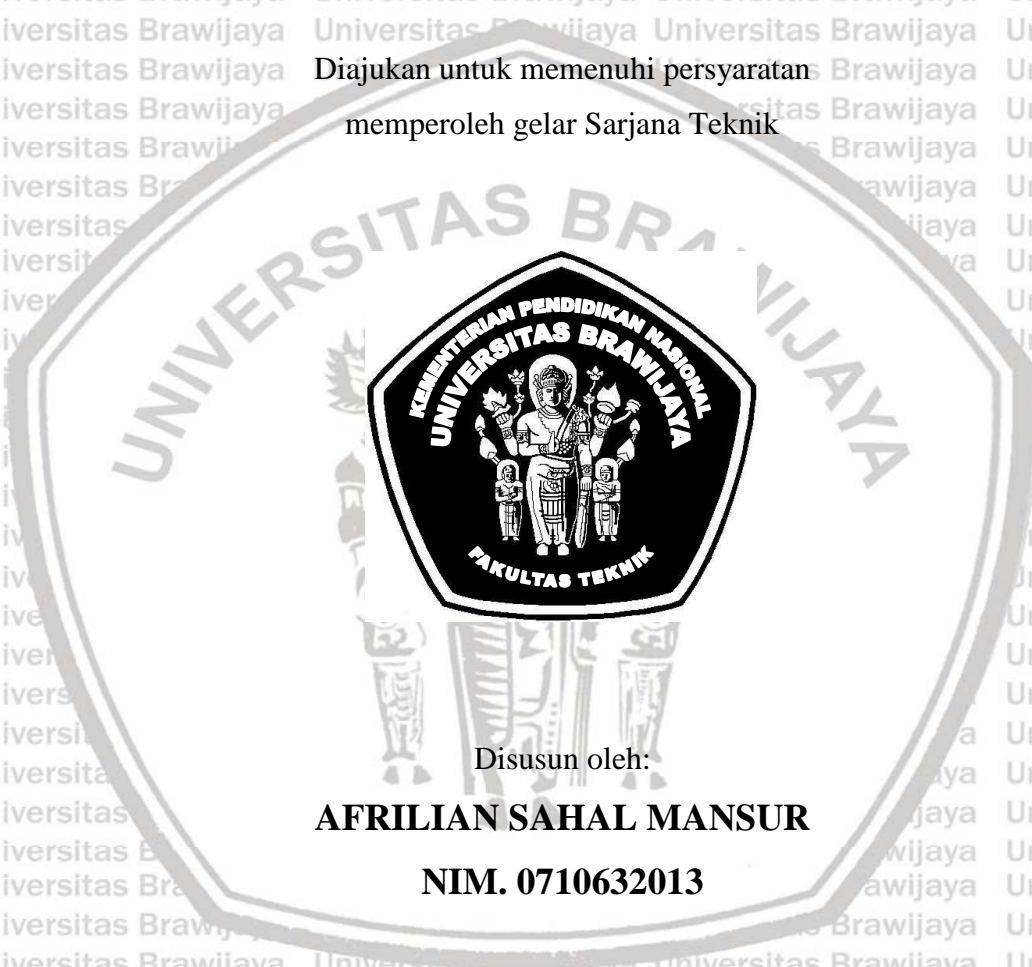


**ALAT OTOMATISASI KRAN DAN PEMANAS AIR
DISPENSER BERBASIS MIKROKONTROLER ATMEGA8535**

**SKRIPSI
KONSENTRASI TEKNIK ELEKTRONIKA**

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik



Disusun oleh:

AFRILIAN SAHAL MANSUR

NIM. 0710632013

**KEMENTERIAN PENDIDIKAN NASIONAL
UNIVERSITAS BRAWIJAYA
FAKULTAS TEKNIK**

MALANG

2011





LEMBAR PERSETUJUAN

ALAT OTOMATISASI KRAN DAN PEMANAS AIR DISPENSER BERBASIS MIKROKONTROLER ATMEGA8535

SKRIPSI

KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik



Disusun oleh:

AFRILIAN SAHAL MANSUR

NIM. 0710632013

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Ir. M. Julius St., MS.

Mochammad Rif'an, ST., MT.

NIP 19540720 198203 1 002

NIP 19710301 200012 1 001



LEMBAR PENGESAHAN

ALAT OTOMATISASI KRAN DAN PEMANAS AIR DISPENSER BERBASIS MIKROKONTROLER ATMEGA8535

SKRIPSI KONSENTRASI TEKNIK ELEKTRONIKA

Diajukan untuk memenuhi persyaratan
memperoleh gelar Sarjana Teknik

Disusun oleh:

AFRILIAN SAHAL M.

NIM. 0710632013

Skripsi ini telah diuji dan dinyatakan lulus pada
Tanggal 11 Agustus 2011

Dosen Penguji

Panca Mudjiraharjo, ST., MT.
NIP. 19700329 200012 001

Ir. Ponco Siwindarto, M.Eng.Sc.
NIP. 19590304 198903 1 001

Ir. Nurussa'adah, MT.
NIP 19680706 199203 2 001

Mengetahui,
Ketua Jurusan Teknik Elektro

Rudy Yuwono ST., M.Sc.
NIP. 19710615 199802 1 003

PENGANTAR

Puji syukur kehadirat Allah SWT atas rahmat, hidayah, ridho, nikmat, dan karunia-Nya serta kekuatan-Nya sehingga penulis dapat menyelesaikan skripsi yang berjudul “*Alat Otomatisasi Kran Dan Pemanas Air Dispenser Berbasis Mikrokontroler ATmega8535*” dengan lancar dan baik.

Penyusunan skripsi ini diajukan untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Teknik di jurusan Teknik Elektro Fakultas Teknik Universitas Brawijaya. Selama penyelesaian skripsi ini penulis menyadari adanya dorongan dan bantuan dari berbagai pihak, untuk itu penulis menyampaikan ucapan terima kasih yang sebesar-besarnya kepada:

- 1) Bapak Rudy Yuwono, ST., MSc. Ketua Jurusan Teknik Elektro,
- 2) Bapak M. Aziz Muslim, ST., MT., Ph.D. Sekretaris Jurusan Teknik Elektro,
- 3) Bapak Ir. Moch. Julius St, MT., Ketua Kelompok Dosen Keahlian Program Studi Paket B (Teknik Elektronika) Jurusan Teknik Elektro,
- 4) Bapak Ir. Moch. Julius St, MT., pembimbing I yang telah banyak memberikan pengarahan dan bantuan dalam menyelesaikan skripsi ini,
- 5) Bapak Moch. Rif'an, ST., MT. pembimbing II yang telah banyak memberikan masukan dan bantuan dalam menyelesaikan skripsi ini,
- 6) Keluarga tercinta, Kharisma Debby P, Teman-teman P6 terima kasih atas doa, kasih sayang, perhatian, serta dukungan baik materi maupun non-materi yang tak ternilai yang telah diberikan,
- 7) Semua pihak yang tidak dapat penulis sebutkan satu per satu.

Penulis menyadari ketidaksempurnaan dalam penyusunan skripsi ini, sehingga penulis sangat mengharapkan kritik dan saran dari pembaca untuk dapat memperbaiki segala kekurangan di dalam skripsi ini. Akhirnya, semoga skripsi ini dapat memberikan manfaat dan berguna bagi pembaca.

Malang, 2 Agustus 2011

Penulis

ABSTRAK

AFRILIAN SAHAL M, 2011. *Alat Otomatisasi Kran Dan Pemanas Air Dispenser Berbasis Mikrokontroler ATmega8535.*

Skripsi, Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya.

Dosen Pembimbing : Ir. M Julius St., MS. dan Mochammad Rif'an, ST., MT.

Pada saat ini, *dispenser* yang ada pada umumnya mempunyai fungsi dan prinsip kerja alat yang masih manual khususnya pada kran air. Melihat kekurangan tersebut, menjadikan kekurangan itu sebagai dasar dalam mengembangkan sistem kerja dispenser yang akan dibuat. Dalam proyek akhir ini akan memperbaharui sistem manual kran aliran air menjadi sistem otomatis, sehingga dalam pengambilan air tidak perlu menekan kran, hanya menombol menu yang ada, maka air akan keluar dengan sendiri.

Proses otomatisasi pengambilan air terjadi pada saat mengatur volume dan suhu air menggunakan tombol push button, dimana ada dua menu, yaitu menu volume dan menu suhu. Pada saat tombol tersebut di set maka akan memberikan sinyal input pada mikrokontroler, oleh mikrokontroler sinyal tersebut diproses untuk mengaktifkan kran solenoid sehingga air dapat keluar dengan volume dan presentase suhu yang diinginkan. Sistem pengontrol dari alat ini menggunakan mikrokontroler ATmega 8535 sebagai pemroses data dan sebagai input dari mikrokontroler berupa tombol yaitu tombol volume dan tombol suhu. sedangkan unit/bagian yang akan dikontrol adalah kran air dan pemanas (heater).

Dengan adanya alat ini diharapkan para pengguna dengan mudah menggunakan dispenser dengan mengeluarkan air sesuai dengan keinginan dan takaran yang sudah disesuaikan. Alat yang dirancang mampu mengeluarkan air panas dan biasa sesuai dengan volume yang diinginkan antara 100 ml, 150 ml,.....500 ml dengan kesalahan rata-rata 4,2 %.

KATA KUNCI : *Dispenser*, Sensor Suhu, Volume, Mikrokontroler ATMEGA8535.

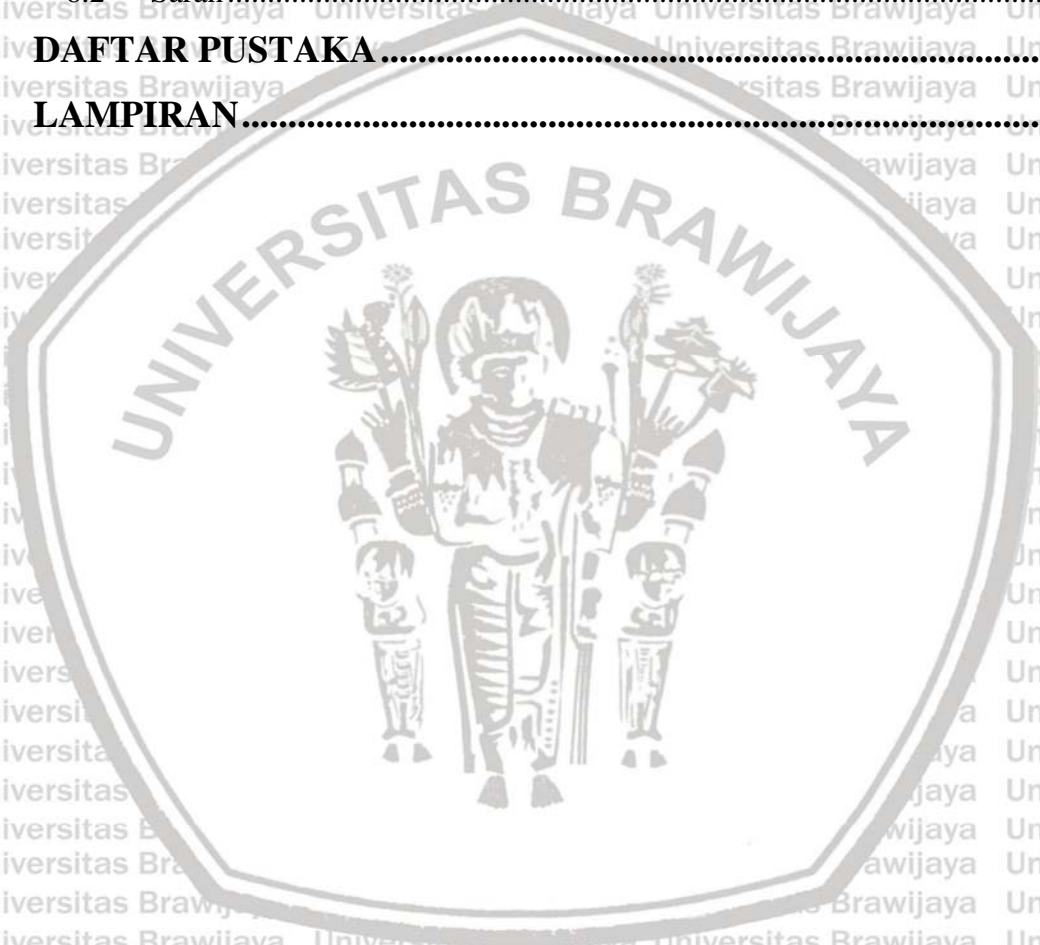
DAFTAR ISI

HALAMAN JUDUL	ix
LEMBAR PERSETUJUAN	ixi
LEMBAR PENGESAHAN	ixii
PENGANTAR	iv
ABSTRAK	v
DAFTAR ISI	vi
DAFTAR GAMBAR	ix
DAFTAR TABEL	x
BAB I PENDAHULUAN	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Ruang Lingkup	2
1.4. Tujuan	3
1.5. Sistematika Penulisan	3
BAB II TINJAUAN PUSTAKA	5
2.1. Tinjauan Sistem	5
2.2. Mikrokontroler AVR ATmega 8535	5
2.2.1. Konfigurasi Pin ATmega 8535	6
2.2.2. Organisasi Memori	8
2.3. Sensor Suhu LM 35	10
2.4. Liquid Crystal Display (LCD)	11
2.4.1. Instruksi Operasi Dasar	15
2.5. Transistor Sebagai Saklar	17
2.6. Push Button (Tombol tekan)	18
2.7. Solenoid	18
2.8. Relay	19
BAB III METODE PENELITIAN	20
3.1. Perancangan Alat	20
3.2. Perancangan Perangkat Lunak	21



3.3.	Pengujian Alat	21
BAB IV PERANCANGAN DAN PEMBUATAN ALAT		23
4.1.	Penentuan Spesifikasi Alat	23
4.2.	Perancangan Perangkat Keras	23
4.2.1	Rangkaian Mikrokontroler AVR ATmega 8535	25
4.2.2.	Rangkaian Push Button (Tombol Tekan).....	25
4.2.3.	Rangkaian Selenoid.....	26
4.2.4	Rangkain <i>Heater</i>	28
4.2.5.	Display LCD (<i>Liquid Crystal Display</i>).....	29
4.2.6.	Rangkaian Sensor Suhu LM35	30
4.2.7.	Perancangan Perangkat Lunak	31
4.2.8.	Flowchart Program.....	32
BAB V PENGUJIAN DAN PEMBAHASAN		33
5.1.	Pengujian Sensor Suhu	33
5.1.1	Tujuan Pengujian	33
5.1.2.	Peralatan Pengujian.....	33
5.1.3.	Prosedur Pengujian	33
5.1.4.	Hasil Pengujian	34
5.2.	Pengujian LCD	35
5.2.1.	Tujuan Pengujian	35
5.2.2.	Peralatan Pengujian.....	35
5.2.3.	Prosedur Pengujian	35
5.3.	Pengujian Rangkaian Drive Relay	36
5.3.1.	Tujuan Pengujian	36
5.3.2.	Peralatan Pengujian.....	36
5.3.3.	Prosedur Pengujian	37
5.3.4.	Hasil Pengujian	37
5.4.	Pengujian Volume Air	38
5.4.1.	Tujuan Pengujian	38
5.4.2.	Peralatan Pengujian.....	38
5.4.3.	Prosedur Pengujian	38
5.4.4.	Hasil Pengujian	38

5.5. Pengujian Sistem Secara Keseluruhan	39
5.5.1. Tujuan Pengujian	39
5.5.2. Peralatan Pengujian	39
5.5.3. Prosedur Pengujian	40
5.5.4. Hasil Pengujian Sistem Keseluruhan	40
BAB VI PENUTUP	42
6.1. Kesimpulan	42
6.2. Saran	42
DAFTAR PUSTAKA	44
LAMPIRAN	445

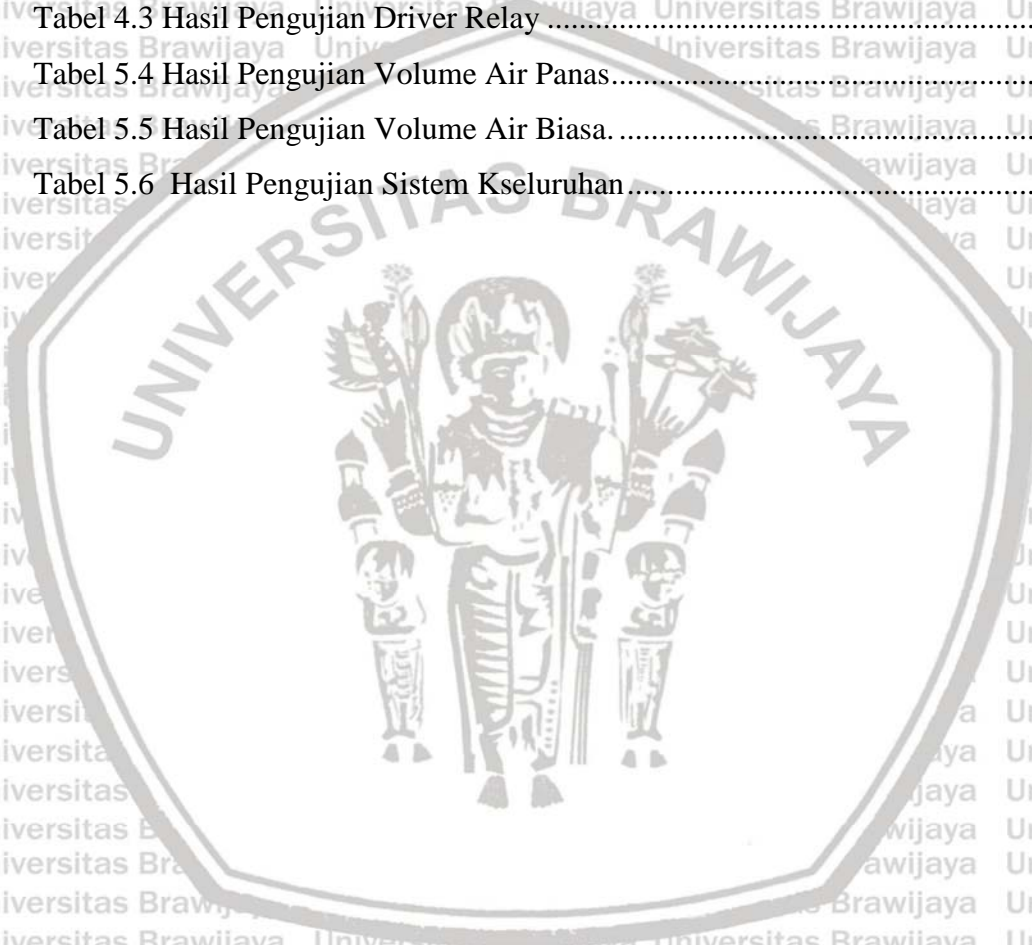


DAFTAR GAMBAR

Gambar 2.1 Diagram Blok AT Mega 8535.....	6
Gambar 2.2 Konfigurasi Pin ATmega 8535	8
Gambar 2.3 Konfigurasi Memori Data AVR ATmega 8535	9
Gambar 2.4 Memori Program AVR ATmega 8535	9
Gambar 2.5 Sensor Suhu LM 35.....	11
Gambar 2.6 Diagram Blok LCD.....	12
Gambar 2.7 Rangkaian Transistor dan Garis Beban.....	17
Gambar 2.8 Simbol Relay.....	19
Gambar 4.1 Diagram Blok Otomatisasi Kran dan Pemanas Air pada <i>Dispenser</i>	24
Gambar 4.2 Mikrokontroler sebagai Sistem Minimum.	25
Gambar 4.3 Rangkaian <i>Push Button</i>	26
Gambar 4.4 Rangkaian Selenoid.....	26
Gambar 4.5 Driver Selenoid	27
Gambar 4.6 Driver <i>Heater</i>	28
Gambar 4.7 Rangkain LCD (<i>Liquid Crystal Display</i>).....	30
Gambar 4.8 Rangkaian Sensor Suhu LM35.....	31
Gambar 4.9 <i>Flowchart</i> Program dari Keseluruhan Sistem.....	32
Gambar 5.1 Rangkaian Pengujian Sensor Suhu.....	34
Gambar 5.2 Rangkaian Pengujian LCD.....	36
Gambar 5.3 Rangkaian Pengujian Driver Relay.....	37
Gambar 5.4 Blok Diagram Secara Keseluruhan	40

DAFTAR TABEL

Tabel 2.1 Konfigurasi Pin-pin LCD	13
Tabel 2.2 Instruksi pada Modul LCD	13
Tabel 2.3 Register Seleksi	15
Tabel 2.4 Fungsi Terminal pada LCD.....	16
Tabel 4.1 Konfigurasi Pin-pin LCD.....	30
Tabel 5.1 Hasil Pengujian Sample Suhu	34
Tabel 4.3 Hasil Pengujian Driver Relay	37
Tabel 5.4 Hasil Pengujian Volume Air Panas.....	38
Tabel 5.5 Hasil Pengujian Volume Air Biasa.....	39
Tabel 5.6 Hasil Pengujian Sistem Kseluruhan.....	41



BAB I

PENDAHULUAN

1.1. Latar Belakang

Di jaman yang semakin maju sekarang ini, persaingan di antara produsen penghasil barang kebutuhan manusia semakin ketat. Salah satu contoh, yaitu perusahaan yang memproduksi alat-alat kebutuhan rumah tangga, karena kini banyak dijual peralatan rumah tangga yang menawarkan keunggulan yang berbeda beda dari tiap produsen panghasil barang tersebut. Banyak cara yang ditempuh produsen pembuat alat-alat rumah tangga dalam menarik minat masyarakat, seperti memperkenalkan alat baru yang belum ada sebelumnya atau mengembangkan sistem kerja alat yang telah ada agar lebih baik lagi. Secara tidak langsung dengan adanya persaingan antar produsen penghasil peralatan rumah tangga kemajuan teknologi juga akan berkembang. Sedangkan alat yang akan dikembangkan pada proyek akhir ini berupa alat rumah tangga yang kini telah banyak dipasarkan yaitu *dispenser*.

Dispenser yang ada sekarang umumnya mempunyai fungsi dan prinsip kerja alat yang masih manual khususnya pada kran air. Melihat kekurangan tersebut, menjadikan kekurangan itu sebagai dasar dalam mengembangkan sistem kerja *dispenser* yang akan dibuat. Dalam proyek akhir ini akan memperbaharui sistem manual kran aliran air menjadi sistem otomatis, sehingga dalam pengambilan air tidak perlu menekan kran, hanya menombol menu yang ada, maka air akan keluar dengan sendiri. Serta yang utama adalah sistem pengaturan volume air yang akan dikeluarkan, dimana dengan menggunakan solenoid valve, mengatur berapa besar volume air yang akan di dikeluarkan, dengan begitu jumlah air yang d dikeluarkan akan tepat tidak akan kurang atau berlebihan.

Proses otomatisasi pengambilan air terjadi pada saat mengatur volume dan suhu air menggunakan push button, dimana ada dua menu, yaitu menu volume dan menu suhu. Pada saat tersebut di set maka akan memberikan sinyal input pada mikrokontroler, oleh mikrokontroler sinyal tersebut diproses untuk mengaktifkan kran solenoid sehingga air dapat keluar dengan volume dan presentase suhu yang diinginkan. Sistem pengontrol dari alat ini menggunakan mikrokontroler AVR

ATMega 8535 sebagai pemroses data dan sebagai input dari mikrokontroler berupa tombol yaitu tombol volume dan tombol suhu. sedangkan unit/bagian yang akan dikontrol adalah kran air dan pemanas (*heater*).

1.2. Rumusan Masalah

Berdasarkan uraian dari latar belakang, rumusan masalah yang dapat diambil secara umum yaitu bagaimana merancang rangkaian otomatisasi kran dan pemanas air pada *dispenser* berbasis mikrokontroler. Rumusan masalah secara khusus tersebut dijabarkan sebagai berikut :

- a) Bagaimana merancang dan membuat sistem pemanas air otomatis pada *dispenser*?
- b) Bagaimana merancang dan menentukan volume air pada *dispenser*.
- c) Bagaimana merancang dan membuat rangkaian *input* (*berupa tombol suhu, volume dan sensor suhu*), rangkaian unit proses (minimum sistem dari mikrokontroler AVR ATMega 8535), dan rangkaian *out put* (*LCD, driver kran solenoid dan driver pemanas*) pada *dispenser*?
- d) Bagaimana merancang dan membuat *listing program* kerja alat yang akan dijalankan oleh mikrokontroler AVR ATMega 8535 untuk mengontrol sistem?

1.3. Ruang Lingkup

Berdasarkan permasalahan yang telah dirumuskan diatas maka diberikan batasan masalah sebagai berikut :

- a) Suhu panas yang digunakan standart sesuai dengan alat (*dispenser*) pada umumnya.
- b) Sistem otomatisasi *dispenser* ini menggunakan AVR ATMega8535.

- c) Sistem otomatisasi dispenser ini menggunakan komposisi air panas dan air dingin yang dicampur dalam 1 bejana dengan menentukan volume air dalam bejana serta mengatur keluarnya air melalui kran.
- d) Komposisi dan volume air pada sistem ini menggunakan timer yang memungkinkan terdapat kesalahan-kesalahan karena air yang keluar melalui kran berdasarkan volume air pada galon dan gravitasi.

1.4. Tujuan

Tujuan penelitian ini adalah merancang otomatisasi kran dan pemanas air pada dispenser berbasis mikrokontroler AVR ATmega 8535. Sedangkan tujuan perancangan proyek akhir secara khusus adalah sebagai berikut :

- a) Merancang dan membuat sistem pemanas air otomatis pada *dispenser*.
- b) Merancang dan menentukan volume air pada dispenser.
- c) Merancang dan membuat rangkaian input (*berupa tombol suhu, volume dan sensor suhu*), rangkaian unit proses (minimum sistem dari mikrokontroler AVR ATmega 8535), dan rangkaian *output* (*LCD, driver kran solenoid dan driver pemanas*) pada *dispenser*?
- d) Membuat *listing program* kerja alat yang akan dijalankan oleh mikrokontroler AVR ATmega 8535 untuk mengontrol sistem.

1.5. Sistematika Penulisan

Sistematika penulisan dalam skripsi ini adalah sebagai berikut:

BAB I : Pendahuluan

Memuat latar belakang, rumusan masalah, tujuan, batasan masalah, metodologi penelitian dan sistematika penulisan

BAB II : Teori Penunjang

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

BAB III : Metodologi

Berisi tentang metode penelitian yang digunakan dalam perencanaan dan pengujian alat.

BAB IV : Perencanaan dan Pembuatan Alat

Perancangan dan perealisasiian alat yang meliputi spesifikasi, perencanaan blok diagram, prinsip kerja dan realisasi alat.

BAB V : Pengujian Alat

Memuat hasil pengujian terhadap alat yang telah direalisasikan

BAB VI Kesimpulan dan Saran

Memuat kesimpulan dan saran-saran untuk pengembangan lebih lanjut dari alat yang telah dibuat.





BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Sistem

Dispenser yang ada sekarang umumnya mempunyai fungsi dan prinsip kerja alat yang masih manual khususnya pada kran air. Melihat kekurangan tersebut, menjadikan kekurangan itu sebagai dasar dalam mengembangkan sistem kerja *dispenser* yang akan dibuat.

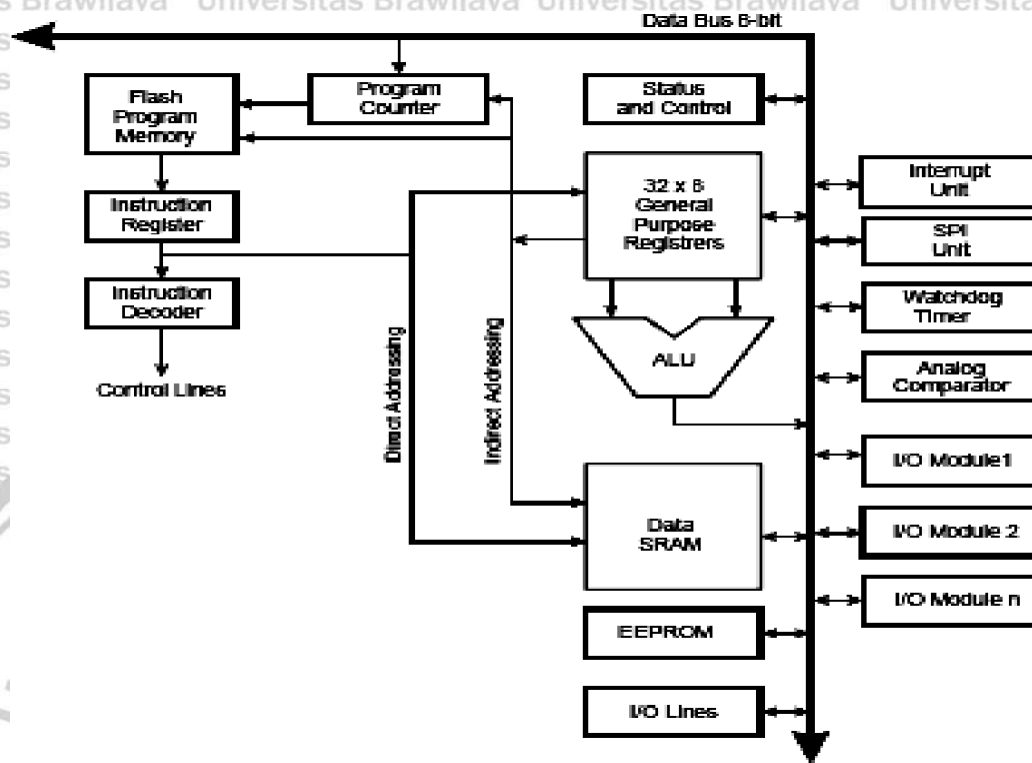
Proses otomatisasi pengambilan air terjadi pada saat mengatur volume dan suhu air menggunakan push button, dimana ada dua menu, yaitu menu volume dan menu suhu. Pada saat tersebut di set maka akan memberikan sinyal input pada mikrokontroler, oleh mikrokontroler sinyal tersebut diproses untuk mengaktifkan kran solenoid sehingga air dapat keluar dengan volume dan presentase suhu yang diinginkan. Sistem pengontrol dari alat ini menggunakan mikrokontroler AVR ATMega 8535.

2.2. Mikrokontroler AVR ATMega 8535

Berdasarkan fungsinya, mikrokontroler secara umum digunakan untuk menjalankan program yang bersifat permanen pada sebuah aplikasi yang spesifik (misal aplikasi yang berkaitan dengan pengontrolan dan monitoring). Sedangkan program aplikasi yang dijalankan pada sistem mikroprosesor biasanya bersifat sementara dan berorientasi pada pengolahan data. Perbedaan fungsi kedua sistem diatas secara praktis mengakibatkan kebutuhan minimal yang harus dipenuhi juga akan berbeda (misal ditinjau dari kecepatan detak operasi, jumlah RAM, panjang register, dan lain sebagainya). Untuk sistem mikrokontroler, program yang dijalankan biasanya tidak memerlukan sumber daya sebanyak dan sebesar itu.

Untuk aplikasi kontrol sederhana dan tingkat menengah, mikrokontroler yang digunakan cukup berbasis 4 sampai 8 bit. Mikrokontroler dengan ukuran lebih besar (misal 16 dan 32 bit) umumnya hanya digunakan untuk aplikasi-aplikasi

khusus pada bidang pengolahan citra atau bidang kontrol yang memerlukan kepresisian tinggi. Diagram blok ATmega 8535 ditunjukkan dalam Gambar 2.1.



Gambar 2.1 Diagram Blok AT Mega 8535

Mikrokontroler atau mikroprosesor adalah suatu piranti yang digunakan untuk mengolah data-data biner (digital) yang didalamnya merupakan gabungan dari rangkaian-rangkaian elektronik yang dikemas dalam bentuk suatu *chip* (IC).

Pada umumnya mikrokontroler terdiri dari bagian-bagian sebagai berikut: Alamat (*address*), Data, Pengendali, Memori (*RAM* atau *ROM*), dan bagian *input-Output*.

Arsitektural AT mega 8535 adalah seperti gambar blok diagram diatas.

2.2.1. Konfigurasi Pin ATmega 8535

AVR - MICRO memiliki kemampuan untuk melakukan komunikasi data serial secara UART (*Universal asynchronous receiver/transmitter*) RS-232 serta pemrograman memori melalui ISP (*In-System Programming*). UART adalah jenis "*asynchronous receiver/transmitter*", bagian dari sebuah komputer hardware yang menterjemahkan data antara bentuk paralel dan serial. UARTs biasanya

digunakan dalam hubungannya dengan komunikasi standar lainnya seperti RS-232. UART biasanya seorang individu (atau bagian dari) sirkuit terpadu digunakan untuk komunikasi serial melalui perangkat komputer atau perangkat port serial. UART sekarang umumnya termasuk dalam microcontrollers. Banyak IC modern sekarang datang dengan UART yang juga dapat berkomunikasi serentak.

Beberapa penjelasan mengenai AVR ATmega 8535 adalah :

1. Fitur Mikrokontroler AVR ATmega 8535 Kapabilitas detail mikrokontroler AVR ATmega 8535 adalah sebagai berikut:

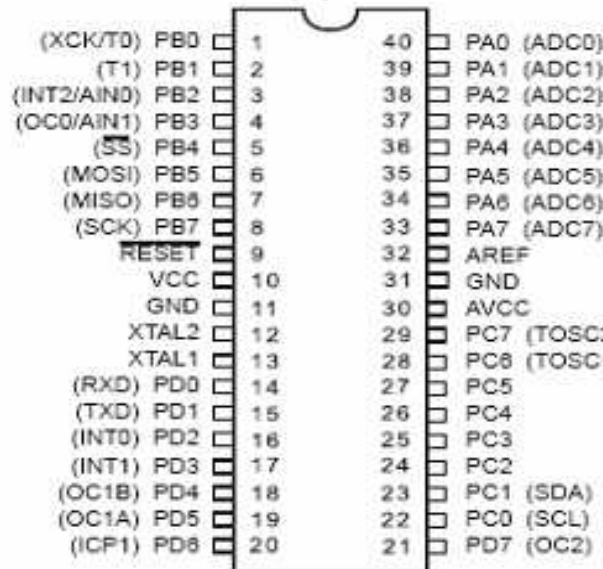
- a) Sistem mikroprosesor 8 bit dengan kecepatan maksimal 16 MHz.
- b) Kapabilitas memori flash 8 KB, SRAM sebesar 512 byte dan EEPROM (Electrically Erasable Programmable Read Only Memory) sebesar 512 byte.
- c) ADC (Pengubah analog-ke-digital) internal dengan ketelitian 10 bit sebanyak 8 saluran.
- d) PWM (*Pulse Wide Modulation*) sebanyak 4 saluran.
- e) Portal komunikasi serial (USART) dengan kecepatan maksimal 2,5 Mbps.
- f) Enam pilihan mode sleep, untuk menghemat penggunaan daya listrik.

2. Konfigurasi Pin Mikrokontroler AVR ATmega 8535

- a) VCC merupakan pin yang berfungsi sebagai pin masukan catu daya.
- b) GND merupakan pin ground.
- c) Port B (PB0 – PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu sebagai Timer/Counter, komperator analog dan SPI.
- d) Port C (PC0 – PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, komperator analog, input ADC dan Timer Osilator.
- e) Port D (PD0 – PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komperator analog, interupsi eksternal dan komunikasi serial.
- f) RESET merupakan pin yang digunakan untuk mereset mikrokontoler.
- g) XTAL1 dan XTAL2 merupakan pin masukan clock eksternal.
- h) AVCC merupakan pin masukan tegangan untuk ADC.

Konfigurasi pin ATmega 8535 ditunjukkan dalam Gambar 2.2.

AREF merupakan pin tegangan referensi ADC.



Gambar 2.2 Konfigurasi Pin ATmega 8535

2.2.2 Organisasi Memori

AVR ATmega 8535 memiliki ruang pengalamatan memori data dan memori program yang terpisah. Memori data terbagi menjadi 3 bagian, yaitu 32 buah register umum, 64 buah register I/O, dan 512 byte SRAM internal.

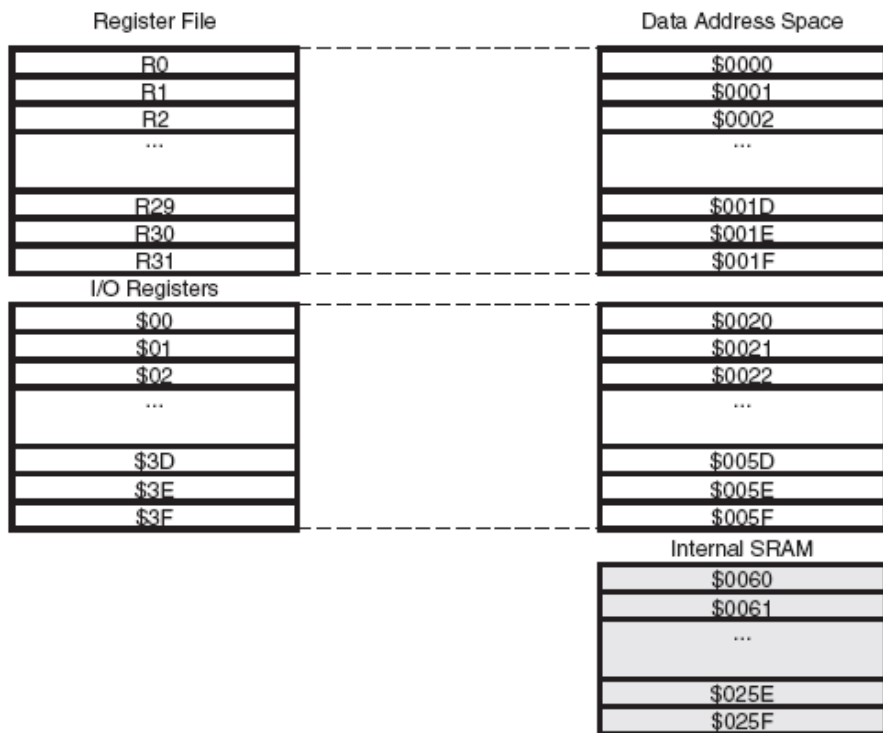
Register keperluan umum menempati *space data* pada alamat terbawah, yaitu \$00 sampai \$1F. Sementara itu register khusus untuk menangani I/O dan kontrol terhadap mikrokontroler menempati 64 alamat berikutnya, yaitu mulai dari \$20 hingga \$5F. Register tersebut merupakan register yang khusus digunakan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

Memori program yang terletak dalam flash PEROM tersusun dalam word atau 2 byte karena setiap intruksi memiliki lebar 16-bit atau 32-bit. AVR ATmega 8 memiliki 4Kbyte x 16-bit flash PEROM dengan alamat mulai \$000 sampai \$FFF.

AVR tersebut memiliki 12-bit Program Counter (PC) sehingga mampu mengalami isi flash. Konfigurasi memori data ditunjukkan dalam Gambar

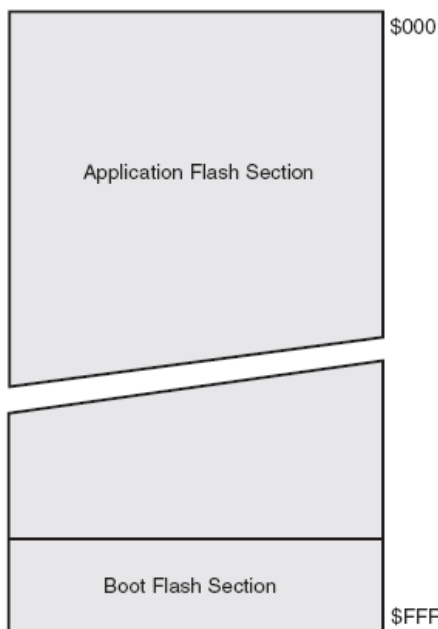
2.3. Dan memori program AVR ATmega 8535 ditunjukkan dalam Gambar 2.4.





Gambar 2.3 Konfigurasi Memori Data AVR ATmega 8535

Sumber : Data Sheet ATmega 8535, 2000 : 17



Gambar 2.4 Memori Program AVR ATmega 8535

Sumber : Data Sheet ATmega8535, 2000 : 16



Selain itu, AVR ATmega8535 juga memiliki memori data berupa EEPROM 8-bit sebanyak 512 byte. Alamat EEPROM dimulai dari \$000 sampai \$1FF.

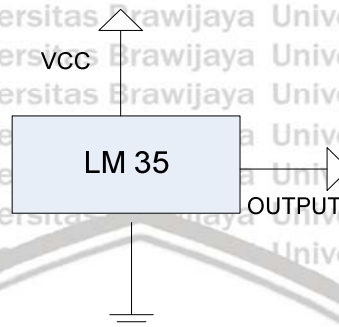
2.3. Sensor Suhu LM 35

Sensor suhu merupakan suatu pengindera suhu yang mampu memberikan Petunjuk jika ada perubahan temperatur. Sensor suhu dapat merespon perubahan Temperatur pada permukaan benda maupun perubahan temperatur pada udara. Perubahan temperatur yang dapat ditindak lanjuti adalah perubahan temperature yang cukup jelas. Maksud perubahan temperattur yang nilainya cukup tinggi, misalnya perubahan temperatur dari 25°C ke 30°C . Jika perubahan temperatur berkisar antara 25°C sampai 26°C maka temperatur hanya dapat didekteksi namun tidak dapat ditindak lanjuti untuk mengontrol sesuatu. Hal ini disebabkan nilai perubahan yang kecil itu akan menyulitkan bila diubah kebesaran yang lain, misalnya kebesaran tegangan.

Sensor suhu digunakan untuk mengubah besaran suhu menjadi sinyal listrik. Sensor suhu harus mempunyai kepekaan terhadap perubahan suhu yang kan diukur. Sinyal listrik ini kemudian diubah oleh converter analog ke digital (ADC) sehingga dapat diolah oleh mikrokontroller. Sensor suhu merupakan bagian dari element ukur dalam sistem control dengan umpan balik karena sifat dinamik dan static dari element ukur

Mempengaruhi pertunjukan tiarga sebenarnya dari variable keluaran, maka element ukur memegang peranan yang penting dalam menentukan perfomasi keseluruhan sistem. Sensor suhu yang digunakan adalah IC LM 35 produksi nasional semikonduktor. IC tersebut mempunyai ketelitian dan ketepatan yang tinggi. Keluaran JC ini mempunyai kelinieran yang tinggi dalam jangkauan yang memadai untuk kemampuan pengontrolan umum. Jangkauan atau kemampuan dari IC ini -55°C sampai $+150^{\circ}\text{C}$. LM 35 mempunyai impedansi keluaran rendah, keluaran linier dan memiliki kalibrasi yang tepat sehingga membuat pembacaan nilai keluaran menjadi mudah. Dengan sentivitas $10\text{mV}/^{\circ}\text{C}$, keluaran mengalami perubahan 10 mV setiap kenaikan 1°C . Beberapa kelebihan yang dimiliki IC ini membuat mudah untuk antarmuka, pembacaan dan pengontrolan. Dapat digunakan untuk catudaya tunggal, juga dengan catu daya

simetris plus dan minus. Dengan arus yang rendah yaitu 60 μA dan mempunyai pemanasan sendiri yang rendah yaitu kurang dari 0,1 C. Diagram blok sensor suhu LM 35 ditunjukkan dalam Gambar 2.5.



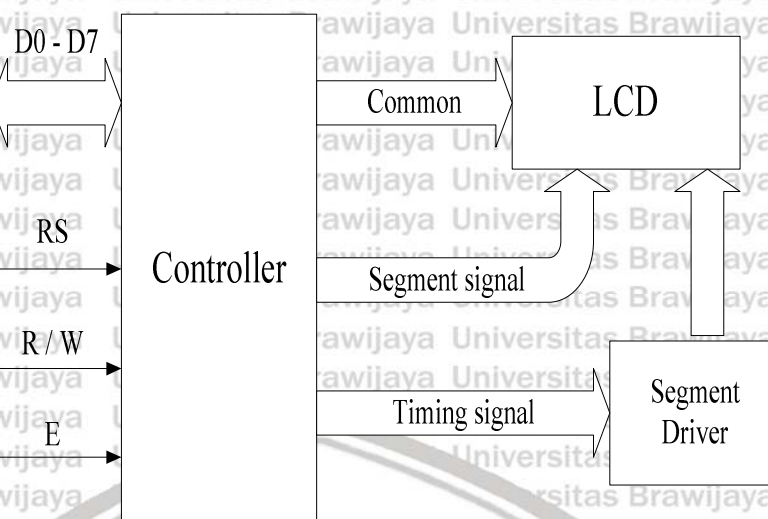
Gambar 2.5 Sensor Suhu LM 35

(Sumber: National DataBook, 1999)

2.4. Liquid Crystal Display (LCD)

Liquid Crystal Display adalah modul tampilan berkonsumsi daya yang relatif rendah dan terdapat sebuah controller CMOS di dalamnya. Controller tersebut sebagai pembangkit karakter dari ROM / RAM dan display data RAM. Semua fungsi tampilan dikontrol oleh suatu intruksi dan modul LCD dapat dengan mudah untuk di-antarmuka-kan dengan mikroprosesor / mikrokontroler. Input yang diperlukan untuk mengendalikan modul ini berupa bus data yang termultipleks dengan bus alamat dan 3 bit sinyal kontrol. Pengendali dot matrik LCD dilakukan secara internal oleh controller internal pada modul LCD sendiri.

Diagram blok LCD diperlihatkan dalam Gambar 2.6.



Gambar 2.6 Diagram Blok LCD

(Sumber: Seiko instruments inc., 1987: 2)

LCD tipe M1632 memiliki cirri-ciri sebagai berikut :

- a) LCD ini terdiri atas 32 karakter dengan 2 baris masing-masing 16 karakter dengan display dot matrik 5 x 7
- b) Karakter generator ROM dengan 192 tipe karakter
- c) Karakter generator RAM dengan 8 tipe karakter
- d) display data RAM 80 x 8 bit
- e) Dapat diinterfacekan ke MPU 8 atau 4
- f) Dilengkapi fungsi tambahan : display clear, cursor home, display ON/OFF, cursor ON/OFF, display character blink, cursor shift, dan display shit
- g) Internal data
- h) Internal otomatis, reset pada saat power ON
- i) PSU tunggal +5 volt.

Tabel 2.1 menunjukkan konfigurasi pin-pin LCD.

Tabel 2.1 Konfigurasi Pin-pin LCD

NO	SYMBOL	LEVEL	FUNCTION
1	Vss	-	0 Grond
2	Vcc	-	5 V + 10%
3	Vee	-	LCD Drive
4	Rs	H/L	H : Data Input L : Instruksi Input
5	R/W	H/L	H : read L : write
6	E	H/L	Enable signal
7-14	DB0-DB7	H/L	Data Bus
15	Light LCD	-	Menyalakan lampu LCD max 200 Ma
16	Light LCD	-	Ground

(Sumber: Seiko instruments inc., 1987: 2)

Instruksi operasi pada modul LCD merupakan kombinasi bilangan biner yang melalui pin saluran data 8 bit (DB0 - DB7), pin seleksi register (RS) dan pin seleksi instruksi (R/W). Instruksi operasi pada modul LCD ditunjukkan dalam

Tabel 2.2.

Tabel 2.2 Instruksi pada Modul LCD

No	INSTRUKSI	RS	RW	D7	D6	D5	D4	D3	D2	D1	D0
1	Display Clear	0	0	0	0	0	0	0	0	0	1
2	Cursor Home	0	0	0	0	0	0	0	0	1	S
3	Entry Mode Set	0	0	0	0	0	0	0	1	I/D	S
4	Display ON/OFF	0	0	0	0	0	0	1	D	C	B
5	Cursor Display Shift	0	0	0	0	0	1	S/C	R/L	*	*
6	Function Set	0	0	0	0	1	D/L	1	*	*	*
7	CGRAM Address Set	0	0	0	1	Acc					
8	DDRAM Address Set	0	0	1	Add						
9	BF/ Address Read	0	1	BF	AC						
10	Data Write To CGRAM	0	0	Write Data							
11	Data Read From CGRAM	1	1	Read Data							

Sumber: www.delta-electronic.com

Fungsi masing-masing instruksi adalah sebagai berikut:

1) *Display Clear*: membersihkan tampilan yang ada pada modul LCD dan mengembalikan kursor kembali ke posisi semula.

2) *Cursor home*: hanya membersihkan semua tampilan dan kursor kembali semula.

3) *Entry mode set*: layar beraksi sebagai tampilan tulis.

$S = 1/0$: menggeser layar

$I/O = 1$: kursor bergerak ke kanan dan layar bergerak ke kiri.

$I/O = 0$: kursor bergerak ke kiri dan layar bergerak ke kanan.

4) *Display on/off control*

$D = 1$: layar on.

$C = 1$: kursor on.

$B = 1$: kursor berkedip-kedip.

$D = 0$: layar off.

$C = 0$: kursor off.

$B = 0$: kursor tidak berkedip-kedip.

5) *Cursor display shift*

$S/C = 1$: modul LCD diidentifikasi sebagai layar.

$S/C = 0$: modul LCD diidentifikasi sebagai kursor.

$R/L = 1$: menggeser satu spasi ke kanan.

$R/L = 0$: menggeser satu spasi ke kiri.

6) *Function set*

$DL = 1$: panjang data modul LCD pada 8 bit (DB7 - DB0).

$DL = 0$: panjang data modul LCD pada 4 bit (DB7 - DB0).

Bit *upper* ditransfer terlebih dahulu kemudian diikuti dengan 4 bit *lower*.

$N = 1/0$: modul LCD menggunakan 2 atau 1 baris karakter.

$P = 1/0$: modul LCD menggunakan 5×10 atau 5×7 dot matrik.

7) *CG RAM address set* : menulis alamat RAM ke karakter.

8) *DD RAM address set* : menulis alamat RAM ke tampilan.

9) *BF/address set* : $BF = 1/0$, modul LCD dalam keadaan sibuk atau tidak sibuk.

10) *Data write to CG RAM or DD RAM*: menulis byte ke alamat terakhir RAM yang dipilih.

11) *Data read from CG RAM or DD RAM*: membaca byte dari alamat terakhir RAM yang dipilih.

Display Data RAM (DD RAM) pada modul LCD masing-masing *line* mempunyai *range* alamat tersendiri. Alamat itu diekspresikan dengan bilangan heksadesimal.

Untuk *line* 1 alamat berkisar antara 00H - 0FH sedangkan untuk *line* 2 berkisar antara 40H - 4FH.

2.4.1 Instruksi Operasi Dasar

1. Register

Kontroller dari LCD mempunyai dua buah register 8 bit yaitu register instruksi (IR) dan register data (DR). IR menyimpan instruksi seperti display clear, cursor shift dan display data (DD RAM) serta character generator (CG RAM). DR menyimpan data untuk ditulis di DD RAM atau CG RAM atau membaca data dari DD RAM atau CG RAM. Ketika data ditulis ke DD RAM atau CG RAM maka DR secara otomatis menulis data ke DD RAM atau CG RAM dan data pada DD RAM atau CG RAM hendak dibaca maka alamat data ditulis pada IR sedangkan data alamat dimasukkan melalui DR dan mikroprosesor membaca data dari DR. Register seleksi ditunjukkan dalam Tabel 2.3.

Tabel 2.3 Register Seleksi

Rs	R/W	OPERASI
0	0	Seleksi IR, IR Write Display Clear
0	1	Busy Flag (DB7) @counter (DB0-DB7)Reas
1	0	Seleksi DR, DR Write
1	1	Seleksi DR,DR Read

2. Busy flag

Busy flag menunjukkan bahwa modul siap untuk menerima instruksi selanjutnya. Sebagaimana yang terlihat pada tabel register seleksi sinyal akan melalui DB7. Jika RS = 0 dan R/W = 1. Jika bernilai 1 maka modul sedang melakukan kerja internal dan instruksi tidak dapat diterima. Sehingga status dari flag ini harus diperiksa sebelum melaksanakan instruksi selanjutnya.

3. Address counter

AC menunjukkan lokasi memori dalam modul LCD. Pemilihan lokasi alamat itu diberikan lewat register instruksi (IR). Ketika data ada pada A, maka

AC secara otomatis menaikkan atau menurunkan alamat tergantung dari Entry Mode Set.

4. Display Data RAM (DD RAM)

Pada LCD masing-masing line mempunyai range alamat tersendiri.

Alamat ini diekspresikan dengan bilangan Hexadesimal. Untuk itu 1 range alamat berkisar 00H-0FH sedangkan untuk line 2 range alamat berkisar antara 40H-4FH.

5. Character Generator ROM (CG ROM)

CG ROM mempunyai tipe dot matrik 5 x 7 dan pada LCD telah tersedia ROM sebagai pembangkit character dalam kode ASCII.

6. Character Generator RAM (CG RAM)

CG RAM digunakan untuk pembuatan karakter tersendiri melalui program. Untuk fungsi terminal pada LCD ditunjukkan dalam Tabel 2.4.

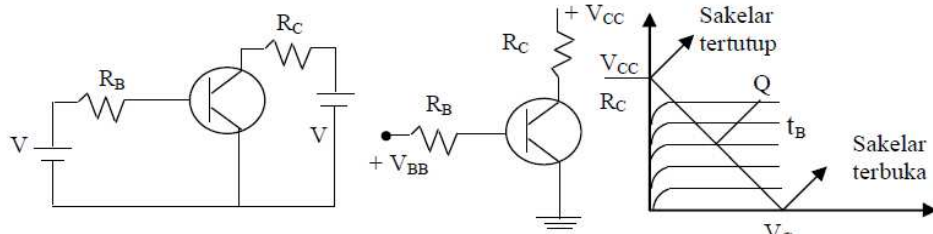
Tabel 2.4 Fungsi Terminal pada LCD

Nama signal	Jml Term	I/O	Tujuan	Fungsi
DB0-DB3	4	I/O	MPU	Sebagai lalulintas data dan instruksi ke atau dari MPU Low Byte
DB4-DB7	4	I/O	MPU	Sebagai lalulintas data atau instruksi 2 arah upper byte. DB7 sebagai busy flag
E	1	I	MPU	Sinyal start (read/write)
R/W	1	I	MPU	Seleksi sinyal 0 = write 1 = read
Rs	1	I	MPU	Seleksi register
VLC	1	-	PS	0 = instruksi reg (wr) Busy flag addr counter (rd) 1 = data reg (wr dan rd)
7	1	-	PS	Mengatur tampilan LCD
Vss	1	-	PS	+5volt



2.5 Transistor Sebagai Saklar

Selain bekerja sebagai penguat, transistor juga dapat bekerja sebagai saklar, transistor memiliki tiga daerah yang dapat dilihat dalam Gambar 2.7.



Gambar 2.7 Rangkaian Transistor dan Garis Beban.

(Sumber: Wasito, 1995, h:12)

Jika sebuah transistor digunakan sebagai saklar, maka transistor tersebut hanya dioperasikan pada salah satu dari dua kondisi yaitu kondisi saturasi (jenuh) dimana transistor seperti saklar tertutup atau kondisi cut off (tersumbat) dimana transistor sebagai yang terbuka. Sedangkan jika transistor bekerja pada on atau off, maka transistor akan bekerja sebagai penguat yaitu jika V_{be} transistor lebih besar 0,5 volt dan lebih kecil dari 0,8 volt. (Wasito, 1995, h:12). Ketika transistor berada dalam kondisi saturasi, maka:

1. Arus pada kolektor maksimum, $I_c = I_c(\text{sat})$.
2. Tegangan pada terminal kolektor emitter, $V_{ce} = 0$ volt
3. Tegangan pada beban yang dihubungkan seri dengan terminal kolektor = V_{ce} .

Sedangkan transistor dalam keadaan cut off, maka:

- 1) Tidak ada arus yang mengalir dikolektor $I_c = 0$ volt.
- 2) Tegangan pada terminal kolektor emitter dengan V_{ce} , yaitu $V_{ce} = V_{ce}$.
- 3) Tegangan pada beban dihubungkan seri pada kaki kolektor adalah nol.

Dalam merancang rangkaian transistor sebagai saklar maka agar saklar dapat menutup, harga $I_b > I_b(\text{sat})$ untuk menjamin dapat mencapai saturasi penuh.

$$I_{C(\text{sat})} \cong \frac{V_{CC}}{R_C} \quad (2-6)$$

(Wasito, 1995, h:12).

Dan arus basis yang tepat menimbulkan penjenjuran adalah sebagai berikut (Malvino, 1996:45):

$$(2-7)$$

$$I_{B(Sat)} = \frac{I_{C(sat)}}{\beta_{dc}}$$

Tegangan kolektor –emiter pada penjenjutan adalah (Malvino, 1996:45):

$$V_{CE} = V_{CE(sat)} \quad (2-8)$$

nilai resistor pada masing-masing kaki basis dapat ditentukan dengan persamaan berikut ini (Malvino, 1996:123):

$$R_B = \frac{V_{CC} - V_{BE}}{I_B} \quad (2-9)$$

2.6. Push Button (Tombol tekan)

Pada dasarnya prinsip push button adalah sama dengan saklar. Biasanya digunakan untuk pengendalian atau pengaturan peralatan, untuk memberikan tanda-tanda hubung, putus dan sebagainya. Push button mempunyai konstruksi sederhana terdiri atas dua bagian utama, yaitu bagian mekanis yang berupa tombol dan bagian listrik yang berupa bagian kontak.

Ada dua macam kondisi pada tombol tekan yaitu kontak NO (*Normally Open*) dan kontak NC (*Normally Close*). Kondisi kontak NO dalam keadaan normalnya terbuka sedangkan untuk kontak NC dalam keadaan normalnya terhubung. Dalam pengoperasiannya push button dapat dibedakan menjadi 2 macam, yaitu tombol tekan sesaat dan tombol tekan pengunci. Tombol tekan sesaat bekerja bila tombolnya ditekan, dan akan kembali bila tombolnya dilepas. Sedangkan untuk tombol tekan pengunci bila tombolnya ditekan akan bekerja dan selama itu akan dipertahankan, walaupun tombolnya dilepaskan. Hal ini disebabkan konstruksi jenis ini dilengkapi dengan kunci mekanis, untuk kembali ke posisi semula harus ditekan sekali lagi.

2.7. Solenoid

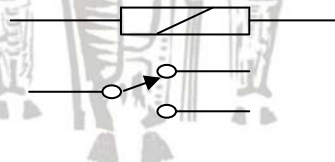
Solenoid merupakan suatu alat elektromekanik yang dapat merubah energi listrik menjadi energi gerak yang digunakan untuk menarik benda pada jarak tertentu. Arus yang mengalir pada lilitan kawat akan membuat gaya magnet. Gaya magnet tersebut dibutuhkan untuk menggerakkan besi penarik yang bergerak

secara berubah-ubah (menekan dan menarik). Dalam pengoperasiannya solenoid mempunyai karakteristik jika diberi catu tegangan pada lilitan kumparan, maka akan timbul garis gaya magnet. Dengan sifat yang dimiliki solenoid ini maka dia dapat dimanfaatkan untuk menggerakkan suatu benda. (www.solenoidcity.com).

2.8. Relay

Relay adalah suatu perangkat switch yang dioperasikan oleh gaya elektromagnetik. Prinsip kerja relay seperti prinsip kerja saklar. Relay yang umum digunakan saat ini adalah jenis elektromagnetik yang terdiri atas kumparan magnetik yang jika mendapat bias arus akan dapat mengendalikan kontak penghubung. Apabila input relay diberi bias maka arus pada kumparan akan terdapat induksi magnetik yang nantinya akan menarik tegak kontak untuk merubah posisi awalnya menjadi terbuka. Pada bagian yang kita inginkan, jika arus berhenti maka tidak ada induksi sehingga kontak akan kembali ke posisi semula/posisi awal

Kontak yang ada pada relay ada dua macam, yaitu *Normally Open (NO)* dan *Normally Close (NC)*. *Normally Open (NO)* adalah relay yang kontakannya terbuka saat tidak bekerja. Sedangkan *Normally Close (NC)* adalah relay yang kontakannya tertutup saat tidak bekerja. Simbol relay ditunjukkan dalam Gambar 2.8.



Gambar 2.8 Simbol Relay

BAB III

METODE PENELITIAN

Metodelogi adalah struktur perancangan, dimana fungsi dari metode ini adalah untuk mempermudah perancangan dengan cara pengerjaan yang terseruktur. Metode yang digunakan dalam perancangan dan pembuatan alat ini. Untuk dapat menghasilkan alat yang dapat bekerja secara optimal maka harus memahami karakteristik komponen utama dan konteks kerjanya dalam suatu rangkaian.

Dalam Perancangan dan pembuatan otomatisasi kran dan pemanas air pada *dispenser* berbasis mikrokontroler ini menghasilkan suatu perangkat yang dapat bekerja otomatis dalam pengambilan air. Oleh karena itu direncanakan terlebih dahulu perangkat-perangkat yang mendukung system kerja alat ini. Pengujian ini terdiri dari pengujian perangkat keras (*hardware*) dan pengujian secara keseluruhan (*hardware* dan *software*). Untuk pengujian perangkat keras (*hardware*) meliputi pengujian terhadap pengujian selenoid, pengujian, *relay*, dan pengujian driver *relay*. sedangkan *software* berupa program yang menggunakan pemrograman bahasa C.

3.1 Perancangan Alat

Alat yang dirancang memiliki spesifikasi sebagai berikut :

- 1) Mikrokontroler yang digunakan adalah AVR ATmega8535. Sebagai pemroses input/masukan dari tombol sehingga akan menghasilkan output/keluaran yang akan mengatur kinerja selenoid.
- 2) Suhu air panas yang diinginkan mulai 30⁰c sampai 100⁰c
- 3) Sensor suhu sebagai inputan dari mikrokontroler untuk mengetahui suhu air dalam *dispenser*.
- 4) Tabung *heater* berisi 1 liter air
- 5) Driver Selenoid, berfungsi untuk menggerakkan atau mendorong selenoid.

6) Volume air yang keluar dari *dispenser* antara 100 ml, 150 ml,.....,500ml

7) LCD, digunakan untuk menampilkan suhu dan volume yang diinginkan.

3.2. Perancangan Perangkat Lunak

Perangkat lunak (*software*) digunakan untuk mendukung kinerja dan dapat mengendalikan proses yang akan dikerjakan perangkat keras (*hardware*) agar sesuai dengan yang direncanakan. Untuk memprogram mikrokontroler menggunakan bahasa C. Perangkat lunak ini memiliki simulator yang dapat digunakan untuk melihat jalannya program, sehingga penelusuran kesalahan dan tahapan program dapat langsung diketahui.

3.3. Pengujian Alat

Tujuan pengujian alat adalah untuk memastikan bahwa sistem yang dibuat telah berjalan, yaitu dengan menguji rangkaian perblok dan keseluruhan rangkaian. Metode pengujian alat adalah sebagai berikut:

- 1) Membuat miniatur dari sistem.
- 2) Pengujian Mikrokontroler AVR ATmega8535 dengan LCD. Mikrokontroler AVR ATmega8535 dan ditampilkan LCD sesuai dengan tujuan dan bekerja dengan baik.
- 3) Pengujian Mikrokontroler AVR ATmega 8 dengan Driver selenoid. Bertujuan untuk mengetahui apakah fungsi dari driver selenoid bekerja dengan baik untuk menggerakkan selenoid agar bekerja dengan baik dan sesuai yang diinginkan.
- 4) Pengujian Sensor suhu LM 35. Bertujuan untuk mengetahui apakah sensor ini dapat mendeteksi perubahan suhu air dan memberikan keluaran yang diinginkan.
- 5) Pengujian volume air yang keluar pada kran solenoid. Bertujuan untuk mengetahui jumlah volume air yang keluar.
- 6) Pengujian perangkat lunak.



Bertujuan untuk mengetahui apakah program dapat berjalan sesuai dengan perencanaan. Metode pengujian dengan menggunakan software bahasa C.

- 7) Pengujian keseluruhan sistem. Bertujuan untuk mengetahui apakah sistem dapat bekerja sesuai dengan perencanaan. Metode yang digunakan adalah dengan mengaplikasikan alat secara keseluruhan dengan *heater dispenser* dan sensor yang baik.



BAB IV

PERANCANGAN DAN PEMBUATAN ALAT

Untuk dapat menghasilkan alat yang dapat bekerja secara optimal maka harus memahami karakteristik komponen utama dan konteks kerjanya dalam suatu rangkaian.

Dalam Perancangan dan pembuatan otomatisasi kran dan pemanas air pada *dispenser* berbasis mikrokontroler ini menghasilkan suatu perangkat yang dapat bekerja otomatis dalam pengambilan air. Oleh karena itu direncanakan terlebih dahulu perangkat-perangkat yang mendukung system kerja alat ini. Pengujian ini terdiri dari pengujian perangkat keras (*hardware*) dan pengujian secara keseluruhan (*hardware* dan *software*). Untuk pengujian perangkat keras (*hardware*) meliputi pengujian terhadap pengujian selenoid, pengujian, *relay*, dan pengujian driver *relay*. sedangkan *software* berupa program yang menggunakan pemrograman bahasa C.

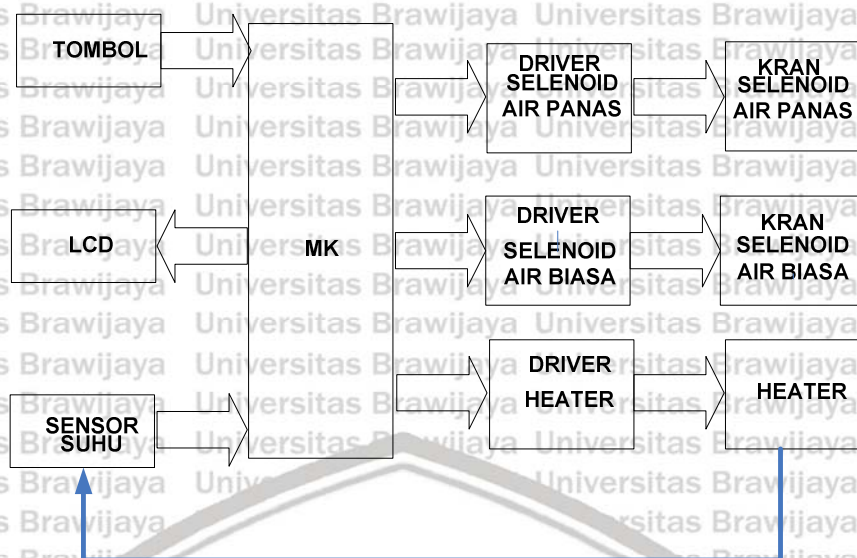
4.1. Penentuan Spesifikasi Alat

Alat yang dirancang memiliki spesifikasi sebagai berikut:

- 1) Suhu mulai dari 30⁰c sampai 100⁰c.
- 2) Resolusi perubahan suhu 10⁰c.
- 3) Volume yang diinginkan 100 ml sampai 500 ml.
- 4) Resolusi perubahan volume 50 ml.

4.2. Perancangan Perangkat Keras

Diagram blok sistem alat otomatisasi kran pemanas air *dispenser* berbasis mikrokontroler ditunjukkan dalam Gambar 4.1.



Gambar 4.1 Diagram Blok Otomatisasi Kran dan Pemanas Air pada *Dispenser*

Keterangan untuk masing-masing blok :

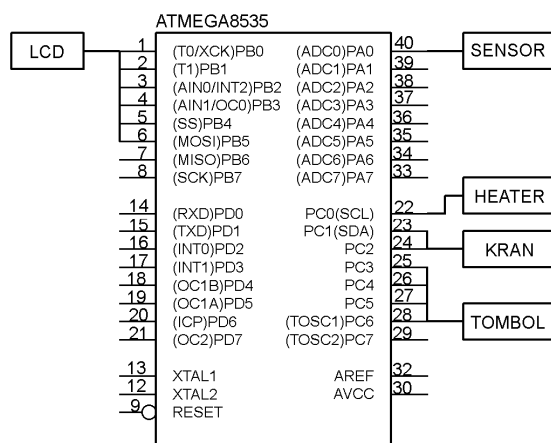
- 1) Tombol push button, yang berfungsi sebagai inputan yang digunakan untuk mengatur memasukkan volume air dan suhu.
- 2) LCD, digunakan untuk menampilkan data suhu dari dispenser serta setting suhu yang diinginkan melalui tombol.
- 3) Rangkaian sensor suhu, berfungsi untuk mengubah suhu dari sensor yang ada di dispenser menjadi tegangan.
- 4) MK, sebagai pemroses input/masukan dari keypad sehingga akan menghasilkan output/keluaran yang akan mengatur kinerja selenoid.
- 5) Driver Selenoid, berfungsi untuk menggerakkan atau mendorong selenoid. Untuk Driver Selenoid 1 untuk air panas begitu juga untuk yang ke 2 untuk air dingin.
- 6) Driver heater, berfungsi untuk mengaktifkan heater.
- 7) Selenoid, berfungsi untuk membuka atau menutup kran air dispenser secara otomatis. Untuk Selenoid 1 untuk air panas begitu juga untuk yang ke 2 untuk air dingin.

4.2.1 Rangkaian Mikrokontroler AVR ATmega 8535

Mikrokontroler dalam perancangan ini merupakan komponen utama selain PC (Personal Komputer) karena komponen inilah yang akan mengatur sistem agar dapat bekerja dengan baik dan optimal.

- 1) Port PB0 s/d PB5 digunakan untuk data LCD.
- 2) Port XTAL1 dan XTAL2 digunakan sebagai clock.
- 3) Port PA0 digunakan sebagai input dari LM 35.
- 4) Port PC0 digunakan sebagai output ke *heater*.
- 5) Port PC1 dan PC2 digunakan sebagai output ke selenoid.
- 6) Port PC3 s/d PC6 digunakan sebagai input tombol.

Gambar mikrokontroler AVR ATmega 8535 ditunjukkan dalam Gambar 4.2.

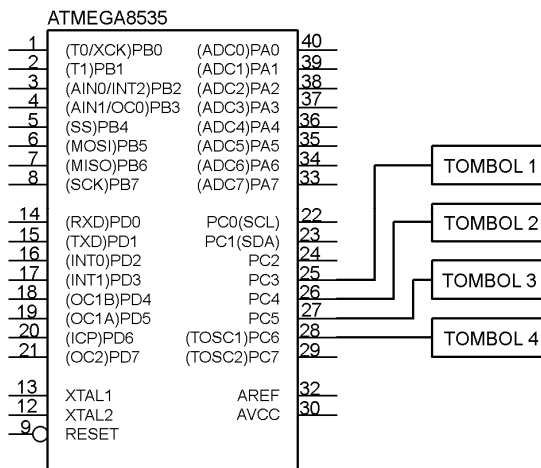


Gambar 4.2 Mikrokontroler sebagai Sistem Minimum.

4.2.2. Rangkaian Push Button (Tombol Tekan)

Rangkaian push button yang digunakan berjumlah 4 tombol. Sepuluh tombol pada baris pertama berfungsi untuk mengatur presentase suhu antara 30^oc sampai 100^oc untuk sepuluh tombol pada baris kedua berfungsi untuk mengatur volume air yang keluar 100ml,150ml,....., 500ml. Sedangkan dua tombol

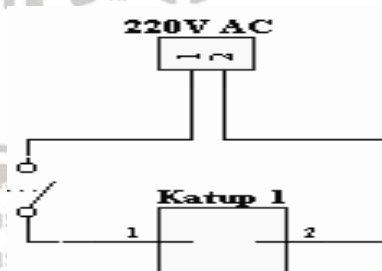
dibawahnya berfungsi sebagai enter dan reset. Rangkaian *push button* ditunjukkan dalam Gambar 4.3.



Gambar 4.3 Rangkaian *Push Button*

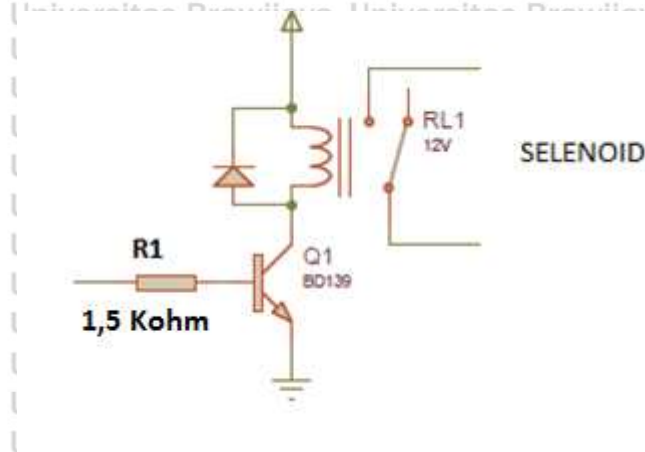
4.2.3. Rangkaian Selenoid

Pada sistem ini selenoid berfungsi sebagai *output* (keluaran) yang berasal dari mikrokontroler. Berikut adalah gambar rangkaian minimum selenoid yang dihubungkan pada port pc1 dan port pc2 pada Mikrokontroler AVR ATmega 8535. Gambar rangkain selenoid ditunjukkan dalam Gambar 4.4.



Gambar 4.4 Rangkaian Selenoid

Untuk menggerakkan selenoid dibutuhkan sebuah driver selenoid yang menggunakan relay sebagai pemicunya. Relay ini terhubung dengan Transistor BD 139 digunakan sebagai *switching/saklar*. Rangkain driver selenoid ditunjukkan dalam Gambar 4.5.



Gambar 4.5 Driver Selenoid

Mikrokontroler memberikan logika tinggi (tegangan sebesar 5V) ke kaki basis transistor BD139 yang menjadikan transistor bekerja karena adanya beda tegangan antara basis dan emitor, yang berdasar datasheet tegangan maksimum VBE adalah sebesar 5 VDC. Arus mengalir menuju emiter dari kolektor menjadikan kolektor menjadikan relay berubah keposisi NO (Normally Open), mengakibatkan driver bekerja. Sebaliknya, ketika akan diberhentikan, mikrokontroler memberikan logika rendah (tegangan sebesar 0V) ke kaki basis transistor BD 139, menjadikan tidak adanya beda tegangan antara basis dan emitor sehingga transistor tidak bekerja dan relay kembali keposisi NC (Normally Close). Berdasar datasheet, penguatan arus atau hfe transistor BD139 minimalnya sebesar 40, sehingga untuk dapat memenuhi penguatan 40, dapat dengan mengatur nilai R1: Berikut perhitungan R:

$$I_{\text{relay}} = V_{cc} / R_{\text{relay}}$$

$$= 12/100$$

$$= 0.12 \text{ A}$$

$$I_{\text{relay}} = 120 \text{ mA}$$

$$I_b = I_{\text{relay}} / h_{fe}$$

$$= 120 / 40$$

$$= 3 \text{ mA}$$

$$R_b = \frac{V_{cc} - V_{be}}{I_b}$$

$$= \frac{5V - 0.7V}{3 \text{ mA}}$$

$$= \frac{4.3}{3 \text{ mA}}$$

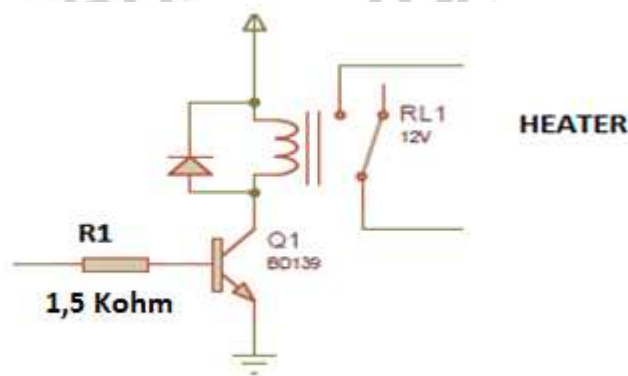
$$= 1,4 \text{ K}\Omega$$

Nilai R1 dapat diketahui nilainya yaitu 1,4 K Ω

4.2.4 Rangkaian Heater

Untuk menggerakkan *heater* dibutuhkan sebuah driver *heater*. Relay ini terhubung dengan Transistor BD 139 digunakan sebagai *switching/saklar*.

Sehingga bisa mengaktifkan *heater*. Driver *heater* ditunjukkan dalam Gambar 4.6.



Gambar 4.6 Driver *Heater*.

Mikrokontroler memberikan logika tinggi (tegangan sebesar 5V) ke kaki basis transistor BD139 yang menjadikan transistor bekerja karena adanya beda tegangan antara basis dan emitor, yang berdasar datasheet tegangan maksimum VBE adalah sebesar 5 VDC. Arus mengalir menuju emiter dari kolektor menjadikan kolektor menjadikan relay berubah keposisi NO (Normally Open), mengakibatkan driver bekerja. Sebaliknya, ketika akan diberhentikan, mikrokontroler memberikan logika rendah (tegangan sebesar 0V) ke kaki basis transistor BD 139, menjadikan tidak adanya beda tegangan antara basis dan emitor sehingga transistor tidak bekerja dan relay kembali keposisi NC (Normally Close). Berdasar datasheet, penguatan arus atau hfe transistor BD139 minimalnya sebesar 40, sehingga untuk dapat memenuhi penguatan 40, dapat dengan mengatur nilai R1: Berikut perhitungan R:

$$I_{\text{relay}} = V_{\text{cc}} / R_{\text{relay}}$$

$$= 12 / 100$$

$$= 0.12 \text{ A}$$

$$I_{\text{relay}} = 120 \text{ mA}$$

$$I_b = I_{\text{relay}} / h_{\text{fe}}$$

$$= 120 / 40$$

$$= 3 \text{ mA}$$

$$R_b = \frac{V_{\text{cc}} - V_{\text{be}}}{I_b}$$

$$= \frac{5\text{V} - 0.7\text{V}}{3 \text{ mA}}$$

$$= \frac{4.3}{3 \text{ mA}}$$

$$= 1,4 \text{ K}\Omega$$

Nilai R1 dapat diketahui nilainya yaitu 1,4 K Ω

4.2.5. Display LCD (*Liquid Crystal Display*)

Liquid Crystal Display adalah modul tampilan berkonsumsi daya yang relatif rendah dan terdapat sebuah kontroller CMOS di dalamnya. Kontroller tersebut sebagai pembangkit karakter dari ROM/RAM dan display data RAM. Semua fungsi tampilan dikontrol oleh suatu intruksi dan modul LCD dapat dengan mudah untuk di-antarmuka-kan dengan mikroprocessor/mikrokontroler. Input yang diperlukan untuk mengendalikan modul ini berupa bus data yang termultipleks dengan bus alamat dan 3 bit sinyal kontrol. Pengendali dot matrik LCD dilakukan secara internal oleh kontroller internal pada modul LCD sendiri.

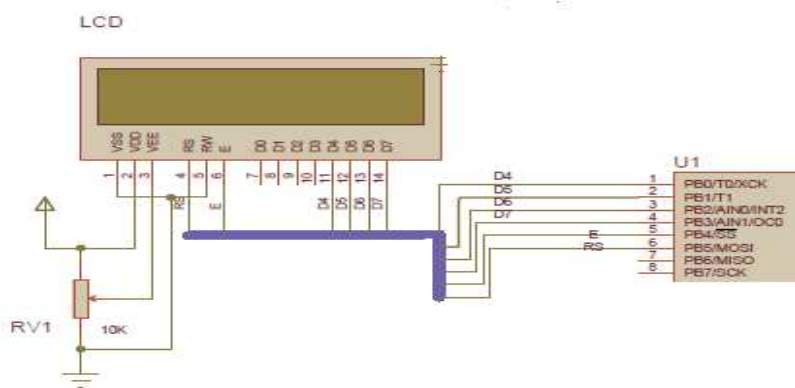
LCD merupakan suatu bentuk kristal cair yang akan beremulsi apabila dikenakan tegangan padanya. Tampilannya ini berupa dot matrik 5 x 7 LCD sehingga jenis huruf yang dapat ditampilkan akan lebih banyak dan lebih baik resolusinya jika dibandingkan dengan 7 segment. Konfigurasi pin-pin LCD ditunjukkan dalam Tabel 4.1.

Tabel 4.1 Konfigurasi Pin-pin LCD

NO	SYMBOL	LEVEL	FUNCTION
1	V _{ss}	-	0 Grond
2	V _{cc}	-	5 V + 10%
3	V _{ee}	-	LCD Drive
4	Rs	H/L	H : Data Input L : Instruksi Input
5	R/W	H/L	H : read L : write
6	E	H/L	Enable signal
7-14	DB0-DB7	H/L	Data Bus
15	Light LCD	-	Menyalakan lampu LCD max 200 Ma
16	Light LCD	-	Ground

(Sumber: Seiko instruments inc., 1987: 2)

Saluran data D4 s/d D7 dihubungkan pada port B.0 – port B.3 Mikrokontroler AVR ATmega 8535. Sedangkan penyemat RS dihubungkan pada port B.5 Mikrokontroler AVR ATmega 8535. Penyemat V_{ee} dihubungkan pada potensio meter 10kΩ, untuk mengatur kecerahan LCD. Rangkaian LCD ditunjukkan dalam Gambar 4.7.



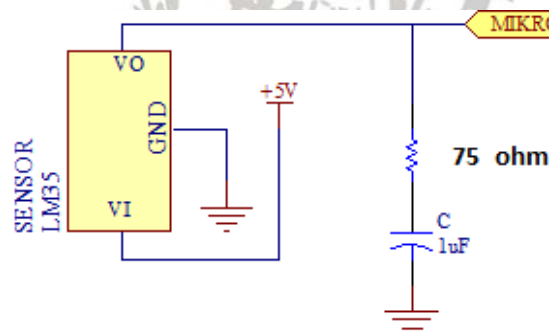
Gambar 4.7 Rangkaian LCD (*Liquid Crystal Display*)

4.2.6. Rangkaian Sensor Suhu LM35

Sensor suhu merupakan suatu pengindera suhu yang mampu memberikan Petunjuk jika ada perubahan temperatur. Sensor suhu dapat merespon perubahan Temperatur pada permukaan benda maupun perubahan temperatur pada udara.

Perubahan temperatur yang dapat ditindak lanjuti adalah perubahan temperature yang cukup jelas. Maksud perubahan temperattur yang nilainya cukup tinggi, misalnya perubahan temperatur dari 25°C ke 30°C. Jika perubahan temperatur berkisar antara 25°C sampai 26°C maka temperatur hanya dapat didekteksi namun tidak dapat ditindak lanjuti untuk mengontrol sesuatu. Hal ini disebabkan nilai perubahan yang kecil itu akan menyulitkan bila diubah kebesaran yang lain, misalnya kebesaran tegangan.

Kepekaan sensor terhadap suhu pada lm 35 adalah sebesar 0,01 v/°C dengan akurasi sebesar ± 1 °C. Dengan menghubungkan kaki GND ke tanah, maka batas bawah keluarannya adalah 0 V untuk 0°C. Sehingga keluarannya sebesar 1 V pada suhu 100 °C. Inilah yang ditentukan sebagai range pengukuran suhunya. Tapi karena suhu tertinggi yang dibutuhkan adalah ± 100 °C, maka suhu yang akan dibahas dibatasi sampai 100 °C. Suatu peredam R dan C seri ditambahkan untuk mengurangi gangguan dari luar. Rangkaian sensor suhu LM35 ditunjukkan dalam Gambar 4.8.



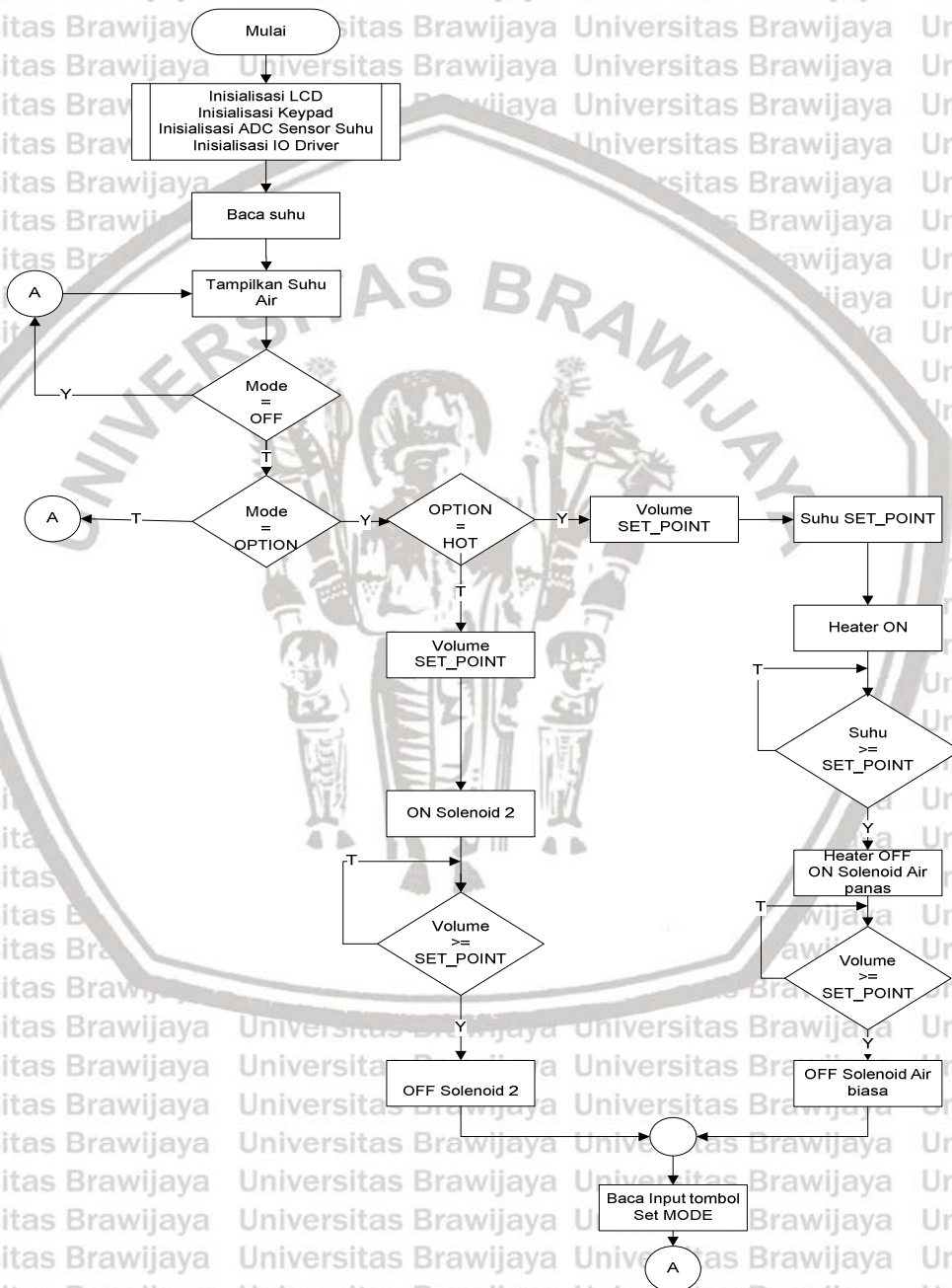
Gambar 4.8 Rangkaian Sensor Suhu LM35
(Data sheet LM35 National Semiconductor)

4.2.7. Perancangan Perangkat Lunak

Perangkat lunak (*software*) digunakan untuk mendukung kinerja dan dapat mengendalikan proses yang akan dikerjakan perangkat keras (*hardware*) agar sesuai dengan yang direncanakan. Untuk memprogram mikrokontroler menggunakan bahasa C. Perangkat lunak ini memiliki simulator yang dapat digunakan untuk melihat jalannya program, sehingga penelusuran kesalahan dan tahapan program dapat langsung diketahui.

4.2.8. Flowchart Program

Sebelum menyusun program bahasa C terlebih dahulu kita harus membuat flowchart (diagram alir) untuk mempermudah kita membuat program yang akan dibuat. Flowchart disini dibuat berdasarkan perencanaan perangkat keras yang dirancang. Flowchart program dari keseluruhan sistem ditunjukkan dalam Gambar 4.9.



Gambar 4.9 Flowchart Program dari Keseluruhan Sistem.

BAB V

PENGUJIAN DAN PEMBAHASAN

Bab ini akan membahas pengujian dan analisis alat yang telah dirancang dan direalisasikan. Pengujian dilakukan pada tiap-tiap blok dengan tujuan untuk mengamati apakah tiap blok rangkaian sudah sesuai dengan yang diharapkan baru kemudian dilanjutkan dengan pengujian keseluruhan sistem.

Pengujian yang dilakukan adalah:

- Pengujian Rangkaian Sensor Suhu.
- Pengujian Mikrokontroler dan LCD.
- Pengujian Rangkaian Driver Relay dan Selenoid.
- Pengujian volume air yang keluar.
- Pengujian Sistem secara Keseluruhan.

5.1. Pengujian Sensor Suhu

5.1.1 Tujuan Pengujian

Pengujian sensor suhu ini bertujuan untuk mengetahui apakah sensor bekerja sesuai dengan yang diharapkan.

5.1.2. Peralatan Pengujian

Dalam pengujian antarmuka keypad diperlukan peralatan-peralatan pengujian. Peralatan pengujian yang diperlukan sebagai berikut:

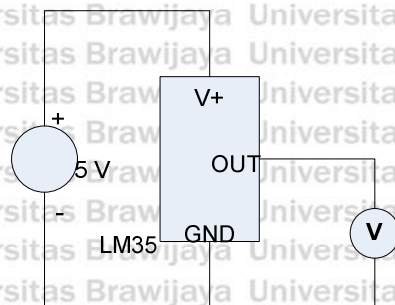
- Catu daya 5 volt.
- Multimeter.
- Termometer.

5.1.3. Prosedur Pengujian

Prosedur pengujian yang dilakukan adalah sebagai berikut:

- Membuat rangkaian pengujian sensor suhu seperti ditunjukkan dalam gambar 5.1.
- Sumber tegangan DC diaktifkan.
- Mencatat nilai tegangan yang tertera pada multimeter.

Rangkaian pengujian sensor suhu ditunjukkan dalam Gambar 5.1.



Gambar 5.1 Rangkaian Pengujian Sensor Suhu

5.1.4. Hasil Pengujian

Hasil pengujian sample suhu ditunjukkan dalam Tabel 5.2.

Tabel 5.1 Hasil Pengujian Sample Suhu

NO	Suhu ($^{\circ}$ C)	Hasil perhitungan V_o sensor (V)	Hasil pengujian V_o sensor (V)	Kesalahan(%)
1.	30	0,30	0,32	6,67
2.	40	0,40	0,42	5,00
3.	50	0,50	0,49	2,00
4.	60	0,60	0,59	1,67
5.	70	0,70	0,71	1,43
6.	80	0,80	0,81	1,23
7.	90	0,90	0,91	1,09
8.	100	1	0,99	1,01

Pada pengujian rangkaian sensor suhu berikut kita dapat mengetahui apakah sensor suhu ini dapat bekerja dengan baik atau tidak. Bila dilihat dari prosentase *error* persen pada setiap suhu yang diuji maka dapat dilihat *error* persen angkanya tidak lebih dari sepuluh persen. Rata-rata kesalahan 2,5%. Dengan ini terbukti bahwa rangkaian ini dapat bekerja sesuai yang diinginkan. Untuk lebih lengkapnya dapat di tunjukkan dalam Tabel 5.1.

5.2. Pengujian LCD

5.2.1. Tujuan Pengujian

Untuk mengetahui dan memastikan bahwa mikrokontroler dan LCD dapat bekerja dengan baik.

5.2.2. Peralatan Pengujian

Dalam pengujian antarmuka modul LCD diperlukan peralatan-peralatan pengujian. Peralatan pengujian yang diperlukan sebagai berikut:

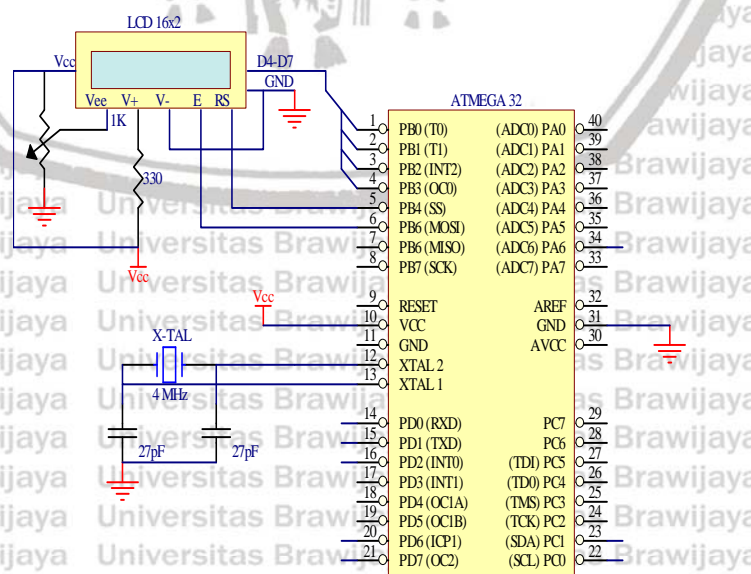
- Catu daya +5 volt.
- Minimum sistem mikrokontroler ATmega 8535.
- Rangkaian LCD.

5.2.3. Prosedur Pengujian

Prosedur pengujian yang dilakukan adalah sebagai berikut:

- Menyusun rangkaian pengujian sesuai dengan rangkaian pengujian dalam Gambar 5.2.
- Membuat program untuk menampilkan data karakter huruf dan karakter angka. Baris pertama menampilkan karakter huruf dan karakter angka pada baris kedua pada LCD.
- Memberikan catu daya pada rangkaian pengujian.
- Mengamati data karakter yang ditampilkan pada LCD.

Rangkaian pengujian LCD ditunjukkan dalam Gambar 5.2



Gambar 5.2 Rangkaian Pengujian LCD

5.2.4. Hasil Pengujian

Pengujian pada rangkaian ini hanya sebatas pengeluaran data dari mikrokontroler untuk ditulis ke LCD seperti ditunjukkan dalam Tabel 5.2.

Tabel 5.2 Penulisan LCD

No	Karakter yang dikirim mikrokontroler	Karakter yang tampak LCD
1	Afrilian	Afrilian (upper)
2	085755220999	085755220999 (lower)

Data yang dikeluarkan oleh mikrokontroler pada Tabel 5.2 bisa ditampilkan dengan benar oleh LCD, baik yang upper line (baris 1 LCD) maupun yang lower line (baris 2 LCD), dari sini dapat dipastikan bahwa mikrokontroler dan LCD dapat bekerja dengan baik sesuai dengan yang diharapkan.

5.3. Pengujian Rangkaian Drive Relay

5.3.1. Tujuan Pengujian

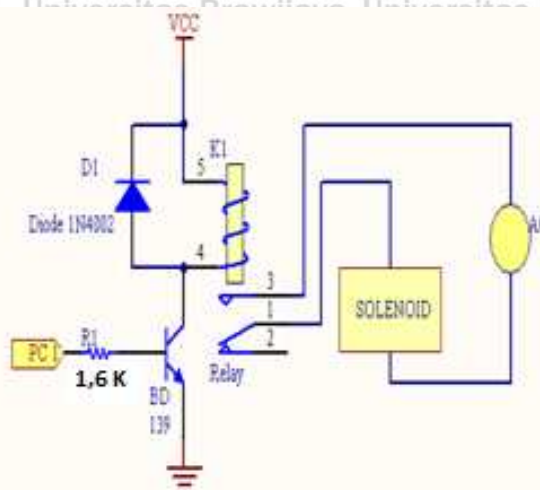
Pengujian ini bertujuan untuk mengetahui driver relay dapat bekerja secara maksimal.

5.3.2. Peralatan Pengujian

Dalam pengujian rangkaian driver relar diperlukan peralatan-peralatan pengujian. Peralatan pengujian yang diperlukan sebagai berikut:

- Catu daya +5 volt.
- Minimum sistem mikrokontroler ATmega 8535.
- Multimeter.

Rangkaian pengujian driver relay dditunjukkan dalam Gambar 5.3.



Gambar 5.3 Rangkaian Pengujian Driver Relay

5.3.3. Prosedur Pengujian

Langkah-langkah dalam pengujian rangkaian ini adalah:

Langkah-langkah pengujian *Driver Relay* adalah sebagai berikut:

- Rangkaian diberi *supply* 5 Volt.
- Diberi logika "0" dan "1" pada Portc.1.
- Tegangan pada logika "0" dan "1" diukur dengan voltmeter.
- Perubahan *relay* dicatat untuk logika "0" dan "1".

5.3.4. Hasil Pengujian

Dari hasil pengujian dapat dilihat bahwa rangkaian dapat bekerja dengan baik. Hasil pengujian driver relay ditunjukkan dalam Tabel 4.3.

Tabel 4.3 Hasil Pengujian Driver Relay

Mikrokontroler	Logika	Tegangan <i>Vbe</i>	Solenoid
Portc.1	0	0,02 Volt	OFF
Portc.1	1	4,01 Volt	ON

Dari hasil pengujian *driver relay* didapatkan bahwa *relay* tidak aktif pada saat logika port mikrokontroler yang digunakan untuk mengontrol *relay* berlogika "0" dengan tegangan 0,02 Volt dan *relay* aktif pada saat kondisi *portc.1* mikrokontroler berlogika "1" dengan tegangan 4,01 Volt.

Untuk logika "0" tegangannya kurang dari 2,4 Volt dan logika "1"

tegangannya lebih dari 2,4 Volt. Sehingga berdasarkan pengujian, maka *driver relay* sudah bekerja dengan baik.

5.4. Pengujian Volume Air

5.4.1. Tujuan Pengujian

Pengujian ini dilakukan untuk mengetahui volume air dapat bekerja sesuai yang kita inginkan.

5.4.2. Peralatan Pengujian

Dalam pengujian volume air diperlukan peralatan-peralatan pengujian.

Peralatan pengujian yang diperlukan sebagai berikut:

- Sumber tegangan AC.
- Gelas ukur.
- Selenoid.
- Timer.

5.4.3. Prosedur Pengujian

Langkah-langkah dalam pengujian volume air ini adalah:

- Menyiapkan gelas ukur.
- Memberi Sumber tegangan AC.
- Mengaktifkan selenoid dengan memberi sumber tegangan AC.
- Mengukur waktu keluaran air dengan timer.

5.4.4. Hasil Pengujian

Dari hasil pengujian dapat dilihat bahwa pengujian volume air dapat bekerja. Hasil pengujian volume air panas ditunjukkan dalam Tabel 5.4.

Tabel 5.4 Hasil Pengujian Volume Air Panas

N0	Volume yang diinginkan (ml)	Pewaktuan (detik)	Hasil perhitungan (ml)	Hasil pengujian (ml)	Kesalahan (%)
1	0 s/d 100	4	100	90	10
2	0 s/d 150	6	150	160	10
3	0 s/d 200	8	200	195	2,5
4	0 s/d 250	10	250	235	6
5	0 s/d 300	11	300	310	3,3
6	0 s/d 350	13	350	335	4,2
7	0 s/d 400	15	400	395	1,25
8	0 s/d 450	17	450	460	2,2
9	0 s/d 500	19	500	510	2

Hasil pengujian volume air biasa ditunjukkan dalam Tabel 5.5.

Tabel 5.5 Hasil Pengujian Volume Air Biasa.

N0	Volume yang diinginkan (ml)	Pewaktuan (detik)	Hasil perhitungan (ml)	Hasil pengujian (ml)	Kesalahan (%)
1	0 s/d 100	3	100	90	10
2	0 s/d 150	4,5	150	160	10
3	0 s/d 200	6	200	195	2,5
4	0 s/d 250	7,5	250	235	6
5	0 s/d 300	9	300	310	3,3
6	0 s/d 350	10,5	350	335	4,2
7	0 s/d 400	12	400	395	1,25
8	0 s/d 450	13,5	450	460	2,2
9	0 s/d 500	15	500	510	2

Pada pengujian volume air yang keluar, dapat mengetahui apakah pengujian volume air yang dikeluarkan ini dapat bekerja dengan baik atau tidak. Dengan resolusi volume 50 ml. Perubahan keadaan saat galon air penuh, sedang, dan tersisa sedikit tidak terlalu jauh perubahannya 0,5 detik. Bila dilihat dari prosentase *error* persen pada keluaran volume air yang diuji maka dapat dilihat *error* persen, angkanya tidak lebih dari sepuluh persen.

Dengan ini terbukti bahwa rangkaian ini dapat bekerja sesuai yang diinginkan. Untuk lebih lengkapnya dapat ditunjukkan dalam Tabel 5.4 dan 5.5.

5.5. Pengujian Sistem Secara Keseluruhan

5.5.1. Tujuan Pengujian

Pengujian ini bertujuan untuk mengetahui apakah sistem yang telah dibuat mampu bekerja sesuai dengan perancangan baik perancangan perangkat keras (*hardware*) maupun perangkat lunak (*software*).

5.5.2. Peralatan Pengujian

Dalam pengujian sistem secara keseluruhan diperlukan peralatan-peralatan pengujian. Peralatan pengujian yang diperlukan sebagai berikut:

- Sumber tegangan AC 220V.
- Rangkaian minimum sistem ATmega 8535.

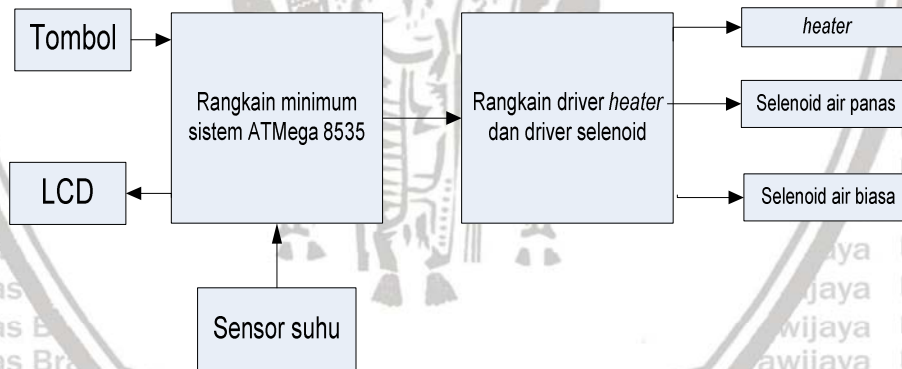
- Rangkaian sensor suhu.
- Rangkaian driver selenoid.
- Rangkaian driver *heater*.
- Selenoid.
- *Heater*.
- LCD 2 × 32

5.5.3. Prosedur Pengujian

Prosedur pengujian yang dilakukan adalah sebagai berikut:

- Memberi catu daya pada minimum sistem.
- Menggabungkan minimum sistem dengan rangkaian sensor suhu.
- Menggabungkan minimum sistem dengan rangkaian driver selenoid.
- Menggabungkan minimum sistem dengan rangkaian driver *heater*.
- Menggabungkan rangkaian driver selenoid dengan selenoid.
- Menggabungkan rangkaian driver selenoid dengan *heater*.
- Ditampilkan melalui LCD.

Blok diagram secara keseluruhan ditunjukkan dalam Gambar 5.4.



Gambar 5.4 Blok Diagram Secara Keseluruhan

5.5.4. Hasil Pengujian Sistem Keseluruhan

Dari hasil pengujian sistem keseluruhan dapat dilihat bahwa pengujian keseluruhan dapat bekerja. Hasil pengujian sistem keseluruhan ditunjukkan dalam Tabel 5.6.

Tabel 5.6 Hasil Pengujian Sistem Kseluruhan

N0	Suhu yang diinginkan (°C)	Suhu yang Terukur (°C)	Volume yang diinginkan (ml)	Pewaktuan (detik)	Hasil Pengujian (ml)	Kesalahan (%)
1	20	27	100	3	80	25
2	30	32	150	4,5	130	15
3	40	39	200	6	190	5,2
4	50	48	250	7,5	220	13,6
5	60	59	300	9	270	11,1
6	70	68	350	10,5	330	6,06
7	80	77	400	12	380	5
8	90	88	450	13,5	440	2,27
9	100	99	500	15	430	16,27

Pada pengujian sistem secara keseluruhan dapat mengetahui apakah pengujian volume air dan suhu yang dikeluarkan ini dapat bekerja dengan baik atau tidak. Dengan resolusi volume 50 ml. Perubahan keadaan saat galon air penuh, sedang, dan tersisa sedikit tidak terlalu jauh perubahannya 0,5 detik. Bila dilihat dari prosentase *error* persen pada keluaran volume air yang diuji maka dapat dilihat rata-rata kesalahan sebesar 11,05%.

BAB VI PENUTUP

6.1. Kesimpulan

Berdasarkan pengujian yang telah dilakukan baik pengujian per blok rangkaian maupun pengujian sistem secara keseluruhan dapat diambil kesimpulan sebagai berikut:

- 1) Sistem pemanas air otomatis dapat dirancang dengan cara mengontrol driver heater dan selenoid melalui mikrokontroler ATMEGA8535 yang telah diprogram. dengan kesalahan rata-rata antara hasil yang diinginkan antara 100 ml, 150 ml,.....500 ml dan hasil pengukuran sebesar 11,05%
- 2) Volume air yang keluar melalui dispenser dapat diatur dengan pemrograman timer yang terdapat dalam mikrokontroler dengan kesalahan rata-rata antara hasil yang diinginkan antara 100 ml, 150 ml,.....500 ml dan hasil pengujian sebesar 4,2%
- 3) Rangkaian tombol suhu dapat digunakan untuk memasukkan suhu dan volume yang diinginkan, mikrokontroler ATMEGA8535 digunakan Sebagai pemroses input/masukan dari tombol sehingga akan menghasilkan output/keluaran yang akan mengatur kinerja selenoid. Dan LCD digunakan untuk menampilkan data suhu dari dispenser serta setting suhu yang diinginkan melalui tombol.
- 4) Program yang disusun dengan menggunakan bahasa c dapat membaca data suhu dan volume yang dimasukkan dari tombol suhu, memproses input untuk mengontrol perangkat output (*heater* dan selenoid) dan menampilkan data suhu dan volume melalui LCD.

6.2 Saran

Dengan melihat hasil yang dicapai dalam perancangan dan kesimpulan yang ada, untuk pengembangan alat ke depan disarankan :

- 1) Menggunakan selenoid *valve* yang lebih presisi sehingga keluaran airnya sesuai yang kita inginkan
- 2) Menambahkan sistem/alat lain apabila listrik padam seperti penyimpanan listrik.



DAFTAR PUSTAKA

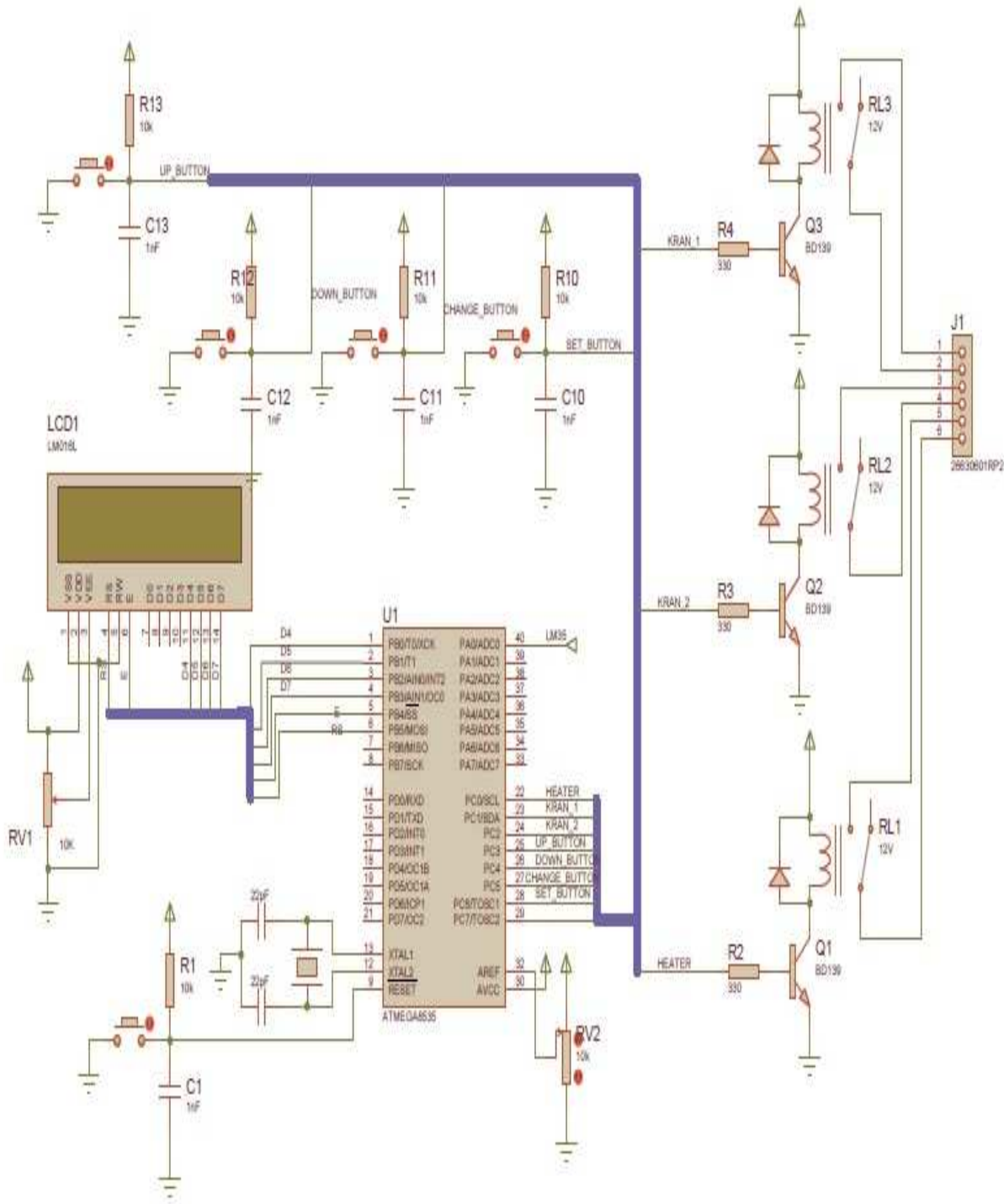
- Anonymous. 1997. *Liquid Crystal Displays, Standart Character Modules application Notes*. Seiko Instruments. Germany.
- Atmel. 1998. *Flash Microcontroller: Architectural Overview*, Atmel Inc. (<http://www.atmel.com>), USA. Diakses tanggal: 2 Agustus 2007
- Ayala, Kenneth J. 1991. *The 8051 Microcontroller: Architecture, Programming, and Application*. St. Paul: West Publishing Company.
- Malvino, Albert Paul. 1994. *Prinsip-Prinsip dan Penerapan Digital*, Edisi Ketiga, terjemahan Irwan Wijaya Jakarta: Erlangga.
- Malvino, Albert Paul. 1987. *Prinsip-Prinsip Elektronika*, Jilid 2, Cetakan Ketiga, terjemahan M. Barmawi, M.O Tjia. Jakarta: Erlangga.
- Saphi, Soedjana dan Osamu Nishino. 1986. *Pengukuran dan alat alat ukur listrik*. Jakarta: PT. Pradnya Paramita.
- National Semiconductor. 1999. *Linear Data Sheet Book*. California: National Semiconductor.
- Wardhana, Lingga. 2006. *Belajar Sendiri Mikrokontroller AVR Seri ATmega 8535*. Penerbit Andi: Yogyakarta.
- Wollard, Barry G., 2003, *Elektronika Praktis*, diIndonesiakan oleh H.Kristono, PT. Pradnya Paramita, Jakarta.
- Wasito S. 2006. *Vademekum Elektronika (Edisi Kedua)*. Jakarta PT Gramedia Pustaka Utama. http://www.seiko.com/seiko_manual.pdf. Diakses tanggal : 25 September 2007.



LAMPIRAN

LAMPIRAN-1

GAMBAR RANGKAIAN KESELURUHAN





LAMPIRAN-2
GAMBAR ALAT



LAMPIRAN-3 LISTING PROGRAM

```

#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdlib.h>
#include "lcd.h"
#include "type.h"
#include "adc_routine.h"
#include "dispenser.h"
#include "timer.h"
char buffer[10];
int time_buff;
int main ()
{
    init_pwm1(200);
    lcd_init();
    ADInit();
    InitValue();
    IOInit();
    DisplayTemp();
    sei();
    while (1)
    {
        a=wait_key();
        if ((a==1) && (a!=3) && (Dispenser.Memory.TempSet <= 99)
        && (Dispenser.Volume < 500))
        {
            if ((Dispenser.Memory.ChangeButton == 2) &&
            (Dispenser.Display.Mode != 1))
            {
                Dispenser.Memory.TempSet+=2;
            }
            if ((Dispenser.Memory.ChangeButton == 1) &&
            (Dispenser.Display.Mode != 1))
            {
                Dispenser.Volume+=50;
            }
        }
        if (a==2)
        {
            if ((Dispenser.Memory.ChangeButton == 2) &&
            (Dispenser.Display.Mode != 1) && (Dispenser.Memory.TempSet >=
            21))
            {
                Dispenser.Memory.TempSet--=2;
            }
            if ((Dispenser.Memory.ChangeButton == 1) &&
            (Dispenser.Display.Mode != 1) && (Dispenser.Volume>100))
            {
                Dispenser.Volume-=50;
            }
        }
    }
}

```

```

        }
    }
    if (a == 3)
    {
        if ((Dispenser.Memory.ChangeButton == 1) &&
            (Dispenser.Display.Mode != 1))
        {
            set_cursor(14,1);
            lcd_string("<<");
            set_cursor(14,2);
            lcd_string(" ");
        }
        if ((Dispenser.Memory.ChangeButton == 2) &&
            (Dispenser.Display.Mode != 1))
        {
            set_cursor(14,1);
            lcd_string(" ");
            set_cursor(14,2);
            lcd_string("<<");
        }
        if (Dispenser.Memory.ChangeButton >= 2)
        {
            Dispenser.Memory.ChangeButton = 0;
        }
        Dispenser.Memory.ChangeButton++;
    }
    if ( a == 4)
    {
        if (Dispenser.Display.Mode == 1)
        {
            set_cursor(13,2);
            lcd_string("S");
            HEATERL;
        }
        if ((Dispenser.Display.Mode == 2) &&
            (Dispenser.Memory.ChangeButton == 1))
        {
            set_cursor(13,2);
            lcd_string("COLD");
            SOLE2_H;
            time_buff = switch_delay(Dispenser.Volume);
            timer_on();
        }
        if (Dispenser.Display.Mode == 3)
        {
            set_cursor(13,2);
            lcd_string("R");
            set_cursor(14,1);
            lcd_string(" ");
            set_cursor(14,2);
            lcd_string("aw");
            if (display_adc(.) <=
                Dispenser.Memory.TempSet)
            {
                HEATERH;
            }
        }
    }
}

```

```
while (display_adc() <=
Dispenser.Memory.TempSet)
{
}
HEATERL;
SOLE1_H;
time_buff=switch_delay_2(Dispenser.Volume);
timer_on();
}
if (display_adc() > Dispenser.Memory.TempSet)
{
//HEATERH;
while (display_adc() >
Dispenser.Memory.TempSet)
{
}
HEATERL;
SOLE1_H;
time_buff=switch_delay_2(Dispenser.Volume);
timer_on();
}
}
if (Dispenser.Display.Mode>=3)
{
Dispenser.Display.Mode=0;
}
Dispenser.Display.Mode++;
}
a=0;
DisplayTemp();
}
return 0;
}
int x;
ISR (TIMER0_COMP_vect)
{
x++;
if (x >=time_buff)
{
x=0;
SOLE1_L;
SOLE2_L;
timer_off();
//Dispenser.Display.Mode=1;
//set_cursor(13,2);
//lcd_string("S");
}
}
```

```

#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdlib.h>
#include "lcd.h"
#include "type.h"
#include "adc_routine.h"
#include "dispenser.h"
void InitValue(void)
{
    Dispenser.Memory.ActTemp=0;
    Dispenser.Memory.TempSet=20;
    Dispenser.Memory.Histerisis=10;
    Dispenser.Memory.Heater=0;
    Dispenser.Memory.Kran_1=0;
    Dispenser.Memory.Kran_2=0;
    Dispenser.Memory.ChangeButton=0;
    Dispenser.Memory.UpButton=0;
    Dispenser.Memory.DownButton=0;
    Dispenser.Memory.SetButton=0;
    Dispenser.Display.Mode=0;
    Dispenser.Display.XCursor=1;
    Dispenser.Display.YCursor=2;
    Dispenser.Volume=100;
    flag_heater=0;
    a=4;
    set_cursor(0,1);lcd_string("AT__C #TS__C");
    set_cursor(0,2);lcd_string("V ___mL#Mode: ");

    Dispenser.Display.XCursor=1;
    Dispenser.Display.YCursor=2;
}
void DisplayTemp(void)
{
    itoa(Dispenser.Memory.ActTemp,buff,10);
    set_cursor(2,1);
    lcd_string(buff);

    itoa(Dispenser.Memory.TempSet,buff,10);
    set_cursor(10,1);
    lcd_string(buff);

    itoa(Dispenser.Volume,buff,10);
    set_cursor(2,2);
    lcd_string(buff);
}
void IOInit(void)
{
    DDRKey =0b00000111;
    PortPullUp = 0b00000111;
    HEATERL;
    SOLE1_L;
    SOLE2_L;
}

```





```
float display_adc(void)
{
    float adc_temp=0;
    char buff[10]={0,0,0,0,0,0,0,0,0,0};
    set_cursor(2,1);
    lcd_string(" ");
    set_cursor(2,1);
    adc_temp=ReadADC(1);
    adc_temp=(adc_temp/1023)*100;
    itoa(adc_temp,buff,10);
    lcd_string(buff);
    return adc_temp;
}

int switch_delay(uint16_t _delay_in)
{
    uint16_t _delay_buff=0;

    switch (_delay_in)
    {
        case 100:
        {
            _delay_buff=time100;
        }break;
        case 150:
        {
            _delay_buff=time150;
        }break;
        case 200:
        {
            _delay_buff=time200;
        }break;
        case 250:
        {
            _delay_buff=time250;
        }break;
        case 300:
        {
            _delay_buff=time300;
        }break;
        case 350:
        {
            _delay_buff=time350;
        }break;
        case 400:
        {
            _delay_buff=time400;
        }break;
        case 450:
        {
            _delay_buff=time450;
        }break;
        case 500:
        {
            _delay_buff=time500;
        }break;
    }
}
```



```
return _delay_buff;
}
int switch_delay_2(uint16_t _delay_in)
{
    uint16_t _delay_buff=0;
    switch (_delay_in)
    {
        case 100:
        {
            _delay_buff=time_100;
        }break;
        case 150:
        {
            _delay_buff=time_150;
        }break;
        case 200:
        {
            _delay_buff=time_200;
        }break;
        case 250:
        {
            _delay_buff=time_250;
        }break;
        case 300:
        {
            _delay_buff=time_300;
        }break;
        case 350:
        {
            _delay_buff=time_350;
        }break;
        case 400:
        {
            _delay_buff=time_400;
        }break;
        case 450:
        {
            _delay_buff=time_450;
        }break;
        case 500:
        {
            _delay_buff=time_500;
        }break;
    }
    return _delay_buff;
}
```

```
uint8_t ReadKeypad(void)
{
    if (SUPButton)
    {
        a=1;
    }
    if (SDownButton)
    {
        a=2;
    }
}
```



```

}
if (SChangeButton)
{
a=3;
}
if (SSetButton)
{
a=4;
}
delay_ms(300);
return a;
}
uint8_t wait_key()
{
a=ReadKeypad();
while (a == 0)
{
a=ReadKeypad();
display_adc();
}
return a;
}

#ifndef __dispenser__
#define __dispenser__
float display_adc(void);
void DisplayTemp(void);
void InitValue(void);
void IOInit(void);
uint8_t ReadKeypad(void);
uint8_t pool_menu();
int switch_delay(uint16_t _delay_in);
int switch_delay_2(uint16_t _delay_in);
uint8_t flag_heater;
uint8_t pool_mode();
uint8_t pool_down();
uint8_t pool_up();
void pool_run();
void pool_set_reset();
uint8_t pool_up_down();
uint8_t a;
uint8_t wait_key();
char buff[6];
#define READTEMP 1
#define READKEY 2
#define DISPLAYED3 3
#define SETOUT 4
#define COUNTMAX 5
#define NoKey 0

```



```

struct
{
  uint16_t Count;
  struct
  {
    unsigned ActTemp:10;
    unsigned TempSet:10;
    unsigned Histerisis:5;
    unsigned SetPoint:7;
    unsigned Heater:1;
    unsigned Kran_1:1;
    unsigned Kran_2:1;
  }Memory;
  struct
  {
    uint8_t Mode;
    unsigned XCursor:1;
    unsigned YCursor:16;
  }Display;
  unsigned Volume:10;
}Dispenser;
#define SUPButton      (!(PINC & 0x08)) //( PINC & (1<<PINC1))
#define SDownButton    (!(PINC & 0x10))
#define SChangeButton  (!(PINC & 0x20))
#define SSetButton     (!(PINC & 0x40))
#define PortPullUp     PORTC
#define DDRKey         DDRC
#define Up             DDC3
#define Down           DDC4
#define Change         DDC5
#define Set            DDC6
#define RELAYPORTPORTC
#define RELAYDDR DDRC
#define HEATERH       (RELAYPORT |= (1<<PINC0))
#define HEATERL       (RELAYPORT &= ~(1<<PINC0))
#define SOLE1_H       (RELAYPORT |= (1<<PINC1))
#define SOLE1_L       (RELAYPORT &= ~(1<<PINC1))
#define SOLE2_H       (RELAYPORT |= (1<<PINC2))
#define SOLE2_L       (RELAYPORT &= ~(1<<PINC2))
#define SetMode       1
#define ChangeMode    0
//suhu dingin
#define time100       100
#define time150       200
#define time200       300
#define time250       400
#define time300       500
#define time350       600
#define time400       700
#define time450       800
#define time500       900
//suhu panas +25

```



```
#define time_100 450
#define time_150 675
#define time_200 900
#define time_250 1125
#define time_300 1350
#define time_350 1575
#define time_400 1800
#define time_450 2025
#define time_500 2250
#endif
#ifndef __dispenser__
#define __dispenser__
float display_adc(void);
void DisplayTemp(void);
void InitValue(void);
void IOInit(void);
uint8_t ReadKeypad(void);
uint8_t pool_menu();
int switch_delay(uint16_t _delay_in);
int switch_delay_2(uint16_t _delay_in);
uint8_t flag_heater;
uint8_t pool_mode();
uint8_t pool_down();
uint8_t pool_up();
void pool_run();
void pool_set_reset();
uint8_t pool_up_down();
uint8_t a;
//long a;
uint8_t wait_key();

char buff[6];
#define READTEMP 1
#define READKEY 2
#define DISPLAYED3 3
#define SETOUT 4
#define COUNTMAX 5
#define NoKey 0
struct
{
uint16_t Count;
struct
{
unsigned ActTemp:10;
unsigned TempSet:10;
unsigned Histerisis:5;
unsigned SetPoint:7;
unsigned Heater:1;
unsigned Kran_1:1;
unsigned Kran_2:1;
uint8_t ChangeButton;
unsigned UpButton:1;
unsigned DownButton:1;
unsigned SetButton:1;
```

```

}Memory;
struct
{
uint8_t Mode;
unsigned XCursor:1;
unsigned YCursor:16;
}Display;
unsigned Volume:10;
}Dispenser;
#define SUPButton    (!(PINC & 0x08)) // (PINC & (1<<PINC1))
#define SDownButton  (!(PINC & 0x10))
#define SChangeButton  (!(PINC & 0x20))
#define SSetButton    (!(PINC & 0x40))
#define PortPullUp    PORTC
#define DDRKey        DDRC
#define Up            DDC3
#define Down          DDC4
#define Change        DDC5
#define Set           DDC6
#define RELAYPORTPORTC
#define RELAYDDR DDRC
#define HEATERH      (RELAYPORT |= (1<<PINC0))
#define HEATERL      (RELAYPORT &= ~(1<<PINC0))
#define SOLE1_H      (RELAYPORT |= (1<<PINC1))
#define SOLE1_L      (RELAYPORT &= ~(1<<PINC1))
#define SOLE2_H      (RELAYPORT |= (1<<PINC2))
#define SOLE2_L      (RELAYPORT &= ~(1<<PINC2))
#define SetMode      1
#define ChangeMode    0
//suhu dingin
#define time100      100
#define time150      200
#define time200      300
#define time250      400
#define time300      500
#define time350      600
#define time400      700
#define time450      800
#define time500      900
//suhu panas +225
#define time_100     450
#define time_150     675
#define time_200     900
#define time_250     1125
#define time_300     1350
#define time_350     1575
#define time_400     1800
#define time_450     2025
#define time_500     2250
#endif

```

```

#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdlib.h>
#include "lcd.h"
#include "adc_routine.h"
#include "dispenser.h"
void ADInit(void)
{
    ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS0);
    ADMUX = 0x00;
    ADCSRA |= (1<<ADSC);
    while (ADBUSY)
    {
    };
}
uint16_t ADResult = ADCW;
ADCW = 0;
ADResult=0;
}
uint16_t ReadADC(uint8_t ch)
{
    switch (ch)
    {
        case 1: {ADMUX = (0<<MUX2) | (0<<MUX1) | (0<<MUX0); }break;
        case 2: {ADMUX = (0<<MUX2) | (0<<MUX1) | (1<<MUX0); }break;
        case 3: {ADMUX = (0<<MUX2) | (1<<MUX1) | (0<<MUX0); }break;
        case 4: {ADMUX = (0<<MUX2) | (1<<MUX1) | (1<<MUX0); }break;
        case 5: {ADMUX = (1<<MUX2) | (0<<MUX1) | (0<<MUX0); }break;
        case 6: {ADMUX = (1<<MUX2) | (0<<MUX1) | (1<<MUX0); }break;
        case 7: {ADMUX = (1<<MUX2) | (1<<MUX1) | (0<<MUX0); }break;
        case 8: {ADMUX = (1<<MUX2) | (1<<MUX1) | (1<<MUX0); }break;
    }
    ADCSRA |= (1<<ADSC);
    while (ADBUSY)
    {
    };
    ADCSRA &= ~(1<<ADSC);
    Dispenser.Memory.ActTemp = ADCW;
    return Dispenser.Memory.ActTemp;
}
#endif
#define __adc_routine__
#define __adc_routine__
#define ADBUSY (ADCSRA & 0x40)
void ADInit(void);
uint16_t ReadADC(uint8_t ch);
#endif

```

```

#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <util/twi.h>
#include <stdlib.h>
#include "lcd.h"
uint8_t tmp;
void lcd_data(unsigned char temp1)
{
    unsigned char temp2 = temp1;
    LCD_PORT |= (1<<LCD_RS);
    temp1 = temp1 >> 4;
    temp1 = temp1 & 0x0F;
    LCD_PORT &= 0xF0;
    LCD_PORT |= temp1;
    lcd_enable();
    temp2 = temp2 & 0x0F;
    LCD_PORT &= 0xF0;
    LCD_PORT |= temp2;
    lcd_enable();
    _delay_us(420); //42
}
void lcd_command(unsigned char temp1)
{
    unsigned char temp2 = temp1;

    LCD_PORT &= ~(1<<LCD_RS);
    temp1 = temp1 >> 4;
    temp1 = temp1 & 0x0F;
    LCD_PORT &= 0xF0;
    LCD_PORT |= temp1;
    lcd_enable();

    temp2 = temp2 & 0x0F;
    LCD_PORT &= 0xF0;
    LCD_PORT |= temp2;
    lcd_enable();
    _delay_us(420); //42
}
void lcd_enable(void)
{
    LCD_PORT |= (1<<LCD_EN);
    _delay_us(1);
    LCD_PORT &= ~(1<<LCD_EN);
}
void lcd_init(void)
{
    LCD_DDR = LCD_DDR | 0x0F | (1<<LCD_RS) | (1<<LCD_EN);
    _delay_ms(200); //20
    LCD_PORT &= 0xF0;
    LCD_PORT |= 0x03;
    LCD_PORT &= ~(1<<LCD_RS);
    lcd_enable();
    _delay_ms(50); //5
    lcd_enable();
}

```



```
_delay_ms(10); //1
lcd_enable();
_delay_ms(10); //1
LCD_PORT &= 0xF0;
LCD_PORT |= 0x02;
lcd_enable();
_delay_ms(10); //1

lcd_command(0x28);
lcd_command(0x0C);
lcd_command(0x06);
lcd_clear();
}

void lcd_clear(void)
{
    lcd_command(CLEAR_DISPLAY);
    _delay_ms(50); //5
}

void lcd_home(void)
{
    lcd_command(CURSOR_HOME);
    _delay_ms(50); //5
}

void set_cursor(uint8_t x, uint8_t y)
{
    switch (y) {
        case 1: tmp=0x80+0x00+x; break;
        case 2: tmp=0x80+0x40+x; break;
        case 3: tmp=0x80+0x10+x; break;
        case 4: tmp=0x80+0x50+x; break;
    }
    lcd_command(tmp);
}

void lcd_string(char *data)
{
    while(*data) {
        lcd_data(*data);
        data++;
    }
}

void lcd_int(unsigned int a)
{
    char buff[6];
    itoa(a,buff,10);
    lcd_string(buff);
}

void cursor_blink_on(void)
{
    //0000 1101
    lcd_command(0x0D);
}

void cursor_blink_off(void)
{
    //0000 1100
    lcd_command(0x0C);
}
```



```
}
#endif
#define _lcd_
void lcd_data(unsigned char temp1);
void lcd_string(char *data);
void lcd_command(unsigned char temp1);
void lcd_enable(void);
void lcd_init(void);
void lcd_home(void);
void lcd_clear(void);
void set_cursor(uint8_t x, uint8_t y);
void cursor_blink_on(void);
void cursor_blink_off(void);
void lcd_int(unsigned int a);

#define CLEAR_DISPLAY 0x01
#define CURSOR_HOME 0x02

// Pinbelegung für das LCD, an verwendete Pins anpassen

#define LCD_PORT PORTB //PB0 -PB3
#define LCD_DDR DDRB
#define LCD_RS PB5
#define LCD_EN PB4

// DB4 bis DB7 des LCD sind mit PD0 bis PD3 des AVR verbunden
#endif
```

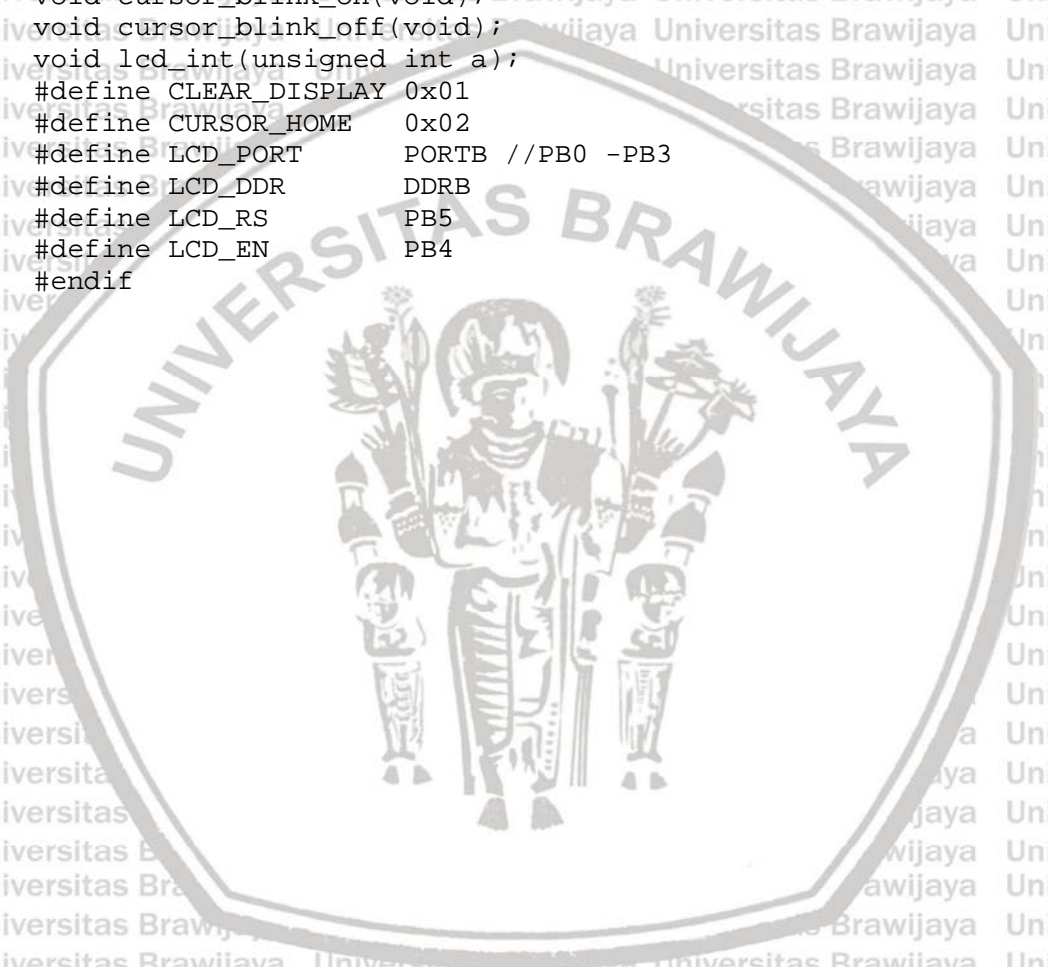




```

#ifndef _lcd_
#define _lcd_
void lcd_data(unsigned char temp1);
void lcd_string(char *data);
void lcd_command(unsigned char temp1);
void lcd_enable(void);
void lcd_init(void);
void lcd_home(void);
void lcd_clear(void);
void set_cursor(uint8_t x, uint8_t y);
void cursor_blink_on(void);
void cursor_blink_off(void);
void lcd_int(unsigned int a);
#define CLEAR_DISPLAY 0x01
#define CURSOR_HOME 0x02
#define LCD_PORT PORTB //PB0 -PB3
#define LCD_DDR DDRB
#define LCD_RS PB5
#define LCD_EN PB4
#endif

```




```
#include <avr/io.h>
#include <avr/sleep.h>
#include <avr/interrupt.h>
#include <util/delay.h>
#include <stdlib.h>
#include "timer.h"

void init_pwm1(unsigned char b)
{
    TCCR0=0b00001000;
    TIMSK=0x02;
    OCR0=b;
}

void timer_on()
{
    TCCR0=0b00001101;
    TIMSK=0x02;
}

void timer_off()
{
    TCCR0=0b00001000;
    TIMSK=0x02;
}

#ifdef __timer__
#define __timer__
void init_pwm1(unsigned char b);
void timer_on();
void timer_off();
struct
{
    uint16_t Counter;
}Timer;
#endif

#ifdef __window__
#define __window__

typedef void *LPVOID;
typedef signed short BOOL_T;
typedef char CHAR_T, *PCHAR_T;
typedef signed char INT8_T, *PINT8_T;
typedef unsigned char UINT8_T, *PUINT8_T;
typedef signed int INT16_T, *PINT16_T;
typedef unsigned int UINT16_T, *PUINT16_T;

#endif

#endif
```