FINAL PROJECT – TE090362

SIMULATOR OF TEMPERATURE CONTROL IN GREENHOUSE

Dwi Febria Herdiana
NRP 2210 030 046
Guntur Rangga Kurniawan
NRP 2210 030 079

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

Electrical Engineering D3 Programme
Industrial Technology Faculty
Sepuluh Nopember *Institute of Technology*
Surabaya *2014*

FINAL PROJECT– **TE090362**

SIMULATOR OF TEMPERATURE CONTROL IN GREEN HOUSE

Dwi Febria Herdiana
NRP 2210 030 046
Guntur Rangga Kurniawan
NRP 2210 030 079

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

Electrical Engineering D3 Programme
Industrial Technology Faculty
Sepuluh Nopember *Institute of Technology*
Surabaya *2014*

FINAL PROJECT – TE090362

SIMULATOR OF TEMPERATURE CONTROL IN GREENHOUSE

Dwi Febria Herdiana
NRP 2210 030 046
Guntur Rangga Kurniawan
NRP 2210 030 079

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

Electrical Engineering D3 Programme
Industrial Technology Faculty
Sepuluh Nopember *Institute of Technology*
Surabaya *2014*

FINAL PROJECT– **TE090362**

SIMULATOR OF TEMPERATURE CONTROL IN GREEN HOUSE

Dwi Febria Herdiana
NRP 2210 030 046
Guntur Rangga Kurniawan
NRP 2210 030 079

Supervisor
Ir. Josaphat Pramudijanto, M.Eng.

Electrical Engineering D3 Programme
Industrial Technology Faculty
Sepuluh Nopember *Institute of Technology*
Surabaya *2014*

PERANCANGAN *VIRTUAL PLANT* TANGKI PENYIMPANAN AIR YANG DIKONTROL MENGGUNAKAN PLC

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan
Memperoleh Gelar Ahli Madya Teknik
Pada
Bidang Studi Komputer Kontrol
Program Studi D3 Teknik Elektro
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Menyetujui :
Dosen Pembimbing,

Ir. Josaphat Pramudijanto, M.Eng
NIP. 19621005 199003 1 003

SURABAYA
JANUARI, 2014

SIMULATOR KONTROL TEMPERATUR DALAM RUMAH KACA

Nama Mahasiswa : Dwi Febria Herdiana
NRP : 2210 030 046
Nama Mahasiswa : Guntur Rangga Kurniawan
NRP : 2210 030 079
Dosen Pembimbing : Ir. Josaphat Pramudijanto, M.Eng.
NIP : 19621005 199003 1 003

ABSTRAK

Rumah kaca adalah sebuah bangunan tempat tanaman dibudidayakan yang terbuat dari gelas atau plastik. Sinar matahari mudah masuk, sehingga menghasilkan energi panas yang berguna bagi tumbuhan, tanah, dan barang lainnya di dalam bangunan. Rumah kaca menjadi penting dalam penyediaan makanan di negara garis lintang tinggi. Cahaya dan suhu yang mudah diatur dapat mengubah tanah tidak subur menjadi subur.

Virtual plant merupakan suatu metode yang dapat memodelkan suatu *plant* nyata. Dalam pembuatan *virtual plant*, dibutuhkan *software* pemodelan objek. Salah satu *software* yang cukup baik digunakan dalam pemodelan yaitu labview. *Interface* diperlukan agar perangkat satu dengan perangkat yang lain mampu berkomunikasi. Komunikasi dari labview ke kontroler umumnya menggunakan mikrokontroler arduino untuk *mapping input*/*output*. Sementara mikrokontroler digunakan sebagai *driver* proses dari sistem yang digunakan.

Manfaat dari pembuatan *virtual plant* ini yaitu memudahkan pembelajaran dalam menerapkan alat – alat industri dengan *plant* yang mudah dibuat dan juga menghemat biaya karena kita tidak perlu memiliki *plant* nyata. Hal ini menjadikan mahasiswa teknik dapat memahami lingkungan kerja di industri sebenarnya sehingga tidak canggung ketika suatu saat nanti terjun langsung di dunia kerja.

Kata Kunci : *Virtual Plant, Arduino, Labview*

Halaman ini sengaja dikosongkan

SIMULATOR TEMPERATURE CONTROL IN GREENHOUSE

Name Student : Dwi Febria Herdiana
NRP : 2210 030 046
Name of Student : Guntur Rangga Kurniawan
NRP : 2210 030 079
Supervisor : Ir. Josaphat Pramudijanto, M.Eng.
NIP : 19621005 199003 1003

ABSTRACT

Greenhouse is a building where plants are cultivated are made of glass or plastic. Easy entry of sunlight, thus producing useful heat energy to plants, soil, and other items inside the building. Greenhouse becomes important in the food supply of high latitude countries. Light and temperature are easily set can change into fertile soil infertile.

Virtual plant is a method that can model a real *plant.* In the creation of *virtual plant,* object modeling *software* required. Enough One of the *software* that is best used in modeling labview. *I nterface* needed to use one with the other device capable of communicating. Communication of contro l ler generally abview to me nggunakan microcontroller Arduino for *mapping input* / *output.* While the PLC is used as the *drivers* of the system used.

The benefit of this is *the* creation of *virtual plant* facilitate learning in applying the tool - the tool industry with a *plant* that is easy to make and also save costs because we do not need to have a real *plant.* This makes the engineering students can understand the actual working environment in the industry so it's not awkward when one day directly involved in the world of work.

Keywords: *Virtual Plant, Arduino, Labview*

Halaman ini sengaja dikosongkan

SIMULATOR OF TEMPERATURE CONTROL IN GREENHOUSE

Name Student : Dwi Febria Herdiana
NRP : 2210 030 046
Name of Student : Guntur Rangga Kurniawan
NRP : 2210 030 079
Supervisor : Ir. Josaphat Pramudijanto, M.Eng.
NIP : 19621005 199003 1003

ABSTRACT

Greenhouse is a building where plants are cultivated are made of glass or plastic. Easy entry of sunlight, thus producing useful heat energy to plants, soil, and other items inside the building. Greenhouse becomes important in the food supply of high latitude countries. Light and temperature are easily set can change into fertile soil infertile.

Virtual plant is a method that can model a real plant. In the creation of virtual plant, object modeling software required. Enough One of the software that is best used in modeling labview. I nterface needed to use one with the other device capable of communicating. Communication of contro l ler generally abview to me nggunakan microcontroller Arduino for mapping input / output. While the PLC is used as the drivers of the system used.

The benefit of this is the creation of virtual plant facilitate learning in applying the tool - the tool industry with a plant that is easy to make and also save costs because we do not need to have a real plant. This makes the engineering students can understand the actual working environment in the industry so it's not awkward when one day directly involved in the world of work.

Halaman ini sengaja dikosongkan

KATA PENGANTAR

Alhamdulillah, puji syukur kami panjatkan kepada Allah Subhanahu Wa Ta'ala yang telah memberikan rahmat dan hidayah-Nya. Dan tak lupa kami ucapkan shalawat serta salam kepada Rasulullah Shalallahu 'Alaihi Wassalam sehingga kami dapat menyelesaikan Tugas Akhir kami yang berjudul :

"Simulator Kontrol Temperatur Dalam Rumah Kaca"

Dalam penyusunan laporan Tugas Akhir ini, kami banyak mendapatkan bantuan dan dukungan dari berbagai pihak. Oleh karena itu, penulis dengan tulus ikhlas menyampaikan banyak terima kasih kepada Ayah dan Ibunda kedua penulis yang telah memberikan dukungan moral, material, serta doa. Bapak Josaphat Pramudijanto, Ir., M.Eng, selaku dosen pembimbing Tugas Akhir kami, atas segala kesabaran dan kesediaannya meluangkan waktu untuk membimbing serta memberi dukungan sehingga Tugas Akhir ini dapat terselesaikan.Bapak Eko Setijadi, S.T, M.T, Ph.D selaku Kaprodi D3 Teknik Elektro, Komputer Kontrol FTI - ITS.Semua pihak yang tidak dapat kami sebutkan satu persatu yang telah memberi dorongan dan bantuan dalam menyelesaikan Tugas Akhir ini baik secara langsung maupun tidak langsung.

Kami menyadari bahwa pembuatan laporan serta Tugas Akhir kami belum sempurna, karena kesempurnaan hanyalah milik Allah Subhanahu Wa Ta'ala. Untuk itu sekiranya mohon maaf atas kekilafan kami apabila terdapat kesalahan dalam pembuatan laporan ini. Besar harapan kami untuk memaafkan kurang sempurnanya pembuatan laporan Tugas Akhir ini.

Surabaya, Januari 2014

Penyusun

Halaman ini sengaja dikosongkan

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB I
PENDAHULUAN

1.1 Latar Belakang

Rumah kaca adalah sebuah bangunan tempat tanaman dibudidayakan yang terbuat dari gelas atau plastik. Sinar matahari mudah masuk, sehingga menghasilkan energi panas yang berguna bagi tumbuhan, tanah, dan barang lainnya di dalam bangunan. Sayuran dan bunga juga dikembangkan di rumah kaca pada akhir musim dingin atau awal musim semi, yang kemudian dipindahkan ke luar begitu cuaca menjadi hangat. Rumah kaca menjadi penting dalam penyediaan makanan di negara garis lintang tinggi. Rumah kaca melindungi tanaman dari panas dan dingin yang berlebihan, melindungi tanaman dari badai debu dan terhindar dari hama. Cahaya dan suhu yang mudah diatur dapat mengubah tanah tidak subur menjadi subur. Manfaat lainnya ialah memberikan persediaan makanan bagi negara yang kekurangan bahan pangan karena iklim yang tidak menentu.

Ruangan yang tertutup dari rumah kaca mempunyai kebutuhan yang unik, dibandingkan dengan yang ada di luar ruangan. Hama, penyakit, *temperature*, harus dikontrol, dan irigasi pun tetap dibutuhkan untuk menyediakan air. Sistem Kontrol Suhu dalam rumah kaca adalah sebuah sistem yang dapat mendeteksi kondisi temperature lingkungan dalam rumah kaca. Setelah mendeteksi, sistem ini mampu merespons kondisi tersebut agar kembali pada kondisi optimal yang telah ditentukan sebelumnya. Pengaturan suhu ini bertujuan agar tanaman didalam rumah kaca dapat tumbuh dengan baik sesuai dengan temperature yang dibutuhkan tanaman dalam lingkungan tumbuh yang alami. Pembuatan sistem ini dapat memakan biaya yang tinggi. Untuk itu kami membuat *simulator*, sebagai bentuk simulasi dari sistem kontrol suhu yang memiliki respons mirip seperti sistem yang sebenarnya.

Dari banyak peristiwa, sistem kontrol suhu dalam rumah kaca dilakukan secara manual atau semi otomatis dan tanpa penggunaan HMI (*Human Machine Interface)* sebagai alat pantau jarak jauh. Hal ini dapat menyebabkan kerusakan tanaman karena penanganan yang lambat oleh

operatornya. Dengan pembuatan *simulator* pengaturan suhu dalam rumah kaca, dapat dipelajari bagaimana membuat sistem pengaturan yang baik yang dapat dimonitor melalui komputer dan dapat dikendalikan dengan ATMega.

1.2 Perumusan Masalah

1. Perbedaan curah hujan di Indonesia menyebabkan temperatur udara tidak konsisten.
2. Sistem pertanian di Indonesia belum banyak menggunakan sistem otomatis pengaturan suhu.

1.3 Batasan Masalah

1. Data temperatur dalam rumah kaca berupa data *real* yang diperoleh dari sensor LM35**.**
2. *Virtual plan* rumah kaca terdiri dari virtual sistem kontrol temperatur otomatis serta sistem virtual buka/tutup valve untuk penyiraman tanaman**.**
3. Suhu minimum dalam rumah kaca virtual adalah 25°C dan suhu maksimum 32°C**.**
4. Menggunakan ATMega 16 dan Arduino MEGA 2560 yang saling berinteraksi untuk mengontrol temperatur pada virtual plant.
5. Jenis sensor virtual yang digunakan adalah sensor temperatur yang menggunakan indikator digital pembacaan temperatur**.**
6. Sistem penyiraman virtual menggunakan tangki dan valve.
7. Data temperatur dalam satu kali simulasi ditampilkan sebagai report dalam Ms. Excell.

1.4 Tujuan

1. Membuat simulasi sistem kontrol otomatis untuk mengatur temperatur pada rumah kaca.
2. Membuat software untuk mengendalikan objek – objek simulasi dengan mikrokontroler.

1.5 Sistematika Laporan

Dalam penyusunan buku Tugas Akhir ini, pembahasan mengenai sistem alat yang dibuat dibagi menjadi lima bab dengan sistematika sebagai berikut :

1. **BAB I PENDAHULUAN**
   Mendiskripsikan tentang latar belakang, perumusan masalah, batasan masalah, tujuan, sistematika penulisan, serta relevansi yang berkaitan dengan judul Tugas Akhir ini.
2. **BAB II TEORI PENUNJANG**
   Berisi penjelasan dasar teori mengenai konsep yang digunakan dalam Perancangan *Virtual Plant* simulator rumah kaca yang Dikontrol Menggunakan *Mikrokontroller* ini. Materi meliputi alat alat *hardware* serta *software*. *Hardware* antara lain mikrokontroler arduino,Mikrokontroller ATMega 16. *Software* yang digunakan yaitu Labview, IDE (*Intregated Development Environment*), dan AVR.
3. **BAB III PERANCANGAN DAN PEMBUATAN ALAT**
   Pembahasan secara detail tentang perancangan *Virtual Plant* baik *hardware* maupun *software* seperti perancangan regulator (*power supply*, serta pembuatan program baik dari mikrokontroler arduino, labview maupun mikrokontroller ATMega 16.
4. **BAB IV PENGUKURAN DAN ANALISA**
   Berisi data - data pengujian alat pada *hardware* dan *software* secara keseluruhan beserta analisanya. Data yang diukur yaitu tegangan pada mikrokontroler arduino. Pengujian yang dilakukan antara lain pengujian terhadap , data array pada labview, dan pemrograman AVR.
5. **BAB V PENUTUP**
   Berisi kesimpulan yang diperoleh dari pembuatan Tugas Akhir ini dan saran - saran untuk pengembangan alat selanjutnya.

1.6 Relevansi

Pembuatan simulasi sistem kontrol suhu dalam rumah kaca ini dapat bermanfaat untuk bahan referensi Tugas Akhir bagi kalangan mahasiswa bidang Teknik Elektro, sebagai referensi pembelajaran bagi akademisi, dan kemajuan teknologi industri pada umunya.

Halaman ini sengaja dikosongkan

BAB II
TEORI PENUNJANG

Dalam bab ini, akan dijelaskan mengenai peralatan-peralatan pendukung yang berkaiatan dengan Perancangan *Virtual Plant*Rumah Kaca Dikontrol menggunakanmikrokontroler dan perangkat keras (*hardware)* antara lain Arduino kitMEGA 2560, *mikrokontroler*, serta perangkat lunak (*software)*sepertiLabView, AVR dan IDE (*Integrated Development Environment*).

2.1 Teori Rumah Kaca

Rumah kaca (disebut juga rumah hijau dan rumah tanaman) adalah sebuah bangunan di mana tanaman dibudidayakan. Sebuah rumah kaca terbuat dari gelas atau plastik; Dia menjadi panas karena radiasi elektromagnetik yang datang dari matahari memanaskan tumbuhan, tanah, dan barang lainnya di dalam bangunan ini. Kaca yang digunakan untuk rumah kerja bekerja sebagai medium transmisi yang dapat memilih frekuensi spektral yang berbeda-beda, dan efeknya adalah untuk menangkap energi di dalam rumah kaca, yang memanaskan tumbuhan dan tanah di dalamnya yang juga memanaskan udara dekat tanah dan udara ini dicegah naik ke atas dan mengalir keluar. Oleh karena itu rumah kaca bekerja dengan menangkap radiasi elektromagnetik dan mencegah konveksi. Rumah kaca sering kali digunakan untuk mengembangkan bunga, buah dan tanaman tembakau. Lebah dari *genusBombus* adalah *polinator* pilihan untuk banyak polinasi rumah kaca, meskipun tipe lebah lain juga digunakan, dan juga polinasi buatan. Banyak sayuran dan bunga yang dikembangkan di rumah kaca pada akhir musim dingin atau awal musim semi, yang kemudian dipindahkan ke luar begitu cuaca menjadi hangat. Ruangan yang tertutup dari rumah kaca mempunyai kebutuhan yang unik, dibandingkan dengan produksi luar ruangan. Hama dan penyakit, dan panas tinggi dan kelembaban, harus dikontrol, dan irigasi dibutuhkan untuk menyediakan air.

Rumah kaca menjadi penting dalam penyediaan makanan di negara garis lintang tinggi. Rumah kaca melindungi tanaman dari panas dan dingin yang berlebihan, melindungi tanaman dari badai debu dan "*blizzard*", dan menolong mencegah hama. Pengontrolan cahaya dan temperatur dapat mengubah tanah tak subur menjadi subur. Rumah kaca dapat memberikan suatu negara persediaan bahan makanan, di mana

tanaman tak dapat tumbuh karena keganasan lingkungan. Hidroponik dapat digunakan dalam rumah kaca untuk menggunakan ruang secara efektif.

2.2 Teori Simulasi

Simulasi merupakan proses yang diperlukan untuk operasionalisasi model, atau penanganan model untuk meniru tingkah laku sistem yang sesungguhnya. Dalam metode simulasi kita dapat memperkirakan dampak dari suatu keputusan yang diambil, tetapi harus diketahui dimana dan kapan simulasi ini dapat diterapkan.

Jadi simulasi adalah tindakan menggunakan model. Kemudian dirancang skenario percobaan guna mendapatkan hasil simulasi yang kelak diolah menjadi jawaban atas sistem nyatanya. Simulasi dapat memperkirakan dampak dari suatu keputusan yang diambil. Meskipun metode simulasi sangat menjanjikan, tetapi harus diketahui dimana dan kapan simulasi ini dapat diterapkan. Keuntungan menggunakan simulasi ialah sebagai berikut:

a. Simulasi merupakan salah satu metode yang mampu memberikan perkiraan sistem yang lebih nyata sesuai kondisi operasional dari kumpulan pekerjaan.
b. Sebagai alternatif desain yang diusulkan atau alternatif terhadap kebijakan dari operasional yang mempu memberikan pelayanan terbaik terhadap pokok kebutuhan yang diperlukan.
c. Memudahkan mengontrolan lebih banyak kondisi dari suatu percobaan sehingga dimungkinkan untuk dicoba diterapkan secara nyata padasistem itu.
d. Menyediakan sarana untuk mempelajari system dalam waktu yang lebih singkat, sehingga menghemat biaya.
e. Dapat dihentikan dan dijalankan kembali, tanpa menimbulkan permasalahan pada sistem.

Namun, dalam simulasi terdapat beberapa kekurangan sebagai berikut:

a. Simulasi tidak akurat, karna teknik ini bukan proses optimisasi dan tidak menghasilkan sebuah jawaban tetapi hanya menghasilkan sekumpulan output dari sistem pada berbagai kondisi yang berbeda. Dalam banyak kasus ketelitiannya sulit diukur.

b. Model simulasi yang baik sangat mahal, bahkan sering dibutuhkan waktu bertahun-tahun untuk mengembangkan model yang sesuai.
c. Tidak semua situasi dapat dievaluasi dengan simulasi.

2.3 Teori Pengendalian Temperatur Dalam Rumah Kaca [7]

Struktur rumah kaca menciptakan lingkungan yang terkendali untuk pertumbuhan tanaman. Salah satu fitur utama dari rumah kaca adalah temperatur. Rumah kaca melindungi tanaman dari dingin, angin, hujan , dan kondisi cuaca lain seperti salju, hujan es dan angin, serta memberikan cahaya dan kehangatan. Rumah kaca mudah mengumpulkan panas melalui jendela kaca dan dinding selama musim panas, dibandingkan selama musim gugur dan musim hujan. Melepaskan kelebihan panas saat cuaca panas ialah sebuah tantangan dalam membangun rumah kaca.

Apabila temperatur meningkat, penguapan pun meningkat. Disinilah peran *humidifier* untuk menarik udara panas dalam rumah kaca lalu menghembuskannya pada permukaan air pada wadah penampungan atau disebut *collecting bucket.* Setelah udara panas "dicuci" dengan air dingin, udara dingin dihembuskan didalam rumah kaca beserta dengan butiran-butiran kecil air (*fogging system*) untuk menjaga kelembaban dalam rumah kaca. Namun ketika temperatur turun, penguapan pun menurun. Bila penguapan kurang, kemungkinan akan terjadi pembusukan pada bagian-bagian tanaman, seperti akar, batang, dan buah karena kelembaban yang tinggi. Selain itu, udara yang lembab juga dapat memicu pertumbuhan lumut dan jamur. Untuk mengatasinya, diperlukan *dehumidifier*. *Dehumidifier* bekerja dengan mengubah molekul udara yang lembab menjadi tetesan air menggunakan koil pendingin dan kipas kecil. Ini terjadi akibat tekanan udara yang tinggi karena menurunya suhu udara. Kandungan air di udara mengental dan menjadi tetesan air yang jatuh pada wadah penampung kemudian mengubahnya menjadi hawa sejuk yang dihembuskan dalam rumah kaca.. Gambar 2.1 dan 2.2 dibawah ini adalah contoh humidifier dan dehumidifier:

Gambar 2.1 Humidifier

Gambar 2.2 Dehumidifier

2.4 ArduinoMEGA 2560[1]

Mikrokontroler adalah suatu *chip* berupa IC (*Integrated Circuit*) yang dapat menerima sinyal *input*, mengolahnya dan memberikan sinyal *output* sesuai dengan program yang diisikan ke dalamnya. Sinyal *input* mikrokontroler berasal dari sensor yang merupakan informasi dari lingkungan sedangkan sinyal *output* ditujukan kepada aktuator yang dapat memberikan efek ke lingkungan. Jadi secara sederhana mikrokontroler dapat diibaratkan sebagai otak dari suatu perangkat/produk yang mempu berinteraksi dengan lingkungan sekitarnya.

Mikrokontroler pada dasarnya adalah komputer dalam satu *chip*, yang di dalamnya terdapat mikroprosesor, memori, jalur *input / output*

(I/O) dan perangkat pelengkap lainnya. Kecepatan pengolahan data pada mikrokontroler lebih rendah jika dibandingkan dengan PC. Pada PC kecepatan mikroprosesor yang digunakan saat ini telah mencapai orde GHz, sedangkan kecepatan operasi mikrokontroler pada umumnya berkisar antara 1 – 16 MHz. Begitu juga kapasitas RAM dan ROM pada PC yang bisa mencapai orde *Gigabyte*, dibandingkan dengan mikrokontroler yang hanya berkisar pada orde *byte/Kbyte*.

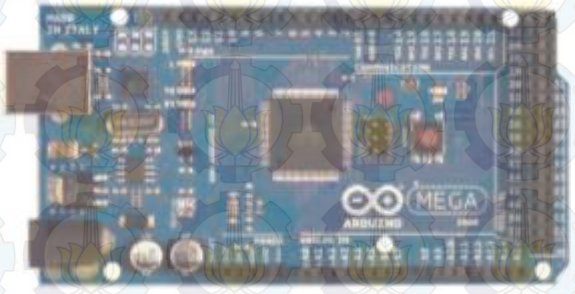
Arduino kitMega 2560 merupakan *platform* yang terdiri dari *software* dan *hardware*. *Hardware*Arduino kit sama dengan mikrokontroler pada umumnya hanya saja pada Arduino ditambahkan penamaan pin agar mudah diingat. *Software*Arduino merupakan *softwareopen source* sehingga dapat diunduh secara gratis. *Software* ini digunakan untuk membuat dan memasukkan program ke dalam Arduino. Pemrograman Arduino tidak sebanyak tahapan mikrokontroler konvensional karena Arduino sudah didesain mudah untuk dipelajari, sehingga para pemula dapat mulai belajar mikrokontroler dengan Arduino.

ArduinoMEGA 2560adalah *board* berbasis mikrokontroler ATmega2560. *Board* ini memiliki 54 digital *input / output* pin (dimana 14 pin dapat digunakan sebagai *output*PWM), 16 *inputanalog*, 16 MHz *osilator* kristal, koneksi USB, *jack* listrik tombol *reset*. ArduinoMEGA memiliki spesifikasi yang hampir sama dengan Arduino jenis lainnya seperti ArduinoDuemilanove dan duecimilia.mudah terkoneksi dengan kabel USB atau dengan adaptor AC to DC.*Board* serta spesifikasi lebih lanjut seperti pada Gambar 2.3 dan Tabel 2.1.

Tabel 2.1 Spesifikasi Arduino MEGA

No.	Deskripsi	Spesifikasi
1	Mikrokontroler	ATmega 2560
2	*Operating voltage*	5V
3	*Inputvoltage* (*recommended*)	7 – 12V
4	*Input voltage* (*maximum*)	6 – 20V
5	Digital *input* / *output* (I/O)	54 Pins (14 PWM)
6	Analog *input*	16 Pin
7	DC *current* per I/O	40mA
8	DC *current* 3,3V	50mA
9	*Flash memory* (ATmega 2560)	256 KB dengan 8 KB used by bootloader

10	SRAM	8KB (ATmega2560)
11	EEPROM	4KB (ATmega 2560)
12	*Clock Speed*	16 MHz



Gambar 2.3 ArduinoMEGA2560

2.5 MikrokontrolerATMega 16

Sering kita mendengar istilah mikrokomputer, mikroprosesor, dan mikrokontroler. Mikroprosesor adalah bagian CPU (*central processing unit*) dari sebuah komputer, tanpa memori, I/O, dan *peripheral*yang dibutuhkan oleh suatu sistem lengkap. Supaya dapat bekerja, Mikroprosesor memerlukan perangkat pendukung seperti RAM,ROM dan I/O.

Bila sebuah mikroprosesor dikombinasi dengan I/O dan memori (RAM/ROM) akan dihasilkan sebuah mikrokomputer. Sebagai terobosan mikrokomputer ini dapat juga dibuat dalam bentuk *single chip* yaitu *single chip microcomputer* (SCM) yang selanjutnya disebut sebagai mikrokontroler.

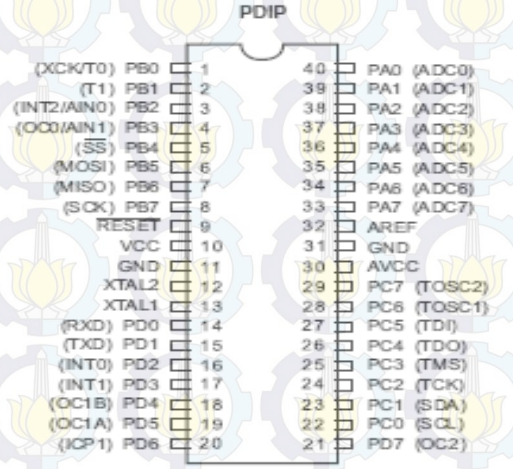
Perbedaan yang menonjol antara mikrokomputer dengan mikrokontroler (SCM) adalah pada penggunaan perangkat I/O dan media penyimpan program. Bila mikrokomputer menggunakan *disket* atau *hard drive* lainnya maka mikrokontroler menggunakan EPROM sebagai penyimpan programnya. Sedangkan keuntungan mikrokontroler dibandingkan dengan mikroprosesor adalah pada mikrokontroler sudah terdapat RAM dan peralatan I/O pendukung sehingga tidak perlu menambahkannya.

Deskripsi dari pin-pin ATmega16 adalah sebagai berikut :

1. VCC : *Supply* tegangan digital.

2. GND : *Ground*
3. *PORT* A : Menjalankan *analog input* ke A/D *converter*. *Port* A jugaberfungsi sebagai 8 *bitdirectional* I/O *port* jika A/D *converter* tidak digunakan. Ketika pin PA0-PA7 digunakan *input* dan secara eksternal *pull-low*, mereka seperti sumber arus jika *internalpull-up* resistor diaktifkan. Pin *port* A adalah *tri-states* ketika kondisi sebuah *reset* menjadi aktif, sekalipun *clock*-nya tidak jalan.
4. *PORT* B : *Port* B adalah 8 *bit directional* I/O *port* dengan *internal pull-up* resistor. *Buffer outputport* B ini mempunyai karakteristik *symmetrical drive* dengan kapabilitas *source* dan *sink* yang tinggi. Sebagai *input*, pin *port* B adalah eksternal *pull-low* seperti sumber arus jika*pull-up* resistor aktif. Pin *port* B adalah *tri-states* ketika kondisi sebuah *reset* menjadi aktif, sekalipun *clock*-nya tidak jalan.
5. *PORT* C : *Port* C adalah 8 *bit directional* I/O *port* dengan *internalpull-up* resistor. *Buffer outputport* C ini mempunyai karakteristik *symmetrical drive* dengan kapabilitas *source* dan *sink* yang tinggi. Sebagai *input*, pin *port* C adalah eksternal *pull-low* seperti sumber arus jika *pull-up* resistor aktif. Pin *port* C adalah *tri-states* ketika kondisi sebuah *reset* menjadi aktif, sekalipun *clock*-nya tidak jalan. Jika *interface* JTAG *enable*, *pull-up* resistor di pin PC5(TDI), PC3(TMS), dan PC2(TCK) akan aktif sekalipun *reset* terjadi.
6. *PORT* D :*Port* D adalah 8 *bit directional* I/O *port* dengan *internalpull-up* resistor. *Buffer outputport* D ini mempunyai karakteristik *symmetrical drive* dengan kapabilitas *source* dan *sink* yang tinggi. Sebagai *input*, pin *port* D adalah eksternal *pull-low* seperti sumber arus jika *pull-up* resistor aktif. Pin *port* D adalah *tri-states* ketika kondisi sebuah *reset* menjadi aktif, sekalipun *clock*-nya tidak jalan.
7. *RESET*: Sebuah *low level* pada pin akan lebih lama daripada lebar pulsa minimum akan menghasilkan *reset* meskipun *clock* tidak berjalan.
8. XTAL1 : *Inputinverting* penguat *oscilator* dan *input internal clock* operasi rangkaian.
9. XTAL2 : *Output* dari *inverting* penguat *oscilator*.

10. AVCC : Pin *supply* tegangan untuk *port* A dan A/D konverter. Sebaiknya eksternalnya dihubungkan ke VCC meskipun ADC tidak digunakan. Jika ADC digunakan, dihubungkan ke VCC melalui *low pas filter*.



Gambar 2.4 Pin Konfigurasi Mikrokontroler ATmega 16

2.6 Mikrokontroller ATMega8

Mikrokontroler ATMega8 merupakan bagian utama dari sistem kontrol, Mikrokontroler ini merupakan jenis mikrokontroler jenis AVR. Mikrokontoler jenis ini dipilih karena mikrokontroler ATMega8 memiliki 28 *port* masukan dan keluaran atau biasa disebut dengan *port* I/O yang dibagi Menjadi *port* B, C, dan D yang dapatdifungsikan sebagai masukan dan Sebagai keluaran sistem. Proses pengisian*(downloading)* program yang mudah karena meliliki fasilitas *in-system programming* yang sudah terdapat didalam ATMega8. Lima pin, MOSI, MISO, SCK, *Reset*, dan *Ground* digunakan untuk memrogram ATMega8 ini.

Mikrokontroler ATMega8 dapat bekerja apabila mendapat tegangan masukan sebesar 5 Volt dengan batas toleransi tegangan sebesar 5,4 Volt. Apabila tegangan masukan melebihi batas toleransi maka ATMega8 akan rusak dan tidak dapat digunakan kembali. Maka dari itu digunakan sistem catu daya teregulasi dengan penambahan transistor 2N3055 sebagai penyetabil tegangan agar batas tegangan

tidak melebihi dari batas toleransi. Proses pengisian pada mikrokontroler ATMega8 dapat mencapai seribukali proses *downloading*

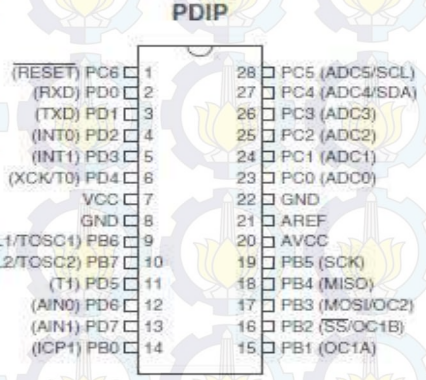
.

2.6.1.SpesifikasiMikrokontrolerATMega8

Setiap mikrokontroler memiliki jenis dan spesifikasi masing-Masing tergantung dari kegunaan dan kebutuhan dari mikrokontroler yang akan digunakan. ATMega8 memiliki spesifikasi antara lain: 1Kb internalSRAM, 8Kb *flashmemory*, 512 *bytes*EEPROM, 23jalur*input-output*, 8bit *timer/counter*,16bit *timer/counter*, 8,9,10 bitPWM, *On-chip analog comparator, fill duplex UART, SPI serial interface for in-system programming* dan *internal power reset.*

2.6.2.Konfigurasi dan Fungsi Kaki Pin ATMega8

Mikrokontroler ATmega8 memiliki kaki pin sebanyak 28 buah yang terdiri dari tiga buah port, yaitu port B, port C, dan port D, dan beberapa pin lain yang memiliki fungsi dan kegunaannya masing-masing. Gambar 2.3 menunjukkan fungsi dan jenis kaki pin dari mikrokontroler ATMega8



Gambar 2.5.KonfigurasiDanFungsiKakiPinATMega8

2.7 Pemrograman LabView

LabView (*Laboratory Virtual Instrumentation Engineering Workbench*) adalah sebuah *software* pemrograman yang diproduksi oleh *National Instruments* dengan konsep yang berbeda. Seperti bahasa pemrograman lainnya yaitu c++, matlab atau *visual basic*, LabView juga mempunyai fungsi dan peranan yang sama, perbedaannya bahwa

LabView menggunakan bahasa pemrograman berbasis grafis atau *block diagram* sementara bahasa pemrograman lainnya menggunakan basis teks. PemrogramanLabView dikenal dengan sebutan VI atau *Virtual Instruments* karena penampilan dan operasinya dapat meniru sebuah *instrument*. Pada LabView, *user* pertama kali membuat *user interface* atau *front panel* dengan menggunakan kontrol dan indikator, yang dimaksud dengan kontrol adalah *knobs, push button, dials* dan peralatan *input* lainnya sedangkan yang dimaksud dengan indikator adalah *graphs, LEDs*, dan peralatan *display* lainnya. Setelah menyusun *user interface*, lalu *user* menyusun *block diagram* yang berisi kode – kode untuk mengontrol *frontpanel*. *Software*LabView terdiri dari tiga komponen utama yaitu:

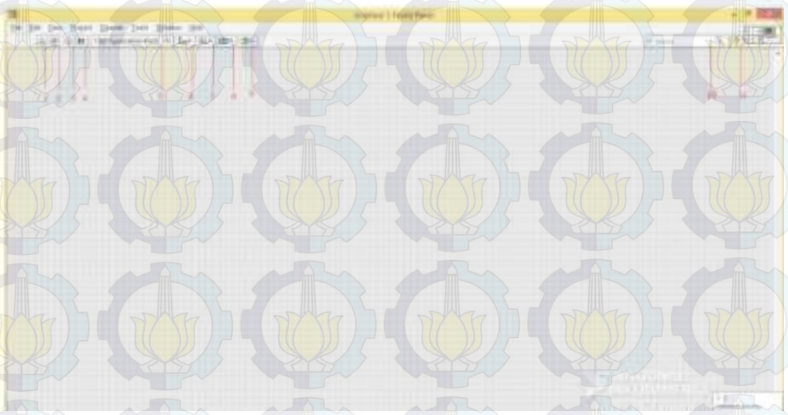
2.7.1 *Front Panel*

Front Panel adalah bagian *window* yang berlatar belakang abu-abu serta mengandung kontrol dan indikator. *Front Panel* digunakan untuk membangun sebuah VI, menjalankan program dan men-*debug* program. Gambar 2.6 merupakan tampilan *front panel*.

Penjelasan mengenai fungsi tiap tombol pada *front panel* antara lain :

1. ***Run***, berfungsi menjalankan VI (*Visual Instrument*s).
2. ***Run Continuosly***, berfungsi menjalankan VI secara terus menerus
3. ***Abort Execution***, berfungsi untuk menghentikan VI.
4. ***Pause***, berfungsi menghentikan VI sesaat.
5. ***Text Settings***, berfungsi untuk mengubah bentuk huruf, ukuran, perataan, dan warna pada teks.
6. ***Align Object***, berfungsi menata posisi beberapa objek supaya rata, termasuk rata kanan, rata kiri, rata atas, dan rata bawah.
7. ***Distribute Object***, berfungsi menata posisi antar objek dengan spasi yang sama.
8. ***Resize Object***, berfungsi membesarkan atau mengecilkan ukuran objek.
9. ***Reorder***, berfungsi menempatkan objek didepan atau di belakang objek lain.
10. ***Search***, befungsi mencari objek yang diinginkan pada *blockdiagram*.

11. ***Show Context Help Window***, berfungsi mengetahui informasi mengenai objek yang disentuh oleh pointer mouse



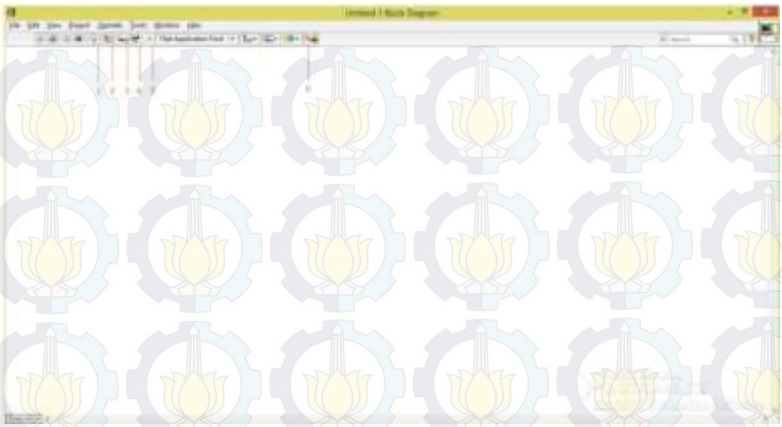
Gambar 2.6 Tampilan *Front Panel* pada Labview

2.7.2 Block Diagram dan VI

Block Diagram adalah bagian *window* yang berlatar belakang putih berisi *source code* yang dibuat dan berfungsi sebagai instruksi untuk *front panel.* Gambar 2.7 merupakan penampilan *block diagram.*

Penjelasan mengenai tombol pada *block diagram* antara lain :

1. ***Highlight Execution***, berfungsi melihat jalannya aliran data pada *block diagram*.
2. ***Retain Wire Values***, berfungsi menyimpan nilai datadi setiap titik ketika program dijalankan.
3. ***Step Into***, berfungsi menjalankan program tiap *step* (langkah) dari *node* ke *node*, sebuah *loop* atau *sub*VI.
4. ***Step Over***, berfungsi melompati *step* sebuah *loop* atau *sub*VI.
5. ***Step Out***, berfungsi keluar dari suatu *node* atau *loop* dan masuk ke *node* berikutnya.
6. ***Clean Up Diagram***, berfungsi merapikan garis dan menata ikon – ikon lebih ringkas dan menghemat ruang.



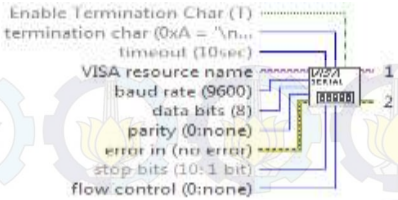
Gambar 2.7 Tampilan *BlockDiagram* pada LabView

Block – block yang digunakan pada Tugas Akhir ini antara lain :

1. ***VISA I/O Resource Name*** :
   Digunakan untuk mengatur COM serial yang dihubungkan antara LabView dengan perangkat keras yang dipakaiseperti padaGambar 2.8.

Gambar 2.8 *VISA I/O Resource Name*

2. ***VISA Configure Serial Port***
   Digunakan untuk mengatur *setting* komunikasi serial diawal sebelum komunikasi dijalankan seperti saluran yang digunakan (***VISA resource name***), kecepatan komunikasi (***baud rate***), jumlah data bit, stop bit, dan lain – lainseperti pada Gambar 2.9.



Gambar 2.9*VISA Configure Serial Port*

3. ***VISA Read***

Digunakan untuk membaca sejumlah *byte* data dari *hardwareinterface* yang ditentukan oleh ***VISA resource name***seperti yang terdapat pada Gambar 2.20.

Gambar 2.10*VISA Read*

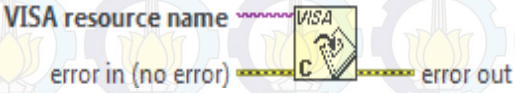
4. ***VISA Write***

Digunakan untuk menulis atau mengirimkan data dari *write buffer* ke suatu *hardware* yang ditentukan oleh ***VISA resource name***seperti yang terdapat pada Gambar 2.11.

Gambar 2.11*VISA Write*

5. ***VISA Close***

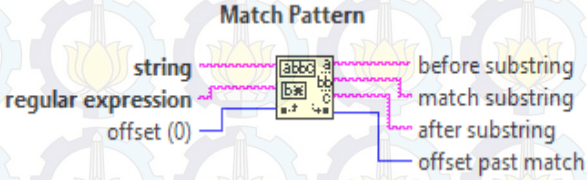
Digunakan untuk menutup komunikasi dengan *hardware* yang ditentukan oleh ***VISA resource name***seperti yang terdapat pada Gambar 2.12.



Gambar 2.12*VISA Close*

6. ***Match pattern***

Berfungsi mencari karakter atau substring pada sebuah string, dimulai dari nilai offset dan jika ditemukan, maka string tersebut dibagi menjadi 3, yaitu sebelum substring, pada sub string, sesudah substring, dan jumlah karakter dari offset ke substring tersebut dapat diketahui Gambar 2.13.



Gambar 2.13*Match Pattern*

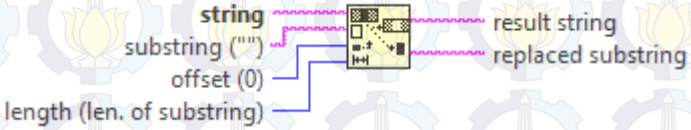
7. ***Fract/Exp string to number***

Fungsi ini digunakan untuk mengubah karakter angkadi dalam string menjadi *floatingpoint* numberGambar 2.14.

Gambar 2.14*Build Array*

8. ***Replace substring***

Berfungsi untuk menggantikan beberapa karakter di dalam string dalam karakter baru yang dimulai dari nilai offset hingga sepanjang nilai lenghtGambar 2.15.



Gambar 2.15 *Replace Substring*

9. ***Read JPG***

Berfungsi membaca file JPG dan membuat data tersebut di tampilakan ke control gambarGambar 2.16.

Gambar 2.16*Read JPG*

10. ***Draw Flattened Pixmap***

Berfungsi menggambarka 1, 4, 8 atau 24 bit RGB *pixmap* ke dalam gambarGambar 2.17.

Gambar 2.17*Draw Flattened Pixmap*

11. ***Equal***

Berfungsi membandingkan nilai antara x dan y. Ketika nilai x atau y sama, maka *output* akan dikeluarkanGambar 2.18.

Gambar 2.18*Equal*

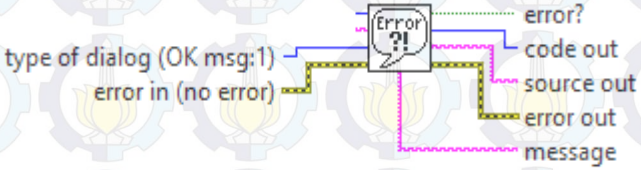
12. ***NOT Logic***

Berfungsi jika X bernilai "*FALSE*" maka fungsi yang dikeluarkan bernilai "*TRUE*" begitu sebaliknya jika X bernilai "*TRUE*" maka yang dikeluarkan bernilai "*FALSE*"seperti pada Gambar 2.19.

Gambar 2.19*NOT Logic*

13. ***Simple Handler Error***

Berfungsi sebagai sebuah indikator bila terjadi *error*. Jika terjadi *error* atau kesalahan akan muncul *box* diskripsi yang menunjukkan bahwa blok diagram terjadi kesalahan, seperti padaGambar 2.20.



Gambar 2.20*Simple Handler Error*

14. ***Quotient & Remainder***

Berfungsi mengubah *string* menjadi *arraystring* dalam hal ini berupa kode angka, seperti padaGambar 2.21.

Gambar 2.21*Quotient & Remainder*

15. ***Unbundle by Name***

Berfungsi untuk menguraikan *cluster* menjadi data – data dengan menampilkan nama label dari setiap data tersebut seperti padaGambar 2.22.

Gambar 2.22*Unbundle by Name*

16. ***Stop***

Berfungsi menghentikan program ketika ada *error* atau kesalahan, seperti pada Gambar 2.23.

Gambar 2.22*Stop*

17. ***Wait (ms)***

Berfungsi sebagai *delay* pada program selama beberapa milidetikGambar 2.24.

Gambar 2.24*Wait (ms)*

18. ***Indicator***

Berfungsi menampilkan status pada program, seperti padaGambar 2.25, Gambar 2.26, Gambar 2.26, Gambar 2.27.

Gambar 2.25*Lamp Indicator*

Gambar 2.26*Tank Indicator*

Gambar 2.27 *StopIndicator*

Gambar 2.28*Temperature Indicator*

19. ***Wire***

Berfungsi untuk mengubungkan ikon – ikon serta menunjukkan aliran dan tipe data, seperti padaGambar 2.29, Gambar 2.30,Gambar 2,31.

Gambar 2.29 Tipe Data Numerik

Gambar 2.30 Tipe Data *String*

Gambar 2.31 Tipe Data *Boolean*

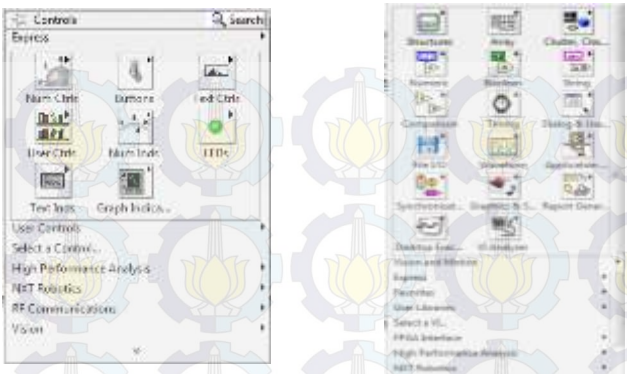
20. ***While loop***

Struktur while loop akan terus mengeksekusi ikon-ikon di dalam bloknya berulang-kali hingga terminal input conditional mendapat nilai FALSE,sebaliknya perulangan akan berhenti ketika mendapat nilai TRUE.

Gambar 2.32 Gambar While Loop

2.7.3 *Control* dan *Function Pallete*

Control dan *Function Pallete*seperti yang terdapat pada Gambar 2.33digunakan untuk membangun sebuah Vi. *Control Pallete* merupakan tempat beberapa kontrol dan indikator pada *front panel*, *control pallete* hanya tersedia di *front panel*, untuk menampilkan *control pallete* dapat dilakukan dengan mengklik *windows>>show controlpallete* atau klik kanan *front panel*. Sedangkan *function pallete*digunakan untuk membangun sebuah *block diagram*,*function pallete* hanya tersedia pada *block diagram*, untuk menampilkannya dapat dilakukan dengan mengklik *windows>>show control pallete* atau klik kanan pada lembar kerja *block diagram*.



(a) (b)

Gambar 2.33(a) *ControlPallete*(b) *Function Pallete*

2.8 IDE (*Integrated Development Environment*)

IDE merupakan *software* pemrograman pada Arduino. Pemrograman ini tergolong mudah dan mampu membuat pengguna lebih cepat dalam menguasai / memelajarinya. Tampilan awal pada IDE terdapat pada Gambar 2.34.

Gambar2.34 Tampilan Awal IDE Arduino

Keterangan mengenai tampilan IDE pada Gambar 2.33 adalah sebagai berikut :

1. ***Verify***, berfungsi menguji apakah ada kesalahan pada program atau *sketch*. Apabila *sketch* sudah benar, maka *sketch* tersebut akan dikompilasi. Kompilasi adalah proses mengubah kode program kedalam kode mesin.
2. ***Upload***, berfungsi mengirimkan kode mesin hasil kompilasi ke board Arduino.
3. ***New***, berfungsi membuka *sketch* baru.
4. ***Open***, berfungsi membka *sketch* yang sudah ada.
5. ***Save***, berfungsi menyimpan hasil program yang ditulis di *sketch*
6. ***Serial Monito***r, berfungsi menampilkan data yang dikirim dan diterima melalui komunikasi serial.
7. ***Sketch Page***, berfungsi sebagau tempat untuk menulis program.
8. ***Status Page***, berfungsi mengetahui status proses ketika program telah dikompilasi atau di-*upload*.

2.9 Sensor Temperatur LM35

IC LM 35 sebagai sensor temperatur yang teliti dan terkemas dalam bentuk *IntegratedCircuit* (IC), dimana output tegangan keluaran sangat linear berpadanan dengan perubahan temperatur. Sensor ini berfungsi sebagai pengubah dari besaran fisis temperatur ke besaran tegangan yang memiliki koefisien sebesar 10 mV /°C yang berarti bahwa kenaikan temperatur 1° C maka akan terjadi kenaikan tegangan sebesar 10 mV.

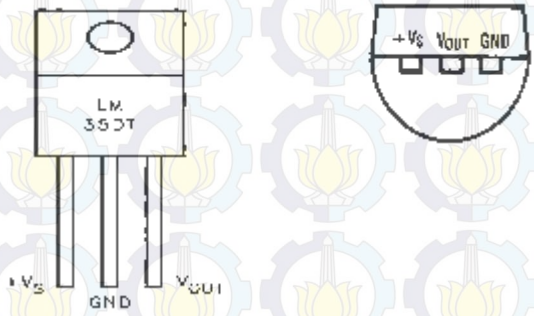
2.9.1 Pendahuluan

Sensor temperatur LM35 adalah komponen elektronika yang memiliki fungsi untuk mengubah besaran temperatur menjadi besaran listrik dalam bentuk tegangan. Sensor Temperatur LM35 yang dipakai dalam penelitian ini berupa komponen elektronika yang diproduksi oleh *NationalSemiconductor*. LM35 memiliki keakuratan tinggi dan kemudahan perancangan jika dibandingkan dengan sensor temperatur yang lain, LM35 juga mempunyai keluaran impedansi yang rendah dan linieritas yang tinggi sehingga dapat dengan mudah dihubungkan dengan rangkaian kendali khusus serta tidak memerlukan penyetelan lanjutan.

Meskipun tegangan sensor ini dapat mencapai 30 Volt akan tetapi yang diberikan kesensor adalah sebesar 5 Volt, sehingga dapat digunakan dengan catu daya tunggal dengan ketentuan bahwa LM35 hanya membutuhkan arus sebesar 60 µA hal ini berarti LM35 mempunyai kemampuan menghasilkan panas (*self-heating*) dari sensor yang dapat menyebabkan kesalahan pembacaan yang rendah yaitu kurang dari 0,5 ºC pada temperatur 25 ºC.

2.9.2 Struktur Sensor LM35

Pin LM35 menujukan fungsi masing-masing pin diantaranya, pin Vs berfungsi sebagai sumber tegangan kerja dari LM35, pin Vout atau tengah digunakan sebagai tegangan keluaran dengan jangkauan kerja dari 0 Volt sampai dengan 1,5 Volt dengan tegangan operasi sensor LM35 yang dapat digunakan antar 4 Volt sampai 30 Volt. Gambar dibawah menunjukan bentuk dari LM35 tampak depan dan tampak bawah



Gambar 2.35 Struktur Kaki LM35

. Keluaran sensor ini akan naik sebesar 10 mV setiap derajat*celcius* sehingga diperoleh persamaan sebagai berikut :

$$V_{LM35} = \text{Temperatur} \times 10\ mV$$

Di bawah ini adalah Karakteristik Sensor LM35.sebagai berikut:

1. Memiliki sensitivitas temperatur, dengan faktor skala linier antara tegangan dan temperatur 10 mVolt/ºC, sehingga dapat dikalibrasi langsung dalam *celcius.*

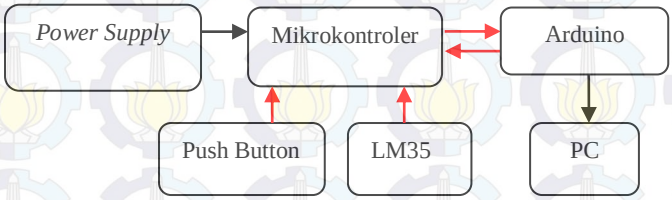
2. Memiliki ketepatan atau akurasi kalibrasi yaitu 0,5ºC pada temperatur 25 ºC.
3. Memiliki jangkauan maksimal operasi temperatur antara -55 ºC sampai +150 ºC.
4. Bekerja pada tegangan 4 sampai 30 Volt.
5. Memiliki arus rendah yaitu kurang dari 60 µA.
6. Memiliki pemanasan sendiri yang rendah (*low-heating*) yaitu kurang dari 0,1 ºC pada udara diam.
7. Memiliki impedansi keluaran yang rendah yaitu 0,1 W untuk beban 1 mA.
8. Memiliki ketidaklinieran hanya sekitar ± ¼ ºC.

BAB III
PERANCANGAN DAN PEMBUATAN ALAT

Pada perancangan *Simulator* Kontrol Temperatur Dalam Rumah Kaca yang dikontrol Menggunakan terdapat 2 tahap yang cukup penting, yaitu perancangan perangkat keras (*hardware*) dan perangkat lunak (*software*). Perancangan perangkat keras meliputi *power supply*, mikrokontroler arduino. Sedangkan perangkat lunak meliputi IDE (*Integrated Development Environment*), LabView.. Hubungan antara *hardware* dengan *software* adalah sebagai pendukung sistem kerja rangkaian sehingga alat dapat berfungsi dengan baik. Penjelasan mengenai perancangan dan pembuatan alat akan dibahas lebih rinci di bawah ini.

3.1 Perancangan Sistem

Diagram alur kerja alat merupakan tahapan – tahapan yang dilakukan dalam pembuatan alat secara umum seperti yang terdapat pada Gambar 3.1.



Keterangan :

= Tegangan +5V

= Koneksi Rangkaian

Gambar 3.1 *Diagram* Alur Sistem

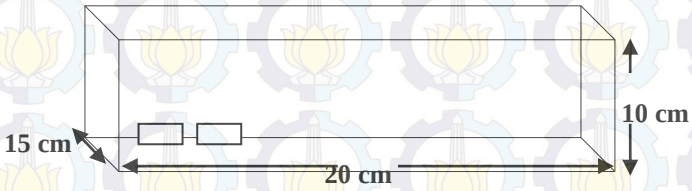
Secara umum sistem yang terdapat pada Gambar 3.1 adalah sistem kerja alat pada Tugas Akhir ini. *Power Supply* berfungsi memberikan tegangan dari Mikrokontroler dijalankan melalui komputer. Komputer bukan hanya melakukan pengalamatan mikrokontroler ATMega 16 namun juga sebagai objek yang dikontrol. Sebab pada komputer juga dibuat program melalui *software* LabView sebagai tampilan pengontrolan temperatur melalui led virtual dimana led virtual tersebut sebagai indikator *heater* dan *fan*. Agar pengontrolan dapat berjalan dengan lebih mudah, maka digunakan mikrokontroler arduino

MEGA2560 sebagai *interface* dalam penentu logika *high* dan *low* maupun *input* atau *output* pada pin mikrokontroler arduino.

Pada sub bab selanjutnya akan dijelaskan mengenai beberapa bagian pendukung sistem alat tersebut mulai dari perangkat keras hingga perangkat lunak.

3.2 Perancangan Mekanik

Pada perancangan mekanik, dibutuhkan *box* dengan ukuran seperti tampak pada Gambar 3.2. *Box* terbuat dari plastik dengan ketebalan 2mm. Isi *box* terdiri dari alat – alat *hardware* seperti arduino, mikrokontroler serta power supply.

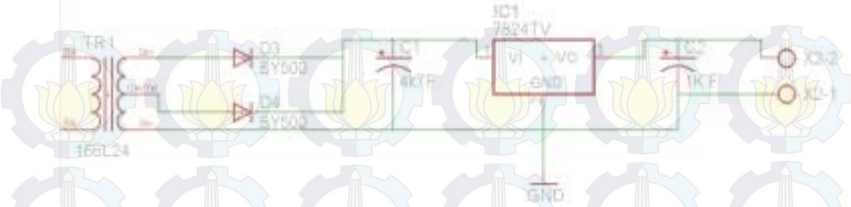


Gambar 3.2 Desain *Box*

Dalam Tugas Akhir ini, mikrokontroler akan berkomunikasi dengan *hardware* yang terdapat pada *box*. Oleh karena itu, *box* didesain agar terdapat lubang sebagai tempat kabel *input* / *output* dari mikrokontroler.

3.3 Perancangan *Power Supply*

Power Supply merupakan alat yang berfungsi sebagai sumber tegangan listrik DC pada rangkaian Tugas Akhir ini. Sumber tegangan untuk *power supply* diperoleh dari sumber tegangan jala – jala listrik 220VAC. Alat ini digunakan sebagai sumber tegangan +5V pada rangkaian. Tegangan +5V untuk Arduino didapat dari PC. Mikronkontroler memperoleh tegangan 5 Volt dari *power supply*. Skematik rangkaian *power supply* terdapat pada Gambar 3.3.



Gambar 3.3 Skematik *Power Supply* +5Vdc

3.4 Perancangan Program pada IDE Mikrokontroler Arduino MEGA 2560

Pada tahap perancangan program, program arduino akan dihubungkan dengan LabView yang mana secara umum arduino digunakan untuk pengaturan kondisi *high* atau *low* pada keadaan dan syarat tertentu maupun difungsikan sebagai *input* atau *output*. Mikrokontroler arduino akan mengeluarkan logika 1 atau logika 0 (*digital*) tergantung dengan sistem yang diinginkan berdasarkan penerimaan karakter ASCII (*American Standard Code For Information Interchange*) yang dikirimkan melalui LabView. Namun dalam Tugas Akhir ini, Arduino difungsikan sebagai *interface* sekaligus I/O, komunikasi serial, pembacaan data untuk melakukan perintah ketika arduino menerima informasi dari LabView.

Pemrograman IDE pada Tugas Akhir ini berdasarkan pada sistem pengaturan temperatur otomatis dan sistem buka/tutup valve, sebagai berikut:

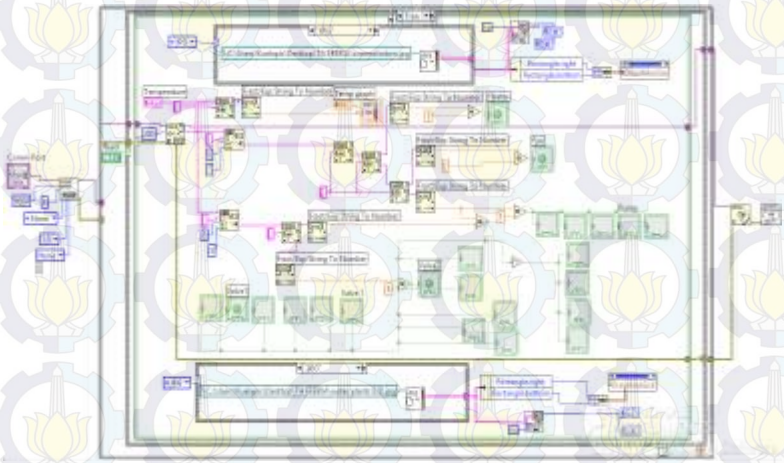
1) Sistem kerjanya, apabila simulasi pada LabView *running*, sensor temperatur virtual akan *ON*. Termometer virtual melakukan pembacaan tegangan dari sensor temperatur LM35 berupa kenaikan level temperatur dan menampilkan angka kenaikan temperatur pada indikator *digital* termometer.
2) Ketika temperatur pada termometer virtual melebihi batas maksimal yang ditentukan, indikator *fan* akan menyala. Ketika temperatur pada termometer virtual kurang dari batas minimal yang ditentukan, indikator *heater* akan menyala.
3) Penyiraman tanaman pada LabView menggunakan tombol ON/OFF yang dikontrol oleh kontroler (dalam hal ini Mikrokontroler), untuk menyalakan pompa. Untuk membuka /

menutup valve, digunakan toggle Mikrokontroler yang terhubung dengan plan pada LabView.

Listing program untuk simulator kontrol temperature dalam rumah kaca dapat dilihat pada bagian lampiran buku ini.

3.5 Perancangan Program pada LabView

LabView juga merupakan program pendukung dalam Tugas Akhir ini yaitu sebagai *plant* dalam bentuk *virtual* atau yang biasa dikenal dengan istilah *virtual plant*. *Virtual plant* yang dibuat merupakan simulasi rumah kaca dimana *fan*, *heater*, pompa dan *valve* akan di control melalui program yang terdapat pada *block diagram* LabView. Gambar 3.4 merupakan program yang dibuat pada software LabView

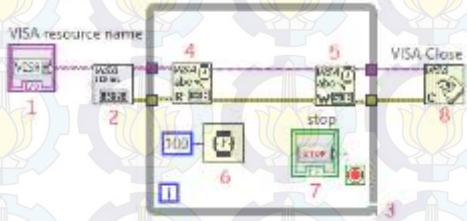


Gambar 3.4 *Block diagram* Pada LabView

Pemrograman dengan menggunakan LabView dibutuhkan untuk mengatur sistem kerja simulasi rumah kaca yang mengontrol suhu ruangan dalam rumah kaca. Pada simulasi rumah kaca tersebut terdapat sensor LM35 sebagai sensor temperatur melebihi batas maksimal yang ditentukan, maka *fan* akan menyala dan jika temperatur kurang dari batas maksimal yang ditentukan, maka *heater* akan menyala.

Komunikasi serial antara Arduino dengan LabView harus memenuhi logika dan persyaratan yang benar agar sesuai dengan sistem kerja yang diinginkan. Oleh karena itu, ketika arduino mulai

mengirimkan karakter ASCII melalui komunikasi serial, maka LabView harus dapat membaca informasi yang dikirimkan melalui komunikasi serial tersebut sehingga LabView mampu memerintahkan / mengeksekusi *input* yang diterima menjadi *output* berupa indikator virtual dan menyalakan indikator . Begitu juga sebaliknya ketika LabView membaca karakter ASCII yang dikirimkan oleh arduino melalui komunikasi serial, LabView akan menjalankan perintah berdasarkan data hasil pengiriman tersebut.

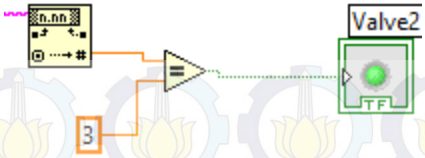


Gambar 3.5 *Block* Kirim dan Terima Serial LabView

Block pada Gambar 3.5 merupakan *block* dasar untuk LabView dapat melakukan pengiriman dan penerimaan data dengan perangkat luar dimana fungsi - fungsi per komponen sudah dijelaskan pada BAB II. Mulai dari *block setting* COM (No. 1), *block* inisialisasi komunikasi serial (No. 2), *block* proses yang berulang-ulang atau kontinyu (No. 3), *block* penerima (No. 4) dan pengirim (No. 5) data serial, *setting delay* untuk penerimaan data serial yang lebih baik (No. 6), *stop button* untuk menghentikan program (No. 7), dan *block* penutup komunikasi serial (No. 8).

Gambar 3.6 *Block* Penerima Data Serial

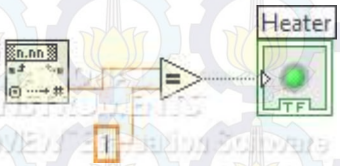
Pada *block* VISA *Read* seperti Gambar 3.6 kita harus menyambungkan VISA *Resource Name* (garis warna ungu) dan *Error Out* (garis warna kuning belang putih) ke VISA *Read*. Kemudian *create constant* (kotak warna biru) dengan angka yang sesuai dengan jumlah *array* yang dikirim dari Arduino MEGA. Data yang dikirim melalui serial adalah data *string* (garis warna merah muda).



Gambar 3.7 *Block* penghidup valve

Gambar 3.7 adalah salah satu bagian dari block diagram untuk Virtual plant simulasi rumah kaca. Dimana *Fract/exp string to number* yang berfungsi digunakan untuk mengubah karakter angkadi dalam string menjadi floating-point number. Dimana Data serial yang berupa *string* akan dirubah menjadi menjadi *floating-point* setelah berubah menjadi data numerik. Data yang telah di dapat akan di bandingkan dengan nilai konstanta yang telah diberikan yaitu nilai ”3” sebagai pembanding dengan menggunakan block *Equal* yang Berfungsi membandingkan nilai antara x dan y. Ketika nilai x atau y sama, maka *output* akan dikeluarkan

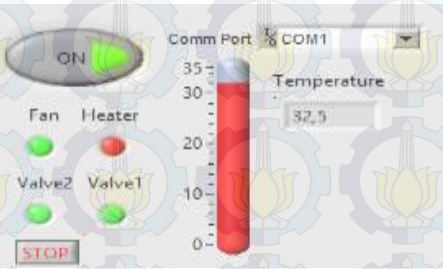
Untuk susunan *diagram block* pada Gambar 3.7 ini, hasil yang diinginkan adalah untuk menghidupkan indikator lampu pada LabView. Untuk menghidupkan indikator, maka *input*-nya harus berupa logika (garis hijau). Data *array* yang sudah diambil kemudian dibandingkan menggunakan *block equal?*. Pada *block* ini *input* harus berjenis sama. Hasil perbandingan tersebut akan menghasilkan logika “*TRUE*” atau “*FALSE*”. Jika bernilai “TRUE” maka indikator akan menyala. Pada *block diagram* Gambar 3.7 block diagram bernilai logic 1 dan akan dibandingkan dengan pembanding ”3”. Karena perbandingannya menghasilkan *output* “TRUE” maka lampu indikator Valve 2 akan menyala.



Gambar 3.8 *Block* Pengambilan Data dan Penampilan Nilai Pada *Front Panel*

Pada Gambar 3.8 adalah salah satu bagian dari block diagram untuk *virtual plant* simulasi rumah kaca. Dimana *Fract/exp string to number* yang berfungsi digunakan untuk mengubah karakter angkadi dalam *string* menjadi *floating-point* number. Dimana Data serial yang berupa *string* akan dirubah menjadi menjadi *floating-point* setelah berubah menjadi data numerik. Data yang telah di dapat akan di bandingkan dengan nilai konstanta yang telah diberikan yaitu nilai ”1”

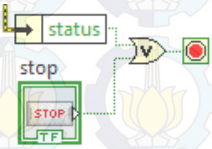
Gambar 3.8 *block* diagram bernilai logic 1 dan akan dibandingkan dengan pembanding ”1”. Karena perbandingannya menghasilkan *output* “*TRUE*” maka lampu indikator *heater* akan menyala.seperti pada Gambar 3.9 *heater on* ketika suhu di bawah batas yang ditentukan.



Gambar 3.9 Tampilan Sistem Virtual Kontrol Temperatur

Gambar 3.10 *Block* Pengirim Data Serial

Pada *Block diagram* seperti Gambar 3.10 *array* string dari *block* Gambar 3.10 tadi kemudian dikirimkan melalui komunikasi serial untuk kemudian diterjemahkan Arduino MEGA menjadi tegangan sebagai *input* Mikrokontroler.



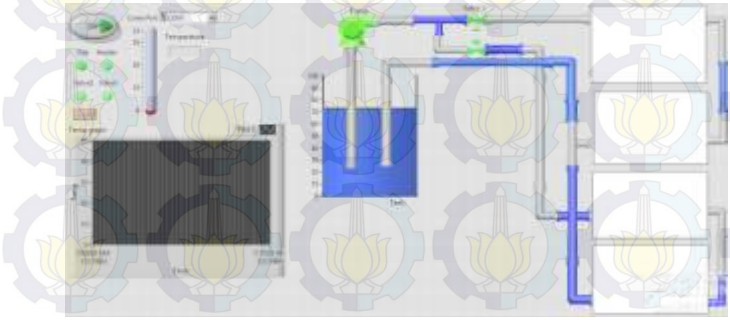
Gambar 3.11 *Block Running Stopped*

Pada Gambar 3.11 adalah fungsi untuk menghentikan program. Berhentinya program bisa disebabkan 2 hal, karena penekanan *stop button* atau karena *error* pada program (garis kuning belang putih). Hal ini bisa terjadi karena ada *OR Gate* yang artinya jika satu kondisi terpenuhi maka akan bernilai "*TRUE*".

3.6 Perancangan *Virtual Plan* Rumah Kaca

Objek yang telah disusun dengan baik tentu tidak lengkap dengan adanya perancangan sistem yang berfungsi mendukung *virtual plant* simulasi rumah kaca. Dalam pembuatan *virtual plant* simulasi rumah kaca, perancangan simulasi rumah kaca menggunakan *palet control* dimana fungsi *palet control* dapat diambil melalui tampilan / jendela *front panel*. *Palet control* digunakan untuk mengambil objek – objek berdasarkan kategori yang disediakan atau dengan menggunakan tombol *search*.

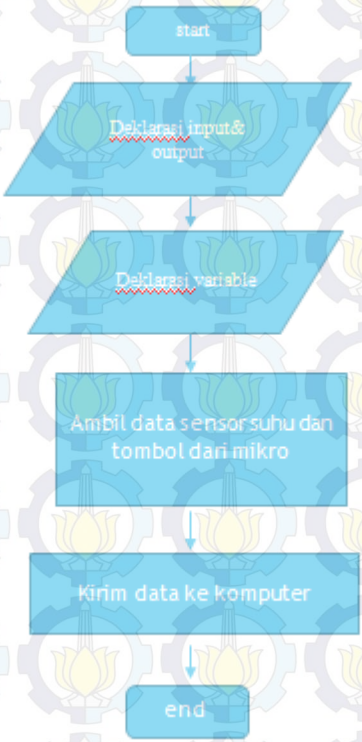
Objek *virtual plant* seperti yang terdapat pada Gambar 3.12, dapat berkomunikasi secara serial dengan program pada LabView. Oleh karena itu, dibutuhkan beberapa instrumen I/O serial antara lain VISA *configure serial port*, VISA *write*, VISA *read*, dan VISA *close*. VISA *configure serial port* berfungsi mengatur *setting* komunikasi serial termasuk saluran yang digunakan (VISA *resource name*), kecepatan komunikasi (*baud rate*), dan lain – lain. VISA *write* berfungsi mengirimkan data ke perangkat *hardware interface* yang telah ditentukan oleh VISA *resource name*. VISA *read* berfungsi membaca sejumlah *byte* data dari perangkat *hardware interface* yang telah ditentukan oleh VISA *resource name*. VISA *close* yang berfungsi menutup komunikasi serial.



Gambar 3.12 Objek *Virtual Plant* Rumah Kaca

3.7 Flowchart Sistem

Di bawah ini adalah gambar Flowchart secara umum dimana Flowchart ini mendefinisikan keseluruhan jalannya *system* kerja alat ini:



Gambar 3.13 Gambar Flow Chart Program Secara Umum

Halaman ini sengaja dikosongkan

BAB IV
PENGUJIAN DAN ANALISA DATA

Dalam perencanaan dan pembuatan suatu sistem, pengujian dan analisa sangat diperlukan, karena dengan pengujian dan analisa dimaksudkan untuk mengetahui kinerja dari sistem dan komponen pendukung yang dibuat sudah sesuai dengan yang direncanakan atau belum. Selain itu dapat mengetahui kelemahan-kelemahan dari alat tersebut sehingga dapat ditemukan cara untuk memperbaiki atau mengembangkannya menjadi lebih baik lagi. bab ini akan dibahas tentang pengujian dan analisa baik itu pada *hardware* dan *software* yang mendukung dari dibuatnya Tugas Akhir ini serta sistem secara keseluruhan.

4.1 Pengujian *Hardware*

Pada pembuatan rangkaian elektronika, sebelum dilakukan penyambungan antar rangkaian terlebih dahulu dilakukan pengujian terhadap rangkaian tersebut. Hal ini dilakukan untuk mengetahui hasil dari tiap-tiap rangkaian sebelum dilakukan pengujian secara keseluruhan.

4.1.1 Pengujian Arduino MEGA 2560

Pengujian terhadap Arduino MEGA 2560 ini dilakukan dengan memberikan tegangan +5 Volt pada rangkaian dan memrogram Arduino MEGA 2560 untuk diukur tegangan pada tiap pin menggunakan voltmeter. Pengujian dilakukan dengan mengukur tegangan pada tiap pin Arduino MEGA 2560 dengan dua kondisi, yaitu kondisi logika 0 dan 1. Gambar 4.1 menunjukkan program arduino dengan mengambil contoh pemrograman pada 10 pin pada Arduino MEGA 2560

```
void setup()
{pinMode(0,OUTPUT);
pinMode(1,OUTPUT);
pinMode(2,OUTPUT);
pinMode(3,OUTPUT);
pinMode(4,OUTPUT);
pinMode(5,OUTPUT);
pinMode(6,OUTPUT);
pinMode(7,OUTPUT);
pinMode(8,OUTPUT);
pinMode(9,OUTPUT);
pinMode(10,OUTPUT);
void loop()
{digitalWrite(0,HIGH);
digitalWrite(1,HIGH);
digitalWrite(2,HIGH);
digitalWrite(3,HIGH);
digitalWrite(4,HIGH);
digitalWrite(5,HIGH);
digitalWrite(6,HIGH);
digitalWrite(7,HIGH);
digitalWrite(8,HIGH);
digitalWrite(9,HIGH);
digitalWrite(10,HIGH);
```

Gambar 4.1 Program untuk Kondisi *Active High* (Logika 1)

```
void setup()
                {pinMode(0,OUTPUT);
                 pinMode(1,OUTPUT);
                 pinMode(2,OUTPUT);
                 pinMode(3,OUTPUT);
                 pinMode(4,OUTPUT);
                 pinMode(5,OUTPUT);
                 pinMode(6,OUTPUT);
                 pinMode(7,OUTPUT);
                 pinMode(8,OUTPUT);
                 pinMode(9,OUTPUT);
                pinMode(10,OUTPUT);
                        void loop()
                 {digitalWrite(0,LOW);
                  digitalWrite(1,LOW);
                  digitalWrite(2,LOW);
                  digitalWrite(3,LOW);
                  digitalWrite(4,LOW);
                  digitalWrite(5,LOW);
                  digitalWrite(6,LOW);
                  digitalWrite(7,LOW);
                  digitalWrite(8,LOW);
                  digitalWrite(9,LOW);
                 digitalWrite(10,LOW);
```

Gambar 4.2 Program untuk Kondisi *Active Low* (Logika 0)

Dibawah ini merupakan data yang diambil dari pengujian keluaran tegangan yang terdapat pada masing-masing pin di dalam Arduino MEGA 2560 .Tabel hasil pengkuran tegangan 10 pin di Aduino MEGA 2560 Tabel 4.1:

Tabel 4.1 Hasil Pengukuran Tegangan 10 pin Arduino MEGA

Pin	*Active High*	*Active Low*
Pin 0	4,92 V	107,2 mV
Pin 1	4,92 V	12,5 mV
Pin 2	4,92 V	8,8 mV
Pin 3	4,92 V	5,4 mV
Pin 4	4,92 V	5,4 mV
Pin 5	4,92 V	3,4 mV
Pin 6	4,90 V	3,7 mV
Pin 7	4,91 V	3,7 mV
Pin8	4,90 V	3,9 mV
Pin9	4,91 V	3,7 mV
Pin10	4,90 V	3,4 mV

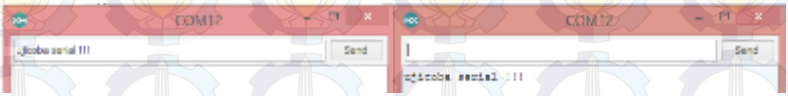
Dari tabel data hasil pengujian di atas, tegangan keluaran dari tiap-tiap pin rata-rata sama, hal ini menunjukkan bahwa tiap-tiap pin pada Arduino dapat bekerja dengan baik.

4.1.2 Pengujian Komunikasi Serial

Arduino MEGA 2560 berfungsi sebagai *interface* I/O antara kontroler dan *virtual plant* pada LabView sehingga dibutuhkan komunikasi antar keduanya. Arduino MEGA 2560 sebagai pengakuisisi data, bertugas mengirim data dari kontroler berupa I/O digital dan analog menjadi kumpulan *array string* dan sebaliknya LabView juga mengirimkan data ke Arduino MEGA 2560. Maka dari itu Arduino MEGA 2560 harus dapat mengirim dan menerima data. Program sederhana untuk menguji hal ini ditunjukkan Gambar 4.3 dan hasil uji cobanya pada Gambar 4.4.

```
void setup()
{ Serial.begin(9600);}
void loop()
{ if(Serial.available())
  {byte dataku=Serial.read();
    Serial.print (dataku);}
}
```

Gambar 4.3 Program Kirim dan Terima Data Serial



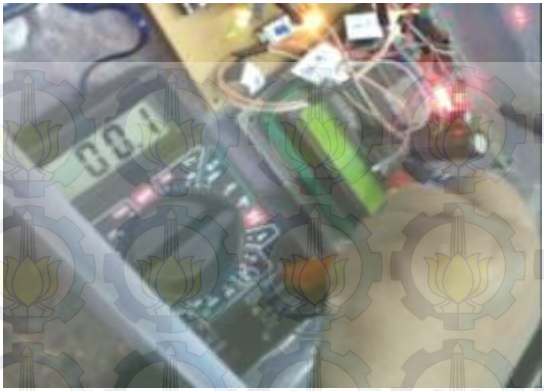
(a) (b)

Gambar 4.4 Hasil Pengujian Komunikasi Serial (a) Kirim Data dan (b) Terima Data Arduino MEGA 2560

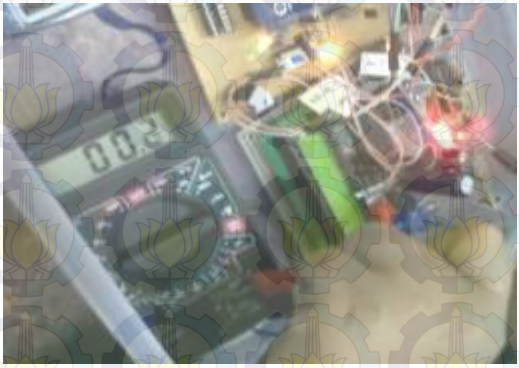
Dari hasil ujicoba diatas, Arduino MEGA 2560 dapat menerima data sesuai data yang dikirimkan ke Arduino MEGA 2560. Hal ini menunjukkan bahwa Arduino MEGA 2560 dapat melakukan komunikasi serial dengan baik.

4.1.3 Pengujian Mikrokontroller ATMega 16

Pengujian terhadap Mikrokontroller ATMega 16 ini dilakukan dengan memberikan tegangan +5 Volt pada rangkaian dan diukur tegangan pada tiap pin menggunakan voltmeter. Pengujian dilakukan dengan mengukur tegangan pada Pin Input atau output Mikrokontroller ATMega 16. (Gambar 4.5 , Gambar 4.6 , Gambar 4.7) menunjukkan keluaran input Mikrokontroller ATMega16.



Gambar 4.5 Pengukuran Tegangan Pada PA0



Gambar 4.6 Pengukuran Tegangan Pada PA1

Gambar 4.7 Tegangan PA2

Dibawah ini merupakan data yang diambil dari pengujian keluaran tegangan yang terdapat pada masing-masing pin di dalam *Mikrokontroller* ATMEGA 16 .Tabel hasil pengkuran tegangan input pin di *Mikrokontrolle*r ATMEGA 16 Tabel 4.2:

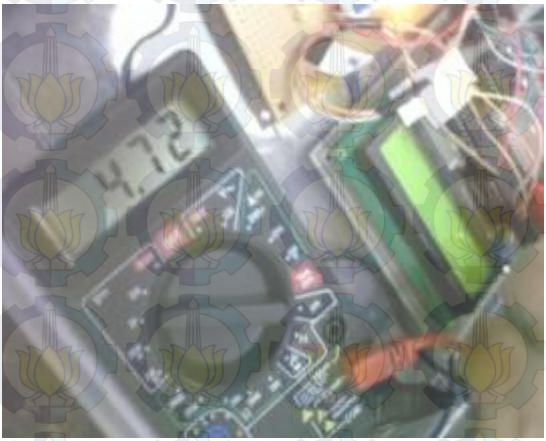
Tabel 4.2 Hasil Pengukuran Tegangan pin Mikrokontroller

PIN	TEGANGAN
PIN PA0	0,6 V
PIN PA1	0,2 V
PIN PA2	0,1 V

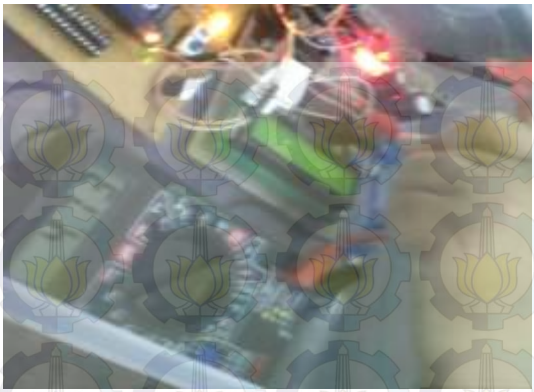
4.1.4 Pengujian tegangan Output Mikrokontroller

Pengujian terhadap Mikrokontroller ATMega 16 ini dilakukan dengan memberikan tegangan +5 Volt pada rangkaian dan diukur tegangan pada tiap pin menggunakan voltmeter. Pengujian dilakukan dengan mengukur tegangan pada pin *input* atau *output* Mikrokontroller ATMega 16. Gambar 4.8 , Gambar 4.9 , Gambar 4.10 menunjukkan keluaran *input* Mikrokontroller ATMega16.

Gambar 4.8 Tegangan PB0



Gambar 4.9 Tegangan PB1



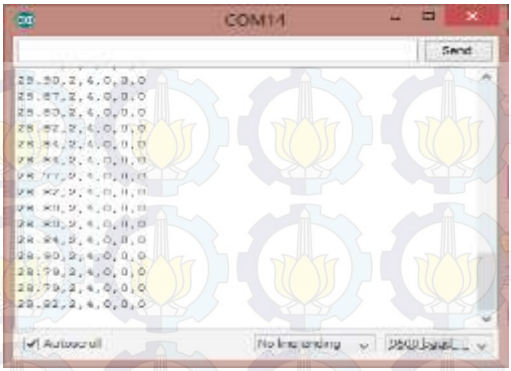
Gambar 4.10 Tegangan PB2

4.1 Pengujian *Software*

Sistem perangkat lunak merupakan sistem antarmuka yang biasa digunakan untuk sistem komunikasi dengan *user* (pengguna). *User* menggunakan perangkat lunak ini untuk mempermudah pengaturan keadaan dari suatu *plant* dengan kondisi tertentu.

4.1.1 Pengujian Program Mikrokontroler

Output yang dikeluarkan dari Arduino MEGA 2560 dijadikan *input* kontroler dan juga sebaliknya *output* kontroler akan dijadikan *input* Arduino MEGA 2560. LabView sendiri juga mengirimkan data berupa kenaikan dan penurunan temperatur. Komunikasi serial antara LabView dan Arduino juga harus lancar agar tidak adanya kesalahan dalam logika program yang lain. Ini dikarenakan jika data yang dikirim LabView salah, maka Arduino MEGA 2560 juga akan menghasilkan *output* yang salah. Pada bagian lampiran ditampilkan program untuk Arduino MEGA 2560 secara keseluruhan. Berikuti ini pengujian program pembacaan temperatur yang menampilkan data dari LM35 melalui serial monitor:



Gambar 4.11 Pengujian Data Pembacaan Temperatur Dari LM35

4.1.1 Pengujian Program LabView

Pembuatan *simulator* temperatur *ini* menggunakan LabView 2013, Perangkat lunak ini digunakan sebagai antarmuka *virtual plant* dan mengatur segala kegiatan pada objek. Ada dua hal penting dalam yaitu pengiriman dan penerimaan data serial LabView. Sebelum pengujian ini dilakukan, pastikan pengaturan alamat COM yang digunakan pada LabView sama dengan alamat COM yang terpasang pada *port* USB, jika tidak maka LabView tidak akan bereaksi apa-apa karena tidak adanya *input*.

Hal yang pertama adalah pengujian terhadap pengiriman data serial dari Arduino MEGA 2560. Pengujian program pada LabView ditampilkan pada bagian lampiran.

4.1.2 Pengujian Sistem Secara Keseluruhan

Pengujian akhir yang dilakukan dalam pembuatan Tugas Akhir ini adalah pengujian sistem secara keseluruhan, baik itu dilakukan secara manual maupun otomatis dan diteliti kesesuaiannya dari kontroler tehadap *virtual plant* yang dibuat. Pengujian ini dilakukan untuk mengetahui jalannya sistem dari program yang telah dibuat dan untuk mengetahui kekurangan dari sistem.

Yang pertama adalah pengujian secara manual, yaitu pengujian program Arduino MEGA 2560 pada *Virtual Plant* LabView tanpa menghubungkannya dengan mikrokontroler dengan cara memberikan inputan pada tegangan 5 V dari *power supply* pada pin *input* Arduino

untuk melihat reaksinya pada *virtual plant* LabView. Jika pemrosesan sudah dilakukan maka dapat dipastikan indikator pada bagian *front panel* dari *LabView* akan menyala atau *ON*. Di bawah ini merupakan Tabel pengukuran suhu pada *LabView* yang diberikan inputan secara manual tanpa melalui mikrokontroler:

Tabel 4.3 *Data Pengujian Manual*

Tanggal	***waktu***	***Batas normal (°C)***	***Suhu saat ini (°C)***
09/1/2014	1:14:34 PM	30	29
09/1/2014	1:14:34 PM	30	29
09/1/2014	1:14:35 PM	30	29
09/1/2014	1:14:36 PM	30	29
09/1/2014	1:14:37 PM	30	29
09/1/2014	1:14:38 PM	30	29
09/1/2014	1:14:39 PM	30	29
09/1/2014	1:14:40 PM	30	30
09/1/2014	1:14:41 PM	30	30
09/1/2014	1:14:42 PM	30	30
09/1/2014	1:14:43 PM	30	30
09/1/2014	1:14:44 PM	30	30
09/1/2014	1:14:45 PM	30	31
09/1/2014	1:14:46 PM	30	31
09/1/2014	1:14:47 PM	30	31
09/1/2014	1:14:48 PM	30	31

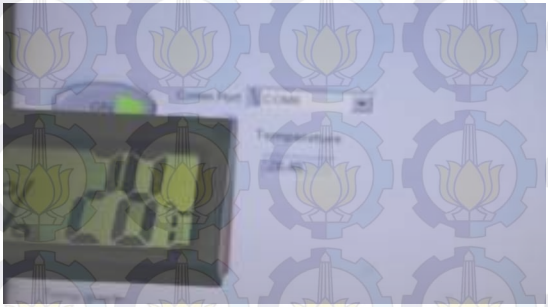
Tabel diatas diperoleh dengan cara memberi batas normal suhu pada *LabView* yaitu sebesar 30 derajat *celcius* dimana dalam selang waktu selama 5 detik antara pukul 14:34 sampai pukul 14:35 suhu rumah kaca berada di bawah batas normal yaitu sebesar 29 derajat *celcius* sedangkan dalam selang waktu selama 4 detik antara pukul 14:45 sampai pukul 14:48 suhu rumah kaca berada di atas batas normal yaitu sebesar 31 derajat *celcius.*

Yang kedua adalah pengujian secara otomatis. Keseluruhan proses yang terjadi sama dengan pengujian manual, hanya saja pada pengujian ini akan dihubungkan dengan Mikrokontroller.

Tabel 4.4 Data Pengujian keseluruhan

Tabel	Waktu	Batas Bawah Suhu(*°C*)	Suhu Yang Terukur(*°C*)
20/1/2014	10:14:34 PM	28	28.49
20/1/2014	10:14:34 PM	28	28.59
20/1/2014	10:14:34 PM	28	28.62
20/1/2014	10:14:34 PM	28	28.54
20/1/2014	10:14:34 PM	28	28.38
20/1/2014	10:14:34 PM	28	28.59
20/1/2014	10:14:34 PM	28	28.43
20/1/2014	10:14:34 PM	28	28.46
20/1/2014	10:14:34 PM	28	28.66
20/1/2014	10:14:34 PM	28	28.53
20/1/2014	10:14:34 PM	28	28.51
20/1/2014	10:14:34 PM	28	28.58
20/1/2014	10:14:34 PM	28	28.38
20/1/2014	10:14:34 PM	28	28.43
20/1/2014	10:14:34 PM	28	28.54

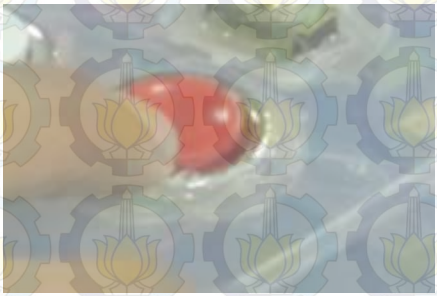
Tabel 4.4 diatas diperoleh dengan cara memberi Batas Normal suhu pada LabView yaitu sebesar 28 derajat *celcius* dimana dalam selang waktu selama 5 detik antara pukul 10:14:34 sampai pukul 10:14:35 suhu rumah Kaca berada di batas normal yaitu sebesar 28 derajat *celcius* dimana jika suhu Rumah kaca naik maka blower akan menyala sedangkan jika suhu berada di bawah maka heather akan menyala.



Gambar 4.6 Pengujian Menggunakan Termometer Manual

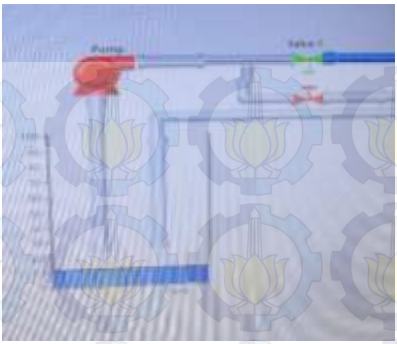
Pada Gambar 4.6 terlihat bahwa suhu pada termometer manual menunjukkan suhu 28 derajat *celcius* dimana saat melakukan pengujian alat mendekatkan termometer dengan LM35 dan setelah di lakukan tes dengan menghidupkan Program LabView menujukkan hasil temperatur yang hampir sama dimana temperatur digital menunjukkan suhu 28 derajat *celcius* sedangkan temperatrur digital menunjukkan suhu 28,48 derajat *celcius* dari hasil gambar di atas suhu *virtual* pada LabView sama dengan hasil pengujian menggunakan termometer dimana pada LabView suhu lebih akurat

Sedangkan untuk tangki air terdapat *valve* dan pompa yang digunakan untuk mengalirkan air. *Valve* dan pompa menggunakan *push button* untuk mengisi air pada tangki tersebut. Dimana saat menekan Push button 1 akan hidup dan mengisi tangki, seperti pada Gambar 4.7.



Gambar 4.7 *Push Button* Merah

Ketika *Push button* merah ditekan maka air pada *virtual tank* akan mengisi dan ketika *push button* merah ditekan lagi maka air dalam *virtual tank* akan berhenti mengisi seperti pada Gambar 4.8

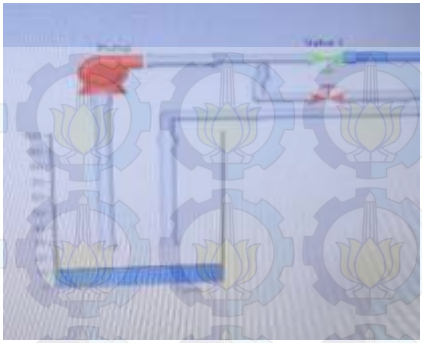


Gambar 4.8 Air Mengisi *Virtual Tank*

Untuk *valve* 1 pada *virtual plant*. Dimana saat menekan *push button* 2 akan hidup dan menyalakan *valve* 1, Seperti pada gambar 4.9.

Gambar 4.9 *Push Button* Putih

Push button Putih ditekan maka *valve* 1 *virtual plant* akan menyala dan ketika *push button* Putih ditekan lagi maka *valve* 2 pada *virtual plant* akan berhenti seperti pada gambar 4.10.

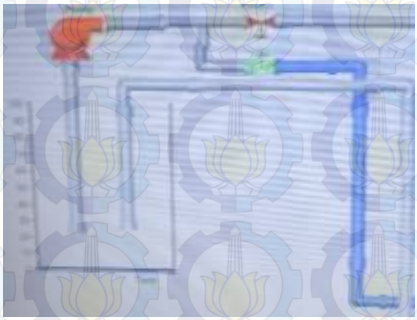


Gambar 4.10 Valve 1 *ON*

Vntuk valve 2 pada *virtual plant*. Dimana saat menekan *push button* Putih akan menyalakan *valve* 2. Seperti pada gambar 4.11 .

Gambar 4.11 *Push Button* Putih

Push button Putih ditekan maka *Valve* 2 virtual plant akan menyala dan ketika *push button* Putih ditekan lagi maka *valve* 2 pada *virtual plant* akan berhenti seperti pada gambar 4.12



Gambar 4.12 Valve 2 *ON*

BAB V
PENUTUP

5.1 Kesimpulan

Dari Tugas Akhir yang telah kami kerjakan, kami dapat menyimpulkan bahwa :

1. Sistem kontrol virtual dapat memodelkan bentuk dan proses berdasarkan *plant* nyata, sehingga minim biaya dan risiko.
2. Sistem kontrol temperature secara virtual dapat mengambil data real pembacaan temperatur dari sensor LM35, sehingga dapat diaplikasikan pada sistem kontrol nyata dengan mengubah indikator *heater* dan *fan* menjadi media nyata yang dikontrol.
3. Arduino MEGA 2560 dapat merepresentasikan I/O dari mikrokontroler menjadi I/O yang dapat diproses oleh Labview, dan begitupula sebaliknya.

5.2 Saran

Untuk pengembangan dan penyempurnaan dari Simulator Kontrol Temperatur Dalam RumahKaca, maka diberikan beberapa saran sebagai berikut :

1. Jika ingin membuat simulator dengan bentuk fisik dan simulasi komputer, maka kita dapat menggantikan indikator *heater*, *fan*, pompa dan *valve* dengan komponen dan peralatan nyata. Dengan begitu simulator fisik dapat dijadikan sistem kontrol dalam bentuk rancang bangun. Sedangkan simulator pada LabView dapat dijadikan sebagai HMI (*Human Machine Interface*), yang dapat difungsikan sebagai monitoring sistem kontrol itu sendiri.

Halaman ini sengaja dikosongkan

DAFTAR PUSTAKA

[1] Arif Sulaiman, ***ARDUINO: Mikrokontroler Bagi Pemula hingga Mahir***, <URL : http://buletin.balaielektronika.com/?p=163>, 1 Juni 2013.

[2] Artanto, Dian, ***Interaksi Arduino Dan LabVIEW***, Elex Media Komputindo, Jakarta, 2012.

[3] Muhammad Fadli Nasution, ***Relay***, <URL:http://ini-robot.blogspot.com/2012/07/relay.html>, 4 Juni 2013.

[4] ---, ***Metode Penelitian***, <URL :http://repository.upi.edu/operator/upload/s_fis_0708831_chapter3.pdf>, 5 Juni 2013.

[5] Setiawan A. R, ***Pengenalan LABVIEW***, <URL : http://elib.unikom.ac.id/download.php?id=59756>, 5 Juni 2013.

[6] Hasan, Ir. M. Igbal. 2002. ***Pokok-Pokok Materi Teori Pengambilan Keputusan***. Jakarta: Ghalia Indonesia.

[7] Suhardiyanto, H. 2009. ***Teknologi Rumah Tanaman untuk Iklim Tropika Basah Pemodelan dan Pengendalian Lingkungan***. Bogor : IPB Press.

Halaman ini sengaja dikosongkan

LAMPIRAN 1

LISTING PROGRAM

```
#include <Thread.h>
Thread SENSOR;
Thread SWITCH;
int PIN_SUHU=A0;
int counter;
//====================
int PIN_SW_HEATER=2;
int PIN_SW_CHILLER=3;
int PIN_SW_PUMP=4;
int PIN_VALVE4=5;
int PIN_VALVE5=6;
int PIN_VALVE6=7;
//========pin out========
int PIN_HEATER=8;
int PIN_CHILER=9;
int PIN_PUMP=10;
//=======================
int TOGGLE_SW_PUMP_A1=A8;
int TOGGLE_SW_PUMP_A2=A9;
int TOGGLE_SW_PUMP_B1=A10;
```

```
int TOGGLE_SW_PUMP_B2=A11;

//=======================
int RL_PUMP=A1;
int RL_CHILER=A2;
int RL_HEATER=A3;
int RL_VALVE1=A4;
int RL_VALVE2=A5;
int RL_VALVE3=A6;

double AVR_SUHU;
double AVR_HUMIDITY;
double AVR_PH;
boolean TAP_HEATER;
boolean TAP_CHILLER;
boolean TAP_PUMP;
//=======================
boolean STATE_HEATER;
boolean STATE_PUMP;
boolean STATE_CHILLER;
void setup(){
 Serial.begin(9600);
```

```
counter=0;
pinMode(PIN_SW_HEATER,INPUT);//SWITCH 1 HEATER
pinMode(PIN_SW_CHILLER,INPUT);//SWITCH 2 CHILLER
pinMode(PIN_SW_PUMP,INPUT);//SWITCH 3 PUMP
pinMode(PIN_VALVE4,INPUT);//SWITCH 4
pinMode(PIN_VALVE5,INPUT);//SWITCH 5
pinMode(PIN_VALVE6,INPUT);//SWITCH 6
//==================================
pinMode(TOGGLE_SW_PUMP_A1,INPUT);
pinMode(TOGGLE_SW_PUMP_A2,INPUT);
pinMode(TOGGLE_SW_PUMP_B1,INPUT);
pinMode(TOGGLE_SW_PUMP_B2,INPUT);
digitalWrite(TOGGLE_SW_PUMP_A1,HIGH);
digitalWrite(TOGGLE_SW_PUMP_A2,HIGH);
digitalWrite(TOGGLE_SW_PUMP_B1,HIGH);
digitalWrite(TOGGLE_SW_PUMP_B2,HIGH);
//==================================
digitalWrite(PIN_SW_HEATER,HIGH);
digitalWrite(PIN_SW_CHILLER,HIGH);
digitalWrite(PIN_SW_PUMP,HIGH);
digitalWrite(PIN_VALVE4,HIGH);
digitalWrite(PIN_VALVE5,HIGH);
```

```
digitalWrite(PIN_VALVE6,HIGH);
//============================
pinMode(PIN_HEATER,OUTPUT);
pinMode(PIN_CHILER,OUTPUT);
pinMode(PIN_PUMP,OUTPUT);
digitalWrite(PIN_HEATER,HIGH);
digitalWrite(PIN_CHILER,HIGH);
digitalWrite(PIN_PUMP,HIGH);
//============================
/*
Setup pin relay
*/
pinMode(RL_PUMP,OUTPUT);
pinMode(RL_CHILER,OUTPUT);
pinMode(RL_HEATER,OUTPUT);
pinMode(RL_VALVE1,OUTPUT);
pinMode(RL_VALVE2,OUTPUT);
pinMode(RL_VALVE3,OUTPUT);
/*
Set output off
*/
digitalWrite(RL_PUMP,LOW);
```

```
digitalWrite(RL_CHILER,LOW);
digitalWrite(RL_HEATER,LOW);
digitalWrite(RL_VALVE1,LOW);
digitalWrite(RL_VALVE2,LOW);
digitalWrite(RL_VALVE3,LOW);


//----------------------------
SENSOR.onRun(Read_sensor);
SENSOR.setInterval(10);
SWITCH.onRun(READ_SWITCH);
SWITCH.setInterval(500);
}
void loop(){
if(SENSOR.shouldRun())SENSOR.run();
if(SWITCH.shouldRun())SWITCH.run();
int i;
char buffer[20];
if(Serial.available()){
 delay(10);
  while(Serial.available() && i<20){
    buffer[i]=char(Serial.read());
```

```
    i++;
   }
   if(i>0){
    String ReceiveData;
    for(int k=0;k<i;k++){
       ReceiveData+=String(buffer[k]);
    }
    String data = getValue(ReceiveData, '=', 1);
    String pin = getValue(ReceiveData, '=', 0);
    int PinCommand=pin.toInt();
    int command_data=data.toInt();
    COMMAND(PinCommand,command_data);
    }
   }
  delay(10);
 }
 //=========================================
 void READ_SWITCH(){
  if(digitalRead(TOGGLE_SW_PUMP_A1)==LOW){
   digitalWrite(RL_VALVE1,HIGH);
  }
 if(digitalRead(TOGGLE_SW_PUMP_A2)==LOW){
```

```
  digitalWrite(RL_VALVE1,LOW);
 }
 if(digitalRead(TOGGLE_SW_PUMP_B1)==LOW){
  digitalWrite(RL_VALVE2,HIGH);
 }
 if(digitalRead(TOGGLE_SW_PUMP_B2)==LOW){
  digitalWrite(RL_VALVE2,LOW);
 }
 //=========================================
 for(int y=2;y<8;y++){
  if(digitalRead(y)==LOW){
    String SendOut="SW:"+String(y-2);
    if(y==2){
     if(!TAP_HEATER){
      STATE_HEATER=true;
      digitalWrite(PIN_HEATER,LOW);
      digitalWrite(RL_HEATER,HIGH);
      delay(100);
      TAP_HEATER=true;
     }else{
      digitalWrite(PIN_HEATER,HIGH);
      digitalWrite(RL_HEATER,LOW);
```

```
    delay(50);
    TAP_HEATER=false;
    STATE_HEATER=false;
   }
  }else if(y==3){
   if(!TAP_CHILLER){
    STATE_CHILLER=true;
    digitalWrite(PIN_CHILER,LOW);
    digitalWrite(RL_CHILER,HIGH);
    delay(50);
    TAP_CHILLER=true;
   }else{
    digitalWrite(PIN_CHILER,HIGH);
    digitalWrite(RL_CHILER,LOW);
    delay(100);
    STATE_CHILLER=false;
    TAP_CHILLER=false;
   }
  }else if(y==4){
   if(!TAP_PUMP){
    STATE_PUMP=true;
    digitalWrite(PIN_PUMP,LOW);
```

```
        digitalWrite(RL_PUMP,HIGH);
        delay(50);
        TAP_PUMP=true;
       }else{
        digitalWrite(PIN_PUMP,HIGH);
        digitalWrite(RL_PUMP,LOW);
        delay(50);
        TAP_PUMP=false;
        STATE_PUMP=false;
       }
     }
   }
    while(digitalRead(y)==LOW){}
   }
}

void COMMAND(int p,int c){
  digitalWrite(p,c);
}
void Read_sensor(){
 if(counter<29){
   int suhu=analogRead(PIN_SUHU);
```

```
    double val_suhu=((double)suhu/1024)*50;//celcius LM35 max 50
derajad
     AVR_SUHU+=val_suhu;
     counter++;
   }else{
    boolean TOGEL1,TOGEL2;
     if(digitalRead(TOGGLE_SW_PUMP_A1)==LOW){
       TOGEL1=true;
     }else if(digitalRead(TOGGLE_SW_PUMP_A2)==LOW){
       TOGEL1=false;
     }
     if(digitalRead(TOGGLE_SW_PUMP_B1)==LOW){
       TOGEL2=true;
     }else if(digitalRead(TOGGLE_SW_PUMP_B2)==LOW){
       TOGEL2=false;
     }


     AVR_SUHU=(AVR_SUHU/30);
     AVR_SUHU=AVR_SUHU*10;

     /*
```

```
*/
///program switching suhu
if(AVR_SUHU<28.00){
 digitalWrite(RL_HEATER,HIGH);
 digitalWrite(PIN_HEATER,LOW);
 digitalWrite(RL_CHILER,LOW);
 digitalWrite(PIN_CHILER,HIGH);
 TAP_CHILLER=true;
 STATE_HEATER=true;
 STATE_CHILLER=false;
}else if(AVR_SUHU >28 && AVR_SUHU <32){
}else if(AVR_SUHU >32.00){
 digitalWrite(RL_HEATER,LOW);
 digitalWrite(RL_CHILER,HIGH);
 digitalWrite(PIN_CHILER,LOW);
 digitalWrite(PIN_HEATER,HIGH);
 TAP_CHILLER=true;
 STATE_CHILLER=true;
 STATE_HEATER=false;
}
```

```
String SEND_DATA;
SEND_DATA+=DOUBLEtoSTRING(AVR_SUHU,2);
if(TOGEL1){
 SEND_DATA+=",";
 SEND_DATA+="1";
}else{
 SEND_DATA+=",";
 SEND_DATA+="2";
}
//Valve
if(TOGEL2){
 SEND_DATA+=",";
 SEND_DATA+="3";
}else{
 SEND_DATA+=",";
 SEND_DATA+="4";
}
//Heater
if(STATE_HEATER){
 SEND_DATA+=",";
 SEND_DATA+="1";
}else{
```

```
  SEND_DATA+=",";
  SEND_DATA+="0";
  }
  //Chiller
  if(STATE_CHILLER){
  SEND_DATA+=",";
  SEND_DATA+="1";
  }else{
  SEND_DATA+=",";
  SEND_DATA+="0";
  }
  //Pump
  if(STATE_PUMP){
  SEND_DATA+=",";
  SEND_DATA+="1";
  }else{
  SEND_DATA+=",";
  SEND_DATA+="0";
  }
  /*
  sending format : v1,v2,h,c,p
  valve1,valve2,heater,chiller,pump
```

```
    */
    Serial.println(SEND_DATA);
    AVR_SUHU=0;
    counter=0;
  }
}
//=====================================================
String DOUBLEtoSTRING(double input,int decimalPlaces){
  String string;
  if(decimalPlaces!=0){
   string = String((int)(input*pow(10,decimalPlaces)));
    if(abs(input)<1){
     if(input>0)
       string = "0"+string;
     else if(input<0)
       string = string.substring(0,1)+"0"+string.substring(1);
     }
       return string.substring(0,string.length()-
decimalPlaces)+"."+string.substring(string.length()-decimalPlaces);
     }else{
       return String((int)input);
     }
}
```

```
//=====================================================
String getValue(String data, char separator, int index)
{
  int found = 0;
  int strIndex[] = {0, -1};
  int maxIndex = data.length()-1;

  for(int i=0; i<=maxIndex && found<=index; i++){
   if(data.charAt(i)==separator || i==maxIndex){
      found++;
      strIndex[0] = strIndex[1]+1;
      strIndex[1] = (i == maxIndex) ? i+1 : i;
   }
  }
  return found>index ? data.substring(strIndex[0], strIndex[1]) : "";
}
```

LAMPIRAN 2

Datasheet Arduino MEGA2560

ARDUINO

Arduino Mega 2560 Datasheet

Overview

The Arduino Mega 2560 is a microcontroller board based on the ATmega2560 (datasheet). It has 54 digital input/output pins (of which 14 can be used as PWM outputs), 16 analog inputs, 4 UARTs (hardware serial ports), a 16 MHz crystal oscillator, a USB connection, a power jack, an ICSP header, and a reset button. It contains everything needed to support the microcontroller; simply connect it to a computer with a USB cable or power it with a AC-to-DC adapter or battery to get started. The Mega is compatible with most shields designed for the Arduino Duemilanove or Diecimila.

Schematic & Reference Design

EAGLE files: arduino-mega2560-reference-design.zip

Summary

Microcontroller	ATmega2560
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	54 (of which 14 provide PWM output)
Analog Input Pins	16
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	256 KB of which 8 KB used by bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz

Power

The Arduino Mega can be powered via the USB connection or with an external power supply. The power source is selected automatically.

External (non-USB) power can come either from an AC-to-DC adapter (wall-wart) or battery. The adapter can be connected by plugging a 2.1mm center-positive plug into the board's power jack. Leads from a battery can be inserted in the Gnd and Vin pin headers of the POWER connector.

The board can operate on an external supply of 6 to 20 volts. If supplied with less than 7V, however, the 5V pin may supply less than five volts and the board may be unstable. If using more than 12V, the voltage regulator may overheat and damage the board. The recommended range is 7 to 12 volts.

The Mega2560 differs from all preceding boards in that it does not use the FTDI USB-to-serial driver chip. Instead, it features the Atmega8U2 programmed as a USB-to-serial converter.

The power pins are as follows:

- **VIN.** The input voltage to the Arduino board when it's using an external power source (as opposed to 5 volts from the USB connection or other regulated power source). You can supply voltage through this pin, or, if supplying voltage via the power jack, access it through this pin.
- **5V.** The regulated power supply used to power the microcontroller and other components on the board. This can come either from VIN via an on-board regulator, or be supplied by USB or another regulated 5V supply.
- **3V3.** A 3.3 volt supply generated by the on-board regulator. Maximum current draw is 50 mA.
- **GND.** Ground pins.

Memory

The ATmega2560 has 256 KB of flash memory for storing code (of which 8 KB is used for the bootloader), 8 KB of SRAM and 4 KB of EEPROM (which can be read and written with the EEPROM library).

Input and Output

Each of the 54 digital pins on the Mega can be used as an input or output, using pinMode(), digitalWrite(), and digitalRead() functions. They operate at 5 volts. Each pin can provide or receive a maximum of 40 mA and has an internal pull-up resistor (disconnected by default) of 20-50 kOhms. In addition, some pins have specialized functions:

- **Serial: 0 (RX) and 1 (TX); Serial 1: 19 (RX) and 18 (TX); Serial 2: 17 (RX) and 16 (TX); Serial 3: 15 (RX) and 14 (TX).** Used to receive (RX) and transmit (TX) TTL serial data. Pins 0 and 1 are also connected to the corresponding pins of the ATmega8U2 USB-to-TTL Serial chip.
- **External Interrupts: 2 (interrupt 0), 3 (interrupt 1), 18 (interrupt 5), 19 (interrupt 4), 20 (interrupt 3), and 21 (interrupt 2).** These pins can be configured to trigger an interrupt on a low value, a rising or falling edge, or a change in value. See the attachInterrupt() function for details.
- **PWM: 0 to 13.** Provide 8-bit PWM output with the analogWrite() function.
- **SPI: 50 (MISO), 51 (MOSI), 52 (SCK), 53 (SS).** These pins support SPI communication using the SPI library. The SPI pins are also broken out on the ICSP header, which is physically compatible with the Uno, Duemilanove and Diecimila.
- **LED: 13.** There is a built-in LED connected to digital pin 13. When the pin is HIGH

value, the LED is on, when the pin is LOW, it's off.

- **I2C: 20 (SDA) and 21 (SCL).** Support I2C (TWI) communication using the Wire library (documentation on the Wiring website). Note that these pins are not in the same location as the I2C pins on the Duemilanove or Diecimila.

The Mega2560 has 16 analog inputs, each of which provide 10 bits of resolution (i.e. 1024 different values). By default they measure from ground to 5 volts, though is it possible to change the upper end of their range using the AREF pin and analogReference() function.

There are a couple of other pins on the board:

- **AREF.** Reference voltage for the analog inputs. Used with analogReference().
- **Reset.** Bring this line LOW to reset the microcontroller. Typically used to add a reset button to shields which block the one on the board.

Communication

The Arduino Mega2560 has a number of facilities for communicating with a computer, another Arduino, or other microcontrollers. The ATmega2560 provides four hardware UARTs for TTL (5V) serial communication. An ATmega8U2 on the board channels one of these over USB and provides a virtual com port to software on the computer (Windows machines will need a .inf file, but OSX and Linux machines will recognize the board as a COM port automatically. The Arduino software includes a serial monitor which allows simple textual data to be sent to and from the board. The RX and TX LEDs on the board will flash when data is being transmitted via the ATmega8U2 chip and USB connection to the computer (but not for serial communication on pins 0 and 1).

A SoftwareSerial library allows for serial communication on any of the Mega2560's digital pins.

The ATmega2560 also supports I2C (TWI) and SPI communication. The Arduino software includes a Wire library to simplify use of the I2C bus; see the documentation on the Wiring website for details. For SPI communication, use the SPI library.

Programming

The Arduino Mega can be programmed with the Arduino software (download). For details, see the reference and tutorials.

The ATmega2560 on the Arduino Mega comes preburned with a bootloader that allows you to upload new code to it without the use of an external hardware programmer. It

communicates using the original STK500 protocol (reference, C header files).
You can also bypass the bootloader and program the microcontroller through the ICSP (In-Circuit Serial Programming) header; see these instructions for details.

Automatic (Software) Reset

Rather then requiring a physical press of the reset button before an upload, the Arduino Mega2560 is designed in a way that allows it to be reset by software running on a connected computer. One of the hardware flow control lines (DTR) of the ATmega8U2 is connected to the reset line of the ATmega2560 via a 100 nanofarad capacitor. When this line is asserted (taken low), the reset line drops long enough to reset the chip. The Arduino software uses this capability to allow you to upload code by simply pressing the upload button in the Arduino environment. This means that the bootloader can have a shorter timeout, as the lowering of DTR can be well-coordinated with the start of the upload.
This setup has other implications. When the Mega2560 is connected to either a computer running Mac OS X or Linux, it resets each time a connection is made to it from software (via USB). For the following half-second or so, the bootloader is running on the Mega2560. While it is programmed to ignore malformed data (i.e. anything besides an upload of new code), it will intercept the first few bytes of data sent to the board after a connection is opened. If a sketch running on the board receives one-time configuration or other data when it first starts, make sure that the software with which it communicates waits a second after opening the connection and before sending this data.
The Mega2560 contains a trace that can be cut to disable the auto-reset. The pads on either side of the trace can be soldered together to re-enable it. It's labeled "RESET-EN". You may also be able to disable the auto-reset by connecting a 110 ohm resistor from 5V to the reset line; see this forum thread for details.

USB Overcurrent Protection

The Arduino Mega2560 has a resettable polyfuse that protects your computer's USB ports from shorts and overcurrent. Although most computers provide their own internal protection, the fuse provides an extra layer of protection. If more than 500 mA is applied to the USB port, the fuse will automatically break the connection until the short or overload is removed.

Physical Characteristics and Shield Compatibility

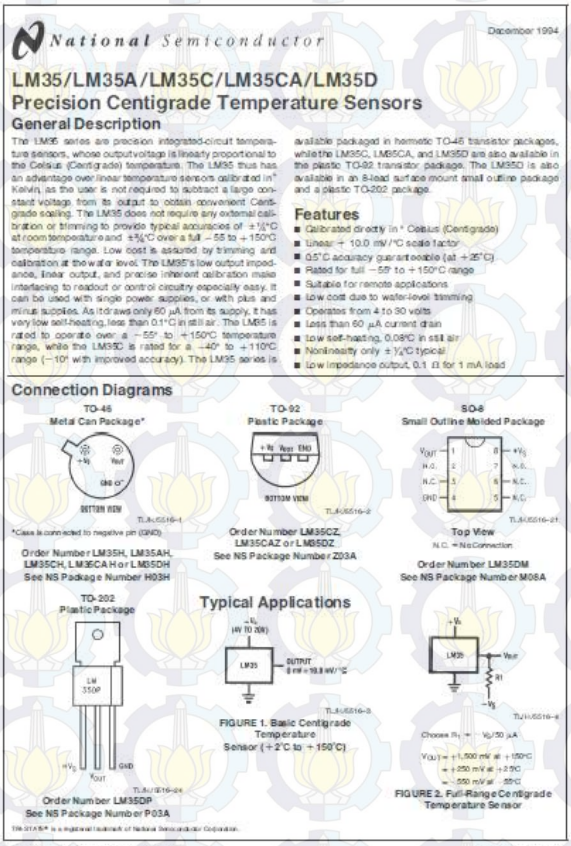
The maximum length and width of the Mega2560 PCB are 4 and 2.1 inches respectively, with the USB connector and power jack extending beyond the former dimension. Three screw holes allow the board to be attached to a surface or case. Note that the distance between digital pins 7 and 8 is 160 mil (0.16"), not an even multiple of the 100 mil spacing of the other pins.

The Mega2560 is designed to be compatible with most shields designed for the Uno, Diecimila or Duemilanove. Digital pins 0 to 13 (and the adjacent AREF and GND pins), analog inputs 0 to 5, the power header, and ICSP header are all in equivalent locations. Further the main UART (serial port) is located on the same pins (0 and 1), as are external interrupts 0 and 1 (pins 2 and 3 respectively). SPI is available through the ICSP header on both the Mega2560 and Duemilanove / Diecimila. *Please note that I2C is not located on the same pins on the Mega (20 and 21) as the Duemilanove / Diecimila (analog inputs 4 and 5).*

Halaman ini sengaja di kosongkan

LAMPIRAN 3

Datasheet LM35



National Semiconductor

December 1994

LM35/LM35A/LM35C/LM35CA/LM35D
Precision Centigrade Temperature Sensors

General Description

The LM35 series are precision integrated-circuit temperature sensors, whose output voltage is linearly proportional to the Celsius (Centigrade) temperature. The LM35 thus has an advantage over linear temperature sensors calibrated in ° Kelvin, as the user is not required to subtract a large constant voltage from its output to obtain convenient Centigrade scaling. The LM35 does not require any external calibration or trimming to provide typical accuracies of ±¼°C at room temperature and ±¾°C over a full −55 to +150°C temperature range. Low cost is assured by trimming and calibration at the wafer level. The LM35's low output impedance, linear output, and precise inherent calibration make interfacing to readout or control circuitry especially easy. It can be used with single power supplies, or with plus and minus supplies. As it draws only 60 µA from its supply, it has very low self-heating, less than 0.1°C in still air. The LM35 is rated to operate over a −55° to +150°C temperature range, while the LM35C is rated for a −40° to +110°C range (−10° with improved accuracy). The LM35 series is available packaged in hermetic TO-46 transistor packages, while the LM35C, LM35CA, and LM35D are also available in the plastic TO-92 transistor package. The LM35D is also available in an 8-lead surface mount small outline package and a plastic TO-202 package.

Features

- Calibrated directly in ° Celsius (Centigrade)
- Linear + 10.0 mV/°C scale factor
- 0.5°C accuracy guaranteeable (at +25°C)
- Rated for full −55° to +150°C range
- Suitable for remote applications
- Low cost due to wafer-level trimming
- Operates from 4 to 30 volts
- Less than 60 µA current drain
- Low self-heating, 0.08°C in still air
- Nonlinearity only ±¼°C typical
- Low impedance output, 0.1 Ω for 1 mA load

Connection Diagrams

TO-46
Metal Can Package*

BOTTOM VIEW

TL/H/5516–1

*Case is connected to negative pin (GND)

Order Number LM35H, LM35AH, LM35CH, LM35CAH or LM35DH
See NS Package Number H03H

TO-92
Plastic Package

BOTTOM VIEW

TL/H/5516–2

Order Number LM35CZ, LM35CAZ or LM35DZ
See NS Package Number Z03A

SO-8
Small Outline Molded Package

Top View

TL/H/5516–21

N.C. = No Connection

Order Number LM35DM
See NS Package Number M08A

TO-202
Plastic Package

Order Number LM35DP
See NS Package Number P03A

Typical Applications

TL/H/5516–3

FIGURE 1. Basic Centigrade Temperature Sensor (+2°C to +150°C)

TL/H/5516–4

Choose $R_1 = -V_S/50\ \mu A$

$V_{OUT} = +1,500\ mV$ at $+150°C$
$= +250\ mV$ at $+25°C$
$= -550\ mV$ at $-55°C$

FIGURE 2. Full-Range Centigrade Temperature Sensor

TRI-STAT® is a registered trademark of National Semiconductor Corporation.

Absolute Maximum Ratings (Note 10)

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

Supply Voltage	+35V to −0.2V
Output Voltage	+6V to −1.0V
Output Current	10 mA
Storage Temp., TO-46 Package,	−60°C to +180°C
TO-92 Package,	−60°C to +150°C
SO-8 Package,	−65°C to +150°C
TO-202 Package,	−65°C to +150°C
Lead Temp.:	
TO-46 Package, (Soldering, 10 seconds)	300°C
TO-92 Package, (Soldering, 10 seconds)	260°C
TO-202 Package, (Soldering, 10 seconds)	+230°C
SO Package (Note 12):	
Vapor Phase (60 seconds)	215°C
Infrared (15 seconds)	220°C
ESD Susceptibility (Note 11)	2500V
Specified Operating Temperature Range: T_{MIN} to T_{MAX} (Note 2)	
LM35, LM35A	−55°C to +150°C
LM35C, LM35CA	−40°C to +110°C
LM35D	0°C to +100°C

Electrical Characteristics (Note 1) (Note 6)

Parameter	Conditions	LM35A			LM35CA			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy (Note 7)	T_A= +25°C	±0.2	±0.5		±0.2	±0.5		°C
	T_A= −10°C	±0.3			±0.3		±1.0	°C
	T_A= T_{MAX}	±0.4	±1.0		±0.4	±1.0		°C
	T_A= T_{MIN}	±0.4	±1.0		±0.4		±1.5	°C
Nonlinearity (Note 8)	$T_{MIN} \le T_A \le T_{MAX}$	**±0.18**		**±0.35**	**±0.15**		**±0.3**	°C
Sensor Gain (Average Slope)	$T_{MIN} \le T_A \le T_{MAX}$	**+10.0**	**+9.9, +10.1**		**+10.0**		**+9.9, +10.1**	mV/°C
Load Regulation (Note 3) 0 ≤ I_L ≤ 1 mA	T_A= +25°C	±0.4	±1.0		±0.4	±1.0		mV/mA
	$T_{MIN} \le T_A \le T_{MAX}$	**±0.5**		**±3.0**	**±0.5**		**±3.0**	mV/mA
Line Regulation (Note 3)	T_A= +25°C	±0.01	±0.05		±0.01	±0.05		mV/V
	4V ≤ V_S ≤ 30V	**±0.02**		**±0.1**	**±0.02**		**±0.1**	mV/V
Quiescent Current (Note 9)	V_S= +5V, +25°C	56	67		56	67		µA
	V_S= +5V	**105**		**131**	**91**		**114**	µA
	V_S= +30V, +25°C	56.2	68		56.2	68		µA
	V_S= +30V	**105.5**		**133**	**91.5**		**116**	µA
Change of Quiescent Current (Note 3)	4V ≤ V_S ≤ 30V, +25°C	0.2	1.0		0.2	1.0		µA
	4V ≤ V_S ≤ 30V	**0.5**		**2.0**	**0.5**		**2.0**	µA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.5**	**+0.39**		**+0.5**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, I_L = 0	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	T_J= T_{MAX}, for 1000 hours	±0.08			±0.08			°C

Note 1: Unless otherwise noted, these specifications apply: −55°C ≤ T_J ≤ +150°C for the LM35 and LM35A; −40°C ≤ T_J ≤ +110°C for the LM35C and LM35CA; and 0°C ≤ T_J ≤ +100°C for the LM35D. V_S = +5Vdc and I_{LOAD} = 50 µA, in the circuit of *Figure 2*. These specifications also apply from +2°C to T_{MAX} in the circuit of *Figure 1*. Specifications in boldface apply over the full rated temperature range.

Note 2: Thermal resistance of the TO-46 package is 400°C/W, junction to ambient, and 24°C/W junction to case. Thermal resistance of the TO-92 package is 180°C/W junction to ambient. Thermal resistance of the small outline molded package is 220°C/W junction to ambient. Thermal resistance of the TO-202 package is 65°C/W junction to ambient. For additional thermal resistance information see table in the Applications section.

Electrical Characteristics (Note 1) (Note 6) (Continued)

Parameter	Conditions	LM35			LM35C, LM35D			Units (Max.)
		Typical	Tested Limit (Note 4)	Design Limit (Note 5)	Typical	Tested Limit (Note 4)	Design Limit (Note 5)	
Accuracy, LM35, LM35C (Note 7)	$T_A = +25°C$	±0.4	±1.0		±0.4	±1.0		°C
	$T_A = -10°C$	±0.5			±0.5		±1.5	°C
	$T_A = T_{MAX}$	±0.8	±1.5		±0.8		±1.5	°C
	$T_A = T_{MIN}$	±0.8		±1.5	±0.8		±2.0	°C
Accuracy, LM35D (Note 7)	$T_A = +25°C$				±0.6	±1.5		°C
	$T_A = T_{MAX}$				±0.9		±2.0	°C
	$T_A = T_{MIN}$				±0.9		±2.0	°C
Nonlinearity (Note 8)	$T_{MIN} \le T_A \le T_{MAX}$	**±0.3**		**±0.5**	**±0.2**		**±0.5**	°C
Sensor Gain (Average Slope)	$T_{MIN} \le T_A \le T_{MAX}$	**+10.0**	**+9.8, +10.2**		**+10.0**		**+9.8, +10.2**	mV/°C
Load Regulation (Note 9) $0 \le I_L \le 1$ mA	$T_A = +25°C$	±0.4	±2.0		±0.4	±2.0		mV/mA
	$T_{MIN} \le T_A \le T_{MAX}$	**±0.5**		**±5.0**	**±0.5**		**±5.0**	mV/mA
Line Regulation (Note 9)	$T_A = +25°C$	±0.01	±0.1		±0.01	±0.1		mV/V
	$4V \le V_S \le 30V$	**±0.02**		**±0.2**	**±0.02**		**±0.2**	mV/V
Quiescent Current (Note 9)	$V_S = +5V, +25°C$	56	80		56	80		µA
	$V_S = +5V$	**105**		**158**	**91**		**138**	µA
	$V_S = +30V, +25°C$	56.2	82		56.2	82		µA
	$V_S = +30V$	**105.5**		**161**	**91.5**		**141**	µA
Change of Quiescent Current (Note 9)	$4V \le V_S \le 30V, +25°C$	0.2	2.0		0.2	2.0		µA
	$4V \le V_S \le 30V$	**0.5**		**3.0**	**0.5**		**3.0**	µA
Temperature Coefficient of Quiescent Current		**+0.39**		**+0.7**	**+0.39**		**+0.7**	µA/°C
Minimum Temperature for Rated Accuracy	In circuit of *Figure 1*, $I_L = 0$	+1.5		+2.0	+1.5		+2.0	°C
Long Term Stability	$T_J = T_{MAX}$ for 1000 hours	±0.08			±0.08			°C

Note 3: Regulation is measured at constant junction temperature, using pulse testing with a low duty cycle. Changes in output due to heating effects can be computed by multiplying the internal dissipation by the thermal resistance.

Note 4: Tested Limits are guaranteed and 100% tested in production.

Note 5: Design Limits are guaranteed (but not 100% production tested) over the indicated temperature and supply voltage ranges. These limits are not used to calculate outgoing quality levels.

Note 6: Specifications in boldface apply over the full rated temperature range.

Note 7: Accuracy is defined as the error between the output voltage and 10mv/°C times the device's case temperature, at specified conditions of voltage, current, and temperature (expressed in °C).

Note 8: Nonlinearity is defined as the deviation of the output-voltage-versus-temperature curve from the best-fit straight line, over the device's rated temperature range.

Note 9: Quiescent current is defined in the circuit of *Figure 1*.

Note 10: Absolute Maximum Ratings indicate limits beyond which damage to the device may occur. DC and AC electrical specifications do not apply when operating the device beyond its rated operating conditions. See Note 1.

Note 11: Human body model, 100 pF discharged through a 1.5 kΩ resistor.

Note 12: See AN-450 "Surface Mounting Methods and Their Effect on Product Reliability" or the section titled "Surface Mount" found in a current National Semiconductor Linear Data Book for other methods of soldering surface mount devices.

Halaman ini sengaja di kosongkan

RIWAYAT HIDUP PENULIS

Nama : Guntur Rangga k
TTL : surabaya, 4 Maret 1993
Jenis Kelamin : Laki - laki
Agama : Islam
Alamat Rumah: sidoyoso Wetan 1/41, Surabaya
Telp/HP : 085732641410
E-mail :ranggamania@gmail.com
Hobi : Masak dan membaca buku

RIWAYAT PENDIDIKAN

- 1998 – 2004 : SD NEGERI RANGKAH 6 SURABAYA
- 2004 – 2007 : SMP NEGERI 11 SURABAYA
- 2007 – 2008 : SMA MUHAMMADIYAH 2 SURABAYA
- 2010 – sekarang : Bidang Studi Komputer Kontrol, Program D3 Teknik Elektro, ITS

PENGALAMAN KERJA

- Kerja Praktek di PETROKIMIA GRESIK

PENGALAMAN ORGANISASI

-

Halaman ini sengaja dikosongkan

Nama	:	Dwi Febria Herdiana
TTL	:	Madiun, 2 Februari 1991
Jenis Kelamin	:	Perempuan
Agama	:	Islam
Alamat Rumah	:	Jalan Dwijaya 10 no.1, Madiun
Telp/HP	:	0838 544 305 47
E-mail	:	dwi.febria10@mhs.ee.its.ac.id
Hobi	:	Membaca, menulis

RIWAYAT PENDIDIKAN

- 1997– 2003 : SD NEGERI KARTOHARJO MADIUN
- 2003 – 2006 : SMP NEGERI 4 MADIUN
- 2006 – 2009 : SMA NEGERI 1 MADIUN
- 2010 – sekarang : Bidang Studi Komputer Kontrol, Program D3 Teknik Elektro, ITS

PENGALAMAN KERJA

- Kerja Praktek di PT. INKA MADIUN

PENGALAMAN ORGANISASI

-

Halaman ini sengaja dikosongkan