedaannya terletak pada arah panah di ujung emitter. Arah panah ini menunjukkan arah aliran arus konvesional yang berlawanan arah dalam kedua jenis ini, tetapi selalu dari bahan jenis P ke jenis N dalam sirkit emitor dasar.

2.2.2. Transistor NPN

Kolektor dan emitter merupakan bahan N dan lapisan di antara keduanya merupakan jenis P. jika arus mengalir ke dalam basis dan melewati sambungan antara basis dan emitter, suatu suplai positif pada kolektor akan menyebabkan arus mengalir di anatara kolektor dan emitor. Dua hal yang harus diperhatikan pada arus kolektor ini adalah :

- Untuk arus basis nol, arus kolektor turun sampai pada tingkat arus kebocoran, yaitu kurang dari 1μA dalam kondisi normal (untuk transistor silikon)
- Untuk arus basis tertentu, arus kolektor yang mengalir akan jauh lebih besar daripada arus basis itu. Arus yang dicapai ini disebut H_{FE}, dengan $H_{FE} = I_C/I_B$.

2.2.3. Karakteristik Operasi Transistor

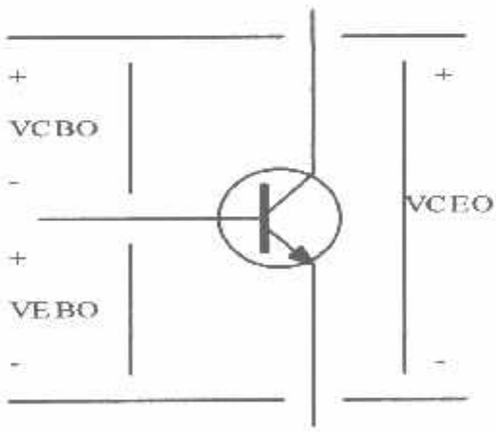
Karakteristik operasi tiap transistor yang menyatakan spesifikasinya tidak boleh dilampaui. Lembaran data memberikan nilai-nilai penting. Beberapa di antaranya diberikan di bawah ini dan diperlihatkan pada gambar 2.2.

V_{CBO} = tegangan basis- kolektor maksimum

V_{CEO} = tegangan emitter-kolektor maksimum

V_{EBO} = tegangan basis-emitter maksimum

P_{tot} = total daya yang diperlukan oleh transistor.



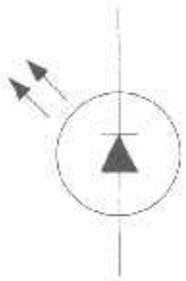
Gambar 2.2. Karakteristik Operasi Tegangan Transistor

Sumber: www.innovativeelectronics.com

2.3. LED *Infra* merah

LED *infra* merah digunakan untuk menghasilkan sinar *infra* merah. Prinsip kerja dari *infra* merah adalah pada waktu LED *infra* merah dibias *forward*, elektron dari pita konduksi melewati *junction* jatuh ke dalam *hole* pita valensi, sehingga elektron tersebut memancarkan energi. Pada dioda penyearah biasa, energi ini dipancarkan sebagai energi panas, sedangkan pada LED *infra* merah energi ini dipancarkan sebagai cahaya.

Simbol LED *infra* merah yang sering digunakan adalah :



Gambar 2.3. Simbol LED *Infra* Merah
Sumber: www.innovativeelectronics.com

LED *infra* merah merupakan *pin junction* yang memancarkan radiasi *infra* merah yang tidak kelihatan oleh mata kita. Apabila pada anoda diberi tegangan dan katoda ke *ground* maka LED menjadi *ON* dan arus akan mengalir dari anoda ke katoda. Pada reaksi semikonduktor, suatu dioda akan terjadi perpindahan elektron dari tipe N ke tipe P. Proses rekombinasi antara elektron dan *hole* menghasilkan pelepasan energi berupa pancaran cahaya.

Efisiensi pancaran cahaya akan berkurang seiring dengan berkurangnya arus input dan kenaikan suhu. Pada LED *infra* merah, cahaya yang dipancarkan

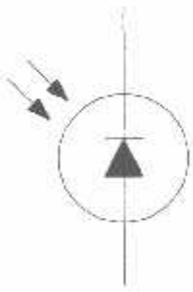
mempunyai panjang gelombang 0,1 mm – 1 μm sehingga pancaran gelombang tersebut tidak tertangkap oleh mata manusia.

2.4. Photodiode

Photodiode merupakan dioda yang peka terhadap cahaya. Suatu sumber cahaya menghasilkan energi panas begitu pula dengan spektrum *infra* merah. Karena spektrum *infra* merah mempunyai energi panas yang lebih besar dari cahaya tampak, maka *photodiode* lebih peka menangkap radiasi dari *infra* merah.

Komponen ini akan mengubah energi cahaya, dalam hal ini energi cahaya *infra* merah menjadi sinyal listrik. Komponen ini harus mampu mengumpulkan sinyal cahaya sebanyak mungkin sehingga sinyal listrik yang dihasilkan kualitasnya cukup baik. Semakin besar intensitas cahaya yang diterima maka sinyal pulsa listrik yang dihasilkan akan baik jika sinyal cahaya diterima intensitasnya lemah maka penerima tersebut harus mempunyai pengumpul cahaya (*light collector*) yang cukup baik dan sinyal pulsa yang dihasilkan oleh sensor cahaya ini harus dikuatkan.

Simbol dari *photodiode* adalah :



Gambar 2.4. Simbol Photodiode
Sumber: www.innovativeelectronics.com

Pada *photodiode* ini terdapat suatu jendela kecil yang memungkinkan cahaya luar dapat masuk mengenai *pin junction*. Pada keadaan normal *photodiode* berlaku sebagai dioda biasa yang dapat menghantarkan listrik dari *anoda* ke *katoda*, namun mempunyai tahanan balik yang besar. Bila cahaya luar mengenai *pin junction* fotodioda, maka tahanan balik akan mengecil dan menimbulkan arus balik, sehingga *photodiode* berlaku sebagai dioda yang dibalik atau dibias *reverse*.

Semakin besar intensitas cahaya yang diterima maka semakin besar pula arus balik yang ditimbulkannya. Bila energi *foton* diserap dalam suatu semikonduktor maka akan dihasilkan pasangan *electron hole* pada lapisan yang telah dibangkitkan oleh *foton* yang saling memisahkan diri karena pengaruh medan listrik, dimana elektron–elektron akan menuju ke sisi N dan *hole* menuju ke sisi P, sehingga dihasilkan arus dari katoda meuju *anoda*. Karena pengaruh suhu *junction* yang lebih tinggi, menciptakan lebih banyak pasangan *electron hole*, sehingga mengakibatkan arus balik yang melewati *junction* bertambah.

Sebuah *photodiode* biasanya dikemas dengan plastik transparan yang juga berfungsi sebagai lensa *fresnel*. Lensa ini merupakan lensa cembung yang mempunyai sifat mengumpulkan cahaya. Walaupun demikian cahaya yang tampakpun masih bisa mengganggu kerja dari *photodiode* karena tidak semua cahaya nampak bisa difilter dengan baik. Oleh karena itu sebuah penerima laser harus mempunyai filter kedua yaitu rangkaian filter yang berfungsi untuk memfilter sinyal *carrier* yang terbawa oleh cahaya laser tersebut.

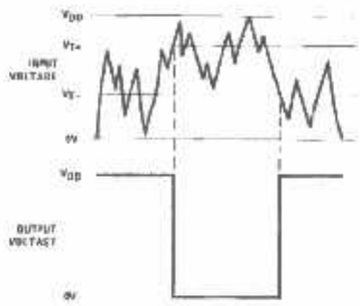
Faktor lain yang juga berpengaruh pada kemampuan penerima infra merah adalah '*active area*' dan '*respond time*'. Semakin besar area penerimaan suatu

photodiode maka semakin besar pula intensitas cahaya yang dikumpulkannya sehingga arus bocor yang diharapkan pada teknik '*reversed bias*' semakin besar. Selain itu semakin besar area penerimaan maka sudut penerimaannya juga semakin besar. Kelemahan area penerimaan yang semakin besar ini adalah *noise* yang dihasilkan juga semakin besar pula. Begitu juga dengan respon terhadap frekuensi, semakin besar area penerimaannya maka respon frekuensinya turun dan sebaliknya jika area penerimaannya kecil maka respon terhadap sinyal frekuensi tinggi cukup baik.

Respond time dari suatu photodioda (penerima) mempunyai waktu respon yang biasanya dalam satuan nano detik. *Respond time* ini mendefinisikan lama agar *photodiode* merespon cahaya *infra* merah yang datang pada area penerima. Sebuah *photodiode* yang baik paling tidak mempunyai *respond time* sebesar 500 *nano* detik atau kurang. Jika *respond time* terlalu besar maka *photodiode* ini tidak dapat merespon sinyal cahaya yang dimodulasi dengan sinyal *carrier* frekuensi tinggi dengan baik, hal ini akan mengakibatkan adanya *data loss*.

2.5. Schmitt Trigger

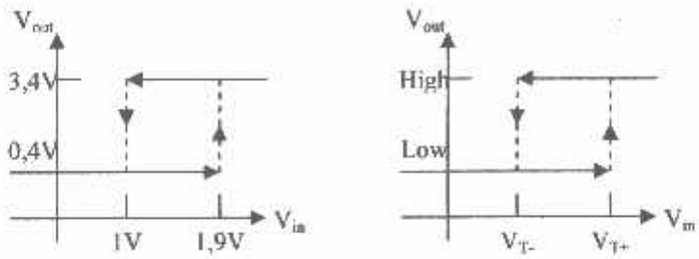
Bentuk gelombang yang buruk dan mempunyai waktu naik turun yang sangat lambat seperti dalam Gambar 2.5, dapat mengakibatkan operasi yang tidak dapat diandalkan, apabila dihubungkan langsung ke penghitung, gerbang, atau rangkaian lainnya. *Schmitt trigger* digunakan untuk mempersegikan sinyal input dan membuat sinyal menjadi lebih baik.



Gambar 2.5. *Schmitt Trigger* untuk pembentukan gelombang
Sumber : *Datasheet Inverter Schmitt Trigger* 74LS14

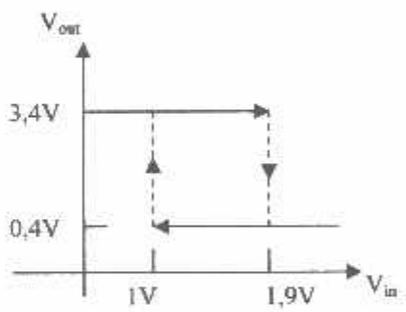
Gambar 2.6 (a) menunjukkan cara kerja sebuah *schmitt trigger*. Tegangan keluaran antara logika 0 dan 1 ialah 0,4V atau 3,4V. Bila V_{out} berada pada keadaan rendah, maka diperlukan untuk menaikkan V_{in} sedikit diatas 1,9V guna menghasilkan suatu perpindahan. Setelah berada pada keadaan tinggi, V_{out} tetap pada 3,4V dan V_{in} turun sedikit dibawah 1V. Pada saat ini, keluaran akan kembali ke keadaan rendah, yaitu 0,4V. Perpindahan yang cepat ini ditunjukkan oleh garis titik-titik.

Gambar 2.6 (b) menunjukkan grafik bagi setiap *schmitt trigger*. Nilai V_m yang mengakibatkan keluaran meloncat dari keadaan rendah ke tinggi disebut tegangan ambang menuju positif, yang dilambangkan sebagai V_{T+}. Demikian pula, V_{in} yang mengakibatkan keluaran berpindah dari keadaan tinggi ke keadaan rendah disebut tegangan ambang menuju negatif, yang dilambangkan sebagai V_{T-}.



Gambar 2.6. Karakteristik *schmitt trigger*
Sumber : *Datasheet Inverter Schmitt Trigger* 74LS14

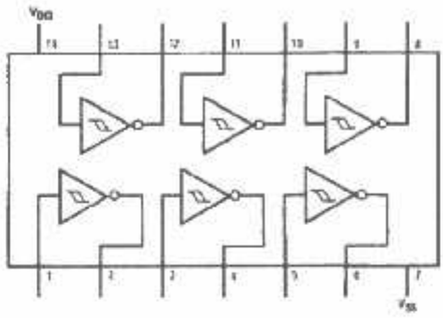
Sebagai contoh *inverter schmitt trigger* adalah IC TTL 74LS14 yang berisi 6 buah *inverter schmitt trigger*. Untuk Vcc sebesar 5V, tegangan ambang secara umum adalah V_T sebesar umum adalah V_{OL} sebesar 0,4V dan V_{OH} sebesar 3,4V. Grafik masukkan dan keluaran IC TTL 74LS14 ditunjukkan pada Gambar 2.7.



Gambar 2.7. Karakteristik *Inverter Schmitt Trigger*
Sumber : *Datasheet Inverter Schmitt Trigger* 74LS14

Akibat adanya inversi, maka histerisis IC TTL 74LS14 menjadi terbalik. Bila masukkan melampaui 1,9V, keluaran berpindah ke keadaan rendah. Bila masukkan kurang dari 1V, keluaran berpindah kembali ke keadaan tinggi. Keluaran dari inverter schmitt trigger ialah kebalikan dari masukkannya. *Schmitt*

trigger dengan gerbang not yang dikemas dalam bentu IC TTL 74LS14 diperlihatkan dalam gambar 2.8.

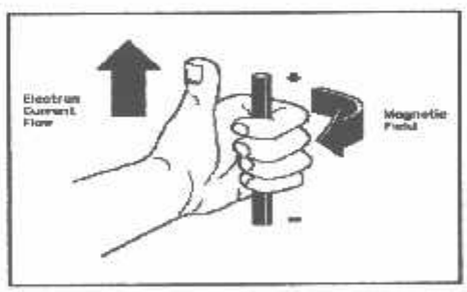


Gambar 2.8. *Inverter Schmitt Trigger IC* 74LS14
Sumber : *Datasheet Inverter Schmitt Trigger* 74LS14

2.6. Pengendali Arah Putaran Motor DC

2.6.1. Teori Dasar Motor DC

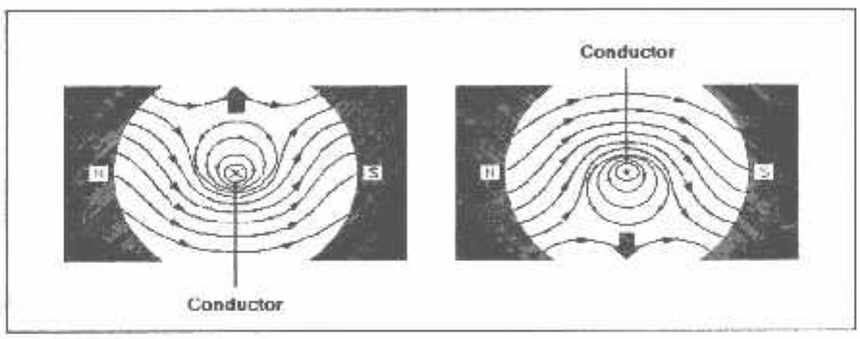
Setiap arus yang mengalir melalui sebuah konduktor akan menimbulkan medan magnet. Arah medan magnet dapat ditentukan dengan kaidah tangan kiri. Ibu jari tangan menunjukkan arah aliran arus listrik sedangkan jari-jari yang lain menunjukkan arah medan magnet yang timbul, seperti yang ditunjukkan oleh gambar 2.9. berikut ini.



Gambar 2.9. Kaidah Tangan Kiri

Sumber: www.innovativeelectronics.com

Jika suatu konduktor yang dialiri arus listrik ditempatkan dalam sebuah medan magnet, kombinasi medan magnet akan ditunjukkan oleh gambar 2.13. Arah aliran arus listrik dalam konduktor ditunjukkan dengan tanda "x" atau ".". Tanda "x" menunjukkan arah arus listrik mengalir menjauhi pembaca gambar, tanda "." menunjukkan arah arus listrik mengalir mendekati pembaca gambar.



Gambar 2.10. Konduktor Berarus Listrik Dalam Medan Magnet

Sumber: www.innovativeelectronics.com

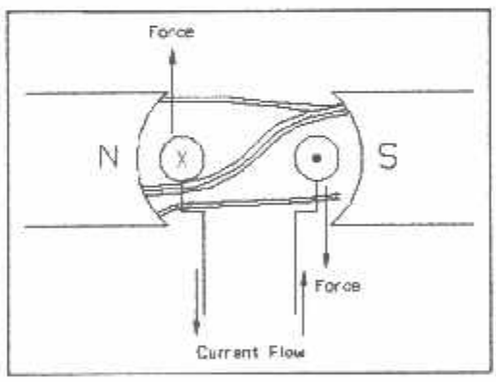
Pada gambar sebelah kiri, arah medan magnet pada sisi atas yang dihasilkan oleh konduktor berlawanan dengan arah medan magnet yang dihasilkan oleh magnet permanen. Sementara pada sisi sebelah bawah, arah medan magnet yang dihasilkan oleh konduktor searah dengan arah medan magnet yang dihasilkan oleh magnet permanen. Dengan kata lain, pada sisi sebelah atas kerapatan fluks magnet lebih sedikit dari pada sisi sebelah bawah. Sebuah gaya dorong akan menyebabkan konduktor bergerak ke sisi sebelah atas.

Pada gambar sebelah kanan, arah medan magnet pada sisi atas yang dihasilkan oleh konduktor searah dengan arah medan magnet yang dihasilkan oleh magnet permanen. Sementara pada sisi sebelah bawah, arah medan magnet

yang dihasilkan oleh konduktor berlawanan dengan arah medan magnet yang dihasilkan oleh magnet permanen. Dengan kata lain, pada sisi sebelah bawah kerapatan fluks magnet lebih sedikit dari pada sisi sebelah atas. Sebuah gaya dorong akan menyebabkan konduktor bergerak ke sisi sebelah bawah.

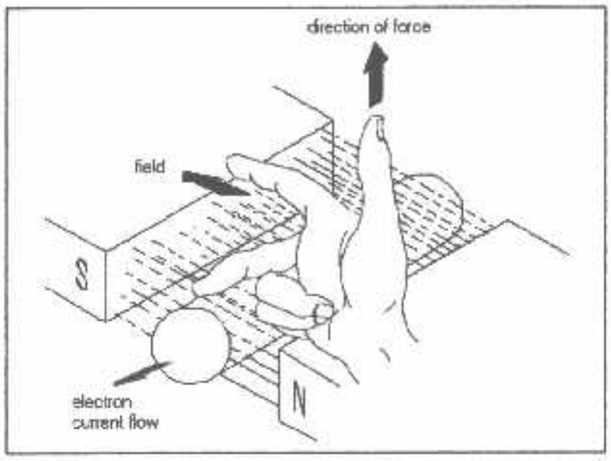
Pada sebuah motor DC, konduktor dibentuk menjadi sebuah loop sehingga ada dua bagian konduktor yang berada didalam medan magnet pada saat yang sama, seperti diperlihatkan pada gambar 2.14.

Konfigurasi konduktor seperti ini akan menghasilkan distorsi pada medan magnet utama dan menghasilkan gaya dorong pada masing-masing konduktor. Pada saat konduktor di tempatkan pada rotor, gaya dorong yang timbul akan menyebabkan rotor berputar searah dengan jarum jam.



Gambar 2.11. Bergeraknya Sebuah Motor
Sumber: www.innovativeelectronics.com

Sebuah cara lagi untuk menunjukkan hubungan antara arus listrik yang mengalir didalam sebuah konduktor, medan magnet dan arah gerak, adalah kaidah tangan kanan untuk motor seperti yang diperlihatkan pada gambar 2.12.



Gambar 2.12. Kaidah Tangan Kanan Untuk Motor

Sumber: www.innovativeelectronics.com

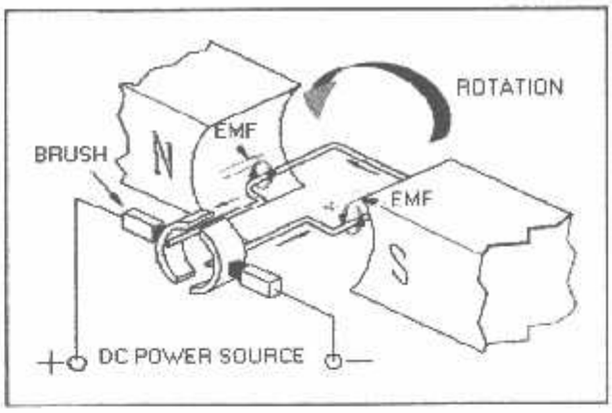
Kaidah tangan kanan untuk motor menunjukkan arah arus yang mengalir didalam sebuah konduktor yang berada dalam medan magnet. Jari tengah menunjukkan arah arus yang mengalir pada konduktor, jari telunjuk menunjukkan arah medan magnet dan ibu jari menunjukkan arah gaya putar. Adapun besarnya gaya yang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

F = B.L.I (Newton)

Dimana : B = kerapatan fluks magnet (weber)

L = panjang konduktor (meter)

I = arus listrik (ampere)

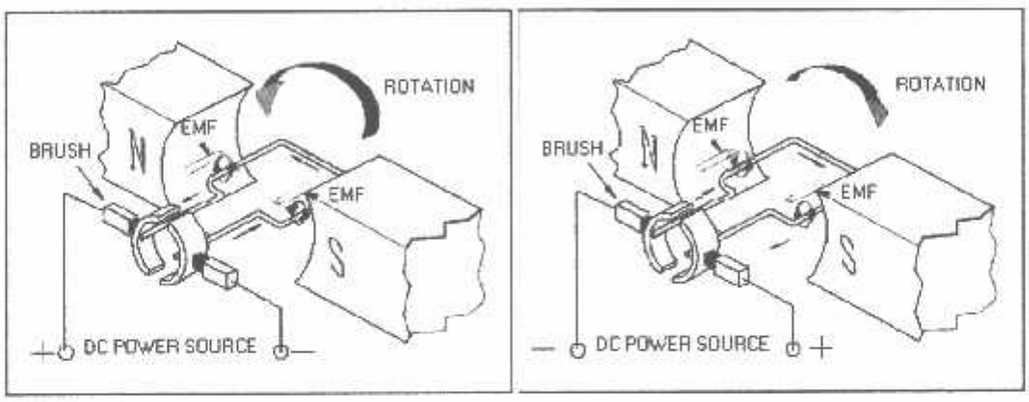


Gambar 2.13. Konstruksi Dasar Motor DC
Sumber: www.innovativeelectronics.com

Pada gambar 2.13. diatas tampak sebuah konstruksi dasar motor dc, pada gambar diatas terlihat bahwa pada saat terminal motor diberi tegangan dc, maka arus elektron akan mengalir melalui konduktor dari terminal negatif menuju ke terminal positif. Karena konduktor berada diantara medan magnet, maka akan timbul medan magnet juga pada konduktor yang arahnya seperti terlihat pada gambar 2.13. diatas. Arah garis gaya medan magnet yang dihasilkan oleh magnet permanen adalah dari kutub utara menuju ke selatan. Sementara pada konduktor yang dekat dengan kutub selatan, arah garis gaya magnet disisi sebelah bawah searah dengan garis gaya magnet permanen sedangkan di sisi sebelah atas arah garis gaya magnet berlawanan arah dengan garis gaya magnet permanen. Ini menyebabkan medan magnet disisi sebelah bawah lebih rapat daripada sisi sebelah atas. Dengan demikian konduktor akan terdorong ke arah atas. Sementara pada konduktor yang dekat dengan kutub utara, arah garis gaya magnet disisi sebelah atas searah dengan garis gaya magnet permanen sedangkan di sisi sebelah bawah arah garis gaya magnet berlawanan arah dengan garis gaya magnet

permanen. Ini menyebabkan medan magnet disisi sebelah atas lebih rapat daripada sisi sebelah bawah. Dengan demikian konduktor akan terdorong ke arah bawah. Pada akhirnya konduktor akan membentuk gerakan berputar berlawanan dengan jarum jam seperti terlihat pada gambar 2.13. diatas.

2.6.2. Pengendalian Arah Putaran Motor DC



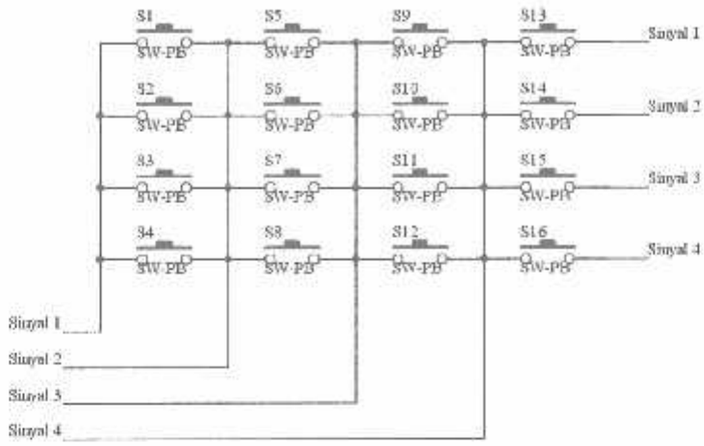
Gambar 2.14. Arah Putaran Motor DC
Sumber: www.innovativeelectronics.com

Dari gambar 2.14. diatas, agar arah putaran motor DC berubah, maka polaritas tegangan pada terminal motor harus dibalik.

2.7. Key Pad

Papan tombol ini digunakan untuk memasukkan data referensi dan mengubah data bila diinginkan. Untuk menterjemahkan informasi yang diterima dari papan tombol, maka *keypad* dihubungkan dengan *port* pada PPI 8255. Papan tombol tersebut mempunyai matrik 4 baris dan 3 kolom. Deretan baris dan kolom

dari papan tombol dihubungkan dengan *port* pada PPI 8255 yang difungsikan sebagai masukan dan keluaran.



Gambar 2.15. Penampang Dasar Keypad
Sumber: *Datasheet keypad* matrik 4x4.

Prinsip kerja keypad ini menggunakan metode matrik. Sinyal atau tegangan (sesuai dengan gambar di atas) diberikan secara bergantian pada masing-masing pin. Bila pin 1 mendapat sinyal 1 (tegangan +5 V), maka pin lainnya mendapat sinyal 0 (tegangan 0 V). Metode pengiriman sinyal seperti ini dapat menggunakan bilangan biner. Bila data yang dikirim 1000, berarti pin 1 mendapat sinyal 1, dan pin lainnya mendapat sinyal 0. Bila data yang dikirim 0010, berarti sinyal 1 diberikan pada pin 3.

Pada saat data yang dikirimkan 0100 (pin 2 aktif), dan tombol yang ada padanya (4, 5, dan 6) ditekan, maka sinyal tersebut akan dikirimkan ke bagian detektor sesuai dengan tombol yang ditekan. Jika tombol 4 ditekan, maka sinyal diteruskan pada detektor pin 1, jika tombol 4 dan 6 ditekan, maka sinyal akan diteruskan ke detektor pin 1 dan pin 3. Data pada detektor ini juga dapat dibaca

secara biner, bila data yang diterima 010, berarti pin 2 pada detektor mendapat sinyal (tombol 5 ditekan). Bila data yang diterima 011, berarti tombol 5 dan tombol 6 ditekan.

2.8. LCD (*Liquid Crystal Display*) M1632

Liquid Crystal Display adalah modul tampilan yang mempunyai konsumsi daya yang relatif rendah dan terdapat sebuah kontroller CMOS didalamnya. Kontroller tersebut sebagai pembangkit ROM/RAM dan *display* data RAM. Semua fungsi tampilan dikontrol oleh suatu instruksi modul LCD dapat dengan mudah diantar mukakan dengan MPU.

Spesifikasi dari LCD M1632:

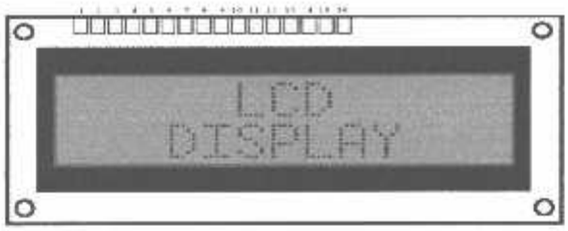
- Terdiri dari 32 karakter yang dibagi menjadi 2 baris dengan tampilan dot matrik 5 X 7 ditambah cursor.
- Karakter generator ROM dengan 192 karakter.
- Karakter generator RAM dengan 8 tipe karakter.
- 80 x 8 bit display data RAM .
- Dapat diantar mukakan dengan MPU 8 atau 4 bit.
- Dilengkapi fungsi tambahan : Display clear, cursor home, display ON/OFF, cursor ON/ OFF, display character blink, cursor shift dan display shift.
- Internal data.
- Internal otomatis dan *reset* pada *power* ON.
- +5 V *power supply* tunggal.

Tabel 2.1. Fungsi Pin LCD

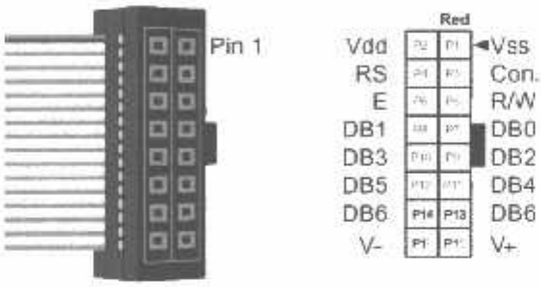
Nama Pin	Jumlah	I/O	Tujuan	Fungsi
DB0-DB3	4	I/O	MPU	Tri state bidirectional lower data bus: data dibaca dari modul ke MPU atau dari MPU ditulis ke modul melalui bus
DB4-DB7	4	I/O	MPU	Tri state bidirectional upper fourdata bus: data dibaca dari modul ke MPU atau dari MPU ditulis ke modul melalui bus
E	1	Input	MPU	Sinyal operasi dimulai: sinyal aktif baca/tulis
R/W	1	Input	MPU	Sinyalpilih data dan tulis (0:tulis,1:baca)

RS	1	-	Power supply	Sinyal pilih register : 0 : Instruction register (write) Busy flag dan addess counter (read) 1 : Data register (write dan read)
Con.	1	-	Power supply	Penyetelan kontras pada tampilan LCD
Vdd	1	-	Power supply	+ 5V
Vss	1	-	Power supply	Ground 0V

Sumber : *Datasheet* LCD M1632



(a)



(b)

Gambar 2.16. LCD M1632 (a),
Konektor dan Pin pada LCD M1632 (b)
Sumber: *Datasheet* LCD M1632

2.8.1. Register

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

DR menyimpan data sementara untuk ditulis ke DDRAM atau CGRAM, atau dibaca dari DD RAM atau CG RAM. Ketika data ditulis ke DDRAM atau CGRAM dari MPU, data di DR secara otomatis ditulis ke DDRAM atau CGRAM dengan operasi internal. Tetapi ketika data dibaca dari DDRAM atau CGRAM

maka alamat data ditulis pada IR.Data tersebut akan dimasukkan ke DR dan MPU akan membaca data dari DR, Setelah operasi pembacaan, alamat berikutnya diset data dari DDRAM atau CGRAM pada alamat tersebut akan dimasukkan ke DR untuk operasi berikutnya.

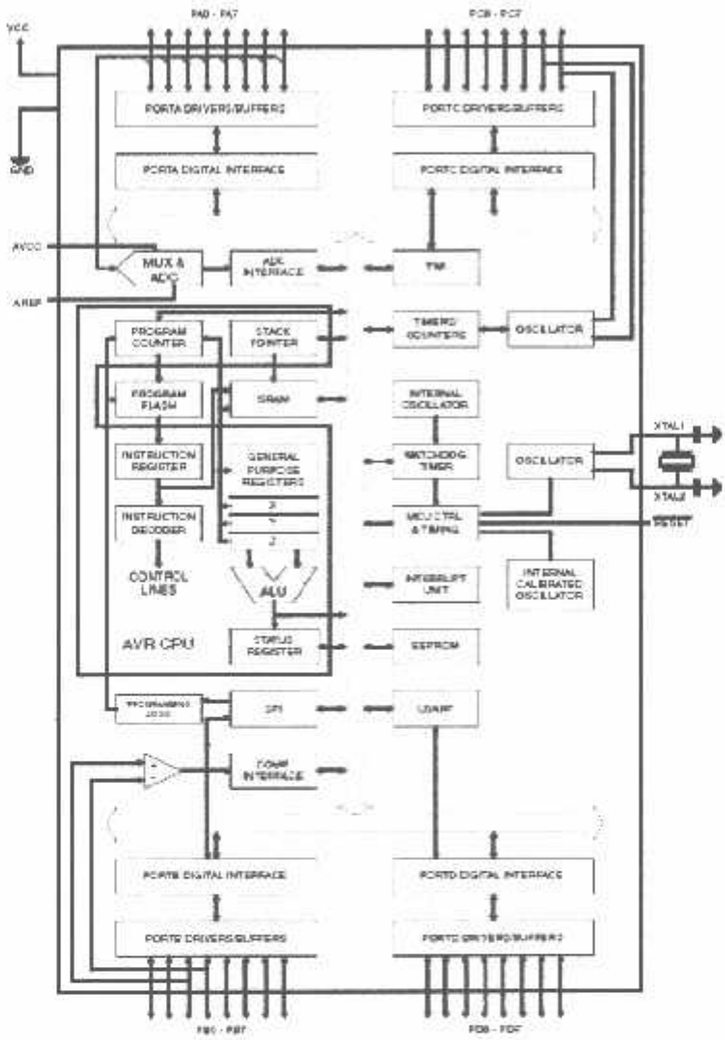
Display Data RAM (DDRAM) mempunyai kapasitas area 80 X 8 bit. Beberapa area dari DDRAM yang tidak digunakan untuk *display* dapat digunakan sebagai *General Data RAM.*

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

2.9. Mikrokontroler ATMEGA 16

Mikrokontroler AVR memiliki arsitektur RISC 8 bit, dimana semua instruksi dikemas dalam kode 16-bit (16-bit word) dan sebagian besar instruksi dieksekusi dalam 1 (satu) siklus clock. Secara umum, AVR dapat dikelompokkan menjadi 4 kelas, yaitu ATtiny, keluarga AT90Sxx, keluarga ATMega, dan AT86RFxx. Pada dasarnya yang membedakan masing-masing kelas adalah memori, peripheral, dan fungsinya. Dari segi arsitektur dan instruksi yang digunakan, mereka bisa dikatakan hampir sama.

2.9.1. Arsitektur ATMega16



Gambar 2.17. Blok diagram fungsional ATMega16

Sumber : www.atmel.com, *datasheet* ATMega16

Dari gambar tersebut dapat dilihat bahwa ATMega16 memiliki bagian sebagai berikut :

1. Saluran I/O sebanyak 32 buah,yaitu Port A, Port B, Port C, dan Port D.
2. ADC 10 bit sebanyak 8 saluran.
3. Tiga buah *Timer/Counter* dengan kemampuan pembandingan.

4. CPU yang terdiri atas 32 buah register.
5. *Watchdog Timer* dengan osilator internal.
6. SRAM sebesar 512 byte.
7. Memori Flash sebesar 16 kb dengan kemampuan *Read While Write*.
8. Unit interupsi internal dan eksternal.
9. Port antarmuka SPI.
10. EEPROM sebesar 512 byte yang dapat diprogram saat operasi.
11. Antarmuka komparator analog.
12. Port USART untuk komunikasi serial.

2.9.2. Fitur ATMega16

Kapabilitas detail dari ATMega16 adalah sebagai berikut :

1. Sistem mikroprosesor 8 bit berbasis *RISC* dengan kecepatan maksimal 16 MHz.
2. Kapabilitas memori *flash* 8 KB, *SRAM* sebesar 512 byte, dan *EEPROM* (*Electrically Erasable Programmable Read Only Memory*) sebesar 512 byte.
3. *ADC* internal dengan fidelitas 10 bit sebanyak 8 *channel*.
4. Portal komunikasi serial (*USART*) dengan kecepatan maksimal 2,5 Mbps.
5. Enam pilihan mode *sleep* menghemat penggunaan daya listrik.

2.9.3. Konfigurasi Pin ATMega16

Konfigurasi pin ATMega16 bisa dilihat pada gambar 2.18. Dari gambar tersebut dapat dijelaskan secara fungsional konfigurasi pin ATMega16 sebagai berikut :

A. VCC merupakan pin yang berfungsi sebagai pin masukan catu daya.

B. GND merupakan pin *ground*.

C. Port A (PA0..PA7) merupakan pin I/O dua arah dan pin masukan ADC.

D. Port B (PB0..PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu Timer/Counter, komparator analog, dan SPI.

E. Port C (PC0..PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, komparator analog, dan *Timer Oscilator*.

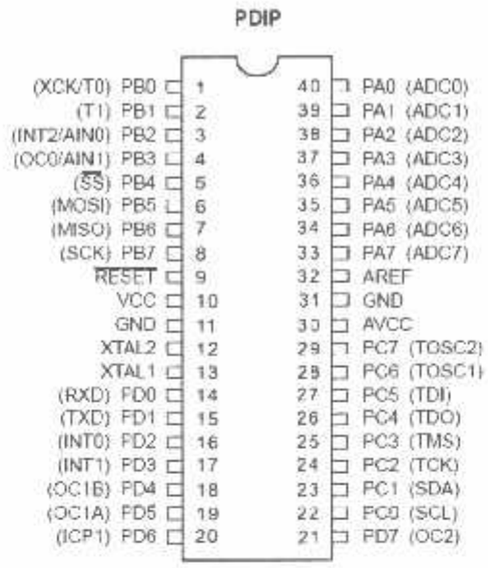
F. Port D (PD0..PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komparator analog, interupsi eksternal, dan komunikasi serial.

G. RESET merupakan pin yang digunakan utuk me-reset mikrokontroler.

H. XTAL1 dan XTAL2 merupakan pin masukan *clock* eksternal.

I. AVCC merupakan pin masukan tegangan untuk ADC.

J. AREF merupakan pin masukan tegangan referensi ADC.



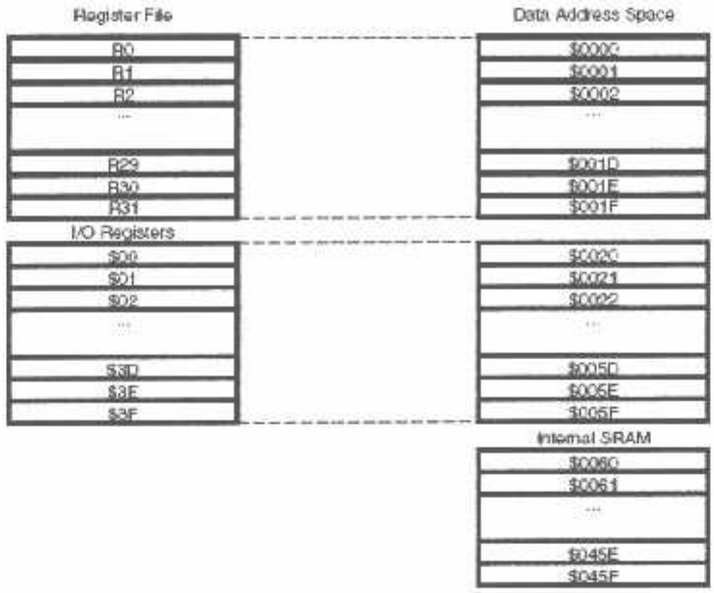
Gambar 2.18. Pin ATMega16

Sumber : www.atmel.com, *datasheet* ATMega16

2.9.4. Peta Memori

AVR ATMega16 memiliki ruang pengamatan memori data dan memori program yang terpisah. Memori data terbagi menjadi 3 bagian, yaitu 32 buah register umum, 64 buah register I/O, dan 512 *byte* SRAM *Internal*.

Register keperluan umum menempati space data pada alamat terbawah, yaitu $00 sampai $1F. Sementara itu, register khusus untuk menangani I/O dan control terhadap mikrokontroler menempati 64 alamat berikutnya, yaitu mulai dari $20 hingga $5F. Register tersebut merupakan register yang khusus digunakan untuk mengatur fungsi terhadap berbagai *peripheral* mikrokontroler, seperti control register, *timer/counter*, fungsi-fungsi I/O, dan sebagainya. Alamat memori berikutnya digunakan untuk SRAM 512 byte, yaitu pada lokasi $60 sampai dengan $25F. Konfigurasi memori data ditunjukkan pada gambar dibawah ini.



Gambar 2.19. Konfigurasi Memori Data AVR ATMega16

Sumber : www.atmel.com, *datasheet* ATMega16

Memori program yang terletak dalam *Flash PEROM* tersusun dalam word atau 2 byte karena setiap instruksi memiliki lebar 16-bit atau 32-bit. AVR ATMega16 memiliki 4K*Byte*X16-bit *Flash PEROM* dengan alamat mulai dari $000 sampai $FFF. AVR tersebut memiliki 12-bit *Program Counter* (PC) schingga mampu mengamati isi *Flash*.

Gambar 2.20. Memory Program AVR ATMega16

Sumber : www.atmel.com, *datasheet* ATMega16

Selain itu, AVR ATMega16 juga memiliki memori data berupa EEPROM 8-bit sebanyak 512 *byte*. Alamat EEPROM dimulai dari $000 sampai $1FF.

2.9.5. Status Register (SREG)

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

a. Bit 7 – I : *Global Interrupt Enable*

   Bit harus diset untuk meng-enable interupsi. Setelah itu, anda dapat mengaktifkan interupsi mana yang akan anda gunakan dengan cara meng-enable bit control register yang bersangkutan secara individu. Bit akan di-clear apabila terjadi suatu interupsi yang dipicu oleh hardware, dan bit tidak akan mengizinkan terjadinya interupsi, serta akan diset kembali oleh instruksi RETI.

b. Bit 6 – T : *Bit Copy Storage*

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

c. Bit 5 – H : *Half Carry Flag*

d. Bit 4 – S : *Sign Bit*

   Bit-S merupakan hasil operasi EOR antara flag-N (negatif) 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 : *Negative Flag*

Apabila suatu operasi menghasilkan bilangan negatif, 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.

2.10. Kontroler PID

Keberadaan kontroler dalam sebuah sistem kontrol mempunyai kontribusi yang besar terhadap perilaku sistem. Pada prinsipnya hal itu disebabkan oleh tidak dapat diubahnya komponen penyusun sistem tersebut. Artinya, karakteristik plant harus diterima sebagaimana adanya, sehingga perubahan perilaku sistem hanya dapat dilakukan melalui penambahan suatu sub sistem, yaitu kontroler.

Salah satu tugas komponen kontroler adalah mengurangi sinyal kesalahan, yaitu perbedaan antara nilai referensi/nilai yang diinginkan dengan nilai aktual. Hal ini sesuai dengan tujuan sistem kontrol yaitu mendapatkan nilai sinyal keluaran sama dengan nilai yang diinginkan (referensi). Semakin kecil kesalahan yang terjadi, semakin baik kinerja sistem kontrol yang diterapkan.

Apabila perbedaan antara nilai referensi dengan nilai keluaran relatif besar, maka kontroler yang baik seharusnya mampu mengamati perbedaan ini untuk segera menghasilkan sinyal kontrol yang dapat mempengaruhi plant. Dengan

demikian, sistem secara cepat mengubah keluaran plant sampai diperoleh selisih dengan nilai referensi sekecil mungkin.

Prinsip kerja kontroler adalah membandingkan nilai aktual keluaran plant dengan nilai referensi, kemudian menentukan nilai kesalahan dan akhirnya menghasilkan sinyal kontrol untuk meminimalkan kesalahan tersebut. Kontroler dapat diklasifikasikan sesuai dengan aksi pengontrolannya sebagai berikut:

2.10.1. Kontroler Proporsional (P)

Proporsional (P – elemen) menangani kesalahan(error) dengan segera, yang mana kesalahan tersebut dikalikan dengan suatu nilai konstanta Kp. Perlu dicatat juga jika nilai kesalahan sama dengan nol, maka keluaran dari proporsional kontroller juga sama dengan nol. Kontroler proporsional selalu memerlukan error untuk menghasilkan output. Hal ini menyebabkan terjadinya offset (error steady state) pada pemakaian kontroler proporsional.

Hubungan antara keluaran kontroler m(t) dan sinyal *error* e(t) dari kontroler Proporsional adalah :

$$m(t) = K_p\, e(t) \quad \text{.....(1)}$$

atau dalam transformasi Laplace :

$$\frac{M(s)}{E(s)} = K_p \quad \text{.....(2)}$$

Dimana K_p adalah *Gain* (penguatan proporsional).

2.10.2. Kontroler Integral (I)

Integral (I – elemen) mempelajari nilai yang telah lalu (lampau), kesalahan diintegrasikan dan dikalikan dengan konstanta Ki. Bentuk integrasi memampukan kontroller untuk menghilangkan kondisi kesalahan yang tetap (steady state error) jika proses membutuhkan nilai masukan bukan nol untuk mendapatkan setpoint yang diinginkan. Suatu integral kontroller akan bereaksi terhadap kesalahan (error) oleh peningkatan nilai yang ditambahkan ke nilai keluarannya. Sehingga hal ini akan memaksa kontroller untuk mencapai nilai setpoint-nya lebih cepat dibandingkan dengan hanya proporsional kontroller dan menekan atau menghilangkan kondisi kesalahan yang tetap, hal ini juga menjamin bahwa proses overshoot di setpoint diawali dari nilai integral yang secara berkelanjutan ditambahkan ke nilai keluarannya. Untuk menghilangkan offset yang terjadi akibat pemakaian kontroler Proporsional, maka digunakanlah kontroler Integral. Kontroler Integral dapat menghasilkan output walaupun sudah tidak ada input. Namun, reaksi dari kontroler ini sangat lambat.

Pada kontroler Integral, harga keluaran kontroler m(t) diubah dengan laju yang sebanding dengan sinyal *error* e(t), sehingga

$$\frac{dm(t)}{dt} = K_i\, e(t) \quad \text{.....(3)}$$

atau

$$m(t) = K_i \int e(t)\, dt \quad \text{.....(4)}$$

K_i merupakan konstanta yang dapat diatur. Dalam transformasi Laplace dapat dinyatakan:

$$\frac{M(s)}{E(s)} = \frac{K_i}{s} \quad \text{.....(5)}$$

Dimana $K_i = K_p / T_i$

2.10.3. Kontroler Diferensial (D)

Derivatif (D – elemen) untuk mengantisipasi nilai yang akan datang, derivatif yang pertama dari kesalahan yang terjadi (error) dikalikan dengan suatu konstanta Kd. Hal ini dapat dipakai untuk mengurangi besarnya overshoot yang dihasilkan oleh komponen integral, tetapi kontroller akan sedikit lebih lambat untuk mencapai setpoint. Hubungan sinyal kontroler m(t) dengan sinyal *error* e(t) dari kontroler Diferensial dapat dinyatakan dengan :

$$m(t) = K_d \frac{de(t)}{dt} \quad \text{.....(6)}$$

atau dalam transformasi Laplace dinyatakan dengan :

$$\frac{M(s)}{E(s)} = K_d\, s \quad \text{.....(7)}$$

di mana $K_d = K_p T_d$

Keluaran kontroler differensial merupakan turunan fungsi input. Kontroler differensial tidak dapat digunakan sendirian, karena unsur D tidak akan

mengeluarkan output bila tidak terjadi perubahan input. Oleh karena itu, kontroler ini harus selalu digunakan bersama kontroler Proporsional, Integral, atau keduanya.

2.10.4. Kontroler Proporsional Integral (PI)

Terjadinya offset pada penggunaan kontroler P menyebabkan perlunya mencari alternatif kontroler yang lain. Maka digunakanlah kontroler I. Namun, karena kelambatan reaksinya, maka kontroler I biasanya dipakai bersama dengan kontroler P menjadi kontroler PI.

Dengan demikian, maka kontroler PI dapat menghasilkan respon yang lebih cepat dari kontroler I, dan juga dapat menghilangkan offset yang ditimbulkan oleh kontroler P.

Aksi kontroler PI dinyatakan dalam persamaan :

$$m(t) = K_p\, e(t) + \frac{K_p}{T_i} e(t)\, dt \qquad \text{.....(8)}$$

atau dalam transformasi Laplace dinyatakan dengan :

$$\frac{m(s)}{E(s)} = K_p(1 + \frac{1}{T_i s}) \qquad \text{.....(9)}$$

Dalam hal ini K_p menyatakan penguatan dan T_i menyatakan waktu integral. Baik K_p maupun T_i dapat diatur. Kontroler Proporsional Integral (PI) dapat dibentuk dari gabungan kontroler Proporsional dan kontroler Integral.

$$(1-z^{-1})M(z) = k_p(1-z^{-1})E(z) + (k_i T/2)(1+z^{-1})E(z) + (k_d/T)(1-z^{-1})^2 E(z)$$

$$M(z) = k_p E(z) + k_i \frac{T}{2}\left(\frac{1+z^{-1}}{1-z^{-1}}\right)E(z) + \frac{k_d}{T}(1-z^{-1})E(z)$$

$$\frac{M(z)}{E(z)} = k_p + k_i \frac{T}{2}\left(\frac{z+1}{z-1}\right) + \frac{k_d}{T}\left(\frac{z-1}{z}\right)$$

$$= k_p + k_i \frac{T}{2}\left(\frac{z+1}{z-1}\right) + \frac{k_d}{T}\left(\frac{z-1}{z}\right)$$

$$= \frac{k_p(z-1)(z) + (k_i \cdot \frac{T}{2})(z+1)(z) + (\frac{k_d}{T})(z-1)(z-1)}{(z-1)z}$$

$$= \frac{k_p(z^2-z) + (k_i \cdot \frac{T}{2})(z^2+z) + (\frac{k_d}{T})(z^2-2z+1)}{(z^2-z)}$$

$$= \frac{k_p z^2 - k_p z + (k_i \cdot \frac{T}{2})z^2 + (k_i \cdot \frac{T}{2})z + (\frac{k_d}{T})z^2 - 2(\frac{k_d}{T})z + (\frac{k_d}{T})}{z^2 - z}$$

$$\frac{M(z)}{E(z)} = \frac{(k_p + (k_i \cdot \frac{T}{2}) + \frac{k_d}{T})z^2 + (-k_p + (k_i \cdot \frac{T}{2}) - 2\frac{k_d}{T})z + \frac{k_d}{T}}{z^2 - z} \quad \text{.....(16)}$$

$$= \frac{b_0 + b_1 z^{-1} + b_1 z^{-2}}{1 + a_1 z^{-1} + a_2 z^{-2}}$$

Jika nilai-nilai tersebut disederhanakan dan dikelompokkan dalam sebuah konstanta sebagai berikut:

$$a_1 = -1 \quad \text{.....(17)}$$

$$a_2 = 0 \quad \text{.....(18)}$$

$$b_0 = k_p + k_p \cdot T/2 + k_d/T \quad \text{.....(19)}$$

$$b_1 = -k_p + k_i \cdot T/2 - 2k_d/T \quad \text{.....(20)}$$

$$b_2 = k_d/T \quad \text{.....(21)}$$

Bentuk fungsi M(z) dari persamaan 17 adalah sebagai berikut :

$$M(z) = a_1 \cdot M(z) \cdot z^{-1} + b_0 E(z) + b_1 \cdot E(z) \cdot z^{-1} + b_2 \cdot E(z) \cdot z^{-2} \quad \text{.....(22)}$$

Persamaan beda yang menyatakan persamaan 23 adalah sebagai berikut :

$$m(k) = a_1 \cdot m(k-1) + b_0 \cdot e(k) + b_1 \cdot e(k-1) + b_2 \cdot e(k-2) \quad \text{.....(23)}$$

Dengan :

k = 0, 1, 2, 3, ...

m(k) = keluaran kontroler PID saat ini

m(k-1) = keluaran kontroler *PID* sebelum saat ini

e(k) = error masukan saat ini

e(k-1) = error masukan sebelum saat ini

e(k-2) = error masukan sebelum sebelum saat ini

BAB III

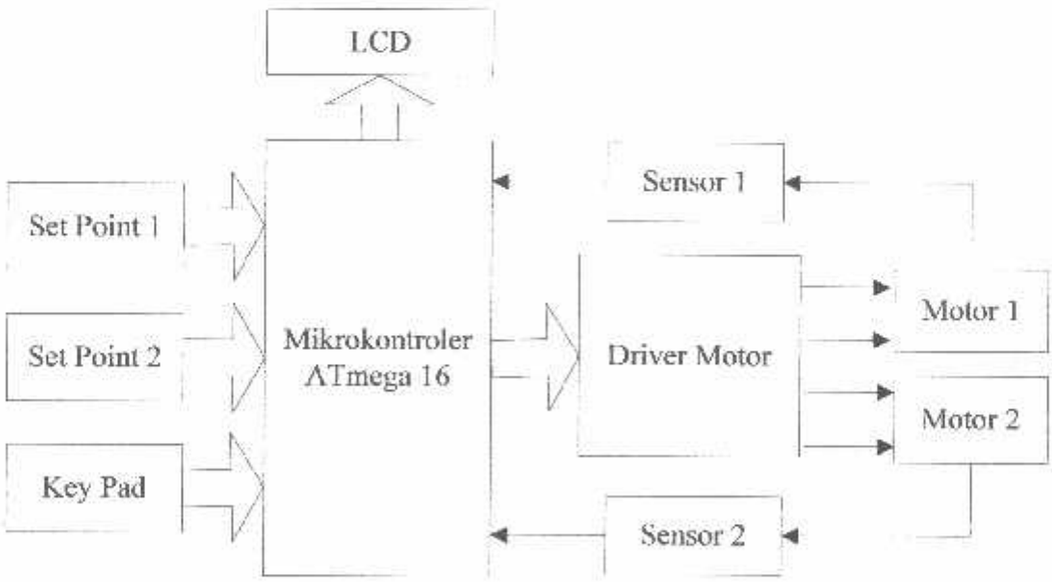
PERANCANGAN DAN PEMBUATAN ALAT

3.1. Pendahuluan

Dalam bab ini akan membahas tentang perencanaan dan pembuatan keseluruhan sistem perangkat keras (*hardware*) dan perangkat lunak (*software*) yang digunakan dalam sistem pengendalian motor DC pada robot manual menggunakan PID kontrol berbasis mikrokontroler ATmega 16.

3.1.1. Blok Diagram Keseluruhan Sistem

Perancangan dan pembuatan alat ditunjukkan dengan gambar blok diagram dibawah ini :



Gambar 3.1. Diagram Blok Keseluruhan Sistem

Keterangan fungsi dari masing-masing blok diagram diatas sebagai berikut :

- **Set Point 1**

  Sebagai input ADC yang dihasilkan dari Variable Resistor 1 (VR 1) berfungsi sebagai pengendali kecepatan dan putaran motor 1.

- **Set Point 2**

  Sebagai input ADC yang dihasilkan dari Variable Resistor 2 (VR 2) berfungsi sebagai pengendali kecepatan dan putaran motor 2.

- **Key Pad**

  Sebagai input nilai Kp, Ti, Td untuk pengaturan PID.

- **LCD**

  Sebagai display inputan dari key pad.

- **Mikrokontroler ATmega16**

  Mikrontroler keluarga AVR ini berfungsi sebagai pengontrol atau pengendali semua sistem yang ada.

- **Driver Motor**

  Sebagai penguat arus yang dikeluarkan oleh pengendali (Mikrokontroler ATmega 16).

- **Motor 1**

  Motor DC 24V yang berfungsi sebagai penggerak roda kanan.

- **Motor 2**

  Motor DC 24V yang berfungsi sebagai penggerak roda kiri.

- **Sensor 1**

  Untuk mengetahui atau mengukur kecepatan putar motor kanan.

- **Sensor 2**

  Untuk mengetahui atau mengukur kecepatan putar motor kiri.

3.2. Prinsip kerja alat

Mikrokontroler akan mendeteksi inputan yang diberikan melalui *joystick* (potensio) yang berupa data analog. Data tersebut diubah menjadi data digital oleh ADC yang terdapat didalam mikrokontroler. Input dari *joystick* merupakan input untuk menggerakkan motor pada robot. Inputan ini memiliki beberapa kondisi seperti di bawah ini :

1. Jika input dari potensio kanan (ADC ≥ 130D) dan input potensio kiri (ADC ≥ 130D) maka robot akan berjalan maju.
2. Jika input dari potensio kanan (ADC ≤ 125D) dan input potensio kiri (ADC ≤ 125D) maka robot akan berjalan mundur.
3. Jika input dari potensio kanan (ADC = 126D – 129D) dan input potensio kiri (ADC = 126D – 129D) maka robot akan berhenti.
4. Jika input dari potensio kanan (ADC ≥ 130D) dan input potensio kiri (ADC ≤ 125D) atau (ADC = 126D – 129D) maka robot akan berputar kekanan.
5. Jika input dari potensio kanan (ADC ≤ 125D) atau (ADC = 126D – 129D) dan input potensio kiri (ADC ≥ 130D) maka robot akan berputar kikiri.

Pada saat motor berputar, sensor kecepatan akan mendeteksi putaran motor. Apakah putaran motor sudah sama dengan input dari *joystick* (*setting point*). Jika belum sama maka nilai input dari *joystick* (*setting point*) akan dikurangi dengan nilai yang dibaca oleh sensor kecepatan dan nilai ini yang disebut dengan nilai

error. Kemudian nilai *error* ini akan diproses dengan menggunakan algoritma PID kontrol hingga nilai *error* ini akan mendekati nol atau sama dengan nol (*error* = 0).

3.3. Perancangan perangkat keras (*hardware*)

3.3.1. Rangkaian sensor kecepatan

Sensor putaran ini terdiri dari *Infra Red* dan *Photodiode*. Rangkaian sensor ini digunakan untuk mendeteksi kondisi gelap dan terang yang kemudian digunakan untuk mendeteksi kecepatan motor.

3.3.1.1. *Infra red*

Led *infra red* merupakan dioda yang mampu memancarkan cahaya *infra* merah. Berdasarkan data yang diperoleh dari *databook* terdapat penurunan tegangan LED IR (V_F)=1,2 volt, arus forward(I_F)=20mA, dan Vcc yang digunakan ialah 5 volt, maka R_1IR dapat diperoleh dengan menggunakan rumus :

$$R_1IR = \frac{(Vcc - V_F)}{I_F}$$

$$= \frac{(5-1{,}2)}{20.10^{-3}} = 190\Omega = 220\Omega$$

3.3.1.2. *Photodiode*

Pada saat menerima sinar *infra* merah dari pemancar, *photodiode* akan mempunyai resistansi yang rendah dan arus yang besar, sehingga tegangan output(V_{out}) pada kanota terhadap *ground* bernilai nol, sebaliknya apabila

photodiode tidak menerima sinar *infra* merah,maka photodiode seperti dioda yang di-bias mundur dan *photodiode* seperti dalam keadaan terbuka (*reverse*).

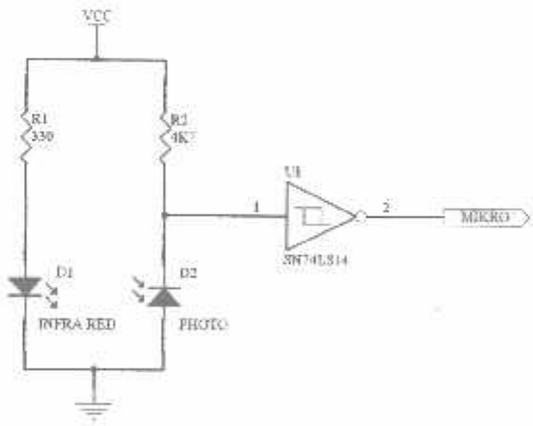
Berdasarkan data yang diperoleh dari *databook*, arus *reverse* pada saat ada cahaya yang mengenai *photodiode*(I_R) = 100μA dengan Vcc yang digunakan adalah 5 volt, maka nilai R_LPD dapat diperoleh dengan menggunakan persamaan(3.2).

$$RLPD = \frac{(Vcc - VD)}{IL}$$

$$= (5 - 0{,}7)/1 \times 100^{-6}$$

$$= 43000\ \Omega = 43\ k\Omega \equiv 47\ k\Omega$$

Rangkaian detektor *infra* merah dan *photodiode* ditunjukkan dalam gambar 3.2. dibawah ini.



Gambar 3.2. Rangkaian sensor kecepatan

Prinsip kerja dari rangkaian *infra* merah adalah bila *photodiode* terhalang maka tahanan/resistansi dari *photodiode* akan bertambah besar dan arus yang

dihasilkan adalah kecil sehingga tegangan output pada *photodiode* (pada katoda) ialah 5 volt. Apabila *photodiode* tidak terhalang sehingga sinar *infra* merah mengenai *photodiode*, maka tahanan/resistansi *photodiode* semakin kecil, dan arus yang dihasilkan semakin besar mengakibatkan tegangan output pada *photodiode* (pada katoda) ialah 0 volt.

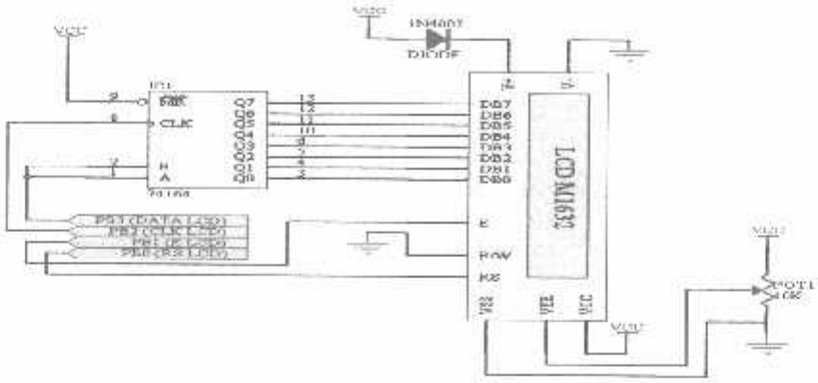
Jarak maksimal sinar inframerah dapat diterima oleh *photodiode* adalah ± 21 cm. Pada pembuatan dan perancangan sensor kecepatan ini alat jarak detektor *infra* merah dan *photodiode* dibuat sekitar 1 cm. Jadi dapat dikatakan jarak ini sudah masuk dalam lingkup kerja dari detektor *infra* merah dan *photodiode*.

Agar sinyal yang akan masuk ke mikrokontroler dapat terhindar dari *noise*, pada masing-masing output detektor masuk pada rangkaian *Schmitt trigger*.

3.3.2. LCD (*Liquid Crystal Display*)

Pada perancangan ini digunakan LCD *dot matrix* 2 x 16 karakter yaitu M1632. Sinyal-sinyal yang diperlukan oleh LCD adalah RS dan *Enable*, sinyal RS dan Enable dipergunakan sebagai input yang outputnya dipakai untuk mengaktifkan LCD. LCD akan aktif apabila mikrokontroler memberikan instruksi tulis pada LCD. Saat kondisi RS *don't care* dan *Enable* 0 maka LCD tetap pada kondisi semula, pengiriman data ke LCD dilakukan saat RS berlogika 0 dan enable berlogika 1. Instruksi dikirim pada LCD bila keadaan RS 1 dan *Enable* 1. Pin LCD ini untuk data terkoneksi pada *Port* B.3 mikrokontoler ATMega 16. Kemudian untuk RS dihubungkan pada *Port* B.0, tulis/baca (*Read/Write*) diberikan logika *low* karena disini LCD bersifat menulis data, dan yang terakhir

Enable (E) dikendalikan dengan *Port* B.1. Gambar rangkaian LCD ditunjukkan pada gambar 3.3.

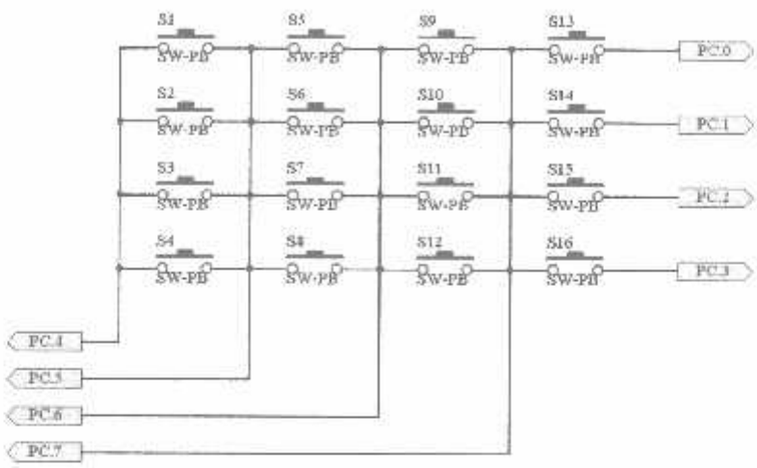


Gambar 3.3. Perancangan Rangkaian LCD (*Liquid Crystal Display*)

3.3.3. Keypad

Keypad ini berfungsi untuk memberikan nilai input pada mikrokontroler yang digunakan untuk merubah dan memasukkan data nilai Kp, Ti dan Td pada program PID kontrol. Pada perancangan ini digunakan keypad 4x4 yang berarti papan tombol ini mempunyai matrik 4 baris dan 4 kolom. Deretan kolom dan baris dari keypad ini dihubungkan dengan *port* C yang difungsikan sebagai masukan dan keluaran. Deretan kolom dihubungkan dengan *ground* (berlogika 0) dan *port* C (PC4 - PC7) yang difungsikan sebagai input mkrokontroller. Sedangkan deretan baris dihubungkan ke *port* C (PC0 - PC3) yang telah diberi data 0001 dan secara kontinyu data tersebut bergeser satu bit ke kiri. Pergeseran data satu bit ini dimaksudkan untuk menentukan posisi tombol yang ditekan dalam satu kolom. Port ini difungsikan sebagai output dari mikrokontroller. Dengan demikian kalau tombol tidak ditekan maka masukan *port* C (PC4 - PC7)

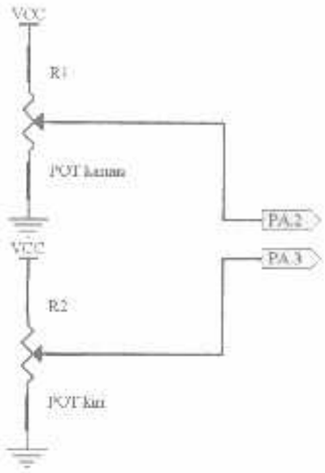
di pin yang terhubung tombol tersebut berlogika 0 dan bila tombol ditekan akan berlogika 1. Rangkaian papan tombol tersebut dapat terlihat pada Gambar 3.4.



Gambar 3.4. Rangkaian Keypad

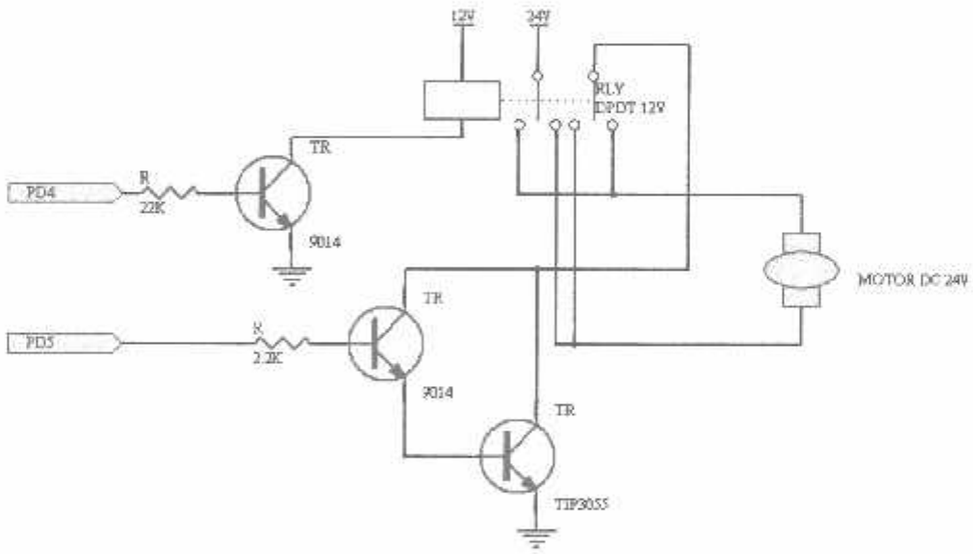
3.3.4. Joystick

Joystick berfungsi sebagai input pada mikrokontroler yang digunakan untuk mengendalikan arah putaran dan kecepatan motor DC pada robot. Pada perancangan ini digunakan *variable* resistor (potensio) sebagai input dari PWM (*Pulse Width Modulation*). Input dari potensio yang berupa data analog diubah oleh ADC yang terdapat dalam mikrokontroler sehingga data tersebut bisa dimengerti oleh mikrokontroler untuk diproses lebih lanjut. Rangkaian joystick dapat dilihat pada gambar 3.5.



Gambar 3.5. Rangkaian *Joystick*

3.3.5. Driver Motor

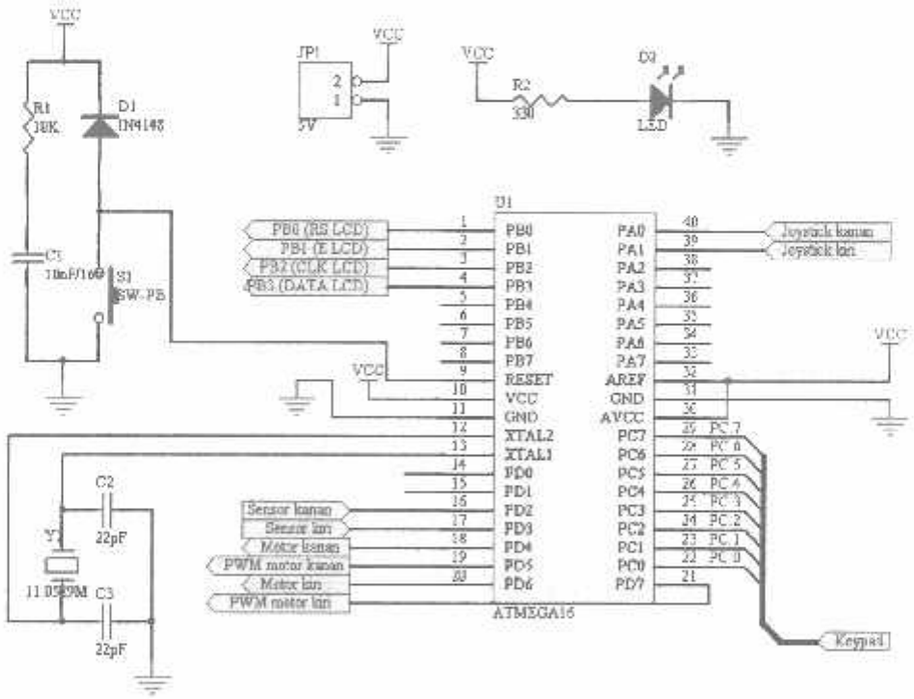


Gambar 3.6. Rangkaian driver motor

Rangkaian ini digunakan untuk menguatkan output arus dari mikrokontroler sehingga output arus yang sangat kecil dari mikrokontroler dapat menggerakkan relay dan memutar motor DC. Disamping itu terdapat juga 2

transistor yang berfungsi sebagai input data PWM. Sebagai rangkaian motor DC, rangkaian ini dapat memutar arah motor DC dengan cara memberikan logika 0/1 (*high / low*) pada *software* mikrokontroler. Jika driver motor diberi logika "1" (*high*) maka relay akan aktif dan motor akan berputar kekanan. Dan apabila driver motor diberi logika "0" (*low*) maka motor akan berputar kekiri. Kecepatan putar motor diatur dari input nilai PWM, semakin lebar pulsa PWM maka motor akan berputar semakin cepat (*duty cycle* = 100 %) dan semakin kecil nilai PWM maka motor akan berhenti berputar (*duty cycle* = 0 %).

3.3.6. Perancangan minimum sistem ATMega16



Gambar 3.7. Rangkaian minimum system ATMega 16

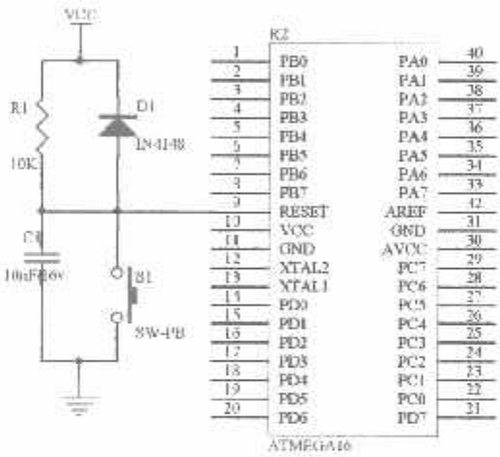
Mikrokontroler yang digunakan adalah mikrokontroler keluarga AVR ATMega 16 yang mempunyai arsitektur *RISC(Reduce Instruction Set)*. ATMega 16 mempunyai 4 *channels* PWM, 8 *channels* ADC 10-bit, dan juga ATMega 16 mempunyai kemampuan PWT (*Programmable Watchdog Timer*) dengan osilator yang terpisah, sehingga memungkinkan untuk mengaplikasikannya sebagai MCU yang cukup handal. Selain itu dengan adanya fitur-fitur tersebut, system hardware juga makin simpel karena tidak perlu lagi membuat rangkaian ADC dan PWM sehingga biaya pembuatan juga dapat lebih ekonomis.

Alokasi penggunaan pin mikrokontroler :

- PA.0 dan PA.1 digunakan sebagai *port* input data analog dari *joystick*. *Port* A merupakan *port* khusus yang didalamnya memiliki ADC sehingga data analog dari *joystick* diubah oleh ADC di *port* A menjadi data digital.
- PB.1 terhubung dengan kaki *Enable* (E) pada LCD dan PB.0 terhubung dengan kaki RS pada LCD.
- *Port* B.3 digunakan sebagai *port* input dan output data LCD serial.
- *Port* C (PC.0 – PC.7) digunakan sebagai *port* input data dari *keypad*.
- PD.2 dan PD.3 digunakan sebagai port input yang akan menerima sinyal digital dari *inverter schmitt trigger* yang terhubung pada sensor kecepatan.

- PD.5 dan PD.6, digunakan sebagai *port* output data motor yang terhubung dengan driver motor.
- PD.4 dan PD.7 digunakan sebagai *port* output data PWM yang terhubung dengan driver motor.

Perancangan rangkaian reset pada mikrokontroler ATMega 16 ialah dengan memberikan logika low pada pin reset mikrokontroler ATMega 16. Rangkaian reset ini diperoleh dari *application note AVR Design Consideration* dari ATMEL. Berikut ialah gambar rancangan rangkaian reset pada ATMega 16 :



Gambar 3.8. Rangkaian reset

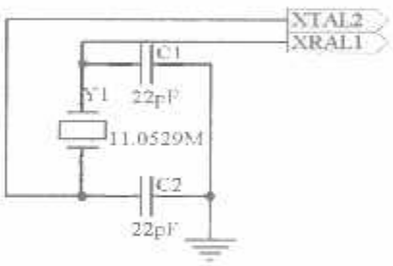
Osilator pada rangkaian minimum sistem ATMega 16 menggunakan kristal 11,0592 MHz dan kapasitor 22 pF. Nilai kapasitor ini diperoleh dari tabel datasheet tentang penggunaan kapasitor untuk rangkaian osilator / sistem clock pada ATMega 16. Penggunaan kristal 11,0592 MHz ini bertujuan agar berhitungan bautrate tidak mengalami error yang disebabkan karena selisih

perhitungan. Perhitungan bautrate pada ATMega 16 dengan menggunakan kristal 11,0592 MHz :

Bautrate yang diinginkan ialah 38400 bps, maka nilai pada *UBRR(USART Baut Rate Register)* dapat ditentukan dengan perhitungan :

$$UBRR = \frac{fosc}{16.Baud} - 1$$
$$UBRR = \frac{11059200}{16.38400} - 1$$
$$UBRR = \frac{11059200}{614400} - 1$$
$$UBRR = 18 - 1 = 17 = 11H$$

Penggunaan kristal 11,0592 MHz memungkinkan hasil perhitungan *baudrate* tidak sisa dan *error* dari selisih perhitungan tidak ada.



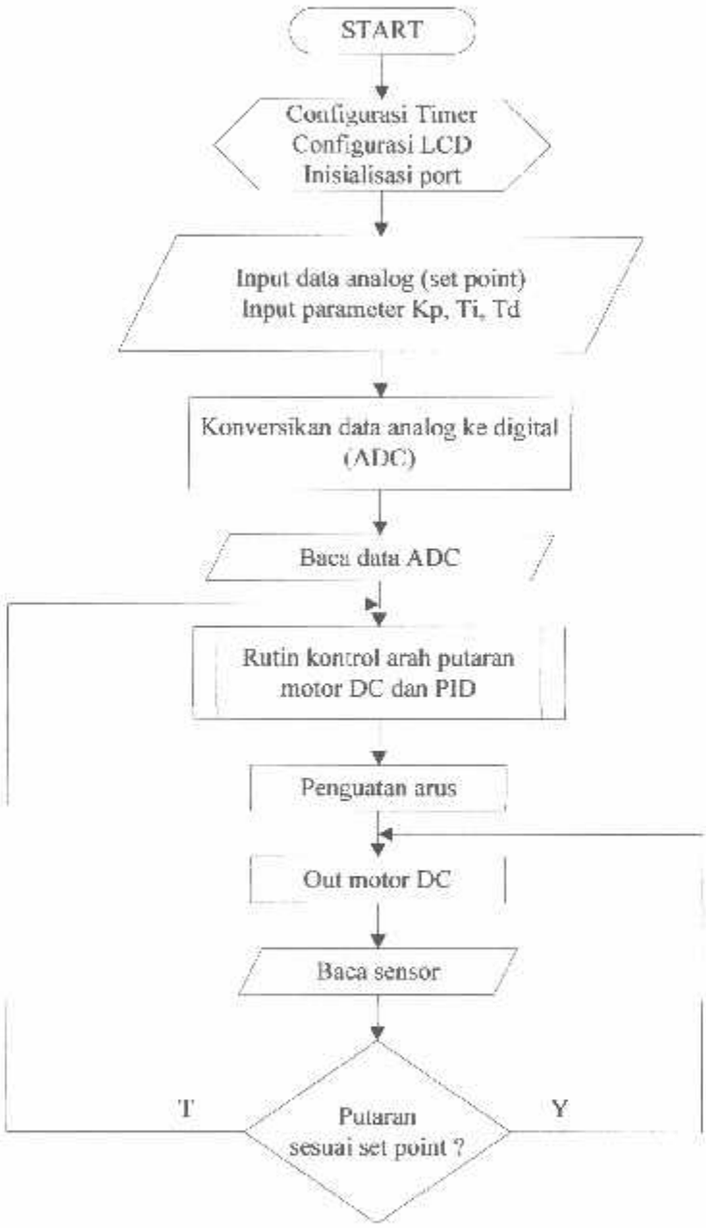
Gambar 3.9. Rangkaian clock

3.4. Perancangan perangkat lunak (*Software*)

Perancangan perangkat lunak (*software*) yang digunakan dalam perancangan sistem pengendalian motor DC pada robot manual menggunakan PID kontrol berbasis mikrokontroller ATMega 16 akan dipaparkan dalam *flowchart* sistem secara keseluruhan. Sistematika jalannya program yang dibuat

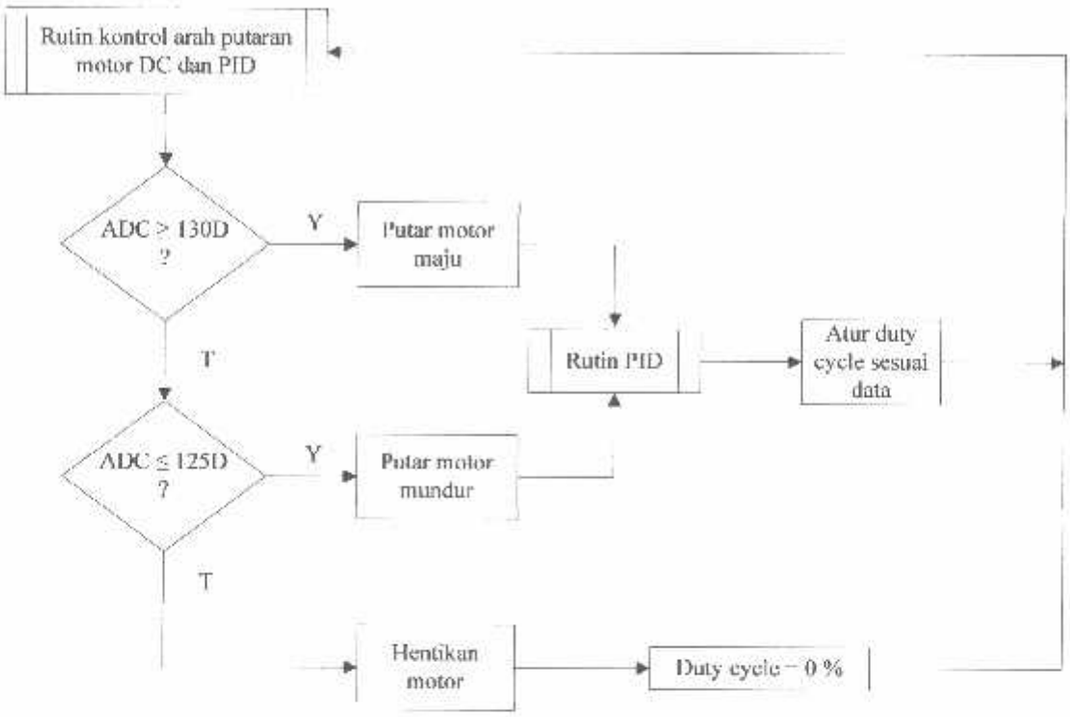
didasarkan pada sistem *hardware* yang telah dirancang. Pembuatan *software* hanya dilakukan pada mikrokontroler menggunakan bahasa C.

3.4.1. Diagram Alir (*Flow Chart*)



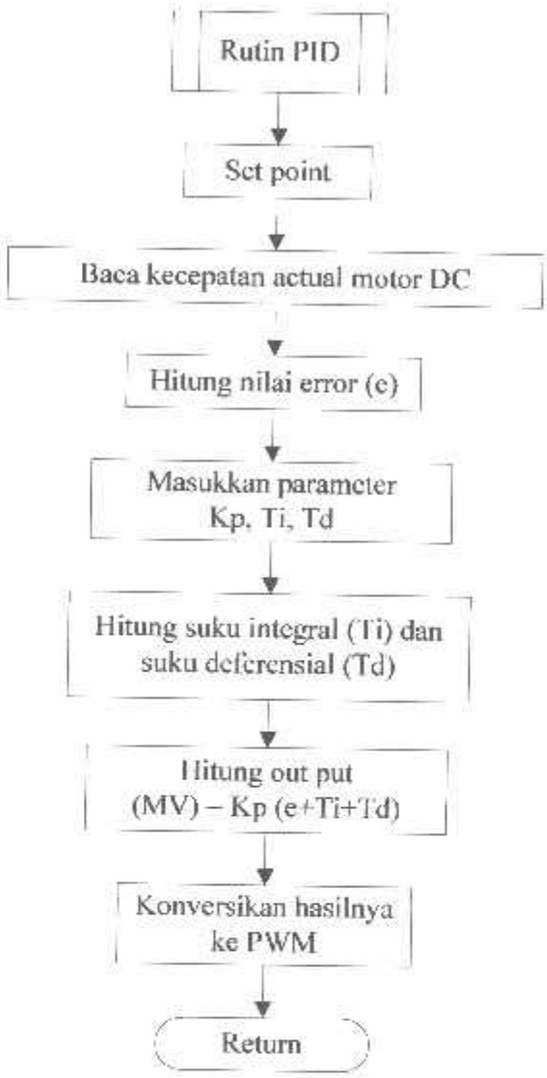
Gambar 3.10. Flow chart total sistem

- **Algoritma rutin kontrol arah putaran motor DC dan PID**



Gambar 3.11. Algoritma rutin kontrol arah putaran motor DC dan PID

- **Algoritma rutin PID**



Gambar 3.12. Algoritma rutin PID

➢ Penjelasan *flowchart* secara keseluruhan :

1. Pertama kali *software* akan melakukan pengenalan terhadap mikrokontroller dan kristal yang digunakan, pada perancangan *software* ini menggunakan mikrokontroller ATMega 16 dan kristal 11.0592MHz. Selanjutnya program mengkonfigurasi pin-pin LCD, jenis LCD yang dipakai, dan juga mendefinisikan karakter baru yang dipakai. Kemudian program mengkonfigurasi timer agar timer pada mikrokontroller ATMega 16 dikenali dan dapat digunakan.
2. Masukkan data analog dari potensio kemudian data tersebut diubah menjadi data digital oleh ADC yang terdapat didalam mikrokontroler schingga data tersebut bisa proses lebih lanjut oleh mikrokontroler.
3. Jika data ADC ≥ 130D maka putar motor maju, jika data ADC ≤ 125D maka putar motor mundur dan jika ADC antara 126D – 129D maka motor tidak berputar (berhenti). Atur d*uty cycle* sesuai data dan *out* putaran motor.
4. Baca kecepatan motor dan hitung nilai *error*. *Error* diperoleh dengan mengurangkan kecepatan aktual dari *set point*. Kemudian masukkan parameter-parameter Ki, Ti,dan Td melalui *keypad* dan nilainya akan ditampilkan di LCD.
5. Hitung suku integral (Ti) dan suku deferensial (Td). Kemudian hitung out put (MV). Konversikan hasil dari hitungan tersebut ke PWM dan program akan kembali ke pengaturan duty cycle.

BAB IV

PENGUJIAN DAN HASIL ANALISA

Pengujian dilakukan untuk mengetahui sejauh mana perlatan dapat bekerja sesuai dengan perencanaan.Langkah pengujian dilakukan melalui 2 tahap, yakni pengujian pada setiap blok dan pengujian pada sistem keseluruhan. Tahap pertama dimaksudkan untuk mengetahui sejauh mana blok-blok rangkaian dapat berjalan, sedangkan tahap kedua dilakukan setelah diperoleh kepastian bahwa tiap blok rangkaian telah berjalan sesuai rencana. Pada tahap ini setiap blok rangkaian diintegrasikan menjadi sebuah sistem pangendalian motor DC yang kemudian dilakukan pengujian secara menyeluruh.

4.1. Pengujian Mikrokontroller ATMega 16

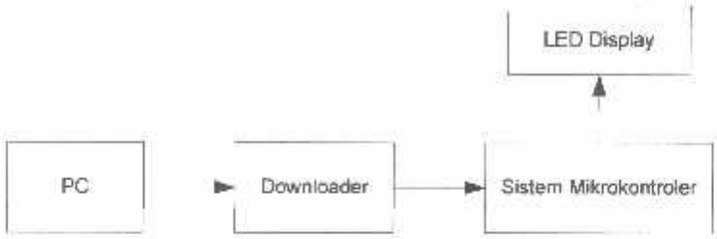
4.1.1. Tujuan

Untuk mengetahui kondisi awal dari Mikrokontroler apakah sudah sesuai yang direncanakan.

4.1.2. Peralatan yang digunakan

1. Catu daya 5V.
2. Komputer.
3. Downloader.
4. Minimum sistem mikrokontroler ATMega16.

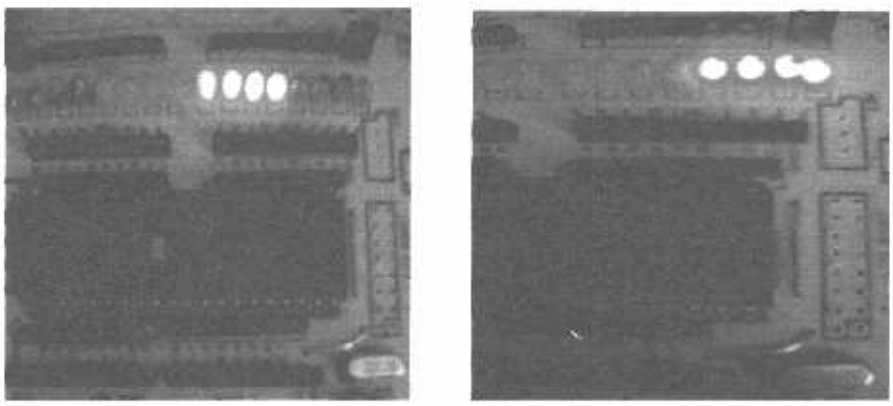
4.1.3. Prosedur pengujian



Gambar 4.1. Diagram Blok Pengujian Mikrokontroler

1. Rangkaian dibuat seperti gambar 4.1.
2. Memberikan catu daya 5 volt.
3. Membuat program yang digunakan untuk menguji mikrokontroler. Program yang digunakan dalam pengujian mikrokontroler ini merupakan program yang sederhana yang meletakkan $F0_H$ dan $0F_H$ secara bergantian pada *PortC* ATMega 16.
4. Mengamati keluaran pada LED Display.

4.1.4. Hasil pengujian



Gambar 4.2. Hasil pengujian tampilan LED

Tabel 4.1. Hasil Pengujian Sistem Mikrokontroler

Kondisi	Keluaran pada LED *Display*							
	Bit 0	**Bit 1**	**Bit 2**	**Bit 3**	**Bit 4**	**Bit 5**	**Bit 6**	**Bit 7**
Satu	0	0	0	0	1	1	1	1
Dua	1	1	1	1	0	0	0	0

Keterangan :

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

Dari hasil pengujian pada tabel 4.1. dapat dilihat bahwa *port C* memberikan logika $0F_H$ dan $F0_H$ secara bergantian sesuai dengan isi program.

4. 2. Pengujian sensor kecepatan

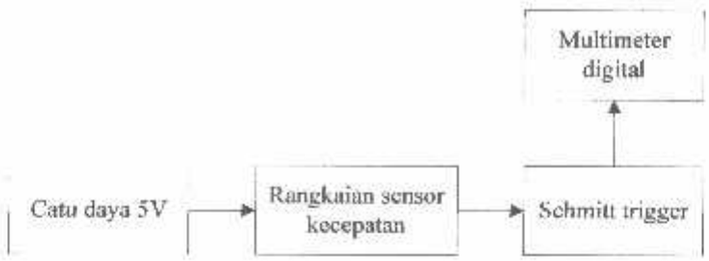
4.2.1. Tujuan

Untuk mengetahui apakah sensor dapat bekerja sesuai dengan rancangan.

4.2.2. Peralatan yang digunakan

1. Catu daya 5V.
2. Rangkaian sensor dan Schmitt trigger.
3. Multimeter digital.

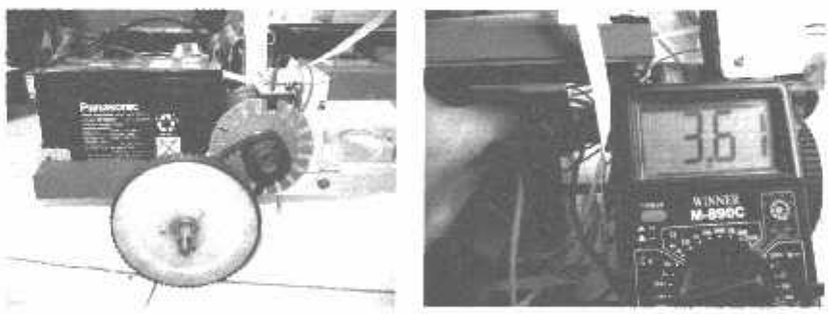
4.2.3. Prosedur pengujian



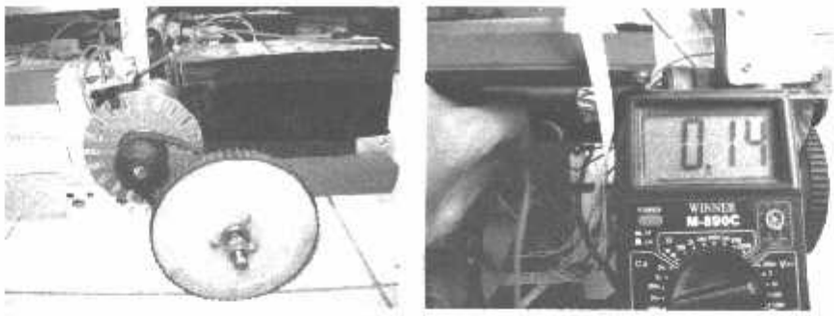
Gambar 4.3. Diagram blok pengujian rangkaian sensor kecepatan

1. Rangkaian dibuat sesuai dengan gambar 4.3.
2. Memberi catu daya 5V.
3. Mengukur tegangan keluaran *photodiode* dengan *AVO meter* apabila pancaran sinar *infra* merah terhalang piringan atau tidak terhalang.

4.2.4. Hasil pengujian



Gambar 4.4. Pengujian rangkaian sensor saat tidak terhalang piringan



Gambar 4.5. Pengujian rangkaian sensor saat terhalang piringan

Table 4.2. Hasil pengujian tegangan *output.*

Sensor	Tegangan Output	
	Terhalang piringan (mV)	Tidak terhalang piringan (mV)
1	0.14	3.61
2	0.14	3.61
Rata-rata	0.14	3.61

4.3. Pengujian LCD

4.3.1. Tujuan

Untuk mengetahui apakah LCD dapat bekerja sesuai dengan rancangan

4.3.2. Peralatan yang digunakan

1. Catu daya 5V.
2. Minimum sistem mikrokontroler dan LCD (*Liquid Crystal Display*).

4.3.3. Prosedur pengujian

Gambar 4.6. Rangkaian LCD (*Liquid Crystal Display*)

1. Rangkaian dibuat seperti gambar 4.6.
2. Memberi catu daya 5V.
3. Membuat program yang digunakan untuk menguji LCD. Program yang digunakan dalam pengujian LCD merupakan program yang sederhana yaitu menuliskan nama pada baris pertama dan menuliskan NIM pada baris ke dua.
4. Mengamati keluaran pada LCD.

4.3.4. Hasil pengujian

Setelah data diolah mikrokontroler maka hasil tampilan LCD berupa tulisan pada baris pertama "Er!cK" dan baris ke dua "0412232".

Gambar 4.7. Tampilan hasil pengujian LCD

4.4. Pengujian driver motor DC

4.4.1. Tujuan

untuk mengetahui apakah rangkaian ini dapat bekerja sebagaimana mestinya, yaitu dapat menggerakkan motor dengan arah yang bersesuaian dengan kombinasi masukannya.

4.4.2. Peralatan yang digunakan

1. Catu daya 24V.
2. Rangkaian driver motor.
3. Multimeter digital.

4.4.3. Prosedur pengujian

Gambar 4.8. Diagram blok pengujian driver motor

1. Rangkaian dibuat seperti gambar 4.8.
2. Memberi catu daya 24V.
3. memberi kombinasi logika pada inputan driver dan mengamati arah putaran motor DC pada keluarannya.
4. Mengukur tegangan pada keluaran driver.

4.4.4. Hasil pengujian

Gambar 4.9. Pengujian rangkaian driver motor

Table 4.3. Hasil pengujian rangkaian driver motor

No	Input ADC (desimal)	Output	
		Tegangan (V)	Arus (A)
1.	255	22.65	4.02
2.	240	22.65	3.56
3.	225	22.65	3.12
4.	210	22.65	2.65
5.	195	22.65	2.23
6.	180	22.65	1.74
7.	165	22.65	1.34
8.	150	22.65	0.85
9.	135	22.65	0.43
10.	126 – 129	0.724	0.724
11.	120	- 22.65	0.41
12.	105	- 22.65	0.86
13.	90	- 22.65	1.33
14.	75	- 22.65	1.74
15.	60	- 22.65	2.22
16.	45	- 22.65	2.65
17.	30	- 22.65	3.11
18.	15	- 22.65	3.56
19.	0	- 22.65	4.03

Keterangan Tabel :

- Input tegangan 24V
- Input merupkan data analog dari joystick yang kemudian diubah menjadi data digital oleh ADC mikrokontroler.

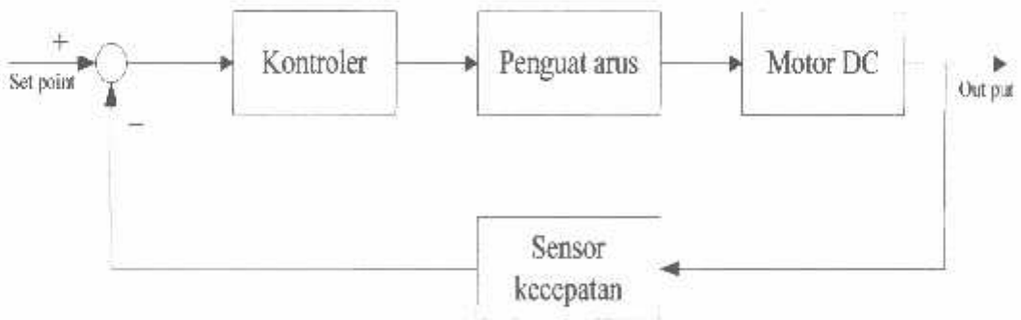
4.5. Pengujian pengendalian kecepatan motor DC pada robot manual menggunakan PID kontrol

4.5.1. Tujuan

Untuk mengetahui apakah keseluruhan sistem dapat berjalan sesuai dengan rancangan atau tidak.

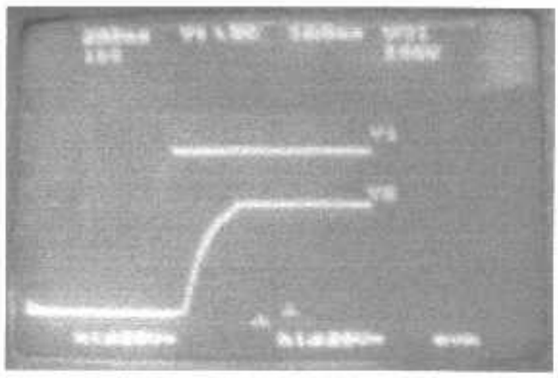
4.5.2. Prosedur pengujian

1. Menghubungkan keseluruhan rangkaian sesuai dengan diagram blok.
2. Memberikan nilai input Kp, Ti dan Td melalui Keypad.
3. Memberikan nilai set point melalui joystick.
4. Mengamati apakah sistem berjalan sesuai dengan rancangan atau tidak.

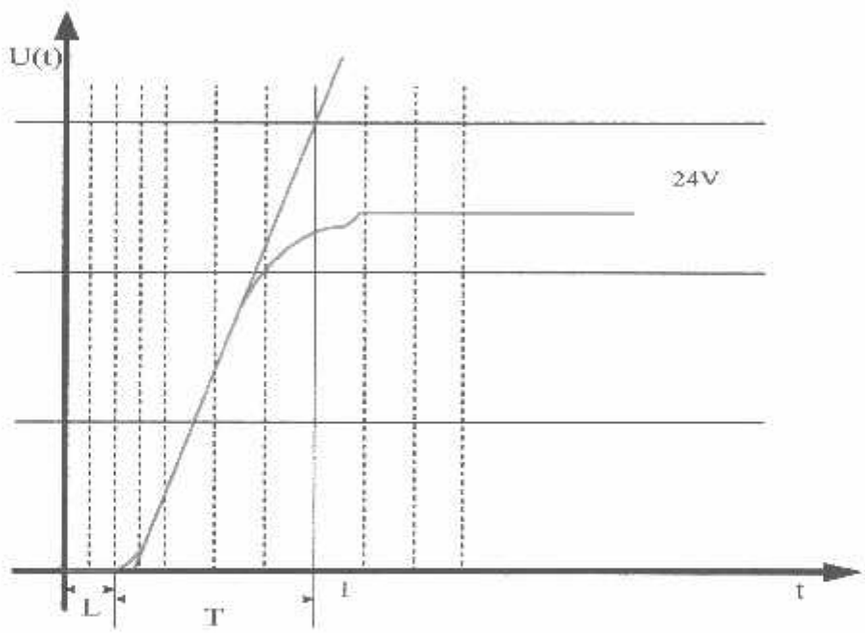


Gambar 4.10. Rangkaian pengujian sistem

4.5.3. Hasil kurva motor DC sebelum diberi PID kontroler



Gambar 4.11. Kurva respon motor DC sebelum diberi PID kontroler



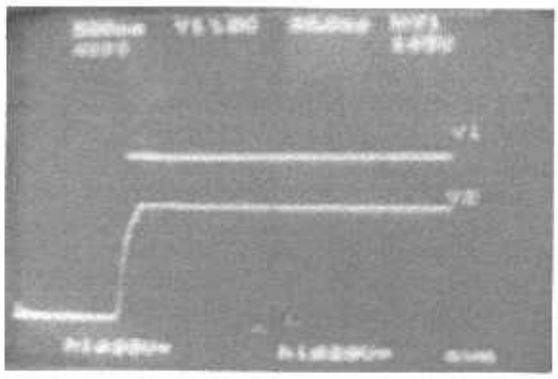
Gambar 4.12. Garis singgung kurva sebelum diberi PID kontroler

Setelah ditarik garis dari gambar 16. diketahui bahwa nilai L adalah 0,2 sedangkan nilai dari tunda waktu T adalah 1 - 0,2 = 0,8 dengan Time/Div = 200 ms maka didapat :

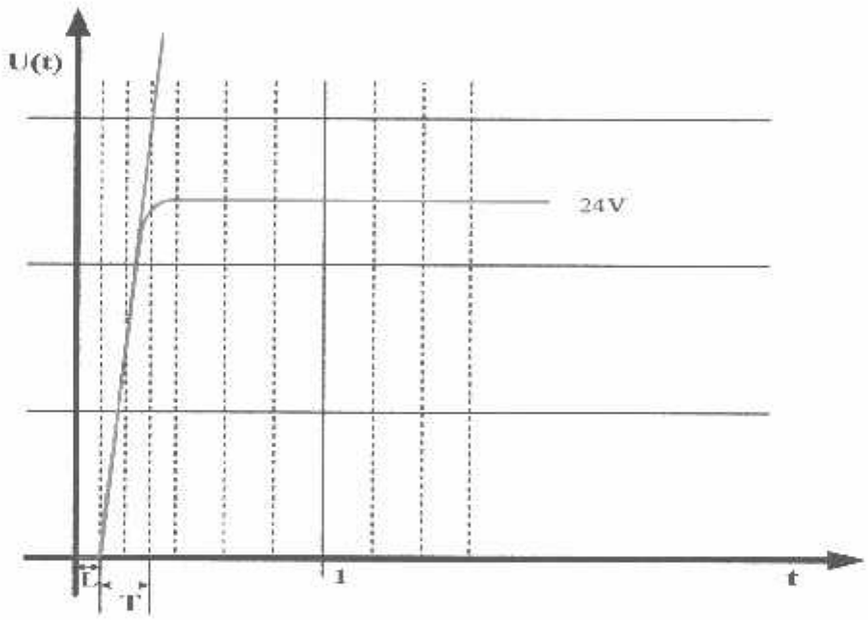
L – 0,2 x 200 ms – 40 ms

T = 0,8 x 200 ms = 160 ms

4.5.4. Hasil kurva motor DC dengan PID kontroler



Gambar 4.13. Kurva respon motor DC dengan PID kontroler



Gambar 4.14. Garis singgung kurva dengan PID kontroler

Dari gambar 18. dapat kita analisa bahwa data hasil pengujian pada PID kontroler yang menggunakan oscioloscop dengan parameter Time/Div = 200 ms maka didapat :

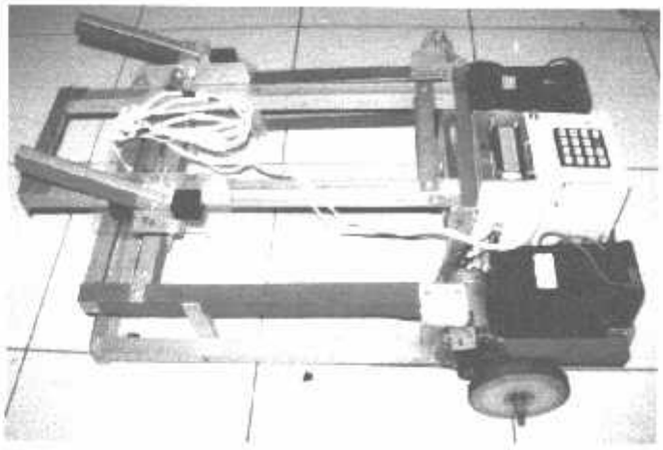
L = 0,1 x 200 ms = 20 ms

T = 0,2 x 200 ms = 40 ms

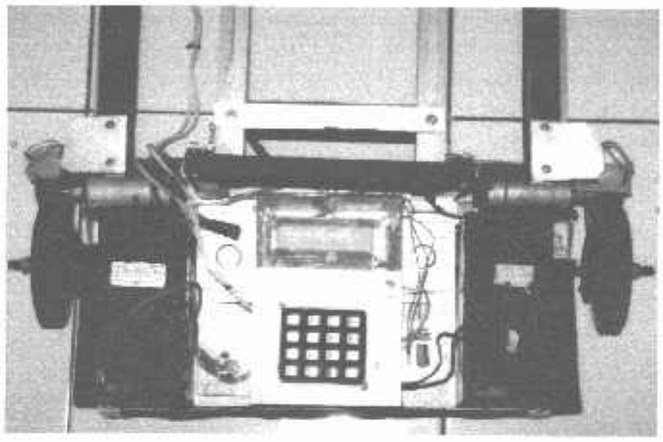
Dari gambar diatas dapat dibuktikan bahwa PID kontroler yang dipergunakan dalam skripsi ini mampu mengurangi waktu tunda (T) sehingga tanggapan respon motor semakin cepat untuk mencapai *steady state* yang mendekati *setting point* yang telah ditetapkan. Dapat kita lihat bahwa kurva memiliki *rise time* yang lebih kecil dibanding sebelumnya, akan tetapi memiliki sedikit *overshot*. Dengan PID kontroler respon terlihat berbanding lurus dengan *setting point* yang telah ditentukan. Pada saat motor start, motor akan sedikit ada lonjakan akan tetapi lebih cepat stabil. Hal ini menandakan bahwa PID kontroler merupakan suatu metode yang sangat bagus dalam pengaturan kecepatan motor.

4.6. Spesifikasi alat

1. Mikrokontroler ATmega 16.
2. LCD M1632.
3. Keypad matriks 4 x 4.
4. Optocoupler model U.
5. Potensio sebagai joystick.
6. Accu 24V (12Vx2).
7. Robot.



(a)



(b)

Gambar 4.15. (a) dan (b) Alat keseluruhan

BAB V

PENUTUP

5.1. Kesimpulan

Dari rangkaian kegiatan perencanaan sampai dengan pengujian keseluruhan sistem yang telah dibuat maka dapat diambil beberapa kesimpulan sebagai berikut :

1. Output tegangan driver motor sebesar 22.65 volt dan memiliki arus sebesar 4.02 ampere.
2. Motor tidak berputar pada saat 126 ≤ ADC ≤ 129 desimal karena output tegangan driver motor sebesar 0.724 volt dan memiliki arus sebesar 0.01 ampere.
3. Dibawah ini merupakan kurva motor sebelum dan sesudah dikontrol :

   - Data kurva sebelum dikontrol

     L = 0,2 x 200 ms = 40 ms

     T = 0,8 x 200 ms = 160 ms

   - Data kurva sesudah dikontrol

     L = 0,1 x 200 ms = 20 ms

     T = 0,2 x 200 ms = 40 ms

Dari data diatas terdapat selisih waktu mati (L) = 20 ms dan selisih waktu tunda (T) = 120 ms. Sehingga respon motor untuk mencapai *steady state* menjadi semakin cepat.

5.2. Saran - saran

Ada beberapa saran yang dapat kami berikan kapada pembaca yang tertarik dengan skripsi ini bahkan ingin mengembangkannya.

- Untuk menghasilkan respon yang baik diperlukan perhitungan yang tepat.
- Banyaknya lubang pada piringan sensor kecepatan sangat mempengaruhi ketelitian pembacaan kecepatan.

DAFTAR PUSTAKA

[1]. *Frans Gunterus, Falsafah Dasar : Sistem Pengendalian Proses.*

[2]. *http://puslit.petra.ac.id/journals/electrical/pid.pdf*

[3]. *http://www.innovativeelectronics.com/innovative_electronics/download_files/artikel/AN116.pdf*

[4]. *http://elektronika-elektronika .blogspot.com*

[5]. *Lingga Warhana, Belajar Sendiri Mikrokontroler AVR Seri ATMega8535 Simulasi, Hardware, dan Aplikasi.*

[6]. *www.atmel.com , data sheet ATMega16*

[7]. *www.parallax.com , LCD M1632*

LAMPIRAN

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Erik Budhiawan
Nim : 04.12.232
Masa Bimbingan : 17-Juni-2008 s/d 17-Desember-2008
Judul Skripsi : Perancangan Sistem Pengendalian Motor DC Pada Robot Manual Menggunakan PID Kontrol Berbasis Mikrokontroler ATmega16

No	Tanggal	Uraian	Paraf Pembimbing
1	21/08 '08	Bab I . Rumusan Masalah	
2	29/08 '08	Bab III . Rangkaian Driver Motor	
3	01/09 '08	Revisi Makalah Seminar Hasil	
4	19/09 '08	Acc. lembar Persetujuan Skripsi	
5			
6			
7			
8			
9			
10			

Malang,

Dosen pembimbing II

I Komang Somawirata, ST, MT
Nip. 1030100361

Form S-4b

INSTITUT TEKNOLOGI NASIONAL
Jl. Bendungan Sigura – gura No.2
MALANG

FORMULIR BIMBINGAN SKRIPSI

Nama : Erik Budhiawan
Nim : 04.12.232
Masa Bimbingan : 17-Juni-2008 s/d 17-Desember-2008
Judul Skripsi : Perancangan Sistem Pengendalian Motor DC Pada Robot Manual Menggunakan PID Kontrol Berbasis Mikrokontroler ATmega16

No	Tanggal	Uraian	Paraf Pembimbing
1	21/08 '08	Revisi bab I, II, III, IV	
2	29/08 '08	Detail Perangkat Lunak	
3	10/09 '08	Listing Program	
4	12/09 '08	Revisi flow Chart Sistem	
5	15/09 '08	Rangkaian pengujian Sistem	
6	18/09 '08	Gambar Rangkaian Seluruh Sistem	
7	19/09 '08	Acc. lembar persetujuan Skripsi	
8			
9			
10			

Malang,
Dosen pembimbing I

Ir. Widodo Pudji M, MT
Nip. 1028700171

Form S-4b

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

FORMULIR PERBAIKAN SKRIPSI

Nama	:	Erik Budhiawan
NIM	:	04.12.232
Jurusan	:	Teknik Elektro S-I
Konsentrasi	:	Teknik Elektronika
Judul Skripsi	:	**Perancangan Sistem Pengendalian Motor DC Pada Robot Manual Menggunakan PID Kontrol Berbasis Mikrokontroler ATMega16**
Hari / Tanggal Ujian Skripsi	:	Rabu, 24 September 2008

No	Tanggal	Uraian	Paraf
1.	24/09/2008	Teori dari PID	

Disetujui,

Penguji I

Soryohadi, ST, MSc
NIP.Y. 1039700309

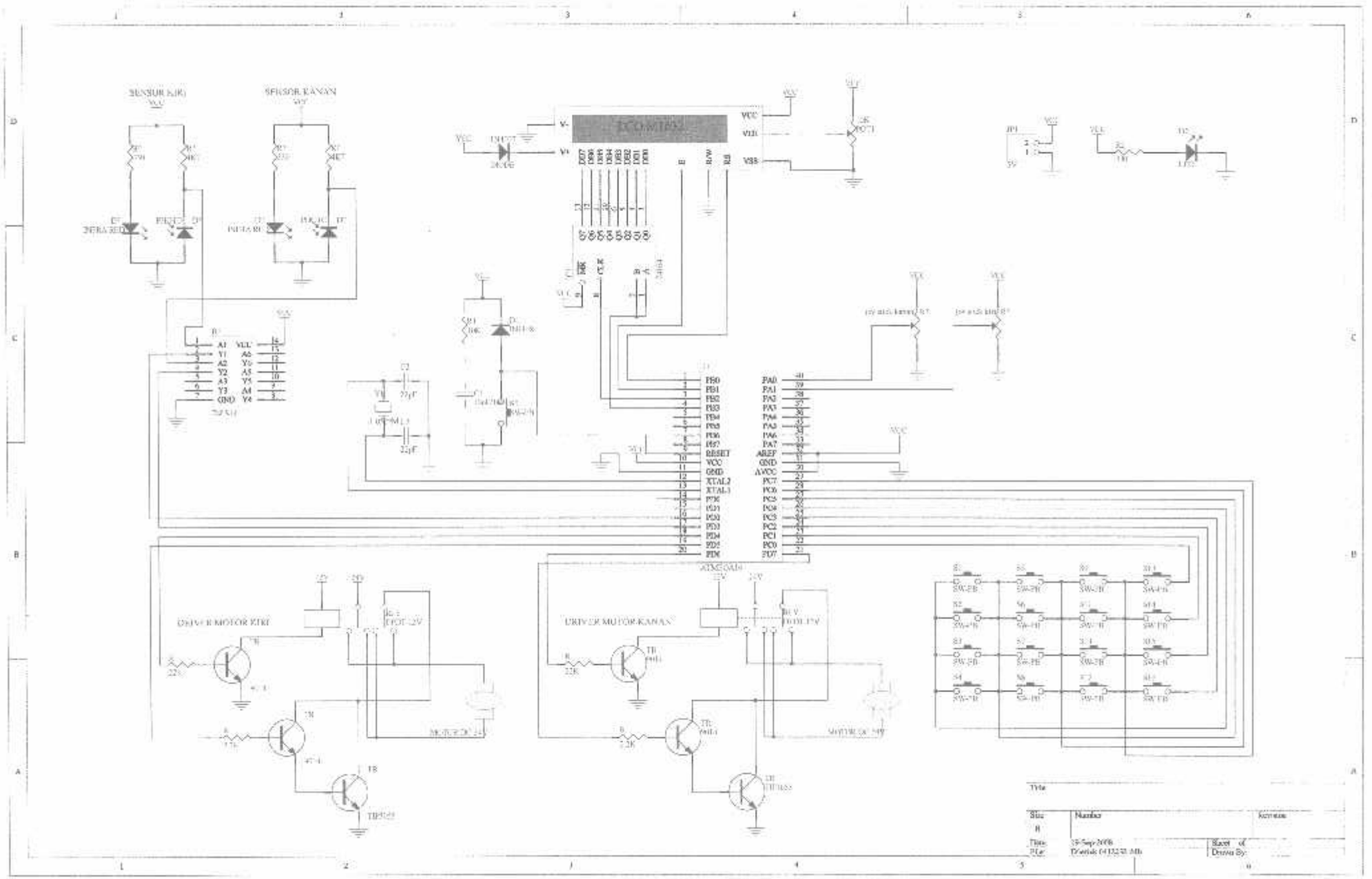
Mengetahui,

Dosen Pembimbing I

Ir. Widodo Pudji M, MT
NIP.Y. 1028700171

Dosen Pembimbing II

I Komang Somawirata, ST, MT
NIP.Y. 1030100361


SENSOR KIRI
SENSOR KANAN
LCD
ATMEGA16
DRIVER MOTOR KIRI
DRIVER MOTOR KANAN

LISTING PROGRAM

```
/* File include */
#include <mega16.h>
#include <delay.h>

/* Pendefinisian *
#define arah2          PORTD.7
#define arah1          PORTD.6
#define pwm2           OCR1A    //kanan
#define pwm1           OCR1B    //kiri
#define enable         PORTB.1
#define clock          PORTB.2
#define MatrikX1       PORTC.4
#define MatrikX2       PORTC.5
#define MatrikX3       PORTC.6
#define MatrikX4       PORTC.7
#define MatrikY1       PINC.0
#define MatrikY2       PINC.1
#define MatrikY3       PINC.2
#define MatrikY4       PINC.3

/* Inisialisasi variabel global */
unsigned char tanda,time,mode;
int a,bkanan,bkiri,kanan,kiri,bkanan1,bkiri1,vkanan,vkiri,tkiri,tkanan,mtka,mtki;
float x,kp,ti,td,b0,b1,b2,mka,mka1,eka,eka1,eka2,mki,mki1,eki,eki1,eki2;
long int ;
unsigned int i;
bit tka,tki,trx,tb;
konstanta_timing[25]={1,666,333,111,167,133,112,96,84,74,66,60,56,52,48,44,42,40
,38,36,34,32,30,28,28};

/* Inisialisasi LCD */
void dataout(unsigned char data_LCD, char kode)
{
 unsigned char i;
 clock=0;
 for(i=0;i<8;i++)
        {
                tx_LCD=(data_LCD & 0x1)==0x1 ? 1 : 0;
                delay_us(4);
                clock=1;
                delay_us(4);
                clock=0;
                data_LCD=data_LCD>>1;
        }
        rs=kode;
        delay_us(40);
        enable=1;
```

```
        delay_us(40);
        enable=0;
}
void pos(unsigned char i,unsigned char n)
{
        if(i==1)
        {
                dataout (0x80+n-1,0);
                delay_us(40);
        }
        else if (i==2)
        {
                dataout (0xc0+n-1,0);
                delay_us(40);
        }
        else;
}
void erick()
{
        dataout(0x01,0);
        delay_ms(2);
}
void initlcd()
{
        delay_ms(1000);
        dataout(0x30,0);
        delay_ms(500);
        dataout(0x30,0);
        delay_us(10000);
        dataout(0x30,0);
        delay_us(4000);
        dataout(0x38,0);
        delay_us(4000);
        dataout(0x08,0);
        delay_us(4000);
        dataout(0x01,0);
        delay_us(4000);
        dataout(0x0c,0);
        delay_us(4000);
        dataout(0x06,0);
        delay_us(4000);
}
void cetak(int i,int n,flash char *text)
{
        pos(i,n);
        while(*text)
        {
                dataout(*text++,1);
                delay_us(40);
```

```
        }
}

/*Inisialisasi keypad */
char Tombolnya()
{
        MatrikX1        = 0;
        MatrikX2        = 1;
        MatrikX3        = 1;
        MatrikX4        = 1;
        delay_ms(10);
        switch ( PINC & 0x0F )
        {
        case 0x0E:
                while(MatrikY1==0){;}
                return '1';
                break;
        case 0x0D:
                while(MatrikY2==0){;}
                return '4';
                break;
        case 0x0B:
                while(MatrikY3==0){;}
                return '7';
                break;
        case 0x07:
                while(MatrikY4==0){;}
                return '.';
                break;
        }
        MatrikX1        = 1;
        MatrikX2        = 0;
        MatrikX3        = 1;
        MatrikX4        = 1;
        delay_ms(10);
        switch ( PINC & 0x0F )
        {
        case 0x0E:
                while(MatrikY1==0){;}
                return '2';
                break;
        case 0x0D:
                while(MatrikY2==0){;}
                return '5';
                break;
        case 0x0B:
                while(MatrikY3==0){;}
                return '8';
                break;
```

```
case 0x07:
        while(MatrikY4==0){;}
        return '0';
        break;
}
MatrikX1    = 1;
MatrikX2    = 1;
MatrikX3    = 0;
MatrikX4    = 1;
delay_ms(10);
switch ( PINC & 0x0F )
{
case 0x0E:
        while(MatrikY1==0){;}
        return '3';
        break;
case 0x0D:
        while(MatrikY2==0){;}
        return '6';
        break;
case 0x0B:
        while(MatrikY3==0){;}
        return '9';
        break;
case 0x07:
        while(MatrikY4==0){;}
        return 'E';
        break;
}
MatrikX1    = 1;
MatrikX2    = 1;
MatrikX3    = 1;
MatrikX4    = 0;
delay_ms(10);
switch ( PINC & 0x0F )
{
case 0x0E:
        while(MatrikY1==0){;}
        return 'R';
        break;
case 0x0D:
        while(MatrikY2==0){;}
        return 'M';
        break;
case 0x0B:
        while(MatrikY3==0){;}
        return 'U';
        break;
case 0x07:
```

```
                while(MatrikY4==0){;}
                return 'D';
                break;
        default:
                return 'Z';
                break;
        }
}

/*Inisialisasi input output */
void init_port()
{
        DDRC=0b11110000;
        PORTC=0b11111111;
        DDRB=0b00011111;
        DDRD=0b11110010;
        DDRA=0b00000000;
        PORTD=0b111111001;
}
void init_ADC()
{
        /* Referensi AVcc*/
        SFIOR=SFIOR&0x0f;
        ADMUX=0x40;
        ADCSRA=0x02;
}

unsigned int konversi(unsigned char channel)
{
        unsigned char high,low;
        int data_konversi;

        /* pilih channel dengan referensi AVcc */
        ADMUX=(ADMUX&0xe0)|channel;

        /* Nyalakan ADC dan Start Konversi */
        ADCSRA=ADCSRA|0b11000000;

        /* tunggu proses konversi selesai */
        while((ADCSRA&0b00010000)!=0x10);

        /* matikan ADC, reset ADIF */
        ADCSRA=(ADCSRA|0b00010000)&0b01111111;

        /* data hasil konversi */
        delay_ms(1);
        low=ADCL;
        high=ADCH&0x03;
        data_konversi=(high%2)*256;
```

```
        high/=2;
        data_konversi=(data_konversi+((high%2)*512))+low;
        return data_konversi;
}
/*inisialisasi timer 1*/
void init_timer1()
{
        /* Timer ini akan digunakan triger CDI */
        /* frekuensi clock 2 MHz */
        TCCR1A=TCCR1A|0b00000000;
        TCCR1B=TCCR1B|0b00000101;
        TIMSK=TIMSK|0x04;
        TIFR=TIFR|0x04;
        TCNT1H=0;
        TCNT1L=0;
}
void init_timer2()
{
        /* Timer ini akan digunakan sebagai penghitung kecepatan */
        /* Mode yang digunakan adalah normal */
        TCCR2=0x06;
        TIMSK=TIMSK|0x40;
        TIFR=TIFR|0x40;
        TCNT2=0;
}
void init_timer0()
{
        /* Timer ini akan digunakan sebagai penghitung kecepatan */
        /* Mode yang digunakan adalah normal */
        TCCR0=0x04;
        TIMSK=TIMSK|0x01;
        TIFR=TIFR|0x01;
        TCNT0=0;
}
void init_ext_interrupt()
{
        /* Set untuk interupt 0 rising edge */
        MCUCR=7;
        GICR=GICR|0xC0;
}
void tampil(unsigned int n)
{
        unsigned int ascii;
        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);
}
/* Turn registers saving off */
```

```
#pragma savereg-
/* interrupt handler */
interrupt [9] void timer1_overflow(void)
{
        #asm
        push r30
        push r31
        in   r30,SREG
        push r30
        #endasm
        bkanan1=bkanan;
        bkanan=0;
        bkiri1=bkiri;
        bkiri=0;
        TIFR=TIFR|0x04;
        TCNT1H=0xEE;
        TCNT1L=0xBF;
        #asm
        pop r30
        out SREG,r30
        pop r31
        pop r30
        #endasm*
}
/* re-enable register saving for the other interrupts */
#pragma savereg+
/* Turn registers saving off */
#pragma savereg-
/* interrupt handler */
interrupt [5] void timer2_overflow(void)
{
        #asm
        push r30
        push r31
        in   r30,SREG
        push r30
        #endasm
   time++;
   if(time>=61)
   {
        bkanan1=bkanan;
        bkanan=0;
        bkiri1=bkiri;
        bkiri=0;
        time=0;
   }

TCNT2=0;
    TIFR=TIFR|0x40;
```

```
        #asm
        pop r30
        out SREG,r30
        pop r31
        pop r30
        #endasm*
}

interrupt [10] void timer0_overflow(void)
{
        #asm
        push r30
        push r31
        in   r30,SREG
        push r30
        #endasm
    time++;
    if(time>=61)
    {
        bkanan1=bkanan;
        bkanan=0;
        bkiri1=bkiri;
        bkiri=0;
        time=0;
     }
     TCNT0=0;
     TIFR=TIFR|0x01;
        #asm
        pop r30
        out SREG,r30
        pop r31
        pop r30
        #endasm*
}
/* re-enable register saving for the other interrupts */
#pragma savereg+

/* Turn registers saving off */
#pragma savereg-
/* interrupt handler */
interrupt [2] void external_int0(void)
{
        #asm
        push r30
        push r31
        in   r30,SREG
        push r30
        #endasm
        bkiri++;
```

```
        #asm
        pop r30
        out SREG,r30
        pop r31
        pop r30
        #endasm
}

interrupt [3] void external_int1(void)
{
        #asm
        push r30
        push r31
        in   r30,SREG
        push r30
        #endasm
        bkanan++;
        #asm
        pop r30
        out SREG,r30
        pop r31
        pop r30
        #endasm
}

char rkey()
{
unsigned char zero;
zero=Tombolnya();
while (zero=='Z') zero=Tombolnya();
return zero;

}
void motorkanan(int ot)
{
if(ot==0){pwm1=0;}
else if(ot>0){arah1=0;;pwm1=ot;}
else {arah1=1;pwm1=-ot;}
}
void motorkiri(int ot)
{
if(ot==0){pwm2=0;}
else if(ot>0){arah2=0;pwm2=ot;}
else {arah2=1;pwm2=-ot;}
}
void motor(int kirim,int kananm)
{
motorkiri(kirim);motorkanan(kananm);
}
```

```
float keyhasil()
{
float ke,ii;
char yh,asi;
bit tdot;
ke=0;tdot=0;ii=1;
while(1)
{
yh=rkey();
if(yh=='E')break;
else if(yh=='.') tdot=1;
else if(tdot==1) {asi=yh-48;ke=(ke*10+asi);ii=ii*10;}
else {asi=yh-48;ke=ke*10+asi;}
dataout(yh,1);
}
return ke/ii;
}
float abs(float ab)
{
float sd;
if(ab<0) sd=-ab;
else sd=ab;
return sd;
}

/* Program Utama */
void main()
{
        /* Inisialisasi */
        init_port();
        TCCR1A=TCCR1A|0xa3;
        TCCR1B=TCCR1B|0x0B;
        pwm1=0;
        pwm2=0;
        init_timer0();
        init_ext_interrupt();
        initlcd();
        init_ADC();
        mka=0;mka1=0;
        b0=48;b1=-9;b2=4;
        mka=0;mka1=0;eka=0;eka1=0;eka2=0;
        mki=0;mki1=0;eki=0;eki1=0;eki2=0;
        #asm("sei")
        pos(1,1);
        erick();
        cetak(1,1,"Masukkan mode:");
        a=rkey();
        if(a==50) mode=2;
        else if(a==49) mode=1;
```

```
else mode=0;
erick();
cetak(1,1,"Masukkan Kp:");
pos(2,1);
kp=keyhasil();
erick();
cetak(1,1,"Masukkan Ti:");
pos(2,1);
ti=keyhasil();
erick();
cetak(1,1,"Masukkan Td:");
pos(2,1);
td=keyhasil();
erick();
cetak(1,14,"rps");
cetak(2,14,"dec");
b0=kp+kp*t/2+kd/t;          //48
b1=-kp+ki*t/2-2kd/t;        //-9
b2=kd/t;                    //4
do
{
pos(1,7);tampil(bkanan1/20);
pos(1,1);tampil(bkiri1/20);
kanan=loopkonv(2,100)/4;
kiri=loopkonv(3,100)/4;
pos(2,7);tampil(kanan);
pos(2,1);tampil(kiri);

if(mode==1)                 //40
{
if(kiri<126) {tkiri=-0.32*kiri+40;tki=0;}
else if(kiri>130) {tkiri=0.32*kiri-41.6;tki=1;}
else tkiri=0;
if(kanan<126) {tkanan=-0.32*kanan+40;tka=0;}
else if(kanan>130) {tkanan=0.32*kanan-41.6;tka=1;}
else tkanan=0;
}

else if(mode==2)            //30
        {
        if(kiri<126) {tkiri=-0.24*kiri+30;tki=0;}
        else if(kiri>130) {tkiri=0.24*kiri-31.2;tki=1;}
        else tkiri=0;
        if(kanan<126) {tkanan=-0.24*kanan+30;tka=0;}
        else if(kanan>130) {tkanan=0.24*kanan-31.2;tka=1;}
        else tkanan=0;
        }
        else
        {
```

```
            if(kiri<126)     {vkiri=1023;tki=0;}
            else if(kiri>130) {vkiri=1023;tki=1;}
            else vkiri=0;
            if(kanan<126)     {vkanan=1023;tka=0;}
            else if(kanan>130) {vkanan=1023;tka=1;}
            else vkanan=0;

            if(kiri<126) {vkiri=-8*kiri+1000;tki=0;}
            else if(kiri>130) {vkiri=8*kiri-1040;tki=1;}
            else vkiri=0;
            if(kanan<126) {vkanan=-8*kanan+1000;tka=0;}
            else if(kanan>130) {vkanan=8*kanan-1040;tka=1;}
            else vkanan=0;
            }

        if(mode!=0)
        {
            if(tkiri==0 )vkiri=0;
            else if
            {
            eki=(tkiri-bkiri1/20)/tkiri;
            vkiri=vkiri+b0*eki+b1*eki1+b2*eki2;
            eki2=eki1;eki1=eki;
            }
            if(tkanan==0)vkanan=0;
            else if
            {
            eka=(tkanan-bkanan1/20)/tkanan;
            mka=mka1+b0*eka+b1*eka1+b2*eka2;
            eka2=eka1;eka1=eka;
            }
        }
        if(tki==1) mtki=-vkiri;
        else mtki=vkiri;
        if(tka==1) mtka=-vkanan;
        else mtka=vkanan;
        motor(mtki,mtka);
        }while(1);
}
```

The *Extended Concise* LCD Data Sheet

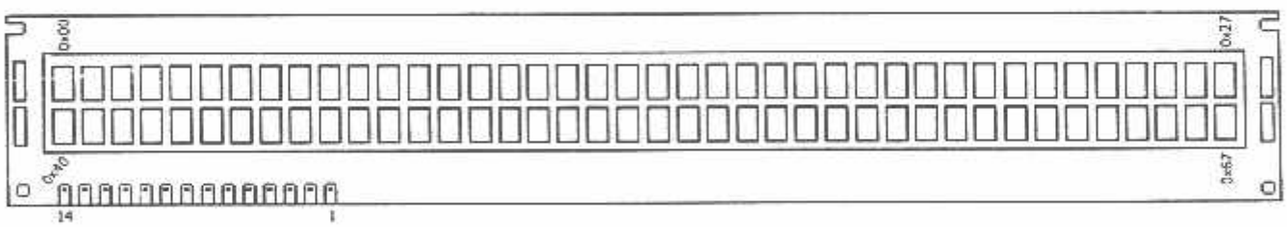
for HD44780

Version: 25.6.1999

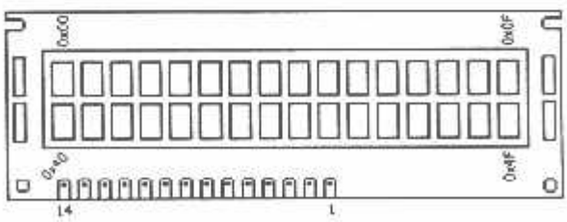
Instruction	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0	Description	Clock-Cycles
NOP	0	0	0	0	0	0	0	0	0	0	No Operation	0
Clear Display	0	0	0	0	0	0	0	0	0	1	Clear display & set address counter to zero	165
Cursor Home	0	0	0	0	0	0	0	0	1	x	Set adress counter to zero, return shifted display to original position. DD RAM contents remains unchanged.	3
Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S	Set cursor move direction (I/D) and specify automatic display shift (S).	3
Display Control	0	0	0	0	0	0	1	D	C	B	Turn display (D), cursor on/off (C), and cursor blinking (B).	3
Cursor / Display shift	0	0	0	0	0	1	S/C	R/L	x	x	Shift display or move cursor (S/C) and specify direction (R/L).	3
Function Set	0	0	0	0	1	DL	N	F	x	x	Set interface data width (DL), number of display lines (N) and character font (F).	3
Set CGRAM Address	0	0	0	1	CGRAM Address						Set CGRAM address. CGRAM data is sent afterwards.	3
Set DDRAM Address	0	0	1	DDRAM Address							Set DDRAM address. DDRAM data is sent afterwards.	3
Busy Flag & Address	0	1	BF	Address Counter							Read busy flag (BF) and address counter	0
Write Data	1	0	Data								Write data into DDRAM or CGRAM	3
Read Data	1	1	Data								Read data from DDRAM or CGRAM	3

x : Don't care						
	I/D	1 0	Increment Decrement	R/L	1 0	Shift to the right Shift to the left
	S	1 0	Automatic display shift	DL	1 0	8 bit interface 4 bit interface
	D	1 0	Display ON Display OFF	N	1 0	2 lines 1 line
	C	1 0	Cursor ON Cursor OFF	F	1 0	5x10 dots 5x7 dots
	B	1 0	Cursor blinking	DDRAM : Display Data RAM CGRAM : Character Generator RAM		
	S/C	1 0	Display shift Cursor move			

LCD Display with 2 lines x 40 characters :



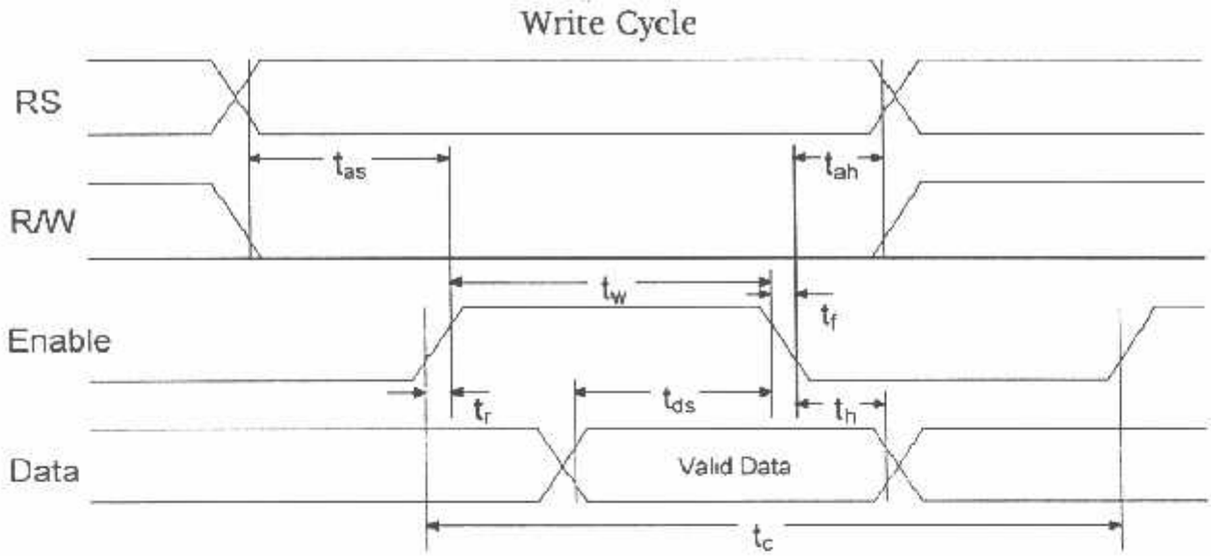
LCD Display with 2 lines x 16 characters :



Pin No	Name	Function	Description
1	Vss	Power	GND
2	Vdd	Power	+ 5 V
3	Vee	Contrast Adj.	(-2) 0 - 5 V
4	RS	Command	Register Select
5	R/W	Command	Read / Write
6	E	Command	Enable (Strobe)
7	D0	I/O	Data LSB
8	D1	I/O	Data
9	D2	I/O	Data
10	D3	I/O	Data
11	D4	I/O	Data
12	D5	I/O	Data
13	D6	I/O	Data
14	D7	I/O	Data MSB

Bus Timing Characteristics

(Ta = - 20 to + 75°C)



Write-Cycle	V_{DD}	2.7 - 4.5 V [2]	4.5 - 5.5 V [2]		2.7 - 4.5 V [2]	4.5 - 5.5 V [2]	
Parameter	**Symbol**	**Min[1]**		**Typ[1]**	**Max[1]**		**Unit**
Enable Cycle Time	t_c	1000	500	-	-	-	ns
Enable Pulse Width (High)	t_w	450	230	-	-	-	ns
Enable Rise/Fall Time	t_r, t_f	-	-	-	25	20	ns
Address Setup Time	t_{as}	60	40	-	-	-	ns
Address Hold Time	t_{ah}	20	10	-	-	-	ns
Data Setup Time	t_{ds}	195	80	-	-	-	ns
Data Hold Time	t_h	10	10	-	-	-	ns

(1) *The above specifications are indications only (based on Hitachi HD44780). Timing will vary from manufacturer to manufacturer.*

(2) *Power Supply :* *HD44780 S :* V_{DD} = 4.5 - 5.5 V
HD44780 U : V_{DD} = 2.7 - 5.5 V

This data sheet refers to specifications for the Hitachi HD44780 LCD Driver chip, which is used for most LCD modules.

Common types are :

1 line x 20 characters
2 lines x 16 characters
2 lines x 20 characters
2 lines x 40 characters
4 lines x 20 characters
4 lines x 40 characters

MOTOROLA

SERIAL-IN PARALLEL-OUT SHIFT REGISTER

SN54/74LS164

SERIAL-IN PARALLEL-OUT SHIFT REGISTER

LOW POWER SCHOTTKY

The SN54/74LS164 is a high speed 8-Bit Serial-In Parallel-Out Shift Register. Serial data is entered through a 2-Input AND gate synchronous with the LOW to HIGH transition of the clock. The device features an asynchronous Master Reset which clears the register setting all outputs LOW independent of the clock. It utilizes the Schottky diode clamped process to achieve high speeds and is fully compatible with all Motorola TTL products.

- Typical Shift Frequency of 35 MHz
- Asynchronous Master Reset
- Gated Serial Data Input
- Fully Synchronous Data Transfers
- Input Clamp Diodes Limit High Speed Termination Effects
- ESD > 3500 Volts

J SUFFIX
CERAMIC
CASE 632-08

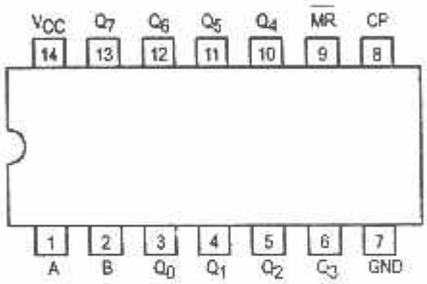
N SUFFIX
PLASTIC
CASE 646-06

D SUFFIX
SOIC
CASE 751A-02

ORDERING INFORMATION

SN54LSXXXJ	Ceramic
SN74LSXXXN	Plastic
SN74LSXXXD	SOIC

CONNECTION DIAGRAM DIP (TOP VIEW)



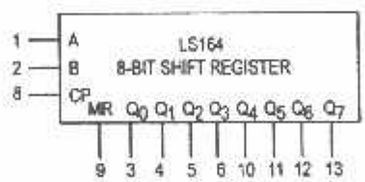
NOTE:
The Flatpak version has the same pinouts (Connection Diagram) as the Dual In-Line Package.

PIN NAMES

		LOADING (Note a) HIGH	LOW
A, B	Data Inputs	0.5 U.L.	0.25 U.L.
CP	Clock (Active HIGH Going Edge) Input	0.5 U.L.	0.25 U.L.
MR	Master Reset (Active LOW) Input	0.5 U.L.	0.25 U.L.
Q_0–Q_7	Outputs (Note b)	10 U.L.	5 (2.5) U.L.

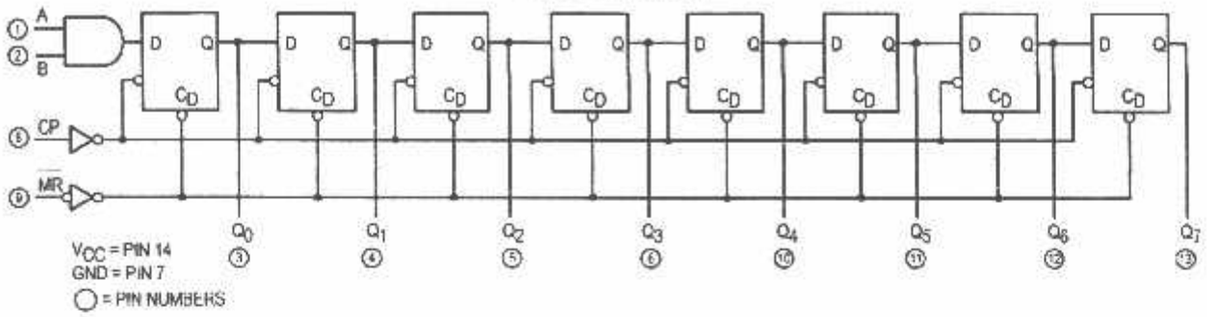
NOTES:
a) 1 TTL Unit Load (U.L.) = 40 µA HIGH/1.6 mA LOW.
b) The Output LOW drive factor is 2.5 U.L. for Military (54) and 5 U.L. for Commercial (74) Temperature Ranges.

LOGIC SYMBOL



V_{CC} = PIN 14
GND = PIN 7

LOGIC DIAGRAM



FUNCTIONAL DESCRIPTION

The LS164 is an edge-triggered 8-bit shift register with serial data entry and an output from each of the eight stages. Data is entered serially through one of two inputs (A or B); either of these inputs can be used as an active HIGH Enable for data entry through the other input. An unused input must be tied HIGH, or both inputs connected together.

Each LOW-to-HIGH transition on the Clock (CP) input shifts data one place to the right and enters into Q_0 the logical AND of the two data inputs (A•B) that existed before the rising clock edge. A LOW level on the Master Reset (MR) input overrides all other inputs and clears the register asynchronously, forcing all Q outputs LOW.

MODE SELECT — TRUTH TABLE

OPERATING MODE	INPUTS			OUTPUTS	
	MR	A	B	Q_0	Q_1–Q_7
Reset (Clear)	L	X	X	L	L – L
Shift	H	l	l	L	q_0 – q_6
	H	l	h	L	q_0 – q_6
	H	h	l	L	q_0 – q_6
	H	h	h	H	q_0 – q_6

L (l) = LOW Voltage Levels
H (h) = HIGH Voltage Levels
X = Don't Care
q_n = Lower case letters indicate the state of the referenced input or output one set-up time prior to the LOW to HIGH clock transition.

GUARANTEED OPERATING RANGES

Symbol	Parameter		Min	Typ	Max	Unit
V_{CC}	Supply Voltage	54 74	4.5 4.75	5.0 5.0	5.5 5.25	V
T_A	Operating Ambient Temperature Range	54 74	−55 0	25 25	125 70	°C
I_{OH}	Output Current — High	54, 74			−0.4	mA
I_{OL}	Output Current — Low	54 74			4.0 8.0	mA

DC CHARACTERISTICS OVER OPERATING TEMPERATURE RANGE (unless otherwise specified)

Symbol	Parameter		Limits Min	Typ	Max	Unit	Test Conditions	
V_{IH}	Input HIGH Voltage		2.0			V	Guaranteed Input HIGH Voltage for All Inputs	
V_{IL}	Input LOW Voltage	54			0.7	V	Guaranteed Input LOW Voltage for All Inputs	
		74			0.8			
V_{IK}	Input Clamp Diode Voltage			−0.65	−1.5	V	V_{CC} = MIN, I_{IN} = −18 mA	
V_{OH}	Output HIGH Voltage	54	2.5	3.5		V	V_{CC} = MIN, I_{OH} = MAX, V_{IN} = V_{IH} or V_{IL} per Truth Table	
		74	2.7	3.5				
V_{OL}	Output LOW Voltage	54, 74		0.25	0.4	V	I_{OL} = 4.0 mA	V_{CC} = V_{CC} MIN, V_{IN} = V_{IH} or V_{IL} per Truth Table
		74		0.35	0.5	V	I_{OL} = 8.0 mA	
I_{IH}	Input HIGH Current				20	µA	V_{CC} = MAX, V_{IN} = 2.7 V	
					0.1	mA	V_{CC} = MAX, V_{IN} = 7.0 V	
I_{IL}	Input LOW Current				−0.4	mA	V_{CC} = MAX, V_{IN} = 0.4 V	
I_{OS}	Short Circuit Current (Note 1)		−20		−100	mA	V_{CC} = MAX	
I_{CC}	Power Supply Current				27	mA	V_{CC} = MAX	

Note 1: Not more than one output should be shorted at a time, nor for more than 1 second.

AC CHARACTERISTICS (T_A = 25°C)

Symbol	Parameter	Limits Min	Typ	Max	Unit	Test Conditions
f_{MAX}	Maximum Clock Frequency	25	36		MHz	V_{CC} = 5.0 V C_L = 15 pF
t_{PHL}	Propagation Delay MR to Output Q		24	36	ns	
t_{PLH} t_{PHL}	Propagation Delay Clock to Output Q		17 21	27 32	ns	

AC SETUP REQUIREMENTS (T_A = 25°C)

Symbol	Parameter	Limits Min	Typ	Max	Unit	Test Conditions
t_W	CP, MR Pulse Width	20			ns	V_{CC} = 5.0 V
t_s	Data Setup Time	15			ns	
t_h	Data Hold Time	5.0			ns	
t_{rec}	MR to Clock Recovery Time	20			ns	

AC WAVEFORMS

*The shaded areas indicate when the input is permitted to change for predictable output performance.

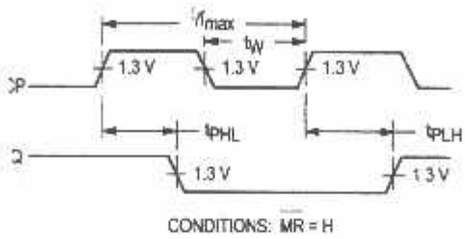


Figure 1. Clock to Output Delays and Clock Pulse Width

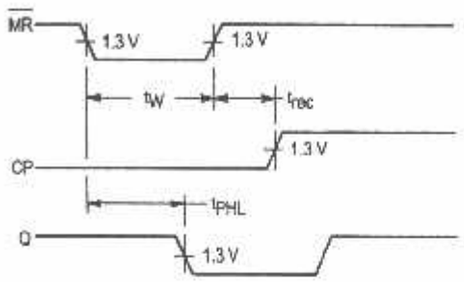


Figure 2. Master Reset Pulse Width, Master Reset to Output Delay and Master Reset to Clock Recovery Time

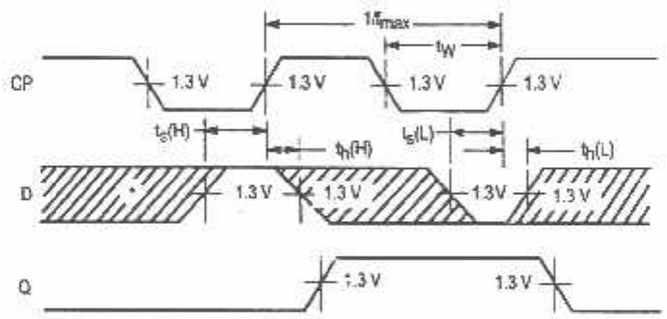


Figure 3. Data Setup and Hold Times

MC54/74HC14A

Hex Schmitt-Trigger Inverter
High–Performance Silicon–Gate CMOS

The MC54/74HC14A is identical in pinout to the LS14, LS04 and the C04. The device inputs are compatible with Standard CMOS outputs; th pullup resistors, they are compatible with LSTTL outputs.

The HC14A is useful to "square up" slow input rise and fall times. Due hysteresis voltage of the Schmitt trigger, the HC14A finds applications noisy environments.

- Output Drive Capability: 10 LSTTL Loads
- Outputs Directly Interface to CMOS, NMOS and TTL
- Operating Voltage Range: 2 to 6V
- Low Input Current: 1µA
- High Noise Immunity Characteristic of CMOS Devices
- In Compliance With the JEDEC Standard No. 7A Requirements
- Chip Complexity: 60 FETs or 15 Equivalent Gates

J SUFFIX
CERAMIC PACKAGE
CASE 632–08

N SUFFIX
PLASTIC PACKAGE
CASE 646–06

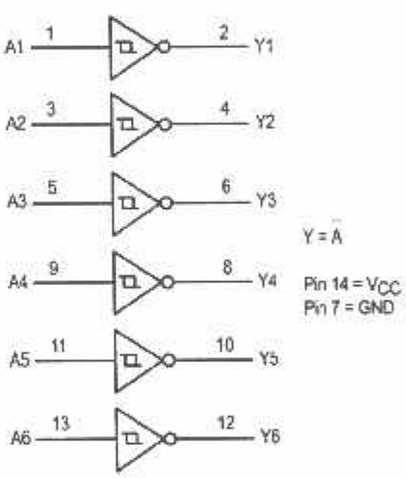
D SUFFIX
SOIC PACKAGE
CASE 751A–03

DT SUFFIX
TSSOP PACKAGE
CASE 948G–01

ORDERING INFORMATION

MC54HCXXAJ	Ceramic
MC74HCXXAN	Plastic
MC74HCXXAD	SOIC
MC74HCXXADT	TSSOP

LOGIC DIAGRAM



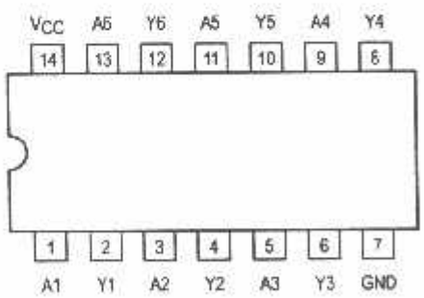
$Y = \overline{A}$

Pin 14 = V_{CC}
Pin 7 = GND

FUNCTION TABLE

Inputs	Outputs
A	Y
L	H
H	L

Pinout: 14–Lead Packages (Top View)



MOTOROLA

MAXIMUM RATINGS*

Symbol	Parameter	Value	Unit
V_{CC}	DC Supply Voltage (Referenced to GND)	– 0.5 to + 7.0	V
V_{in}	DC Input Voltage (Referenced to GND)	– 0.5 to V_{CC} + 0.5	V
V_{out}	DC Output Voltage (Referenced to GND)	– 0.5 to V_{CC} + 0.5	V
I_{in}	DC Input Current, per Pin	± 20	mA
I_{out}	DC Output Current, per Pin	± 25	mA
I_{CC}	DC Supply Current, V_{CC} and GND Pins	± 50	mA
P_D	Power Dissipation in Still Air, Plastic or Ceramic DIP†	750	mW
	SOIC Package†	500	
	TSSOP Package†	450	
T_{stg}	Storage Temperature Range	– 65 to + 150	°C
T_L	Lead Temperature, 1 mm from Case for 10 Seconds		°C
	Plastic DIP, SOIC or TSSOP Package	260	
	Ceramic DIP	300	

This device contains protection circuitry to guard against damage due to high static voltages or electric fields. However, precautions must be taken to avoid applications of any voltage higher than maximum rated voltages to this high–impedance circuit. For proper operation, V_{in} and V_{out} should be constrained to the range GND ≤ (V_{in} or V_{out}) ≤ V_{CC}.

Unused inputs must always be tied to an appropriate logic voltage level (e.g., either GND or V_{CC}). Unused outputs must be left open.

aximum Ratings are those values beyond which damage to the device may occur.
unctional operation should be restricted to the Recommended Operating Conditions.
erating — Plastic DIP: – 10 mW/°C from 65° to 125°C
Ceramic DIP: – 10 mW/°C from 100° to 125°C
SOIC Package: – 7 mW/°C from 65° to 125°C
TSSOP Package: – 6.1 mW/°C from 65° to 125°C
For high frequency or heavy load considerations, see Chapter 2 of the Motorola High–Speed CMOS Data Book (DL129/D).

RECOMMENDED OPERATING CONDITIONS

Symbol	Parameter		Min	Max	Unit
V_{CC}	DC Supply Voltage (Referenced to GND)		2.0	6.0	V
V_{in}, V_{out}	DC Input Voltage, Output Voltage (Referenced to GND)		0	V_{CC}	V
T_A	Operating Temperature Range, All Package Types		– 55	+ 125	°C
t_r, t_f	Input Rise/Fall Time (Figure 1)	V_{CC} = 2.0 V	0	No Limit*	ns
		V_{CC} = 4.5 V	0	No Limit*	
		V_{CC} = 6.0 V	0	No Limit*	

* When V_{in} = 50% V_{CC}, I_{CC} > 1mA

C CHARACTERISTICS (Voltages Referenced to GND)

Symbol	Parameter	Condition	V_{CC} V	Guaranteed Limit			Unit
				–55 to 25°C	≤85°C	≤125°C	
V_{T+} max	Maximum Positive–Going Input Threshold Voltage (Figure 3)	V_{out} = 0.1V $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	1.50 2.15 3.15 4.20	1.50 2.15 3.15 4.20	1.50 2.15 3.15 4.20	V
V_{T+} min	Minimum Positive–Going Input Threshold Voltage (Figure 3)	V_{out} = 0.1V $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	1.0 1.5 2.3 3.0	0.95 1.45 2.25 2.95	0.95 1.45 2.25 2.95	V
V_{T-} max	Maximum Negative–Going Input Threshold Voltage (Figure 3)	$V_{out} = V_{CC} - 0.1V$ $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	0.9 1.4 2.0 2.6	0.95 1.45 2.05 2.65	0.95 1.45 2.05 2.65	V
V_{T-} min	Minimum Negative–Going Input Threshold Voltage (Figure 3)	$V_{out} = V_{CC} - 0.1V$ $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	0.3 0.5 0.9 1.2	0.3 0.5 0.9 1.2	0.3 0.5 0.9 1.2	V
$V_H$max Note 2	Maximum Hysteresis Voltage (Figure 3)	V_{out} = 0.1V or V_{CC} – 0.1V $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	1.20 1.65 2.25 3.00	1.20 1.65 2.25 3.00	1.20 1.65 2.25 3.00	V
$V_H$min Note 2	Minimum Hysteresis Voltage (Figure 3)	V_{out} = 0.1V or V_{CC} – 0.1V $\|I_{out}\| \le 20\mu A$	2.0 3.0 4.5 6.0	0.20 0.25 0.40 0.50	0.20 0.25 0.40 0.50	0.20 0.25 0.40 0.50	V
V_{OH}	Minimum High–Level Output Voltage	$V_{in} \le V_{T-}$ min $\|I_{out}\| \le 20\mu A$	2.0 4.5 6.0	1.9 4.4 5.9	1.9 4.4 5.9	1.9 4.4 5.9	V
		$V_{in} \le V_{T-}$ min $\|I_{out}\| \le 2.4mA$ $\|I_{out}\| \le 4.0mA$ $\|I_{out}\| \le 5.2mA$	3.0 4.5 6.0	2.48 3.98 5.48	2.34 3.84 5.34	2.20 3.70 5.20	
V_{OL}	Maximum Low–Level Output Voltage	$V_{in} \ge V_{T+}$ max $\|I_{out}\| \le 20\mu A$	2.0 4.5 6.0	0.1 0.1 0.1	0.1 0.1 0.1	0.1 0.1 0.1	V
		$V_{in} \ge V_{T+}$ max $\|I_{out}\| \le 2.4mA$ $\|I_{out}\| \le 4.0mA$ $\|I_{out}\| \le 5.2mA$	3.0 4.5 6.0	0.26 0.26 0.26	0.33 0.33 0.33	0.40 0.40 0.40	
I_{in}	Maximum Input Leakage Current	$V_{in} = V_{CC}$ or GND	6.0	±0.1	±1.0	±1.0	µA
I_{CC}	Maximum Quiescent Supply Current (per Package)	$V_{in} = V_{CC}$ or GND I_{out} = 0µA	6.0	1.0	10	40	µA

1. Information on typical parametric values along with frequency or heavy load considerations can be found in Chapter 2 of the Motorola High–Speed CMOS Data Book (DL129/D).
2. $V_H$min > (V_{T+} min) – (V_{T-} max); $V_H$max = (V_{T+} max) – (V_{T-} min).

C CHARACTERISTICS (C_L = 50pF, Input $t_r = t_f$ = 6ns)

Symbol	Parameter	V_{CC} V	Guaranteed Limit			Unit
			–55 to 25°C	≤85°C	≤125°C	
t_{PLH}, t_{PHL}	Maximum Propagation Delay, Input A or B to Output Y (Figures 1 and 2)	2.0	75	95	110	ns
		3.0	30	40	55	
		4.5	15	19	22	
		6.0	13	16	19	
t_{TLH}, t_{THL}	Maximum Output Transition Time, Any Output (Figures 1 and 2)	2.0	75	95	110	ns
		3.0	27	32	36	
		4.5	15	19	22	
		6.0	13	16	19	
C_{in}	Maximum Input Capacitance		10	10	10	pF

TE: For propagation delays with loads other than 50 pF, and information on typical parametric values, see Chapter 2 of the Motorola High–Speed CMOS Data Book (DL129/D)

		Typical @ 25°C, V_{CC} = 5.0 V	
C_{PD}	Power Dissipation Capacitance (Per Inverter)*	22	pF

ed to determine the no–load dynamic power consumption: $P_D = C_{PD} V_{CC}^2 f + I_{CC} V_{CC}$. For load considerations, see Chapter 2 of the otorola High–Speed CMOS Data Book (DL129/D)

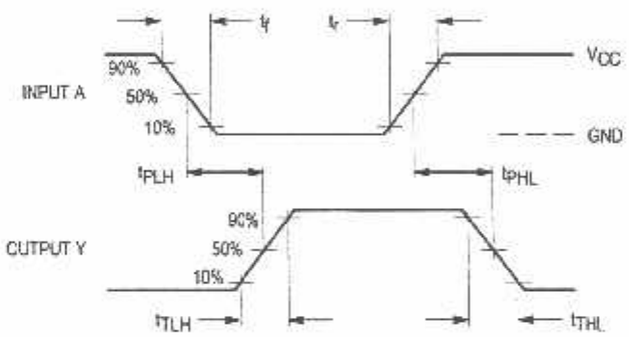
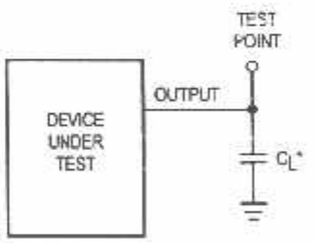
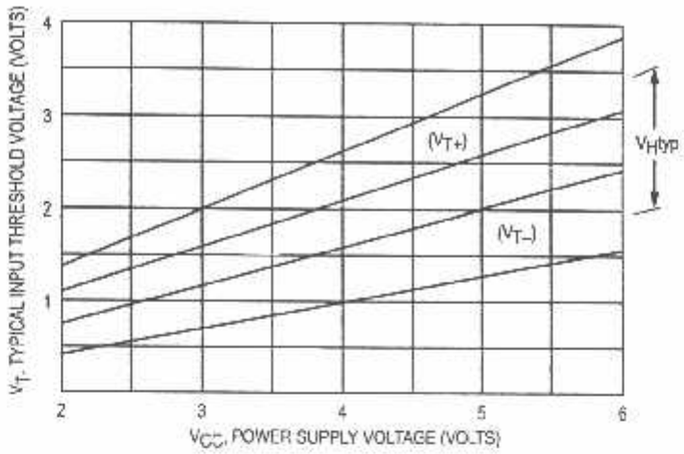


Figure 1. Switching Waveforms



*Includes all probe and jig capacitance

Figure 2. Test Circuit



$V_{Htyp} = (V_{T+}\ typ) - (V_{T-}\ typ)$

Figure 3. Typical Input Threshold, V_{T+}, V_{T-} versus Power Supply Voltage

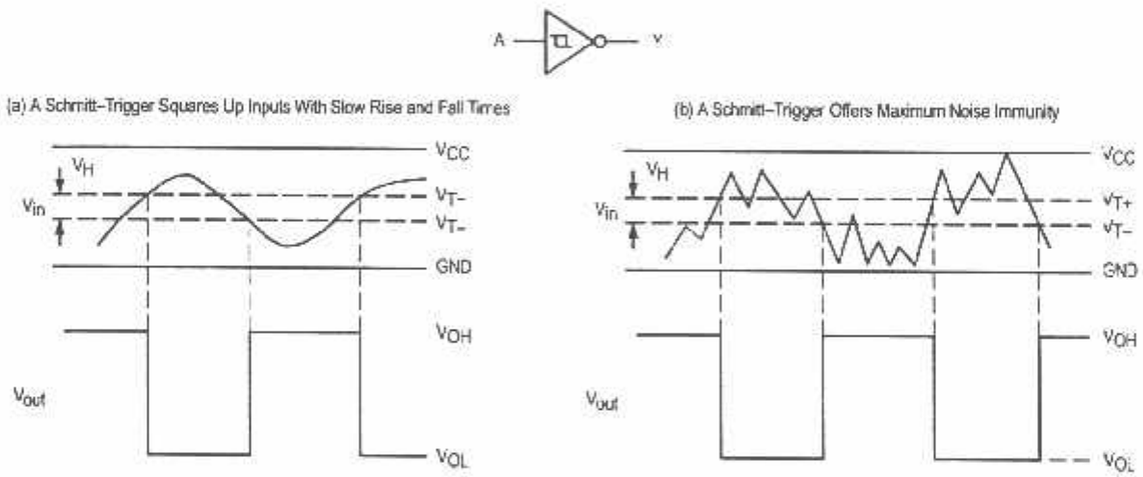
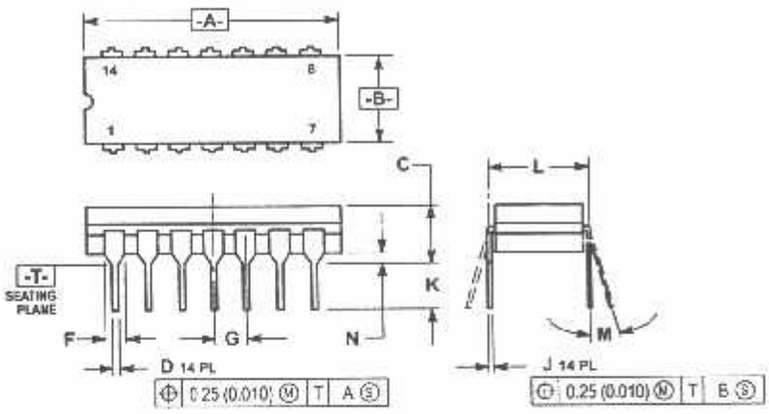


Figure 4. Typical Schmitt–Trigger Applications

OUTLINE DIMENSIONS

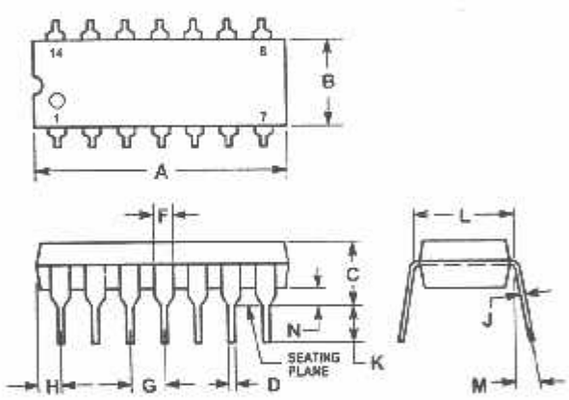
J SUFFIX
CERAMIC DIP PACKAGE
CASE 632–08
ISSUE Y



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: INCH.
3. DIMENSION L TO CENTER OF LEAD WHEN FORMED PARALLEL.
4. DIMENSION F MAY NARROW TO 0.76 (0.030) WHERE THE LEAD ENTERS THE CERAMIC BODY.

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	0.750	0.785	19.05	19.94
B	0.245	0.280	6.23	7.11
C	0.155	0.200	3.94	5.08
D	0.015	0.020	0.39	0.50
F	0.055	0.065	1.40	1.65
G	0.100 BSC		2.54 BSC	
J	0.008	0.015	0.21	0.38
K	0.125	0.170	3.18	4.31
L	0.300 BSC		7.62 BSC	
M	0°	15°	0°	15°
N	0.020	0.040	0.51	1.01

N SUFFIX
PLASTIC DIP PACKAGE
CASE 646–06
ISSUE L

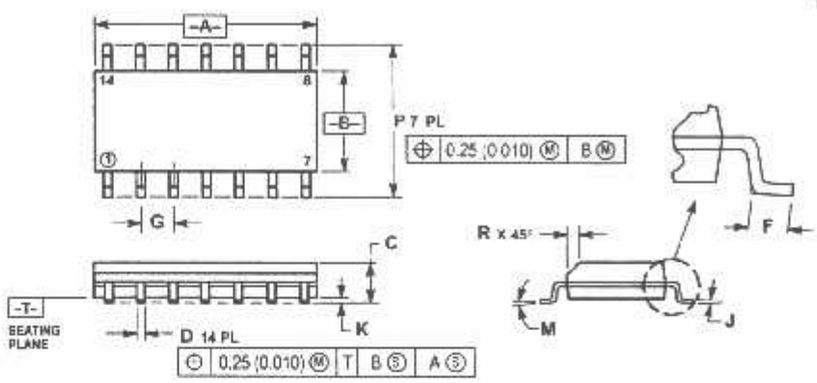


NOTES:
1. LEADS WITHIN 0.13 (0.005) RADIUS OF TRUE POSITION AT SEATING PLANE AT MAXIMUM MATERIAL CONDITION.
2. DIMENSION L TO CENTER OF LEADS WHEN FORMED PARALLEL.
3. DIMENSION B DOES NOT INCLUDE MOLD FLASH.
4. ROUNDED CORNERS OPTIONAL.

	INCHES		MILLIMETERS	
DIM	MIN	MAX	MIN	MAX
A	0.715	0.770	18.16	19.56
B	0.240	0.260	6.10	6.60
C	0.145	0.185	3.69	4.69
D	0.015	0.021	0.38	0.53
F	0.040	0.070	1.02	1.78
G	0.100 BSC		2.54 BSC	
H	0.052	0.095	1.32	2.41
J	0.008	0.015	0.20	0.38
K	0.115	0.135	2.92	3.43
L	0.300 BSC		7.62 BSC	
M	0°	10°	0°	10°
N	0.015	0.039	0.39	1.01

OUTLINE DIMENSIONS

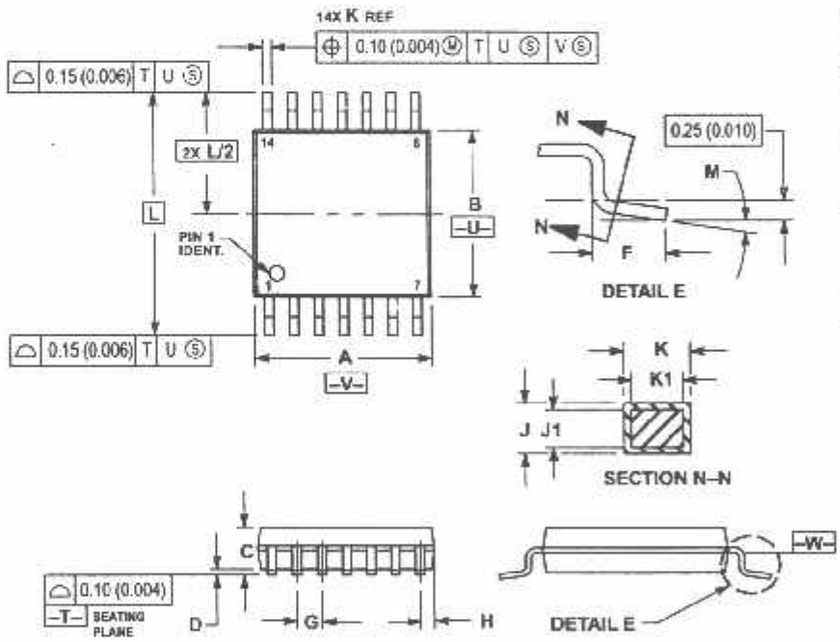
D SUFFIX
PLASTIC SOIC PACKAGE
CASE 751A–03
ISSUE F



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSIONS A AND B DO NOT INCLUDE MOLD PROTRUSION.
4. MAXIMUM MOLD PROTRUSION 0.15 (0.006) PER SIDE.
5. DIMENSION B DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.127 (0.005) TOTAL IN EXCESS OF THE B DIMENSION AT MAXIMUM MATERIAL CONDITION.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	8.55	8.75	0.337	0.344
B	3.80	4.00	0.150	0.157
C	1.35	1.75	0.054	0.068
D	0.35	0.49	0.014	0.019
F	0.40	1.25	0.016	0.049
G	1.27 BSC		0.050 BSC	
J	0.19	0.25	0.008	0.009
K	0.10	0.25	0.004	0.009
M	0°	7°	0°	7°
P	5.80	6.20	0.228	0.244
R	0.25	0.50	0.010	0.019

DT SUFFIX
PLASTIC TSSOP PACKAGE
CASE 948G–01
ISSUE O



NOTES:
1. DIMENSIONING AND TOLERANCING PER ANSI Y14.5M, 1982.
2. CONTROLLING DIMENSION: MILLIMETER.
3. DIMENSION A DOES NOT INCLUDE MOLD FLASH, PROTRUSIONS OR GATE BURRS. MOLD FLASH OR GATE BURRS SHALL NOT EXCEED 0.15 (0.006) PER SIDE.
4. DIMENSION B DOES NOT INCLUDE INTERLEAD FLASH OR PROTRUSION. INTERLEAD FLASH OR PROTRUSION SHALL NOT EXCEED 0.25 (0.010) PER SIDE.
5. DIMENSION K DOES NOT INCLUDE DAMBAR PROTRUSION. ALLOWABLE DAMBAR PROTRUSION SHALL BE 0.08 (0.003) TOTAL IN EXCESS OF THE K DIMENSION AT MAXIMUM MATERIAL CONDITION.
6. TERMINAL NUMBERS ARE SHOWN FOR REFERENCE ONLY.
7. DIMENSION A AND B ARE TO BE DETERMINED AT DATUM PLANE –W–.

	MILLIMETERS		INCHES	
DIM	MIN	MAX	MIN	MAX
A	4.90	5.10	0.193	0.200
B	4.30	4.50	0.169	0.177
C	—	1.20	—	0.047
D	0.05	0.15	0.002	0.006
F	0.50	0.75	0.020	0.030
G	0.65 BSC		0.026 BSC	
H	0.50	0.60	0.020	0.024
J	0.09	0.20	0.004	0.008
J1	0.09	0.16	0.004	0.006
K	0.19	0.30	0.007	0.012
K1	0.19	0.25	0.007	0.010
L	6.40 BSC		0.252 BSC	
M	0°	8°	0°	8°

tures

h-performance, Low-power AVR® 8-bit Microcontroller
/anced RISC Architecture
131 Powerful Instructions – Most Single-clock Cycle Execution
32 x 8 General Purpose Working Registers
Fully Static Operation
Up to 16 MIPS Throughput at 16 MHz
On-chip 2-cycle Multiplier
volatile Program and Data Memories
16K Bytes of In-System Self-Programmable Flash
Endurance: 10,000 Write/Erase Cycles
Optional Boot Code Section with Independent Lock Bits
In-System Programming by On-chip Boot Program
True Read-While-Write Operation
512 Bytes EEPROM
Endurance: 100,000 Write/Erase Cycles
1K Byte Internal SRAM
Programming Lock for Software Security
i (IEEE std. 1149.1 Compliant) Interface
Boundary-scan Capabilities According to the JTAG Standard
Extensive On-chip Debug Support
Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
heral Features
Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
eal Time Counter with Separate Oscillator
ur PWM Channels
– 8-channel, 10-bit ADC
8 Single-ended Channels
7 Differential Channels in TQFP Package Only
2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
– Byte-oriented Two-wire Serial Interface
– Programmable Serial USART
– Master/Slave SPI Serial Interface
– Programmable Watchdog Timer with Separate On-chip Oscillator
– On-chip Analog Comparator
Special Microcontroller Features
– Power-on Reset and Programmable Brown-out Detection
– Internal Calibrated RC Oscillator
– External and Internal Interrupt Sources
– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
I/O and Packages
– 32 Programmable I/O Lines
– 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
Operating Voltages
– 2.7 - 5.5V for ATmega16L
– 4.5 - 5.5V for ATmega16
Speed Grades
– 0 - 8 MHz for ATmega16L
– 0 - 16 MHz for ATmega16
Power Consumption @ 1 MHz, 3V, and 25°C for ATmega16L
– Active: 1.1 mA
– Idle Mode: 0.35 mA
– Power-down Mode: < 1 µA

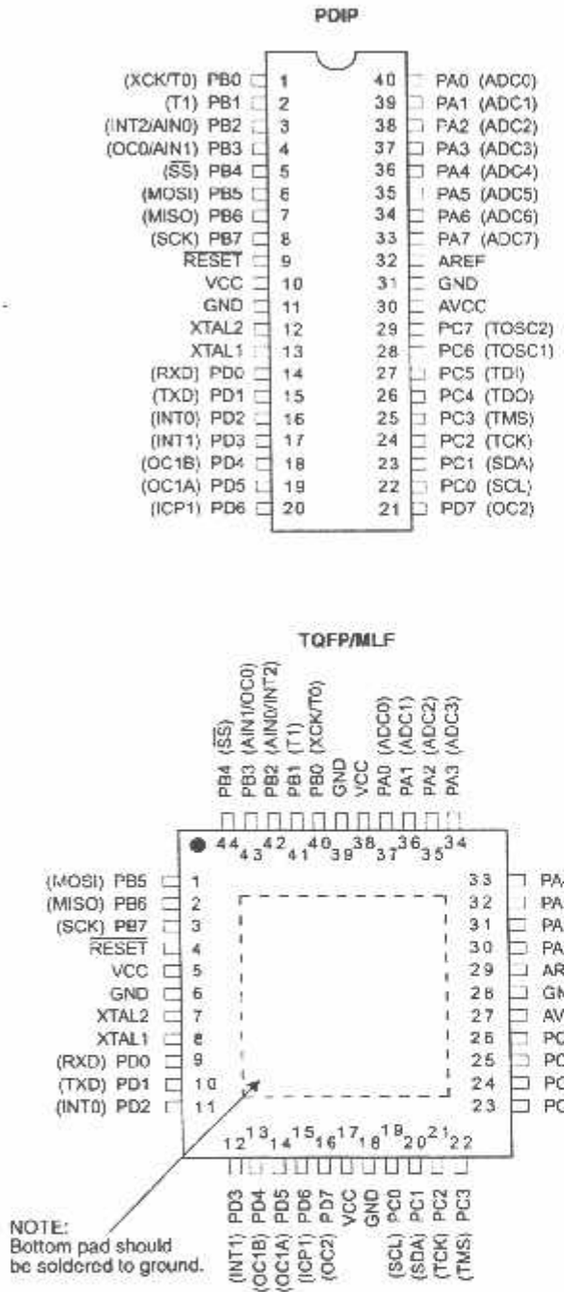
8-bit AVR® Microcontroller with 16K Bytes In-System Programmable Flash

ATmega16
ATmega16L

2466L–AVR–06/05

1 Configurations

Figure 1. Pinout ATmega16



Disclaimer

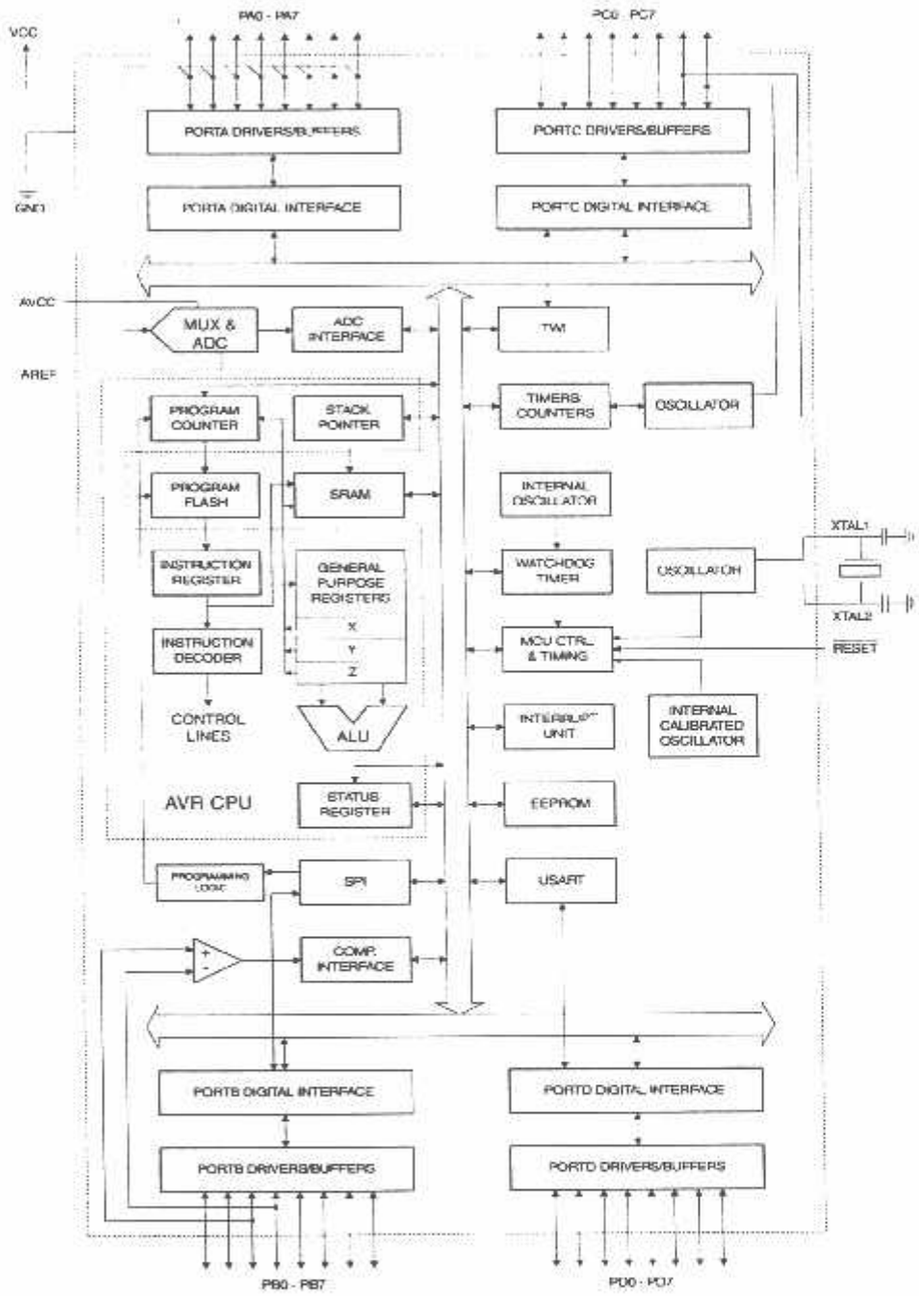
Typical values contained in this datasheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

'erview

The ATmega16 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing powerful instructions in a single clock cycle, the ATmega16 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

ck Diagram

Figure 2. 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 ATmega16 provides the following features: 16K bytes of In-System Programmable Flash Program memory with Read-While-Write capabilities, 512 bytes EEPROM, 1K byte SRAM, 32 general purpose I/O lines, 32 general purpose working registers, a JTAG interface for Boundary-scan, On-chip Debugging support and programming, three flexible Timer/Counters with compare modes, Internal and External Interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain (TQFP package only), a programmable Watchdog Timer with Internal Oscillator, an SPI serial port, and six software selectable power saving modes. The Idle mode stops the CPU while allowing the USART, Two-wire interface, A/D Converter, 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 External 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. In Extended Standby mode, both the main Oscillator and the Asynchronous Timer continue to run.

The device is manufactured using Atmel's high density nonvolatile memory technology. The On-chip ISP Flash allows the program memory to 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 ATmega16 is a powerful microcontroller that provides a highly-flexible and cost-effective solution to many embedded control applications.

The ATmega16 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.

Pin Descriptions

VCC Digital supply voltage.

GND Ground.

Port A (PA7..PA0) Port A serves as the analog inputs to the A/D Converter.

Port A also serves as an 8-bit bi-directional I/O port, if the A/D Converter is not used. Port pins can provide internal pull-up resistors (selected for each bit). The Port A output buffers have symmetrical drive characteristics with both high sink and source capability. When pins PA0 to PA7 are used as inputs and are externally pulled low, they will source current if the internal pull-up resistors are activated. The Port A pins are tri-stated when a reset condition becomes active, even if the clock is not running.

t B (PB7..PB0)	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. Port B also serves the functions of various special features of the ATmega16 as listed on page 56.
C (PC7..PC0)	Port C is an 8-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. If the JTAG interface is enabled, the pull-up resistors on pins PC5(TDI), PC3(TMS) and PC2(TCK) will be activated even if a reset occurs. Port C also serves the functions of the JTAG interface and other special features of the ATmega16 as listed on page 59.
D (PD7..PD0)	Port D is an 8-bit bi-directional 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 ATmega16 as listed on page 61.
ESET	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 36. Shorter pulses are not guaranteed to generate a reset.
TAL1	Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
TAL2	Output from the inverting Oscillator amplifier.
VCC	AVCC is the supply voltage pin for Port A and the A/D Converter. 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.
REF	AREF is the analog reference pin for the A/D Converter.

esources

A comprehensive set of development tools, application notes and datasheets are available for download on http://www.atmel.com/avr.

bout Code xamples

This documentation 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.

terrupts

This section describes the specifics of the interrupt handling as performed in ATmega16. For a general explanation of the AVR interrupt handling, refer to "Reset and Interrupt Handling" on page 11.

errupt Vectors in mega16

Table 18. Reset and Interrupt Vectors

Vector No.	Program Address(2)	Source	Interrupt Definition
1	$000(1)	RESET	External Pin, Power-on Reset, Brown-out Reset, Watchdog Reset, and JTAG AVR Reset
2	$002	INT0	External Interrupt Request 0
3	$004	INT1	External Interrupt Request 1
4	$006	TIMER2 COMP	Timer/Counter2 Compare Match
5	$008	TIMER2 OVF	Timer/Counter2 Overflow
6	$00A	TIMER1 CAPT	Timer/Counter1 Capture Event
7	$00C	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	$00E	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	$010	TIMER1 OVF	Timer/Counter1 Overflow
10	$012	TIMER0 OVF	Timer/Counter0 Overflow
11	$014	SPI, STC	Serial Transfer Complete
12	$016	USART, RXC	USART, Rx Complete
13	$018	USART, UDRE	USART Data Register Empty
14	$01A	USART, TXC	USART, Tx Complete
15	$01C	ADC	ADC Conversion Complete
16	$01E	EE_RDY	EEPROM Ready
17	$020	ANA_COMP	Analog Comparator
18	$022	TWI	Two-wire Serial Interface
19	$024	INT2	External Interrupt Request 2
20	$026	TIMER0 COMP	Timer/Counter0 Compare Match
21	$028	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 246.

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.

it Timer/Counter0 ı PWM

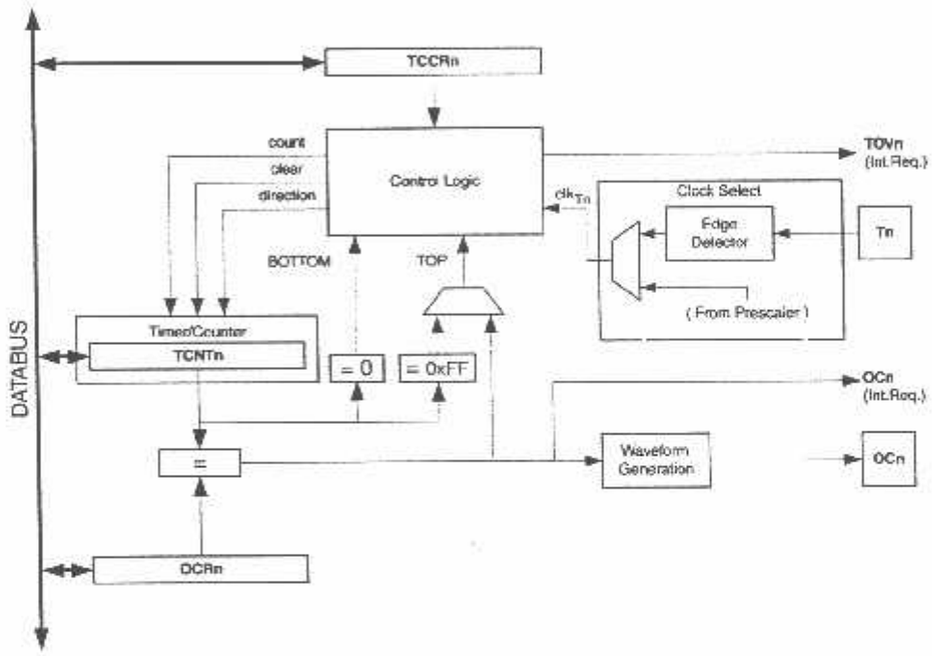
Timer/Counter0 is a general purpose, single compare unit, 8-bit Timer/Counter module. The main features are:

- **Single Compare Unit Counter**
- **Clear Timer on Compare Match (Auto Reload)**
- **Glitch-free, Phase Correct Pulse Width Modulator (PWM)**
- **Frequency Generator**
- **External Event Counter**
- **10-bit Clock Prescaler**
- **Overflow and Compare Match Interrupt Sources (TOV0 and OCF0)**

˜view

A simplified block diagram of the 8-bit Timer/Counter is shown in Figure 27. For the actual placement of I/O pins, refer to "Pinout ATmega16" on page 2. CPU accessible I/O Registers, including I/O bits and I/O pins, are shown in bold. The device-specific I/O Register and bit locations are listed in the "8-bit Timer/Counter Register Description" on page 81.

Figure 27. 8-bit Timer/Counter Block Diagram



gisters

The Timer/Counter (TCNT0) and Output Compare Register (OCR0) are 8-bit registers. Interrupt request (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 T0 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_{T0}).

The double buffered Output Compare Register (OCR0) is compared with the Timer/Counter value at all times. 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 (OC0). See "Output Compare Unit" on page 71. for details. The compare match event will also set the Compare Flag (OCF0) which can be used to generate an output compare interrupt request.

itions

Many register and bit references in this document are written in general form. A lower case "n" replaces the Timer/Counter number, in this case 0. However, when using the register or bit defines in a program, the precise form must be used i.e., TCNT0 for accessing Timer/Counter0 counter value and so on.

The definitions in Table 37 are also used extensively throughout the document.

Table 37. Definitions

BOTTOM	The counter reaches the BOTTOM when it becomes 0x00.
MAX	The counter reaches its MAXimum when it becomes 0xFF (decimal 255).
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 the fixed value 0xFF (MAX) or the value stored in the OCR0 Register. The assignment is dependent on the mode of operation.

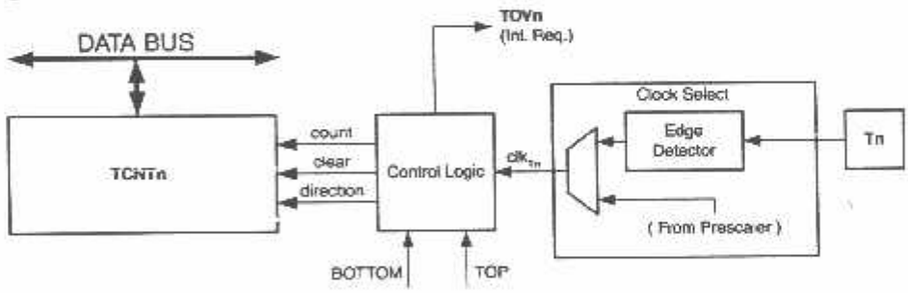
r/Counter Clock ces

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 (CS02:0) bits located in the Timer/Counter Control Register (TCCR0). For details on clock sources and prescaler, see "Timer/Counter0 and Timer/Counter1 Prescalers" on page 85.

unter Unit

The main part of the 8-bit Timer/Counter is the programmable bi-directional counter unit. Figure 28 shows a block diagram of the counter and its surroundings.

Figure 28. Counter Unit Block Diagram



Signal description (internal signals):

count	Increment or decrement TCNT0 by 1.
direction	Select between increment and decrement.
clear	Clear TCNT0 (set all bits to zero).
clk_{Tn}	Timer/Counter clock, referred to as clk_{T0} in the following.
TOP	Signalize that TCNT0 has reached maximum value.

BOTTOM Signalize that TCNT0 has reached minimum value (zero).

Depending of the mode of operation used, the counter is cleared, incremented, or decremented at each timer clock (clk_{T0}). clk_{T0} can be generated from an external or internal clock source, selected by the Clock Select bits (CS02:0). When no clock source is selected (CS02:0 = 0) the timer is stopped. However, the TCNT0 value can be accessed by the CPU, regardless of whether clk_{T0} 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 WGM01 and WGM00 bits located in the Timer/Counter Control Register (TCCR0). There are close connections between how the counter behaves (counts) and how waveforms are generated on the Output Compare output OC0. For more details about advanced counting sequences and waveform generation, see "Modes of Operation" on page 74.

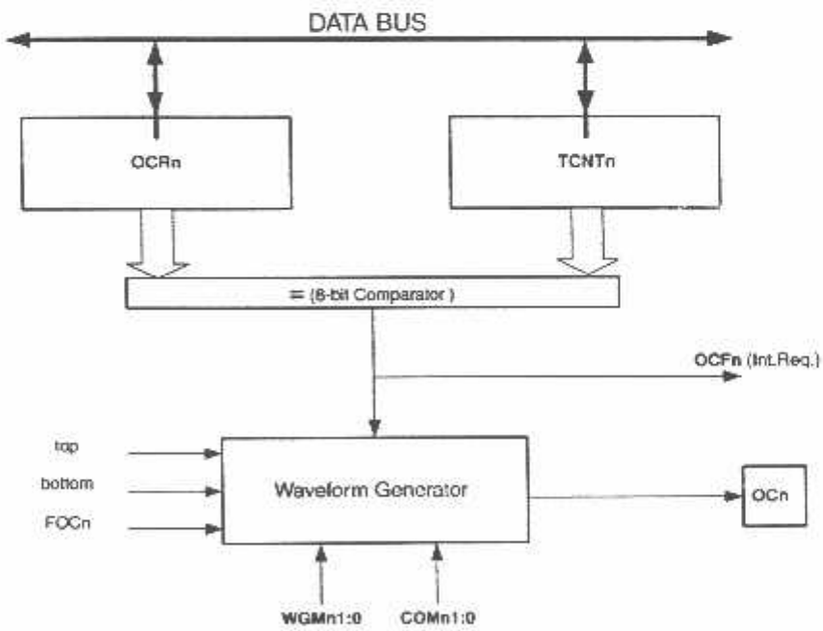
The Timer/Counter Overflow (TOV0) Flag is set according to the mode of operation selected by the WGM01:0 bits. TOV0 can be used for generating a CPU interrupt.

ut Compare Unit

The 8-bit comparator continuously compares TCNT0 with the Output Compare Register (OCR0). Whenever TCNT0 equals OCR0, the comparator signals a match. A match will set the Output Compare Flag (OCF0) at the next timer clock cycle. If enabled (OCIE0 = 1 and Global Interrupt Flag in SREG is set), the Output Compare Flag generates an output compare interrupt. The OCF0 Flag is automatically cleared when the interrupt is executed. Alternatively, the OCF0 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 WGM01:0 bits and Compare Output mode (COM01:0) bits. The max 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 74.).

Figure 29 shows a block diagram of the output compare unit.

Figure 29. Output Compare Unit, Block Diagram



The OCR0 Register is double buffered when using any of the 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 OCR0 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 OCR0 Register access may seem complex, but this is not case. When the double buffering is enabled, the CPU has access to the OCR0 Buffer Register, and if double buffering is disabled the CPU will access the OCR0 directly.

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 (FOC0) bit. Forcing compare match will not set the OCF0 Flag or reload/clear the timer, but the OC0 pin will be updated as if a real compare match had occurred (the COM01:0 bits settings define whether the OC0 pin is set, cleared or toggled).

ıre Match Blocking by ı Write

All CPU write operations to the TCNT0 Register will block any compare match that occur in the next timer clock cycle, even when the timer is stopped. This feature allows OCR0 to be initialized to the same value as TCNT0 without triggering an interrupt when the Timer/Counter clock is enabled.

ı the Output Compare

Since writing TCNT0 in any mode of operation will block all compare matches for one timer clock cycle, there are risks involved when changing TCNT0 when using the output compare unit, independently of whether the Timer/Counter is running or not. If the value written to TCNT0 equals the OCR0 value, the compare match will be missed, resulting in incorrect waveform generation. Similarly, do not write the TCNT0 value equal to BOTTOM when the counter is downcounting.

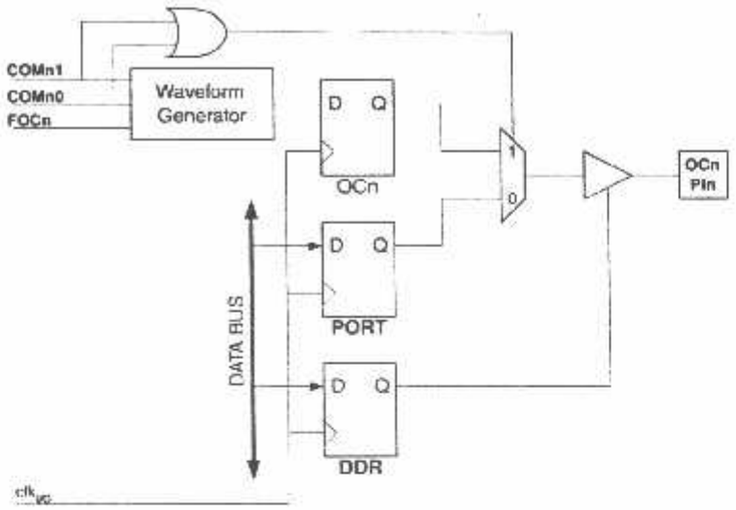
The setup of the OC0 should be performed before setting the Data Direction Register for the port pin to output. The easiest way of setting the OC0 value is to use the Force Output Compare (FOC0) strobe bits in Normal mode. The OC0 Register keeps its value even when changing between waveform generation modes.

Be aware that the COM01:0 bits are not double buffered together with the compare value. Changing the COM01:0 bits will take effect immediately.

mpare Match Output it

The Compare Output mode (COM01:0) bits have two functions. The Waveform Generator uses the COM01:0 bits for defining the Output Compare (OC0) state at the next compare match. Also, the COM01:0 bits control the OC0 pin output source. Figure 30 shows a simplified schematic of the logic affected by the COM01:0 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 COM01:0 bits are shown. When referring to the OC0 state, the reference is for the internal OC0 Register, not the OC0 pin. If a System Reset occur, the OC0 Register is reset to "0".

Figure 30. Compare Match Output Unit, Schematic



The general I/O port function is overridden by the Output Compare (OC0) from the Waveform Generator if either of the COM01:0 bits are set. However, the OC0 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 OC0 pin (DDR_OC0) must be set as output before the OC0 value is visible on the pin. The port override function is independent of the Waveform Generation mode.

The design of the output compare pin logic allows initialization of the OC0 state before the output is enabled. Note that some COM01:0 bit settings are reserved for certain modes of operation. See "8-bit Timer/Counter Register Description" on page 81.

mpare Output Mode and veform Generation

The Waveform Generator uses the COM01:0 bits differently in normal, CTC, and PWM modes. For all modes, setting the COM01:0 = 0 tells the waveform generator that no action on the OC0 Register is to be performed on the next compare match. For compare output actions in the non-PWM modes refer to Table 39 on page 82. For fast PWM mode, refer to Table 40 on page 82, and for phase correct PWM refer to Table 41 on page 82.

A change of the COM01:0 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 FOC0 strobe bits.

es 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 (WGM01:0) and Compare Output mode (COM01:0) bits. The Compare Output mode bits do not affect the counting sequence, while the Waveform Generation mode bits do. The COM01:0 bits control whether the PWM output generated should be inverted or not (inverted or non-inverted PWM). For non-PWM modes the COM01:0 bits control whether the output should be set, cleared, or toggled at a compare match (See "Compare Match Output Unit" on page 72.).

For detailed timing information refer to Figure 34, Figure 35, Figure 36 and Figure 37 in "Timer/Counter Timing Diagrams" on page 79.

al Mode

The simplest mode of operation is the normal mode (WGM01:0 = 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 8-bit value (TOP = 0xFF) and then restarts from the bottom (0x00). In normal operation the Timer/Counter Overflow Flag (TOV0) will be set in the same timer clock cycle as the TCNT0 becomes zero. The TOV0 Flag in this case behaves like a ninth bit, except that it is only set, not cleared. However, combined with the timer overflow interrupt that automatically clears the TOV0 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.

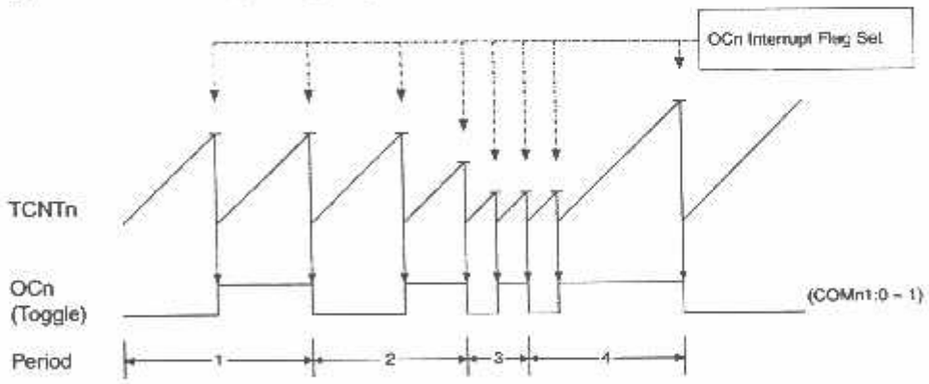
The output compare unit can be used to generate interrupts at some given time. Using the output compare to generate waveforms in Normal mode is not recommended, since this will occupy too much of the CPU time.

Timer on Compare (CTC) Mode

In Clear Timer on Compare or CTC mode (WGM01:0 = 2), the OCR0 Register is used to manipulate the counter resolution. In CTC mode the counter is cleared to zero when the counter value (TCNT0) matches the OCR0. The OCR0 defines 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 31. The counter value (TCNT0) increases until a compare match occurs between TCNT0 and OCR0, and then counter (TCNT0) is cleared.

Figure 31. CTC Mode, Timing Diagram



An Interrupt can be generated each time the counter value reaches the TOP value by using the OCF0 Flag. If the interrupt is enabled, the interrupt handler routine can be used for updating the TOP value. However, changing 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 OCR0 is lower than the current value of TCNT0, the counter will miss the compare match. The counter will then have to count to its maximum value (0xFF) and wrap around starting at 0x00 before the compare match can occur.

For generating a waveform output in CTC mode, the OC0 output can be set to toggle its logical level on each compare match by setting the Compare Output mode bits to toggle mode (COM01:0 = 1). The OC0 value will not be visible on the port pin unless the data direction for the pin is set to output. The waveform generated will have a maximum frequency of $f_{OC0} = f_{clk\_I/O}/2$ when OCR0 is set to zero (0x00). The waveform frequency is defined by the following equation:

$$f_{OCn} = \frac{f_{clk\_I/O}}{2 \cdot N \cdot (1 + OCRn)}$$

The N variable represents the prescale factor (1, 8, 64, 256, or 1024).

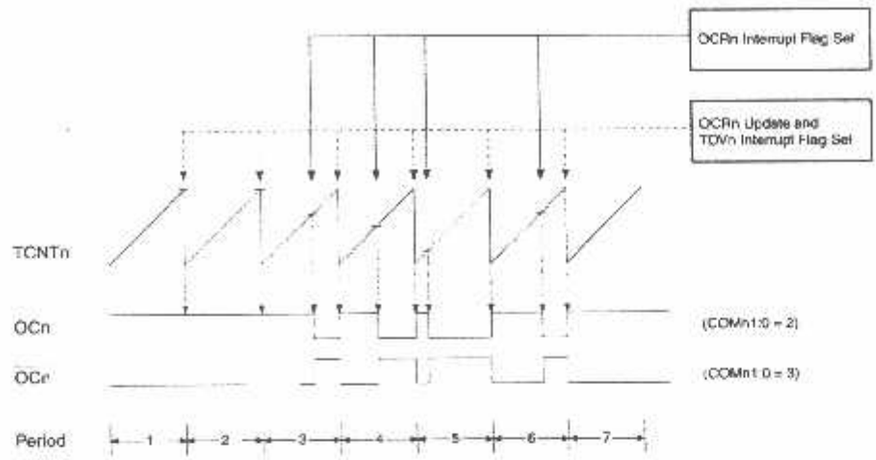
As for the Normal mode of operation, the TOV0 Flag is set in the same timer clock cycle that the counter counts from MAX to 0x00.

'WM Mode

The fast Pulse Width Modulation or fast PWM mode (WGM01:0 = 3) provides a high frequency PWM waveform generation option. The fast PWM differs from the other PWM option by its single-slope operation. The counter counts from BOTTOM to MAX then restarts from BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC0) is cleared on the compare match between TCNT0 and OCR0, and set at BOTTOM. In inverting Compare Output mode, the output is set on compare match and cleared at BOTTOM. Due to the single-slope operation, the operating frequency of the fast PWM mode can be twice as nigh as the phase correct PWM mode 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), and therefore reduces total system cost.

In fast PWM mode, the counter is incremented until the counter value matches the MAX value. The counter is then cleared at the following timer clock cycle. The timing diagram for the fast PWM mode is shown in Figure 32. The TCNT0 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 TCNT0 slopes represent compare matches between OCR0 and TCNT0.

Figure 32. Fast PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches MAX. If the interrupt is enabled, the interrupt handler routine can be used for updating the compare value.

In fast PWM mode, the compare unit allows generation of PWM waveforms on the OC0 pin. Setting the COM01:0 bits to 2 will produce a non-inverted PWM and an inverted PWM output can be generated by setting the COM01:0 to 3 (See Table 40 on page 82). The actual OC0 value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by setting (or clearing) the OC0 Register at the compare match between OCR0 and TCNT0, and clearing (or setting) the OC0 Register at the timer clock cycle the counter is cleared (changes from MAX to BOTTOM).

The PWM frequency for the output can be calculated by the following equation:

$$f_{OCnPWM} = \frac{f_{clk\_I/O}}{N \cdot 256}$$

The N variable represents the prescale factor (1, 8, 64, 256, or 1024).

The extreme values for the OCR0 Register represents special cases when generating a PWM waveform output in the fast PWM mode. If the OCR0 is set equal to BOTTOM, the output will be a narrow spike for each MAX+1 timer clock cycle. Setting the OCR0 equal to MAX will result in a constantly high or low output (depending on the polarity of the output set by the COM01:0 bits.)

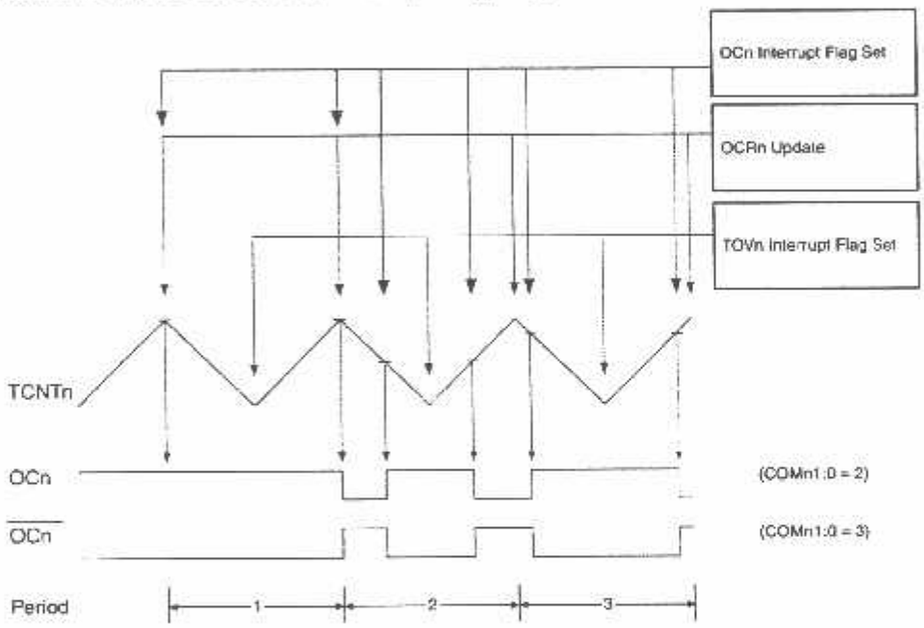
A frequency (with 50% duty cycle) waveform output in fast PWM mode can be achieved by setting OC0 to toggle its logical level on each compare match (COM01:0 = 1). The waveform generated will have a maximum frequency of $f_{OC0} = f_{clk\_I/O}/2$ when OCR0 is set to zero. This feature is similar to the OC0 toggle in CTC mode, except the double buffer feature of the output compare unit is enabled in the fast PWM mode.

se Correct PWM Mode

The phase correct PWM mode (WGM01:0 = 1) provides a high resolution phase correct PWM waveform generation option. The phase correct PWM mode is based on a dual-slope operation. The counter counts repeatedly from BOTTOM to MAX and then from MAX to BOTTOM. In non-inverting Compare Output mode, the Output Compare (OC0) is cleared on the compare match between TCNT0 and OCR0 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 is fixed to eight bits. In phase correct PWM mode the counter is incremented until the counter value matches MAX. When the counter reaches MAX, it changes the count direction. The TCNT0 value will be equal to MAX for one timer clock cycle. The timing diagram for the phase correct PWM mode is shown on Figure 33. The TCNT0 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 TCNT0 slopes represent compare matches between OCR0 and TCNT0.

Figure 33. Phase Correct PWM Mode, Timing Diagram



The Timer/Counter Overflow Flag (TOV0) is set each time the counter reaches BOTTOM. The Interrupt Flag can be used to generate an interrupt each time the counter reaches the BOTTOM value.

In phase correct PWM mode, the compare unit allows generation of PWM waveforms on the OC0 pin. Setting the COM01:0 bits to 2 will produce a non-inverted PWM. An inverted PWM output can be generated by setting the COM01:0 to 3 (see Table 41 on page 82). The actual OC0 value will only be visible on the port pin if the data direction for the port pin is set as output. The PWM waveform is generated by clearing (or setting) the OC0 Register at the compare match between OCR0 and TCNT0 when the counter increments, and setting (or clearing) the OC0 Register at compare match between

OCR0 and TCNT0 when the counter decrements. The PWM frequency for the output when using phase correct PWM can be calculated by the following equation:

$$f_{OCnPCPWM} = \frac{f_{clk\_I/O}}{N \cdot 510}$$

The N variable represents the prescale factor (1, 8, 64, 256, or 1024).

The extreme values for the OCR0 Register represent special cases when generating a PWM waveform output in the phase correct PWM mode. If the OCR0 is set equal to BOTTOM, the output will be continuously low and if set equal to MAX the output will be continuously high for non-inverted PWM mode. For inverted PWM the output will have the opposite logic values.

At the very start of Period 2 in Figure 33 OCn has a transition from high to low even though there is no Compare Match. The point of this transition is to guarantee symmetry around BOTTOM. There are two cases that give a transition without Compare Match:

- OCR0A changes its value from MAX, like in Figure 33. When the OCR0A value is MAX the OCn pin value is the same as the result of a down-counting Compare Match. To ensure symmetry around BOTTOM the OCn value at MAX must be correspond to the result of an up-counting Compare Match.
- The Timer starts counting from a value higher than the one in OCR0A, and for that reason misses the Compare Match and hence the OCn change that would have happened on the way up.

er/Counter Timing grams

The Timer/Counter is a synchronous design and the timer clock (clk_{T0}) is therefore shown as a clock enable signal in the following figures. The figures include information on when Interrupt Flags are set. Figure 34 contains timing data for basic Timer/Counter operation. The figure shows the count sequence close to the MAX value in all modes other than phase correct PWM mode.

Figure 34. Timer/Counter Timing Diagram, no Prescaling

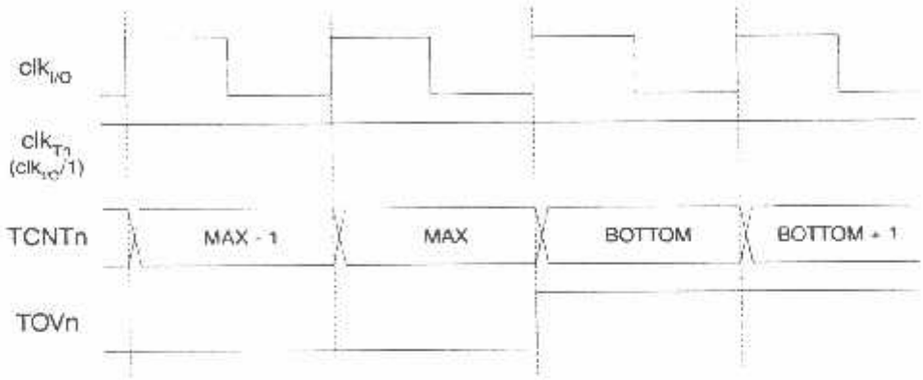


Figure 35 shows the same timing data, but with the prescaler enabled.

Figure 35. Timer/Counter Timing Diagram, with Prescaler ($f_{clk\_I/O}/8$)

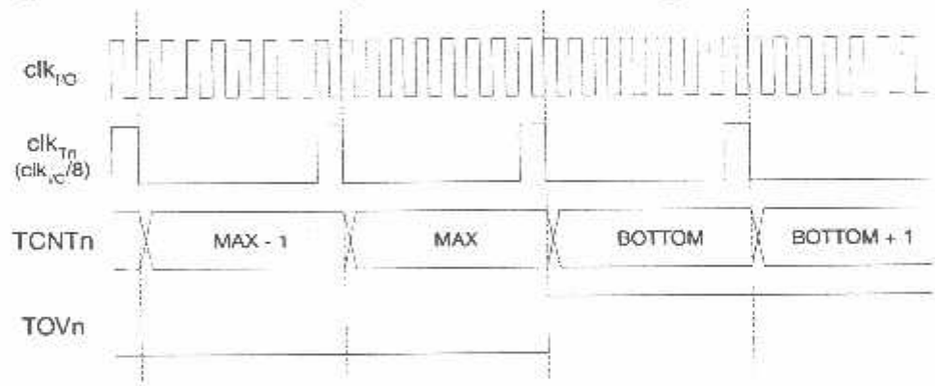


Figure 36 shows the setting of OCF0 in all modes except CTC mode.

Figure 36. Timer/Counter Timing Diagram, Setting of OCF0, with Prescaler ($f_{clk\_I/O}/8$)

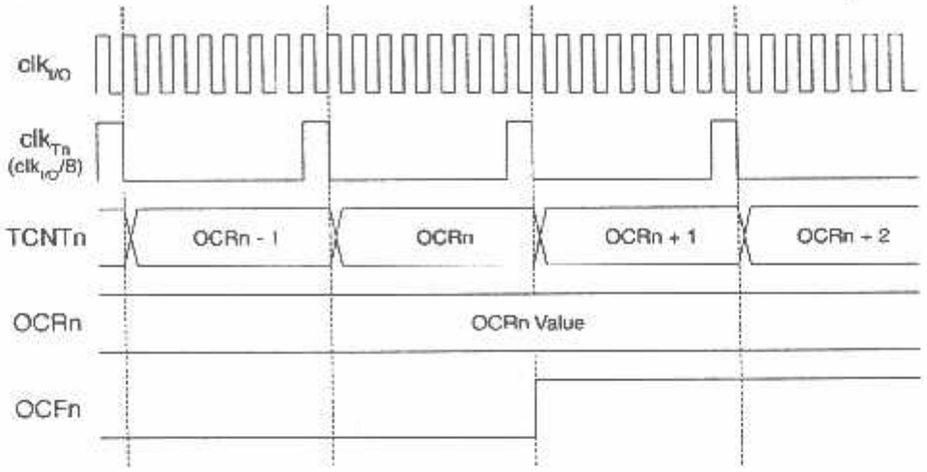
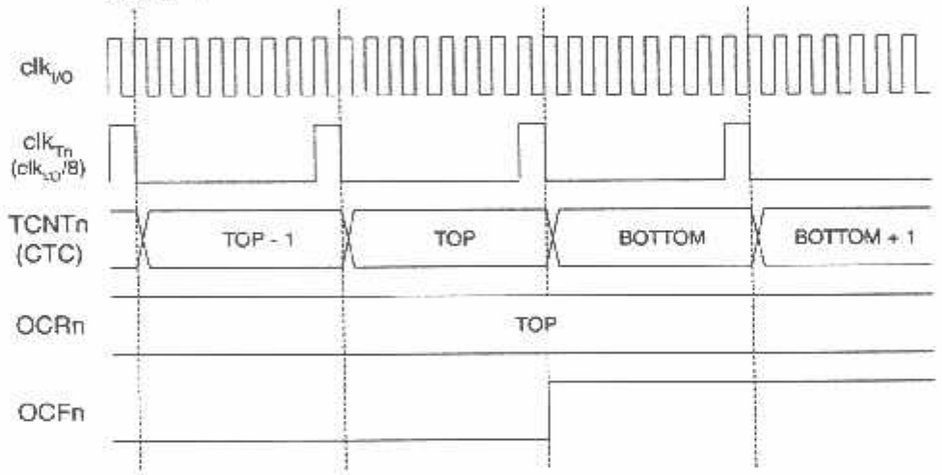


Figure 37 shows the setting of OCF0 and the clearing of TCNT0 in CTC mode.

Figure 37. Timer/Counter Timing Diagram, Clear Timer on Compare Match Mode, with Prescaler ($f_{clk\_I/O}/8$)



it Timer/Counter
;ister Description

er/Counter Control
ister – TCCR0

Bit	7	6	5	4	3	2	1	0	
	FOC0	WGM00	COM01	COM00	WGM01	CS02	CS01	CS00	TCCR0
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	

- **Bit 7 – FOC0: Force Output Compare**

The FOC0 bit is only active when the WGM00 bit specifies a non-PWM mode. However, for ensuring compatibility with future devices, this bit must be set to zero when TCCR0 is written when operating in PWM mode. When writing a logical one to the FOC0 bit, an immediate compare match is forced on the Waveform Generation unit. The OC0 output is changed according to its COM01:0 bits setting. Note that the FOC0 bit is implemented as a strobe. Therefore it is the value present in the COM01:0 bits that determines the effect of the forced compare.

A FOC0 strobe will not generate any interrupt, nor will it clear the timer in CTC mode using OCR0 as TOP.

The FOC0 bit is always read as zero.

- **Bit 6, 3 – WGM01:0: Waveform Generation Mode**

These bits control the counting sequence of the counter, the source for the maximum (TOP) counter value, and what type of Waveform Generation to be used. Modes of operation supported by the Timer/Counter unit are: Normal mode, Clear Timer on Compare Match (CTC) mode, and two types of Pulse Width Modulation (PWM) modes. See Table 38 and "Modes of Operation" on page 74.

Table 38. Waveform Generation Mode Bit Description[1]

Mode	WGM01 (CTC0)	WGM00 (PWM0)	Timer/Counter Mode of Operation	TOP	Update of OCR0	TOV0 Flag Set-on
0	0	0	Normal	0xFF	Immediate	MAX
1	0	1	PWM, Phase Correct	0xFF	TOP	BOTTOM
2	1	0	CTC	OCR0	Immediate	MAX
3	1	1	Fast PWM	0xFF	TOP	MAX

Note: 1. The CTC0 and PWM0 bit definition names are now obsolete. Use the WGM01:0 definitions. However, the functionality and location of these bits are compatible with previous versions of the timer.

- **Bit 5:4 – COM01:0: Compare Match Output Mode**

These bits control the Output Compare pin (OC0) behavior. If one or both of the COM01:0 bits are set, the OC0 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 OC0 pin must be set in order to enable the output driver.

When OC0 is connected to the pin, the function of the COM01:0 bits depends on the WGM01:0 bit setting. Table 39 shows the COM01:0 bit functionality when the WGM01:0 bits are set to a normal or CTC mode (non-PWM).

Table 39. Compare Output Mode, non-PWM Mode

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Toggle OC0 on compare match
1	0	Clear OC0 on compare match
1	1	Set OC0 on compare match

Table 40 shows the COM01:0 bit functionality when the WGM01:0 bits are set to fast PWM mode.

Table 40. Compare Output Mode, Fast PWM Mode[1]

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match, set OC0 at TOP
1	1	Set OC0 on compare match, clear OC0 at TOP

Note: 1. A special case occurs when OCR0 equals TOP and COM01 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 75 for more details.

Table 41 shows the COM01:0 bit functionality when the WGM01:0 bits are set to phase correct PWM mode.

Table 41. Compare Output Mode, Phase Correct PWM Mode[1]

COM01	COM00	Description
0	0	Normal port operation, OC0 disconnected.
0	1	Reserved
1	0	Clear OC0 on compare match when up-counting. Set OC0 on compare match when downcounting.
1	1	Set OC0 on compare match when up-counting. Clear OC0 on compare match when downcounting.

Note: 1. A special case occurs when OCR0 equals TOP and COM01 is set. In this case, the compare match is ignored, but the set or clear is done at TOP. See "Phase Correct PWM Mode" on page 77 for more details.

- **Bit 2:0 – CS02:0: Clock Select**

The three Clock Select bits select the clock source to be used by the Timer/Counter.

Table 42. Clock Select Bit Description

CS02	CS01	CS00	Description
0	0	0	No clock source (Timer/Counter stopped).
0	0	1	$clk_{I/O}$/(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 T0 pin. Clock on falling edge.
1	1	1	External clock source on T0 pin. Clock on rising edge.

If external pin modes are used for the Timer/Counter0, transitions on the T0 pin will clock the counter even if the pin is configured as an output. This feature allows software control of the counting.

/Counter Register – 0

Bit	7	6	5	4	3	2	1	0	
	TCNT0[7:0]								TCNT0
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 Timer/Counter Register gives direct access, both for read and write operations, to the Timer/Counter unit 8-bit counter. Writing to the TCNT0 Register blocks (removes) the compare match on the following timer clock. Modifying the counter (TCNT0) while the counter is running, introduces a risk of missing a compare match between TCNT0 and the OCR0 Register.

! Compare Register –

Bit	7	6	5	4	3	2	1	0	
	OCR0[7:0]								OCR0
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 Register contains an 8-bit value that is continuously compared with the counter value (TCNT0). A match can be used to generate an output compare interrupt, or to generate a waveform output on the OC0 pin.

:ounter Interrupt Mask :r – TIMSK

Bit	7	6	5	4	3	2	1	0	
	OCIE2	TOIE2	TICIE1	OCIE1A	OCIE1B	TOIE1	OCIE0	TOIE0	TIMSK
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 1 – OCIE0: Timer/Counter0 Output Compare Match Interrupt Enable**

When the OCIE0 bit is written to one, and the I-bit in the Status Register is set (one), the Timer/Counter0 Compare Match interrupt is enabled. The corresponding interrupt is

executed if a compare match in Timer/Counter0 occurs, i.e., when the OCF0 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

- **Bit 0 – TOIE0: Timer/Counter0 Overflow Interrupt Enable**

When the TOIE0 bit is written to one, and the I-bit in the Status Register is set (one), the Timer/Counter0 Overflow interrupt is enabled. The corresponding interrupt is executed if an overflow in Timer/Counter0 occurs, i.e., when the TOV0 bit is set in the Timer/Counter Interrupt Flag Register – TIFR.

ɪr/Counter Interrupt Flag ster – TIFR

Bit	7	6	5	4	3	2	1	0	
	OCF2	TOV2	ICF1	OCF1A	OCF1B	TOV1	OCF0	TOV0	TIFR
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 1 – OCF0: Output Compare Flag 0**

The OCF0 bit is set (one) when a compare match occurs between the Timer/Counter0 and the data in OCR0 – Output Compare Register0. OCF0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, OCF0 is cleared by writing a logic one to the flag. When the I-bit in SREG, OCIE0 (Timer/Counter0 Compare Match Interrupt Enable), and OCF0 are set (one), the Timer/Counter0 Compare Match Interrupt is executed.

- **Bit 0 – TOV0: Timer/Counter0 Overflow Flag**

The bit TOV0 is set (one) when an overflow occurs in Timer/Counter0. TOV0 is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, TOV0 is cleared by writing a logic one to the flag. When the SREG I-bit, TOIE0 (Timer/Counter0 Overflow Interrupt Enable), and TOV0 are set (one), the Timer/Counter0 Overflow interrupt is executed. In phase correct PWM mode, this bit is set when Timer/Counter0 changes counting direction at $00.

ıer/Counter0 and ıer/Counter1 ıscalers

Timer/Counter1 and Timer/Counter0 share the same prescaler module, but the Timer/Counters can have different prescaler settings. The description below applies to both Timer/Counter1 and Timer/Counter0.

nal Clock Source

The Timer/Counter can be clocked directly by the system clock (by setting the CSn2:0 = 1). This provides the fastest operation, with a maximum Timer/Counter clock frequency equal to system clock frequency ($f_{CLK\_I/O}$). Alternatively, one of four taps from the prescaler can be used as a clock source. The prescaled clock has a frequency of either $f_{CLK\_I/O}/8$, $f_{CLK\_I/O}/64$, $f_{CLK\_I/O}/256$, or $f_{CLK\_I/O}/1024$.

:aler Reset

The prescaler is free running, i.e., operates independently of the clock select logic of the Timer/Counter, and it is shared by Timer/Counter1 and Timer/Counter0. Since the prescaler is not affected by the Timer/Counter's clock select, the state of the prescaler will have implications for situations where a prescaled clock is used. One example of prescaling artifacts occurs when the timer is enabled and clocked by the prescaler (6 > CSn2:0 > 1). The number of system clock cycles from when the timer is enabled to the first count occurs can be from 1 to N+1 system clock cycles, where N equals the prescaler divisor (8, 64, 256, or 1024).

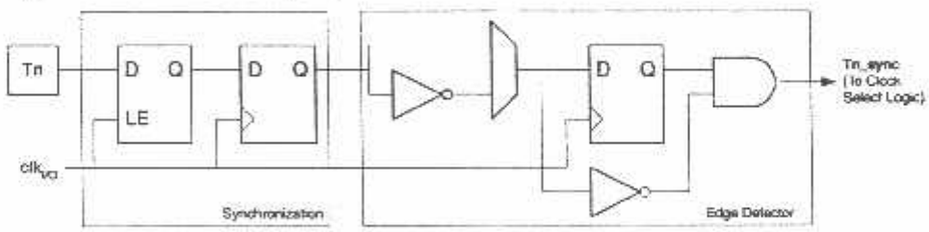
It is possible to use the Prescaler Reset for synchronizing the Timer/Counter to program execution. However, care must be taken if the other Timer/Counter that shares the same prescaler also uses prescaling. A prescaler reset will affect the prescaler period for all Timer/Counters it is connected to.

ıal Clock Source

An external clock source applied to the T1/T0 pin can be used as Timer/Counter clock (clk_{T1}/clk_{T0}). The T1/T0 pin is sampled once every system clock cycle by the pin synchronization logic. The synchronized (sampled) signal is then passed through the edge detector. Figure 38 shows a functional equivalent block diagram of the T1/T0 synchronization and edge detector logic. The registers are clocked at the positive edge of the internal system clock ($clk_{I/O}$). The latch is transparent in the high period of the internal system clock.

The edge detector generates one clk_{T1}/clk_{T0} pulse for each positive (CSn2:0 = 7) or negative (CSn2:0 = 6) edge it detects.

Figure 38. T1/T0 Pin Sampling



The synchronization and edge detector logic introduces a delay of 2.5 to 3.5 system clock cycles from an edge has been applied to the T1/T0 pin to the counter is updated.

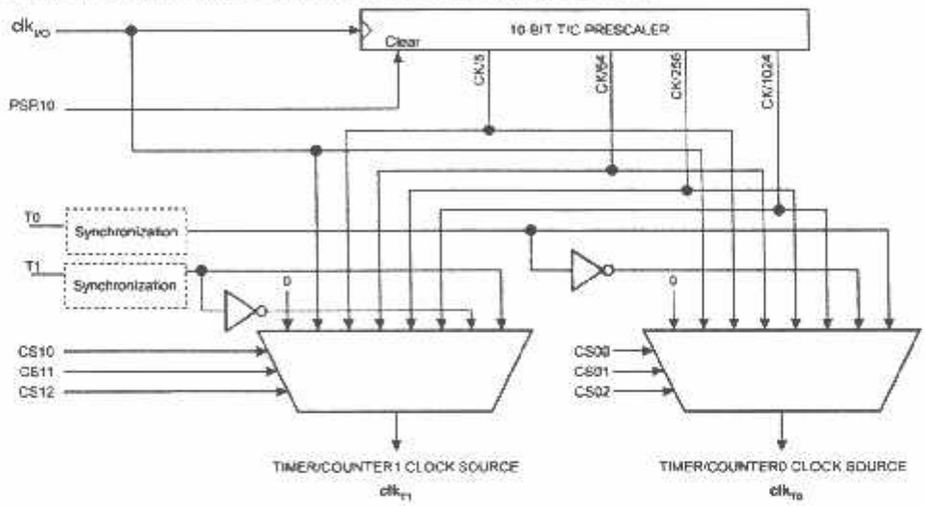
Enabling and disabling of the clock input must be done when T1/T0 has been stable for at least one system clock cycle, otherwise it is a risk that a false Timer/Counter clock pulse is generated.

Each half period of the external clock applied must be longer than one system clock cycle to ensure correct sampling. The external clock must be guaranteed to have less

than half the system clock frequency ($f_{ExtClk} < f_{clk\_I/O}/2$) given a 50/50% duty cycle. Since the edge detector uses sampling, the maximum frequency of an external clock it can detect is half the sampling frequency (Nyquist sampling theorem). However, due to variation of the system clock frequency and duty cycle caused by Oscillator source (crystal, resonator, and capacitors) tolerances, it is recommended that maximum frequency of an external clock source is less than $f_{clk\_I/O}/2.5$.

An external clock source can not be prescaled.

Figure 39. Prescaler for Timer/Counter0 and Timer/Counter1[1]



Note: 1. The synchronization logic on the input pins (T1/T0) is shown in Figure 38.

al Function IO Register –
I

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 0 – PSR10: Prescaler Reset Timer/Counter1 and Timer/Counter0**

When this bit is written to one, the Timer/Counter1 and Timer/Counter0 prescaler will be reset. The bit will be cleared by hardware after the operation is performed. Writing a zero to this bit will have no effect. Note that Timer/Counter1 and Timer/Counter0 share the same prescaler and a reset of this prescaler will affect both timers. This bit will always be read as zero.

ıalog to Digital
ınverter

ıtures

- 10-bit Resolution
- 0.5 LSB Integral Non-linearity
- ±2 LSB Absolute Accuracy
- 13 - 260 µs Conversion Time
- Up to 15 kSPS at Maximum Resolution
- 8 Multiplexed Single Ended Input Channels
- 7 Differential Input Channels
- 2 Differential Input Channels with Optional Gain of 10x and 200x[1]
- Optional Left adjustment for ADC Result Readout
- 0 - V_{CC} ADC Input Voltage Range
- Selectable 2.56V ADC Reference Voltage
- Free Running or Single Conversion Mode
- ADC Start Conversion by Auto Triggering on Interrupt Sources
- Interrupt on ADC Conversion Complete
- Sleep Mode Noise Canceler

Note: 1. The differential input channels are not tested for devices in PDIP Package. This feature is only guaranteed to work for devices in TQFP and QFN/MLF Packages

The ATmega16 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows 8 single-ended voltage inputs constructed from the pins of Port A. The single-ended voltage inputs refer to 0V (GND).

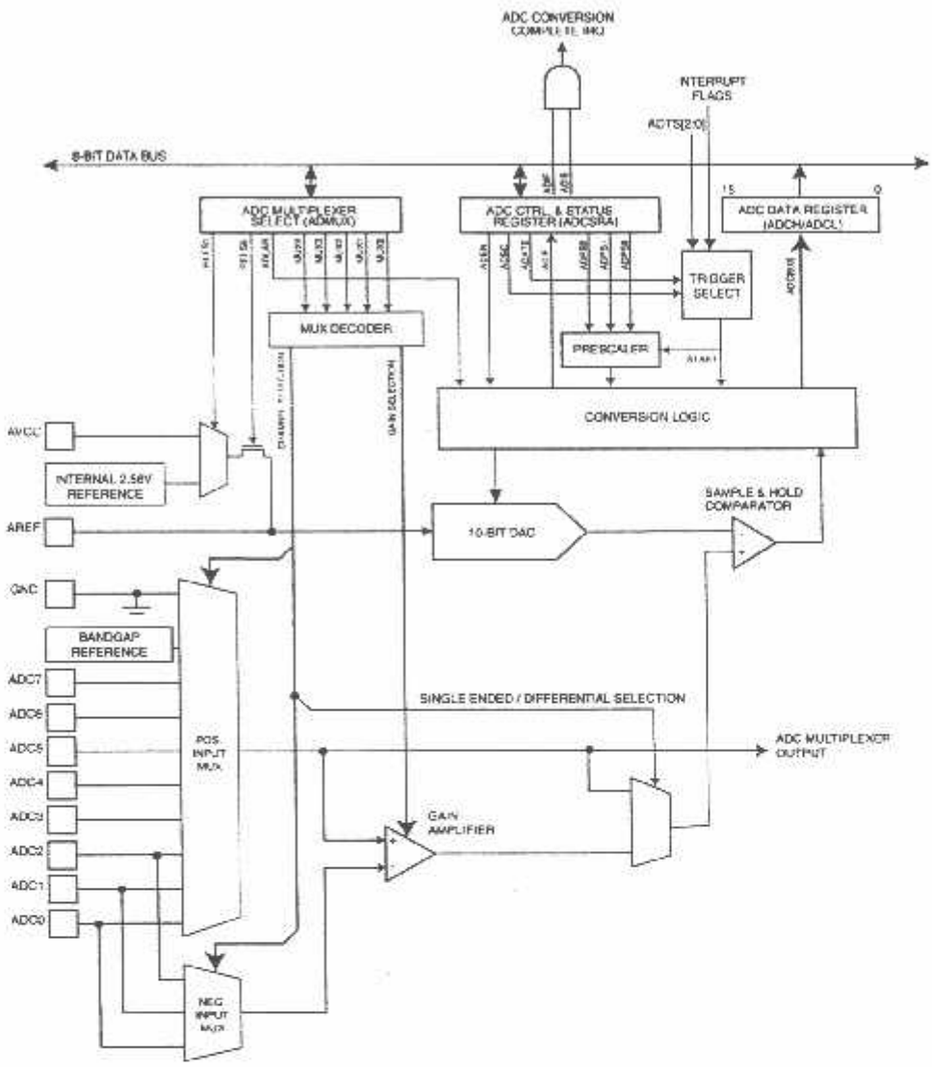
The device also supports 16 differential voltage input combinations. Two of the differential inputs (ADC1, ADC0 and ADC3, ADC2) are equipped with a programmable gain stage, providing amplification steps of 0 dB (1x), 20 dB (10x), or 46 dB (200x) on the differential input voltage before the A/D conversion. Seven differential analog input channels share a common negative terminal (ADC1), while any other ADC input can be selected as the positive input terminal. If 1x or 10x gain is used, 8-bit resolution can be expected. If 200x gain is used, 7-bit resolution can be expected.

The ADC contains a Sample and Hold circuit which ensures that the input voltage to the ADC is held at a constant level during conversion. A block diagram of the ADC is shown in Figure 98.

The ADC has a separate analog supply voltage pin, AVCC. AVCC must not differ more than ±0.3 V from V_{CC}. See the paragraph "ADC Noise Canceler" on page 210 on how to connect this pin.

Internal reference voltages of nominally 2.56V or AVCC are provided On-chip. The voltage reference may be externally decoupled at the AREF pin by a capacitor for better noise performance.

Figure 98. Analog to Digital Converter Block Schematic



eration

The ADC converts an analog input voltage to a 10-bit digital value through successive approximation. The minimum value represents GND and the maximum value represents the voltage on the AREF pin minus 1 LSB. Optionally, AVCC or an internal 2.56V reference voltage may be connected to the AREF pin by writing to the REFSn bits in the ADMUX Register. The internal voltage reference may thus be decoupled by an external capacitor at the AREF pin to improve noise immunity.

The analog input channel and differential gain are selected by writing to the MUX bits in ADMUX. Any of the ADC input pins, as well as GND and a fixed bandgap voltage reference, can be selected as single ended inputs to the ADC. A selection of ADC input pins can be selected as positive and negative inputs to the differential gain amplifier.

If differential channels are selected, the differential gain stage amplifies the voltage difference between the selected input channel pair by the selected gain factor. This

amplified value then becomes the analog input to the ADC. If single ended channels are used, the gain amplifier is bypassed altogether.

The ADC is enabled by setting the ADC Enable bit, ADEN in ADCSRA. Voltage reference and input channel selections will not go into effect until ADEN is set. The ADC does not consume power when ADEN is cleared, so it is recommended to switch off the ADC before entering power saving sleep modes.

The ADC generates a 10-bit result which is presented in the ADC Data Registers, ADCH and ADCL. By default, the result is presented right adjusted, but can optionally be presented left adjusted by setting the ADLAR bit in ADMUX.

If the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH, to ensure that the content of the Data Registers belongs to the same conversion. Once ADCL is read, ADC access to Data Registers is blocked. This means that if ADCL has been read, and a conversion completes before ADCH is read, neither register is updated and the result from the conversion is lost. When ADCH is read, ADC access to the ADCH and ADCL Registers is re-enabled.

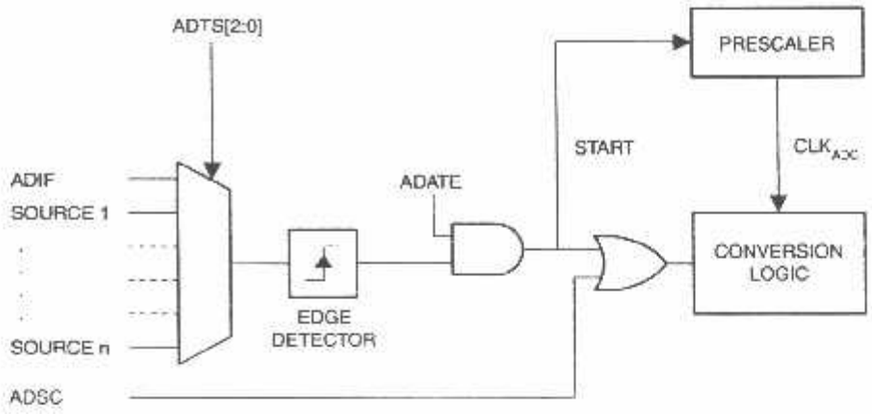
The ADC has its own interrupt which can be triggered when a conversion completes. When ADC access to the Data Registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result is lost.

·ting a Conversion

A single conversion is started by writing a logical one to the ADC Start Conversion bit, ADSC. This bit stays high as long as the conversion is in progress and will be cleared by hardware when the conversion is completed. If a different data channel is selected while a conversion is in progress, the ADC will finish the current conversion before performing the channel change.

Alternatively, a conversion can be triggered automatically by various sources. Auto Triggering is enabled by setting the ADC Auto Trigger Enable bit, ADATE in ADCSRA. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in SFIOR (see description of the ADTS bits for a list of the trigger sources). When a positive edge occurs on the selected trigger signal, the ADC prescaler is reset and a conversion is started. This provides a method of starting conversions at fixed intervals. If the trigger signal still is set when the conversion completes, a new conversion will not be started. If another positive edge occurs on the trigger signal during conversion, the edge will be ignored. Note that an Interrupt Flag will be set even if the specific interrupt is disabled or the global interrupt enable bit in SREG is cleared. A conversion can thus be triggered without causing an interrupt. However, the Interrupt Flag must be cleared in order to trigger a new conversion at the next interrupt event.

Figure 99. ADC Auto Trigger Logic

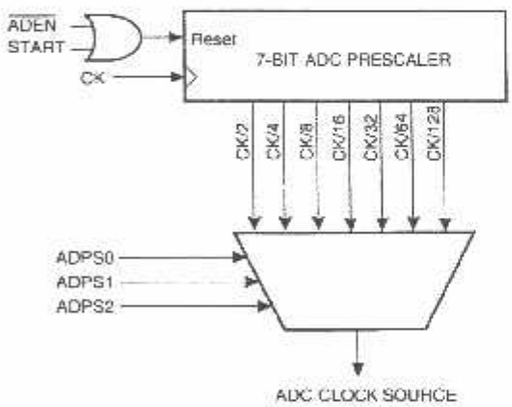


Using the ADC Interrupt Flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The ADC then operates in Free Running mode, constantly sampling and updating the ADC Data Register. The first conversion must be started by writing a logical one to the ADSC bit in ADCSRA. In this mode the ADC will perform successive conversions independently of whether the ADC Interrupt Flag, ADIF is cleared or not.

If Auto Triggering is enabled, single conversions can be started by writing ADSC in ADCSRA to one. ADSC can also be used to determine if a conversion is in progress. The ADSC bit will be read as one during a conversion, independently of how the conversion was started.

caling and ersion Timing

Figure 100. ADC Prescaler



By default, the successive approximation circuitry requires an input clock frequency between 50 kHz and 200 kHz to get maximum resolution. If a lower resolution than 10 bits is needed, the input clock frequency to the ADC can be higher than 200 kHz to get a higher sample rate.

The ADC module contains a prescaler, which generates an acceptable ADC clock frequency from any CPU frequency above 100 kHz. The prescaling is set by the ADPS bits in ADCSRA. The prescaler starts counting from the moment the ADC is switched on by

setting the ADEN bit in ADCSRA. The prescaler keeps running for as long as the ADEN bit is set, and is continuously reset when ADEN is low.

When initiating a single ended conversion by setting the ADSC bit in ADCSRA, the conversion starts at the following rising edge of the ADC clock cycle. See "Differential Gain Channels" on page 208 for details on differential conversion timing.

A normal conversion takes 13 ADC clock cycles. The first conversion after the ADC is switched on (ADEN in ADCSRA is set) takes 25 ADC clock cycles in order to initialize the analog circuitry.

The actual sample-and-hold takes place 1.5 ADC clock cycles after the start of a normal conversion and 13.5 ADC clock cycles after the start of a first conversion. When a conversion is complete, the result is written to the ADC Data Registers, and ADIF is set. In single conversion mode, ADSC is cleared simultaneously. The software may then set ADSC again, and a new conversion will be initiated on the first rising ADC clock edge.

When Auto Triggering is used, the prescaler is reset when the trigger event occurs. This assures a fixed delay from the trigger event to the start of conversion. In this mode, the sample-and-hold takes place 2 ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic. When using Differential mode, along with Auto triggering from a source other than the ADC Conversion Complete, each conversion will require 25 ADC clocks. This is because the ADC must be disabled and re-enabled after every conversion.

In Free Running mode, a new conversion will be started immediately after the conversion completes, while ADSC remains high. For a summary of conversion times, see Table 81.

Figure 101. ADC Timing Diagram, First Conversion (Single Conversion Mode)

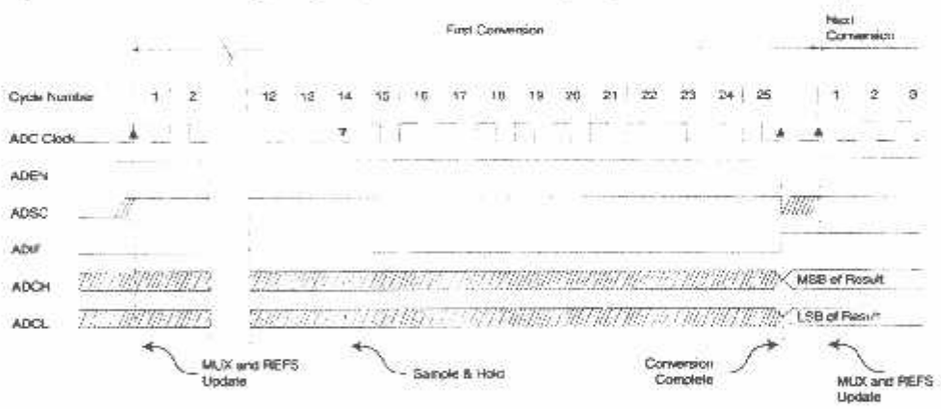


Figure 102. ADC Timing Diagram, Single Conversion

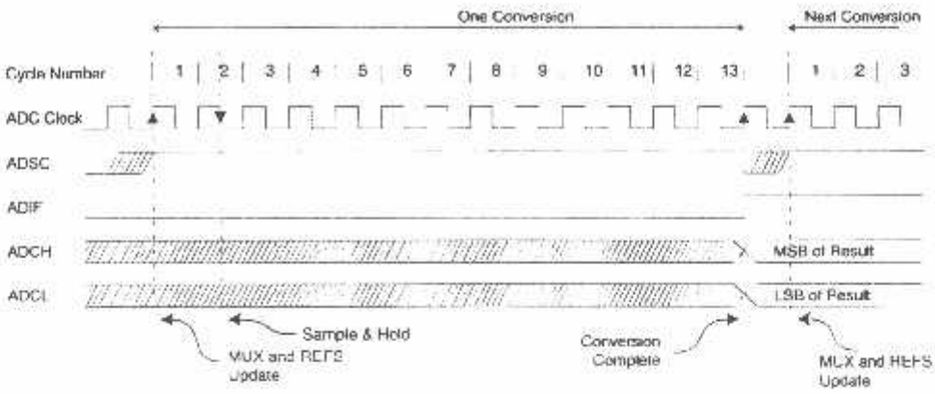


Figure 103. ADC Timing Diagram, Auto Triggered Conversion

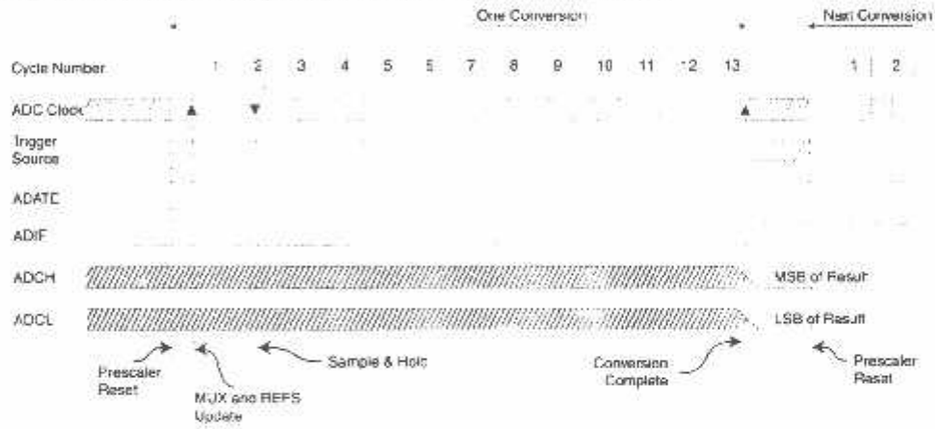


Figure 104. ADC Timing Diagram, Free Running Conversion

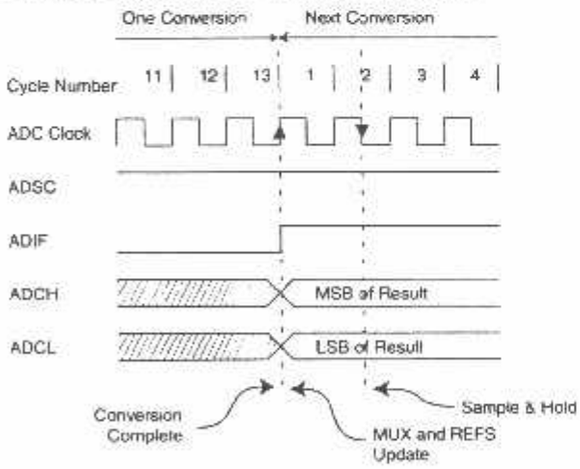


Table 81. ADC Conversion Time

Condition	Sample & Hold (Cycles from Start of Conversion)	Conversion Time (Cycles)
First conversion	13.5	25
Normal conversions, single ended	1.5	13
Auto Triggered conversions	2	13.5
Normal conversions, differential	1.5/2.5	13/14

erential Gain Channels

When using differential gain channels, certain aspects of the conversion need to be taken into consideration.

Differential conversions are synchronized to the internal clock CK_{ADC2} equal to half the ADC clock. This synchronization is done automatically by the ADC interface in such a way that the sample-and-hold occurs at a specific phase of CK_{ADC2}. A conversion initiated by the user (i.e., all single conversions, and the first free running conversion) when CK_{ADC2} is low will take the same amount of time as a single ended conversion (13 ADC clock cycles from the next prescaled clock cycle). A conversion initiated by the user when CK_{ADC2} is high will take 14 ADC clock cycles due to the synchronization mechanism. In Free Running mode, a new conversion is initiated immediately after the previous conversion completes, and since CK_{ADC2} is high at this time, all automatically started (i.e., all but the first) free running conversions will take 14 ADC clock cycles.

The gain stage is optimized for a bandwidth of 4 kHz at all gain settings. Higher frequencies may be subjected to non-linear amplification. An external low-pass filter should be used if the input signal contains higher frequency components than the gain stage bandwidth. Note that the ADC clock frequency is independent of the gain stage bandwidth limitation. For example, the ADC clock period may be 6 µs, allowing a channel to be sampled at 12 kSPS, regardless of the bandwidth of this channel.

If differential gain channels are used and conversions are started by Auto Triggering, the ADC must be switched off between conversions. When Auto Triggering is used, the ADC prescaler is reset before the conversion is started. Since the gain stage is dependent of a stable ADC clock prior to the conversion, this conversion will not be valid. By disabling and then re-enabling the ADC between each conversion (writing ADEN in ADCSRA to "0" then to "1"), only extended conversions are performed. The result from the extended conversions will be valid. See "Prescaling and Conversion Timing" on page 205 for timing details.

ıging Channel or ·ence Selection

The MUXn and REFS1:0 bits in the ADMUX Register are single buffered through a temporary register to which the CPU has random access. This ensures that the channels and reference selection only takes place at a safe point during the conversion. The channel and reference selection is continuously updated until a conversion is started. Once the conversion starts, the channel and reference selection is locked to ensure a sufficient sampling time for the ADC. Continuous updating resumes in the last ADC clock cycle before the conversion completes (ADIF in ADCSRA is set). Note that the conversion starts on the following rising ADC clock edge after ADSC is written. The user is thus advised not to write new channel or reference selection values to ADMUX until one ADC clock cycle after ADSC is written.

If Auto Triggering is used, the exact time of the triggering event can be indeterministic. Special care must be taken when updating the ADMUX Register, in order to control which conversion will be affected by the new settings.

If both ADATE and ADEN is written to one, an Interrupt event can occur at any time. If the ADMUX Register is changed in this period, the user cannot tell if the next conversion is based on the old or the new settings. ADMUX can be safely updated in the following ways:

1. When ADATE or ADEN is cleared.
2. During conversion, minimum one ADC clock cycle after the trigger event.
3. After a conversion before the Interrupt Flag used as trigger source is cleared.

When updating ADMUX in one of these conditions, the new settings will affect the next ADC conversion.

Special care should be taken when changing differential channels. Once a differential channel has been selected, the gain stage may take as much as 125 µs to stabilize to the new value. Thus conversions should not be started within the first 125 µs after selecting a new differential channel. Alternatively, conversion results obtained within this period should be discarded.

The same settling time should be observed for the first differential conversion after changing ADC reference (by changing the REFS1:0 bits in ADMUX).

: Input Channels

When changing channel selections, the user should observe the following guidelines to ensure that the correct channel is selected:

In Single Conversion mode, always select the channel before starting the conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the conversion to complete before changing the channel selection.

In Free Running mode, always select the channel before starting the first conversion. The channel selection may be changed one ADC clock cycle after writing one to ADSC. However, the simplest method is to wait for the first conversion to complete, and then change the channel selection. Since the next conversion has already started automatically, the next result will reflect the previous channel selection. Subsequent conversions will reflect the new channel selection.

When switching to a differential gain channel, the first conversion result may have a poor accuracy due to the required settling time for the automatic offset cancellation circuitry. The user should preferably disregard the first conversion result.

/oltage Reference

The reference voltage for the ADC (V_{REF}) indicates the conversion range for the ADC. Single ended channels that exceed V_{REF} will result in codes close to 0x3FF. V_{REF} can be selected as either AVCC, internal 2.56V reference, or external AREF pin.

AVCC is connected to the ADC through a passive switch. The internal 2.56V reference is generated from the internal bandgap reference (V_{BG}) through an internal amplifier. In either case, the external AREF pin is directly connected to the ADC, and the reference voltage can be made more immune to noise by connecting a capacitor between the AREF pin and ground. V_{REF} can also be measured at the AREF pin with a high impedant voltmeter. Note that V_{REF} is a high impedant source, and only a capacitive load should be connected in a system.

If the user has a fixed voltage source connected to the AREF pin, the user may not use the other reference voltage options in the application, as they will be shorted to the

external voltage. If no external voltage is applied to the AREF pin, the user may switch between AVCC and 2.56V as reference selection. The first ADC conversion result after switching reference voltage source may be inaccurate, and the user is advised to discard this result.

If differential channels are used, the selected reference should not be closer to AVCC than indicated in Table 122 on page 297.

C Noise Canceler

The ADC features a noise canceler that enables conversion during sleep mode to reduce noise induced from the CPU core and other I/O peripherals. The noise canceler can be used with ADC Noise Reduction and Idle mode. To make use of this feature, the following procedure should be used:

1. Make sure that the ADC is enabled and is not busy converting. Single Conversion Mode must be selected and the ADC conversion complete interrupt must be enabled.
2. Enter ADC Noise Reduction mode (or Idle mode). The ADC will start a conversion once the CPU has been halted.
3. If no other interrupts occur before the ADC conversion completes, the ADC interrupt will wake up the CPU and execute the ADC Conversion Complete interrupt routine. If another interrupt wakes up the CPU before the ADC conversion is complete, that interrupt will be executed, and an ADC Conversion Complete interrupt request will be generated when the ADC conversion completes. The CPU will remain in active mode until a new sleep command is executed.

Note that the ADC will not be automatically turned off when entering other sleep modes than Idle mode and ADC Noise Reduction mode. The user is advised to write zero to ADEN before entering such sleep modes to avoid excessive power consumption. If the ADC is enabled in such sleep modes and the user wants to perform differential conversions, the user is advised to switch the ADC off and on after waking up from sleep to prompt an extended conversion to get a valid result.

g Input Circuitry

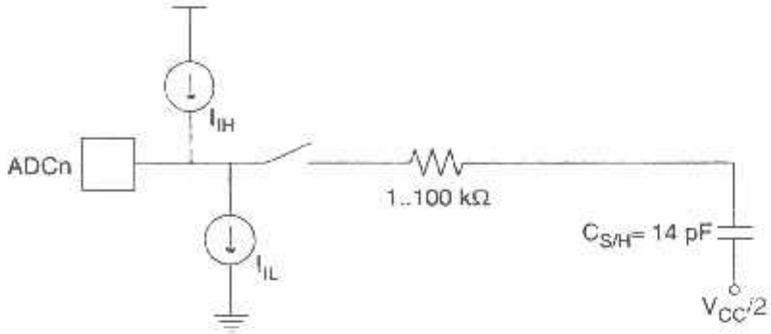
The Analog Input Circuitry for single ended channels is illustrated in Figure 105. An analog source applied to ADCn is subjected to the pin capacitance and input leakage of that pin, regardless of whether that channel is selected as input for the ADC. When the channel is selected, the source must drive the S/H capacitor through the series resistance (combined resistance in the input path).

The ADC is optimized for analog signals with an output impedance of approximately 10 kΩ or less. If such a source is used, the sampling time will be negligible. If a source with higher impedance is used, the sampling time will depend on how long time the source needs to charge the S/H capacitor, with can vary widely. The user is recommended to only use low impedant sources with slowly varying signals, since this minimizes the required charge transfer to the S/H capacitor.

If differential gain channels are used, the input circuitry looks somewhat different, although source impedances of a few hundred kΩ or less is recommended.

Signal components higher than the Nyquist frequency ($f_{ADC}/2$) should not be present for either kind of channels, to avoid distortion from unpredictable signal convolution. The user is advised to remove high frequency components with a low-pass filter before applying the signals as inputs to the ADC.

Figure 105. Analog Input Circuitry

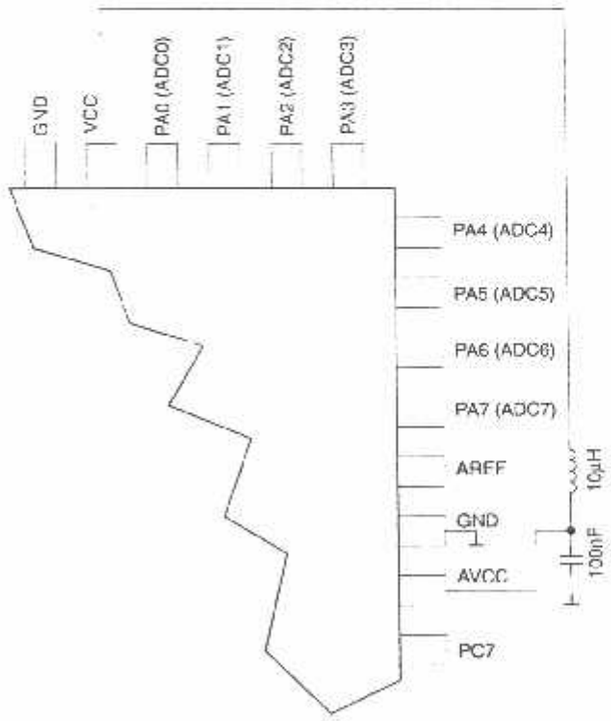


alog Noise Canceling hniques

Digital circuitry inside and outside the device generates EMI which might affect the accuracy of analog measurements. If conversion accuracy is critical, the noise level can be reduced by applying the following techniques:

1. Keep analog signal paths as short as possible. Keep them well away from high-speed switching digital tracks.
2. The AVCC pin on the device should be connected to the digital V_{CC} supply voltage via an LC network as shown in Figure 106.
3. Use the ADC noise canceler function to reduce induced noise from the CPU.
4. If any ADC port pins are used as digital outputs, it is essential that these do not switch while a conversion is in progress.

Figure 106. ADC Power Connections



Iset Compensation hemes

The gain stage has a built-in offset cancellation circuitry that nulls the offset of differential measurements as much as possible. The remaining offset in the analog path can be measured directly by selecting the same channel for both differential inputs. This offset residue can be then subtracted in software from the measurement results. Using this kind of software based offset correction, offset on any channel can be reduced below one LSB.

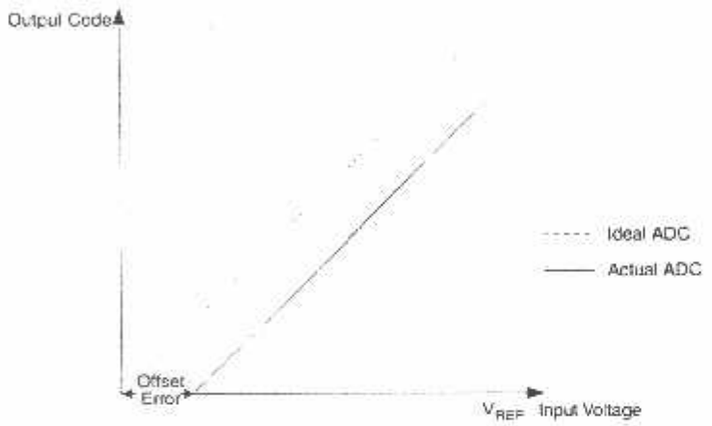
C Accuracy Definitions

An n-bit single-ended ADC converts a voltage linearly between GND and V_{REF} in 2^n steps (LSBs). The lowest code is read as 0, and the highest code is read as 2^n-1.

Several parameters describe the deviation from the ideal behavior:

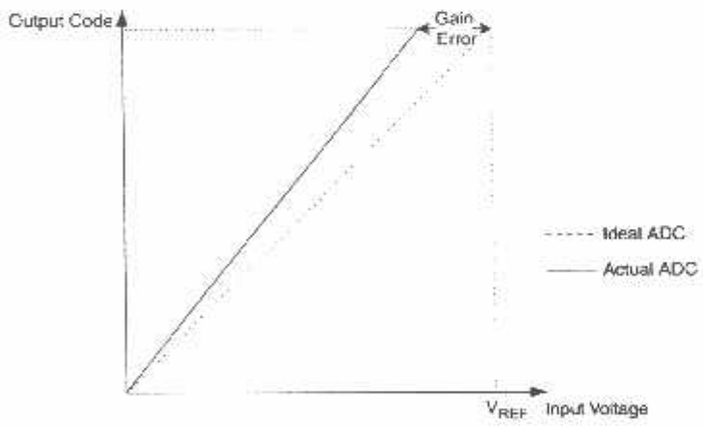
- Offset: The deviation of the first transition (0x000 to 0x001) compared to the ideal transition (at 0.5 LSB). Ideal value: 0 LSB.

Figure 107. Offset Error



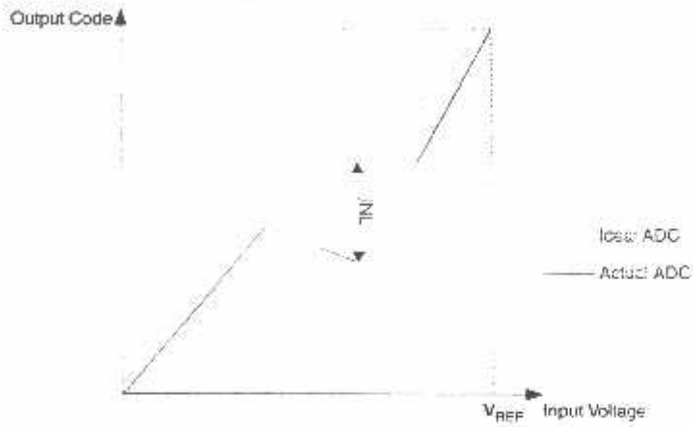
- Gain Error: After adjusting for offset, the Gain Error is found as the deviation of the last transition (0x3FE to 0x3FF) compared to the ideal transition (at 1.5 LSB below maximum). Ideal value: 0 LSB

Figure 108. Gain Error



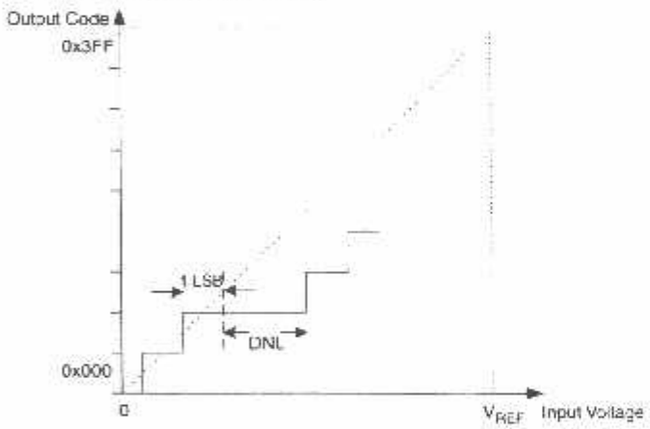
- Integral Non-linearity (INL): After adjusting for offset and gain error, the INL is the maximum deviation of an actual transition compared to an ideal transition for any code. Ideal value: 0 LSB.

Figure 109. Integral Non-linearity (INL)



- Differential Non-linearity (DNL): The maximum deviation of the actual code width (the interval between two adjacent transitions) from the ideal code width (1 LSB). Ideal value: 0 LSB.

Figure 110. Differential Non-linearity (DNL)



- Quantization Error: Due to the quantization of the input voltage into a finite number of codes, a range of input voltages (1 LSB wide) will code to the same value. Always ±0.5 LSB.
- Absolute Accuracy: The maximum deviation of an actual (unadjusted) transition compared to an ideal transition for any code. This is the compound effect of Offset, Gain Error, Differential Error, Non-linearity, and Quantization Error. Ideal value: ±0.5 LSB.

)C Conversion Result

After the conversion is complete (ADIF is high), the conversion result can be found in the ADC Result Registers (ADCL, ADCH).

For single ended conversion, the result is

$$ADC = \frac{V_{IN} \cdot 1024}{V_{REF}}$$

where V_{IN} is the voltage on the selected input pin and V_{REF} the selected voltage reference (see Table 83 on page 215 and Table 84 on page 216). 0x000 represents ground, and 0x3FF represents the selected reference voltage minus one LSB.

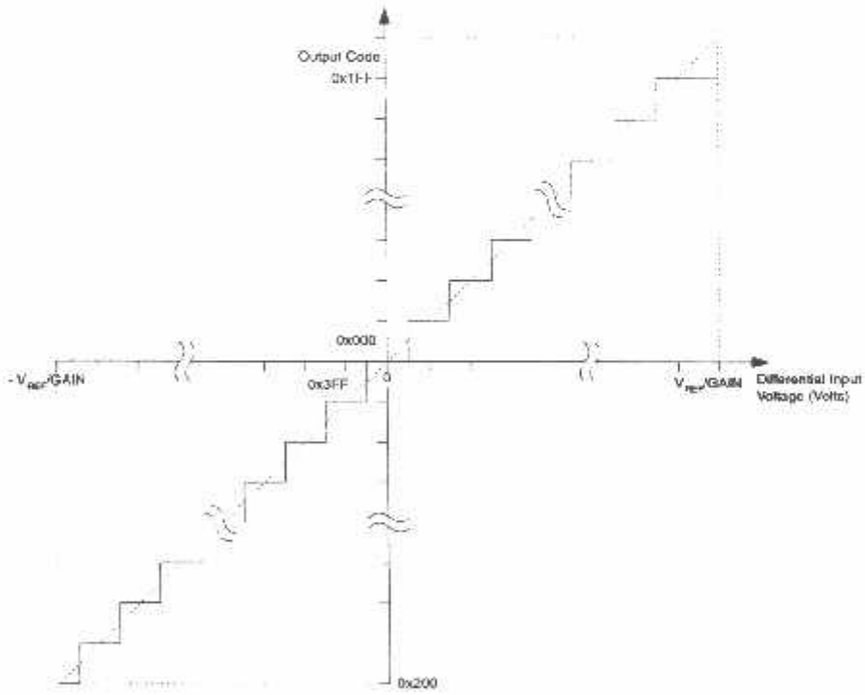
If differential channels are used, the result is

$$ADC = \frac{(V_{POS} - V_{NEG}) \cdot GAIN \cdot 512}{V_{REF}}$$

where V_{POS} is the voltage on the positive input pin. V_{NEG} the voltage on the negative input pin, GAIN the selected gain factor, and V_{REF} the selected voltage reference. The result is presented in two's complement form, from 0x200 (-512d) through 0x1FF (+511d). Note that if the user wants to perform a quick polarity check of the results, it is sufficient to read the MSB of the result (ADC9 in ADCH). If this bit is one, the result is negative, and if this bit is zero, the result is positive. Figure 111 shows the decoding of the differential input range.

Table 82 shows the resulting output codes if the differential input channel pair (ADCn - ADCm) is selected with a gain of GAIN and a reference voltage of V_{REF}.

Figure 111. Differential Measurement Range



regardless of any ongoing conversions. For a complete description of this bit, see "The ADC Data Register – ADCL and ADCH" on page 218.

- **Bits 4:0 – MUX4:0: Analog Channel and Gain Selection Bits**

The value of these bits selects which combination of analog inputs are connected to the ADC. These bits also select the gain for the differential channels. See Table 84 for details. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set).

Table 84. Input Channel and Gain Selections

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
00000	ADC0	N/A		
00001	ADC1			
00010	ADC2			
00011	ADC3			
00100	ADC4			
00101	ADC5			
00110	ADC6			
00111	ADC7			
01000	N/A	ADC0	ADC0	10x
01001		ADC1	ADC0	10x
01010[1]		ADC0	ADC0	200x
01011[1]		ADC1	ADC0	200x
01100		ADC2	ADC2	10x
01101		ADC3	ADC2	10x
01110[1]		ADC2	ADC2	200x
01111[1]		ADC3	ADC2	200x
10000		ADC0	ADC1	1x
10001		ADC1	ADC1	1x
10010		ADC2	ADC1	1x
10011		ADC3	ADC1	1x
10100		ADC4	ADC1	1x
10101		ADC5	ADC1	1x
10110		ADC6	ADC1	1x
10111		ADC7	ADC1	1x
11000		ADC0	ADC2	1x
11001		ADC1	ADC2	1x
11010		ADC2	ADC2	1x
11011		ADC3	ADC2	1x
11100		ADC4	ADC2	1x

Table 84. Input Channel and Gain Selections (Continued)

MUX4..0	Single Ended Input	Positive Differential Input	Negative Differential Input	Gain
11101		ADC5	ADC2	1x
11110	1.22 V (V_{BG})	N/A		
11111	0 V (GND)			

Note: 1. The differential input channels are not tested for devices in PDIP Package. This feature is only guaranteed to work for devices in TQFP and QFN/MLF Packages

DC Control and Status egister A – ADCSRA

Bit	7	6	5	4	3	2	1	0	
	ADEN	ADSC	ADATE	ADIF	ADIE	ADPS2	ADPS1	ADPS0	ADCSRA
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 – ADEN: ADC Enable**

Writing this bit to one enables the ADC. By writing it to zero, the ADC is turned off. Turning the ADC off while a conversion is in progress, will terminate this conversion.

- **Bit 6 – ADSC: ADC Start Conversion**

In Single Conversion mode, write this bit to one to start each conversion. In Free Running Mode, write this bit to one to start the first conversion. The first conversion after ADSC has been written after the ADC has been enabled, or if ADSC is written at the same time as the ADC is enabled, will take 25 ADC clock cycles instead of the normal 13. This first conversion performs initialization of the ADC.

ADSC will read as one as long as a conversion is in progress. When the conversion is complete, it returns to zero. Writing zero to this bit has no effect.

- **Bit 5 – ADATE: ADC Auto Trigger Enable**

When this bit is written to one, Auto Triggering of the ADC is enabled. The ADC will start a conversion on a positive edge of the selected trigger signal. The trigger source is selected by setting the ADC Trigger Select bits, ADTS in SFIOR.

- **Bit 4 – ADIF: ADC Interrupt Flag**

This bit is set when an ADC conversion completes and the Data Registers are updated. The ADC Conversion Complete Interrupt is executed if the ADIE bit and the I-bit in SREG are set. ADIF is cleared by hardware when executing the corresponding interrupt handling vector. Alternatively, ADIF is cleared by writing a logical one to the flag. Beware that if doing a Read-Modify-Write on ADCSRA, a pending interrupt can be disabled. This also applies if the SBI and CBI instructions are used.

- **Bit 3 – ADIE: ADC Interrupt Enable**

When this bit is written to one and the I-bit in SREG is set, the ADC Conversion Complete Interrupt is activated.

- **Bits 2:0 – ADPS2:0: ADC Prescaler Select Bits**

These bits determine the division factor between the XTAL frequency and the input clock to the ADC.

Table 85. ADC Prescaler Selections

ADPS2	ADPS1	ADPS0	Division Factor
0	0	0	2
0	0	1	2
0	1	0	4
0	1	1	8
1	0	0	16
1	0	1	32
1	1	0	64
1	1	1	128

ADC Data Register – L and ADCH

4R = 0

Bit	15	14	13	12	11	10	9	8	
	–	–	–	–	–	–	ADC9	ADC8	ADCH
	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADC1	ADC0	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

R = 1

Bit	15	14	13	12	11	10	9	8	
	ADC9	ADC8	ADC7	ADC6	ADC5	ADC4	ADC3	ADC2	ADCH
	ADC1	ADC0	–	–	–	–	–	–	ADCL
	7	6	5	4	3	2	1	0	
Read/Write	R	R	R	R	R	R	R	R	
	R	R	R	R	R	R	R	R	
Initial Value	0	0	0	0	0	0	0	0	
	0	0	0	0	0	0	0	0	

When an ADC conversion is complete, the result is found in these two registers. If differential channels are used, the result is presented in two's complement form.

When ADCL is read, the ADC Data Register is not updated until ADCH is read. Consequently, if the result is left adjusted and no more than 8-bit precision is required, it is sufficient to read ADCH. Otherwise, ADCL must be read first, then ADCH.

The ADLAR bit in ADMUX, and the MUXn bits in ADMUX affect the way the result is read from the registers. If ADLAR is set, the result is left adjusted. If ADLAR is cleared (default), the result is right adjusted.

- **ADC9:0: ADC Conversion Result**

These bits represent the result from the conversion, as detailed in "ADC Conversion Result" on page 214.

Special FunctionIO Register – SFIOR

Bit	7	6	5	4	3	2	1	0	
	ADTS2	ADTS1	ADTS0	–	ACME	PUD	PSR2	PSR10	SFIOR
Read/Write	R/W	R/W	R/W	R	R/W	R/W	R/W	R/W	
Initial Value	0	0	0	0	0	0	0	0	

- **Bit 7:5 – ADTS2:0: ADC Auto Trigger Source**

If ADATE in ADCSRA is written to one, the value of these bits selects which source will trigger an ADC conversion. If ADATE is cleared, the ADTS2:0 settings will have no effect. A conversion will be triggered by the rising edge of the selected Interrupt Flag. Note that switching from a trigger source that is cleared to a trigger source that is set, will generate a positive edge on the trigger signal. If ADEN in ADCSRA is set, this will start a conversion. Switching to Free Running mode (ADTS[2:0]=0) will not cause a trigger event, even if the ADC Interrupt Flag is set.

Table 86. ADC Auto Trigger Source Selections

ADTS2	ADTS1	ADTS0	Trigger Source
0	0	0	Free Running mode
0	0	1	Analog Comparator
0	1	0	External Interrupt Request 0
0	1	1	Timer/Counter0 Compare Match
1	0	0	Timer/Counter0 Overflow
1	0	1	Timer/Counter Compare Match B
1	1	0	Timer/Counter1 Overflow
1	1	1	Timer/Counter1 Capture Event

- **Bit 4 – Res: Reserved Bit**

This bit is reserved for future use. To ensure compatibility with future devices, this bit must be written to zero when SFIOR is written.

Table 82. Correlation between Input Voltage and Output Codes

V_{ADCn}	Read code	Corresponding Decimal Value
$V_{ADCm} + V_{REF}/GAIN$	0x1FF	511
$V_{ADCm} + 511/512\ V_{REF}/GAIN$	0x1FF	511
$V_{ADCm} + 510/512\ V_{REF}/GAIN$	0x1FE	510
...	...	...
$V_{ADCm} + 1/512\ V_{REF}/GAIN$	0x001	1
V_{ADCm}	0x000	0
$V_{ADCm} - 1/512\ V_{REF}/GAIN$	0x3FF	-1
...	...	...
$V_{ADCm} - 511/512\ V_{REF}/GAIN$	0x201	-511
$V_{ADCm} - V_{REF}/GAIN$	0x200	-512

Example:

ADMUX = 0xED (ADC3 - ADC2, 10x gain, 2.56V reference, left adjusted result)

Voltage on ADC3 is 300 mV, voltage on ADC2 is 500 mV.

ADCR = 512 * 10 * (300 - 500) / 2560 = -400 = 0x270

ADCL will thus read 0x00, and ADCH will read 0x9C. Writing zero to ADLAR right adjusts the result: ADCL = 0x70, ADCH = 0x02.

: Multiplexer Selection Ister – ADMUX

Bit	7	6	5	4	3	2	1	0	
	REFS1	REFS0	ADLAR	MUX4	MUX3	MUX2	MUX1	MUX0	ADMUX
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 7:6 – REFS1:0: Reference Selection Bits**

These bits select the voltage reference for the ADC, as shown in Table 83. If these bits are changed during a conversion, the change will not go in effect until this conversion is complete (ADIF in ADCSRA is set). The internal voltage reference options may not be used if an external reference voltage is being applied to the AREF pin.

Table 83. Voltage Reference Selections for ADC

REFS1	REFS0	Voltage Reference Selection
0	0	AREF, Internal Vref turned off
0	1	AVCC with external capacitor at AREF pin
1	0	Reserved
1	1	Internal 2.56V Voltage Reference with external capacitor at AREF pin

- **Bit 5 – ADLAR: ADC Left Adjust Result**

The ADLAR bit affects the presentation of the ADC conversion result in the ADC Data Register. Write one to ADLAR to left adjust the result. Otherwise, the result is right adjusted. Changing the ADLAR bit will affect the ADC Data Register immediately,

BAB V
PENUTUP

5.1 Kesimpulan

Dari pembahasan Perancangan dan pembuatan *data logger* untuk *monitoring* suhu dan kelembaban pada green house, dapat diambil kesimpulan, yaitu :

1. Hasil Perbandingan antara sensor SHT 11A pada alat dengan Termometer *Digital* tidak terlalu berbeda jauh, karena sensor SHT 11A sudah di kalibrasi saat pembuatannya. Kesalahan rata-rata pada perbandingan yang didapatkan adalah sebesar 1,04%.
2. Alat dapat menyimpan suhu dan kelembaban dan dapat mengirim data melalui wifi

5.2 Saran

Alat yang dibuat ini masih memiliki keterbatasan, nantinya diharapkan dapat dikembangkan untuk mengatasi keterbatasan itu. Sehingga mendapatkan alat yang diharapkan dapat mendekati alat yang ideal.

1. Bagi peneliti selanjutnya diharapkan menggunakan mikrokontroler yang kecepatan eksekusi lebih cepat dibandingkan dengan AT89S52 sehingga dapat mengambil banyak data dalam waktu yang agak singkat. Kecepatan PC juga harus dapat mengimbangi kecepatan mikrokontroller, jika tidak maka akan ada data yang hilang atau tertindih data yang sebelumnya.
2. Bagi peneliti selanjutnya diharapkan untuk melakukan pengujian alat pada greenhouse yang terdapat di kota terdekat agar mendapatkan perbandingan suhu dan kelembaban yang ada didalam miniature green house dengan green house sebenarnya, serta perbandingan alat yang terdapat pada green house dengan sistem *data logger* yang saya buat.