OTOMATISASI ALAT PENGATUR KADAR AIR PADA BENIH JAGUNG HIBRIDA MENGGUNAKAN MIKROKONTROLLER ATMEGA 8535

SKRIPSI

KONSENTRASI TEKNIK SISTEM KONTROL

Diajukan untuk memenuhi persyaratan

Memperoleh gelar Sarjana Teknik

Disusun oleh:

CANDRA KURNIAWAN

NIM. 0510633017-63

KEMENTERIAN PENDIDIKAN NASIONAL

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2010

LEMBAR PERSETUJUAN

OTOMATISASI ALAT PENGATUR KADAR AIR PADA BENIH JAGUNG HIBRIDA MENGGUNAKAN MIKROKONTROLLER ATMEGA 8535

SKRIPSI

KONSENTRASI TEKNIK SISTEM KONTROL

Diajukan untuk memenuhi persyaratan
Memperoleh gelar Sarjana Teknik

Disusun oleh:

CANDRA KURNIAWAN
NIM. 0510633017-63

Telah diperiksa dan disetujui oleh:

Ir. Purwanto, MT.	**Bambang Siswojo, ST., MT.**
NIP. 19540424 198601 1 001	**NIP. 19621211 198802 1 001**

LEMBAR PENGESAHAN

OTOMATISASI ALAT PENGATUR KADAR AIR PADA BENIH JAGUNG HIBRIDA MENGGUNAKAN MIKROKONTROLLER ATMEGA 8535

SKRIPSI

KONSENTRASI TEKNIK SISTEM KONTROL

Diajukan untuk memenuhi persyaratan
Memperoleh gelar Sarjana Teknik

Disusun oleh:

CANDRA KURNIAWAN
NIM. 0510633017-63

Skripsi ini telah di uji dan dinyatakan lulus pada
Tanggal 27 Desember 2010

MAJELIS PENGUJI

Goegoes Dwi Nusantoro, ST., MT	**Erni Yudaningtyas, Ir., MT**
NIP. 19711013 200604 1 001	**NIP. 19650913 199002 2 001**

Ponco Siwindarto, Ir., MS
NIP. 19590304 198903 1 001

Mengetahui
Ketua Jurusan Teknik Elektro

Rudy Yuwono, ST., MSc
NIP. 19710615 199802 1 003

PENGANTAR

Assalamualaikum wr.wb. Alhamdulillah, puji dan syukur penulis panjatkan kehadirat Allah SWT yang telah memberikan rahmat dan hidayah-Nya, sehingga penulis dapat menyelesaikan skripsi ini dengan baik. Skripsi berjudul **"Otomatisasi Alat Pengatur Kadar Air Pada Benih Jagung Hibrida Menggunakan Mikrokontroller ATmega 8535"** ini disusun sebagai syarat untuk memperoleh gelar Sarjana Teknik di Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.

Pada kesempatan ini penulis ingin menyampaikan ucapan terima kasih kepada :

1. Ibu dan Bapak tercinta, kakak-kakak dan adik-adikku, serta saudara-saudaraku yang telah banyak memberikan kasih sayang, dukungan dan doa.
2. Bapak Rudy Yuwono, ST., M.Sc selaku Ketua Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya.
3. Bapak M.Aziz Muslim, ST.,MT.,Ph.D selaku Sekretaris Jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya
4. Bapak Ir.Purwanto, MT. selaku KKDK Sistem Kontrol dan juga sebagai dosen pembimbing skripsi yang telah banyak memberikan saran, motivasi, serta pengarahan dalam penyusunan skripsi ini.
5. Bapak Bambang Siswojo, ST., MT. selaku dosen pembimbing skripsi yang telah banyak memberikan saran, motivasi, serta pengarahan dalam penyusunan skripsi ini.
6. Bapak, Ibu dosen serta segenap staf dan karyawan Jurusan Teknik Elektro baik secara langsung maupun tidak langsung yang telah banyak membantu dalam menyelesaikan skripsi ini.
7. Yustisia Allaintien tersayang yang telah banyak mengorbankan waktu, tenaga, pikiran dan selalu memberi semangat dan doa.

8. Wibi, Nuchan, Dion, Dimas, Anton, Nurdin dan Asfiyani serta anak-anak GS lainnya yang selalu menghiburku dengan penuh canda tawa dan memberi semangat tiada henti.
9. Mas Vikri, Mas Hadi, Mustofa, Mas Dedi dan Dimas (Lawang) yang telah banyak membantu dan mengajari saya.
10. Rekan-rekan mahasiswa Teknik Elektro Brawijaya, khususnya teman-teman paket D'05 & keluarga besar Streamline serta sahabat-sahabatku terimakasih untuk semuanya.
11. Semua pihak yang telah memberikan bantuan serta dukungan baik secara langsung maupun tidak langsung atas penyusunan skripsi ini.

Dalam penyusunan skripsi ini, penulis menyadari bahwa skripsi ini belumlah sempurna, karena keterbatasan ilmu dan kendala-kendala lain yang terjadi selama pengerjaan skripsi ini. Semoga tulisan ini dapat bermanfaat dan dapat digunakan untuk pengembangan lebih lanjut. Wassalamualaikum wr.wb.

Malang, 16 Desember 2010

Penulis

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

RINGKASAN

CANDRA KURNIAWAN, Jurusan Teknik Elektro, Fakultas Teknik Universitas Brawijaya Malang, Desember 2010. Otomatisasi Alat Pengatur Kadar Air Pada Benih Jagung Hibrida Menggunakan Mikrokontroller ATmega 8535.

Salah satu hal yang mempengaruhi kualitas benih jagung adalah kadar air, yang mana saat ini rata-rata benih jagung yang dihasilkan petani di Indonesia mempunyai kadar air 30%-35%. Sementara mutu yang sesuai SNI adalah 12%-15%.. Maka diperlukan suatu penurunan kadar air agar sesuai dengan standar tersebut. Selama ini cara yang lazim digunakan oleh sebagian besar petani adalah cara tradisional. Cara ini sangat tidak praktis dan tidak efektif karena tidak dapat dilakukan pada musim penghujan, kurang cepat, kurang merata, rawan terhadap gangguan hama seperti jamur dan hewan pengerat serta mudah terkontaminasi. Maka diperlukan sebuah sistem kendali suhu dan kadar air otomatis yang dapat membantu proses penurunann kadar air benih jagung tersebut.

Dalam perancangan alat ini digunakan mikrokontroler Atmega 8535 sebagai kontroler yang akan mengontrol plan sistem yang terdiri dari heater, kipas, motor pengaduk dan motor untuk buka tutup valve. Sensor yang digunakan untuk mendeteksi suhu ruang pengering adalah LM35. Sensor untuk mendeteksi kadar air jagung adalah sensor kadar air. Sedangkan sensor yang digunakan untuk mendeteksi berat jagung adalah strain gauge. Keluaran sensor akan diproses ADC (Analog to Digital Converter) sebelum diolah oleh mikrokontroler.

Setelah dilakukan pengujian pada keseluruhan sistem, alat mampu menurunkan kadar air jagung menjadi 12%-15% dalam waktu 40 menit.

BAB I
PENDAHULUAN

1.1 Latar Belakang

Keberhasilan pengembangan produksi benih jagung tidak hanya ditentukan oleh naiknya produksi. Sebab, pasar menghendaki pasokan benih jagung bermutu sesuai SNI dan berkesinambungan. Bila petani mampu menjaga kualitas, benih jagung yang dihasilkan memiliki keunggulan kompetitif dalam hal harga jual. Oleh sebab itu, petani dituntut untuk memahami dan melaksanakan penanganan pascapanen.

Salah satu hal yang mempengaruhi kualitas benih jagung adalah kadar air, yang mana saat ini rata-rata benih jagung yang dihasilkan petani di Indonesia mempunyai kadar air 30%-35%. Sementara mutu yang sesuai SNI adalah 12%-15%. Sehingga diperlukan suatu penurunan kadar air agar sesuai dengan standar tersebut. Penurunan kadar air benih jagung dilakukan setelah benih dipanen dan sebelum benih memasuki tahap *processing* (pembersihan, pengukuran, dan pengemasan). Ada dua cara untuk menurunkan kadar air benih jagung yaitu cara tradisional dengan proses pengeringan dapat dilakukan melalui penjemuran di bawah terik sinar matahari atau menggunakan mesin pengering. Selama ini cara yang lazim digunakan oleh sebagian besar petani adalah cara tradisional. Cara ini sangat tidak praktis dan tidak efektif karena tidak dapat dilakukan pada musim penghujan, kurang cepat, kurang merata, rawan terhadap gangguan hama seperti jamur dan hewan pengerat serta mudah terkontaminasi.

Oleh karena itu diperlukan sebuah sistem kendali suhu dan kadar air otomatis yang dapat membantu proses pengaturan kadar air benih jagung tersebut. Maka dalam penelitian ini akan dirancang suatu alat yang nantinya dapat menurunkan kadar air benih jagung secara otomatis dengan memanfaatkan mikrokontroller ATMEGA 8535 sebagai kontroller

otomatisnya. Karena ATMEGA 8535 mempunyai kecepatan proses yang relatif cepat serta mempunyai ADC internal dengan resolusi 10 bit.

1.2 Rumusan Masalah

Dalam perancangan ini perumusan ditekankan pada :

Bagaimana merancang sistem kontroller pada alat pengatur kadar air benih jagung hibrida.

1.3 Batasan Masalah

Dalam perancangan otomatisasi alat pengatur kadar air pada benih jagung hibrida permasalahan dibatasi dalam beberapa hal yaitu :

1. Alat pengatur kadar air yang dikontrol mengunakan model miniatur bukan menggunakan yang sebenarnya.
2. Benih jagung yang digunakan adalah jenis hibrida yang sudah dipipil, dengan sembarang ukuran biji, dan tidak membahas tingkat kebersihan dari benih jagung.
3. Suhu ruang pemanas diatur antara 41^{o}C - 44^{o}C
4. Kadar air jagung yang diinginkan adalah 12%-15%.
5. Berat jagung yang akan dikeringkan sebesar 1 kg – 2 kg.

1.4 Tujuan

Tujuan yang ingin dicapai dari tugas akhir ini adalah terciptanya alat otomatis untuk mengatur kadar air benih jagung sesuai standar SNI (Standar Nasional Indonesia) yaitu 12%-15%. Sehingga petani akan di untungkan karena harga jual benih akan jauh lebih tinggi di pasar.

1.5 Sistematika Pembahasan

BAB I Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi pembahasan, dan sistematika pembahasan.

BAB II Tinjauan Pustaka

Membahas teori-teori yang mendukung dalam perencanaan dan pembuatan alat.

BAB III Metodologi

Berisi tentang metode penelitian dan perencanaan alat serta pengujian.

BAB IV Perencanaan dan Pembuatan Alat

Perancangan dan perealisasian alat yang belum ada judulnya.

BAB V Pengujian Alat

Memuat hasil pengujian terhadap alat yang telah direalisasikan.

BAB VI Kesimpulan dan Saran

Memuat kesimpulan dan saran-saran.

BAB II
TINJAUAN PUSTAKA

Untuk memudahkan dalam memahami cara kerja rangkaian maupun dasar-dasar perencanaan dari alat ini, maka perlu adanya penjelasan dan uraian teori penunjang yang digunakan dalam penulisan skripsi ini. Teori penunjang yang akan dijelaskan dalam bab ini adalah:

· Sensor (Sensor Suhu LM35, Sensor Kadar Air, dan Sensor Berat).
· Pengkondisi Sinyal untuk ADC internal.
· Pemanas (heater).
· Mikrokontroller Atmega 8535.
· ADC Internal Atmega 8535.
· LCD.
· Driver.
· Motor DC.
· Alat Pengaduk.

2.1. Sensor.

Sensor adalah jenis tranduser yang digunakan untuk mengubah besaran mekanis, magnetis, panas, sinar, dan kimia menjadi tegangan dan arus listrik. Sensor merupakan salah satu komponen yang sangat penting dalam mendukung terjadinya kontrol proses yang mana berfungsi sebagai berikut:

a. Menyediakan *input* dari proses dan dari lingkungan eksternal.
b. Mengubah informasi fisik misalnya suhu, tekanan, laju aliran dan posisi menjadi sinyal listrik.
c. Terkait dengan variabel fisik pada cara yang diketahui sehingga sinyal listriknya dapat digunakan untuk memonitor dan mengontrol proses.

Sensor sering digunakan untuk pendeteksian pada saat melakukan pengukuran atau pengendalian. Beberapa jenis sensor yang banyak digunakan dalam rangkaian elektronik antara lain sensor cahaya, sensor suhu, dan sensor tekanan.

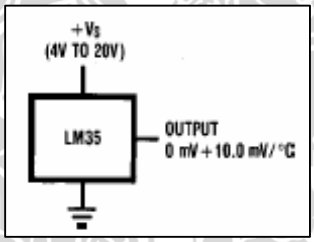
2.1.1. Sensor Suhu LM35

LM35 merupakan sensor suhu terintegrasi yang mempunyai tegangan keluaran yang linier. LM35 mempunyai impedansi keluaran yang rendah, keluaran yang linier dan kalibrasi yang tepat sehingga mudah untuk di hubungkan dengan rangkaian lain.

Adapun fitur yang ada pada LM35 adalah sebagai berikut:

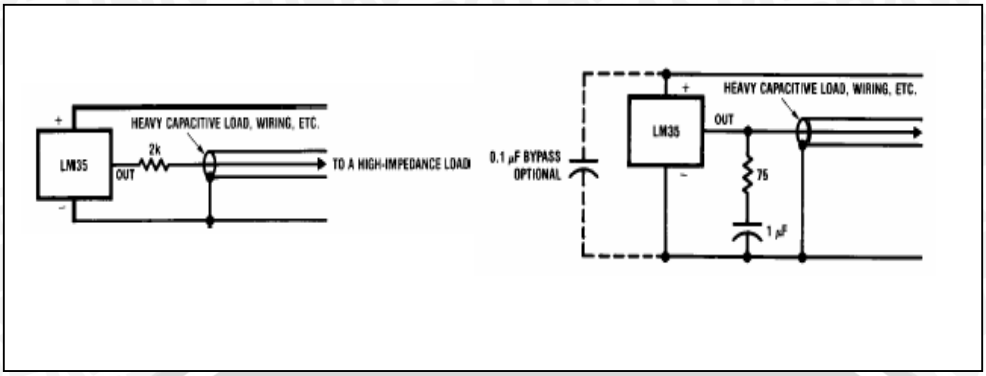
- Kalibrasi dalam derajat celcius
- Faktor skala linier adalah 10 mV/^{0}C
- Jangkauan suhu -55^{0}C sampai 150^{0}C
- Tegangan operasi dari 4 V sampai 30 V
- Ketidak linierran hanya $\pm$ ¼^{0}C
- Impedansi keluaran kecil sebesar 0,1 Ohm untuk arus beban 1mA

Dalam Gambar 2.1 ditunjukkan skema sensor suhu LM35.



Gambar 2.1 Skema LM35
Sumber: Datasheet LM35

LM35 mempunyai batas kemampuan untuk mendrive beban kapasitif. LM35 dapat mendrive 50 pF tanpa penanganan khusus. Jika beban bertambah maka untuk mengantisipasinya dengan mengisolasi atau mendecouple beban dengan resistor atau dengan menambah kapasitansi yang disusun seri dengan resistor antara output dan ground seperti ditunjukan dalam Gambar 2.2.



Gambar 2.2. LM35 dengan decouple resistor dan LM35 dengan RC damper
Sumber : Datasheet LM35

2.1.2. Sensor kadar air (sensor kapasitif).

Sensor yang digunakan adalah sensor kapasitif yang terbuat dari dua lempeng tembaga atau bisa menggunakan dua lempeng plat besi. Sensor ini bekerja dengan adanya perubahan nilai fisis kadar air yang terkandung dalam material yang diukur yang mempengaruhi variabel penentu kapasitansi alumunium. Output dari sensor ini berupa tegangan antara 0-5 volt.

2.1.3. Sensor berat (Load cell)

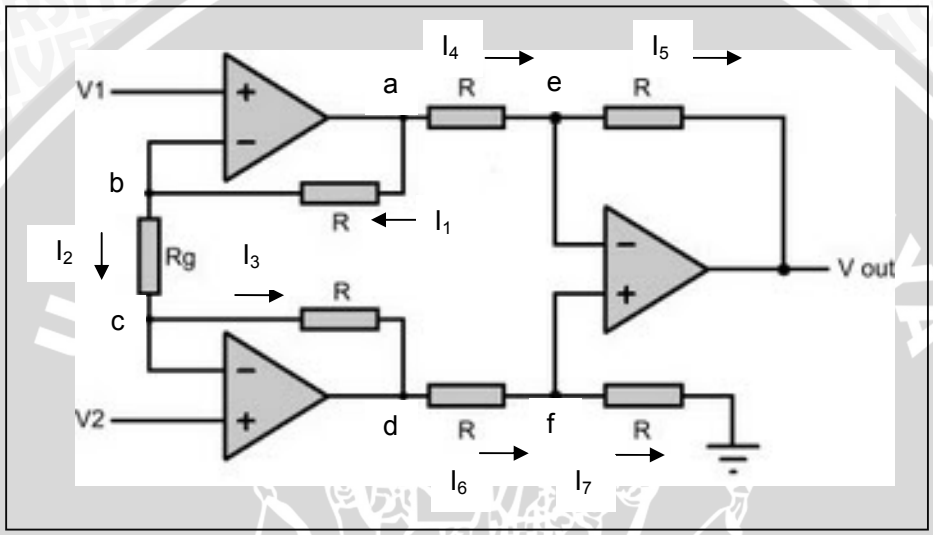
Load cell adalah tranducer yang digunakan untuk mengubah tekanan menjadi sinyal elektrik. Biasanya alat ini terdiri dari empat strain gauges dalam wheatstone brigde configuration, tetapi ada juga yang terdiri dari satu atau dua strain gauges. Sinyal output elektrik biasanya direpresentasikan dalam milivolt dan memerlukan penguatan oleh instrument amplifier sebelum dapat digunakan. Output dari tranduser dimasukan dalam algoritma untuk menghitung tekanan pada tranducer.

2.2. Pengkondisi Sinyal

Rangkaian pengkondisi sinyal mempunyai fungsi sebagai pembanding antara tegangan referensi dengan tegangan output dari sensor, yang mana diharapkan tegangan keluaran dari rangkaian sensor dapat dikondisikan sesuai dengan range yang dirancang pada ADC internal.

2.2.1. Penguat Instrumentasi

Penguat ini merupakan penguat serba guna dan bermanfaat yang terdiri atas tiga op-amp dan tujuh buah tahanan. Rangkaian ini tersusun atas rangkaian penguat differensial dan penguat penyangga. Untuk mengatur penguatan yang diinginkan diatur dengan mengubah-ubah nilai *Rg*. Rumusan dan gambar dari penguat instrumentasi adalah sebagai berikut :



Gambar 2.3. Penguat Instrumentasi

Sumber: http://franzaditya.blogspot.com/

Karena tidak ada arus yagn masuk ke dalam op-amp, maka $V_1 = V_b$ dan $V_2 = V_c$. Arus pada suatu kawat penghantar akan sama besar selama tidak ada percabangan, sehingga

$$I_1 = I_2$$

$$\frac{V_a - V_b}{R} = \frac{V_b - V_c}{R_g}$$

$$\frac{V_a - V_1}{R} = \frac{V_1 - V_2}{R_g}$$

$$\frac{V_a}{R} = \frac{V_1 - V_2}{R_g} + \frac{V_1}{R}$$

$$V_a = \frac{R}{R_g}(V_1 - V_2) + V_1$$

$$I_2 = I_3$$

$$\frac{V_b - V_c}{R_g} = \frac{V_c - V_d}{R}$$

$$\frac{V_1 - V_2}{R_g} = \frac{V_2 - V_d}{R}$$

$$\frac{V_1}{R_g} - \left(\frac{1}{R_g} + \frac{1}{R}\right)V_2 = -\frac{V_d}{R}$$

$$V_d = \left(\frac{R}{R_g} + 1\right)V_2 - \frac{R}{R_g}V_1$$

Karena tidak ada arus yang masuk ke dalam op-amp, maka

$$I_4 = I_5$$

$$\frac{V_a - V_e}{R} = \frac{V_e - V_{out}}{R}$$

$$V_a = 2V_e - V_{out}$$

$$I_6 = I_7$$

$$\frac{V_d - V_f}{R} = \frac{V_f - 0}{R}$$

$$V_d = 2V_f$$

Selisih tegangan masukan *inverting* dan tegangan masukan *non-inverting* ≈ 0, sehingga

$$V_e = V_f$$

$$V_d = 2V_e$$

$$V_a = V_d - V_{out}$$

$$V_{out} = V_d - V_a$$

$$V_{out} = \left(\left(\frac{R}{R_g}+1\right)V_2 - \frac{R}{R_g}V_1\right) - \left(\frac{R}{R_g}(V_1 - V_2) + V_1\right)$$

$$V_{out} = \frac{R}{R_g}V_2 + V_2 - \frac{R}{R_g}V_1 - \frac{R}{R_g}V_1 + \frac{R}{R_g}V_2 - V_1$$

$$V_{out} = \left(\frac{2R}{R_g}+1\right)V_2 - \left(\frac{2R}{R_g}+1\right)V_1$$

$$V_{out} = \left(\frac{2R}{R_g}+1\right)(V_2 - V_1) \quad \text{volt}$$

2.3. Pemanas (Heater)

Pemanas yang digunakan dalam perancangan mesin pengering ini menggunakan elemen pemanas yang ada pada alat pengering rambut (*hair dryer*). Udara panas yang dihasilkan oleh *hair dryer* cukup untuk memanaskan udara dalam ruang pengeringan. Prinsip kerja elemen pemanas ini memanfaatkan kawat berlapis nikrom yang dililit memutar dan dibungkus oleh mika. Kawat nikrom merupakan campuran dari dua logam nikel dan kromium.

Gambar 2.4. Pemanas (Heater)

Sumber: http://google.com/hair dryer/

Kelebihan dari kawat nikrom antara lain :

1. Kawat nikrom adalah konduktor listrik yang buruk sehingga paduan logam ini memiliki nilai resistansi yang tinggi. Sehingga dapat menyimpan udara panas yang mengalir darinya.
2. Kawat nikrom tidak beroksidasi dengan panas. Logam lainnya seperti besi mudah berkarat pada suhu panas.

2.4. Mikrokontroller ATmega 8535

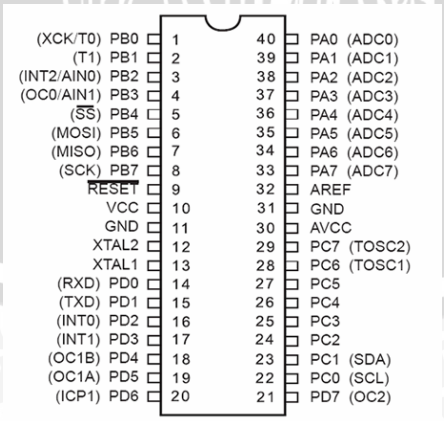
Mikrokontroller adalah suatu mikroprosesor plus. Mikrokontroller adalah pusat kerja dari suatu sistem elektronika seperti halnya mikroprosesor sebagai otak komputer. Adapun nilai plus bagi mikrokontroller adalah terdapatnya memori dan port input/output dalam suatu kemasan IC yang kompak. Kemampuannya yang programmable,fitur yang lengkap seperti ADC internal, EEPROM internal, port I/O, komunikasi serial. Juga harga yang terjangkau memungkinkan mikrokontroller digunakan pada berbagai sistem elektronis,seperti pada robot, automasi industri, sistem alarm, peralatan telekomunikasi, hingga sistem keamanan. Mikrokontroler AVR memiliki arsitektur RISC 8 bit, dimana semua instruksi dikemas dalam kode 16 bit dan sebagian besar instruksi dalam 1 (satu) siklus clock, berbeda dengan instruksi MCS51 yang membutuhkan 12 siklus clock. Hal ini terjadi karena kedua jenis mikrokontroler tersebut memiliki arsitektur yang berbeda. AVR berteknologi RISC (Reduced Instruction Set Computing), sedangkan seri MCS51 berteknologi CISC (Complex Instruction Set Computing). Secara umum, AVR dapat dikelompokkan menjadi 4 kelas, yaitu keluarga ATtiny, keluarga AT90Sxx, keluarga ATMega, dan AT86RFxx. Pada dasarnya, yang membeda-bedakan masing-masing kelas adalah memori, peripheral, dan fungsinya. Dari segi arsitektur dan instruksi yang digunakan, mereka bisa dikatakan sama.Piranti dapat diprogram secara *in-system programming* (ISP) dan dapat diprogram berulang-ulang selama 10.000 kali baca/tulis didalam sistem.

2.4.1 Konfigurasi Pin ATMega8535

Secara fungsional konfgurasi ATMega8535 sebagai berikut;

1. VCC merupakan pin yang berfungsi sebagai pin masukan catu daya.
2. GND merupakan pin *Ground.*
3. Port A (PA0…PA7) merupakan pin I/O dua arah dan pin masukan catu ADC.
4. Port B (PB0…PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu Timer/Counter, Komparator analog, dan SPI.
5. Port C (PC0…PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, Komparator analog, dan *Timer Oscillator*
6. Port D (PD0…PD7) merupakan pin I/O dua arah dan pin fungsi khusus,yaitu komparator analog, Interupsi eksternal, dan komunikasi serial.
7. RESET merupakan pin yang digunakan untuk me-reset mikrokontroller.
8. XTAL1 danXTAL2 merupakan pin masukan *clock* eksternal.
9. AVCC merupakan pin masukan tegangan untuk ADC.
10. AREF merupakan pin masukan tegangan referensi ADC.

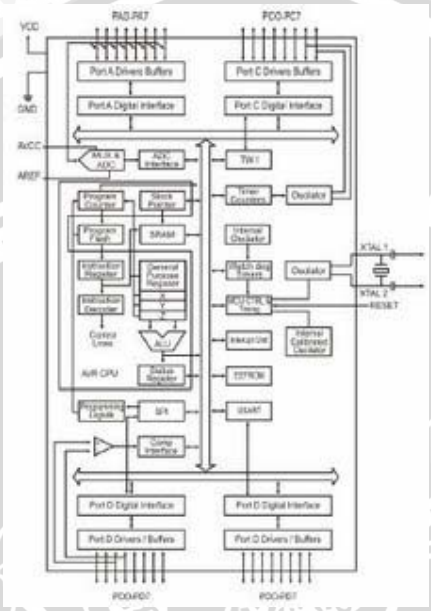
Gambar berikut menunjukkkan konfigurasi pin ATmega 8535



Gambar 2.5. Konfigurasi pin ATMEGA 8535

Sumber: *Datasheet* ATmega 8535.

2.4.2. Arsitektur ATMega8535



Gambar 2.6. Diagram Blok ATMEGA 8535

Sumber: *Datasheet* ATmega 8535.

Dari gambar tersebut dapat dilihat bahwa ATMega8535 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 8 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.

2.4.3. Fitur ATMega8535

Kapabilitas detail dari ATMega8535 adalah sebagai berikut:

1. System mikroprosesor 8 bit berbasis RISC dengan kecepatan maksimal 16 Mhz.
2. Kapabilitas memory flash 8KB,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.

2.5. ADC Internal ATMEGA 8535.

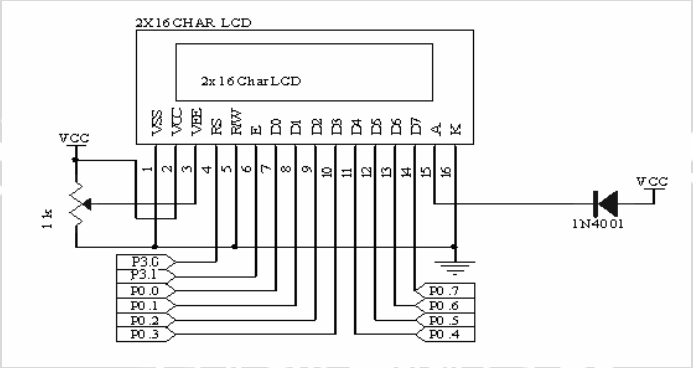
ATmega8535 menyediakan fasilitas ADC dengan resolusi 10 bit. ADC ini dihubungkan dengan 8 channel Analog Multiplexer yang memungkinkan terbentuk 8 input tegangan single- ended yang masuk melalui pin pada PortA. ADC memiliki pin supply tegangan analog yang terpisah yaitu AVCC. Besarnya tegangan AVCC adalah ±0.3V dari VCC.

Tegangan referensi ADC dapat dipilih menggunakan tegangan referensi internal maupun eksternal. Jika menggunakan tegangan referensi internal, bisa dipilih on-chip internal reference voltage yaitu sebesar 2.56V atau sebesar AVCC. Jika menggunakan tegangan referensi eksternal, dapat dihubungkan melalui pin AREF.

ADC mengkonversi tegangan input analog menjadi data digital 8 bit atau 10 bit. Data digital tersebut akan disimpan didalam ADC Data Register yaitu ADCH dan ADCL. Sekali ADCL dibaca, maka akses ke data register tidak bisa dilakukan. Dan ketika ADCH dibaca, maka akses ke data register kembali enable.

2.6. LCD

Dalam perancangan ini menggunakan LCD tipe M1632 (16 kolom x 2 baris). Bus data LCD (D0-D7) terhubung dengan port 0 mikrikontroler (P0.0-P0.7).LCD ini di operasikan hanya menerima data maka pin R/W dihubungkan dengan ground. Sedangkan pin 3.2 dihubungkan dengan RS. Untuk mengaktifkan E (enable) LCD digunakan keluaran pin 3.1. Tingkat kecerahan LCD diatur dengan resistor variabel10 kΩ. Gambar rangkaian dapat dilihat dibawah ini.



Gambar 2.7. Rangkaian LCD

Sumber : Perancangan

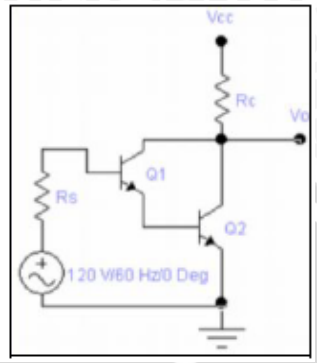
2.7. Driver Motor DC

Driver atau rangkaian elektronik untuk penggerak motor DC menggunakan prinsip rangkaian Darlington. Darlington adalah rangkaian elektronika yang terdiri dari sepasang transistor bipolar (dwi kutub) yang tersambung secara tandem (seri). Sambungan seri seperti ini dipakai untuk mendapatkan penguatan (*gain*) yang tinggi, karena hasil penguatan pada transistor yang pertama akan dikuatkan lebih lanjut oleh transistor kedua. Keuntungan dari rangkaian Darlington adalah penggunaan ruang yang lebih kecil dari pada rangkaian dua buah transistor biasa dengan bentuk konfigurasi yang sama. Penguatan arus listrik atau gain dari rangkaian transistor Darlington ini sering dituliskan dengan notasi β atau *hfe* .

Secara ideal besarnya penguatan adalah :

$$hfe_D = \beta_1 \beta_2$$

Hubungan Darlington dapat dilihat dalam Gambar 2.8.



Gambar 2.8 Hubungan Darlington

Sumber: Modul Praktikum FI – 2104 Elektronika Dasar. 2005. *Transistor Sebagai Saklar Penguat Gandengan DC Darlington dan Penguat Diferensial*. Laboratorium Elektronika dan Instrumentasi Fisika ITB. Bandung

$$K_v = \frac{v_0}{v_i} = \frac{R_0 i_0}{R_1 i_1} = K_i \frac{R_0}{R_i}$$

Dengan penguatan arus

$$K_i = \beta_i \beta_2$$

$$R_0 = R_c // \frac{1}{h_{0e2}} \cong R_c$$

$$R_i = h_{ie1} + (1 + \beta_i) h_{ie2}$$

Oleh karena *hie* berada pada emitor Q1, maka jika dilihat dari basis Q1 tampak mempunyai nilai (1+βi) kalinya. Sehingga:

$$K_{vi} = \frac{v_0}{v_i} = \frac{\beta_1 \beta_2 R_c}{h_{ie1} + (1 + \beta_i) h_{ie2}}$$

2.8. Motor DC

Motor DC adalah motor yang memerlukan suplai tegangan searah pada kumparan jangkar dan kumparan medan untuk diubah menjadi energi mekanik. Berdasarkan karakteristiknya, motor arus searah ini mempunyai daerah

pengaturan putaran yang luas dibandingkan dengan motor arus bolak-balik, sehingga sampai sekarang masih banyak digunakan pada pabrik-pabrik yang mesin produksinya memerlukan pengaturan putaran yang luas.

Prinsip Kerja Motor DC:

Pada motor DC, kumparan medan yang dialiri arus listrik akan menghasilkan medan magnet yang melingkupi kumparan jangkar dengan arah tertentu. Konverter energi baik energi listrik menjadi energi mekanik (motor) maupun sebaliknya dari energi mekanik menjadi energi listrik (generator) berlangsung melalui mediummedan magnet. Energi yang akan diubah dari suatu sistem ke sistem yang lain, sementara akan tersimpan pad medium medan magnet untuk kemudian dilepaskan menjadi energi system lainya. Dengan demikian, medan magnet disini selain berfungsi sebagi tempat penyimpanan energi juga sekaligus proses perubahan energi

2.9. Alat pengaduk

Alat pengaduk yang digunakan berupa sebatang besi berbentuk huruf " T " terbalik yang dibagian atasnya di putar menggunakan sebuah motor DC.

BAB III
METODE PENELITIAN

Untuk menyelesaikan rumusan masalah dan merealisasikan tujuan penelitian yang terdapat di bab pendahuluan maka diperlukan metode untuk menyelesaikan masalah tersebut. Metode yang digunakan dapat diuraikan sebagai berikut :

· Studi Literatur
· Penentuan spesifikasi alat
· Perealisasian alat
· Pengujian alat
· Pengambilan kesimpulan

3.1 Studi Literatur

Literatur yang dibutuhkan adalah dasar teori yang berhubungan dengan alat yang akan dirancang, yaitu sebagai berikut:

· Sensor (Sensor Suhu LM35, Sensor Kadar Air, dan Sensor Berat).
· Pengkondisi Sinyal untuk ADC internal.
· Pemanas (heater).
· Mikrokontroller Atmega 8535.
· ADC Internal Atmega 8535.
· LCD.
· Driver.
· Motor DC.
· Alat pengaduk.

3.2 Penentuan Spesifikasi Alat

Sebelum melakukan perencanaan dan perealisasian alat, maka ditentukan spesifikasi alat yang akan dibuat. Adapun spesifikasi alat yang akan direalisasikan sebagai berikut:

· Sensor Suhu LM35 untuk membaca suhu ruanagan pengering.
· Pengkondisi Sinyal digunakan untuk menguatkan sinyal keluaran dari sensor-sensor yang digunakan

· Mikrokontroler AVR Tipe ATmega 8535 berfungsi sebagai pengontrol utama sistem dan telah dilengkapi dengan ADC internan 10 bit
· LCD tipe M1632 (16 kolom x 2 baris) digunakan untuk menampilkan suhu ruangan, berat jagung, dan kadar air jagung.
· Keypad berfungsi untuk memberi masukan data melalui tombol-tombol yang terdapat pada papan keypad tersebut
· Motor DC gearbox akan digunakan untuk memutar alat pengaduk.

3.3 Perealisasian Alat

3.3.1 Perancangan Perangkat Keras dan Realisasi Tiap Blok

a. Pembuatan blok diagram lengkap sistem
b. Penentuan dan perhitungan komponen yang akan digunakan
c. Merakit perangkat keras masing-masing blok

3.3.2 Perancangan dan Penyusunan Perangkat Lunak

Setelah kita mengetahui seperti apa perangkat keras yang kita rancang, maka kita membutuhkan perangkat lunak untuk mengendalikan dan mengatur kerja dari alat ini. Parameter yang diperoleh dari hasil perhitungan kemudian diterapkan kedalam mikrokontroler ATmega 8535 dengan menggunakan bahasa Assembly.

3.4 Pengujian Alat

Untuk memastikan bahwa sistem ini berjalan sesuai yang direncanakan maka diperlukan dilakukan pengujian alat meliputi perangkat keras *(hardware)* yang dilakukan baik per blok rangkaian maupun keseluruhan sistem.

3.4.1 Pengujian Perangkat Keras

Pengujian perangkat keras dilakukan dengan tujuan untuk menyesuaikan nilai tegangan dan arus yang diijinkan bekerja dalam komponen berdasarkan data sekunder komponen yang diambil dari buku data komponen elektronika maupun dari datasheet.

3.4.2 Pengujian Keseluruhan Sistem

Pengujian sistem secara keseluruhan dilakukan dengan tujuan untuk mengetahui kerja alat setelah perangkat keras dan perangkat lunak diintegrasikan bersama.

3.5 Pengambilan Kesimpulan

Pengambilan kesimpulan dilakukan setelah didapatkan hasil dari pengujian. Jika hasil yang diperoleh telah sesuai dengan spesifikasi yang direncanakan maka alat tersebut telah memenuhi harapan dan memerlukan pengembangan untuk penyempurnaannya.

BAB IV
PERANCANGAN DAN PEMBUATAN ALAT

Perancangan alat ini dilakukan secara bertahap yakni blok demi blok sehingga akan memudahkan dalam penganalisaan disetiap bloknya maupun secara keseluruhan.

Perancangan ini terdiri dari:

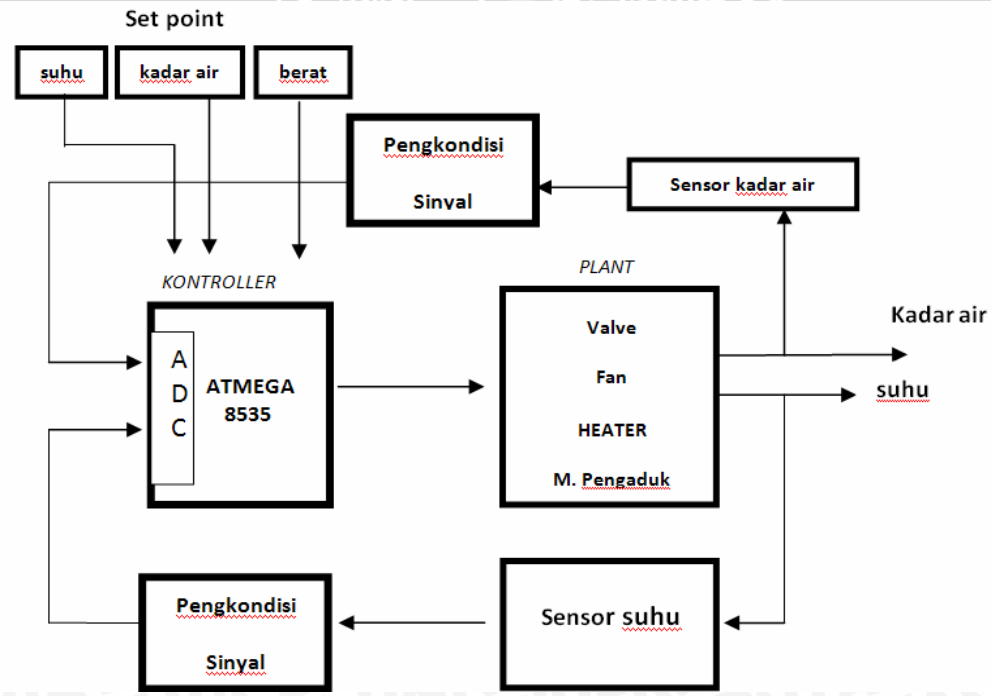
Ø Cara kerja alat

Ø Perancangan perangkat keras (perancangan mekanik alat, sensor kadar air, pengkondisi sinyal pada ADC internal, alat pamanas, driver motor gear box, rangkaian mikrokontroler Atmega 8535.

Ø Perancangan perangkat lunak (perancangan algoritma perangkat lunak).

4.1. Cara Kerja Alat

Blok diagram alat ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Blok Diagram Alat.

Sumber: Perancangan.

Secara garis besar prinsip kerja alat adalah sebagai berikut :

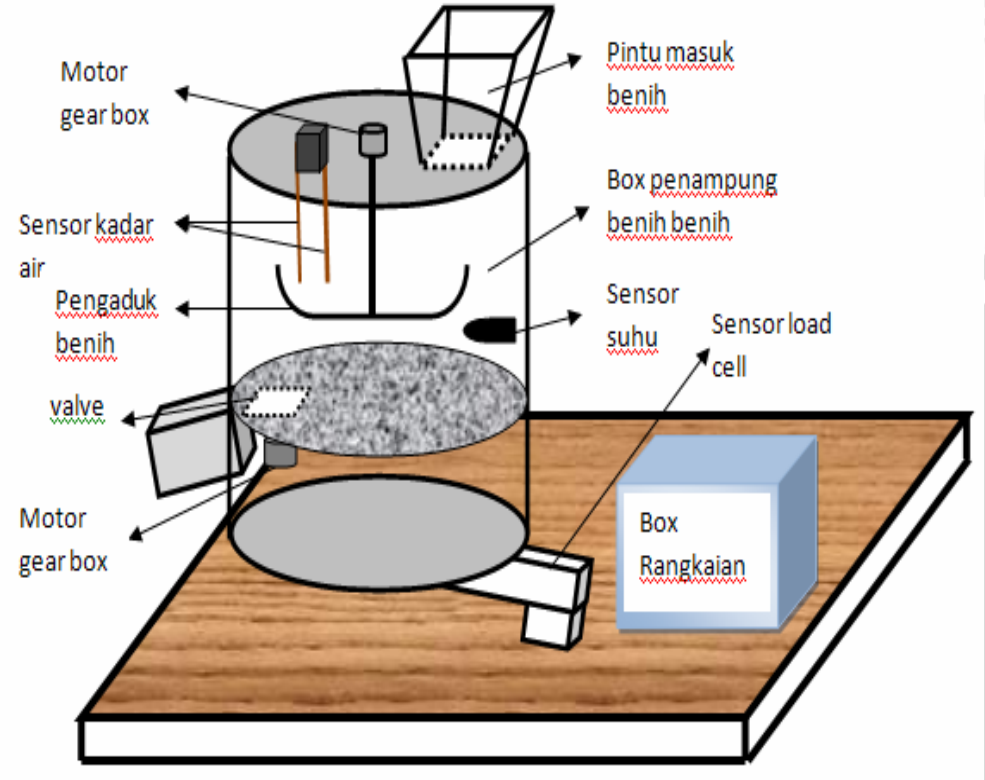
Ø Sensor suhu yang diletakkan didalam ruang pengering akan membaca kondisi ruang pengering dan akan memberikan sinyal sebagai masukan pada mikrokontroler.

Ø Suhu dalam ruang pengering dikondisikan untuk mencapai set point pada 41-44^{o}C dengan memanfaatkan heater sebagai aktuatornya. Heater dikondisikan untuk mengeluarkan udara panas. Suhu awal ruang pengeringan mengikuti suhu ruang normal berkisar pada 26 ^{o}C. Perubahan nilai suhu akan tertampil pada LCD sampai mencapai set point yang diinginkan. Dan apabila suhu telah lebih dari set point maka mikrokontroler akan menginstruksikan *heater* untuk off.

Ø Kipas dalam ruang pengering akan meniupkan udara panas dari heater bersamaan dengan motor pengaduk benih jagung yang sama2 bertujuan agar udara panas tersebar merata dalam ruang penampung benih. Sensor berat akan melakukan penimbangan setiap waktu dan perubahan berat jagung akan di tampilkan pada LCD. Sensor kadar air dikondisikan dengan set point 14%. Saat kadar air jagung benih jagung telah mencapai 14% maka mikrokontroller akan mengaktifkan valve untuk terbuka dan benih jagung akan keluar secara otomatis dari ruang penampung.

4.2 Perancangan Mekanik Alat

Perancangan mekanik alat ditunjukkan dalam Gambar 4.2.



Gambar 4.2 Perancangan mekanik alat.
Sumber: Perancangan.

4.3 Perancangan Perangkat Keras

4.3.1. Sensor Suhu LM35

LM35 merupakan sensor suhu terintegrasi yang mempunyai tegangan keluaran yang linier. LM35 mempunyai impedansi keluaran yang rendah, keluaran yang linier dan kalibrasi yang tepat sehingga mudah untuk di hubungkan dengan rangkaian lain.

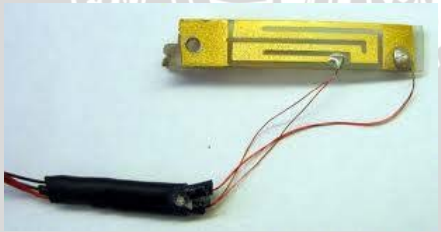
Gambar 4.3 Sensor suhu LM35
Sumber: Foto.

4.3.2. Sensor kadar air (sensor kapasitif).

Sensor yang digunakan adalah sensor kapasitif yang terbuat dari dua lempeng alumunium atau bisa menggunakan dua lempeng plat tembaga. Sensor ini bekerja dengan adanya perubahan nilai fisis kadar air yang terkandung dalam material yang diukur yang mempengaruhi variabel penentu kapasitansi alumunium. Output dari sensor ini berupa tegangan antara 0-5 volt.

4.3.3. Sensor berat (Load cell)

Load cell adalah tranducer yang digunakan untuk mengubah tekanan menjadi sinyal elektrik. Biasanya alat ini terdiri dari empat strain gauges dalam wheatstone brigde configuration, tetapi ada juga yang terdiri dari satu atau dua strain gauges. Sinyal output elektrik biasanya direpresentasikan dalam milivolt dan memerlukan penguatan oleh instrument amplifier sebelum dapat digunakan. Output dari tranduser dimasukan dalam algoritma untuk menghitung tekanan pada tranducer.



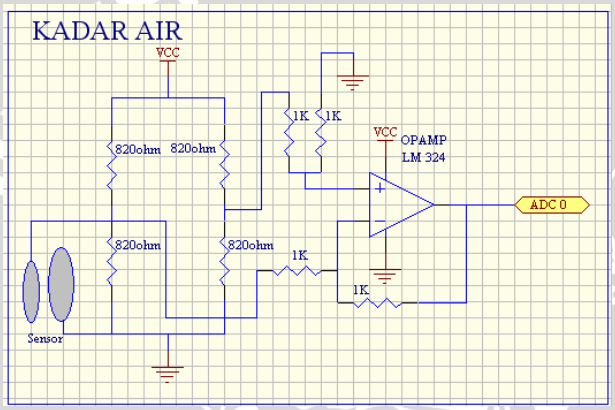
Gambar 4.4. Sensor berat
Sumber: Foto.

4.3.4. Pengkondisi Sinyal

Rangkaian pengkondisi sinyal mempunyai fungsi sebagai pembanding antara tegangan referensi dengan tegangan output dari sensor, yang mana diharapkan tegangan keluaran dari rangkaian sensor dapat dikondisikan sesuai dengan range yang dirancang pada ADC internal.

4.3.4.1. Rangkaian Pengkondisi Sinyal pada Sensor Kadar Air

. Rangkaian pengkondisi sinyal pada sensor kadar air ditunjukkan dalam gambar 4.5.

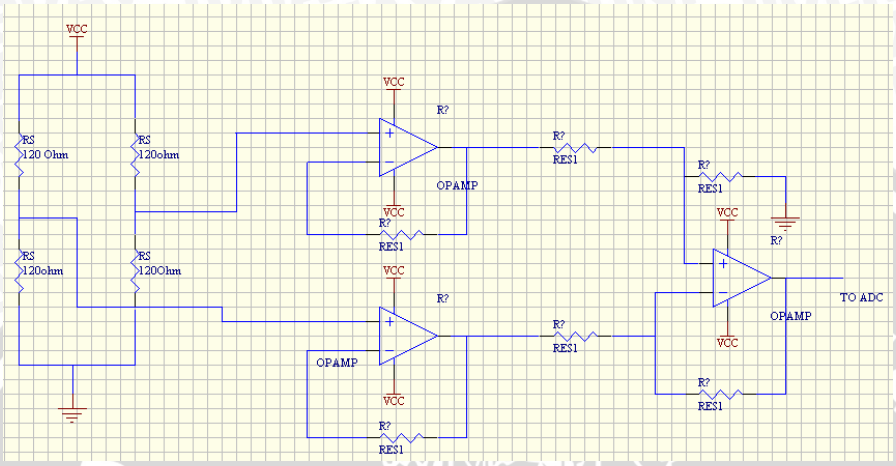


Gambar 4.5. Rangkaian pengkondisi sinyal pada sensor kadar air

Sumber: Perancangan

4.3.4.2. Rangkaian Pengkondisi Sinyal pada Sensor Berat

Rangkaian pengkondisi sinyal pada sensor berat ditunjukkan dalam gambar 4.6.



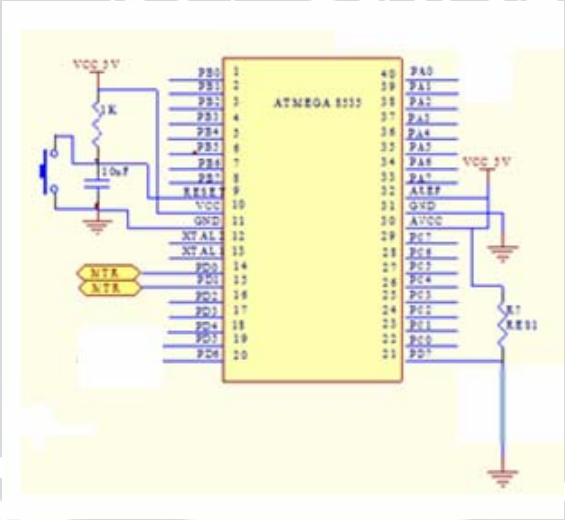
Gambar 4.6. Rangkaian pengkondisi sinyal pada sensor berat

Sumber: Perancangan

4.3.5 Rangkaian Mikrokontroler ATmega 8535

Pada alat ini digunakan mikrokontroler sebagai pusat dari pengolah data.

Minimum sistem dari mikrokontroler AT8535 ditunjukkan dalam Gambar 4.7



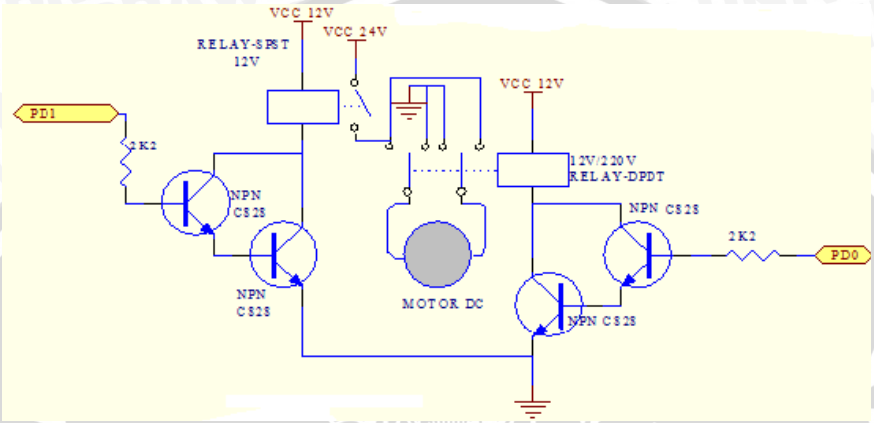
Gambar 4.7. Minimum Sistem Mikrokontroler Atmega 8535.

Sumber: Perancangan.

Mikrokontroler AT8535 mempunyai 4 *port*, 32 jalur yang dapat diprogram menjadi masukan atau keluaran.

4.3.6 Rangkaian *Driver* Motor *Gearbox dan* Motor *Gearbox*

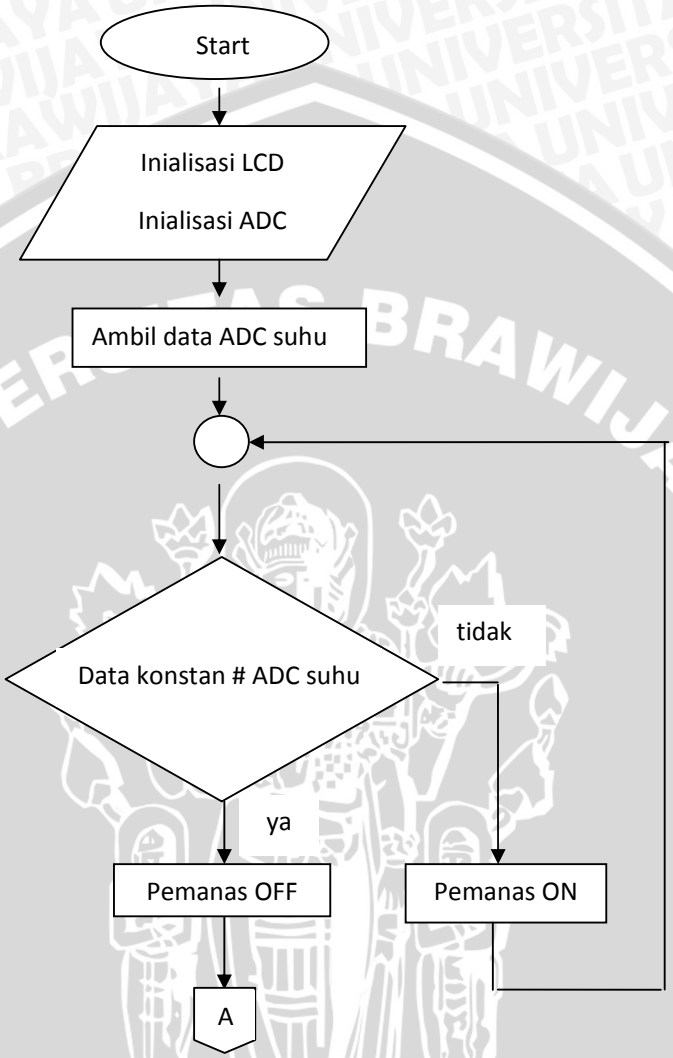
Rangkaian *driver* motor gearbox dan motor *gearbox* ditunjukkan dalam Gambar 4.13.

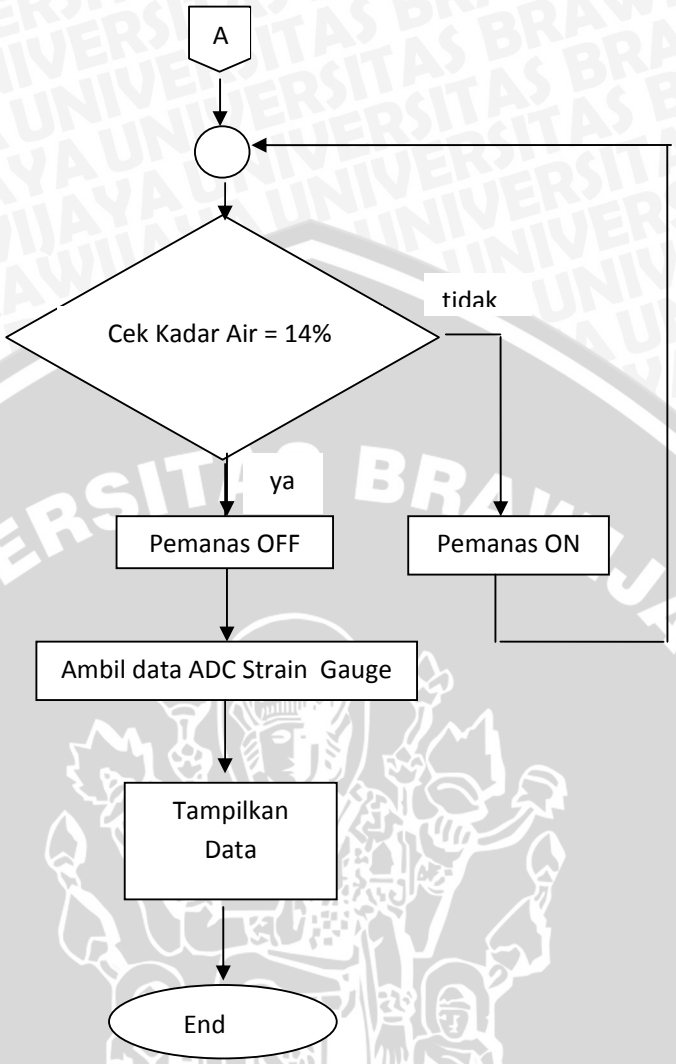


Gambar 4.8. Rangkaian *Driver* Motor *gearbox* dan Motor *gearbox*.
Sumber: Perancangan

4.4 Perancangan Perangkat Lunak

Diagram alir program ditunjukkan dalam Gambar 4.9.





Gambar 4.9. Diagram alir program.

Sumber: Perancangan

BAB V
PENGUJIAN ALAT

Pengujian alat ini bertujuan untuk menentukan apakah alat yang telah dibuat berfungsi dengan baik dan sesuai dengan perencanaan. Pengujian ini meliputi pengujian setiap blok maupun pengujian secara keseluruhan. Pengujian setiap blok ini dilakukan untuk mempermudah analisis apabila alat tidak bekerja sesuai dengan perencanaan.

5.1 Pengujian Perangkat Keras

5.1.1 Pengujian Minimum Sistem Mikrokontroler

a. Tujuan

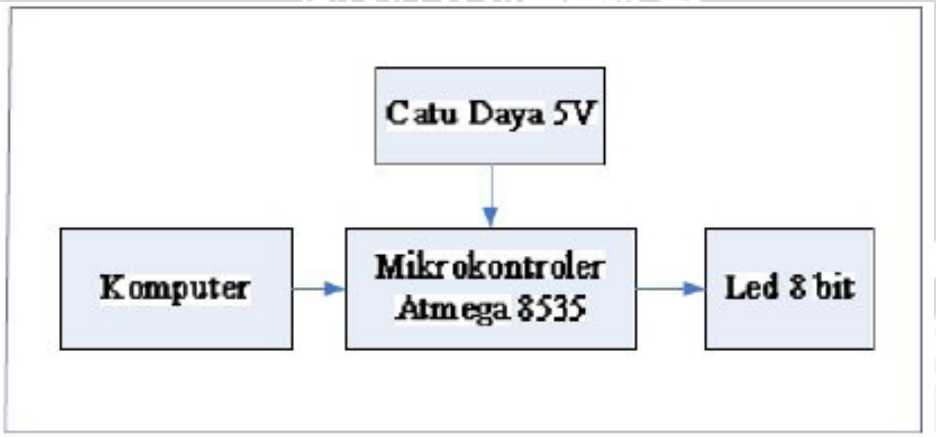
Mengetahui kondisi awal dari sistem mikrokontroler agar sesuai dengan yang diharapkan.

b. Peralatan

- Komputer dengan kabel penghubungnya
- Minimum sistem mikrokontroler ATmega 8535
- Lampu led mewakili keluaran 8 bit
- Catu daya 5 Volt

c. Langkah Pengujian

- Merangkai peralatan seperti Gambar 5.1. Keluaran terhubung dengan led yang mewakili keluaran 8 bit.



Gambar 5.1 Blok Diagram Pengujian Mikrokontroler.
Sumber: Pengujian.

- Mengisi mikrokontroler dengan program sederhana yaitu dengan mengeluarkan 0FH dan F0H pada *Port* A, kemudian *download* pada mikrokontroler AT8535.
- Mengaktifkan catu daya.
- Mencatat data keluaran yang di wakili oleh lampu led 8 bit ke dalam bentuk biner.

d. Hasil Pengujian dan Analisis

Hasil pengujian sistem mikrokontroler ditunjukkan dalam Tabel 5.1.

Tabel 5.1. 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
I	1	1	1	1	0	0	0	0
II	0	0	0	0	1	1	1	1

Sumber: Pengujian.

Dari tabel 5.1 terlihat bahwa *Port* A memberikan logika 0FH dan F0H secara bergantian sesuai dengan isi program. Dengan demikian rangkaian minimum sistem mikrokontroler ATmega 8535 sudah bisa bekerja dengan baik.

5.1.2 Pengujian Sensor Suhu LM35

a. Tujuan

Mengetahui apakah sensor berfungsi dengan baik atau tidak.

b. Peralatan

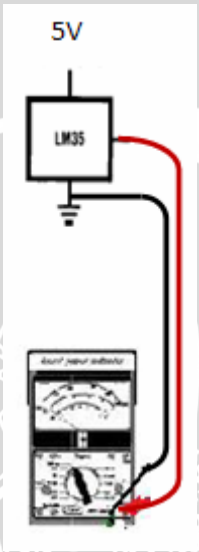
- Sensor suhu LM35
- Avometer
- Termometer (tipe raksa, merk medicca)
- Hair dryer
- Catu daya 220 volt dan 5 volt.

c. Langkah Pengujian

- Mengaktifkan catu daya 5 volt.
- Siapkan avometer dengan skala 10 volt.
- Menghubungkan avometer dengan sensor LM35 seperti pada gambar 5.2

- Ukur suhu ruangan dengan termometer dan lihat tegangan yang terukur pada avometer
- panasi sensor LM35 dengan hair dryer kemudian lihat berapa tegangan yang terukur pada avometer kemudian ulangi beberapa kali dengan suhu yang berbeda beda

Pengujian sensor suhu LM35 diunjukkan dalam gambar 5.2



Gambar 5.2. Pengujian Sensor LM35

Sumber: Pengujian.

d. Hasil Pengujian

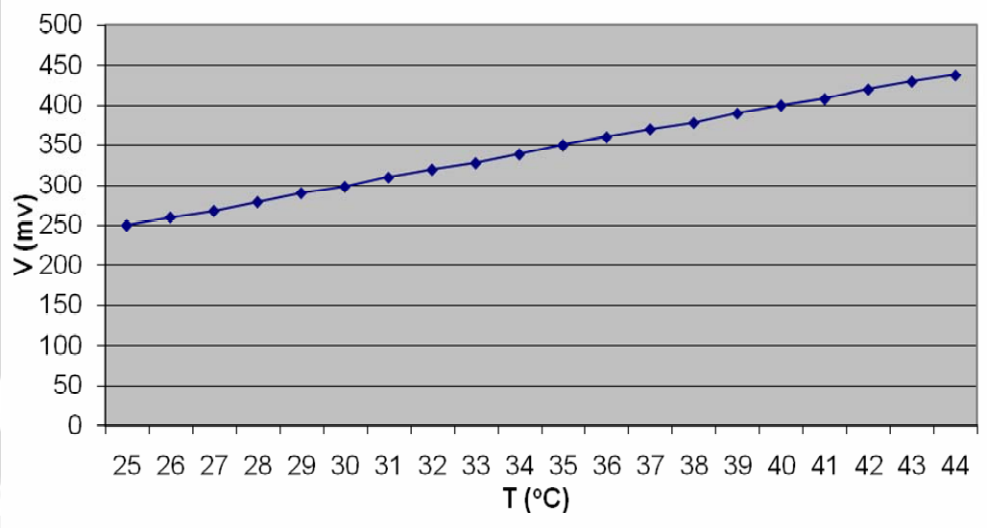
Tabel 5.2. Hasil Pengujian Sensor Suhu LM35

Suhu (^{o}C)	Tegangan yang terukur (mv)
25	250
26	260
27	268
28	279
29	290
30	298
31	310
32	320
33	328
34	339

35	350
36	360
37	370
38	378
39	390
40	400
41	408
42	420
43	430
44	438

Sumber: Pengujian.

Grafik hasil pengujian sensor suhu LM35 dapat dilihat dalam gambar 5.3 berikut



Gambar 5.3. Grafik Hasil Pengujian Sensor LM35

Sumber: Pengujian.

Hasil pengujian sensor suhu dapat dilihat dalam gambar 5.4 berikut

Gambar 5.4. Pengujian sensor suhu LM35 pada suhu 29^{o}C

Sumber: Pengujian.

e. Perbandingan Hasil Pengujian Sensor Suhu Pada Tampilan LCD dengan Thermometer

Tabel 5.3. Perbandingan Hasil Pengujian Sensor Suhu Pada Tampilan LCD dengan Thermometer

Suhu Pada LCD (^{o}C)	Suhu Pada Thermometer (^{o}C)	Error (%)
25	25	0
26	27	3,85
27	27	0
28	28	0
29	30	3,45
30	31	3,33
31	31	0
32	32	0
33	34	3,03
34	35	2,94
35	35	0
36	36	0
37	37	0
38	38	0
39	39	0

40	41	2,5
41	41	0
42	43	2,38
43	43	0
44	44	0

Sumber: Pengujian.

Dari tabel 5.3 akan didapatkan besarnya persentase error rata-rata pembacaan sensor suhu pada LCD dengan thermometer sebesar:

$$\% \text{ Error rata-rata} = \frac{3,85+3,45+3,33+3,03+2,94+2,5+2,38}{20}$$

$$= 1,07\ \%$$

5.1.2 Pengujian Sensor Kadar air

a . Tujuan

Mengetahui apakah sensor kadar air berfungsi dengan baik atau tidak

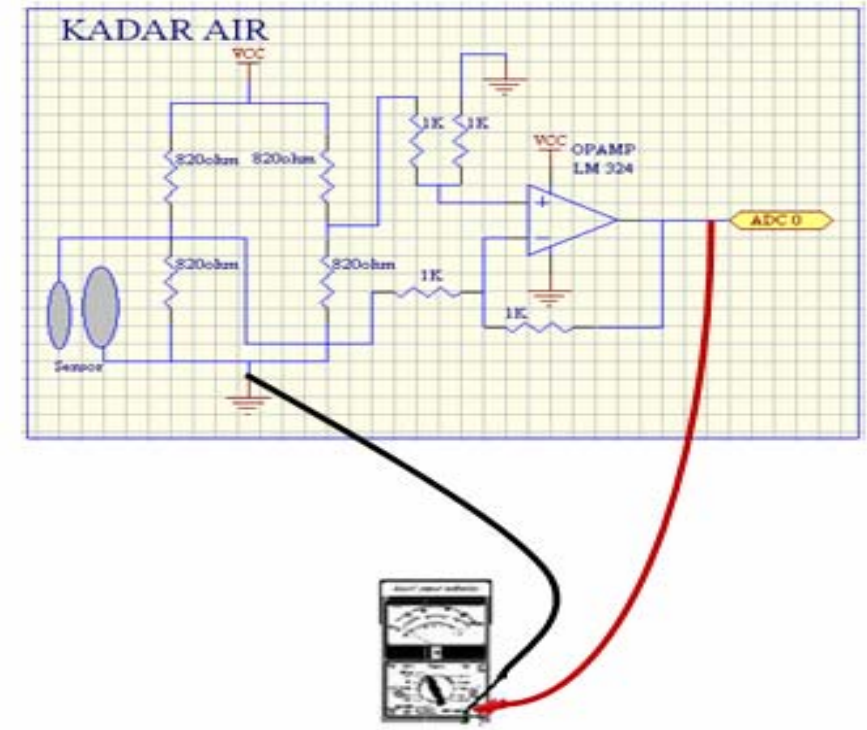
b. Peralatan

- Sensor kadar air
- Avometer
- Alat ukur kadar air (tipe MC7825G)
- 4 gelas jagung dengan tingkat kekeringan yang berbeda-beda
- catu daya 5 volt

c. Langkah Pengujian

- Mengaktifkan catu daya 5 volt
- Menghubungkan avometer pada sensor sesuai dengan gambar 5.3.
- Tancapkan ujung alat ukur kadar air pada gelas yang berisi jagung dan lihat kadar air nya
- Tancapkan ujung sensor pada gelas yang berisi jagung dan lihat tegangan yang terukur pada avometer
- Ulangi pengujian pada gelas jagung yang lain dan lihat tegangan yang terukur pada avometer

Pengujian sensor kadar air ditunjukkan dalam gambar 5.5 berikut



Gambar 5.5. Pengujian sensor kadar air

Sumber: Pengujian.

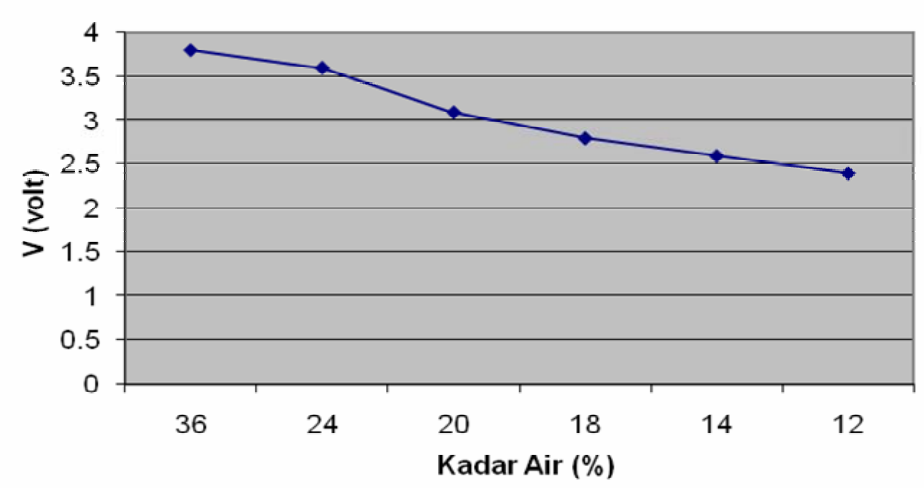
d. Hasil Pengujian

Tabel 5.4. Hasil pengujian sensor kadar air

Kadar air jagung (%)	Tegangan (volt)
36	3,8
24	3,6
20	3,1
18	2,8
14	2,6
12	2,4

Sumber: Pengujian

Grafik hasil pengujian sensor kadar air dapat dilihat dalam gambar 5.6 berikut



Gambar 5.6. Grafik Hasil Pengujian Sensor Kadar Air

Sumber: Pengujian.

Hasil pengujian sensor kadar air dapat dilihat dalam gambar 5.7 berikut

Gambar 5.7. Pengujian Sensor Kadar Air Dengan Kadar Air Jagung 14%

Sumber: Pengujian.

e. Perbandingan Hasil Pengujian Sensor Kadar Air Pada Tampilan LCD dengan Alat Ukur Kadar Air.

Tabel 5.5. Perbandingan Hasil Pengujian Sensor Kadar Air Pada Tampilan LCD dengan Alat Ukur Kadar Air.

Kadar Air Pada LCD (%)	Kadar Air Pada Alat Ukur (^{o}C)	Error (%)
78	77	1,28
77	77	0
76	76	0
75	74	1,33
74	73	1,35
71	71	0
70	70	0
66	64	3,03
61	59	3,06
58	57	1,72
54	54	0
50	48	4
46	46	0
43	43	0
38	37	2,6
32	32	0
27	26	3,7
25	25	0
20	20	0
15	14	6,6
12	12	0

Sumber: Pengujian.

Dari tabel 5.5 akan didapatkan besarnya persentase error rata-rata pembacaan sensor kadar air pada LCD dengan alat ukur sebesar:

% Error rata-rata = $\frac{1,28+1,33+1,35+3,03+3,06+1,72+2,6+3,7+6,66}{21}$

= 1,17 %

5.1.3 Pengujian Sensor Berat

a. Tujuan

Untuk mengetahui apakah sensor berat bekerja dengan baik atau tidak

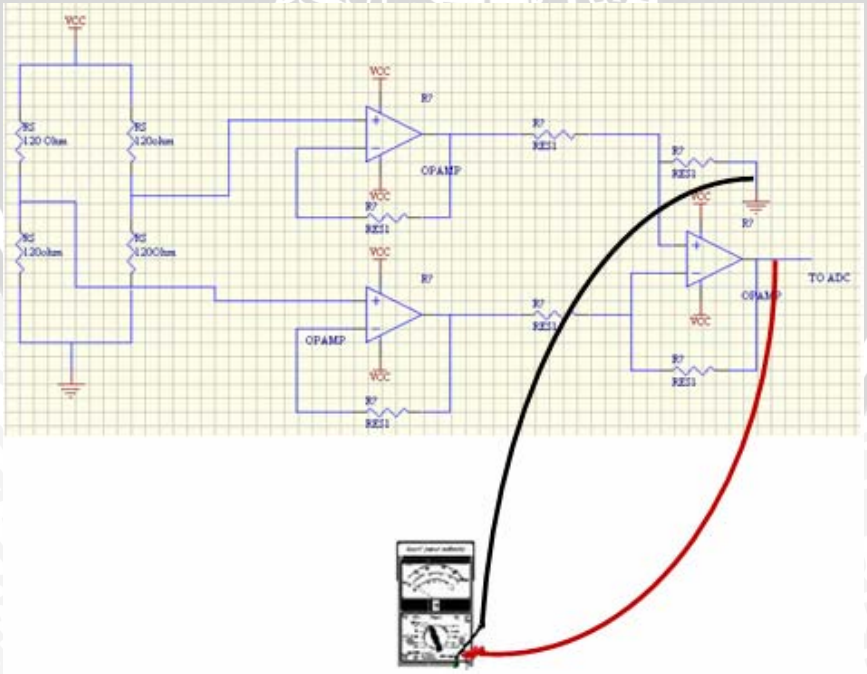
b. Peralatan

- avometer
- sensor berat
- beban pemberat 40 gram 8 buah
- catu daya 5 volt
- Timbangan analog (merk TANITA)

c. Langkah Pengujian

- Mengaktifkan catu daya 5 volt
- Menghubungkan avometer pada sensor sesuai dengan gambar 5.8.
- Pasang beban diatas sensor berat
- Lihat tegangan yang terukur pada avometer
- Ulangi pengukuran dengan beban yang berbeda dan ulangi pembacaan tegangan pada avometer

Pengujian sensor berat ditunjukkan dalam gambar 5.8



Gambar 5.8. Pengujian Sensor Berat

Sumber: Pengujian.

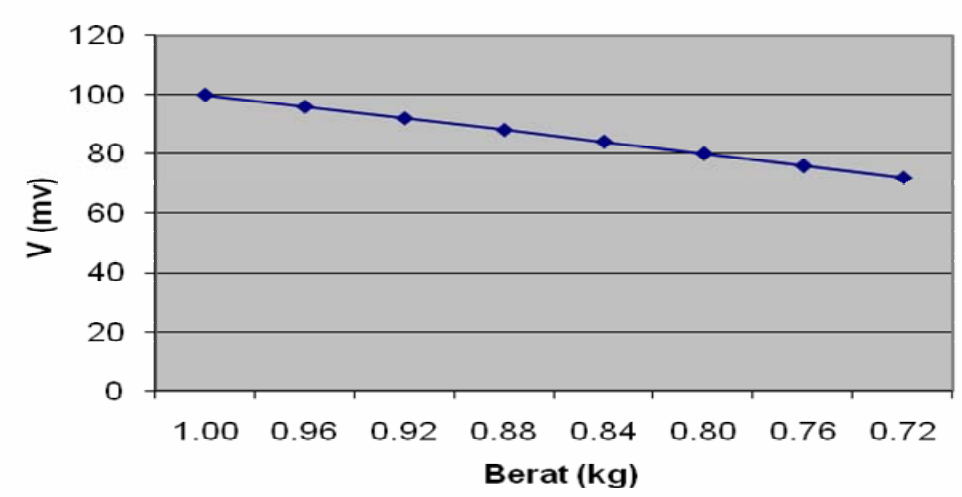
d. Hasil Pengujian

Tabel 5.6. Hasil pengujian sensor berat

Beban yang diukur (kg)	Tegangan (mv)
1,00	100
0,96	96
0,92	92
0,88	88
0,84	84
0,80	80
0.76	76
0,72	72

Sumber: Pengujian.

Grafik hasil pengujian sensor berat dapat dilihat dalam gambar 5.9 berikut



Gambar 5.9. Grafik Hasil Pengujian Sensor Berat

Sumber: Pengujian.

Hasil pengujian sensor berat dapat dilihat dalam gambar 5.10 berikut

Gambar 5.10. Pengujian Sensor Berat Dengan Beban 1 kg

Sumber: Pengujian.

5.1.4. Pengujian driver motor gearbox dan motor gearbox

a. Tujuan

Untuk mengetahu apakah driver motor gearbox dan motor gearbox berfungsi dengan baik atau tidak

b. Peralatan

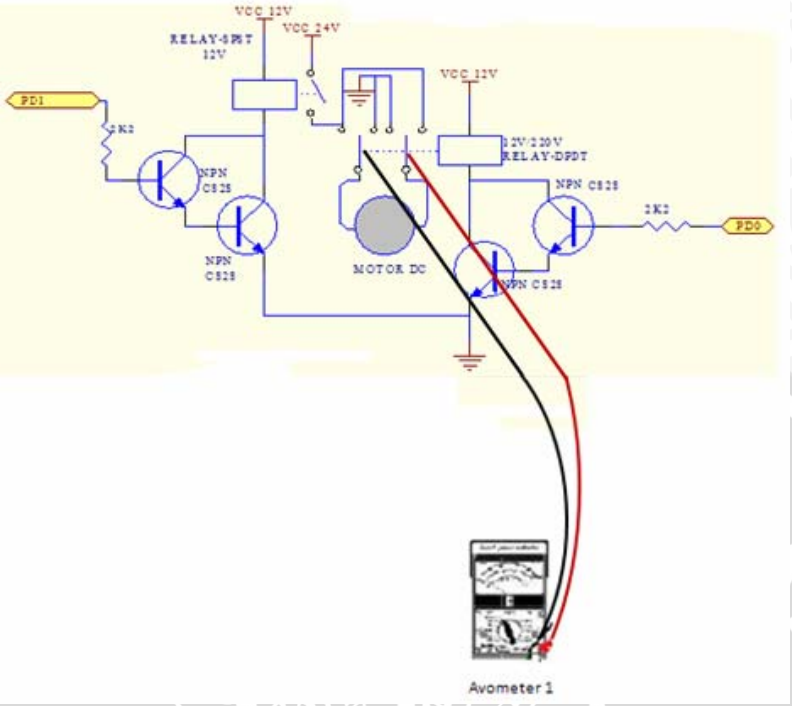
-Avometer

- catu daya 5 volt dan 12 volt

- driver relay

c. Langkah Pengujian

- mengaktifkan catu daya 5 volt dan 12 volt
- menghubungkan avometer dengan driver motor seperti pada gambar 5.4
- Triger pada driver relay beri logika 0
- baca tegangan yang terukur pada avometer
- Ulangi dengan triger pada driver relay berlogika 1
- baca tegangan yang terukur pada avometer.



Gambar 5.11 Pengujian *driver* motor dan motor *gearbox*

Sumber: Pengujian.

Keterangan Gambar :

- Avometer : Untuk mengukur tegangan motor saat *ON* dan *OFF*

d. Hasil Pengujian

Tabel 5.7. Hasil pengujian driver motor

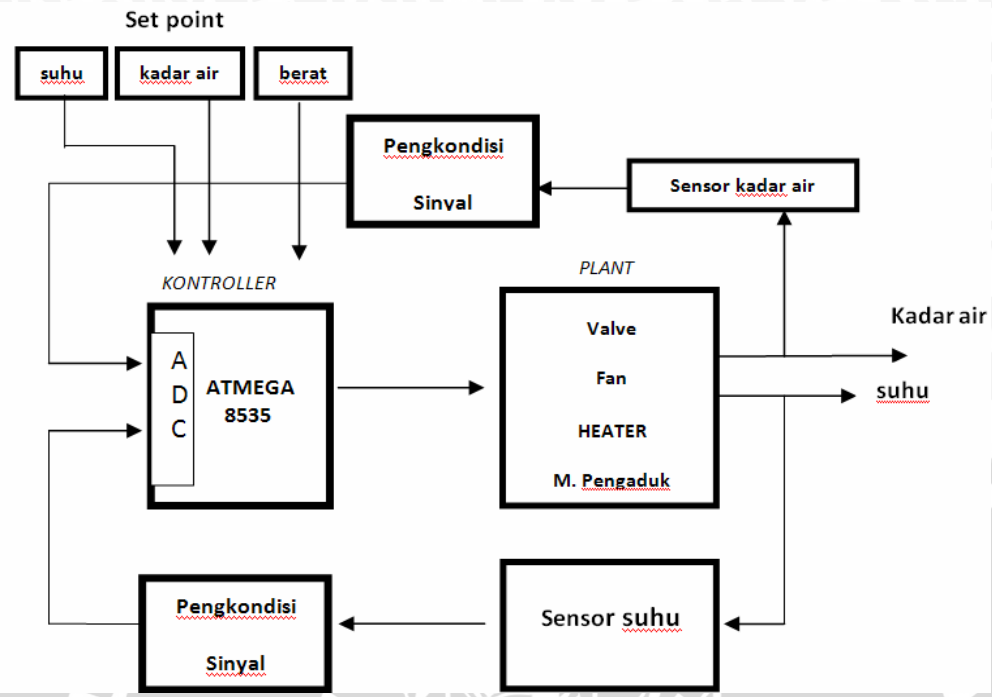
Kondisi triger	Tegangan pada motor (volt)
Logika 0	0 (motor mati)
Logika 1	12 (motor jalan)

Sumber: Pengujian

5.2. Pengujian sistem secara keseluruhan

a. Tujuan

Pengujian ini bertujuan untuk mengetahui unjuk kerja perangkat keras dengan perangkat lunak setelah diintegrasikan bersama-sama. Cara pengujiannya yaitu dengan menjalankan alat ini dan mengamatinya, apakah alat sudah bisa berjalan dengan baik. Blok diagram pengujian sistem secara keseluruhan ditunjukkan dalam Gambar 5.12.



Gambar 5.12. Blok Diagram Pengujian Sistem Secara Keseluruhan.

Sumber: Pengujian.

b. Peralatan

- Catu daya 220 V.
- Alat pengatur kadar air.
- Benih jagung 2,5kg
- Alat ukur kadar air.
- Thermometer.
- osiloskop storage

c. Langkah pengujian

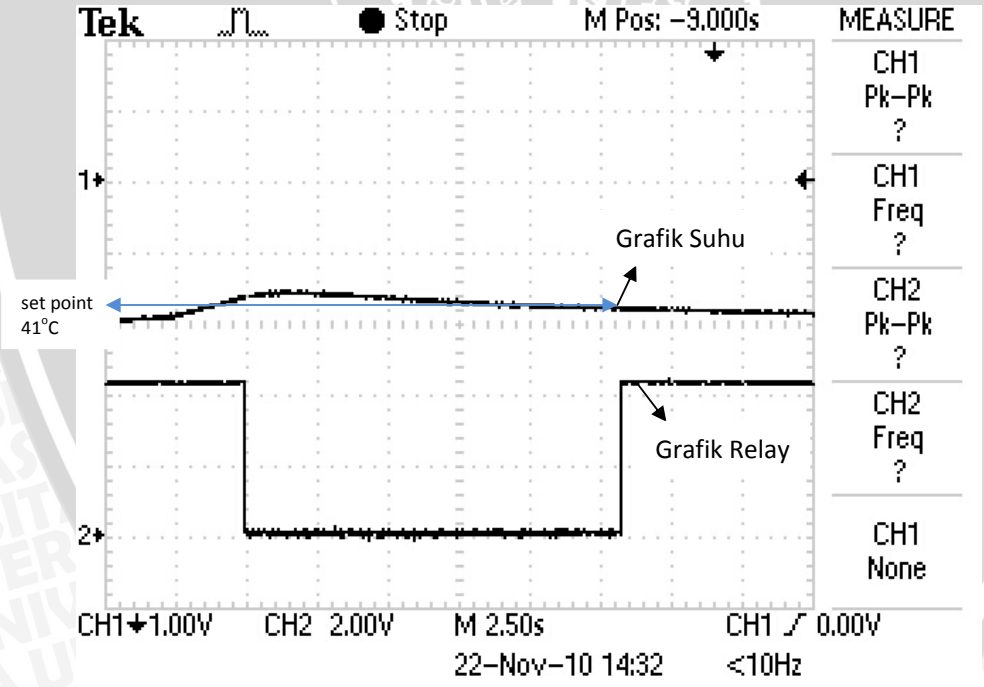
- Hidupkan alat pengatur kadar air.
- Masukkan benih jagung ke dalam bak penampung.
- Ukur suhu ruangan dalam bak penampung dengan thermometer secara berkala tiap 2 menit.
- Hitung lama waktu yang dibutuhkan alat hingga valve terbuka dan jagung mulai keluar.
- Ukur kadar air jagung yang telah dikeringkan dengan alat ukur kadar air

d. Hasil Pengujian

Dari pengujian sistem secara keseluruhan maka didapatkan hasil sebagai berikut:

- Suhu ruangan dalam bak penampung yang terukur dengan thermometer adalah 41^{o}C sampai 44^{o}C
- Saat berlangsungnya proses pengeringan jagung terlihat pada tampilan LCD bahwa berat jagung turun yang menandakan sensor berat bekerja dengan baik.
- Saat proses pengeringan motor mampu berjalan dengan baik sehingga alat pengaduk bekerja.
- Saat LCD menunjukkan kadar air 12%, valve terbuka dan jagung keluar dengan otomatis dari bak penampung.

Grafik keluaran plan sistem pada osiloskop dengan input suhu dan output relay dapat dilihat dalam gambar 5.13 berikut

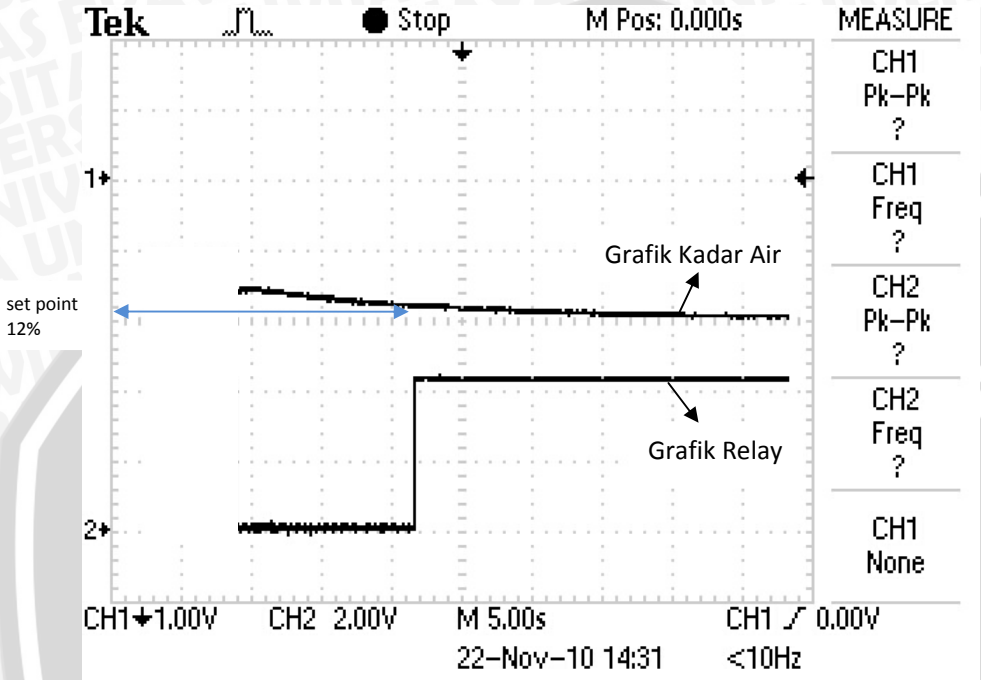


Gambar 5.13.Grafik keluaran plan sistem pada osiloskop dengan input suhu dan output relay

Sumber: Pengujian.

Dari grafik pada gambar 5.13 telihat pada saat suhu kurang dari set point maka relay ON dan saat suhu mencapai set point maka relay OFF.

Grafik keluaran plan sistem pada osiloskop dengan input kadar air dan output relay dapat dilihat dalam gambar 5.14 berikut

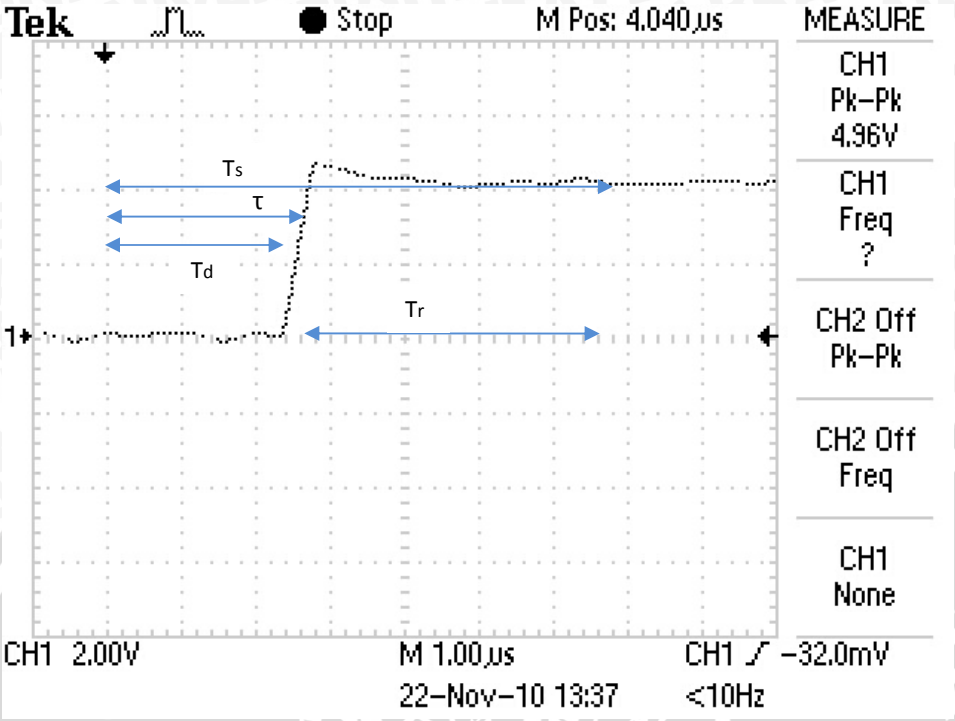


Gambar 5.14.Grafik keluaran plan sistem pada osiloskop dengan input kadar air dan output relay

Sumber: Pengujian.

Dari grafik pada gambar 5.14 telihat pada saat kadar air melebihi dari set point maka relay OFF dan saat kadar air mencapai set point maka relay ON.

Grafik keluaran plan sistem pada osiloskop untuk menunjukkan performa relay dapat dilihat dalam gambar 5.15 berikut



Gambar 5.15.Grafik keluaran plan sistem pada osiloskop untuk output relay pemanas.

Sumber: Pengujian.

Dari grafik pada gambar 5.15 didapatkan data sebagai berikut:

1) Respon mencapai *steady state* dalam waktu 3,8µs

2) *Delay time*, t**d** : ukuran waktu yang menyatakan faktor keterlambatan respon *output*

terhadap *input*, di ukur mulai t=0 s/d respon mencapai 50% dari respon *steady state.*

- td yang didapatkan dari pengujian ini adalah 2,6 µs.

3) *Time constan*, τ: ukuran waktu yang menyatakan kecepatan respon, yang di ukur mulai t=0 s/d respon mencapai 63,2% dari respon *steady state.*

- yang didapatkan dari pengujian ini adalah 2,8 µs.

4) *Settling time*, **ts** : ukuran waktu yang menyatakan respon telah masuk ±5% atau ±2% atau ±0,5% dari respon *steady state.*

- ts yang didapatkan dari pengujian ini adalah 6,75 µs.

5) *Rise time*, tr: ukuran waktu yang menyatakan keberadaan suatu respon, yang di ukur mulai respon 5% s/d 95% dari respon *steady state*.

- yang didapatkan dari pengujian ini adalah 3,75µs.

6) *Error steady state* yang terjadi sebesar 16%

$$= \frac{4,2V-5V}{5} \times 100\%$$

$$= 16\%$$

Hasil dari pengujian sistem keseluruhan alat pengatur kadar air untuk set point suhu adalah sebagai berikut:

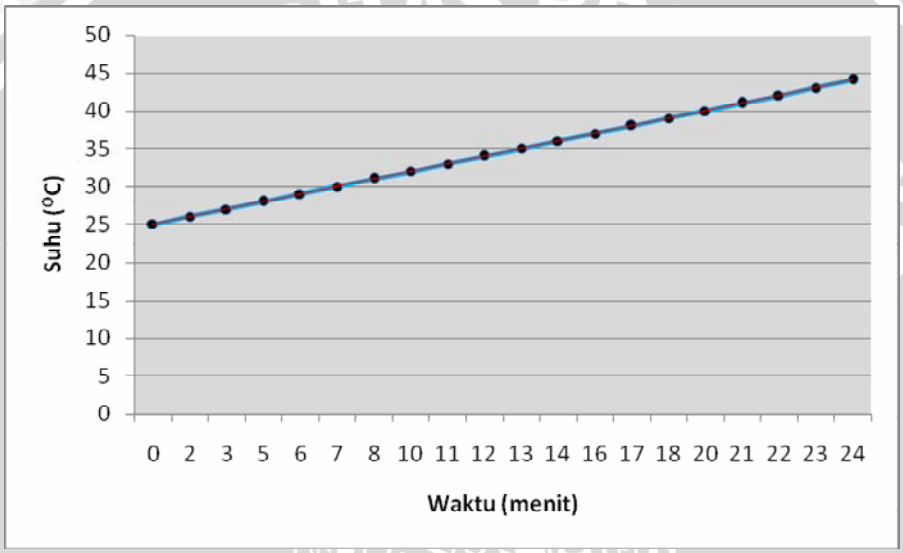
Tabel 5.8. Hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point suhu

Suhu (^{o}C)	Waktu (menit)
26	2
27	3
28	5
29	6
30	7
31	8
32	10
33	11
34	12
35	13
36	14
37	16
38	17
39	18
40	20
41	21
42	22

43	23
44	24

Sumber: Pengujian

Grafik hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point suhu adalah sebagai berikut



Gambar 5.16. Grafik hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point suhu

Sumber: Pengujian

Berdasarkan Grafik Pengujian 5.16 menunjukkan bahwa ruang pemanas mampu mencapai set point 41^{o}C dalam waktu 21 menit.

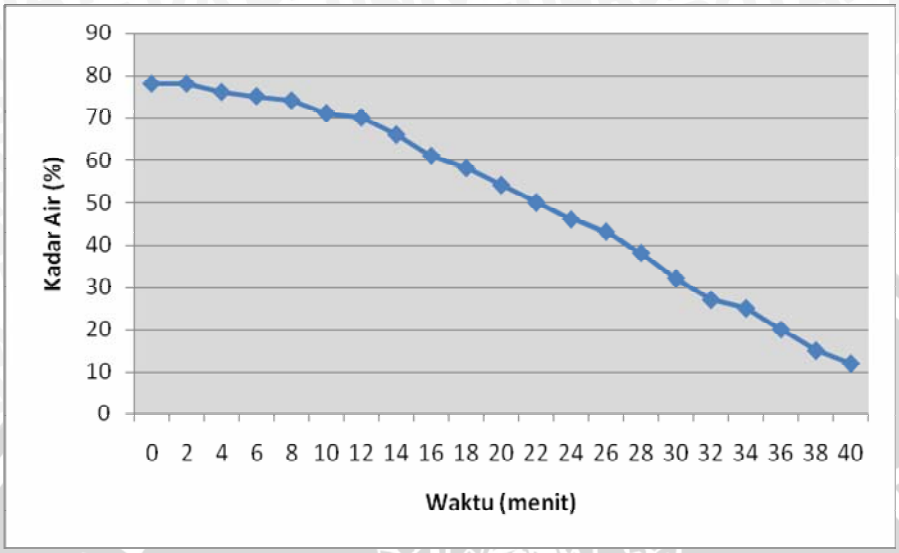
Hasil dari pengujian sistem keseluruhan alat pengatur kadar air untuk set point kadar air adalah sebagai berikut:

Tabel 5.9. Hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point kadar air

Kadar Air (%)	Waktu (menit)
78	0
78	2
76	4
75	6
74	8
71	10
70	12
66	14
61	16
58	18
54	20
50	22
46	24
43	26
38	28
32	30
27	32
25	34
20	36
15	38
12	40

Sumber: Pengujian

Grafik hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point suhu adalah sebagai berikut



Gambar 5.17. Grafik hasil pengujian sistem keseluruhan alat pengatur kadar air untuk set point kadar air.

Sumber: Pengujian

Berdasarkan Grafik Pengujian 5.17 menunjukkan bahwa alat mampu mengeringkan benih jagung dengan set point kadar air 12% dalam waktu 40 menit.

Dari hasil pengujian sistem keseluruhan alat pengatur kadar air dapat diambil hubungan antara kadar air dan berat benih jagung yang diukur sebagai berikut:

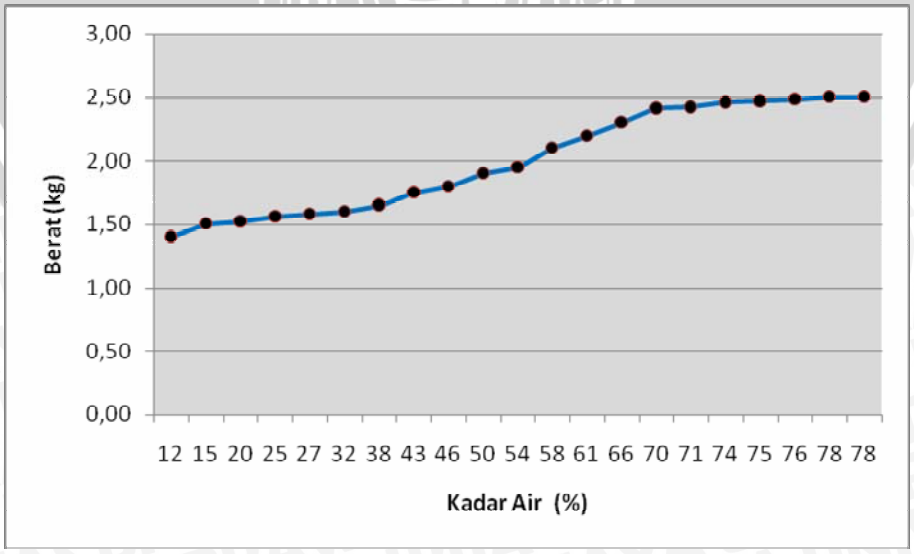
Tabel 5.10. Hubungan antara kadar air dan berat benih jagung

Kadar Air (%)	Berat (kg)
78	2.50
78	2.50
76	2.48
75	2.47
74	2.46
71	2.42

70	2.41
66	2.30
61	2.20
58	2.10
54	1.95
50	1.90
46	1.80
43	1.75
38	1.65
32	1.60
27	1.58
25	1.56
20	1.52
15	1.50
12	1.40

Sumber: Pengujian

Grafik hubungan antara kadar air dan berat benih jagung yang diukur sebagai berikut:



Gambar 5.18. Grafik hubungan antara kadar air dan berat benih jagung yang diukur

Sumber: Pengujian

Dari Grafik 5.18 dapat dinyatakan bahwa semakin berkurangnya kadar air pada benih jagung maka akan menyebabkan semakin berkurang juga berat dari benih jagung tersebut.

BAB VI
KESIMPULAN DAN SARAN

6.1. Kesimpulan

Dari hasil pengujian, perencanaan, pembuatan serta pengujian alat, maka dapat ditarik kesimpulan sebagai berikut:

1. Dari hasil pengujian, relay dapat mengaktifkan plan dalam waktu 2,4 µs.
2. Dari hasil pengujian menunjukkan bahwa ruang pemanas mampu mencapai set point 41^{o}C dalam waktu 21 menit.
3. Dari hasil pengujian menunjukkan bahwa alat mampu mengeringkan benih jagung dengan set point kadar air 12% dalam waktu 40 menit.
4. Dari hasil pengujian dapat dinyatakan bahwa semakin berkurangnya kadar air pada benih jagung maka akan menyebabkan semakin berkurang juga berat dari benih jagung tersebut.

6.2. Saran

1. Untuk memperbesar penguatan sensor berat maka dapat ditambahkan IC 521.
2. Untuk meningkatkan kepresisian sensor kadar air maka lempengan tembaga pada sensor dapat diganti dengan kuningan atau konduktor dengan konduktansi yang lebih baik.

DAFTAR PUSTAKA

Arief, R., S. Saenong, T. M. Lando, Fauziah Koes, dan Rahmawati. 2001. *Pengaruh cara pengeringan terhadap mutu dan daya simpan benih jagung*. Penelitian Pertanian 20(3): 41-47.

ATMEL. 2007. *ATMEGA8535/ATMEGA8535L, 8-bit AVR Microcontroller with 8 Kbytes in System Programable Flash.*

Balai Pengkajian Teknologi Pertanian (1998). *Budidaya Kedelai dan Jagung. Palangkaraya*. Departemen Pertanian.

Couglin F. Robert dan Driscoll F. Frederick. 1992. *Penguat Operational dan Rangkaian Terpadu Linear*. Cetakan Kedua. Penerjemah: Soemitro, Herman Widodo. Jakarta: Erlangga.

Frederick F. Driscoll, 1994, *Penguat Operasional dan Rangkaian Terpadu Linier*, Erlangga, Jakarta.

Malvino, Albert Paul. 1996. *Prinsip-Prinsip Elektronika.* Edisi Ketiga. Penerjemah: Gunawan, Hanapi. Jakarta: Erlangga

Modul Praktikum FI – 2104 Elektronika Dasar. 2005. *Transistor Sebagai Saklar Penguat Gandengan DC Darlington dan Penguat Diferensial*. Laboratorium Elektronika dan Instrumentasi Fisika ITB. Bandung

Ogata, Katsuhiko. 1997. *Teknik Kontrol Automatik*. Jakarta. Penerbit Erlangga.

Owen R. Fennema, Markus Karel, Daryl B. lund. 1975. *Physical principles of food preservation*.Marcel Dekker,inc. New York

Sukarman dan Hasanah, Maharani. 2003. *Perbaikan Mutu Benih Aneka Tanaman Perkebunan Melalui Cara Panen dan Penanganan Benih.* Bogor. Jurnal Litbang Pertanian

Taib,Gunarif. 1988. *Operasi Pengeringan Pada Pengolahan Hasil Pertanian*. Jakarta. PT.Melton Putra.

Ir. Suharto. *Teknologi pengawetan pangan*.1991.PT rineka cipta, jakarta

http://www.ristek.go.id.pdf/jagung/

Diakses 04 Desember 2009

http://www.galerimesin.com/ Pembuatan Alat Ukur Kadar Air Biji-bijian/

Diakses 04 Desember 2009

http://www.balitsereal.litbang.deptan.go.id/index.php?option=com_content&task=view&id=63& Itemid=141

Diakses 05 Desember 2009

http://www.agrina-online.com/index.php

Diakses 05 Desember 2009

http://derryariadi.blogspot.com/

Diakses 05 Desember 2009

http://Franciscus Aditya Priyawan Blog.com

Diakses 05 Desember 2009

LAMPIRAN I

Foto Alat

Foto mekanik alat ditunjukkan dalam foto 1

Foto1. Mekanik Alat

Foto mekanik alat dan rangkaian elektronik ditunjukkan dalam foto 2

Foto 2. Mekanik Alat dan Rangkaian Elektronika

LAMPIRAN II

Listing Program

```
;---------------------------------------
;-------------jagung----------------------
;---------------------------------------
heater          bit     p1.0
pengaduk        bit     p1.1
valve           bit     p1.2
valve_s                 bit     p1.3

;
ADC_A                   bit     P3.7
ADC_B                   bit     P3.6
ADC_C                   bit     P3.5


ADC_OE                  bit     P3.4
ADC_START bit           P3.1
ADC_EOC                 bit     P3.0

lcd_x           equ     50h
satuan          equ     49h
puluhan                 equ     51h
ratusan         equ     52h
diti            equ     53h
lcd_ratusan     equ     54h
lcd_puluhan     equ     55h
lcd_satuan      equ     56h

                Org     00h
                ljmp    mulai

;=======================================================
;
```

```
write_inst:
            clr     P3.3    ;RS     untuk menuliskan isntruksi
write_LCD:
            mov     lcd_X,a
            mov     P0,A
            setb    P3.2    ;E
            clr     P3.2    ;E
            mov     a,lcd_X
            swap    a
            mov     P0,A
            setb    P3.2    ;E
            clr     P3.2    ;E
            call    delay
            ret
write_data:
            setb    P3.3    ;RS     untuk menuliskan data / character
            jmp     write_LCD

;
delay:      mov     R7,#0
delay1:     mov     R5,#50h
            djnz    R5,$
            djnz    R7,delay1
            ret
;
Ldelay:     mov     R2,#030h
Ld1:        call    delay
            djnz    R2,Ld1
            ret
;
barisa: mov A,#80h          ; menulis di baris atas
```

```
tulis16:
        mov   R3,#16          ; sebanyak 16 character
        call  write_inst
tulis1: clr   A
        movc  A,@A+DPTR ; ambil data dari pointer
        Inc   DPTR
        call  write_data
        djnz  R3,Tulis1
        ret

barisb: mov   A,#0C0h         ; menulis di baris bawah
        jmp   tulis16


; kondisi awal
ambil_ADC:
        clr   ADC_OE
        clr   ADC_START
        nop
        nop
        nop
        setb  ADC_START ; start of conversion
        nop
        clr   ADC_START
not_EOC:
        jnb   ADC_EOC,not_EOC
delaya: djnz  R2,$
        djnz  R3,delaya
        setb  ADC_OE      ; Baca Data melalui P3
        djnz  R3,$
        mov   A,P2
        cpl   a
```

```
                mov     r1,a
                ret

CONVERSI:
                MOV   A,#10
                MOV   B,A
                MOV   A,R1
                DIV    AB
                MOV   DITI,A
                MOV   A,B
                MOV   SATUAN,A
;
                MOV   A,#10
                MOV   B,A
                MOV   A,DITI
                DIV    AB
                MOV   RATUSAN,A
                MOV   A,B
                MOV   PULUHAN,A
                RET

kalibrasi_suhu:
                mov     r2,#124
shp:            dec     r1
                djnz    r2,shp

                cjne    r1,#44,lihat1
lihat1:         jnc     lih1
                cjne    r1,#41,lihat2
lihat2:         jc      lih2
                ret
lih1:           clr     heater
```

```
                ret
lih2:           setb    heater
                ret

kalibrasi_kadar_air:
                mov     r2,#129
shpa:           dec     r1
                djnz    r2,shpa
                mov     a,r1
                cpl     a
                mov     r1,a
                ret

kalibrasi_berat_jagung:
                mov     r2,#120
ashpa:          dec     r1
                djnz    r2,ashpa
                mov     a,r1
                cpl     a
                mov     r1,a
                ret

TAMPILAN:
                MOV   DPTR,#ANGKA
                LCALL         PERSATUAN
                lcall    PERPULUHAN
                lcall   PERSERATUS
                RET
```

```
PERSATUAN:
        CLR   A
        MOV   A,lcd_satuan
        CALL  WRITE_INST
        CLR   A
        MOV   A,SATUAN
        MOVC        A,@A+DPTR
        CALL  WRITE_DATA
        RET

PERPULUHAN:
        CLR   A
        MOV   A,lcd_puluhan
        CALL  WRITE_INST
        CLR   A
        MOV   A,PULUHAN
        MOVC        A,@A+DPTR
        CALL  WRITE_DATA
        RET

PERSERATUS:
        CLR   A
        MOV   A,Lcd_ratusan
        CALL  WRITE_INST
        CLR   A
        MOV   A,RATUSAN
        MOVC        A,@A+DPTR
        CALL  WRITE_DATA
        RET

Mulai:
        clr     heater
```

```
        clr     pengaduk
        clr     valve
        clr     valve_s
        call    write_inst
        Mov     A,#3fh
        call    write_inst
        Mov     A,#20h
        call    write_inst
        Mov     A,#28h
        call    write_inst
        Mov     A,#08h
        call    write_inst
        Mov     A,#01h
        call    write_inst
        Mov     A,#0EH
        call    write_inst
        Mov     A,#06h
        call    write_inst
        Mov     A,#0Dh
        call    write_inst
        Mov     A,#0CH
        call    write_inst

kembali:
        Mov     DPTR,#candra
        Mov     R4,#3
lagi:   call    barisa
        call    barisb
        call    ldelay
        Djnz    r4,lagi
        setb    pengaduk

lagiy:
```

```
;====suhu====
        clr     ADC_A
        setb    ADC_B
        setb    ADC_C
        call    Ambil_adc
        call    kalibrasi_suhu
        call    conversi
        mov     lcd_ratusan,#0c7h
        mov     lcd_puluhan,#0c8h
        mov     lcd_satuan,#0c9h
        call    tampilan
;====kadar air====

        setb    ADC_A
        setb    ADC_B
        setb    ADC_C
        call    Ambil_adc
        cpl     a
        mov     r1,a
        call    kalibrasi_kadar_air
        call    conversi
        mov     lcd_ratusan,#0cch
        mov     lcd_puluhan,#0cdh
        mov     lcd_satuan,#0ceh
        call    tampilan

;====sensor berat====

        clr     ADC_A
        clr     ADC_B
        setb    ADC_C
        call    Ambil_adc
```

```
        cpl     a
        mov     r1,a
        call    kalibrasi_berat_jagung
        call    conversi
        mov     lcd_ratusan,#0c1h
        mov     lcd_puluhan,#0c2h
        mov     lcd_satuan,#0c3h
        call    tampilan
        ljmp    lagiy
control:
        setb    valve
        setb    valve_s
        call    ldelay
        call    ldelay
        clr     valve
        clr     valve_s
        ret

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;===Tampilan Pertama===
candra:
        DB      '    candra     '
        DB      ' T.Elektro (D)  '
        DB      '    UNIBRAW     '
        DB      ' Nim:0510633017 '
        DB      'berat suhu K.air'
        DB      '               '

ANGKA:  DB          '0123456789   BCDEF'

end
```

LAMPIRAN III

Datasheet Komponen

Features

- High-performance, Low-power AVR® 8-bit Microcontroller
- Advanced RISC Architecture
  - 130 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
- Nonvolatile Program and Data Memories
  - 8K 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
  - 512 Bytes Internal SRAM
  - Programming Lock for Software Security
- Peripheral 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
  - Real Time Counter with Separate Oscillator
  - Four PWM Channels
  - 8-channel, 10-bit ADC
    8 Single-ended Channels
    7 Differential Channels for TQFP Package Only
    2 Differential Channels with Programmable Gain at 1x, 10x, or 200x for TQFP Package Only
  - 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, 44-lead PLCC, and 44-pad MLF
- Operating Voltages
  - 2.7 - 5.5V for ATmega8535L
  - 4.5 - 5.5V for ATmega8535
- Speed Grades
  - 0 - 8 MHz for ATmega8535L
  - 0 - 16 MHz for ATmega8535

ATMEL

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

ATmega8535
ATmega8535L

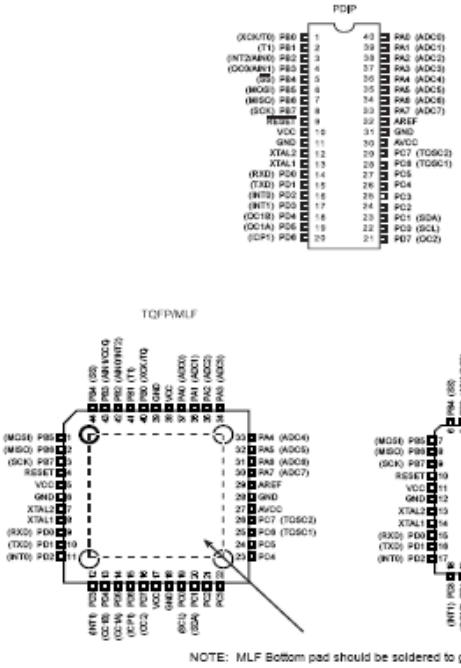
Preliminary

Rev. 2502E–AVR–12/03

ATMEL

Pin Configurations

Figure 1. Pinout ATmega8535



NOTE: MLF Bottom pad should be soldered to ground.

Disclaimer

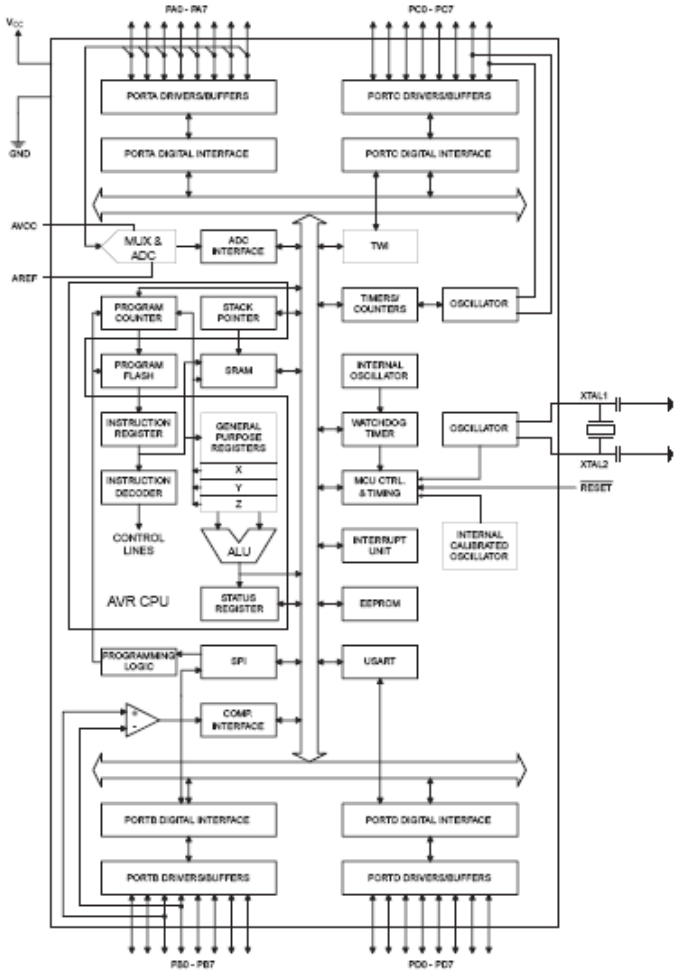
Typical values contained in this data sheet are based on simulations and characterization of other AVR microcontrollers manufactured on the same process technology. Min and Max values will be available after the device is characterized.

Overview

The ATmega8535 is a low-power CMOS 8-bit microcontroller based on the AVR enhanced RISC architecture. By executing instructions in a single clock cycle, the ATmega8535 achieves throughputs approaching 1 MIPS per MHz allowing the system designer to optimize power consumption versus processing speed.

Block Diagram

Figure 2. Block Diagram



The AVR core combines a rich instruction set with 32 general purpose working registers. All 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 ATmega8535 provides the following features: 8K bytes of In-System Programmable Flash with Read-While-Write capabilities, 512 bytes EEPROM, 512 bytes SRAM, 32 general purpose I/O lines, 32 general purpose working registers, three flexible Timer/Counters with compare modes, internal and external interrupts, a serial programmable USART, a byte oriented Two-wire Serial Interface, an 8-channel, 10-bit ADC with optional differential input stage with programmable gain in TQFP package, 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 SRAM, Timer/Counters, SPI port, and interrupt system to continue functioning. The Power-down mode saves the register contents but freezes the Oscillator, disabling all other chip functions until the next interrupt or Hardware Reset. In Power-save mode, the asynchronous timer continues to run, allowing the user to maintain a timer base while the rest of the device is sleeping. The ADC Noise Reduction mode stops the CPU and all I/O modules except asynchronous timer and ADC, to minimize switching noise during ADC conversions. In Standby mode, the crystal/resonator Oscillator is running while the rest of the device is sleeping. This allows very fast start-up combined with low-power consumption. 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 ATmega8535 is a powerful microcontroller that provides a highly flexible and cost effective solution to many embedded control applications.

The ATmega8535 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.

AT90S8535 Compatibility

The ATmega8535 provides all the features of the AT90S8535. In addition, several new features are added. The ATmega8535 is backward compatible with AT90S8535 in most cases. However, some incompatibilities between the two microcontrollers exist. To solve this problem, an AT90S8535 compatibility mode can be selected by programming the S8535C fuse. ATmega8535 is pin compatible with AT90S8535, and can replace the AT90S8535 on current Printed Circuit Boards. However, the location of fuse bits and the electrical characteristics differs between the two devices.

AT90S8535 Compatibility Mode

Programming the S8535C fuse will change the following functionality:

- The timed sequence for changing the Watchdog Time-out period is disabled. See "Timed Sequences for Changing the Configuration of the Watchdog Timer" on page 43 for details.
- The double buffering of the USART Receive Register is disabled. See "AVR USART vs. AVR UART – Compatibility" on page 143 for details.

Pin Descriptions

V_{CC}	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.
Port 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 ATmega8535 as listed on page 58.
Port 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.
Port 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 ATmega8535 as listed on page 62.
$\overline{\text{RESET}}$	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 35. Shorter pulses are not guaranteed to generate a reset.
XTAL1	Input to the inverting Oscillator amplifier and input to the internal clock operating circuit.
XTAL2	Output from the inverting Oscillator amplifier.
AVCC	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.
AREF	AREF is the analog reference pin for the A/D Converter.

About Code Examples

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.

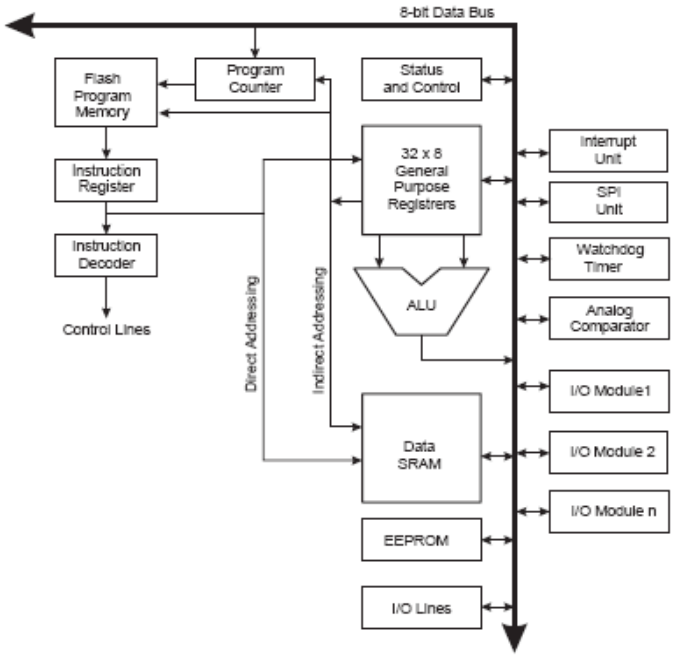
AVR CPU Core

Introduction

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

Architectural Overview

Figure 3. Block Diagram of the AVR MCU Architecture



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

enables instructions to be executed in every clock cycle. The program memory is In-System Re-Programmable Flash memory.

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

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

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

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

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

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

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

A flexible interrupt module has its control registers in the I/O space with an additional Global Interrupt Enable bit in the Status Register. All interrupts have a separate Interrupt Vector in the Interrupt Vector table. The interrupts have priority in accordance with their Interrupt Vector position. The lower the Interrupt Vector address, the higher the priority.

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

ALU – Arithmetic Logic Unit

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

Status Register

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

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

The AVR Status Register – SREG – is defined as:

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

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

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

- **Bit 6 – T: Bit Copy Storage**

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

- **Bit 5 – H: Half Carry Flag**

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

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

The S-bit is always an exclusive or between the Negative Flag N and the Two's Complement Overflow Flag V. See the "Instruction Set Description" for detailed information.

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

The Two's Complement Overflow Flag V supports two's complement arithmetics. See the "Instruction Set Description" for detailed information.

- **Bit 2 – N: Negative Flag**

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

- **Bit 1 – Z: Zero Flag**

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

- **Bit 0 – C: Carry Flag**

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

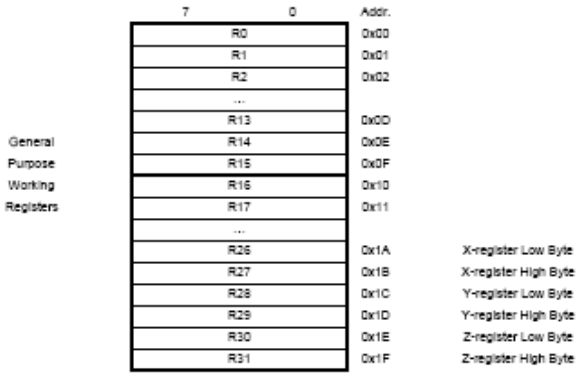
General Purpose Register File

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

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

Figure 4 shows the structure of the 32 general purpose working registers in the CPU.

Figure 4. AVR CPU General Purpose Working Registers



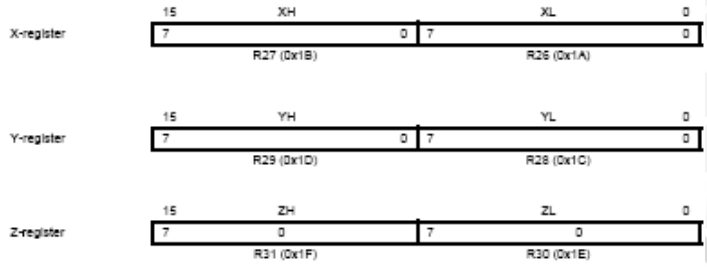
Most of the instructions operating on the Register File have direct access to all registers, and most of them are single cycle instructions.

As shown in Figure 4, each register is also assigned a data memory address, mapping them directly into the first 32 locations of the user Data Space. Although not being physically implemented as SRAM locations, this memory organization provides great flexibility in access of the registers, as the X-, Y-, and Z-pointer Registers can be set to index any register in the file.

The X-register, Y-register, and Z-register

The registers R26..R31 have some added functions to their general purpose usage. These registers are 16-bit address pointers for indirect addressing of the Data Space. The three indirect address registers X, Y, and Z are defined as described in Figure 5.

Figure 5. The X-, Y-, and Z-registers



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

Stack Pointer

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

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

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

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

Instruction Execution Timing

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

Figure 6 shows the parallel instruction fetches and instruction executions enabled by the Harvard architecture and the fast-access Register File concept. This is the basic pipelining concept to obtain up to 1 MIPS per MHz with the corresponding unique results for functions per cost, functions per clocks, and functions per power-unit.

Figure 6. The Parallel Instruction Fetches and Instruction Executions

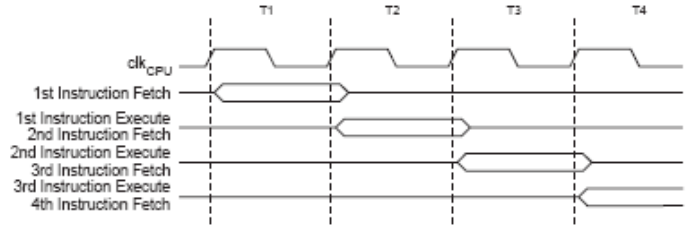
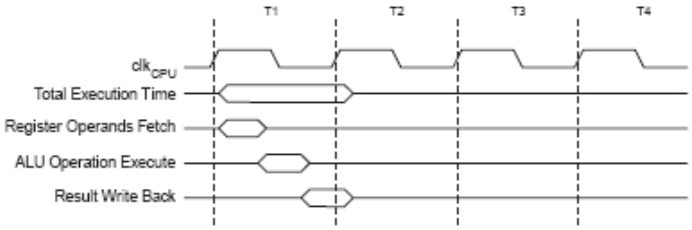


Figure 7 shows the internal timing concept for the Register file. In a single clock cycle an ALU operation using two register operands is executed, and the result is stored back to the destination register.

Figure 7. Single Cycle ALU Operation



Reset and Interrupt Handling

The AVR provides several different interrupt sources. These interrupts and the separate Reset Vector each have a separate Program Vector in the program memory space. All interrupts are assigned individual enable bits which must be written logic one together with the Global Interrupt Enable bit in the Status Register in order to enable the interrupt. Depending on the Program Counter value, interrupts may be automatically disabled when Boot Lock bits BLB02 or BLB12 are programmed. This feature improves software security. See the section "Memory Programming" on page 234 for details.

The lowest addresses in the program memory space are, by default, defined as the Reset and Interrupt Vectors. The complete list of vectors is shown in "Interrupts" on page 44. The list also determines the priority levels of the different interrupts. The lower the address, the higher the priority level is. RESET has the highest priority, and next is INT0 – the External Interrupt Request 0. The Interrupt Vectors can be moved to the start of the Boot Flash section by setting the IVSEL bit in the General Interrupt Control Register (GICR). Refer to "Interrupts" on page 44 for more information. The Reset Vector can

also be moved to the start of the Boot Flash section by programming the BOOTRST Fuse, see "Boot Loader Support – Read-While-Write Self-Programming" on page 221.

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

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

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

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

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

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

Assembly Code Example

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

C Code Example

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

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

Assembly Code Example
sei *; set global interrupt enable* **sleep** *; enter sleep, waiting for interrupt* ; note: will enter sleep before any pending ; interrupt(s)
C Code Example
_SEI(); */* set global interrupt enable */* _SLEEP(); */* enter sleep, waiting for interrupt */* /* note: will enter sleep before any pending interrupt(s) */

Interrupt Response Time

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

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

AVR ATmega8535 Memories

This section describes the different memories in the ATmega8535. The AVR architecture has two main memory spaces, the Data Memory and the Program Memory space. In addition, the ATmega8535 features an EEPROM Memory for data storage. All three memory spaces are linear and regular.

In-System Reprogrammable Flash Program Memory

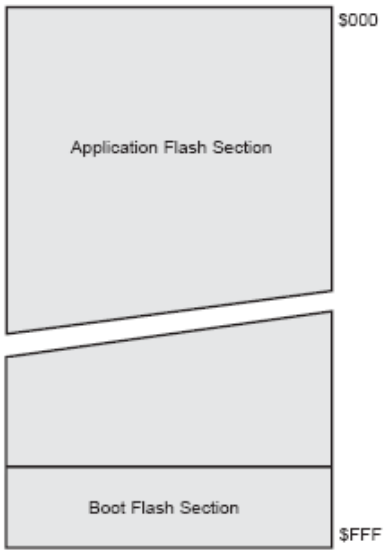
The ATmega8535 contains 8K bytes On-chip In-System Reprogrammable Flash memory for program storage. Since all AVR instructions are 16 or 32 bits wide, the Flash is organized as 4K x 16. For software security, the Flash Program memory space is divided into two sections, Boot Program section and Application Program section.

The Flash memory has an endurance of at least 10,000 write/erase cycles. The ATmega8535 Program Counter (PC) is 12 bits wide, thus addressing the 4K program memory locations. The operation of Boot Program section and associated Boot Lock bits for software protection are described in detail in "Boot Loader Support – Read-While-Write Self-Programming" on page 221. "Memory Programming" on page 234 contains a detailed description on Flash Programming in SPI or Parallel Programming mode.

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

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

Figure 8. Program Memory Map



SRAM Data Memory

Figure 9 shows how the ATmega8535 SRAM Memory is organized.

The 608 Data Memory locations address the Register File, the I/O Memory, and the internal data SRAM. The first 96 locations address the Register File and I/O Memory, and the next 512 locations address the internal data SRAM.

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

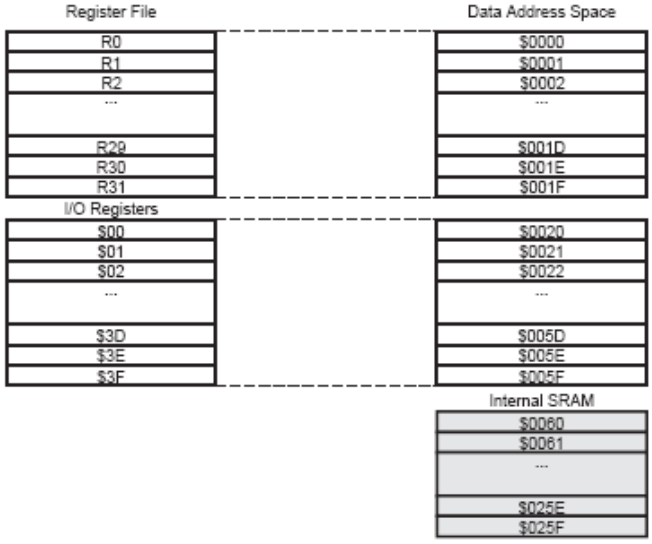
The direct addressing reaches the entire data space.

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

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

The 32 general purpose working registers, 64 I/O Registers, and the 512 bytes of internal data SRAM in the ATmega8535 are all accessible through all these addressing modes. The Register File is described in "General Purpose Register File" on page 9.

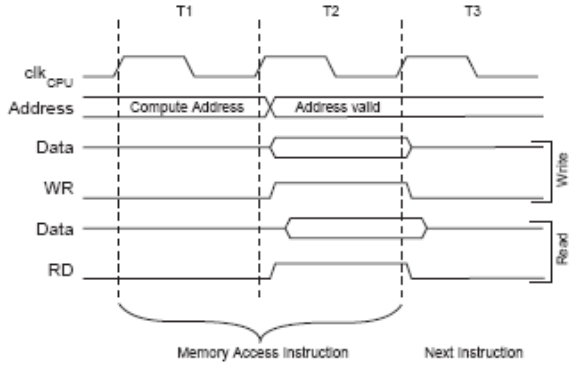
Figure 9. Data Memory Map



Data Memory Access Times

This section describes the general access timing concepts for internal memory access. The internal data SRAM access is performed in two clk_{CPU} cycles as described in Figure 10.

Figure 10. On-chip Data SRAM Access Cycles



EEPROM Data Memory

The ATmega8535 contains 512 bytes of data EEPROM memory. It is organized as a separate data space, in which single bytes can be read and written. The EEPROM has an endurance of at least 100,000 write/erase cycles. The access between the EEPROM and the CPU is described in the following, specifying the EEPROM Address Registers, the EEPROM Data Register, and the EEPROM Control Register.

"Memory Programming" on page 234 contains a detailed description on EEPROM Programming in SPI or Parallel Programming mode.

EEPROM Read/Write Access

The EEPROM Access Registers are accessible in the I/O space.

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

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

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

The EEPROM Address Register – EEARH and EEARL

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

- **Bits 15..9 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8535 and will always read as zero.

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

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

The EEPROM Data Register – EEDR

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

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

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

The EEPROM Control Register – EECR

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

- **Bits 7..4 – Res: Reserved Bits**

These bits are reserved bits in the ATmega8535 and will always read as zero.

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

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

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

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

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

The EEPROM Write Enable Signal EEWE is the write strobe to the EEPROM. When address and data are correctly set up, the EEWE bit must be written to one to write the

value into the EEPROM. The EEMWE bit must be written to one before a logical one is written to EEWE, otherwise no EEPROM write takes place. The following procedure should be followed when writing the EEPROM (the order of steps 3 and 4 is not essential):

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

The EEPROM can not be programmed during a CPU write to the Flash memory. The software must check that the Flash programming is completed before initiating a new EEPROM write. Step 2 is only relevant if the software contains a Boot Loader allowing the CPU to program the Flash. If the Flash is never updated by the CPU, step 2 can be omitted. See "Boot Loader Support – Read-While-Write Self-Programming" on page 221 for details about Boot programming.

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

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

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

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

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

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

Table 1. EEPROM Programming Time

Symbol	Number of Calibrated RC Oscillator Cycles[1]	Typ Programming Time
EEPROM Write (from CPU)	8448	8.4 ms

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

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

Assembly Code Example

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

C Code Example

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

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

Assembly Code Example

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

C Code Example

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

EEPROM Write During Power-down Sleep Mode

When entering Power-down sleep mode while an EEPROM write operation is active, the EEPROM write operation will continue, and will complete before the write access time has passed. However, when the write operation is completed, the Oscillator continues running, and as a consequence, the device does not enter Power-down entirely. It is therefore recommended to verify that the EEPROM write operation is completed before entering Power-down.

Preventing EEPROM Corruption

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

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

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

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

I/O Memory

The I/O space definition of the ATmega8535 is shown in page 296.

All ATmega8535 I/Os and peripherals are placed in the I/O space. The I/O locations are accessed by the IN and OUT instructions, transferring data between the 32 general purpose working registers and the I/O space. I/O Registers within the address range 0x00 - 0x1F are directly bit-accessible using the SBI and CBI instructions. In these registers, the value of single bits can be checked by using the SBIS and SBIC instructions. Refer to the instruction set section for more details. When using the I/O specific commands IN and OUT, the I/O addresses 0x00 - 0x3F must be used. When addressing I/O Registers as data space using LD and ST instructions, 0x20 must be added to these addresses.

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

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

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

Analog-to-Digital Converter

Features

- 10-bit Resolution
- 0.5 LSB Integral Non-linearity
- ±2 LSB Absolute Accuracy
- 65 - 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 channel are not tested for devices in PDIP and PLCC Package. This feature is only guaranteed to work for devices in TQFP and MLF Packages.

The ATmega8535 features a 10-bit successive approximation ADC. The ADC is connected to an 8-channel Analog Multiplexer which allows eight single-ended voltage inputs constructed from the pins of Port A. The single-ended voltage inputs refer to 0V (GND).

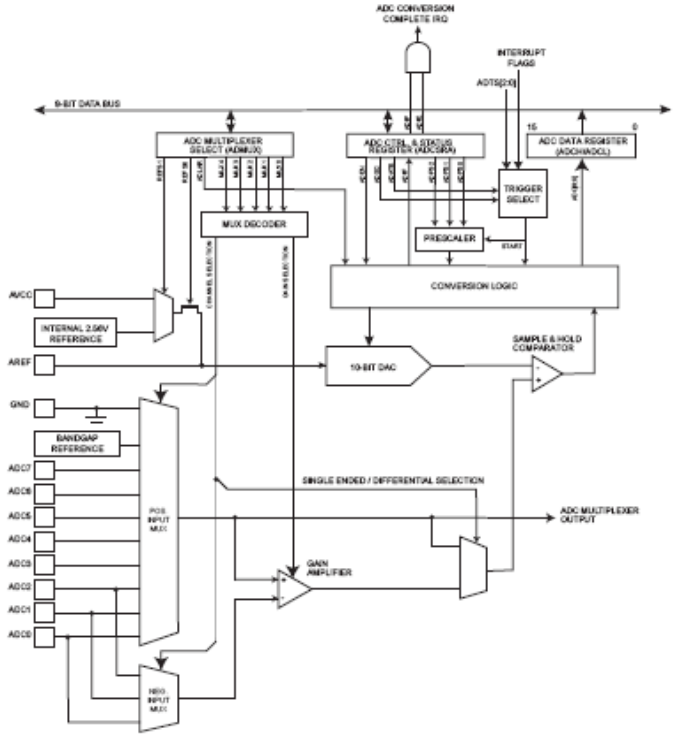
The device also supports 16 differential voltage input combinations. Two of the differential inputs (ADC1, ADC0 and ADC3, ADC2) are equipped with a programmable gain stage, providing amplification steps of 0 dB (1x), 20 dB (10x), or 46 dB (200x) on the differential input voltage before the A/D conversion. 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 ±0.3V from V_{CC}. See the paragraph "ADC Noise Canceler" on page 211 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



Operation

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.

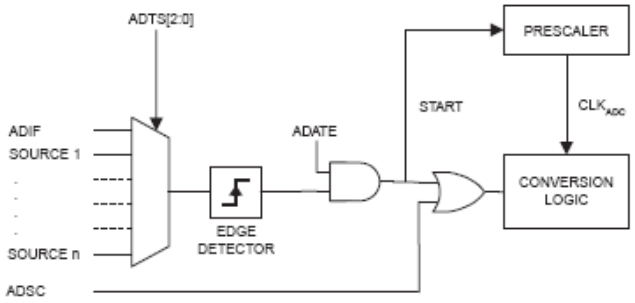
The ADC has its own interrupt which can be triggered when a conversion completes. When ADC access to the data registers is prohibited between reading of ADCH and ADCL, the interrupt will trigger even if the result is lost.

Starting 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

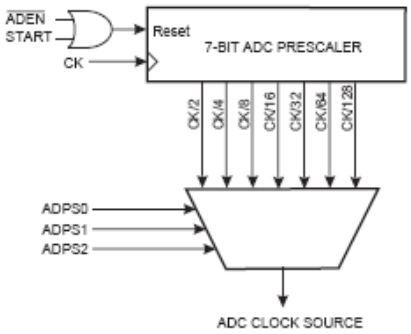


Using the ADC Interrupt Flag as a trigger source makes the ADC start a new conversion as soon as the ongoing conversion has finished. The ADC then operates in Free Running mode, constantly sampling and updating the ADC Data Register. The first conversion must be started by writing a logical one to the 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.

Prescaling and Conversion 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 209 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 an 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 two ADC clock cycles after the rising edge on the trigger source signal. Three additional CPU clock cycles are used for synchronization logic.

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 82.

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

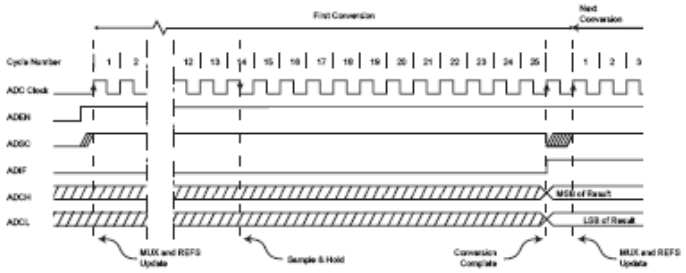


Figure 102. ADC Timing Diagram, Single Conversion

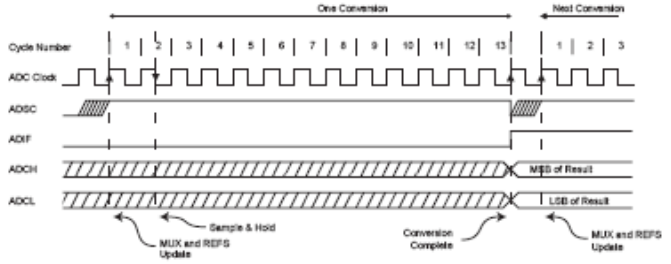


Figure 103. ADC Timing Diagram, Auto Triggered Conversion

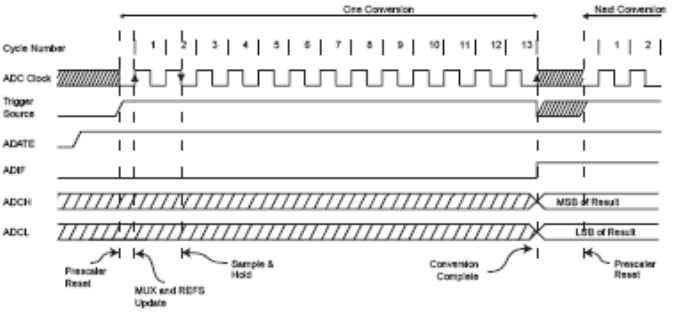
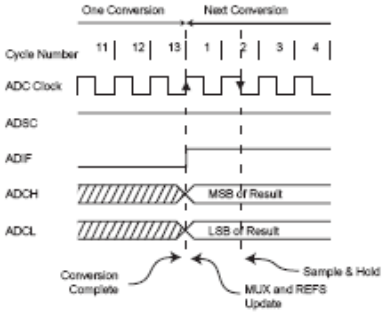


Figure 104. ADC Timing Diagram, Free Running Conversion



National Semiconductor

December 1995

LF351 Wide Bandwidth JFET Input Operational Amplifier

General Description

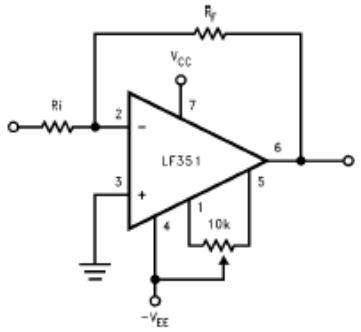
The LF351 is a low cost high speed JFET input operational amplifier with an internally trimmed input offset voltage (BI-FET II™ technology). The device requires a low supply current and yet maintains a large gain bandwidth product and a fast slew rate. In addition, well matched high voltage JFET input devices provide very low input bias and offset currents. The LF351 is pin compatible with the standard LM741 and uses the same offset voltage adjustment circuitry. This feature allows designers to immediately upgrade the overall performance of existing LM741 designs.

The LF351 may be used in applications such as high speed integrators, fast D/A converters, sample-and-hold circuits and many other circuits requiring low input offset voltage, low input bias current, high input impedance, high slew rate and wide bandwidth. The device has low noise and offset voltage drift, but for applications where these requirements are critical, the LF356 is recommended. If maximum supply current is important, however, the LF351 is the better choice.

Features

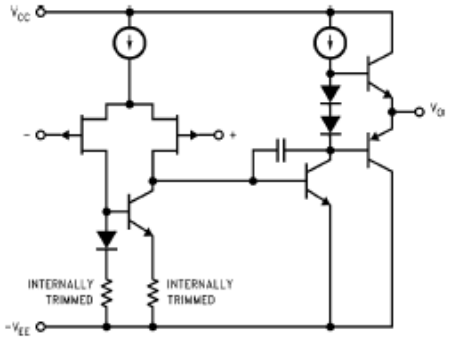
■ Internally trimmed offset voltage	10 mV
■ Low input bias current	50 pA
■ Low input noise voltage	25 nV/√Hz
■ Low input noise current	0.01 pA/√Hz
■ Wide gain bandwidth	4 MHz
■ High slew rate	13 V/μs
■ Low supply current	1.8 mA
■ High input impedance	$10^{12}\Omega$
■ Low total harmonic distortion $A_V = 10$, $R_L = 10k$, $V_O = 20$ Vp-p, BW = 20 Hz–20 kHz	<0.02%
■ Low 1/f noise corner	50 Hz
■ Fast settling time to 0.01%	2 μs

Typical Connection



TL/H/5648–11

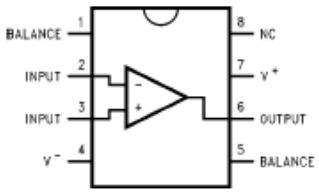
Simplified Schematic



TL/H/5648–12

Connection Diagrams

Dual-In-Line Package



TL/H/5648–13

Order Number LF351M or LF351N
See NS Package Number M08A or N08E

Absolute Maximum Ratings

If Military/Aerospace specified devices are required, please contact the National Semiconductor Sales Office/Distributors for availability and specifications.

Supply Voltage	±18V
Power Dissipation (Notes 1 and 6)	670 mW
Operating Temperature Range	0°C to +70°C
$T_{j(MAX)}$	115°C
Differential Input Voltage	±30V
Input Voltage Range (Note 2)	±15V
Output Short Circuit Duration	Continuous
Storage Temperature Range	−65°C to +150°C
Lead Temp. (Soldering, 10 sec.)	
Metal Can	300°C
DIP	260°C
θ_{jA}	
N Package	120°C/W
M Package	TBD
Soldering Information	
Dual-In-Line Package	
Soldering (10 sec.)	260°C
Small Outline Package	
Vapor Phase (60 sec.)	215°C
Infrared (15 sec.)	220°C

See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" for other methods of soldering surface mount devices.

ESD rating to be determined.

DC Electrical Characteristics (Note 3)

Symbol	Parameter	Conditions	LF351			Units
			Min	Typ	Max	
V_{OS}	Input Offset Voltage	R_S = 10 kΩ, T_A = 25°C Over Temperature		5	10 13	mV mV
$\Delta V_{OS}/\Delta T$	Average TC of Input Offset Voltage	R_S = 10 kΩ		10		µV/°C
I_{OS}	Input Offset Current	T_j = 25°C, (Notes 3, 4) $T_j \leq$ 70°C		25	100 4	pA nA
I_B	Input Bias Current	T_j = 25°C, (Notes 3, 4) $T_j \leq$ 70°C		50	200 8	pA nA
R_{IN}	Input Resistance	T_j = 25°C		10^{12}		Ω
A_{VOL}	Large Signal Voltage Gain	V_S = ±15V, T_A = 25°C V_O = ±10V, R_L = 2 kΩ Over Temperature	25 15	100		V/mV V/mV
V_O	Output Voltage Swing	V_S = ±15V, R_L = 10 kΩ	±12	±13.5		V
V_{CM}	Input Common-Mode Voltage Range	V_S = ±15V	±11	+15 −12		V V
CMRR	Common-Mode Rejection Ratio	$R_S \leq$ 10 kΩ	70	100		dB
PSRR	Supply Voltage Rejection Ratio	(Note 5)	70	100		dB
I_S	Supply Current			1.8	3.4	mA

AC Electrical Characteristics (Note 3)

Symbol	Parameter	Conditions	LF351			Units
			Min	Typ	Max	
SR	Slew Rate	$V_S = \pm 15V$, $T_A = 25°C$		13		V/μs
GBW	Gain Bandwidth Product	$V_S = \pm 15V$, $T_A = 25°C$		4		MHz
e_n	Equivalent Input Noise Voltage	$T_A = 25°C$, $R_S = 100\Omega$, $f = 1000$ Hz		25		$nV/\sqrt{Hz}$
i_n	Equivalent Input Noise Current	$T_j = 25°C$, $f = 1000$ Hz		0.01		$pA/\sqrt{Hz}$

Note 1: For operating at elevated temperature, the device must be derated based on the thermal resistance, θ_{jA}.

Note 2: Unless otherwise specified the absolute maximum negative input voltage is equal to the negative power supply voltage.

Note 3: These specifications apply for $V_S = \pm 15V$ and $0°C \le T_A \le +70°C$. V_{OS}, I_B and I_{OS} are measured at $V_{CM} = 0$.

Note 4: The input bias currents are junction leakage currents which approximately double for every 10°C increase in the junction temperature, T_j. Due to the limited production test time, the input bias currents measured are correlated to junction temperature. In normal operation the junction temperature rises above the ambient temperature as a result of internal power dissipation, P_D. $T_j = T_A + \theta_{jA} P_D$ where θ_{jA} is the thermal resistance from junction to ambient. Use of a heat sink is recommended if input bias current is to be kept to a minimum.

Note 5: Supply voltage rejection ratio is measured for both supply magnitudes increasing or decreasing simultaneously in accordance with common practice. From ±15V to ±5V.

Note 6: Max. Power Dissipation is defined by the package characteristics. Operating the part near the Max. Power Dissipation may cause the part to operate outside guaranteed limits.