SKRIPSI

RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA8535/16/32 DAN ATMEGA128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR

Disusun Oleh

MADE AGUS ANGGASANA PUTRA

08.12.213

PROGRAM STUDI TEKNIK ELEKTRO S-1

KONSENTRASI TEKNIK ELEKTRONIKA

FAKULTAS TEKNOLOGI INDUSTRI

INSTITUT TEKNOLOGI NASIONAL MALANG

2012

LEMBAR PERSETUJUAN

RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA8535/16/32 DAN ATMEGA128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR

SKRIPSI

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

Disusun oleh :

MADE AGUS ANGGASANA PUTRA

NIM. 08.12.213

Mengetahui,

Ketua Program Studi Teknik Elektro S-1

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Diperiksa dan Disetujui

Mengetahui Pembimbing I	Mengetahui Pembimbing II
M. Ibrahim Ashari, ST, MT	**Lauhil Mahfudz Hayusman, ST, MT**
NIP.P. 1030100358	**1144**

PROGRAM STUDI TEKNIK ELEKTRO S-1
KONSENTRASI TEKNIK ELEKTRONIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL MALANG
2012

SURAT PERNYATAAN ORISINALITAS

Yang bertanda tangan di bawah ini :

Nama : MADE AGUS ANGGASANA PUTRA

NIM : 08.12.213

Program Studi : TEKNIK ELEKTRO S-1

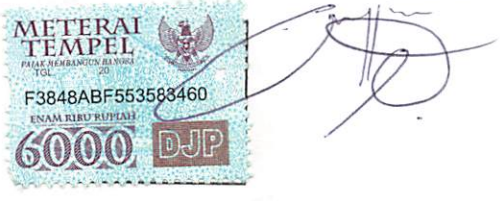
Konsentrasi : TEKNIK ELEKTRONIKA

Dengan ini menyatakan bahwa Skripsi yang saya buat adalah hasil karya sendiri, tidak merupakan plagiasi dari karya orang lain. Dalam Skripsi ini tidak memuat karya orang lain, kecuali dicantumkan sumbernya sesuai dengan ketentuan yang berlaku.

Demikian surat pernyataan ini saya buat, dan apabila di kemudian hari ada pelanggaran atas surat pernyataan ini, saya bersedia menerima sangsinya.

Malang, 28 November 2012

Yang membuat Pernyataan,



MADE AGUS ANGGASANA PUTRA

RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA 8535/16/32 DAN ATMEGA 128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR

Made Agus Anggasana Putra, NIM 08.12.213
Institut Teknologi Nasional Malang
Fakultas Teknologi Industri
Jurusan Teknik Elektro S-1
Konsentrasi Teknik Elektronika
e-mail : agus.anggasana@gmail.com

Dosen Pembimbing : M. Ibrahim Ashari, ST, MT dan
Lauhil Mahfudz Hayusman, ST, MT

Abstrak

Mikrokontroler merupakan salah satu mata kuliah penting dalam teknik elektro khusunya konsentrasi elektronika. Perkembangannya sampai saat ini telah merujuk pada penggunaan mikrokontroler seri ATMega Avr. Untuk menunjang sarana praktik pembelajaran maka dirancanglah modul praktikum mikrokontroler berbasis sistem minimum avr di Laboratorium Elektronika Digital ITN Malang. Diharapkan nantinya modul pelatihan mikrokontroler ini dapat memberi alternatif baru dalam pembelajaran dasar untuk kreatifitas mahasiswa Teknik Elektro ITN Malang yang mengarah pada perkembangan teknologi mikrokontroler ATMega Avr.

Modul praktikum mikrokontroler di Laboratorium Elektronika Digital ITN Malang ini terdiri dari sistem minimum berbasis AVR dengan chip ATMega8535/16/32 dan ATMega128L serta papan masukan/keluaran yang berisi beberapa antarmuka komponen dasar seperti LED, saklar, Dot Matriks display, LCD, 7segment;s, dan motor steper sebagai simulasi pemrograman yang dilakukan pada sistem minimum yang dirancang. Penulisan dan kompilasi program menggunakan CodeVisionAVR yang berbasis bahasa C.

Kata kunci : Mikrokontroler, ATMega Avr, modul praktikum mikrokontroler.

RANCANG BANGUN MODUL PELATIHAN MIKROKONTROLER ATMEGA 8535/16/32 DAN ATMEGA 128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR

Made Agus Anggasana Putra, NIM 08.12.213
Institut Teknologi Nasional Malang,
Fakultas Teknologi Industri,
Jurusan Teknik Elektro S-1,
Konsentrasi Teknik Elektronika
e-mail : agus.anggasana@gmail.com

Dosen Pembimbing : **M. Ibrahim Ashari, ST, MT dan**
Lauhil Mahfudz Hayusman, ST, MT

Abstract

Microcontroller is one of the important college subjects in electrical engineering especially the concentration of electronics. Progress of microcontroller at this present has been referred to the use of a Avr ATMega series. Microcontroller practicum modules based on avr minimum system is designed to support the practical means of learning in Digital Electronics Laboratory of Institut Teknologi Nasional Malang. This microcontroller practicum module is expected later can provide new alternative basic learning in creativity of Institut Teknologi Nasional Malang Students that lead the development of Avr ATMega technology.

The microcontroller practicum modulse that will be applied on Digital Electronics Laboratorof Institut Teknologi Nasional Malang y consist of Avr-minimum sistem based on ATMega8535/16/32 chip and ATMega128L chip, peripheral I/O board interface that content of basic component interface such as LED, switch, LED Dot Matrix Display, 7segment's to simulate programming designed on a minimum system. Program is write and compile using the CodeVisionAVR C-based language.

Keywordsx: Microcontroller, Avr ATMega, microcontroller practicum modules.

KATA PENGANTAR

Puji syukur kehadirat Ida Sang Hyang Widhi Wasa, Tuhan yang Maha Esa karena telah memberikan anugrah dan berkah-Nya, sehingga penulis dapat menyelesaikan laporan Skripsi ini dengan baik dan lancar.

Laporan Skripsi ini merupakan salah satu persyaratan akademik dalam menyelesaikan program Strata 1 Jurusan Teknik Elektro, Konsentrasi Elektronika, Institut Teknologi Nasional Malang. Adapun judul laporan Skripsi ini adalah:

RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA 8535/16/32 DAN ATMEGA 128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR

Selanjutnya pada kesempatan ini penulis juga menyampaikan rasa terimakasih yang sebesar-besarnya kepada pihak–pihak yang telah banyak membantu penulis selama penyusunan tugas akhir, diantaranya :

1. Bapak Ir. Yusuf Ismail Nahkoda, MT selaku Ketua Jurusan Teknik Elektro S-1 ITN Malang.
2. Bapak Dr. Aryuanto Soetedjo, ST, MT selaku Sekertaris Jurusan Teknik Elektro S-1 ITN Malang.
3. Bapak M. Ibrahim Ashari, ST, MT selaku Dosen Pembimbing I
4. Bapak Lauhil Mahfudz Hayusman, ST, MT selaku Dosen Pembimbing II
5. Bapak M. Ibrahuim Ashari, ST, MT selaku Dosen Wali.
6. Bapak Drs. I Wayan Nomer, Ibu Dra. Ni Wayan Suadi dan saudara perempuan saya Luh Gde Febriyani R. S.Sc yang telah memberikan dukungan baik materi, moril maupun spiritual dan untuk selalu berdoa dan berusaha beserta nasehat yang telah diberikan sampai saat ini.
7. Pacar :-D yang telah memberikan segala atensi dan pengertian selama penulisan laporan skripsi ini

8. Ardin Kaconk, Mazid Amnan, Eng Antonius Dian Toekidjo, dan Reza yang telah memberikan ruang tinggal, bantuan moril, serta mengurusi jadwal makan selama penyusunan skripsi ini di Puskopad22A
9. Seluruh dosen dan pegawai Kampus 2 ITN Malang.
10. Semua teman-teman Laboratorium Elektronika Digital dan Analog, Komunitas Robotika ITN Malang, dan teman-teman seperjuangan elektro '08 semuanya.
11. Semua pihak yang telah membantu penulis dalam menyelesaikan skripsi ini yang tidak bisa penulis sebutkan satu persatu.

Penulis berharap agar laporan laporan Skripsi ini dapat memberikan banyak manfaat bagi semua pihak yang membutuhkan, khususnya bagi rekan-rekan mahasiswa. Penulis menyadari bahwa dalam penyusunan laporan ini masih banyak kekurangan, oleh karena itu mohon maaf apabila dalam laporan ini terdapat hal-hal yang kurang berkenan dihati para pembaca.

Penulis juga mengharap koreksi, kritik serta saran-saran yang bermanfaat demi kesempurnaan laporan Laporan Skripsi ini.

Malang, Juli 2012

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR SINGKATAN

ADC	= *Analog to Digital Converson*
AVR	= *Alf and Vegard RISC Processor*
CISC	= *Complex Instruction Set Computing*
DAC	= *Digital to Analog Conversion*
DDR	= *Data Direction Register*
EEPROM	= *Electrically Erasable Programmabble Read Only Memory*
HLL	= *High Level Language*
IDE	= *Integrated Development Environtment*
ISP	= *In-system programming*
LCD	= *Liquid Crystal Display*
LED	= *Light Emitting Diode*
MISO	= *Master In Slave Out*
MOSI	= *Master Out Slave In*
PDIP	= *Plastic Dual In-line Package*
RISC	= *Reduce Instruction Set Computing*
SCK	= *Serial clock*
SPI	= *Serial Peripheral Interface*
SRAM	= *Static Random Access Memory*
TQFP	= *Thin Quad Flat Package*
TWI	= *Two Wire Setial Interface*
USART	= *Universal Synchronous and Asynchronous serial Receiver and Transmitter*
USB	= *Universal Serial Bus*

BAB I
PENDAHULUAN

1.1 Latar Belakang

Mikrokontroler dapat kita gunakan untuk berbagai aplikasi misalnya sistem pengendalian, otomatisasi industri, akuisisi data, telekomunikasi dan lain-lain. Andrianto(2008:1) menyatakan "Keunggulan menggunakan mikrokontroler yaitu harganya yang lebih murah, dapat diprogram berulang kali, dan dapat diprogram sesuai keinginan kita". Melihat begitu banyak manfaat dalam mempelajari mikrokontroler dan pemrogramannya, maka pelaksanaan praktikum yang menyangkut mata kuliah mikrokontroler sangat penting sebagai sarana pengembangan ide mahasiswa dalam mengimplementasikan ilmu yang aplikatif dalam kehidupan.

Pada saat ini pelaksanaan praktikum mikrokontroler di Laboratorium Elektronika Digital ITN Malang masih menggunakan modul mikrokontroler seri MCS-51 dimana instruksi bahasa pemrogramannya pun masih menggunakan instruksi mnemonic yaitu bahasa *assembler*. Padahal dalam perkembangan saat ini peran mikrokontroler seri MCS-51 sudah tergantikan dengan mikrokontroler seri ATMega yang penggunaannya lebih unggul.

AVR memiliki keunggulan dibandingkan mikrokontroler lain, keunggulan mikrokontroler AVR yaitu AVR memiliki kecepatan eksekusi pemrograman yang lebih cepat karena sebagian besar instruksi dieksekusi dalam 1 siklus *clock*, lebih cepat dibandingkan dengan mikrokontroler MCS51 yang memiliki arsitektur CISC(*Complex Instruction Set Compute*) di mana mikrokontroler MCS51 membutuhkan 12 siklus *clock* untuk mengeksekusi 1 instruksi(Andrianto, 2008:2).

Pada penelitian skripsi ini akan dibuat suatu modul pelatihan mikrokontroler ATMega8535/16/32 dan ATMega128L yang diimplementasikan pada Laboratorium Elektronika Digital ITN Malang berbasis minimum sistem AVR. Dimana perangkat antarmuka *Input/Output* yang disediakan merupakan perangkat antarmuka dasar sederhana dan instruksi bahasa pemrograman menggunakan bahasa C. Ardi(2010:3) mengatakan tentang bahasa C bahwa "Bahasa HLL(*High Level Language*) atau bahasa menengah lebih manusiawi sehingga pemrograman yang dibuat mudah dibaca dan dibuat. Menghemat waktu dan tenaga dalam pembuatan dibandingkan *assembler*". Pemilihan penggunaan bahasa C juga didasari bahwa dalam tugas akhir kuliah mahasiswa kebanyakan mengembangkan aplikasi dengan bahasa C, sehingga

penggunaan bahasa *assembler* saja dalam praktikum pada dasarnya akan menjadi tidak relevan dengan kebutuhan mahasiswa. Selain itu bahasa C telah banyak digunakan oleh pengembang untuk mengembangkan berbagai aplikasi diseluruh dunia, hal ini akan sangat membantu dalam referensi penggunaan bahasa C sendiri. Pemrograman dengan bahasa C sendiri dapat digunakan dengan berbagai perangkat lunak yang ada, namun dalm kasus ini penulis memilih CodeVisionAVR karena merupakan perangkat lunak yang dikhususkan untuk pengembangan mikrokontroler jenis AVR. Selain itu juga bahwa CodeVisionAVR memiliki fitur IDE yang bagus, fungsi pembuat kerangka software dengan CodeWizardAVR, library yang cukup lengkap sehingga secara kesatuan akan membuat pemahaman programer lebih baik untuk mengenal dasar pemrograman bahasaC pada mikrokontroler. Versi evaluasi dari CodeVisionAVR dapat didownload bebas secara gratis. Pembuatan modul ini diharapkan nantinya dapat menjadi pilihan alternatif yang dimiliki mahasiswa dalam pembelajaran praktik mikrokontroler di Laboratorium Elektronika Digital ITN Malang sehingga dapat memahami dan lebih antusias tentang mikrokontroler, yang dimana penggunaan lebih luas dapat digunakan sebagai implementasi aplikatif dalam bidang kehidupan.

1.2 Rumusan Masalah

Pada penelitian skripsi ini dapat dirumuskan masalah sebagai berikut :

a. Bagaimana menerapkan mikrokontroler ATMega8535/16/32 dan ATMega 128L sebagai sistem kontrol berbasis minimum sistem AVR?
b. Bagaimana merancang peripheral *Input/Output* yang terdiri dari LED, Saklar(*Switch*), *Seven Segment Display*, *Dot Matriks Display*, LCD, dan *Motor Stepper*?
c. Bagaimana merancang agar sistem kontrol dan peripheral *Input/Output* dapat terhubung secara sinkron menjadi sebuah modul praktikum mikrokotroler?

1.3 Tujuan dan Manfaat Penelitian

Tujuan dari penulisan skripsi ini adalah :

a. Menerapkan mikrokontroler ATMega8535/16/32 dan ATMega128L sebagai sistem kontrol terhadap *peripheral Input/Output* yang terdiri dari LED, *switch*(saklar), *7segment*, LED *Dot Matriks Display*, *LCD*, dan *Stepper Motor* dalam bentuk suatu paket modul praktikum mikrokontroler yang diimplementasikan di Laboratorium Elektronika Digital ITN Malang

b. Memberikan solusi yang efisien dan praktis dalam pembelajaran praktikum mikrokontroler di Laboratorium Elektronika Digital ITN Malang

Adapun manfaat yang diharapkan bisa diberikan dari penulisan skripsi ini adalah:

a. Dapat digunakan sebagai pilihan alternatif dalam melaksanakan praktikum mikrokontroler di Laboratorium Elektronika Digital ITN Malang
b. Dapat digunakan sebagai sarana yang membuat mahasiswa lebih antusias belajar dan memahami dasar mikrokontroler

1.4 Batasan Masalah

Agar permasalahan yang dibahas dalam penulisan skripsi ini tidak terlalu meluas, maka ruang lingkup pembahasan adalah sebagai berikut :

a. Membahas bagaimana perancangan perangkat keras modul praktikum mikrokontroler.
b. *Input/Output* yang digunakan merupakan antarmuka tingkat dasar sebagai sarana pembelajaran praktikum seperti *LED Display, Switch Input, Seven Segment Display, Matrix LED, Stepper motor, dan LCD.*
c. Mikrokontroler yang digunakan dalam minimum sistem AVR adalah seri dari mikrokontroler Atmel AVR ATMega8535/16/32(ATMEga32 pada percobaan) dan ATMega128L.
d. Bahasa pemrograman menggunakan bahasa C dengan editor dan kompilsai menggunakan program *CodeVision*AVR.
e. Panduan modul praktikum akan membahas tentang *input/output* dasar, *timer/counter, scanning 7segment, scanning* LED matriks(5x7), LCD, *stepper motor.*
f. Tidak membahas secara detail pembuatan komponen *Input/Output.*
g. Tidak membahas secara detail mengenai komunikasi dari PC.
h. Tidak membahas pembuatan perangkat ISP(*In System Programming*).
i. Tidak membahas tentang pembuatan catudaya

1.5 Sistematika Penulisan

Untuk mempermudah dan memahami pembahasan penulisan skripsi ini, maka sistimatika penulisan disusun sebagai berikut :

Bab I : **Pendahuluan**

Berisi Latar Belakang, Rumusan Masalah, Tujuan Penelitian, Batasan Permasalahan, dan Sistimatika Penulisan.

Bab II : **Landasan Teori**

Berisi tentang teori–teori yang akan dipakai baik dalam perancangan alat.

Bab III : **Perancangan dan Analisa Sistem**

Dalam bab ini berisi mengenai analisa kebutuhan sistim baik software maupun hardware yang diperlukan untuk membuat kerangka global yang menggambarkan mekanisme dari sistim yang akan dibuat.

Bab IV : **Pengujian alat dan pembahasan hasil**

Berisi tentang implementasi dari perancangan sistim yang telah dibuat serta pengujian terhadap sistim tersebut.

Bab V : **Penutup**

Merupakan bab terakhir yang memuat intisari dari hasil pembahasan yang berisikan kesimpulan dan saran yang dapat digunakan sebagai pertimbangan untuk pengembangan penulisan selanjutnya.

BAB II
LANDASAN TEORI

Bab ini berisi semua referensi yang berkaitan dengan penulisan skripsi tentang Modul praktikum Mikrokontroler ATMega8535/16/32 dan ATMega128L di Laboratorium Elektronika Digital ITN Malang berbasis Minimum Sistem AVR

2.1 Atmel Mega AVR

Mikrokontroler Atmel Mega AVR merupakan suatu mikrokontroler yang termasuk kedalam jenis mikrokontroler **RISC** (*Reduced Instruction Set Computing*) 8 bit. Pada mikrokontroler dengan teknologi RISC semua instruksi dikemas dalam kode 16 bit dan sebagian besar instruksi dieksekusi dalam 1 clock(Sholihul, 2003:1). AVR atau sebuah kependekan dari *Alf and Vegard's Risc Processor* merupakan chip mikrokontroler yang diproduksi oleh Atmel.

Atmel megaAVR merupakan seri keluarga mikrokontroler Atmel AVR yang didesain untuk mendukung rancangan sistem applikasi yang membutuhkan serangkaian jumlah kode yang besar, pemrosesan instruksi yang lebih cepat, konsumsi daya yang efisien tetapi tetap *powerfull*. Beberapa fitur yang disediakan oleh keluarga Atmel megaAVR adalah :

1. *Broad Family* – megaAVR menawarkan pilihan luas dalam tipe – tipe devais keluarga mikrokontroler Atmel megaAVR yang meliputi kondisi pemilihan dalam kapasitas memori, jumlah pin dan *peripheral*, serta sharing penggunaan kembali kode pemrograman.
2. *PicoPower technology* – jenis megaAVR tertentu memiliki fitur konsumsi daya yang sangat rendah
3. *High Integration* – fitur dalam chip megaAVR menyediakan kemampuan on-chip flash, SRAM, EEPROM terintegrasi, SPI, TWI, USART, USB, CAN, LIN, watchdog timer, pemilihan osilator eksternal dan internal untuk aplikasi yang membutuhkan kepresisian, dan pin *Input/Output* dengan fungsi umum. Fitur ini memungkinkan chip megaAVR didesain seminimal mungkin menjadi minimum sistem dengan kemampuan yang kompleks dalam berbagai bidang aplikasi.
4. *Analog function* – mimiliki fitur pendukung pemrosesan analog yang lebih maju, seperti ADC, DAC, sensor temperature yang terintegrasi dengan referensi tegangan, komparator analog. Dengan integrasi level pemrosesan analog yang lebih

tinggi maka akan lebih meringkas desain minimum sistem yang membutuhkan komponen pemrosesan analog eksternal.

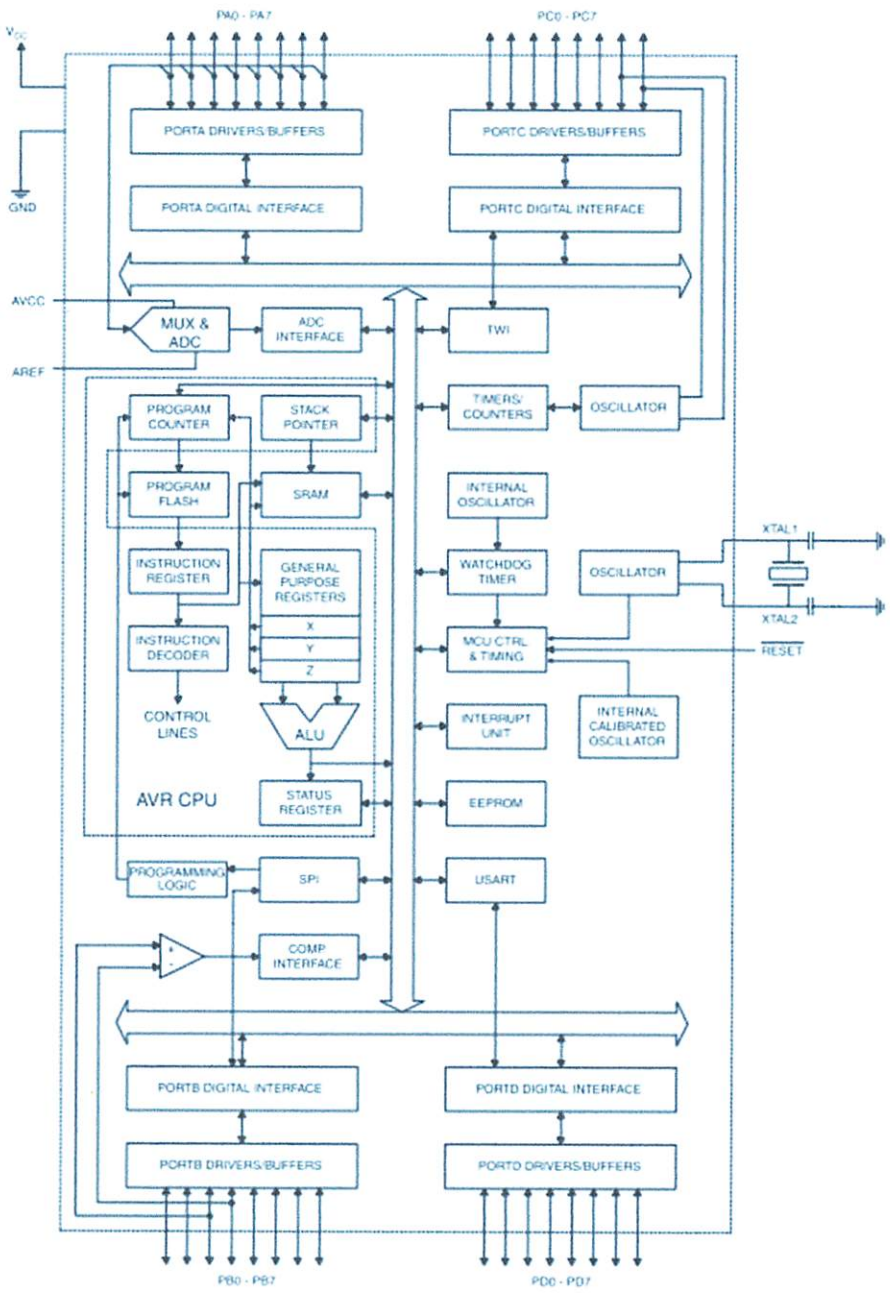
5. *Rapid development* – pengembangan pada mikrokontroler megaAVR begitu cepat dengan fitur *in-system programming*(ISP) dan *on-chip debug.*
   (Atmel, 2012)

Jenis mikrokontroler ATMEL Mega AVR yang digunakan dalam penelitian ini adalah ATMega8535/16/32 dan ATMega128L

2.1.1 ATMega8535/16/32

Mikrokontroler ATMega AVR seri 8535, 16, dan 32 memiliki kesamaan arsitektur serta konfigurasi Pinout pada tipe PDIP(*Plastic Dual In-Line Package*). Yang membedakan diantara ketiga seri mikrokontroler Atmel ini adalah kapasitas memori pada EEPROM, memori SRAM internal, serta memori Flashnya masing – masing.

2.1.1.1 Arsitektur Mikrokontroler ATMega8535/16/32



(Sumber : *Databook* ATMega8535, *Databook* ATMega16, *Databook* ATMega32)

Gambar 2.1
Diagram Blok Arsitektur Mikrokontroler ATMega8535

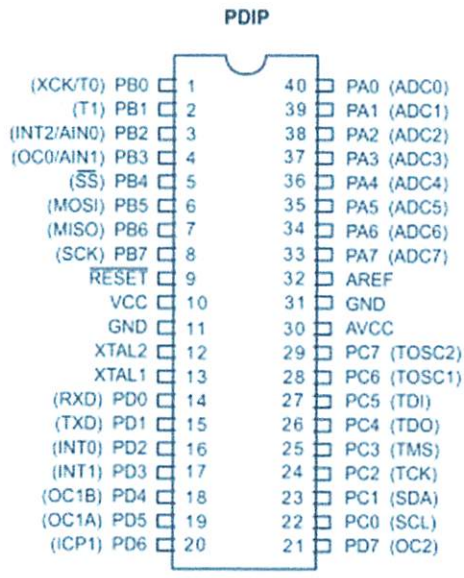
Dari gambar tersebut, dapat dilihat bahwa mikrokontroler ATMega8535/16/32 merupakan sistem mikroprosesor 8 bit berbasis RISC dengan kecepatan sampai 16MHz memiliki bagian – bagian sebagai berikut; saluran *Input/Output* sebanyak 32 buah(*Port*A, *Port*B, *Port*C, dan *Port*D), ADC(*Analog to Digital Converter*) 8 saluran dengan resolusi 10 bit, tiga buah *Timer/Counter* dengan kemampuan pembandingan(*Analog Comparator*), CPU yang terdiri dari 32 Register dan *Watchdog*

timer dengan *Internal Oscillator,* 4 *channel* PWM(Atmel, 2010). Selain itu mikrokontroller ini juga masing – masing memiliki

- **ATMega8535** : SRAM internal sebesar 512*Bytes*, *flash memory* sebesar 8 *KiloBytes* dengan kemampuan *Read-While-Write*, unit interupsi internal dan eksternal, *Port* antarmuka SPI, EEPROM sebesar 512 *Bytes* yang dapat diprogram saat operasi, dan *Port* USART untuk komunikasi serial.
- **ATMega16** : SRAM internal sebesar 512*Bytes*, *flash memory* sebesar 16 *KiloBytes* dengan kemampuan *Read-While-Write*, unit interupsi internal dan eksternal, *Port* antarmuka SPI, EEPROM sebesar 1*KiloBytes* yang dapat diprogram saat operasi, dan *Port* USART untuk komunikasi serial.
- **ATMega 32** : SRAM internal sebesar 1024*Bytes*, *flash memory* sebesar 32 *KiloBytes* dengan kemampuan *Read-While-Write*, unit interupsi internal dan eksternal, *Port* antarmuka SPI, EEPROM sebesar 2*KiloBytes* yang dapat diprogram saat operasi, dan *Port* USART untuk komunikasi serial.

  (Atmel, 2010)

2.1.1.2 Konfigurasi Pinout ATMega 8535/16/32



(Sumber : *Databook* ATMega 8535, ATMega16, Atmega32)

Gambar 2.2
Konfigurasi Pinout ATMega8535/16/32 (PDIP)

ATMega8535/16/32 mempunyai empat buah *Port* yang bernama *Port*A, *Port*B, *Port*C, dan *Port*D. Keempat *Port* tersebut merupakan jalur *bidirectional* dengan pilihan *internal pull-up*. Tiap *Port* mempunyai tiga buah register bit, yaitu DDxn, *PORT*xn, dan PINxn. Huruf 'x'mewakili nama huruf dari *Port* sedangkan huruf 'n' mewakili nomor bit. Bit DDxn terdapat pada *Input/Output address* DDRx, bit *PORT*xn terdapat pada *Input/Output address PORT*x, dan bit PINxn terdapat pada *Input/Output address* PINx. Bit DDxn dalam register DDRx (*Data Direction Register*) menentukan arah pin(Sholihul, 2003:2). Bila DDxn diset 1 maka Px berfungsi sebagai pin output. Bila DDxn diset 0 maka Px berfungsi sebagai pin input.Bila *PORT*xn diset 1 pada saat pin terkonfigurasi sebagai pin input, maka resistor *pull-up* akan diaktifkan. Untuk mematikan resistor *pull-up*, *PORT*xn harus diset 0 atau pin dikonfigurasi sebagai pin output. Pin *Port* adalah *tri-state* setelah kondisi reset. Bila *PORT*xn diset 1 pada saat pin terkonfigurasi sebagai pin output maka pin *Port* akan berlogika 1. Dan bila *PORT*xn diset 0 pada saat pin terkonfigurasi sebagai pin output maka pin *Port* akan berlogika 0. Saat mengubah kondisi *Port* dari kondisi *tri-state* (DDxn=0, *PORT*xn=0) ke kondisi output *High* (DDxn=1, *PORT*xn=1) maka harus ada kondisi peralihan apakah itu kondisi *pull-up enabled* (DDxn=0, *PORT*xn=1) atau kondisi output *Low* (DDxn=1, *PORT*xn=0).

Tabel 2.1
Konfigurasi Pin *Port* Mikrokontroler ATMega 8535/16/32

(Sumber : *Databook* ATMega 8535, ATMega16, ATMega32)

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

Berikut penjelasan dari fungsi pada masing – masing konfigurasi *Port* mikrokontroler ATMega8535/16/32 :

- *Port*A : menyediakan input analog untuk fitur ADC, selain itu *port*A juga bisa digunakan sebagai jalur *bi-directional Input/Output* jika fitur ADC tidak digunakan.

Tabel 2.2
Fungsi Alternatif Konfigurasi Pin *Port*A

(Sumber : *Databook* ATMega 8535, ATMega16, ATMega32)

Port Pin	Alternate Function
PA7	ADC7 (ADC input channel 7)
PA6	ADC6 (ADC input channel 6)
PA5	ADC5 (ADC input channel 5)
PA4	ADC4 (ADC input channel 4)
PA3	ADC3 (ADC input channel 3)
PA2	ADC2 (ADC input channel 2)
PA1	ADC1 (ADC input channel 1)
PA0	ADC0 (ADC input channel 0)

Dari data pada tabel 2.2 dapat diketahui bahwa Port A dapat berfungsi sebagai jalur input AD *Converter* 8 *channel.*Bila tidak digunakan dalam mode pemrograman alternatif tersebut maka PortA dapat digunakan sebagai *biderctional Input/Output.*

- *Port*B : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain *timer/counter*, komparator analog, serta fitur SPI.

Tabel 2.3
Fungsi Alternatif Konfigurasi Pin *Por*B

(Sumber : *Databook* ATMega 8535, ATMega16, ATMega32)

Port Pin	Alternate Functions
PB7	SCK (SPI Bus Serial Clock)
PB6	MISO (SPI Bus Master Input/Slave Output)
PB5	MOSI (SPI Bus Master Output/Slave Input)
PB4	$\overline{SS}$ (SPI Slave Select Input)
PB3	AIN1 (Analog Comparator Negative Input) OC0 (Timer/Counter0 Output Compare Match Output)
PB2	AIN0 (Analog Comparator Positive Input) INT2 (External Interrupt 2 Input)
PB1	T1 (Timer/Counter1 External Counter Input)
PB0	T0 (Timer/Counter0 External Counter Input) XCK (USART External Clock Input/Output)

Dari data tabel 2.3 *Port*B4-7 merupakan konfigurasi jalur data SPI. Sedangakan *Port*B0-3 merupakan konfigurasi timer/counter. Bila tidak digunakan pada mode permograman tersebut maka *Port*B dapat digunakan menjadi *bidirectional Input/Output*

- *Port*C : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain TWI, komparator analog, serta *Timer Oscillator.*

Tabel 2.4
Fungsi Alternatif Konfigurasi Pin *Port*C

(Sumber : *Databook* ATMega 8535, ATMega16, ATMega32)

Port Pin	Alternate Function
PC7	TOSC2 (Timer Oscillator Pin 2)
PC6	TOSC1 (Timer Oscillator Pin 1)
PC1	SDA (Two-wire Serial Bus Data Input/Output Line)
PC0	SCL (Two-wire Serial Bus Clock Line)

Dari data tabel 2.4 *Port*C0-1 merupakan konfigurasi jalur data TWI(*Two Wire Interface*). Sedangakan *Port*C6-7 merupakan konfigurasi timer/counter. Bila tidak digunakan pada mode permograman tersebut maka *Port*C dapat digunakan menjadi *bidirectional Input/Output*

- *Port*D : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain komparator analog, interupsi eksternal, serta komunikasi serial.

Tabel 2.5
Fungsi Alternatif Konfigurasi Pin *Port*D

(Sumber : *Databook* ATMega 8535, ATMega16, ATMega32)

Port Pin	Alternate Function
PD7	OC2 (Timer/Counter2 Output Compare Match Output)
PD6	ICP1 (Timer/Counter1 Input Capture Pin)
PD5	OC1A (Timer/Counter1 Output Compare A Match Output)
PD4	OC1B (Timer/Counter1 Output Compare B Match Output)
PD3	INT1 (External Interrupt 1 Input)
PD2	INT0 (External Interrupt 0 Input)
PD1	TXD (USART Output Pin)
PD0	RXD (USART Input Pin)

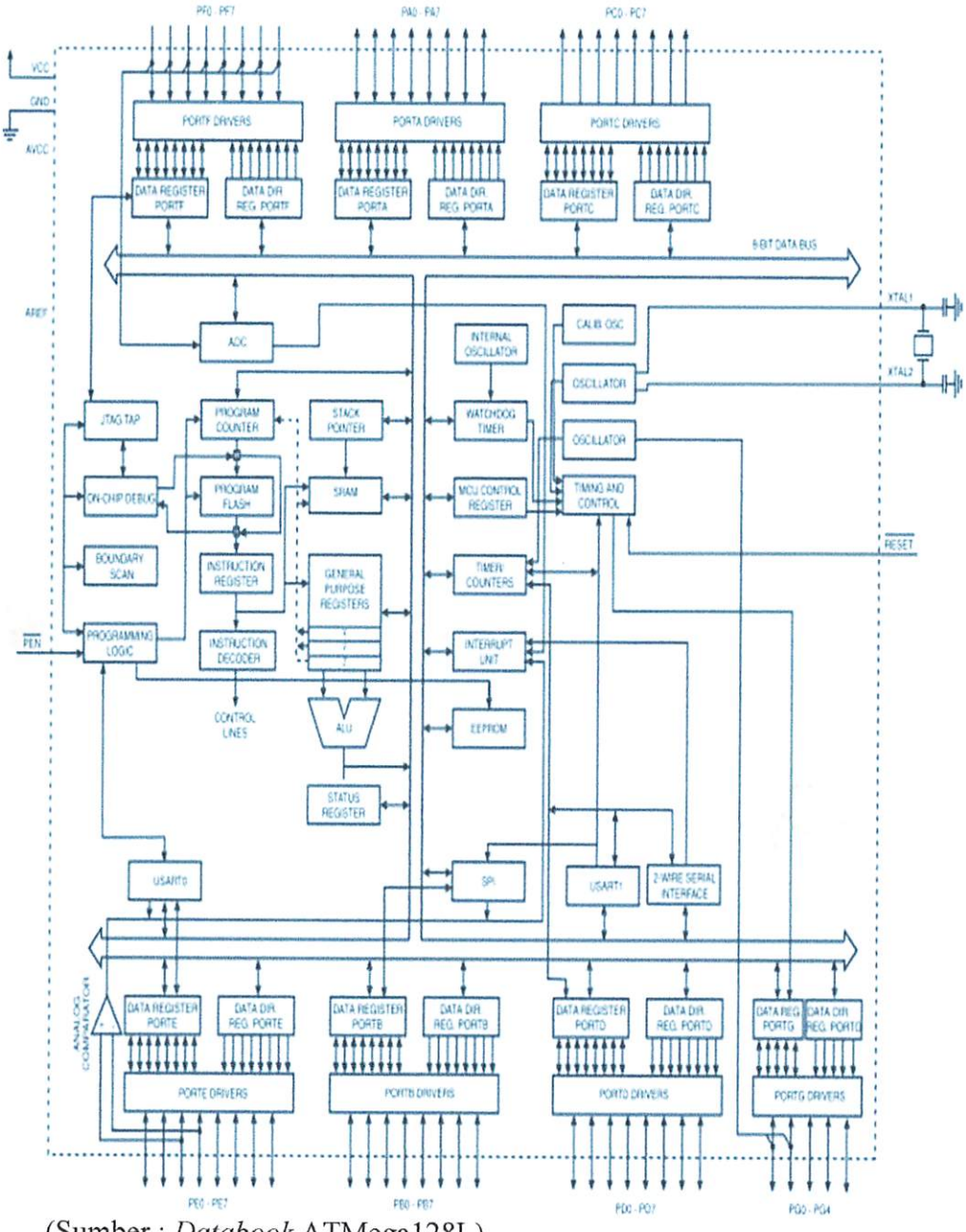
Dari data tabel 2.5 *Port*D4-7 merupakan konfigurasi *timer*. Sedangakan *Port*D2-3 merupakan konfigurasi fungsi interupsi. Sementara *Port*D0-1 merupakan konfigurasi fungsi komunikasi serial. Bila tidak digunakan pada mode permograman tersebut maka *Port*D dapat digunakan menjadi *bidirectional Input/Output*

2.1.2 ATMega128L

Berbeda dengan tipe ATMega8535/16/32, mikrokontroler tipe ATMega128L memiliki konfigurasi pinout yang lebih besar serta fitur – fitur khusus yang lebih banyak.

...

2.1.2.1 Arsitektur Mikrokontroler ATMega128L



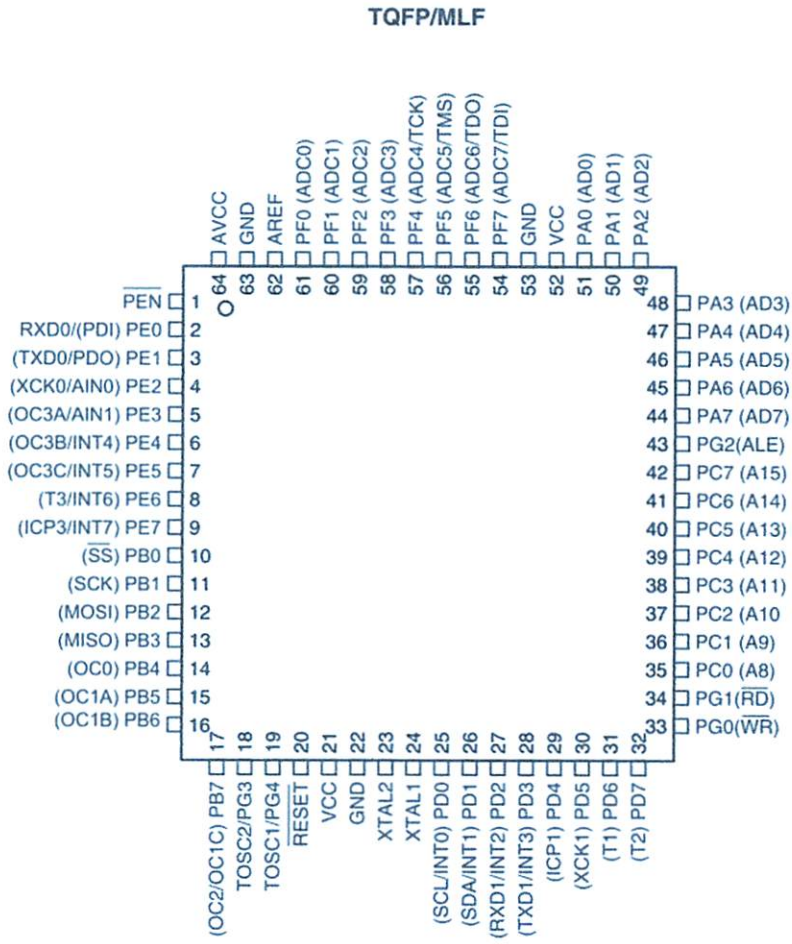
(Sumber : *Databook* ATMega128L)

Gambar 2.3
Diagram Blok Arsitektur Mikrokontroler ATMega128L

Dari gambar tersebut, dapat dilihat bahwa mikrokontroler ATMega128L merupakan sistem mikroprosesor 8 bit berbasis advance RISC dengan teknologi *Low* power berkecepatan sampai 16MHz memiliki bagian – bagian sebagai berikut; saluran *Input/Output* sebanyak 53 buah(*Port*A, *Port*B, *Port*C, *Port*D , *Port*E, *Port*F dan *Port*G), ADC(*Analog to Digital Converter*) 8 saluran dengan resolusi 10 bit, empat buah

Timer/Counter dengan kemampuan pembandingan(*Analog Comparator*), CPU yang terdiri dari 32 Register dan *Watchdog timer* dengan *Internal Oscillator*, 8 PWM *channel*. Selain itu mikrokontroller ini juga memiliki SRAM internal sebesar 4*KiloBytes*, *flash memory* sebesar 128*KiloBytes* dengan kemampuan *Read-While-Write*, unit interupsi internal dan eksternal, *Port* antarmuka SPI, EEPROM sebesar 4*KiloBytes* yang dapat diprogram saat operasi, dan *Port* USART untuk komunikasi serial(Atmel, 2011).

2.1.2.2 Konfigurasi Pinout Mikrokontroler ATMega128L



(Sumber : *Databook* ATMega128L)

Gambar 2.4
Konfigurasi Pinout Mikrokontroler ATMega128L (TQFP)

ATMega128L mempunyai tujuh buah *Port* yang bernama *Port*A, *Port*B, *Port*C, *Port*D, *Port*E, *Port*F, dan *Port*G. Ketujuh *Port* tersebut merupakan jalur *bidirectional* dengan pilihan *internal pull-up*. Tiap *Port* mempunyai tiga buah register bit, yaitu DDxn, *PORT*xn, dan PINxn. Huruf 'x'mewakili nama huruf dari *Port* sedangkan huruf 'n' mewakili nomor bit. Bit DDxn terdapat pada *Input/Output*

address DDRx, bit *PORT*xn terdapat pada *Input/Output address PORT*x, dan bit PINxn terdapat pada *Input/Output address* PINx. Bit DDxn dalam register DDRx (*Data Direction Register*) menentukan arah pin. Bila DDxn diset 1 maka Px berfungsi sebagai pin output. Bila DDxn diset 0 maka Px berfungsi sebagai pin input.Bila *PORT*xn diset 1 pada saat pin terkonfigurasi sebagai pin input, maka resistor *pull-up* akan diaktifkan. Untuk mematikan resistor *pull-up*, *PORT*xn harus diset 0 atau pin dikonfigurasi sebagai pin output. Pin *Port* adalah *tri-state* setelah kondisi reset. Bila *PORT*xn diset 1 pada saat pin terkonfigurasi sebagai pin output maka pin *Port* akan berlogika 1. Dan bila *PORT*xn diset 0 pada saat pin terkonfigurasi sebagai pin output maka pin *Port* akan berlogika 0. Saat mengubah kondisi *Port* dari kondisi *tri-state* (DDxn=0, *PORT*xn=0) ke kondisi output *High* (DDxn=1, *PORT*xn=1) maka harus ada kondisi peralihan apakah itu kondisi *pull-up enabled* (DDxn=0, *PORT*xn=1) atau kondisi output *Low* (DDxn=1, *PORT*xn=0)(Atmel, 2011).

Biasanya, kondisi *pull-up enabled* dapat diterima sepenuhnya, selama lingkungan impedansi tinggi tidak memperhatikan perbedaan antara sebuah strong *High* driver dengan sebuah *pull-up*. Jika ini bukan suatu masalah, maka bit PUD pada register SFIOR dapat diset 1 untuk mematikan semua *pull-up* dalam semua *Port*. Peralihan dari kondisi input dengan *pull-up* ke kondisi output *Low* juga menimbulkan masalah yang sama. Kita harus menggunakan kondisi *tri-state*(DDxn=0, *PORT*xn=0) atau kondisi output *High* (DDxn=1, *PORT*xn=0) sebagai kondisi transisi. Hal ini ditunjukan dalam tabel 2.6

Tabel 2.6
Konfigurasi Pin *Port* Mikrokontroler ATMega128L

(Sumber : *Databook* ATMega128L)

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

Berikut penjelasan dari fungsi pada masing – masing konfigurasi *Port* mikrokontroler ATMega128L:

- *Port*A : menyediakan jalur *bi-directional Input/Output* dan sebagai bit informasi alamat memori antarmuka eksternal(*Low byte*). Hal ini ditunjukan pada tabel 2.7

Tabel 2.7
Fungsi Alternatif Konfigurasi Pin *Port*A ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PA7	AD7 (External memory interface address and data bit 7)
PA6	AD6 (External memory interface address and data bit 6)
PA5	AD5 (External memory interface address and data bit 5)
PA4	AD4 (External memory interface address and data bit 4)
PA3	AD3 (External memory interface address and data bit 3)
PA2	AD2 (External memory interface address and data bit 2)
PA1	AD1 (External memory interface address and data bit 1)
PA0	AD0 (External memory interface address and data bit 0)

- *Port*B : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain PWM, *timer/counter*, komparator analog, serta fitur SPI. Hal ini ditunjukan pada tabel 2.8

Tabel 2.8
Fungsi Alternatif Konfigurasi Pin *Port*B ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Functions
PB7	OC2/OC1C(1) (Output Compare and PWM Output for Timer/Counter2 or Output Compare and PWM Output C for Timer/Counter1)
PB6	OC1B (Output Compare and PWM Output B for Timer/Counter1)
PB5	OC1A (Output Compare and PWM Output A for Timer/Counter1)
PB4	OC0 (Output Compare and PWM Output for Timer/Counter0)
PB3	MISO (SPI Bus Master Input/Slave Output)
PB2	MOSI (SPI Bus Master Output/Slave Input)
PB1	SCK (SPI Bus Serial Clock)
PB0	$\overline{SS}$ (SPI Slave Select input)

- *Port*C : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus sebagai bit informasi alamat memori antarmuka eksternal(*High byte*). Hal ini ditunjukan pada tabel 2.9

Tabel 2.9
Fungsi Alternatif Konfigurasi Pin *Port*C ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PC7	A15
PC6	A14
PC5	A13
PC4	A12
PC3	A11
PC2	A10
PC1	A9
PC0	A8

- *Port*D : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain input *timer/counter*, interupsi eksternal, serta komunikasi serial. Hal ini ditunjukan pada tabel 2.10

Tabel 2.10
Fungsi Alternatif Konfigurasi Pin *Port*D ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PD7	T2 (Timer/Counter2 Clock Input)
PD6	T1 (Timer/Counter1 Clock Input)
PD5	XCK1[1] (USART1 External Clock Input/Output)
PD4	ICP1 (Timer/Counter1 Input Capture Pin)
PD3	INT3/TXD1[1] (External Interrupt3 Input or UART1 Transmit Pin)
PD2	INT2/RXD1[1] (External Interrupt2 Input or UART1 Receive Pin)
PD1	INT1/SDA[1] (External Interrupt1 Input or TWI Serial DAta)
PD0	INT0/SCL[1] (External Interrupt0 Input or TWI Serial CLock)

- *Port*E : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain komparator analog, interupsi eksternal, serta komunikasi serial(UART). Hal ini ditunjukan pada tabel 2.11

Tabel 2.11
Fungsi Alternatif Konfigurasi Pin *Port*E ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PE7	INT7/ICP3[1] (External Interrupt 7 Input or Timer/Counter3 Input Capture Pin)
PE6	INT6/ T3[1] (External Interrupt 6 Input or Timer/Counter3 Clock Input)
PE5	INT5/OC3C[1] (External Interrupt 5 Input or Output Compare and PWM Output C for Timer/Counter3)
PE4	INT4/OC3B[1] (External Interrupt 4 Input or Output Compare and PWM Output B for Timer/Counter3)
PE3	AIN1/OC3A [1] (Analog Comparator Negative Input or Output Compare and PWM Output A for Timer/Counter3)
PE2	AIN0/XCK0[1] (Analog Comparator Positive Input or USART0 external clock input/output)
PE1	PDO/TXD0 (Programming Data Output or UART0 Transmit Pin)
PE0	PDI/RXD0 (Programming Data Input or UART0 Receive Pin)

- *Port*F : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain kanal ADC serta JTAG *interface*. Hal ini ditunjukan pada tabel 2.12

Tabel 2.12
Fungsi Alternatif Konfigurasi Pin *Port*F ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PF7	ADC7/TDI (ADC input channel 7 or JTAG Test Data Input)
PF6	ADC6/TDO (ADC input channel 6 or JTAG Test Data Output)
PF5	ADC5/TMS (ADC input channel 5 or JTAG Test mode Select)
PF4	ADC4/TCK (ADC input channel 4 or JTAG Test Clock)
PF3	ADC3 (ADC input channel 3)
PF2	ADC2 (ADC input channel 2)
PF1	ADC1 (ADC input channel 1)
PF0	ADC0 (ADC input channel 0)

- *Port*G : menyediakan fitur *bi-directional Input/Output* dan beberapa fungsi pin dengan fitur khusus, antara lain RTC *oscillator timer/counter*. Hal ini ditunjukan pada tabel 2.13.

Tabel 2.13
Fungsi Alternatif Konfigurasi Pin *Port*G ATMega128L

(Sumber : *Databook* ATMega128L)

Port Pin	Alternate Function
PG4	TOSC1 (RTC Oscillator Timer/Counter0)
PG3	TOSC2 (RTC Oscillator Timer/Counter0)
PG2	ALE (Address Latch Enable to external memory)
PG1	$\overline{RD}$ (Read strobe to external memory)
PG0	$\overline{WR}$ (Write strobe to external memory)

2.2 *Peripheral Input/Output*

Peripheral adalah sesuatu yang mengacu pada peralatan yang dihubungkan secara eksternal pada perangkat kontrol utama. Dalam hal ini peripheral *Input/Output* meliputi komponen elektronika yang dapat dihubungkan secara eksternal pada kontrol utama minimum sistem sebagai perangkat *Input/Outpu*). Peripheral *Input/Output* pada modul mikrokontroler yang akan dirancang meliputi.

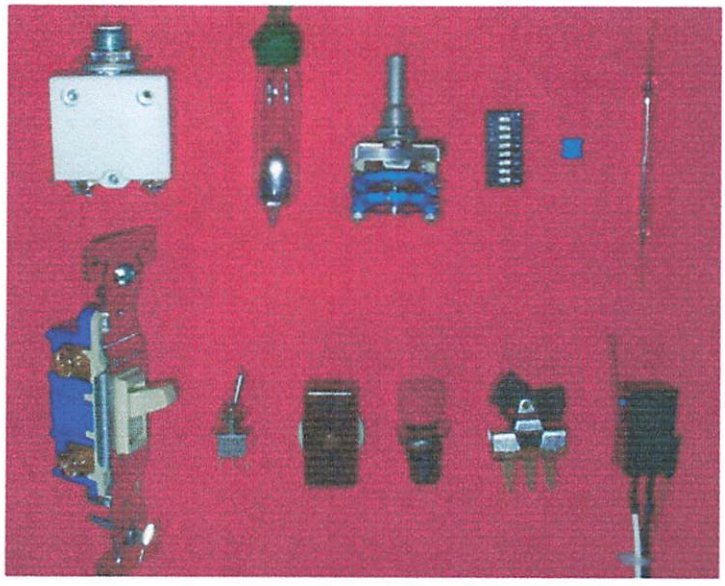
2.2.1 *Switch Input*

Dalam elektronika, *Switch* ialah suatu komponen elektronika yang dapat memutus arus, menginterupsi arus atau membagi arus pada suatu konduktor ke konduktor lain. *Switch* yang paling familiar adalah suatu komponen elektronika yang dioprasikan manual secara elektromekanikal dan memiliki satu atau lebih set kondisi kontak elektrik yang dihubungkan dengan sirkuit eksternal. Setiap suatu keadaan kontak dapat berarti suatu dari dua situasi yang ada yaitu situasi loop tertutup(*closed*) dimana kontak bersentuhan untuk mengalirkan arus listrik pada rangkaian/sirkuit yang dihubungkannya, atau situasi terbuka(*open*) dimana kontak tak bersentuhan sehingga tak ada arus yang mengalir pada rangkaian/sirkuit yang dihubungkannya(Wikipedia, 2012).

Dalam dunia elektronika suatu *Switch* ideal memiliki kriteria sebagai berikut :

- Tak memiliki nilai batas arus saat berada pada kondisi "ON"

- Memiliki nilai kapasitansi tak hingga saat berada pada kondisi "OFF"
- Tak menurunkan nilai tegangan selama berada pada kondisi "ON"
- Waktu respon adalah 0 detik, ini berarti tak ada penundaan terhadap respon selama pergantian kondisi dilakukan dari keadaan "OFF" ke "ON" maupun sebaliknya(wikipedia, 2012).



(Sumber : www.wikipedia.org/en/switch)

Gambar 2.5
Beberapa Contoh *Switch*

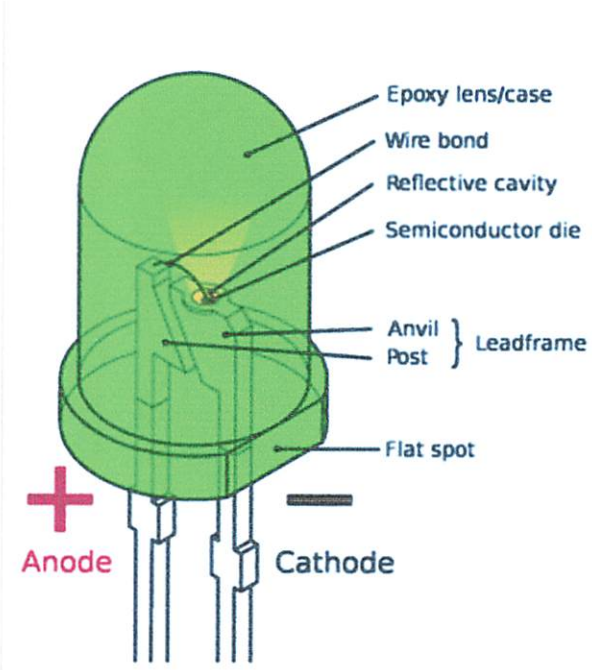
Pada gambar 2.5 ditunjukan beberapa macam *switch*(saklar) elektromekanikal yang sering digunakan. Tampak pada gambar beberapa tipe saklar geser dan *push button.*

2.2.2 LED *Display*

Dioda cahaya atau lebih dikenal dengan sebutan LED (*light-emitting diode*) adalah suatu semikonduktor yang memancarkan cahaya monokromatik yang tidak koheren ketika diberi tegangan maju. Sebuah LED adalah sejenis dioda semikonduktor istimewa. Seperti sebuah dioda normal, LED terdiri dari sebuah chip bahan semikonduktor yang diisi penuh, atau didop, dengan ketidakmurnian untuk menciptakan sebuah struktur yang disebut p-n junction. Pembawa muatan elektron dan *hole* mengalir ke junction elektroda dengan tegangan berbeda. Ketika elektron bertemu dengan *hole*, dia jatuh ke tingkat energi yang lebih rendah, dan melepas energi dalam bentuk photon(Wikipedia, 2011).

Tak seperti lampu pijar dan neon, LED mempunyai kecenderungan polarisasi. Chip LED mempunyai kutub positif dan negatif (p-n) dan hanya akan menyala bila

diberikan arus maju. Ini dikarenakan LED terbuat dari bahan semikonduktor yang hanya akan mengizinkan arus listrik mengalir ke satu arah dan tidak ke arah sebaliknya. Chip LED pada umumnya mempunyai tegangan rusak yang relatif rendah. Bila diberikan tegangan beberapa volt ke arah terbalik, biasanya sifat isolator searah LED akan rusak menyebabkan arus dapat mengalir ke arah sebaliknya. Karakteristik chip LED pada umumnya adalah sama dengan karakteristik dioda yang hanya memerlukan tegangan tertentu untuk dapat beroperasi.



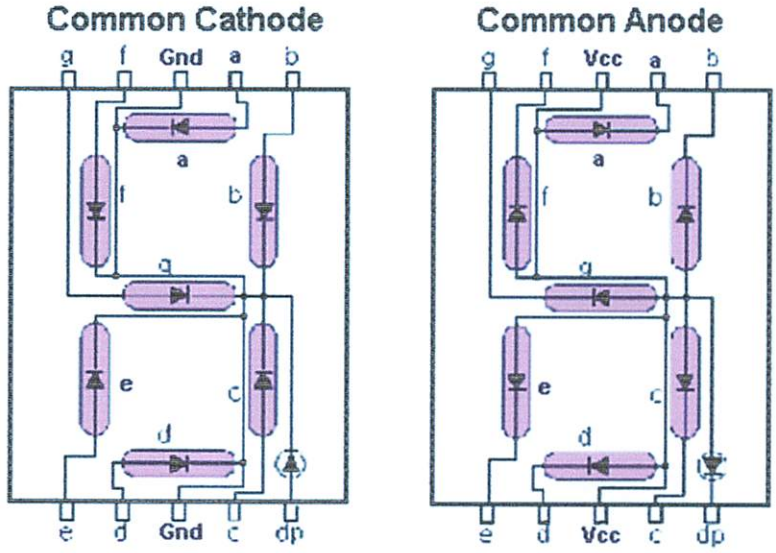
(Sumber : www.wikipedia.org/en/LED)

Gambar 2.6
Bagian – bagaian LED

Gambar 2.6 menjelaskan tentang bagian mendetail LED. LED terdiri dari kutub anoda dan katoda yang terbungkus penutup plastik. Anoda dan katoda terpasang kaki sebagai sarana menyolder komponen LED itu sendiri. Bagian anoda dan katoda dijelaskan sesuai gambar, yakni katoda memiliki bentuk pipihan pada sisinya serta ditandai pada kutub kaki yang lebih luas penampangnya. Hal ini sebgai referensi saat melakukan pemasangan komponen pada papan rangkaian.

2.2.3 *Seven Segment's Display*

Seven segment display adalah sebuah rangkaian yang dapat menampilkan angka-angka desimal maupun heksadesimal. Seven segment display biasa tersusun atas 7 bagian yang setiap bagiannya merupakan LED (*Light Emitting Diode*) yang dapat menyala. Jika 7 bagian diode ini dinyalakan dengan aturan yang sedemikian rupa, maka ketujuh bagian tersebut dapat menampilkan sebuah data heksadesimal(Wikipedia, 2012).



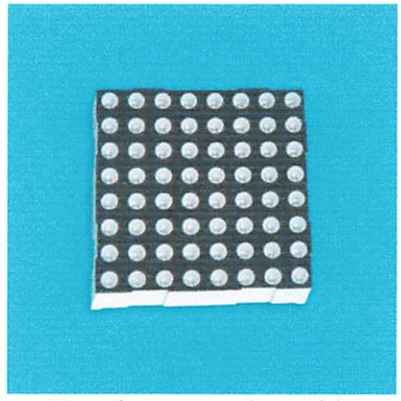
(Sumber : google.co.id)

Gambar 2.7
Skema rangkaian seven segment common katoda dan common anoda

Seven-segment display membutuhkan 7 sinyal input untuk mengendalikan setiap diode di dalamnya. Setiap diode dapat membutuhkan input *High* atau *Low* untuk mengaktifkannya, tergantung dari jenis seven-segmen display tersebut. Jika Seven-segment bertipe *common-cathode*, maka dibutuhkan sinyal *High* untuk mengaktifkan setiap diodenya. Sebaliknya, untuk yang bertipe *common-annode*, dibutuhkan input *Low* untuk mengaktifkan setiap diodenya.

2.2.4 LED *Dot Matriks*

Display matriks LED pada dasarnya adalah susunan beberapa LED yang disusun membentuk matriks baris dengan baris dan kolom yang bervariasi sesuai dengan tipenya. *Display* matriks LED ini dapat dikendalikan dengan teknik *multiplexing*. Dimana ada kontrol terpisah antara kendali jalur kolom dan baris dari display matriks LED tersebut(Heri, 2008:61).



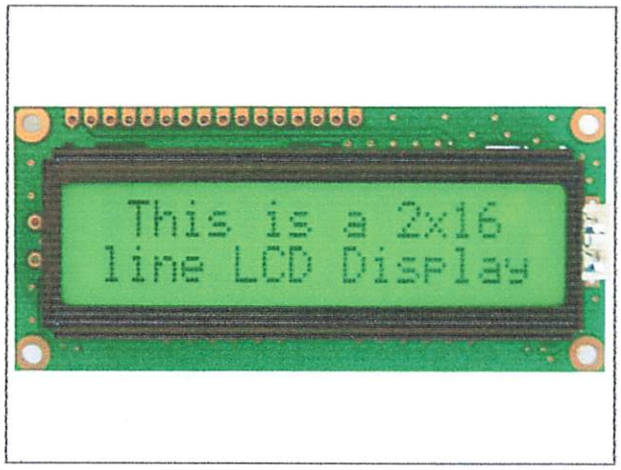
(Sumber : google.co.id)

Gambar 2.8
LED *Dot Matriks* dimensi 8x8

Untuk menyalakan setiap LED pada baris dan kolom dilakukan dengan memberikan tegangan pada anoda dan menghungkan katoda ke ground. Nyala LED pada masing – masing baris dan kolom inilah yang nantinya menghasilkan kombinasi pembentukan karakter tertentu.

2.2.5 *Alphanumeric* LCD

LCD *display module* terdiri dari dua bagian, yang pertama yaitu panel LCD sebagai media penampil informasi dalam bentuk huruf/angka yang terdiri dari baris dan kolom tergantung spesifikasi dimensi yang diberikan. Hal ini mempengaruhi jumlah karakter yang dapat ditampilkan LCD. Bagian kedua merupakan sebuah sistem elektronik yang dibentuk dengan mikrokontroler yang ada dibalik panel LCD, berfungsi mengatur tampilan informasi serta berfungsi mengatur komunikasi dengan port mikrokontroler(master) yang memakai LCD itu(Surya, 20011:14). Dengan demikian pemakain LCD cukup dengan mengirimkan sinyal kode – kode ASCII dari informasi yang ditampilkan seperti layaknya memakai sebuah printer.



(Sumber : google.co.id)

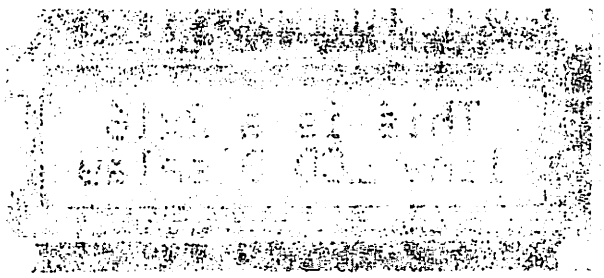
Gambar 2.9
LCD *module display* dimensi 2x16

Untuk LCD dengan dimensi 2x16 memiliki 8 jalur(DB0...DB7) data sebagai jalur komunikasi dengan mikrokontroler. Semua jalur ini nantinya dipakai sebagai media penulisan dan pembacaan untuk kode ASCII yang dikirimkan mikrokontroler serta penerimaan kode – kode itu pada LCD untuk diolah menjadi tampilan pada *display* LCD maupun sebagai pengaturan kerja dari LCD tersebut. Selain itu dilengkapi pula dengan E, R/W, dan RS seperti layaknya komponen yang kompatibel dengan mikroprosesor. Kombinasi ini sesuai dengan arsitektur yang diciptakan pabrikan semisal motorola dan intel. Selain itu kode – kode tersebut juga merepresentasikan sinyal sebagai pengatur proses kerja LCD.

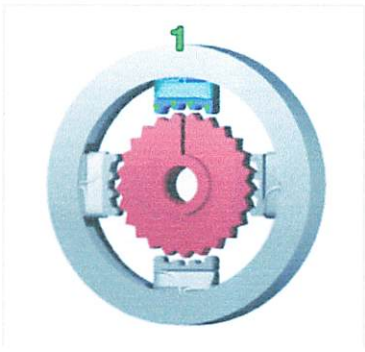
2.2.6 *Stepper Motor*

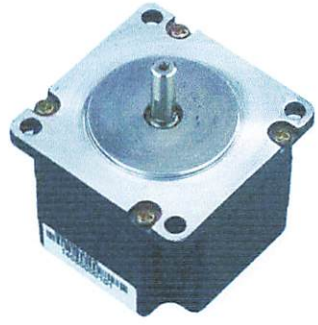
Motor stepper adalah suatu alat penggerak yang memanfaatkan gaya tarik magnet. Rotornya berhenti pada posisi kutub yang dieksitasi oleh arus yang mengalir pada lilitan. Rotor pada motor biasanya berputar secara kontinyu jika motor dieksitasi, tetapi rotor pada motor stepper berubah dari posisi diam dengan mengubah eksitasi kutub. Arus yang mengalir pada setiap lilitan hanya sesaat sehingga bentuk arusnya berupa pulsa. Rotor berputar karena pulsa yang bergantian. Kecepatan putaran rotor ditentukan oleh kecepatan perpindahan pulsa dan sudut putaran sebanding dengan banyaknya pulsa yang diberikan(Penulis, URL : repository.usu.co.id).

Rotor yang digunakan terbuat dari baja lunak dan memiliki sejumlah gigi yang jumlahnya kurang dari jumlah kutub pada stator. Stator memiliki beberapa pasang kutub dimana setiap pasang kutub diaktifkan melalui elektromagnetik oleh arus yang mengalir melalui kumparan yang dililitkan pada masing–masing kutub. Pada saat sepasang kutub



dalam keadaan aktif sehingga akan timbul medan magnet yang kemudian menarik pasangan gigi terdekat, sehingga gigi akan bergerak ke segaris dengan kutub.



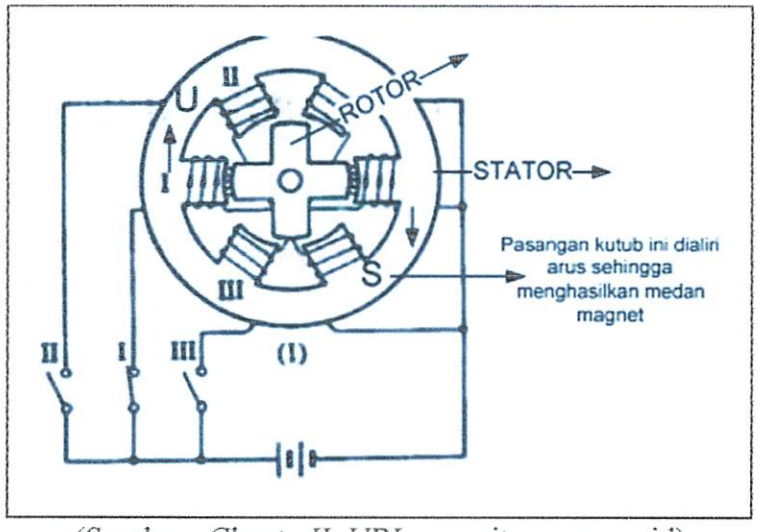


s

(Sumber wikipedia) (Sumber : google.co.id)

(a) (b)

Gambar 2.10
(a.) animasi desain gigi *stepper motor* (b.) *stepper motor*



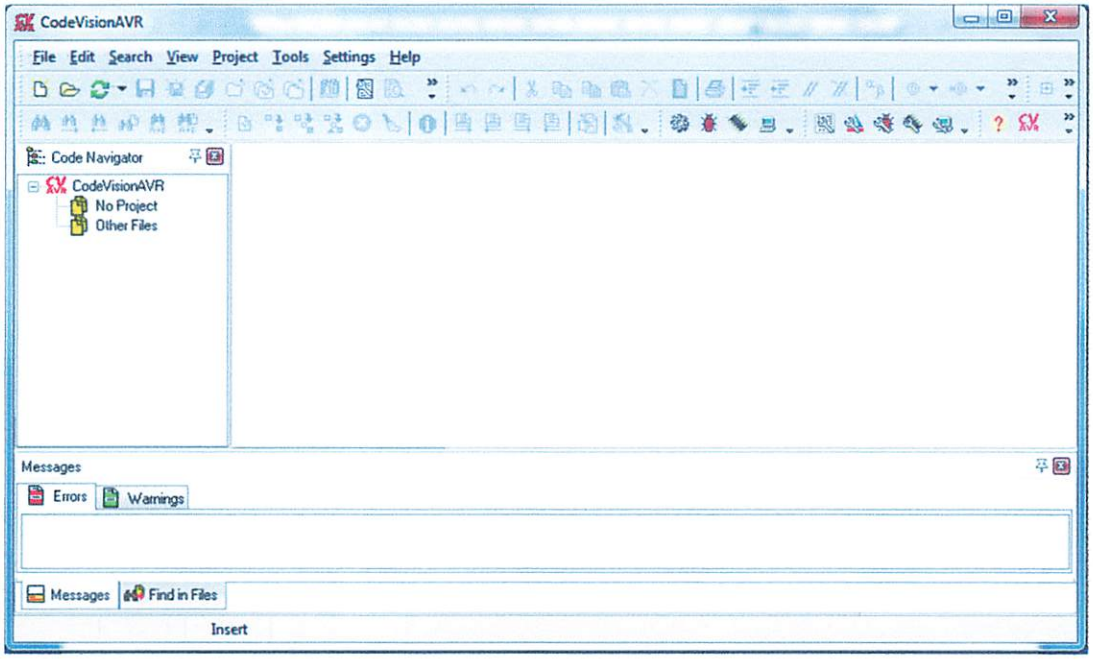
(Sumber : ChapterII, URL:repository.usu.co.id)

Gambar 2.11
Diagram langkah *stepper motor*

Untuk menggerakkan sebuah motor stepper setiap pasang kumparan stator harus disambungkan dengan aliran listrik dan diputuskan secara bergantian dalam urutan yang benar. Dengan demikian, input ke motor berupa deretan pulsa yang menghasilkan output ke setiap pasang kumparan stator. Sistem penggerak yang biasa digunakan terdiri dari dua blok utama yaitu pengatur urutan logika dimana menerima pulsa–pulsa input dan menghasilkan pulsa–pulsa output dalam urutan sebagai mana yang dibutuhkan untuk mengontrol penggerak agar menghasilkan pulsa output dengan amplitudo yang sesuai. (Penulis, URL: repository.usu.co.id).

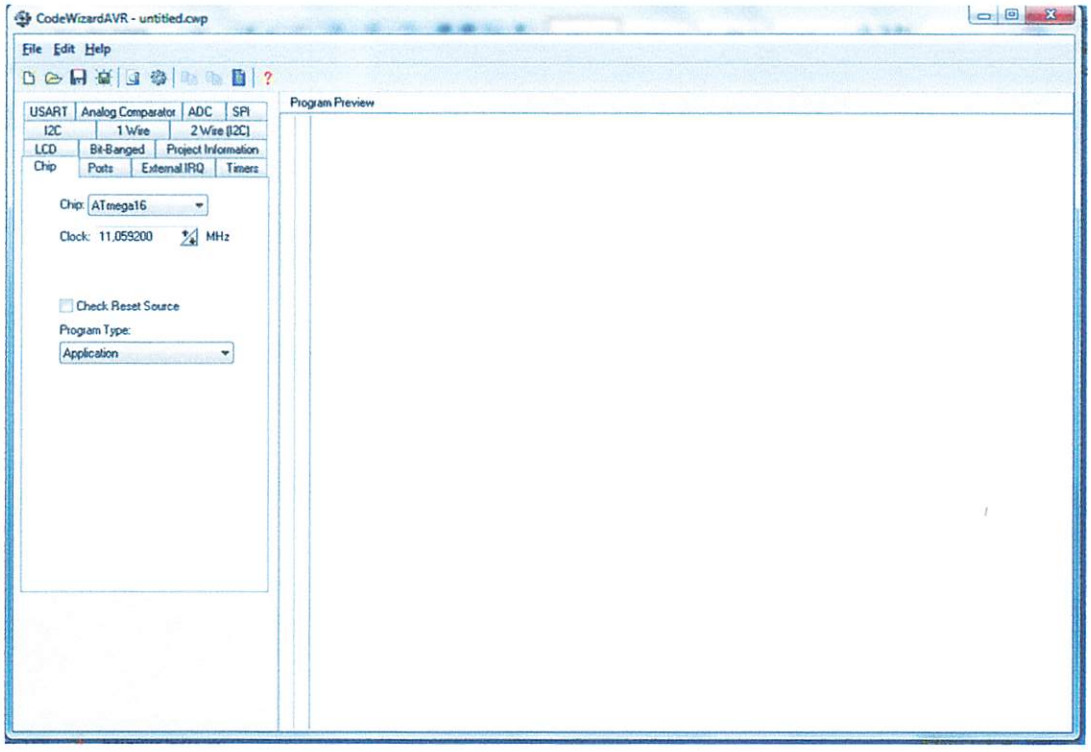
2.3 CodeVision AVR

CodeVision AVR pada dasarnya amerupakan perangkat lunak pemrograman mikrokontroler keluarga AVR berbasis bahasa C. Ada tiga komponen penting yang telah diintegrasikan dalam perangkat lunak ini yaitu *Compiler C*, IDE dan *program generator*(Surya, 2011). Berdasarkan spesifikasi yang dikeluarkan oleh perusahaan pengembangnya, *Compiler* C yang digunakan hampir mengimplementasikan semua komponen standar yang ada pada bahas C standar ANSI(seperti arsitektur program, jenis tipe data, jenis operator, dan library fungsi – fungsi standar berikut penamaannya). Walaupun demikian, dibandingkan bahasa C untuk aplikasi komputer, *compiler* C untuk mikrokontroler ini memiliki perbedaan yang disesuaikan dengan arsitektur mikrokontroler AVR tempat program akan ditanamkan(*embedded*). Khusus untuk fungsi library, disamping fungsi library standar(seperti fungsi – fungsi matematik, manipulasi string, pengaksesan memori, dan sebagainya), CodeVision AVR juga menyediakan fungsi – gungsi tambahan yang sangat bermanfaat dalam pemrograman antarmuka mikrokontroler AVR dengan perangkat lunak yang umum digunakan dalam aplikasi kontrol. Beberapa fungsi library yang paling penting diantaranya adalah fungsi – fungsi untuk pengaksesan LCD, komunikasi I^2C, IC RTC(*Real Time Clock*), sensor suhu LM75, SPI(*Serial Peripheral Interface*) dan lain sebagainya. Selain menu – menu pilihan yang umum dijumpai pada setiap perangkat lunak windows, CodeVisionAVR ini telah mengintegrasikan perangkat lunak *downloader* (*in system programing*) yang dapat digunakan dalam mentarnsfer kode mesin hasil kompilasi kedalam sistem memori mikrokontroler AVR yang sedang diprogram.



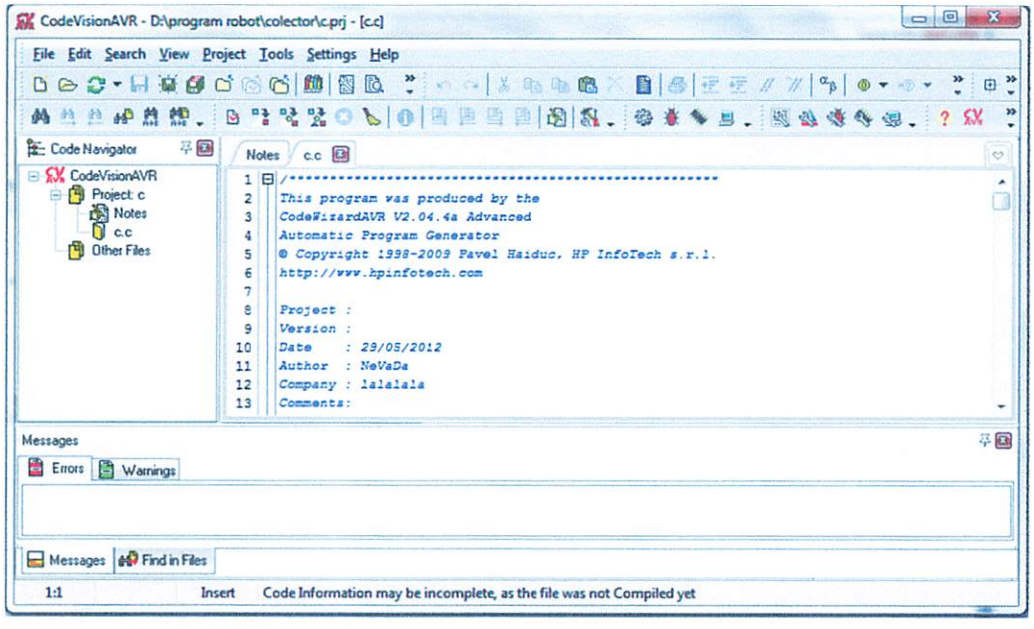
Gambar 2.12
Tampilan CodeVision AVR

Selain itu, CodeVision AVR juga menyediakan sebuah *tool* yang dinamakan dengan Code Generator atau CodeWizard AVR(lihat gambar 2.13). Secara praktis *tool* ini sangat bermanfaat dalam membuat kerangka program (*template*) serta member kemudahan bagi programer dalam penginilisasian register – register yang terdapat pada mikrokontroler AVR yang akan diprogram. Dinamakan Code Generator karena *tool* ini akan membangkitkan kode – kode program secara otomatis setelah fase inisialisai pada jendela CodeVision AVR selesai dilakukan. Gambar 2.14 memperlihatkan beberapapenggal baris kode program yang dibangkitkan secara otomatis oleh CodeWizard AVR.

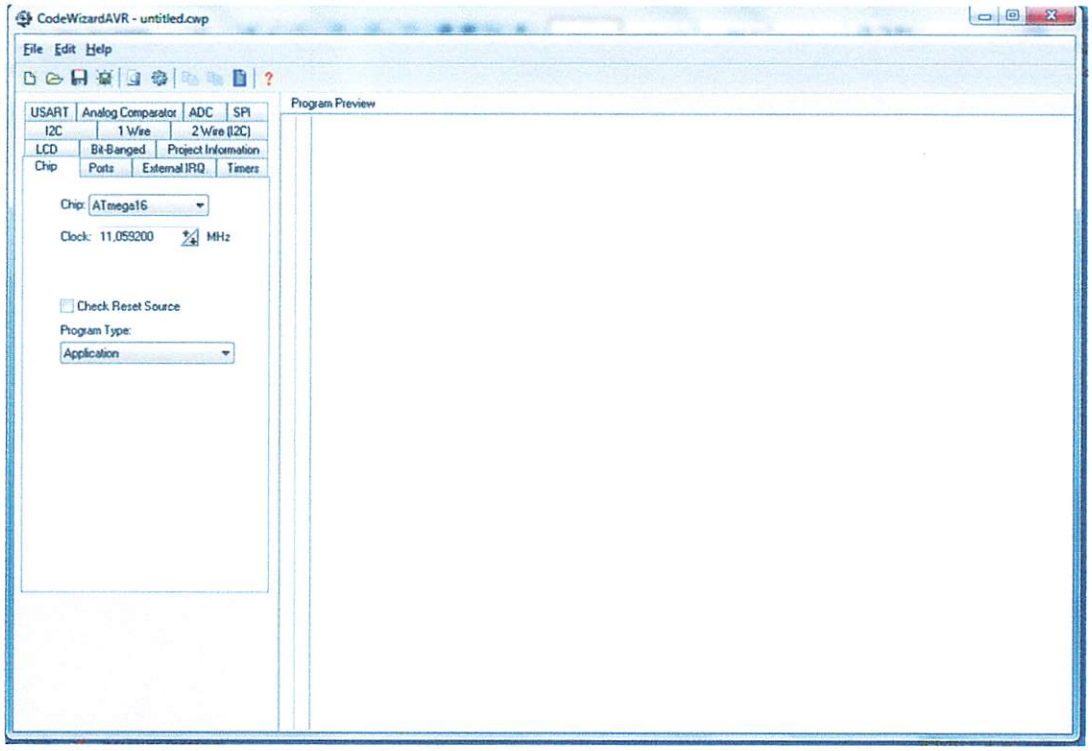


Gambar 2.13
Tampilan *tool* CodeWizard AVR

CodeVision AVR versi evaluasi dapat diunduh pada alamat www.hpinfotech.ro. CodeVision AVR versi evaluasi ini dapat kita pakai gratis, dengan kapasitas pemrograman maksimum 2 *kilobytes*. Versi standar yang komersil dapat menggunakan seluruh kapasitas memori mikrokontroler yang ada.

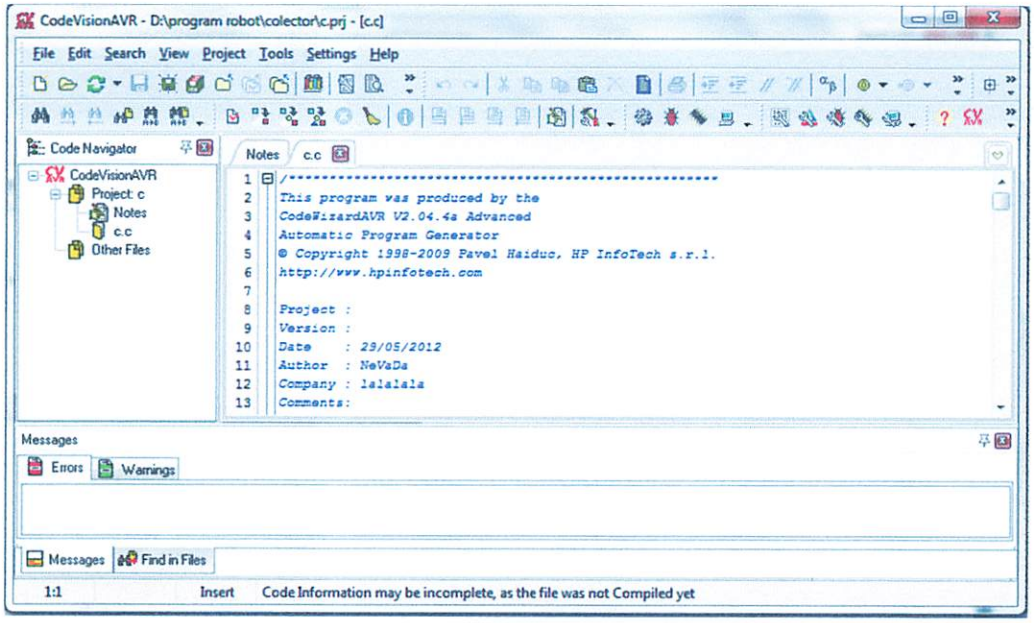


Gambar 2.14
Kode – kode yang dihasilkan CodeWizard AVR



Gambar 2.13
Tampilan *tool* CodeWizard AVR

CodeVision AVR versi evaluasi dapat diunduh pada alamat www.hpinfotech.ro. CodeVision AVR versi evaluasi ini dapat kita pakai gratis, dengan kapasitas pemrograman maksimum 2 *kilobytes*. Versi standar yang komersil dapat menggunakan seluruh kapasitas memori mikrokontroler yang ada.



Gambar 2.14
Kode – kode yang dihasilkan CodeWizard AVR

BAB III
PERANCANGAN DAN ANALISA SISTEM

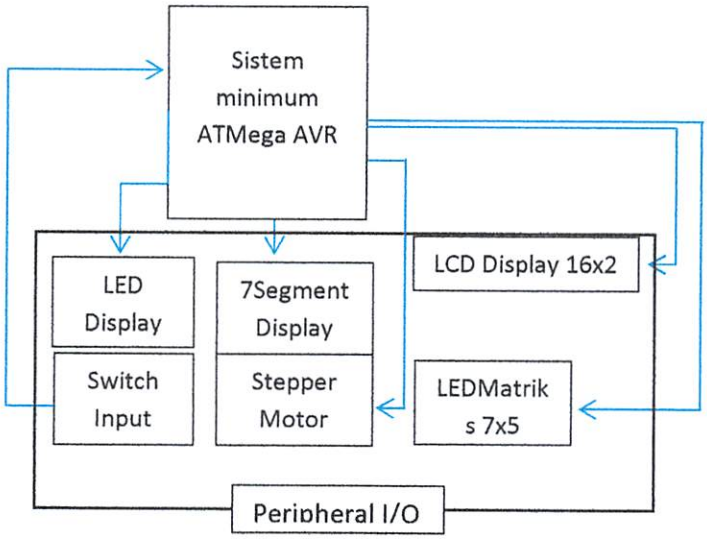
Bab ini menjelaskan mengenai analisa sistem dan perancangan alat. Analisa ditunjukan untuk memberikan analisa secara umum tentang perancangan alat dan memberikan solusi terhadap permasalahn yang dihadapi. Dari analisa ini diharapkan dapat menghasilkan suatu ide perancangan yang sesuai sebagai solusi dari analisa masalah yang dihadapi.

3.1 Identifikasi masalah

Pada skripsi ini masalah yang dihadapi ialah bagaimana memberikan alternatif baru dalam dalam pembelajaran perkembangan mikrokontroler pada zaman sekarang. Sebagaimana dapat diketahui bahwa pada modul praktikum lama masih menggunakan mikrokontroler dengan spesifikasi MCS-51 yang dimana pada perkembangannya telah tertinggal oleh teknologi Atmel Mega AVR. Dari permasalahn maka dihasilkanlah sebuah ide penulis untuk merancang modul praktikum mikrokontroler ATMega yang berbasis sistem minimum AVR.

3.2 Gambaran umum desain alat

Modul praktikum mikrokontroler berbasis sistem minimum ATMega AVR adalah suatu alat yang berupa minimum sistem dengan spesifikasi inti mikrokontroler dari keluaran Atmel Mega AVR dan peripheral I/O sebagai indikator simulasi praktis terhadap fitur – fitur yang ada pada mikrokontroler tersebut. Nantinya sejumlah program akan ditanamkan pada chip inti mikrokontroller yang kemudian akan mengolah data sesuai dengan hubungan antarmuka antara masing – masing peripheral I/O dengan jalur I/O mikrokontroler.



Gambar 3.1
Diagram Blok Modul praktikum Mikrokontroler

3.2.1 Fungsi masing – masing rancangan alat

Dari diagram tersebut diatas dapat diterangkan bahwa desain modul praktikum mikrokontroler yang akan dibuat memiliki bagian – bagian yang berfungsi sebagai berikut :

1. Minimum sistem yang berbasis ATMEGA AVR, dimana jenis ATMEGA AVR yang digunakan adalah jenis ATMEGA8535/16/32 dengan jenis paket DIP(*Dual In-line Package*) 40 pin dan ATMEGA128L dengan jenis paket TQFP(*Thin Quad Flat Package*) 64 pin. Minimum sistem ini berfungsi sebagai suatu sistem inti otak dari segala proses pengolahan data dan juga penyedia fungsi pin – pin I/O.
2. Peripheral I/O sebagai pemeberi kontrol input maupun indikator dari proses output pengolahan data yang dilakukan mikrokontroler melalui koneksi antarmuka dengan pin – pinnya. Adapun bagian – bagian dari peripheral I/O tersebut memiliki masing – masing fungsi :
   2.1 LED : indikasi tampilan dalam bentuk output bit
   2.2 Switch Output : kontrol inputan data ke Mikrokontroler dalam bit
   2.3 7Segment Display : indikasi kombinasi output yang membentuk decimal

2.4 Matriks LED 7x5 : indikasi Output dalam tampilan LED yang membentuk karakter dengan dimensi 7 baris dan 5 kolom

2.5 LCD Display 16 x 2: indikasi output yang lebih komplek dengan membentuk kombinasi karakter dalam dimensi 2 baris dan 16 kolom

2.6 Motor Stepper : indikasi output dalam bentuk aktuator

3.3 Prinsip kerja alat

Prinsip kerja alat modul praktikum mikrokontroler ini secara umum adalah memberikan masing-masing kode program pada sistem kontrol minimum sistem AVR sesuai dengan antarmuka peripheral I/O board yang terpasang melalui koneksi antara pin mikrokontroler dengan pin I/O tersebut. Dalam hal ini peripheral I/O ada yang berfungsi sebagai *input* dan ada pula yang berfungsi sebagai *output.*

Dalam perancangan skripsi ini penulis membuat suatu paket percobaan yang terdiri dari simulasi masing–masing bagian peripheral I/O. Adapun beberapa simulasi tersebut adalah :

1. Percobaan LED display
2. Percobaan input Switch
3. Percobaan seven segment
4. Percobaan Matriks LED
5. Percobaan LCD
6. Percobaan Stepper Motor

Percobaan pertama adalah LED display. Percobaan ini mensimulasikan bagaimana menampilkan outputan 8 bit LED yang terhubung pada pin mikrokontroler. Dengan 8 bit LED yang ada kita dapat membuat berbagai variasi hasil outputan yang diinginkan sesuai alur program yang ditanamkan.

Percobaan kedua adalah input switch. Percobaan ini mensimulasikan bagaimana memberi sebuah kontrol input melalui antarmuka saklar data pada pin mikrokontroler yang nantinya mengindikasikan hasil output berupa nyala LED dari instruksi proses pengolahan data yang dirancang.

Percobaan ketiga adalah seven segment's display. Percobaan ini mensimulasikan bagaimana menghasilkan outputan berupa kobinasi desimal pada seven segment. Kombinasi desimal ketujuh segmen pada seven segment's dapat membentuk kombinasi angka dan beberapa huruf. Ini dapat mengkreasikan tampilan apa yang ingin kita hasilkan.

Percobaan keempat adalah percobaan LCD display. LCD display memiliki tampilan yang lebih kompleks. LCD mengkombinasikan karakter sehingga nantinya dapat membentuk suatu baris karakter serta variasi tampilan yang diinginkan sebagai display media.

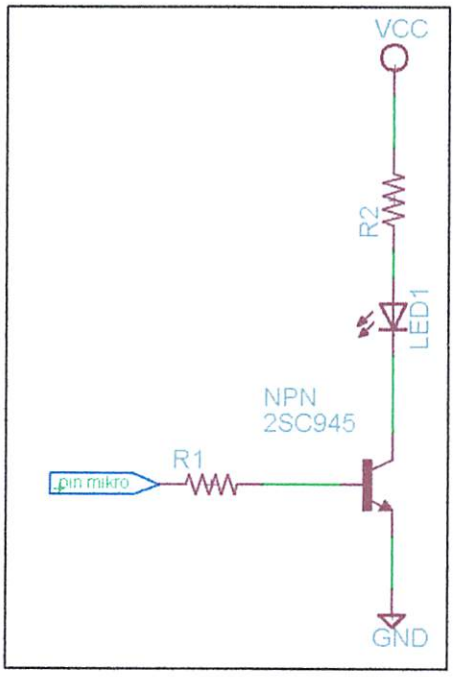
Percobaan kelima adalah percobaan Stepper Motor. Stepper Motor merupakan suatu aktuator yang bergerak dengan keadaan step per step. Percobaan ini menjelaskan bagaimana dasar kontrol step motor stepper yang nantinya dikombinasikan untuk membentuk suatu putaran.

Percobaan keenam adalah Matriks LED display 7x5. Percobaan ini mensimulasikan bagaimana dengan 35 titik yang ada dapat menghasilkan suatu kombinasi karakter yang diinginkan.

3.4 Perancangan perangkat keras

3.4.1 Perancangan rangkaian output LED

LED merupakan diode yang mampu memancarkan cahaya. Rangkaian LED yang digunakan pada peripheral I/O adalah sebagai berikut :



Gambar 3.2
Skema Rangkaian LED

Dari skema rangkaian pada bagian LED, arus forward(If) yang diinginkan +/- 10 mA, dan Vcc yang dapat digunakan I/O board mencapai 5Volt DC. Untuk memperoleh nilai R2 maka digunakan rumus

$$Vcc = If \; x \; R2$$

$$R2 = \frac{Vcc}{If}$$

$$R2 = \frac{5}{10 \; x \; 10^{-3}}$$

$$R2 = 500\Omega = 470\Omega \; (nilai \; yang \; mendekati) \; \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (3.4.1 - 1)$$

Keterangan:

$$Vcc = Tegangan \; Suply$$

$$R = Hambatan$$

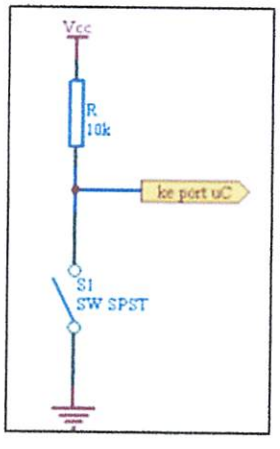
$$If = arus \; forward \; LED$$

Prinsip kerja rangkaian diatas adalah pada saat transistor NPN diberi arus basis, maka arus dari kolektor akan mengalir ke emitor. Hal ini dikarenakan transitor berfungsi sebagai saklar elektrik. Karna perbedaan potensial antara kolektor dan emitor maka arus akan mengalir dan menyalakan LED.

3.4.2 Perancangan rangkaian input saklar

Untuk memberi masukan kedalam pin mikrokontroler dengan cara termudah adalah dengan menggunakan saklar. Prinsip kerja yang sederhana untuk membaca saklar ini nantinya dapat dikembangkan untuk membaca piranti masukan lain seperti sensor.

Saklar merupakan piranti masukan mekanis yang berfungsi untuk memberikan inputan data ke mikrokontroler. Dalam peripheral I/O board ini antarmuka menggunakan saklar geser.



Gambar 3.3
Skema rangkaian saklar geser

Metode perancangan dengan skema diatas memungkin kan kita untuk memberi inputan pada pin mikrokontroler dengan pull-up eksternal. Sehingga apabila kita menghupungkan pada port 0, pembacaan logika dengan skema rangkaian ini yaitu apabila saklar terbuka akan berlogika 1 dan saat tertutup akan berlogika 0.

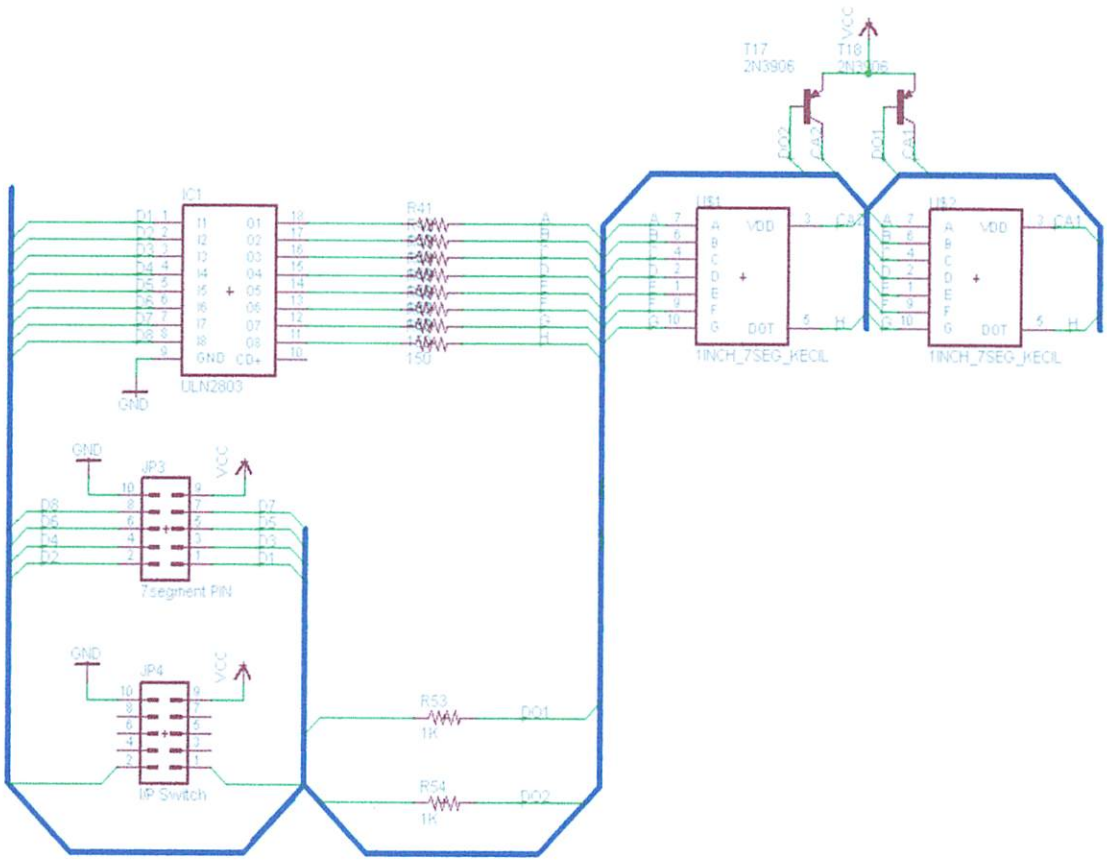
Menganalisa hasil inputan saklar pada mikrokonroler sendiri dibutuhkan ketelitian karena sifat saklar itu sendiri pada kondisi transisi dapat menimbulkan efek *debounce*. Hal ini seperti *noise* yang dapat mengganggu pembacaan data oleh mikrokontroler. Oleh karena itu biasanya pada saklar yang rawan terhadap kondisi debounce ini sendiri kita perlu menyiasati pembacaan data dengan memberikan *delay* saat inputan terjadi akibat kondisi transisi dari saklar itu sendiri atau dapat pula menggunakan prinsip gerbang logika yang nantinya akan menghasilkan suatu inputan

data stabil dari hasil transisi saklar sehingga pembacaan data itu sendiri tetap akurat sebagai inputan.

3.4.3 Perancangan rangkaian seven segment display

Tampilan yang lebih menarik dari LED adalah seven segment. Seven segment, seperti namanya, merupakan sebuah piranti luaran yang tersusun dari 7 buah LED berbentuk garis. Tujuh buah LED tersebut diberi symbol A – G (pada umumnya) atau A-H dengan tambahan tampilan LED dot(titik). Ketujuh buah LED tersebut dijadikan satu pada salah satu kutubnya. Pada percobaan ini menggunakan penggabungan pada kutub anoda (*common anoda*), dimana kutubnya disambungkan dengan VCC.

Untuk skema rangkaian seven segment ditunjukan pada gambar dibawah ini

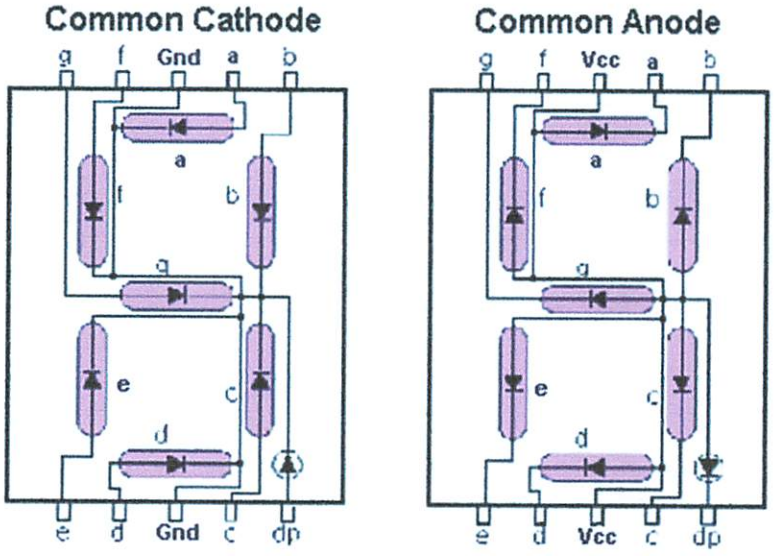


Gambar 3.4
Skema rangkaian seven segment

Pada rangkaian ini pin1–pin8 pada JP3 merupakan pin yang dihubungkan ke pin mikrokontroler sebagai kontrol inputan data yang diberikan ke seven segment itu sendiri. Penggunaan transistor berfungsi sebagai saklar yang nantinya menghidupkan atau mematikan keadaan tampilan seven segment itu sendiri, dimana kaki transistor

secara langsung mengontrol inputan tegangan pada seven segment. Sementara transistor di kontrol melalui antarmuka pin1 dan pin2 pada JP4 mikrokontroler.

Pin1-pin8 akan menghasilkan data 8 bit. Untuk menganalisa kontrol kombinasi outputan pada kedelapan data tersebut maka dapat membacanya secara perbit maupun data heksadesimal. Adapun urutan pin1 hingga pin8 masing – masing mewakili data terhadap kombinasi titik pada *seven segment's display*.



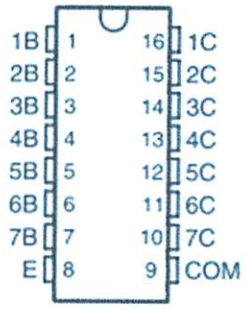
Gambar 3.5
Kombinasi titk pada seven segment's baik *common* anoda maupun *common* katoda

Dari gambar 3.5 maka dapat diberikan suatu urutan persamaan data dimana masing – masing pin1-pin8 mewakili titik a-dp(dot) pada seven segment display. Sedangkan kontrol seven segment mewakili daerah comon pada display seven segment display tersebut.

3.4.4 Perancangan rangkaian dot matriks display

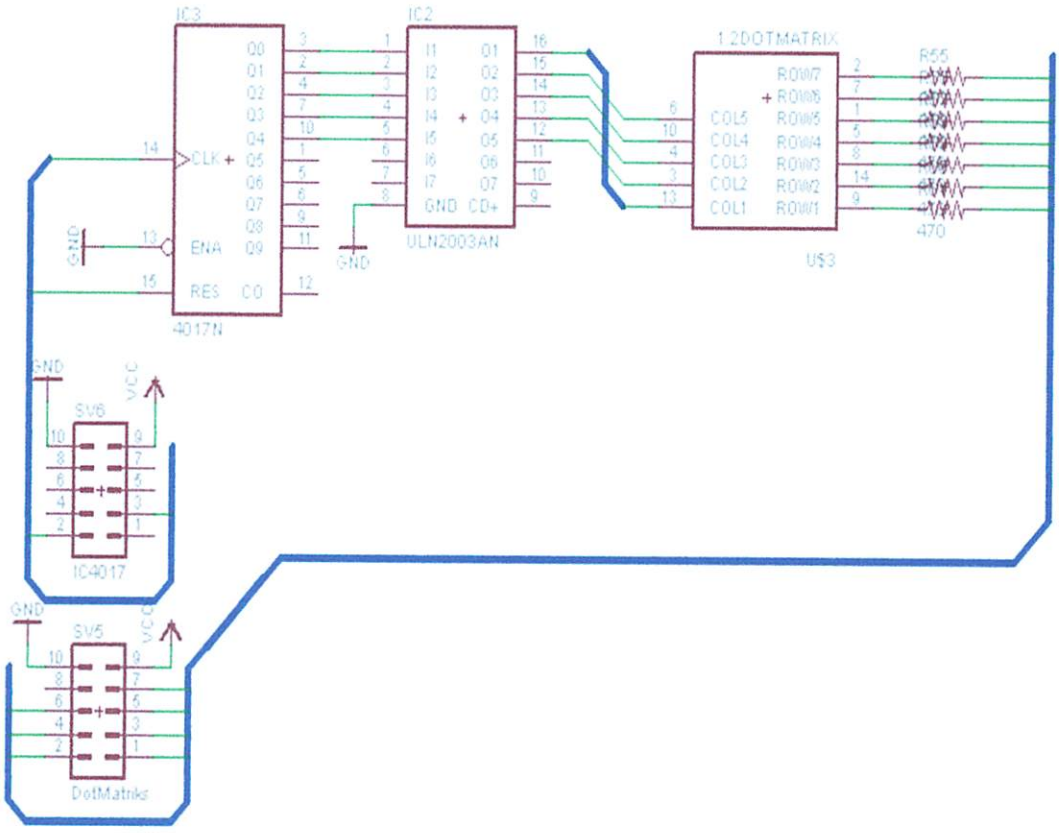
Dot matriks display merupakan susunan LED yang membentuk dimensi beberapa kolom x beberapa baris. Dengan susunan seperti ini dot matriks dapat menghasilkan tampilan karakter dari kombinasi nyala LED di setiap dot(titik)nya.

Dalam perancangan ini, digunakan IC ULN 2003A sebagai array driver dan IC 4017 sebagai kontrol driver. Spesifikasi IC ULN 2003A sendiri merupakan IC darlington transistor array yang memiliki output *powerfull* sehingga dapat membantu mengurangi beban supply pada mikrokontroler untuk dor matriks sendiri.



Gambar 3.6
Tampak atas IC ULN2003A

Berikut rancangan skema rangkaian dot matriks display



Gambar 3.7
Skema rangkaian Dot Matriks Display 5 x 7

Rangkaian ini mendukung fitur *cascade*. Jadi apabila nantinya aka nada penambahan jumlah dotmatriks display maka tinggal menambah drivel dengan IC 4017 dan IC ULN 2003A sesuai jumlah pin yang dibutuhkan, sementara pin yang terkoneksi ke mikrokontroler tetap. Pada rangkaian diatas Pin1 – Pin7 pada SV5 dihubungkan langsung ke pin mikrokontroler. Sementara Pin2 – Pin3 pada SV5 merupakan pin kontrol terhadap driver IC 4017.

Teknik menyalakan dot matriks sendiri menggunakan teknik scanning dimana pemberian nilai pada suatu set kolom yang nantinya jika dikombinasikan kelimanya akan membentuk karakter tertentu. IC 4017 merupakan suatu IC shift register yang apabila diberikan suatu data dengan perioda cycle tertentu akan menghasilkan output menggeser. Scanning tadi dimaksudkan dengan mengirim suatu kombinasi menurut urutan tertentu melalui mikrokontroler kemudian data tersebut nantinya akan di outputkan satu per satu sesuai dimensi karakter yang dibuat berdasarkan dimensi dot matriks display kedalam masing – masing kolom. Secara kasat mata pengiriman data ini terlihat terjadi serempak padahal IC 4017 itu sendiri mengatur pengiriman satu per satu data kolom dot matriks secara periodik yang cepat (dalam kisaran ms) sesuai perioda dalam program mikrokontroler.

3.4.5 Perancangan rangkaian LCD display

Dalam perancangan ini menggunakan sebuah layar LCD dengan dimensi 16x2. Dimana hal ini berarti LCD terdiri dari tampilan 2 baris yang masing – masing barisnya berisikan 16 karakter. Berikut fungsi masing – masing pin sinyal LCD

Tabel 3.1
Fungsi masing – masing pin sinyal LCD

Nama signal	Fungsi
DB0-DB7	Merupakan saluran data, berisi perintah dan data yang akan ditampilkan di LCD
Enable	Sinyal operasi awal, sinyal ini mengaktifkan data tulis atau baca
R/W	Sinyal seleksi tulis atau baca 0 : tulis 1 : baca
RS	Sinyal pemilih register 0 : instruksi register (tulis) 1 : data register (baca dan tulis)

Secara garis besar pemrograman LCD diatur oleh 3 sinyal yaitu RS, R/W, Enable dan 8 buah saluran data. Karena pada tugas akhir ini LCD difungsikan sebagai komunikai 4 bit, jadi saluran data yang dipakai hanya D4 s/d D7. VR pada pin 3 digunakan untuk mengatur kontras LCD, sedangkan pin 15 dihubungkan dengan diode agar tegangan yang masuk sesuai dengan datasheet yaitu 4,3 V.

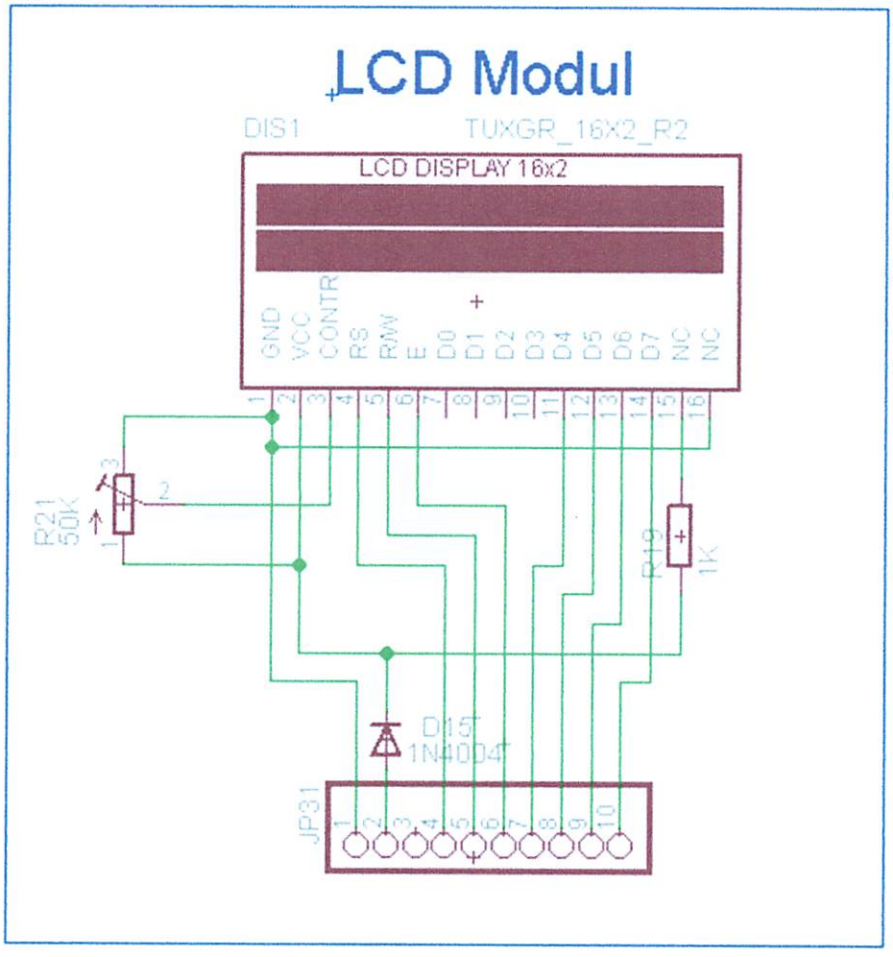
Tegangan jatuh diode = 0,7V

Vcc = 5V

Maka tegangan input = Vcc – tegangan jatuh diode

= (5 – 0,7) V = 4, 3 V . . . . . .(3.4.5-1)

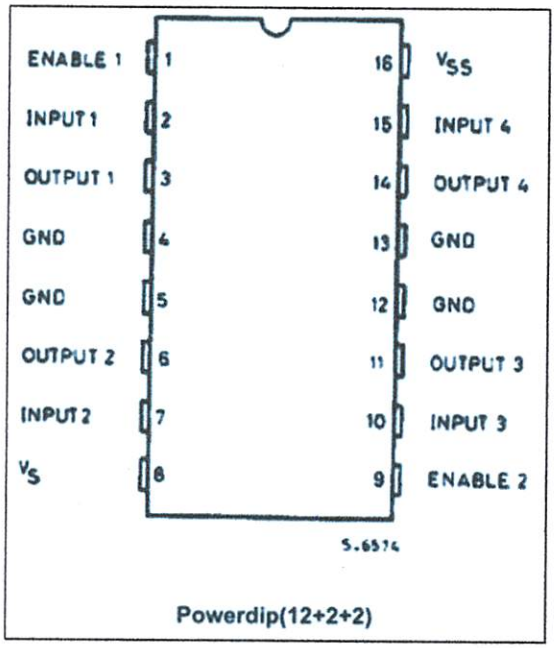
Berikut skema rangkaian antarmuka LCD



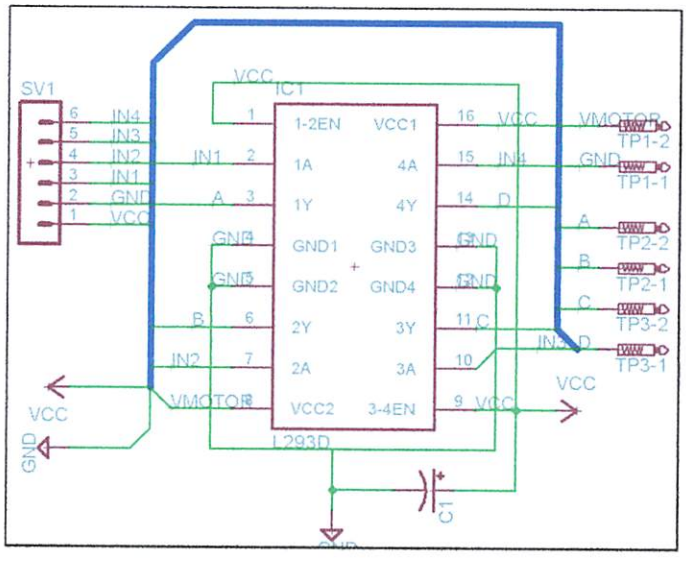
Gambar 3.8
Skema rangkaian LCD

3.4.6 Perancangan rangkaian driver motor stepper

Driver motor stepper dirancang menggunakan IC L293D. L293D merupakan driver motor H-Bridge yang terpaket dalam satu cip ic. Driver L293D ini memiliki tegangan kerja dari 6C sampai 36 , arus RMS maksimum 600mA, dan arus impuls tak berulang maksimum 1,2A. Selain itu L293D juga telah dilengkapi diode clamp internal.



Gambar 3.9
Tampak atas L293D tipe DIP



Gambar 3.10
Skema rangkaian driver motor stepper

Menggerakan motor stepper sungguh sangat berbeda dengan menggerakan motor DC. Prinsipnya motor stepper tak hanya membutuhkan tegangan saja, tetapi juga

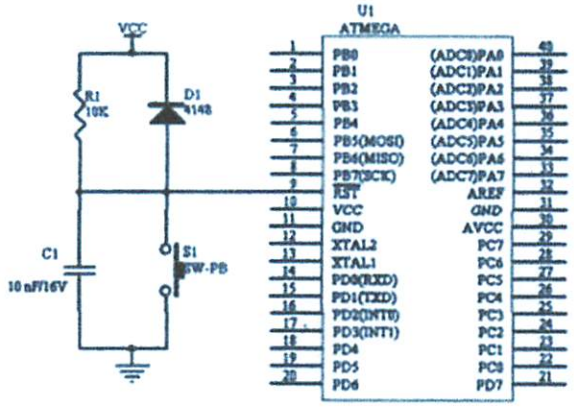
suatu urutan data terhadap masing – masing transistornya sebagai penentu arah putaran motor itu sendiri.

Jenis motor yang digunakan dapat berupa motor stepper jenis polar maupun bipolar. Pada skripsi ini digunakan motor jenis bipolar. Motor yang digunakan merupakan pabrikan Mitsumi.Pin konektornya terdiri dari 4 konektor yang dapat diprogram secara *half* step maupun *full* step. Untuk pola full step, Motor stepper akan berputar sebesar 360^0 dengan banyaknya step sebanyak 200 step atau 1 stepnya mewakili 1.8^0. Untuk pola Half step, Motor stepper akan berputar sebesar 360^0 dengan banyaknya step sebanyak 400 step atau 1 stepnya mewakili 0.9^0. Selain itu mode *full step* akan menghasilkan sebuah putaran dengan kecepatan tinggi dan torsi kecil, sebaliknya mode *half step* akan menghasilkan putaran dengan kecepatan rendah dan torsi yang besar.

3.4.7 Perancangan minimum sistem ATMega8535/16/32

3.4.7.1 Perancangan rangkaian reset dan clock

Perancangan rangkaian reset pada mikrokontroler ATMega 8535/16/32 ialah dengan memberikan logika low pada pin reset mikrokontoler ATMega8535/16/32. Rangkaian reset ini diproleh dari *application note AVR Design Consideration* dari ATMEL. Berikut ialah gambar rancangan rangkaian reset pada ATMega8535/16/32 :



Gambar 3.11
Rangkaian reset

Osilator pada rangtaian minimm sistem ATMega8535/16/32 menggunakan kristal 11,0592MHz dan kapasitor 22 pF. Nilai kapasitor ini diperoleh dari tabel datasheet tentang penggunaan kapasitor untuk rangkaian osilator / sistem clock pada ATMega8535/16/32. Sesuai Databook ATMega8535, 16, dan 32 dengan penggunaan kristal 11,059200MHz dan UBRR 17 maka didapat perhitungan sebagai berikut :

$$UBRR = \frac{fosc}{16xBaud} - 1$$

$$Baud = \frac{11059200}{16\,x\,(17+1)} - 1$$

$$Baud = \frac{11059200}{61440} - 1$$

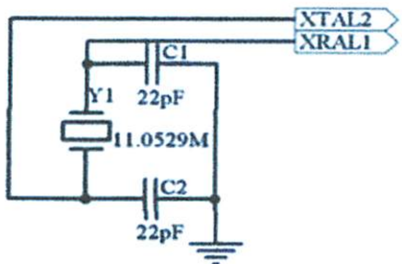
$$Baud = 38400 \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots \quad (3.4.7.1 - 1)$$

Keterngan :

$UBBR = nilai\ register\ UBRRH\ dan\ UBRRL(0 - 4095\ desimal)$

$fosc = frekuensi\ osilator$

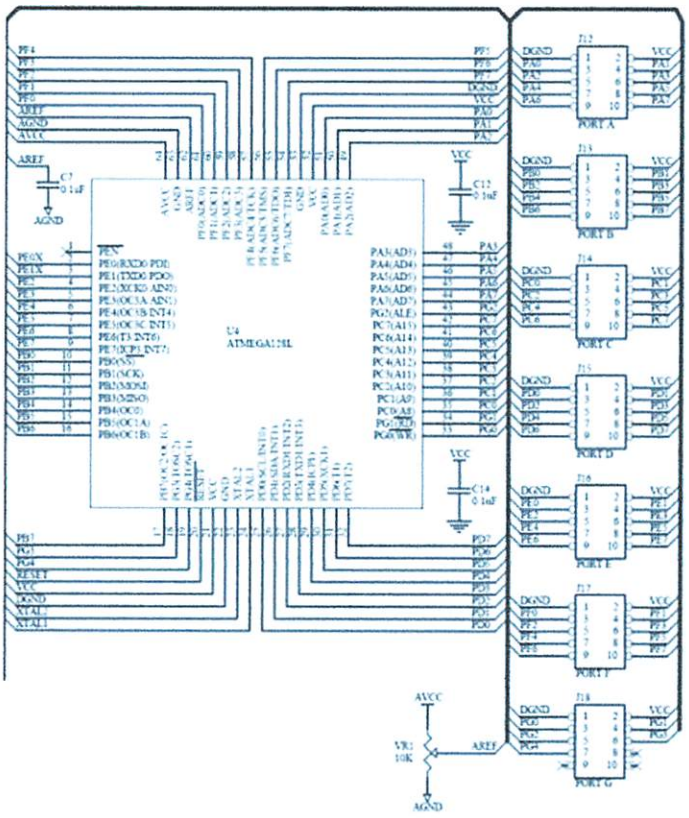
$Baud = Baud\ rate(bit\ per\ detik)$



Gambar 3.12
Rangkaian Clock

3.4.8 Modul DT-AVR ATMega128L bootloader micro system

DT-AVR ATMega128L *bootloader micro system* (BMS) merupakan sebuah modul single chip berbasis mikrokontroler ATmega128L. DT-AVR ATmega128L BMS dilengkapi dengan program bootloader sehingga tidak membutuhkan divais programmer. Dengan menggunakan bootloader pada DT-AVR ATmega128L BMS, pengguna dapat menggunakan jalur UART sebagai jalur komunikasi dengan komputer, sekaligus menggunakannya untuk melakukan *remote programming* jika ada perbaikan program (*update*). Software yang digunakan untuk memprogram mikrokontroler adalah AVR Bootloader.

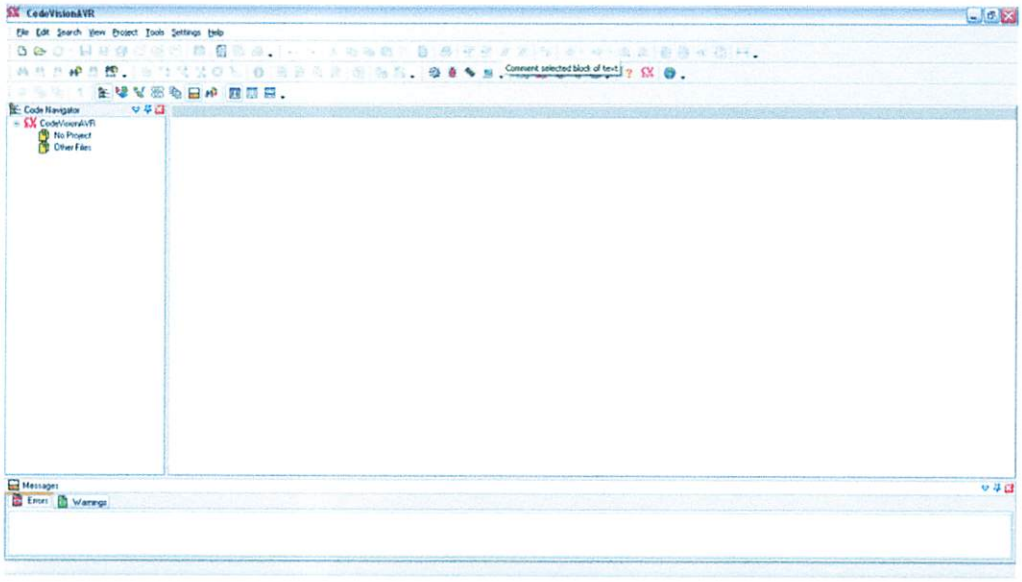


Gambar 3.13
Skema rangkaian modul minimum sistem ATMega128L

3.5 Perancangan software

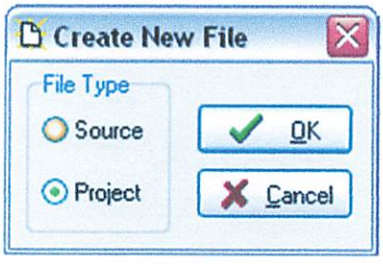
Software dalam penyusunan tugas akhir ini menggunakangunakan CodeVision AVR, dimana software ini hampir sama dengan IDE(*IntergratedDevelopment Environment*) lainya. CodeVision AVR dilengkapi dengan *source code editor, compiler, linker*, dan dapat memanggil Atmel AVR Studio untuk *debuger*nya. CodeVision AVR versi evaluasi dapat di-download pada www.hpinfotecth.ro. CodeVision AVR versi evaluasi dapat kita pakai secara gratis.

Untuk memulai menjalankan program setelah melakukan proses instalasi buka CodeVision AVR melalui **Start |All Program || CodeVision||| CodeVision AVR C compiler.**

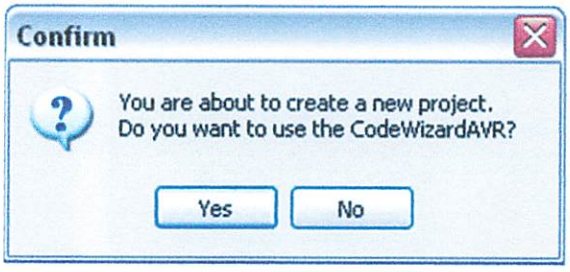


Gambar 3.14
Tampilan awal Code Vision AVR

Kemudian untuk memulai new project pilih file| new| pilih file type-> project

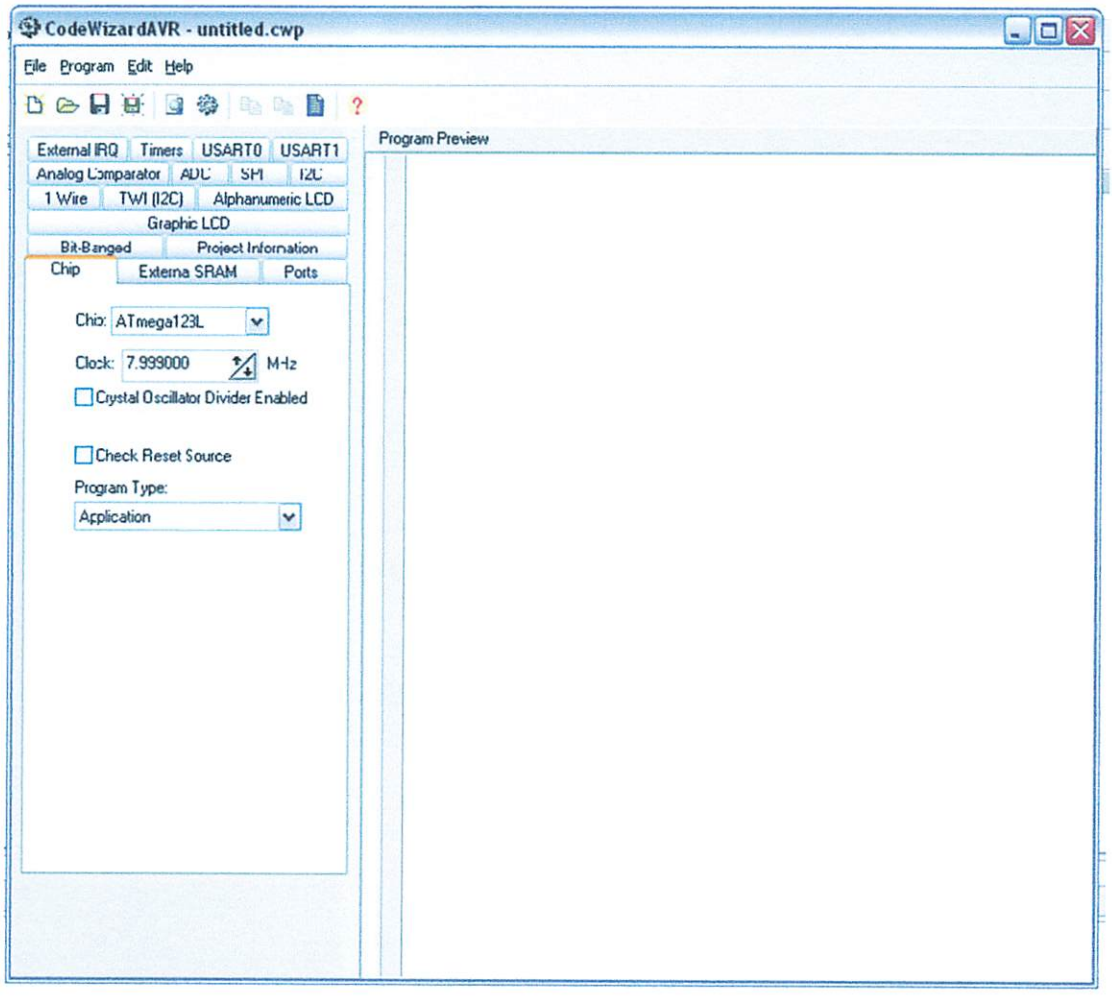


Gambar 3.15
Membuat project baru



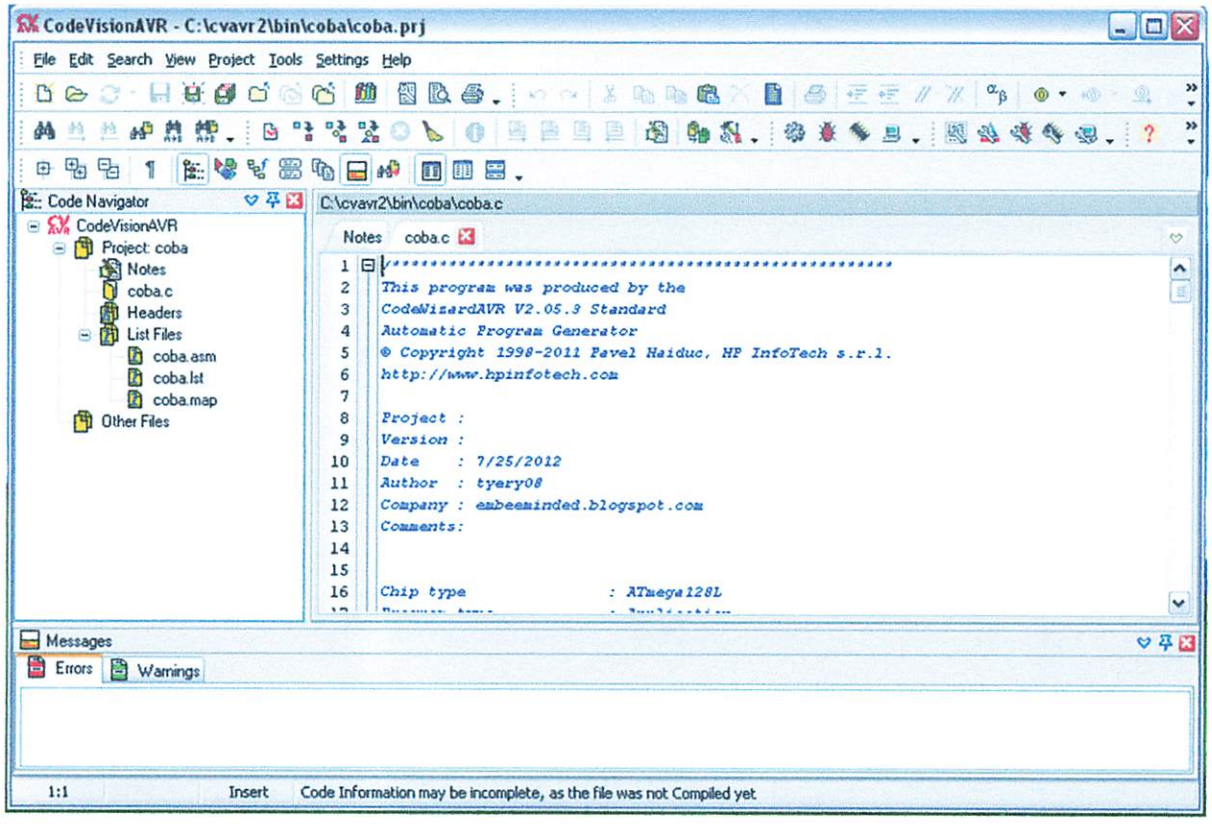
Gambar 3.16
Pertanyaan membuat project baru

Akan muncul tampilan seperti diatas dan akan menanyakan apakah akan mengunakan codevision avr wizard untuk membuat project baru, selanjutnya pilih saja yes. Kemudian akan tarnpil konfiguasi USART, Analog Komparator, ADC, I2C, LCD, Bit-banged, project informatison, chip, port, external RQ dan Timer. Kita tinggal mengatur program yang akan kita buat melalui codevision avr wizard ini. Misalnya untuk konfiurasi chip: ATmega32, clock: 11.059200 Mhz. Untuk pengaturan port sebagai input/output pilih port, dan seterusnya.



Gambar 3.17
Pengaturan konfigurasi chip

Jika kita sudah menkonfigurasj project pilih file| generate. Kemudian beri nama file source (*.c), file project (*.Prj) dan file project codewizard(*.cwp). Setelah diberi nama tekan save pada masing – masing pilihan jendela *windows*. Misalkan saya mencoba dengan memberi nama coba pada masing – masing pilihan, maka tampilannya sebagai berikut :



Gambar 3.18
Hasil konfigurasi dengan codewizard CodevisonAVR

BAB IV
PENGUJIAN ALAT DAN PEMBAHASAN HASIL

Pada bab ini membahas tentang pengujian alat yang dirancang, dimana meliputi perangkat keras dan perangkat lunak. Untuk mengetahui sistem yang dirancang sesuai terhadap fungsi yang diharapkan, dilakukan pengujian terhadap sistem tersebut. Berikut dijelaskan mengenai prosedur dan hasil pengujian.

4.1 Pengujian LED

4.1.1 Tujuan pengujian

Tujuan pengujian ini adalah untuk mengetahui logika menyalakan output LED melalui port mikrokontroler.

4.1.2 Peralatan yang digunakan

1. Peripheral I/O LED board
2. Minimum sistem AVR
3. Kabel konektor
4. DC Power supply
5. Laptop/PC sebagai media pemrograman

4.1.3 Langkah – langkah pengujian

1. Hidupkan minimum sistem board dengan memberi tegangan DC.
2. Koneksikan port LED pada peripheral I/O dengan kabel konektor ke pin mikrokontroler(PORTD)
3. Membuat program menyalakan LED melalui codevison AVR dan mendownload program tersebut ke mikrokontroler
4. Mencatat hasil pemrograman ke dalam tabel dan menganalisa hasilnya.

4.1.4 Hasil pengujian LED

Tabel 4.1
Hasil pengujian output LED

No	Data PORTD (Heksa)	Hasil output pada LED							
		Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
1	00 H	0	0	0	0	0	0	0	0
2	01 H	0	0	0	0	0	0	0	1
3	02 H	0	0	0	0	0	0	1	0
4	04 H	0	0	0	0	0	1	0	0
5	08 H	0	0	0	0	1	0	0	0
6	10 H	0	0	0	1	0	0	0	0
7	20 H	0	0	1	0	0	0	0	0
8	30 H	0	1	0	0	0	0	0	0
9	40 H	1	0	0	0	0	0	0	0
10	FF H	1	1	1	1	1	1	1	1

4.1.5 Analisa percobaan LED output

Dari hasil pengujian dapat diketahui bahwa untuk menyalakan LED dengan driver yang telah dirancang diberikan logika '1' pada pin output mikrokontroler yang terhubung. Logika ini karena LED common anoda dikontrol menggunakan transistor NPN sebagai saklarnya, sehingga saat diberi logika 1 maka transistor berada pada kondisi saturasi yang menyebabkan terjadinya perbedaan tegangan antara kolektor dan emitor, kemudian arus mengalir secara forward bias melalui LED dan menyebabkan LED menyala

4.2 Pengujian Saklar

4.2.1 Tujuan pengujian

Mengetahui inputan yang diberikan saklar terhadap koneksi ke pin mikrokontroler.

4.2.2 Peralatan yang digunakan

1. Peripheral I/O Switch board dan LED board
2. Minimum sistem AVR
3. Kabel konektor
4. DC Power Supply
5. Laptop/PC sebagai media pemrograman

4.2.3 Langkah – langkah pengujian

1. Berikan sumber tegangan pada minimum sistem
2. Koneksikan Port Switch pada I/O board ke port mikrokontroler yang difungsikan sebagai inputan (pada percobaan PORT A) dan port LED pada I/O board ke port ouput mikrokontroler(pada percobaan PORTD)
3. Membuat program untuk menyalakan LED pada PORT D melalui saklar data pada PORTA dengan CodeVisionAVR kemudian mendownload program ke mikrokontroler.
4. Berikan inputan data melalui saklar, dan amati hasil outputan pada LED.
5. Catatkan hasil pengujian pada tabel

4.2.4 Hasil pengujian Saklar

Tabel 4.2 Hasil Pengujian Saklar

No.	Data Saklar yang diinputkan melalui PORTA	Hasil LED yang menyala pada PORTD
1.	Tidak ada	Tidak ada
2.	Bit 0 Enable	LED Bit 0
3.	Bit 1 Enable	LED Bit 1
4.	Bit 2 Enable	LED Bit 2
5.	Bit 3 Enable	LED Bit 3
6.	Bit 4 Enable	LED Bit 4
7.	Bit 5 Enable	LED Bit 5
8.	Bit 6 Enable	LED Bit 6
9.	Bit 7 Enable	LED Bit 7

4.2.5 Analisa percobaan Saklar

Data tabel 4.2 menunjukan bahwa masing – masing LED pada pin mikrokontroler PORTD menyala sesuai dengan inputan data pada saklar data di PORTA

4.3 Pengujian Seven Segment

4.3.1 Tujuan pengujian

Mengetahui logika pembentukan karakter tampilan seven segment

4.3.2 Peralatan yang digunakan

1. Peripheral I/O 7 segment's board
2. AVR minimum sistem
3. Kabel konektor
4. DC power supply
5. Laptop/PC sebagai media pemrograman

4.3.3 Langkah – langkah pengujian

1. Hidupkan minimum sistem dengan memberikan tegangan supply
2. Koneksikan port 7 Segment's ke port output mikrokontroler(dalam percobaan PORTD) dan kontrol seven segment's ke port output(dalam PORTC.6 dan PORTC.7)
3. Membuat program menyalakan seven segment dengan memberi inputan pada PORTD serta mengontrol nyala setiap seven segment dengan inputan data pada port C.7 dan PortC.6
4. Mengamati hasilnya dan mencatat dalam tabel

4.3.4 Hasil pengujian 7 segment's

Tabel 4.3
Hasil pengujian input data seven segment's

No	Data 7 Segment								Data Heksadesimal	Hasil tampilan karakter
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
1	1	1	0	0	0	0	0	0	C0 H	Angka 0
2	1	1	1	1	1	0	0	1	F9 H	Angka 1
3	1	0	1	0	1	0	0	0	A4 H	Angka 2
4	1	0	1	1	0	0	0	0	B0 H	Angka 3
5	1	0	0	1	1	0	0	1	99 H	Angka 4
6	1	0	0	1	0	0	1	0	92 H	Angka 5
7	1	0	1	1	0	0	1	0	B2H	Angka 6
8	1	1	1	1	1	0	0	0	F8 H	Angka 7
9	1	0	0	0	0	0	0	0	80 H	Angka 8
10	1	0	0	1	0	0	0	0	90 H	Angka 9

Tabel 4.4
Hasil pengujian kontrol 7 Segment's

No	Input Kontrol yang diberikan	7 Segment's yang menyala
1	PORTC.7 Enable	Seven Segment 1
2	PORTC.6 Enable	Seven Segment 2

4.3.5 Analisa percobaan 7 Segment's

Pada saat pemberian logika 0 ke data seven segment's menyebabkan kondisi LED menyala pada masing – masing segment's.Hal ini dikarenakan common terhubung pada Vcc, sehingga agar terjadi perbedaan potensial dan LED mendapat forward bias maka pin data seven segment diberikan logika 0. Pada saat pemberian kontrol pada masing – masing transistor menyebabkan masing- masing 7 sement' yang dikontrol mengeluarkan tampilan.Untuk input kontrol enable diberikan logika 0 karena transistor yang digunakan bertipe PNP sehingga terjadi perbedaan tegangan antara kolektor dan emitor yang menyebabkan tegangan mengalir menuju emitor dari kolektor mengisi Vcc seven segment.

4.4 Pengujian LCD

4.4.1 Tujuan pengujian

Mengetahui bagaimana menampilkan karakter pada rangkaian LCD

4.4.2 Peralatan yang digunakan

1. Peripheral I/O LCD board
2. Minimum sistem AVR
3. Kabel konektor
4. DC power supply
5. Laptop/PC sebagai media pemrograman

4.4.3 Langkah – langkah pengujian

1. Hidupkan Minimum sistem dengan memberikan tegangan supply
2. Koneksikan PORT LCD ke PORT mikrokotroler(dalam percobaan PORTB)
3. Membuat program untuk menampilkan kalimat "MODUL PRAKTIKUM MIKROKONTROLER" melalui CodeVision AVR pada LCD dan mendownload hasilnya ke mikrokontroler.
4. Menganalisa hasil tampilan pada LCD.

4.4.4 Hasil pengujian LCD

Gambar 4.1
Hasil Pengujian LCD

4.4.5 Analisa percobaan LCD

LCD dapat menampilkan tulisan yang diinginkan dengan baik sesuai dengan karakter yang diinputkan.

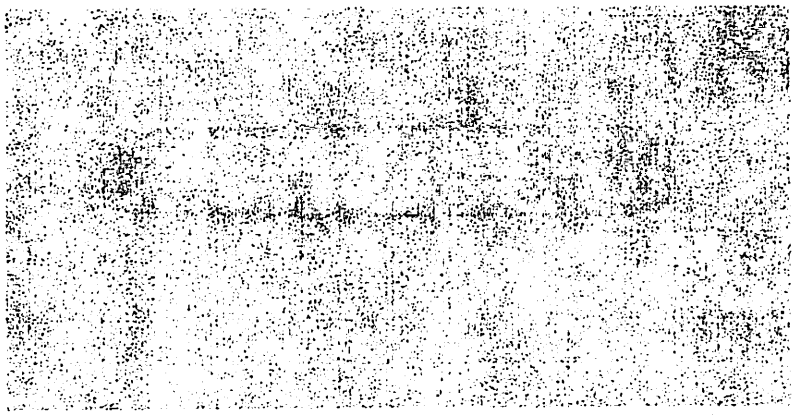
4.5 Pengujian Motor Stepper

4.5.1 Tujuan pengujian

Mengetahui prosedur menjalankan motor steper berdasarkan rangkaian yang dirancang, serta mengimplementasikannya dalam bahasa pemrograman.

4.5.2 Peralatan yang digunakan

1. Peripheral I/O Stepper board
2. AVR minimum sistem
3. Kabel Konektor
4. DC Power Supply
5. Laptop/PC sebagai media pemrograman



4.5.3 Langkah – langkah pengujian

1. Nyalakan minimum sistem dengan memberikan DC power supply
2. Hubungkan konektor antara port mikrokontroler dengan pin pada stepper board peripheral I/O
3. Membuat program untuk menggerakan driver motor melalui CodevisionAVR dan mendownload hasilnya ke mikrokontroler.
4. Amati hasilnya dalam tabel.

4.5.4 Hasil pengujian Motor Steper

Tabel 4.5 Hasil Pengujian motor steper

No	Data Inputan				Hasil putaran
	A	B	C	D	
1	1	0	0	0	Motor berputar berlawanan arah jarum jam
	0	1	0	0	
	0	0	1	0	
	0	0	0	1	
2	0	0	0	1	Motor berputar searah jarum jam
	0	0	1	0	
	0	1	0	0	
	1	0	0	0	

4.5.5 Analisa percobaan Motor Steper

Motor stepper berputar sesuai dengan urutan eksitasi yang diberikan terhadap masing – masing transistornya. Sementara kecepatan motor ditentukan melalui delay pemberian data antara eksitasi satu titik ke titik lainnya.

4.6 Percobaan Dot Matriks

4.6.1 Tujuan pengujian

Mengetahui dasar logika pembentukan karakter pada led dot matriks

4.6.2 Peralatan yang digunakan

1. Peripheral I/O Dot Matriks Board
2. Minimum sistem AVR
3. Kabel konektor
4. DC Power Supply
5. Laptop/PC sebagai media pemrograman

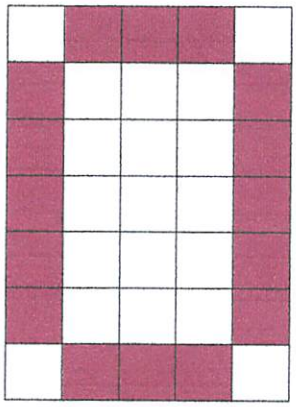
4.6.3 Langkah – langkah pengujian

1. Hidupkan minimum sistem dengan memberikan tegangan
2. Koneksikan Port DotMatriks board dengan port mikrokontroler (padapercobaan PORTD) dan PortIC4017 dengan PORTC mikrokontroler
3. Membuat program untuk menyalakan DotMatriks melalui CodeVisionAvr dan mendownload program ke mikrokontroler.
4. Amati hasil tampilan dan mencatat data pada hasil pengujian.

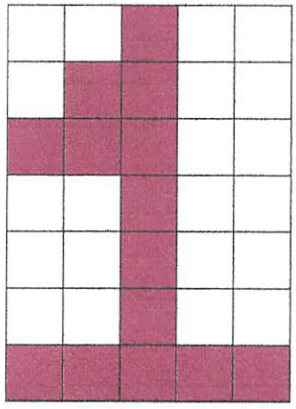
4.6.4 Hasil pengujian Dot Matriks 5x7

Pengujian menguji pembentukan karakter pada dot matriks 5x7 dengan memberikan data pada scan per kolom

Input data melalui scaning kolom(dalam Heksa)
(41H,3CH,3CH,3CH,41H)



Input data melalui scaning kolom(dalam Heksa)
(3BH, 39H, 00H, 3FH, 3FH)



Gambar 4.2
Hasil Tampilan Karakter dot matriks

4.6.5 Analisa percobaan Dot Matriks

Penampilan karakter pada led dot matriks dapat dilakukan dengan memberi tegangan pada kutub anoda dan menghubungkan kutub katoda dengan ground di setiap masing – masing LEDnya. Teknik ini dilakukan pada setiap baris dan kolom yang menyususn LED dot matriks tersebut sehingga nantinya membentuk suatu tampilan karakter yang diinginkan

BAB V
PENUTUP

5.1 Kesimpulan

Kesimpulan yang dapat diambil dari penulisan skripsi ini adalah :

1. Berdasarkan hasil percobaan led bahwa dengan skema rangkaian yang dibuat, untuk memberi hasil menyala pada masing–masing led diberikan logika 1(*high*) sesuai hasil tabel 4.1.
2. Berdasarkan hasil pengujian saklar, setiap transisi kondisi logika dari *low* ke *high* maka saklar memberi inputan pada pin mikrokontroler. Hal ini akan mempengaruhi kondisi output yang diinginkan pada *port* mikrokontroler.
3. Berdasarkan hasil percobaan *7segment's*, untuk menyalakan setiap led pada susunan 7segment's diberikan logika *low(0)*.Sementara input kontrol setiap *7segment's* diberi logika *low* untuk menyalakan *7segment's*. Hal ini ditunjukan pada tabel 4.3 dan tabel 4.4.
4. Berdasarkan hasil percobaan motor stepper, motor dapat berputar karena diberikan data secara bergantian pada setiap bagian penggeraknya. Urutan data mempengaruhi arah putaran, sementara selang waktu antara data yang diberikan mempengaruhi kecepatan motor stepper. Hal ini ditunjukan pada tabel 4.5.
5. Berdasarkan hasil percobaan led dot matriks, setiap titik led yang menyala karna diberikan nilai *high* pada anoda dan *low* pada katodanya. Kombinasi hasil nyala tiap titik led membentuk suatu karakter berdasarkan data inputan yang diberikan melalui mikrokontroler. Hal ini ditunjukan pada gambar 4.2.
6. Penerapan Mikrokontroler ATMega8535/16/32 dan ATMega128L sebagai sistem kontrol minimum sistem AVR dapat digunakan sebagai kontrol *peripheral Input/Output* yang terdiri dari LED *display*, *seven segment's display*, LED *dot* matriks *display*, LCD, *Switch*(Saklar) *Input*, dan *stepper* motor.

7. Peripheral I/O yang dirancang dapat digunakan menjadi simulator indikasi *Input/Output* minimum sistem sesuai hasil percobaan pada tabel 4.1 untuk pengujian LED, tabel 4.2 untuk pengujian saklar, tabel 4.3 dan 4.4 untuk pengujian *7 segment's*, tabel 4.5 untuk pengujian motor steper.

5.2 Saran

Saran untuk penulisan skripsi ini adalah :

1. Perancangan skema rangkaian secara manual mendapat kendala berupa sulitnya pencetakan skema ke papan PCB sehingga menyebabkan kualitas rangkaian PCB manual memiliki nilai estetika yang tak terlalu bagus. Untuk hal ini diharapkan nantinya demi menunjang pembuatan skripsi khususnya yang membutuhkan perancangan alat dapat dilakukan koordinasi antara pihak institusi dengan mahasiswa atau dalam hal lain dapat digunakan jasa pencetakan papan PCB.
2. Pembuatan modul seperti ini masih dapat mengalami perkembangan dalam segi perangkat lunak yang dapat mensimulasikan fitur – fitur mikrokontroler yang sekiranya menunjang pembuatan sistem yang lebih kompleks
3. DT-AVR ATmega128L bootloader micro system dapat digali penggunaanya agar lebih bermanfaat, khususnya untuk perancangan yang memerlukan indikator Input/Output banyak.

DAFTAR PUSTAKA

Andrianto, Heri. 2008. *Pemroograman Mikrokontroler AVR ATMEGA16 Menggunakan Bahasa C(CodeVision AVR)*. Bandung : Informatika.

Prayascitaram, I Gde Made Surya Bumi.2011. *Perancangan dan Pembuatan Modul Pelatihan RobotikaUntuk Pengembangan Tim Robot ITN Malang*. **LCD** 14-15.

Susilo, Dedy. 2010. *48 jam kupas tuntas Mikrokontroler MCS51 & AVR*. Yogyakarta : ANDI.

Winoto, Ardi. 2010. *Mikrokontroler AVR ATMega8/32/16/8535 dan Pemrogramannya dengan Bahasa C pada WinAVR*. Edisi Revisi. Bandung : Informatika.

Atmel. 2006. *ATMega8535 Datasheet*. URL: http://www.atmel.com/Images/doc2502.pdf

Atmel. 2010. *ATMega16 Datasheet*. URL: http://www.atmel.com/Images/doc2466.pdf

Atmel. 2011. *ATMega32 Datasheet*. URL: http://www.atmel.com/Images/doc2503.pdf

Atmel. 2011. *ATMega128L Datasheet*. URL: http://www.atmel.com/Images/doc2467.pdf

Hadi, M.Sholihul. 2003. *Mengenal Mikrokontroler AVR ATMega16*. URL : http://ilmukomputer.org/wp-content/uploads/2008/08/sholihul-atmega16.pdf

Penulis. 2008. *ChapterII*. URL : http://repository.usu.ac.id/bitstream/123456789/27219/4/Chapter%20II.pdf

Wikipedia. 2012. *Light Emitting Diode*. URL : http://en.wikipedia.org/w/index.php?title=Special:Book&bookcmd=download&collection_id=9a7663e3d12e3d5a&writer=rl&return_to=Light-emitting+diode

Wikipedia. 2012. *Switch*. URL: http://en.wikipedia.org/w/index.php?title=Special:Book&bookcmd=download&collection_id=aa72a801d4de6eca&writer=rl&return_to=Switch

Wikipedia. 2012. *Seven Segment's Display. URL :* http://en.wikipedia.org/w/index.php?title=Special:Book&bookcmd=download&collection_id=6289f7437823354b&writer=rl&return_to=Seven-segment+display

LAMPIRAN

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
PT. BNI (PERSERO) MALANG
BANK NIAGA MALANG
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

FORMULIR PERBAIKAN SKRIPSI

Dalam pelaksanaan ujian skripsi jenjang Strata Satu (S-1) Program Studi Teknik Elektro Konsentrasi Teknik Elektronika, maka perlu adanya perbaikan skripsi untuk mahasiswa :

NAMA : **MADE AGUS ANGGASANA PUTRA**
JURUSAN : **Teknik Elektro S-1**
KONSENTRASI : **Teknik Elektronika**
MASA BIMBINGAN : **Semester Genap 2011-2012**
JUDUL : **RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA8535/16/32 DAN ATMEGA128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR**

Tanggal	Uraian	Paraf
Panguji I 2 – 08 - 2012	Ditambahkan ke "spesifik modul praktikum" untuk mempermudah pemahaman dalam praktikum	
	Pemilihan bahasa C dan software CodeVisionAVR perlu dijelaskan	
	Untuk setiap percobaan/peripheral perlu dijelaskan secara rinci	
Pengguji II 2 – 08 - 2012	Usahakan lebih interaktif dan lebih komunikatif dalam penyajiannya pada panduan praktikum.	

PANITIA UJIAN SKRIPSI

Ketua Majelis Penguji

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto Soetedjo, ST. MT
NIP.Y.1030800417

ANGGOTA PENGUJI

Penguji I

Dr. Eng. Aryuanto Soetedjo, ST. MT
NIP.Y.1030800417

Penguji II

Ir. Eko Nurcahyo , MT
NIP. Y. 1028700172

PERKUMPULAN PENGELOLA PENDIDIKAN UMUM DAN TEKNOLOGI NASIONAL MALANG
INSTITUT TEKNOLOGI NASIONAL MALANG
FAKULTAS TEKNOLOGI INDUSTRI
FAKULTAS TEKNIK SIPIL DAN PERENCANAAN
PROGRAM PASCASARJANA MAGISTER TEKNIK
Kampus I : Jl. Bendungan Sigura-gura No. 2 Telp. (0341) 551431 (Hunting), Fax. (0341) 553015 Malang 65145
Kampus II : Jl. Raya Karanglo, Km 2 Telp. (0341) 417636 Fax. (0341) 417634 Malang

BERITA ACARA UJIAN SKRIPSI
FAKULTAS TEKNOLOGI INDUSTRI

Nama	:	**MADE AGUS ANGGASANA PUTRA**
Nim	:	**08.12.213**
Jurusan	:	**Teknik Elektro**
Konsentrasi	:	**Teknik Elektronika S-1**
Masa Bimbingan	:	**Semester Genap 2011-2012**
Judul	:	**RANCANG BANGUN MODUL PRAKTIKUM MIKROKONTROLER ATMEGA8535/16/32 DAN ATMEGA128L DI LABORATORIUM ELEKTRONIKA DIGITAL ITN MALANG BERBASIS MINIMUM SISTEM AVR**

Dipertahankan dihadapan Tim Penguji Skripsi Jenjang Program Strata Satu (S-1)

Pada Hari : Selasa
Tanggal : 7 Agustus 2012
Dengan Nilai : 83 (A)

PANITIA UJIAN SKRIPSI

Ketua Majelis Penguji

Ir. Yusuf Ismail Nakhoda, MT
NIP.Y.1018800189

Sekretaris Majelis Penguji

Dr. Eng. Aryuanto Soetedjo, ST. MT
NIP.Y.1030800417

ANGGOTA PENGUJI

Penguji I

Dr. Eng. Aryuanto Soetedjo, ST. MT
NIP.Y.1030800417

Penguji II

Ir. Eko Nurcahyo , MT
NIP. Y. 1028700172

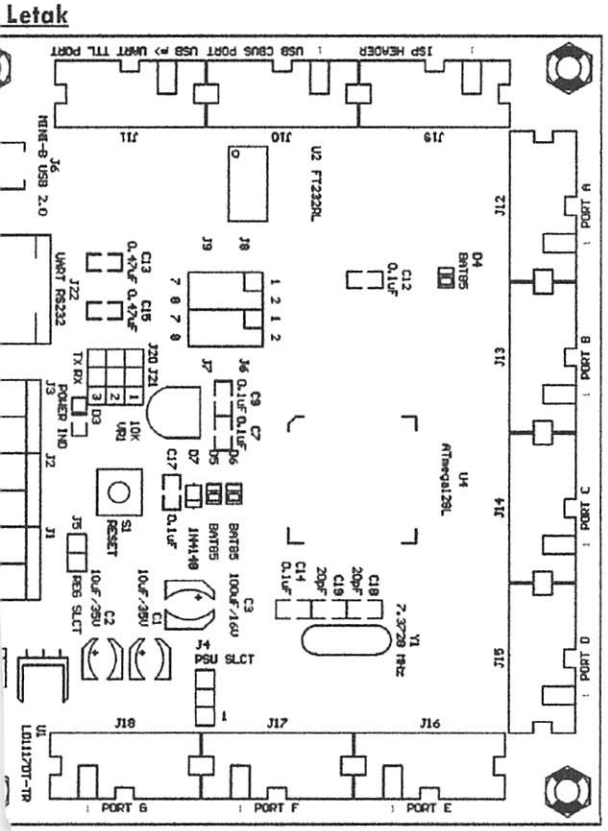
T-AVR ATmega128L
OTLOADER MICRO SYSTEM

AVR ATmega128L BOOTLOADER RO SYSTEM (BMS) merupakan sebuah modul *chip* berbasis mikrokontroler **ATmega128L**. DT-AVR ga128L BMS dilengkapi dengan program *oader* sehingga tidak membutuhkan divais *rammer*. Dengan menggunakan *bootloader* pada DT-ATmega128L BMS, pengguna dapat menggunakan UART sebagai jalur komunikasi dengan komputer, ligus menggunakannya untuk melakukan *remote ramming* jika ada perbaikan program (*update*). *ware* yang digunakan untuk memprogram okontroler adalah AVR Bootloader© v1.0.

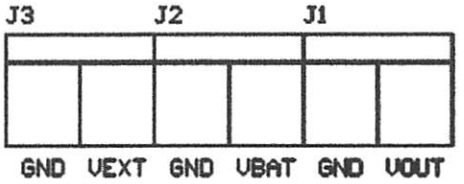
sifikasi

- Berbasis mikrokontroler **ATmega128L** dengan Flash memory sebesar 124 Kbyte (4 Kbyte telah digunakan untuk *bootloader*) dan 8 channel ADC 10 bit.
- Dilengkapi dengan program *bootloader* yang dapat diprogram menggunakan *software* AVR Bootloader© v1.0.
- Memiliki hingga 53 jalur Input/Output.
- Tersedia jalur komunikasi serial UART melalui USB atau UART RS-232 melalui konektor RJ45, sekaligus sebagai jalur untuk pemrograman mikrokontroler.
- Frekuensi osilator sebesar 7.3728 MHz.
- Tersedia rangkaian reset manual dengan *tactile switch*.
- Dilengkapi dengan pilihan regulator 3,3V atau 5V dengan arus maksimum 800 mA.
- Tersedia pilihan catu daya input: 6 - 12 VDC (via regulator) atau 3,3 - 5,5 VDC (tanpa regulator).
- Tersedia terminal tegangan output.

Letak



Terminal J3 (VEXT) merupakan jalur catu daya masuk untuk tegangan 6 - 12 VDC (via regulator).
Terminal J2 (VBAT) merupakan jalur catu daya masuk untuk tegangan hingga 5,5 VDC (tanpa melalui regulator)
Terminal J1 (VOUT) merupakan jalur tegangan keluar (dari VEXT via regulator ataupun dari VBAT).



Jumper PSU SLCT (J4) digunakan untuk memilih sumber catu daya, dari sumber eksternal secara langsung atau melalui regulator.

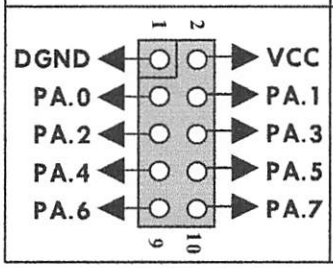
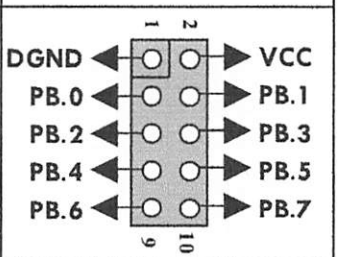
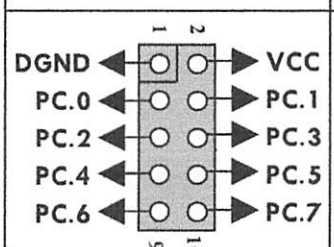
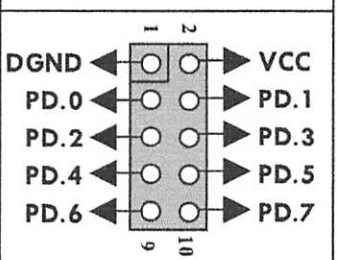
Posisi PSU SLCT (J4)	
	
Catu daya modul berasal dari VEXT (via regulator)	Catu daya modul berasal langsung dari VBAT

Penting!
Perhatikan *range* tegangan kerja (3,3 - 5,5VDC) jika sumber catu daya menggunakan VBAT (tanpa regulator).

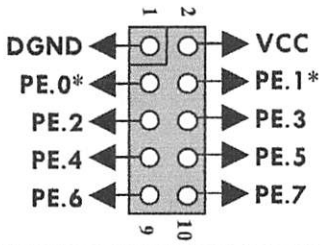
Jumper REG SLCT (J5) digunakan untuk memilih tegangan output dari regulator.

Posisi REG SLCT (J5)	
 Tegangan output regulator = 3,3V	 Tegangan output regulator = 5V

Header PORT A (J12), PORT B (J13), PORT C (J14), PORT D (J15), PORT E (J16), PORT F (J17), dan PORT G (J18) berfungsi sebagai jalur pin I/O.

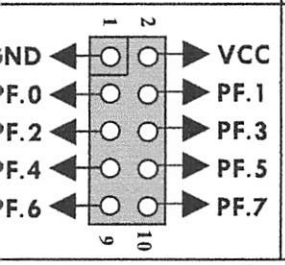
PORT A (J12)	PORT B (J13)
	
PORT C (J14)	**PORT D (J15)**
	

PORT E (J16)

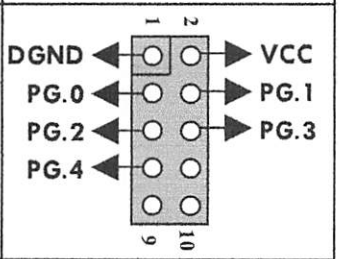


ka PE.0 dan PE.1 dihubungkan ke rangkaian USB au RS-232, maka pin PE.0 dan PE.1 tidak hubung ke PORT E

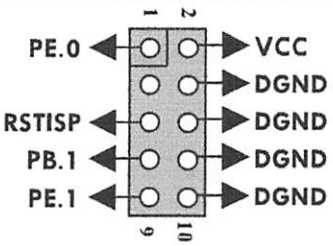
PORT F (J17)



PORT G (J18)

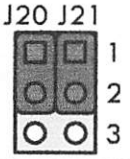
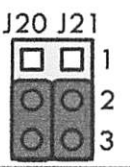


ISP HEADER (J19)



dak disarankan memprogram secara ISP melalui P HEADER. Jika modul diprogram ulang secara P, maka *bootloader* akan terhapus.

per TX (J20) dan RX (J21) digunakan untuk memilih si dan jalur PE.0 (J21) dan PE.1 (J20).

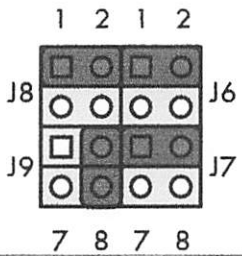
ngsi PE.0 dan PE.1	Posisi J20 dan J21
UART (via USB) atau UART RS-232 (via RJ45)	(J20 = 1 - 2) (J21 = 1 - 2) 
UART TTL atau I/O digital (via PORT E)	(J20 = 2 - 3) (J21 = 2 - 3) 

ing!

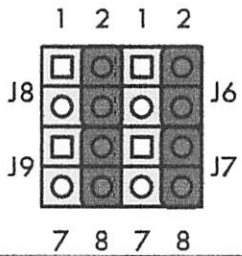
dan PE.1 akan selalu terhubung ke ISP HEADER (J19) a tergantung pengaturan *jumper* J20 dan J21.

per J6, J7, J8, dan J9 digunakan untuk memilih jalur unikasi maupun jalur pemrograman melalui loader (dengan menekan tombol **Auto Detect Device** a *software* AVR Bootloader v1.0).

Posisi J6, J7, J8, dan J9

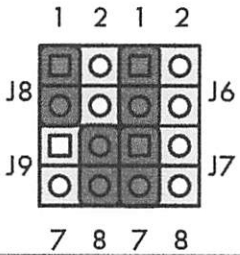


Jalur PE.0 dan PE.1 terhubung ke rangkaian USB (mini USB *port* berfungsi sebagai jalur komunikasi maupun pemrograman via *bootloader*)



Jalur PE.0 dan PE.1 terhubung ke rangkaian RS-232 (RJ45 J22 sebagai jalur komunikasi maupun pemrograman via *bootloader*)

Posisi J6, J7, J8, dan J9



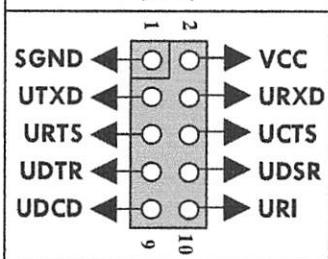
Rangkaian USB dan RS-232 berfungsi sebagai USB to RS-232 Converter (antara mini USB *port* dan RJ45) dan tidak ada yang terhubung ke PE.0 ataupun PE.1.

Header USB => UART TTL PORT (J11) merupakan jalur UART TTL dari USB Converter.
Header USB CBUS PORT (J10) merupakan jalur Control Bus dari USB Converter.

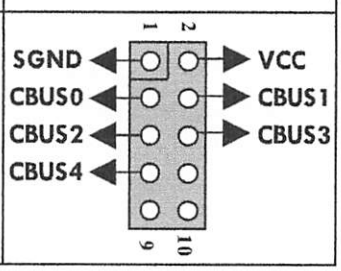
Penting!

Perhatikan koneksi URXD dan UTXD jika USB digunakan sebagai jalur komunikasi atau pemrograman.

USB => UART TTL PORT (J11)

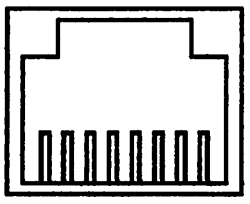


USB CBUS PORT (J10)



Nama	Fungsi	Deskripsi Singkat
CBUS0	TXLED	Menghasilkan pulsa *low* saat ada pengiriman data
CBUS1	RXLED	Menghasilkan pulsa *low* saat ada penerimaan data
CBUS2	PWREN	Berlogika *low* jika terhubung dan terkonfigurasi oleh USB *port* komputer, berlogika *high* saat mode *suspend*
CBUS3	TXDEN	Sebagai sinyal *enable* untuk pengiriman data pada RS485
CBUS4	SLEEP	Berlogika *low* saat mode *suspend*

ktor RJ45 UART RS232 adalah konektor untuk jalur nikasi UART RS-232 igai Data Communication iment).



Nama	Fungsi
DSR	Tidak digunakan
DCD	Tidak digunakan
DTR	Tidak digunakan
GND	Referensi ground
RXD	Jalur data keluar
TXD	Jalur data masuk
CTS	Jalur keluar sinyal Request To Send
RTS	Jalur masuk sinyal Clear To Send

yang dapat digunakan oleh mikrokontroler hanya dan RXD, sinyal lain digunakan untuk pemrograman lui *bootloader* dan USB to RS-232 Converter.

lasan lebih lanjut mengenai fitur mikrokontroler dan Converter (FT232R) terdapat pada *datasheet*.
pemrograman disertakan pada Manual AVR oader v1.0.

D/DVD

CodeVisionAVR© versi evaluation.
AVR Bootloader© v1.0 dari Innovative Electronics.
Manual AVR Bootloader.
Manual, Skema, & How2Use DT-AVR ATmega128L MS.
Program uji Tester128.exe dan Testing128.c dalam ahasa C CodeVisionAVR©.
Datasheet.
Website Innovative Electronics.

dur Pengujian

am yang telah disertakan (testing128.hex) akan eluarkan logika *low* pada setiap pin dalam masing-g *port* (PORTA, PORTB, PORTC, PORTD, PORTE, F, dan PORTG) secara bergantian, kecuali pin PE.0 E.1 yang difungsikan untuk komunikasi serial.

cah-langkah untuk menguji modul adalah sebagai ut:

turlah *jumper* J20 dan J21 pada posisi 1-2 agar E.0 dan PE.1 berfungsi sebagai jalur komunikasi erial.

turlah J6, J7, J8, dan J9 agar menggunakan jalur SB untuk *bootloader* atau jalur RS-232 untuk *ootloader*.

turlah *jumper* J4 pada posisi 1-2 agar semua omponen pada modul mendapat sumber tegangan ari regulator

epas *jumper* pada J5 agar tegangan output egulator bernilai 5 VDC.

ubungkan kabel USB ke modul dan PC jika enggunakan jalur USB untuk *bootloader*, atau ubungkan kabel serial ke COM *port* komputer dan onektor RJ45 pada modul jika menggunakan jalur S-232 untuk *bootloader*.

6. Hubungkan sumber tegangan 6-12 VDC ke VEXT.
7. Jika jalur USB digunakan untuk *bootloader* dan belum pernah ada instalasi *driver* FT232RL, maka Windows® akan meminta instalasi *driver* tersebut terlebih dahulu (tersedia dalam CD/DVD).
8. Jalankan program AVR Bootloader V1.0.exe.
9. Tekan tombol **Auto Detect Device**, program ini akan mendeteksi adanya modul yang dihubungkan ke COM *port* komputer. Jika ada modul yang terdeteksi, AVR Bootloader v1.0 akan memberikan informasi mengenai divais dan COM *port* komputer yang digunakan.
10. Jika tidak ada modul yang terdeteksi, maka akan muncul pesan "*No bootloader device found, please select it manually*". Tekan tombol *reset* pada modul dan ulangi penekanan tombol **Auto Detect Device.**
11. Buka *file* testing128.hex melalui menu **File > Load FLASH.** Pastikan **centang** semua **Operation Flow (Check Signature, Erase, Blank Check, Program Verify).** Centang **Programmed Section** pada menu **Flash.**
12. Tekan tombol **Run** untuk memprogram FLASH.
13. Jalankan program TESTER128.EXE. Tentukan COM *port* yang akan digunakan. Tekan tombol **Connect.**
14. Jika komunikasi serial berjalan dengan sukses maka akan tampil 1 baris tulisan yaitu "DT-AVR Boot, Innovative Electronics".
15. Output yang dihasilkan pada PORT A, PORT B, PORT C, PORT D, PORT E, PORT F, dan PORT G dapat diperiksa mengggunakan osiloskop, voltmeter, atau dihubungkan langsung dengan rangkaian LED atau DT-I/O LED LOGIC TESTER sehingga tampak nyala LED yang bergantian.

♦ *Terima Kasih atas kepercayaan Anda menggunakan produk kami, bila ada kesulitan, pertanyaan atau saran mengenai produk ini silahkan menghubungi technical support kami:*

support@innovativeelectronics.com

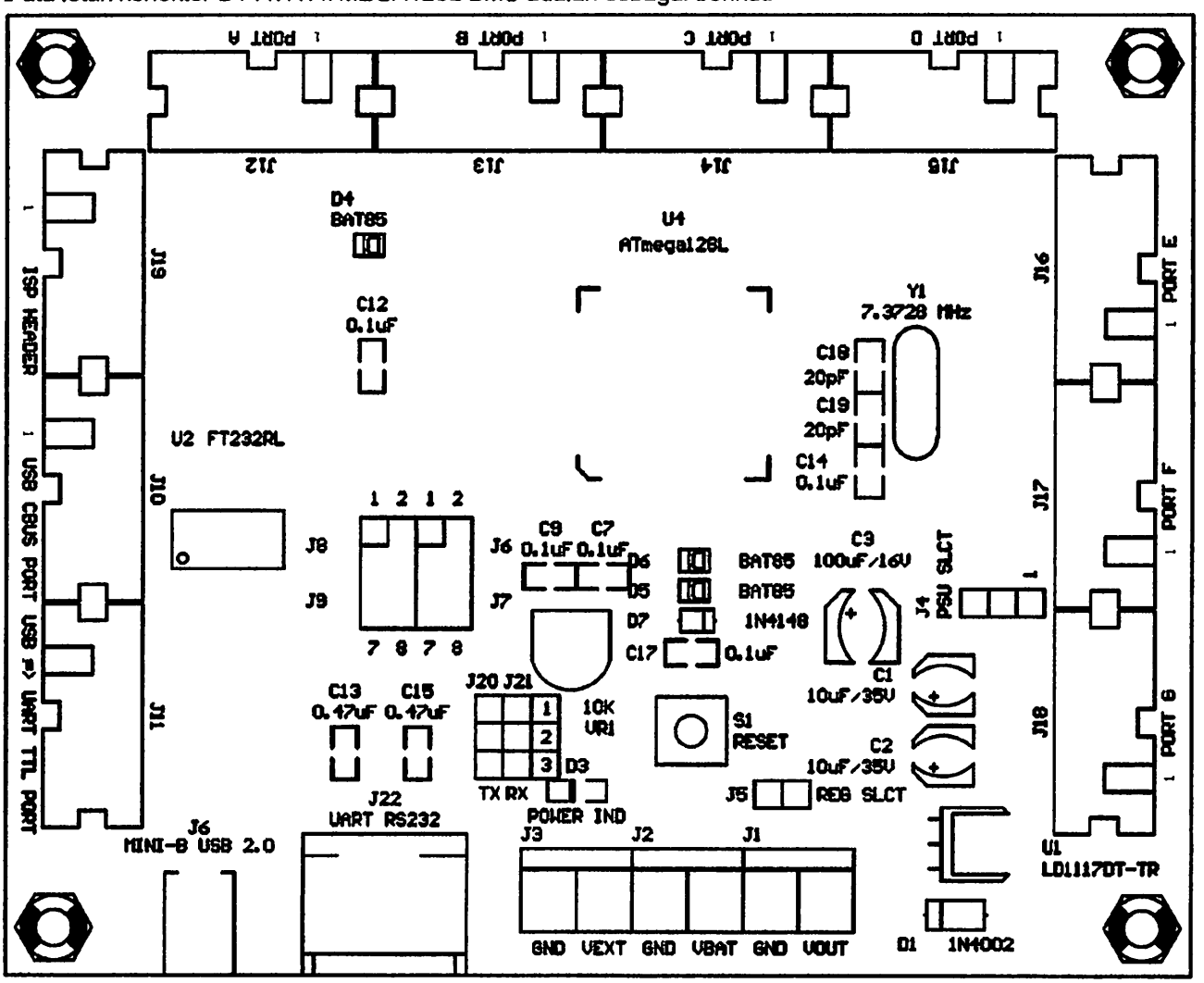
DT-AVR ATMEGA128L BMS *Application Note*

How2Use DT-AVR ATMEGA128L BMS

Oleh: Tim IE

Application Note (AN) ini disusun untuk memberikan penjelasan tentang cara penggunaan DT-AVR ATMEGA128L Bootloader Micro System beserta *software* pendukungnya.

Tata letak konektor DT-AVR ATMEGA128L BMS adalah sebagai berikut:



Gambar 1
Tata Letak DT-AVR ATMEGA128L BMS

Persiapan *hardware* DT-AVR ATMEGA128L BMS adalah sebagai berikut:

1. Atur *jumper* J20 dan J21 pada posisi 1-2 agar PE.0 dan PE.1 berfungsi sebagai jalur komunikasi serial.
2. Atur *jumper* J6, J7, J8, dan J9 agar menggunakan jalur USB untuk *bootloader* atau jalur RS-232 untuk *bootloader*.
3. Hubungkan kabel USB ke modul dan PC jika menggunakan jalur USB untuk *bootloader*, atau hubungkan kabel serial ke COM *port* komputer dan konektor RJ45 pada modul jika menggunakan jalur RS232 untuk *bootloader*.
4. Jika terdapat rangkaian atau modul lain yang akan dihubungkan ke DT-AVR ATMEGA128L BMS,

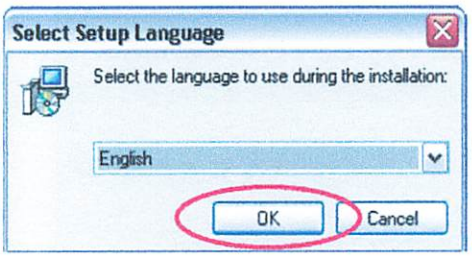
disarankan untuk menghubungkan rangkaian tersebut dengan DT-AVR ATMEGA128L BMS terlebih dahulu. Perhatikan koneksi, terutama untuk jalur VCC dan GND jangan sampai terbalik.

5. Atur *jumper* J4 pada posisi 1-2 (sumber tegangan dari VEXT) dan lepas *jumper* pada J5 (5V).
6. Hubungkan catu daya 6 - 12 VDC ke konektor "VEXT" untuk memberi tegangan ke modul.

Instalasi CodeVisionAVR evaluasi adalah sebagai berikut:

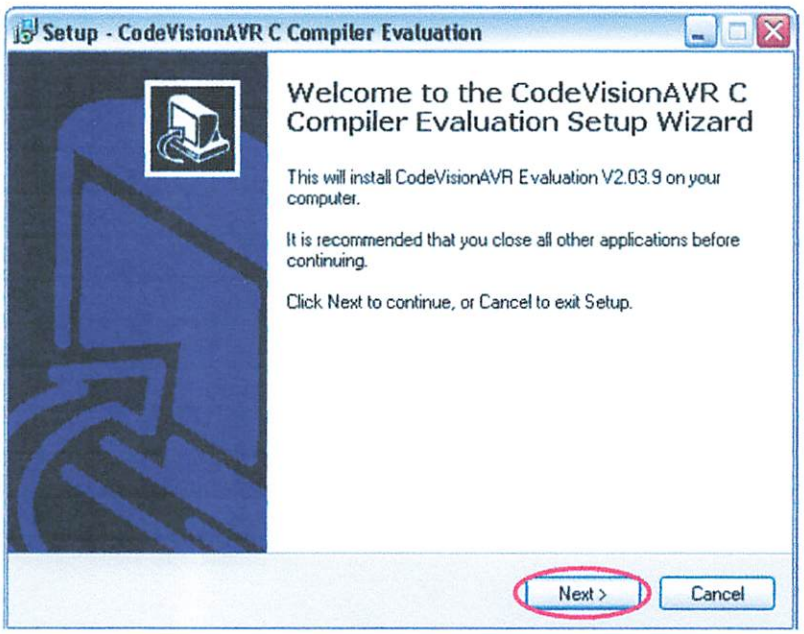
1. CodeVisionAVR evaluasi terdapat pada CD/DVD program yang disertakan. *File* **setup.exe** ada pada *folder* **CVAVR Evaluation**. Jalankan *file* setup.exe untuk melakukan proses instalasi.

2. Pilih bahasa yang akan digunakan, lalu tekan tombol **OK.**

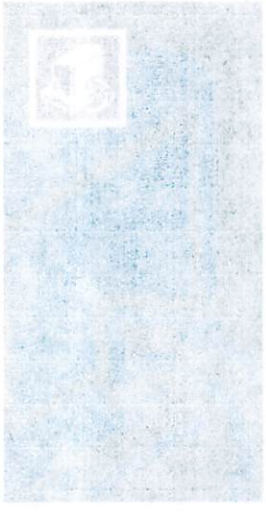


Gambar 2
Pilih Bahasa Instalasi

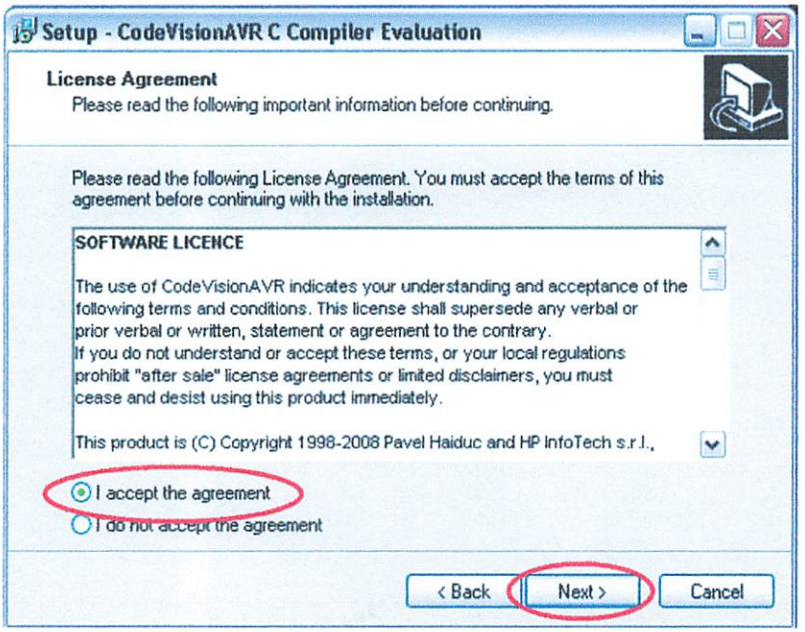
3. Tampilan awal instalasi CodeVisionAVR. Tekan tombol **Next >** untuk melanjutkan proses instalasi.



Gambar 3
Awal Instalasi CodeVisionAVR

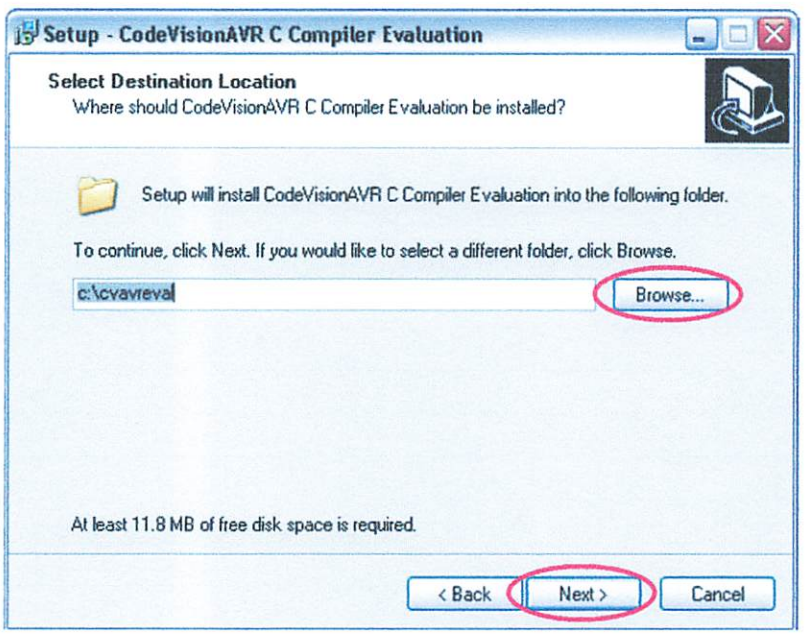


4. Kemudian masuk pada *License Agreement*. Klik pada "I accept the agreement" lalu tekan tombol **Next >** untuk menyetujui lisensi dan melanjutkan proses instalasi.



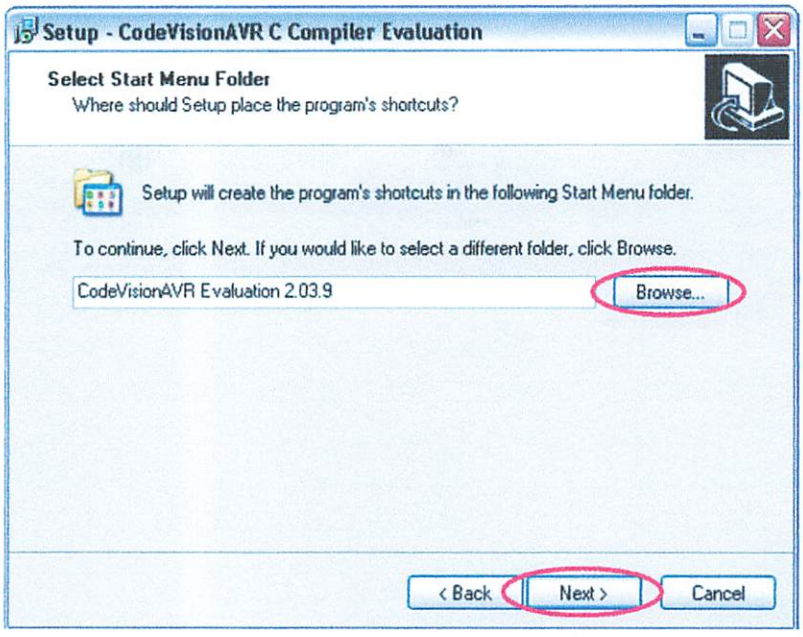
Gambar 4
Persetujuan Lisensi CodeVisionAVR

5. Pilih lokasi instalasi CodeVisionAVR, lalu tekan tombol **Next >**.



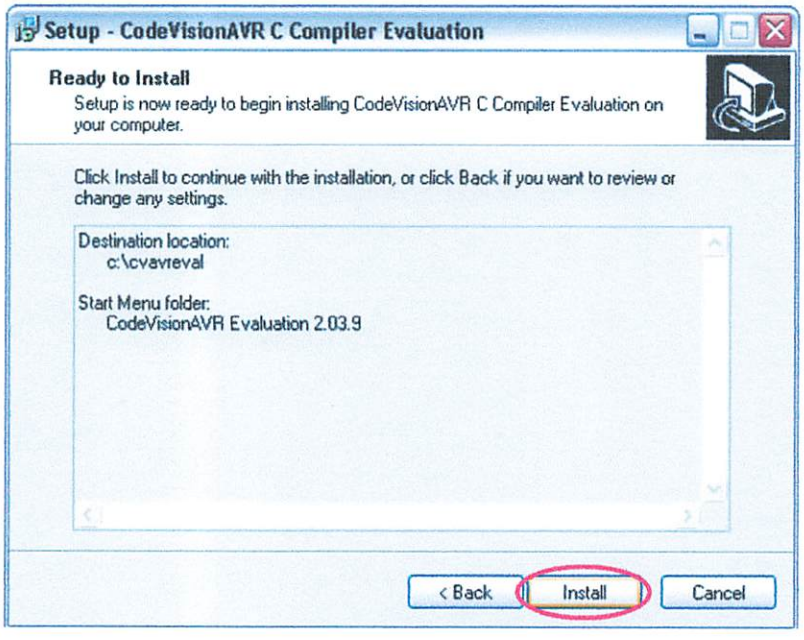
Gambar 5
Pemilihan Lokasi Instalasi

6. Pilih *folder* pada menu Start, lalu tekan tombol **Next >**.



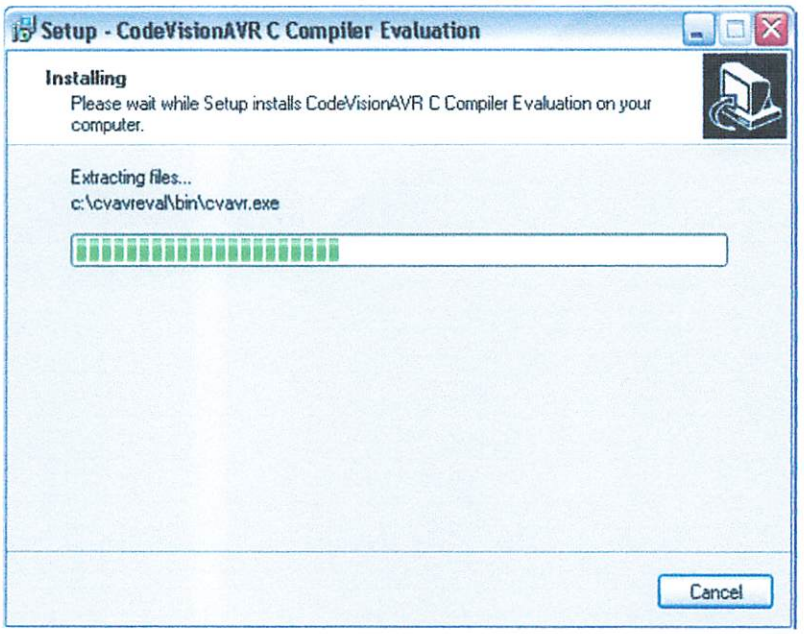
Gambar 6
Tahap Pemilihan Folder Pada Menu Start

7. Persiapan instalasi CodeVisionAVR. Tekan tombol **Install** untuk memulai proses instalasi.



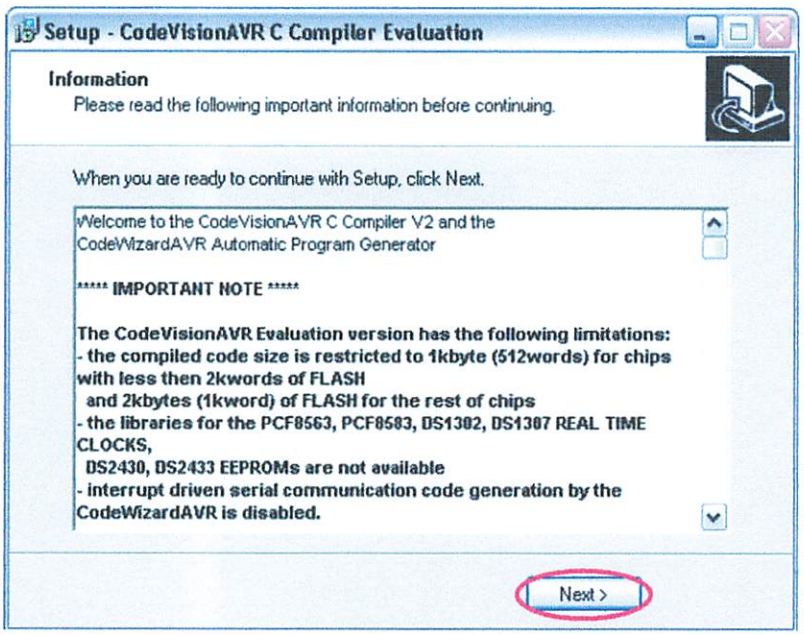
Gambar 7
Tahap Persiapan Instalasi

8. Proses instalasi CodeVisionAVR Evaluation.



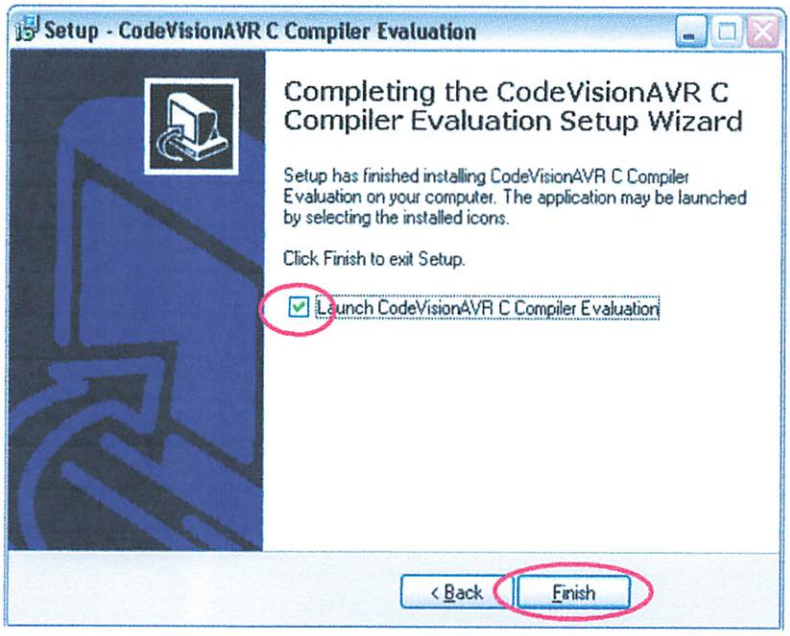
Gambar 8
Proses Instalasi CodeVisionAVR Evaluation

9. Kemudian masuk pada tampilan informasi. Tekan tombol **Next >** untuk melanjutkan proses instalasi.



Gambar 9
Tampilan Informasi

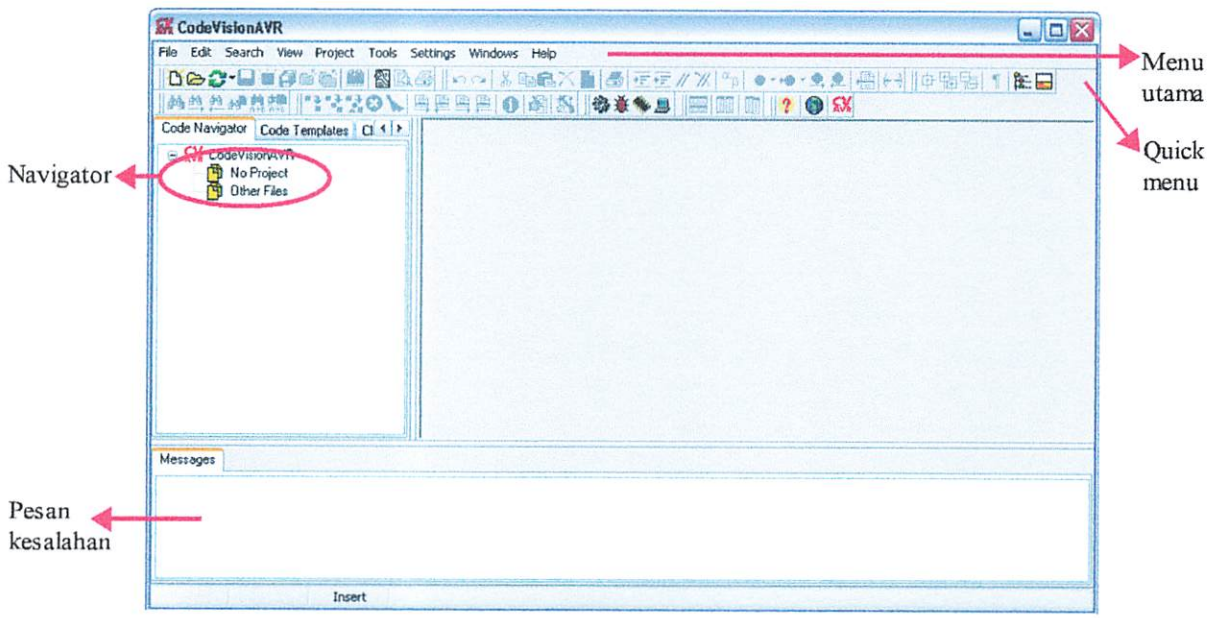
10. Proses instalasi selesai, centang "Launch CodeVisionAVR C Compiler Evaluation" untuk membuka CodeVisionAVR, atau sebaliknya jika tidak ingin membuka CodeVisionAVR setelah proses instalasi selesai. Tekan tombol **Finish** untuk menyelesaikan proses instalasi.



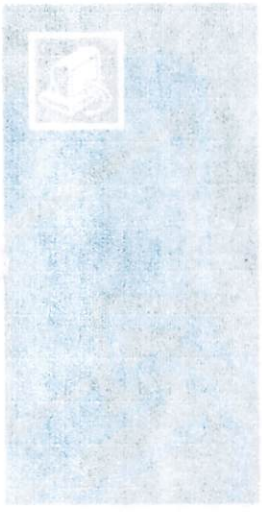
Gambar 10
Proses Instalasi CodeVisionAVR Selesai

Tahap pembuatan aplikasi sederhana menggunakan CodeVisionAVR Evaluation adalah sebagai berikut:

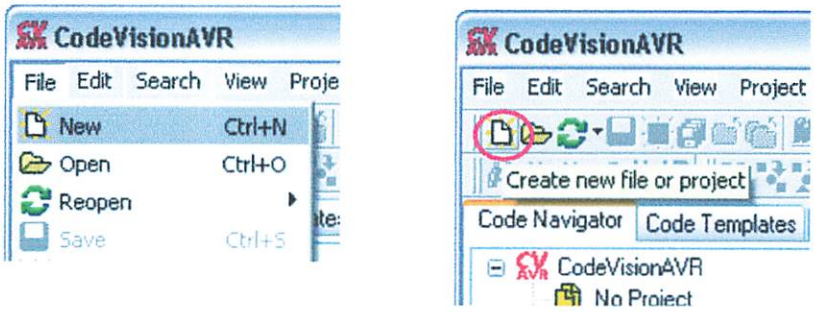
1. Tampilan dasar CodeVisionAVR Evaluation dan menu yang tersedia.



Gambar 11
Tampilan Dasar CodeVisionAVR Evaluation

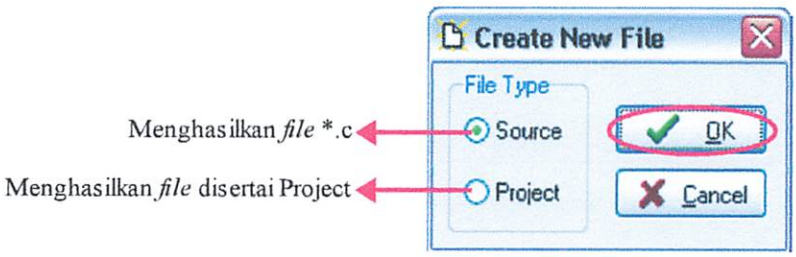


2. Buat *file* baru melalui menu utama, pilih **File**, lalu pilih **New.** Bisa juga melalui Quick menu **Create new file or project**.



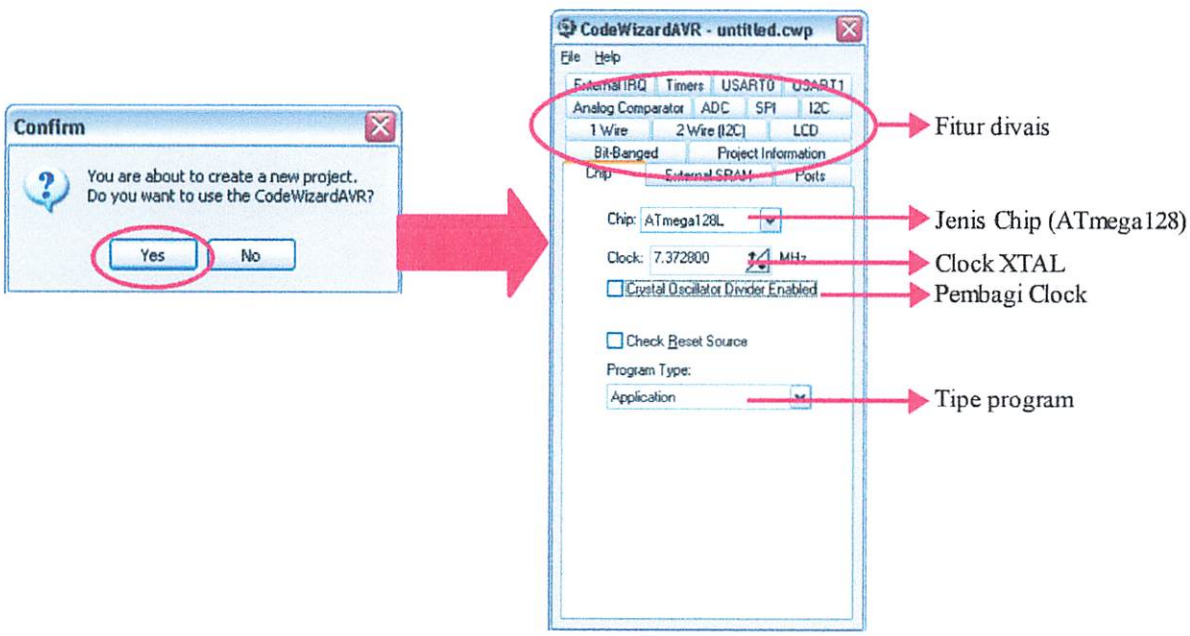
Gambar 12
Membuat *File* Baru

3. Kita akan melihat tampilan **Create New File**. Pada tampilan ini kita bisa membuat *file* disertai **project** atau kita hanya membuat *source file* yang berekstensi ***.c** saja. Kemudian tekan **OK**.



Gambar 13
Tampilan Create New File

4. Untuk mempermudah memulai pembuatan *file*, disarankan memilih opsi **Project**. Karena CodeVisionAVR Evaluation akan membantu melalui **CodeWizardAVR**. Pada **CodeWizardAVR** ini dapat memilih jenis **Chip** dan aktifasi fitur dari **Chip**. Pada saat muncul tampilan konfirmasi tekan tombol **Yes**, sehingga **CodeWizardAVR** muncul.



Gambar 14
Tampilan Konfirmasi dan CodeWizardAVR

5. Pada pembuatan contoh aplikasi ini akan dibahas mengenai program **testing128.c** yang disertakan dalam CD/DVD program. Fitur divais yang digunakan adalah komunikasi serial **USART** yang digunakan untuk melakukan komunikasi secara serial dengan komputer. Berikut penjelasan dari program testing128.c.

```
#include <mega128.h>
#include <delay.h>

// Standard Input/Output functions
#include <stdio.h>
```

Pada program di atas menunjukkan *library* yang digunakan. Pada baris pertama menunjukkan jenis *Chip* yang digunakan. Pada baris kedua merupakan *library delay* yang digunakan untuk menghasilkan waktu *delay*. Yang tercetak biru pada program menunjukkan *statement* dari program. Setiap kalimat yang didahului dengan garis miring rangkap 2 pada program tidak akan dieksekusi. Pada baris keempat menunjukkan *library* Standard Input/Output, yang digunakan untuk komunikasi serial.

6. Berikut penjelasan mengenai fungsi main() dari program testing128.c:

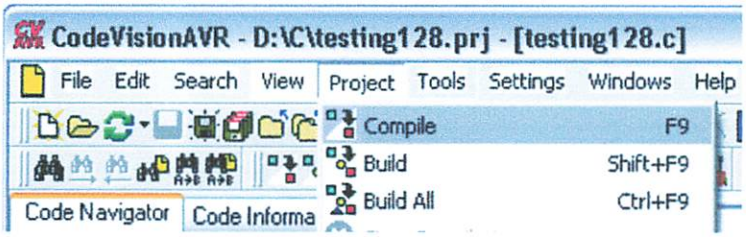
```
void main(void)
{
// Declare your local variables here
unsigned char counter=0,serialCount = 0;

// Input/Output Ports initialization
// Port A initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTA=0x00;
DDRA=0xFF;

// Port B initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTB=0x00;
DDRB=0xFF;

// Port C initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTC=0x00;
DDRC=0xFF;

// Port D initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTD=0x00;
DDRD=0xFF;

// Port E initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTE=0x00;
DDRE=0xFF;

// Port F initialization
// Func7=Out Func6=Out Func5=Out Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State7=0 State6=0 State5=0 State4=0 State3=0 State2=0 State1=0 State0=0
PORTF=0x00;
DDRF=0xFF;

// Port G initialization
// Func4=Out Func3=Out Func2=Out Func1=Out Func0=Out
// State4=0 State3=0 State2=0 State1=0 State0=0
PORTG=0x00;
DDRG=0x1F;

// USART0 initialization
// Communication Parameters: 8 Data, 1 Stop, No Parity
// USART0 Receiver: On
```

```
// USART0 Transmitter: On
// USART0 Mode: Asynchronous
// USART0 Baud rate: 115200
UCSR0A=0x00;
UCSR0B=0x18;
UCSR0C=0x06;
UBRR0H=0x00;
UBRR0L=0x03;

while (1)
      {
      // Place your code here

      if(UCSR0A>>7 & 1)
          {
          if (getchar()=='A')
              {
              printf("%d -> DT-AVR Boot, Innovative Electronics\r",serialCount);
              serialCount++;
              }
          }

      PORTA = ~(0x01 << counter);
      PORTB = ~(0x01 << counter);
      PORTC = ~(0x01 << counter);
      PORTD = ~(0x01 << counter);
      PORTE = ~(0x01 << counter);
      PORTF = ~(0x01 << counter);
      PORTG = ~(0x01 << counter);
      counter++;
      if (counter>7) counter = 0;

      delay_ms(200);
      };
}
```

Pada fungsi **main()**, yang pertama dilakukan adalah deklarasi variabel **counter** dan **serialCount.** Kemudian diikuti dengan inisialisasi I/O dan inisialisasi UART. Selanjutnya program akan melakukan *looping* terus menerus. Pada *looping* tersebut, yang pertama dilakukan adalah memeriksa isi register **UCSR0A** untuk mengetahui apakah ada data serial yang masuk. Jika ada, maka data serial tersebut akan diambil menggunakan fungsi **getchar()**. Jika data serial yang diterima adalah karakter 'A', maka modul akan mengirimkan tulisan "DT-AVR Boot, Innovative Electronics".

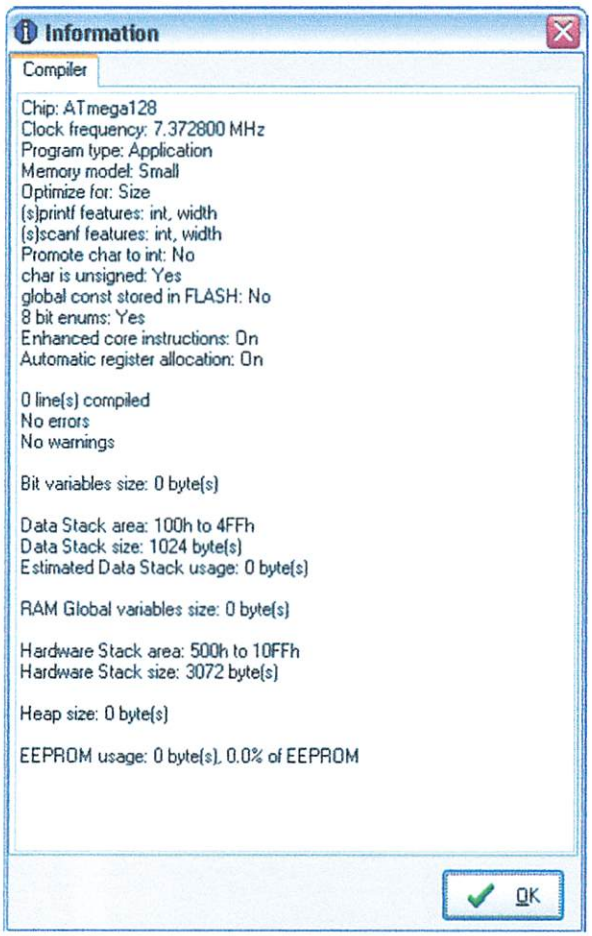
Pada Port A, Port B, Port C, Port D, Port E, Port F, dan Port G akan selalu dikeluarkan logika *low* pada 1 pin dalam setiap *port* secara bergantian, kecuali pin PE.0 dan PE.1 yang difungsikan untuk komunikasi serial.

9. Langkah selanjutnya adalah melakukan *compiling* terhadap program yang telah dibuat. Proses ini dilakukan untuk memeriksa kesalahan penulisan program. Proses ini terdapat pada menu **Project - Compile** atau bisa juga dengan menekan tombol **F9** pada *keyboard*.



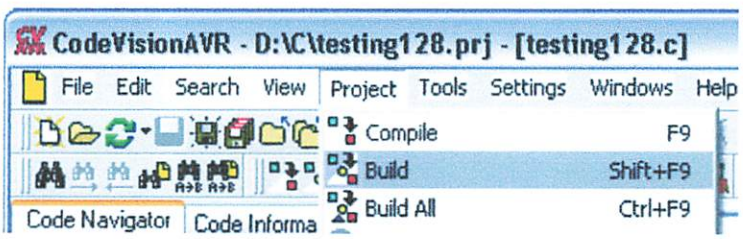
Gambar 15
Menu *Compile*

10. Jika *listing* program yang dibuat tidak mengandung kesalahan (termasuk kesalahan *syntax* atau penulisan), maka akan tampil *form* sebagai berikut.



Gambar 16
Hasil Proses *Compile*

11. Setelah proses *compiling* maka langkah selanjutnya adalah membentuk *file* HEX (*.hex). Proses ini bisa dilakukan jika proses *compiling* berhasil dilakukan. Pembentukan *file* HEX ini melalui menu **Project - Bulid**. Atau bisa juga dilakukan dengan menekan tombol **Shift** + **F9** pada keyboard.

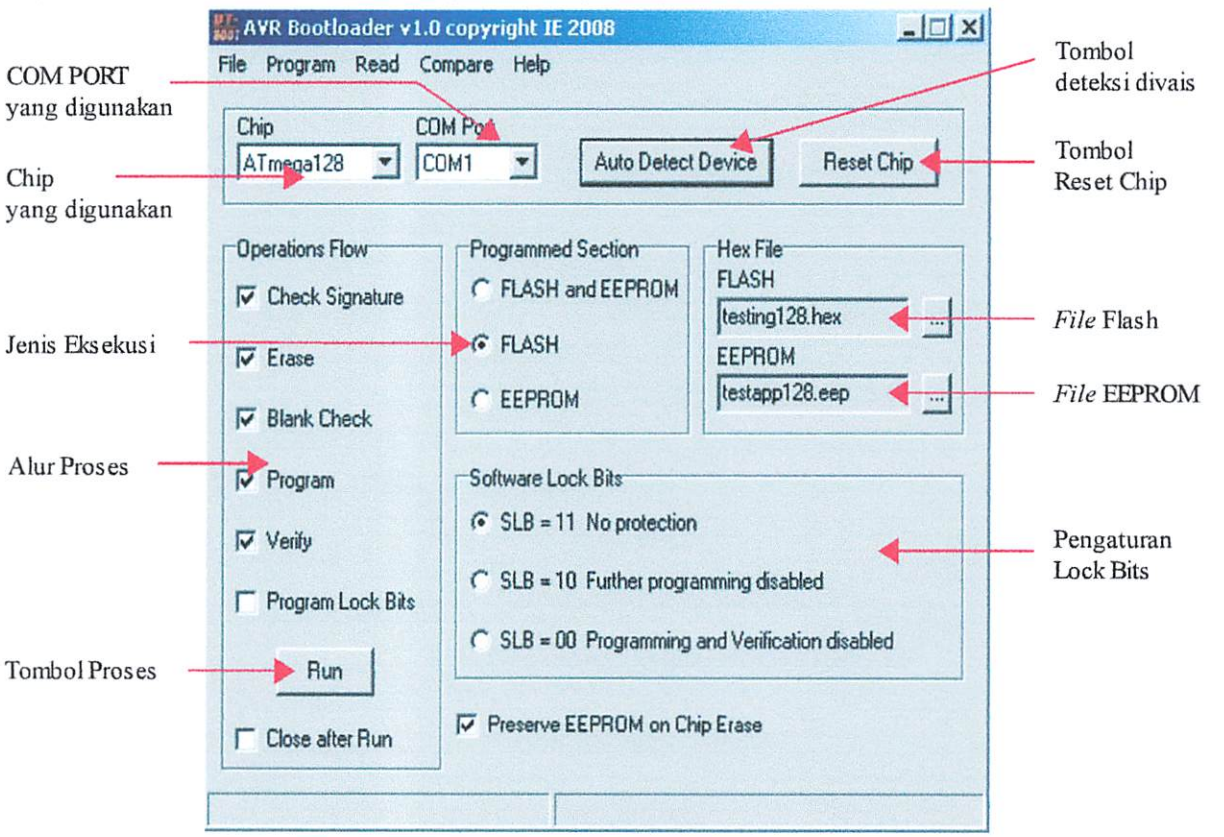


Gambar 17
Proses Membuat *File* HEX

12. Jika pada langkah di atas berhasil maka akan dihasilkan *file* HEX (*.hex). *File* HEX ini diletakkan pada *folder* yang sama dengan *project* (atau terletak dalam *folder* EXE di dalamnya) dengan nama sama dengan nama *project*. Dalam contoh ini, *file* bernama **testing128.hex**.

AVR Bootloader v1.0 adalah sebuah perangkat lunak dari Innovative Electronics yang mendukung pemrograman mikrokontroler secara *bootloader* pada DT-AVR ATMEGA128L BMS. AVR Bootloader v1.0 terdapat dalam CD/DVD yang disertakan bersama DT-AVR ATMEGA128L BMS.

1. Tampilan **AVR Botloader v1.0** serta penjelasan fungsi program.



Gambar 18
Tampilan AVR Bootloader v.1.0

2. Sebelum menjalankan program AVR Bootloader v1.0, DT-AVR ATMEGA128L BMS harus dihubungkan ke *port* USB atau serial komputer dan dinyalakan. Pada DT-AVR ATMEGA128 BMS yang menjadi divais programmer adalah mikrokontroler ATMEGA128 itu sendiri. Jalankan program AVR Bootloader v1.0, lalu tekan tombol **Auto Detect Device**. Program ini akan mendeteksi adanya modul yang dihubungkan ke komputer. Jika AVR Bootloader menampilkan pesan tidak adanya modul yang terdeteksi, tekan tombol *reset* pada modul dan jalankan kembali AVR Bootloader atau pilih IC yang digunakan secara manual.

3. Bagian **Hex File** berfungsi untuk memilih *file* yang akan diprogramkan ke mikrokontroler. Perlu diketahui bahwa AVR Bootloader v1.0 membedakan *file* yang akan diprogramkan ke Flash Memory dan EEPROM. Tekan tombol **"..." (Browse)** untuk memilih *file* yang diinginkan. Pilih *file* HEX untuk memprogram Flash Memory. *File* HEX (testing128.hex) dapat pula dibuka melalui menu **File > Load FLASH**.

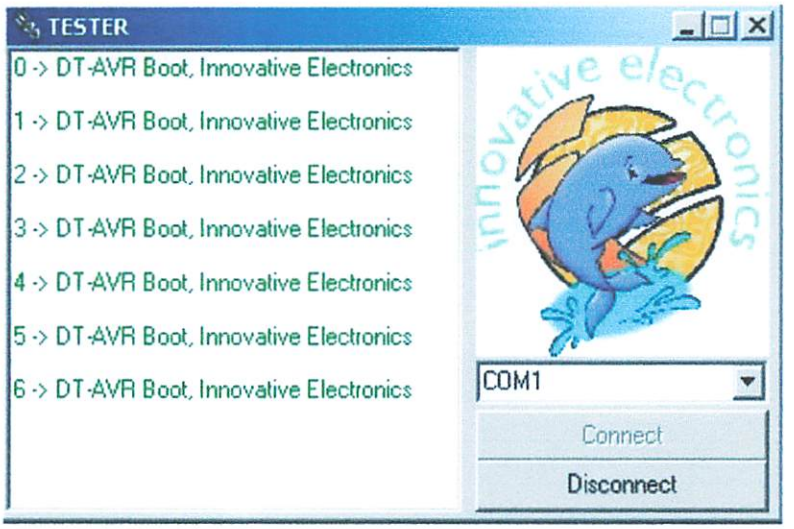
4. Pada bagian **Programmed Section**, pilih **Flash** untuk memrogram, memeriksa hasil pemrograman dan membaca Flash Memory mikrokontroler.

5. Pilih menu pada **Operations Flow** untuk memilih beberapa opsi proses yang akan dijalankan, atau dapat juga melalui menu **Program** kemudian pilih proses yang akan dilakukan. Untuk memprogram file **testing128.hex** pada contoh ini, **centang** semua **Operation Flow (Check Signature, Erase, Blank Check, Program Verify)** serta centang **Programmed Section** pada menu **Flash**. Kemudian tekan tombol **Run** untuk memprogram mikrokontroler.

6. Untuk mencoba hasil pemrograman file **testing128.hex**, jalankan program **TESTER128.exe** yang disertakan pada CD/DVD.

7. Pilih **COMPort** yang akan digunakan melalui menu combobox lalu tekan tombol "**Connect**".

8. Bila komunikasi serial berjalan dengan sukses maka akan tampil tulisan "DT-AVR Boot, Innovative Electronics" secara terus menerus dengan interval waktu sekitar 1 detik sekali.



Gambar 19
Tampilan Program TESTER128.exe

9. Untuk melihat output yang dihasilkan pada Port A, Port B, Port C, Port D, Port E (kecuali PE.0 dan PE.1), Port F, dan Port G dapat mengggunakan osiloskop, voltmeter, atau dihubungkan langsung dengan rangkaian LED atau DT-I/O LED LOGIC TESTER sehingga tampak nyala LED yang bergantian.

Selamat berinovasi!

DT-HiQ

AVR Bootloader v1.0

PETUNJUK PENGGUNAAN

Daftar Isi

1. PENDAHULUAN

AVR Bootloader v1.0 adalah sebuah perangkat lunak dari Innovative Electronics yang mendukung pemrograman mikrokontroler secara *bootloader*. AVR Bootloader v1.0 digunakan untuk memrogram mikrokontroler dengan kemampuan *self-programming* pada modul-modul DT-AVR Bootloader Micro System. Perangkat lunak berbasis Windows® ini menyediakan antarmuka yang sederhana dan mudah digunakan pengguna serta dapat dihubungkan dengan CodeVisionAVR©, BASCOM-AVR©, atau IDE mikrokontroler AVR lainya.

1.1. SPESIFIKASI AVR BOOTLOADER

Spesifikasi AVR Bootloader v1.0 adalah sebagai berikut:

- Antarmuka UART berkecepatan 115200 bps.
- Mendukung Flash, EEPROM, dan Lock Bit *Programming*.
- Mendukung format *file* Intel HEX atau EEP (untuk EEPROM).
- Kompatibel dengan Windows® XP atau Vista.

1.2. MODUL YANG DIDUKUNG

Saat ini AVR Bootloader v1.0 mendukung pemrograman modul DT-AVR Bootloader Micro System berikut ini:

- DT-AVR ATMEGA128L Bootloader Micro System.
- DT-AVR ATMEGA168 Bootloader Micro System.

1.3. PERSYARATAN SISTEM

Persyaratan minimum adalah:

- Prosesor Pentium® atau di atasnya.
- 32 MB RAM.
- Ruang kosong hard disk 1 MB.
- CD-ROM/DVD-ROM drive.
- COM Port dengan antarmuka UART RS-232 berkecepatan 115200 bps atau USB (Virtual COM Port).
- Windows® XP atau Vista.

2. AVR BOOTLOADER

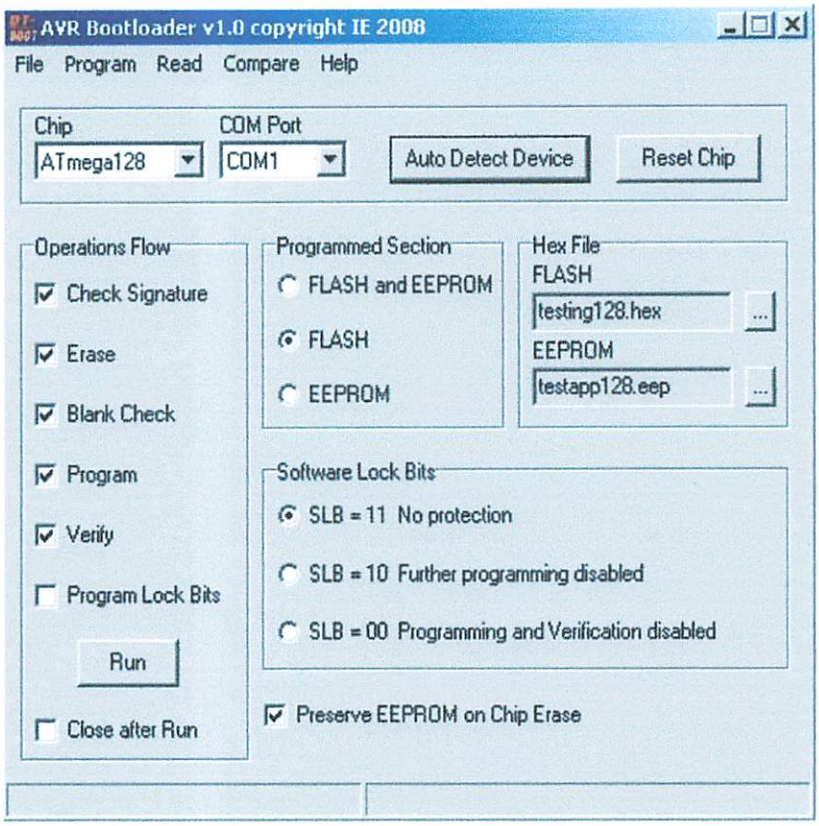
Pada bagian ini akan dijelaskan mengenai perangkat lunak yang disertakan beserta penjelasan mengenai pilihan fitur pemrograman.

2.1. MENJALANKAN AVR BOOTLOADER

Untuk menjalankan AVR Bootloader v1.0, cara-cara yang dapat dilakukan antara lain:

- Klik ganda pada *icon* **AVR Bootloader v1.0 .exe** pada Windows Explorer.
- Klik kanan pada *file* **AVR Bootloader v1.0 .exe** dan pilih "Open".

2.2. TAMPILAN PROGRAM



Gambar 1
Tampilan AVR Bootloader

2.3. MENU DAN *SHORTCUT*

2.3.1. MENU FILE

1. **Load FLASH**
   Fungsi : Membuka *file* .hex untuk memori Flash.
   Tombol *Shortcut* : Ctrl + F
2. **Load EEPROM**
   Fungsi : Membuka *file* .hex atau .eep untuk memori EEPROM.
   Tombol *Shortcut* : Ctrl + M
3. **Recent Files**
   Fungsi : Menampilkan daftar 10 *file* yang terakhir dibuka.
4. **Exit**
   Fungsi : Keluar dari perangkat lunak.
   Tombol *Shortcut* : Ctrl + X

2.3.2. MENU PROGRAM

1. **Erase**
   Fungsi : Menghapus IC target.
   Tombol *Shortcut* : Ctrl + E
2. **Blank Check**
   Fungsi : Memeriksa apakah IC target kosong.
   Tombol *Shortcut* : Ctrl + B

3. **FLASH**
   Fungsi : Menulis kode yang tersimpan pada Hex File - Flash ke memori Flash target.
   Tombol *Shortcut* : Ctrl + P

4. **EEPROM**
   Fungsi : Menulis kode yang tersimpan pada Hex File - EEPROM ke memori EEPROM target.
   Tombol *Shortcut* : Ctrl + O

5. **Lock Bits**
   Fungsi : Memunculkan jendela dialog untuk pengaturan Lock Bits.
   Tombol *Shortcut* : Ctrl + L

6. **All**
   Fungsi : Menjalankan serangkaian instruksi sesuai dengan pilihan pada *Operation Flow.*
   Tombol *Shortcut* : F9

2.3.3. MENU READ

1. **FLASH**
   Fungsi : Membaca memori Flash IC target dan menyimpannya ke dalam *file.*
   Tombol *Shortcut* : Ctrl + R

2. **EEPROM**
   Fungsi : Membaca memori EEPROM IC target dan menyimpannya ke dalam *file.*
   Tombol *Shortcut* : Ctrl + Q

3. **Chip Signature**
   Fungsi : Membaca kode *chip signature* IC target.
   Tombol *Shortcut* : Ctrl + C

4. **Lock Bits**
   Fungsi : Membaca pengaturan Lock Bits IC target.
   Tombol *Shortcut* : Ctrl + T

5. **Fuse Bits**
   Fungsi : Membaca pengaturan Fuse Bits IC target.
   Tombol *Shortcut* : Ctrl + U

6. **Bootloader Version**
   Fungsi : Membaca versi perangkat lunak *bootloader* yang tertanam di IC target.
   Tombol *Shortcut* : Ctrl + K

2.3.4. MENU COMPARE

1. **FLASH**
   Fungsi : Membandingkan isi Hex File - Flash dan memori Flash IC target.
   Tombol *Shortcut* : Ctrl + V

2. **EEPROM**
   Fungsi : Membandingkan isi Hex File - EEPROM dan memori EEPROM IC target.
   Tombol *Shortcut* : Ctrl + G

2.3.5. MENU HELP

- **About**

  Fungsi : Menampilkan versi perangkat lunak AVR Bootloader dan *link* ke *website* Innovative Electronic.

2.4. PANEL OPERATION FLOW

Panel Operation Flow berfungsi untuk menentukan perintah-perintah apa saja yang secara otomatis akan dilakukan saat pemrograman menggunakan menu **Program - All** atau penekanan tombol **"Run"**. Perintah-perintah yang dapat dipilih adalah *Check Signature, Erase, Blank Check, Program, Verify,* dan *Program Lock Bits.*

Untuk perintah *Erase,* jika **Preserve EEPROM on Chip Erase** dicentang/dipilih, maka memori yang akan dihapus saat perintah *Erase* dilaksanakan adalah memori Flash saja. Sebaliknya, jika **Preserve EEPROM on Chip Erase** tidak dicentang/dipilih, maka memori yang akan dihapus saat perintah *Erase* dilaksanakan adalah memori Flash dan EEPROM.

Untuk perintah *Program* dan *Verify,* jika pada panel **Programmed Section** yang dipilih adalah "FLASH and EEPROM" maka perintah *Program* dan *Verify* akan dilakukan pada kedua bagian memori tersebut. Sebaliknya jika hanya 1 saja yang dipilih (FLASH saja atau EEPROM saja), maka perintah *Program* dan *Verify* hanya akan dilakukan pada bagian memori terpilih.
Sebelum memilih perintah *Program* dan *Verify* pastikan bahwa *file* yang akan diproses sudah ditentukan sebelumnya. *File* yang akan dituliskan ke Flash atau ke EEPROM akan ditampilkan pada panel **Hex File.**

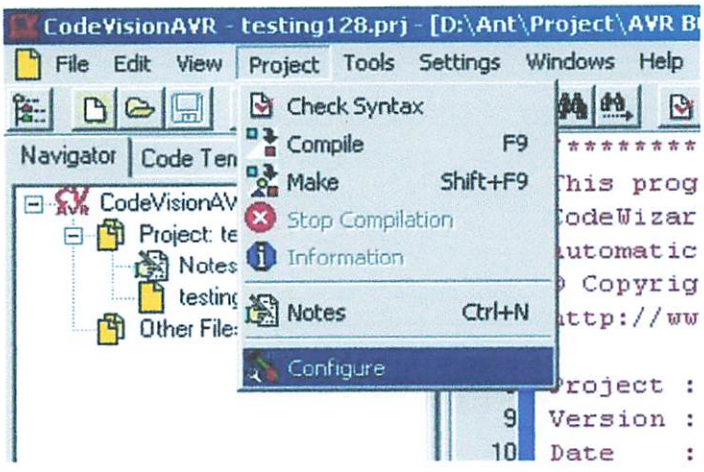
Untuk perintah *Program Lock Bits,* tingkat pengamanan yang akan diberikan pada kode program yang tersimpan pada memori Flash ditentukan melalui pilihan pada panel **Software Lock Bits.** Jika tingkat pengamanan yang dipilih adalah "No Protection" maka penambahan kode pada memori Flash atau pembacaan isi memori Flash tanpa harus melakukan perintah Erase masih dimungkinkan.
Pada pengamanan tingkat selanjutnya, penambahan kode pada memori Flash tidak bisa dilakukan lagi, kecuali kita terlebih dahulu melakukan perintah Erase.
Pada tingkat tertinggi, penambahan kode pada memori Flash dan pembacaan isi memori Flash tidak bisa dilakukan lagi, kecuali kita terlebih dahulu melakukan perintah *Erase.*

Setelah melakukan seluruh perintah pada Operation Flow dengan memilih menu **Program - All** atau menekan tombol **"Run"**, secara otomatis AVR Bootloader v1.0 akan melakukan **Reset Chip** sehingga modul DT-AVR Bootloader Micro System akan segera keluar dari *bootloader* dan menjalankan program aplikasi yang telah diprogramkan.
Akan tetapi jika pilihan pemrograman dilakukan secara manual (misalkan dengan menekan tombol **Auto Detect Device,** kemudian memilih menu **Read - Chip Signature** dan seterusnya) maka modul DT-AVR Bootloader Micro System akan tetap berada pada bagian *bootloader.* Oleh karena itu tombol **Reset Chip** harus ditekan agar modul DT-AVR Bootloader Micro System keluar dari *bootloader* dan menjalankan program aplikasi yang telah diprogramkan.

Centang/pilih **Close after Run** jika kita ingin AVR Bootloader v1.0 otomatis ditutup setelah melakukan seluruh perintah pada Operation Flow. Fungsi ini cukup berguna jika nantinya AVR Bootloader v1.0 ini ditautkan pada perangkat lunak IDE AVR lainya seperti misalnya CodeVisionAVR ataupun BASCOM-AVR.

3. CONTOH PENGATURAN UNTUK CODEVISIONAVR

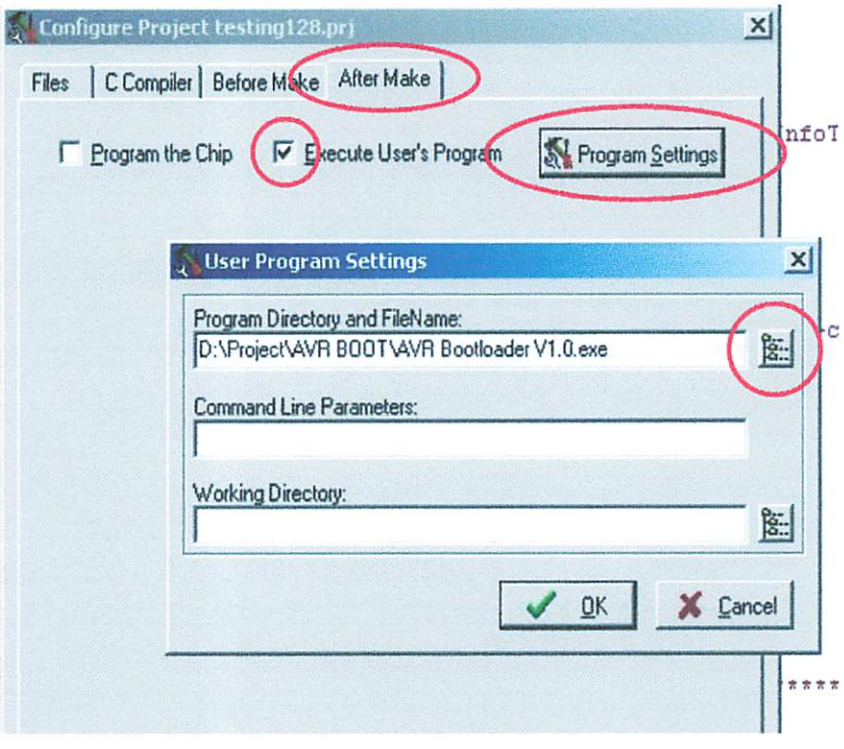
Cara untuk menggabungkan AVR Bootloader v1.0 dengan CodeVisionAVR adalah dengan mengatur agar AVR Bootloader v1.0 secara otomatis dibuka setelah melakukan proses kompilasi program. Pengaturan ini dapat dilakukan melalui menu **Project → Configure.**



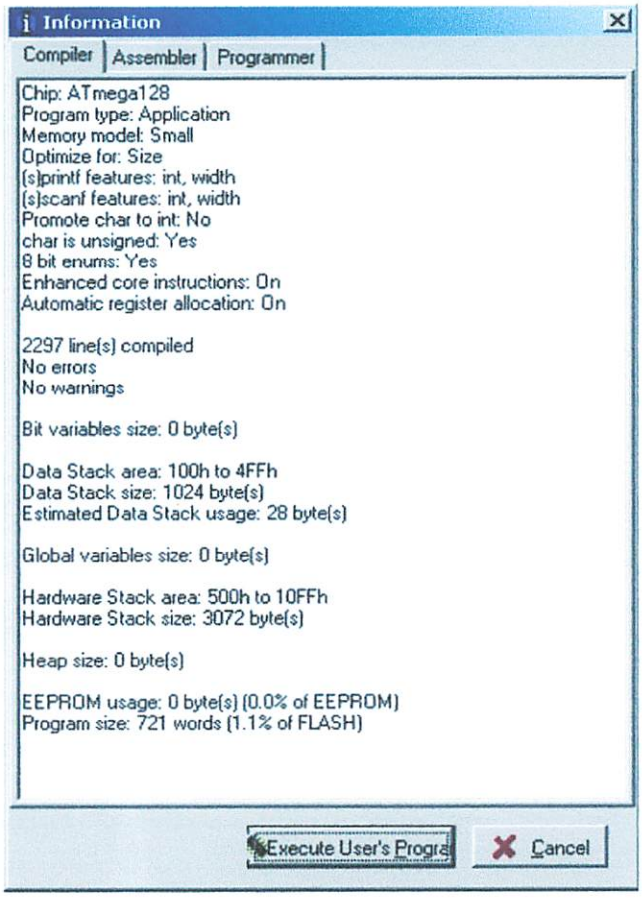
Gambar 2
Menu Konfigurasi *Project*

Pilih tab **"After Make"** kemudian centang/pilih **"Execute User's Program"**. Setelah itu tekan tombol **Program Settings** untuk memunculkan jendela dialog **"User Program Settings"** seperti ditampilkan pada Gambar 3.

Atur pilihan **"Program Directory and Filename"** agar menunjukan pada lokasi perangkat lunak AVR Bootloader v1.0. Setelah itu tekan tombol **"OK"**.



Gambar 3
Jendela Dialog User Program Settings



Gambar 4
Jendela Dialog Informasi Setelah Proses Make Berhasil

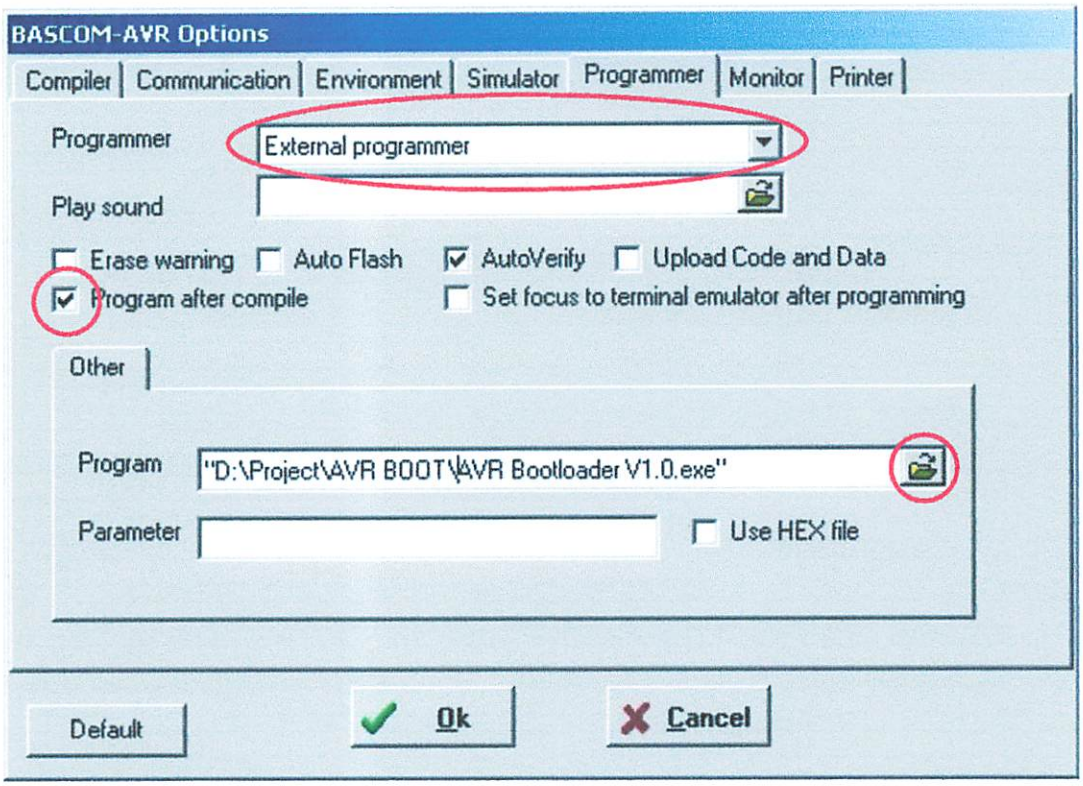
Setelah melakukan langkah-langkah tersebut, jika proses **Compile** dan **Make** (misalkan dengan menekan tombol **shift + F9**) berhasil dilakukan, maka akan muncul jendela dialog informasi seperti yang ditampilkan pada Gambar 4.

Jika proses **Make** berhasil dan jendela dialog seperti pada Gambar 4 muncul, maka kita cukup menekan tombol Enter pada *keyboard* atau menekan tombol **"Execute User's Program"** pada jendela dialog informasi untuk menjalankan AVR Bootloader v1.0. Setelah AVR Bootloader v1.0 terbuka, kita cukup menekan tombol Enter pada *keyboard* atau menekan tombol **"Run"** untuk memulai pemrograman.

Jangan lupa untuk mencentang pilihan **"Close after Run"** agar AVR Bootloader v1.0 otomatis tertutup setelah seluruh proses pemrograman selesai.

4. CONTOH PENGATURAN UNTUK BASCOM-AVR

Cara untuk menggabungkan AVR Bootloader v1.0 dengan BASCOM-AVR adalah dengan mengatur agar AVR Bootloader v1.0 secara otomatis dibuka setelah melakukan proses kompilasi program. Pengaturan ini dapat dilakukan dengan mengatur tipe *programmer* yang digunakan melalui menu **Options → Programmer.**



Gambar 5
Menu Pengaturan *Programmer*

Pilih tipe *programmer* menjadi **"External programmer"** kemudian centang pilihan **"Program after compile"** supaya AVR Bootloader v1.0 otomatis terbuka setelah proses kompilasi berhasil.
Atur pula lokasi *file* agar menunjukan pada lokasi perangkat lunak AVR Bootloader v1.0. Pada contoh di gambar 5, dimisalkan lokasi *file* berada di **"D:\Project\AVR BOOT\AVR Bootloader V1.0.exe"**.

Setelah melakukan langkah-langkah tersebut, jika proses **Compile** dan **Make** (misalkan dengan menekan tombol **F7**) berhasil dilakukan, maka AVR Bootloader v1.0 akan secara otomatis dijalankan. Setelah AVR Bootloader v1.0 terbuka, kita cukup menekan tombol Enter pada *keyboard* atau menekan tombol **"Run"** untuk memulai pemrograman.

PANDUAN PRAKTIKUM DASAR
MIKROKONTROLER KELUARGA AVR

MENGGUNAKAN

MINIMUM SISTEM AVR DAN PERIPHERAL I/O BOARD

1. PENDAHULUAN

1.1 TUJUAN

Tujuan keseluruhan praktikum ini adalah agar praktikan mampu memrogram mikrokontroler AVR menggunakan fitur yang dimiliki oleh minimum sistem AVR dengan bantuan peripheral I/O board

1.2 RUANG LINGKUP

Percobaan ini mencakup percobaan menggunakan minimum sistem AVR, *peripheral I/O board*, dengan bantuan *Personal Computer*(PC) dan perangkat lunak CodeVisionAVR

1.3 SISTEMATIKA PANDUAN PRAKTIKUM DASAR MIKROKONTROLER KELUARGA AVR

Panduan Praktikum Dasar Mikrokontroler Keluarga AVR ini terbagi dalam 9 bab. Adapun garis besar masing – masing bab adalah sebagai berikut :

a. Bab I Pendahuluan
   - Membahas mengenai tujuan praktikum, ruang lingkup, serta sistematika Panduan Praktikum Dasar Mikrokontroler Keluarga AVR.

b. Bab II Penjelasan Perangkat Praktikum
   - Menjelaskan mengenai fitur – fitur minimum sistem AVR, *peripheral I/O board*, cara menggunakan perangkat keras, cara menggunakan perangkat lunak.

c. Bab III Input/Output dasar dengan port mikrokontroler
   - Praktikum mengenai penggunaan port mikrokontroler sebagai *Input/Output* dasar.

d. Bab IV Interupsi
   - Praktikum mengenai sistem interupsi eksternal pada mikrokontroler keluarga AVR

e. Bab V *Timer/Counter*
   - Praktikum mengenai sistem Timer/Counter pada mikrokontroler keluarga AVR

f. Bab VI Akses LCD
   - Praktikum mengenai penggunaan port mikrokontroler sebagai kontrol untuk menampilkan karakter pada LCD

g. Bab VII *Scanning 7Segment*
   - Praktikum mengenai teknik *scanning* pada tampilan *7segment*.

h. Bab VIII Scanning Led Matriks
   - Praktikum mengenai teknik scanning pada tampilan *Led Matriks.*

i. Bab IX Motor Stepper
   - Praktikum mengenai kontrol terhadap *motor stepper.*

2. PENJELASAN PRANGKAT PRAKTIKUM

2.1 PERANGKAT KERAS

Perangkat keras yang digunakan dalam setiap praktikum adalah minimum sistem AVR dan juga peripheral I/O board sebagai indikator input/output.

Minimum Sistem AVR

Sesuai dengan namanya, minimum sistem AVR merupakan minimum sistem yang memiliki spesifikasi sebagai berikut :

- Berbasis keluarga mikrontroler AVR (lebih spesifik yaitu ATMega 32 dan DT-AVR 128L *Bootloader Microsystem*(BMS))
- 32 input/output bidirectional pada ATMega 32 dan 64 input/output bidirectional pada DT – AVR 128L BMS

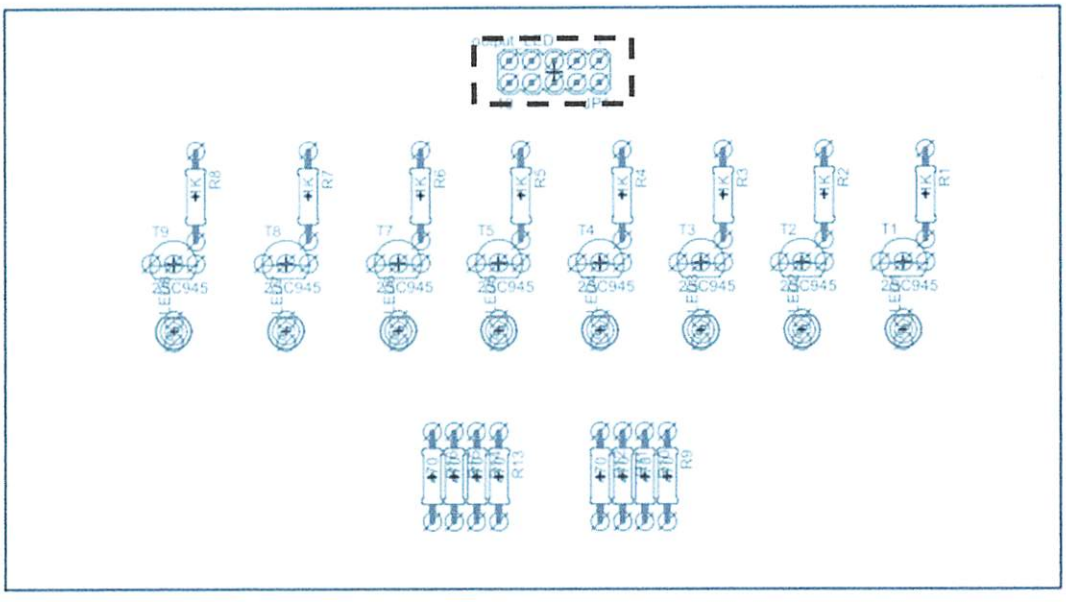
Peripheral I/O Board

Merupakan kumpulan modul yang memiliki spesifikasi sebagai berikut :

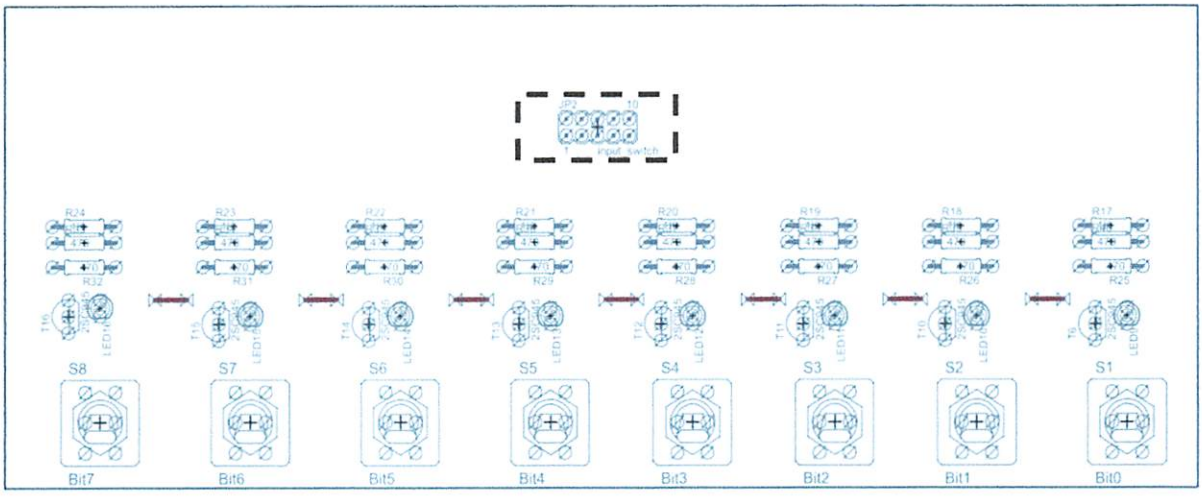
- 8 LED sebagai *output*. Konektor LED terdapat pada output_LED port(konektor yang bertuliskan output_LED)
- 8 *Toggle Switch* sebagai *input*. Konektor *toggle switch* terdapat pada input switch *port*(konektor yang bertuliskan input_switch)
- 2 *seven segment* sebagai *scanning output*. Konektor datanya terdapat pada 7segmentPIN *port*(konektor yang bertuliskan 7segmentPIN). Konektor pemilihnya terdapat pada control *port*(konektor yang bertuliskan control)
- Sebuah *Dot Matriks* LED 5x7 sebagai *scanning output*. Konektor datanya terdapat pada DotMatriks *port*(konektor yang bertuliskan DotMatriks). Konektor kontrol terdapat pada IC4017 *port*(konektor yang bertuliskan IC4017)
- Sebuah *Stepper Motor* sebagai *output*. Konektornya terdapat pada masing – masing A, B, C, D, VCC, dan GND *port*.
- Sebuah LCD modul sebagai *output*. Konektor datanya terdapat pada *port* LCD.

2.2 Cara menghubungkan perangkat keras.

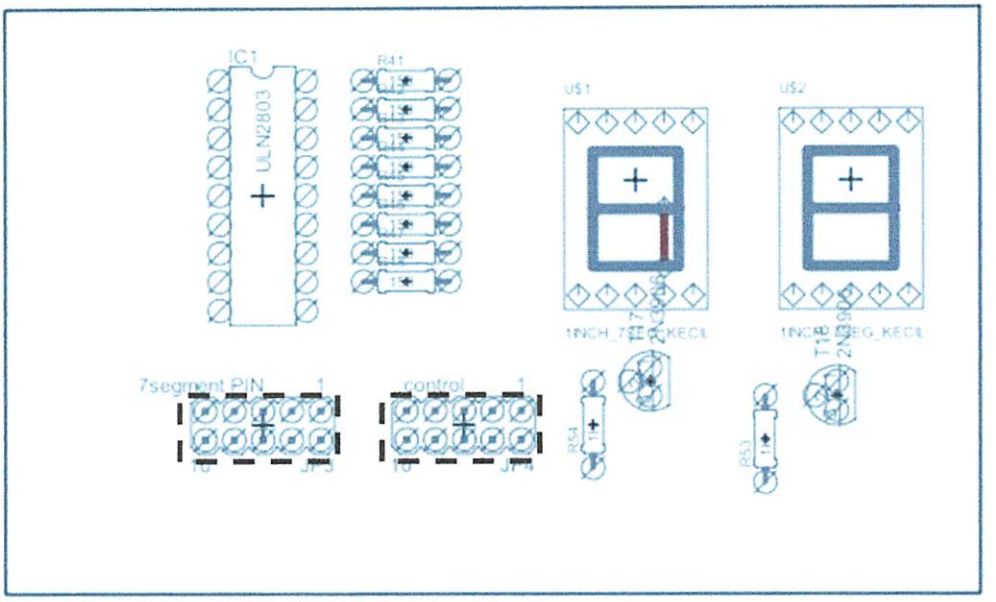
Hubungan minimum sistem AVR dan peripheral I/O board berkisar pada port pin mikrokontroler yakni PORTA, PORTB, PORTC, dan PORTD (PORTE, PORTF, PORTF, PORTG pada ATMega 128L) serta port – port seperti output_LED, input switch, 7segmentPIN, DotMatriks, control, IC4017, A, B, C, D, VCC, GND, dan LCD.



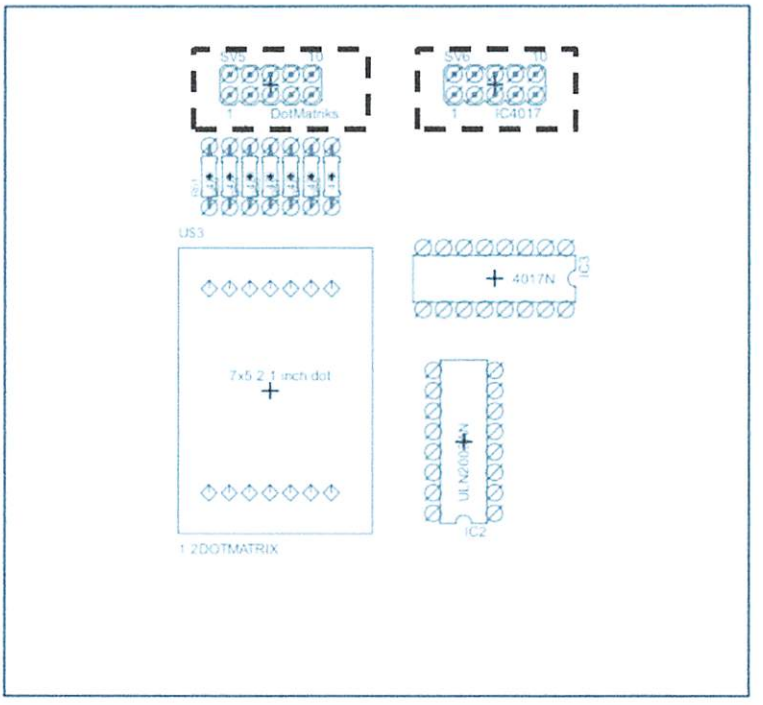
Gambar 1. Konektor output_LED ditandai pada bagian putus - putus



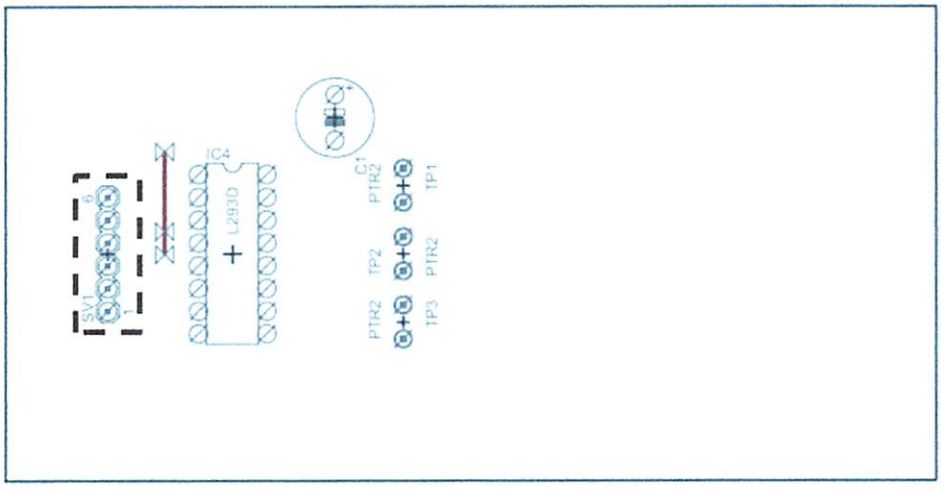
Gambar 2. Konektor switch input ditandai pada bagian putus - putus



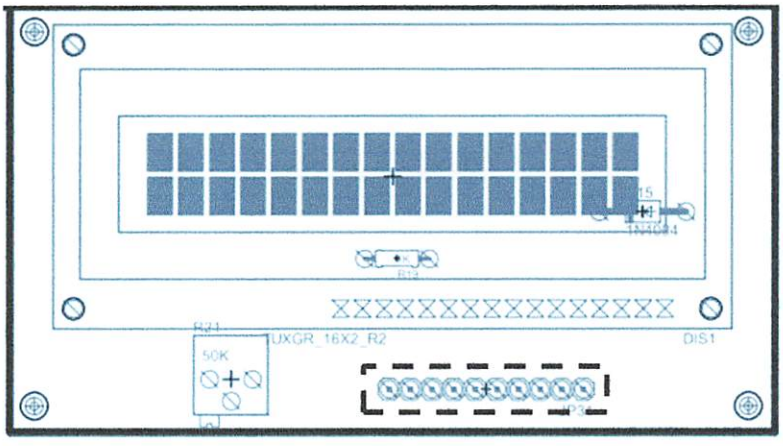
Gambar 3. Konektor 7segment PIN dan control masing – masing ditandai pada bagian putus - putus



Gambar 4. Konektor DotMatriks dan IC4017 masing – masing ditandai pada bagian putus - putus



Gambar 5. Konektor pada bagian motor stepper(A, B, C, D, GND, VCC) masing–masing ditandai pada bagian putus - putus



Gambar 6. Konektor pada LCD ditandai pada bagian putus-putus

Tipe kabel yang harus dibuat untuk menghubungkan minimum sistem AVR dengan modul – modul pada *peripheral I/O board* memerlukan bahan – bahan sebagai berikut :

- 7 set black housing 2x1 pin
- Kabel pelangi secukupnya

Kabel konektor ini bersifat universal untuk menghubungkan konektor pin, sehingga dapat digunakan pada konektor minimum sistem AVR dengan *peripheral I/O board* sesuai kebutuhan

Gambar 7. Contoh kabel konektor dengan black housing 2x1 pin dilihat dari tampak samping pada sebelah kiri dan tampak atas pada sebelah kanan

Pastikanlah mengecek koneksi kabel yang dibuat dengan set black housing 2x1 pin dan kabel pelangi dengan menggunakan multimeter antara satu sisi dengan sisi yang lain agar tidak terjadi korsleting atau gagal sambungan dalam pemasangan.

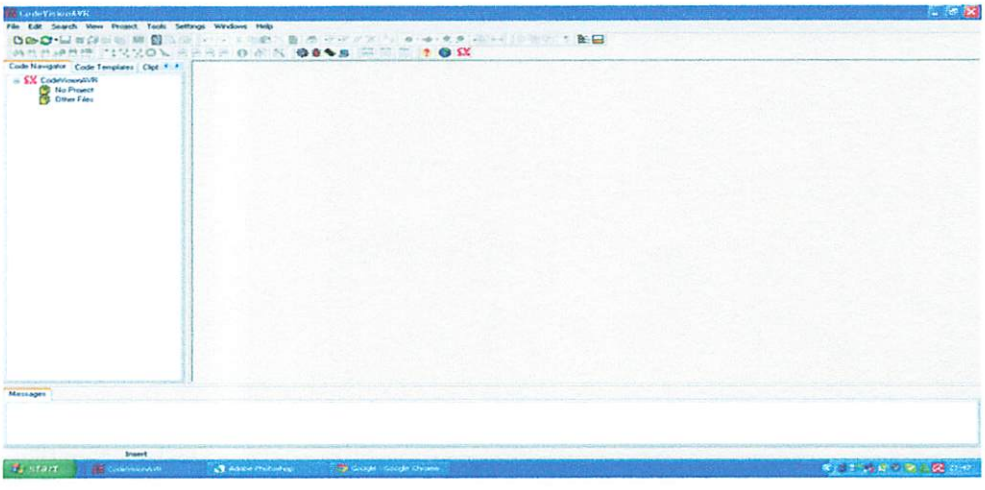
2.3 PERANGKAT LUNAK

Perangkat lunak yang digunakan adalah **CodeVisionAVR** versi evaluasi yang dapat di unduh secara gratis dan bebas pada *website* www.hpinfotech.ro . CodeVision AVR digunakan sebagai editor, kompilasi program, serta mendukung fitur download program melalui ISP port. Program ini menyediakan pembangkit kerangka program, hal ini sangat memudahkan programer untuk membuat program atau untuk mengenal dasar bahasaC yang dimana telah didukung dengan berbagai *library* pemrograman, yang dikenal dengan CodeWizardAVR.

2.4 CARA MENGGUNAKAN PERANGKAT LUNAK

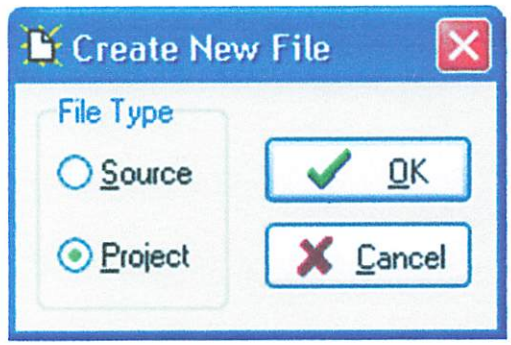
Untuk memulai project pada CodeVisionAVR tak harus melakukan coding dari awal, cukup dengan mengatur jendela wizard maka program akan tergenerate secara otomatis. Untuk cara memulai project pada CodeVisionAVR dapat dilihat pada gambar berikut.

a. Membuka CodeVision AVR pada windows



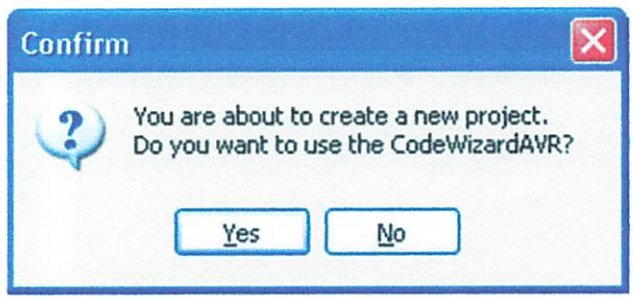
Gambar 8. Tampilan awal CodeVisionAVR

b. Selanjutnya untuk memulai pilih menu "File -> New" maka akan tampil seperti jendela dibawah.



Gambar 9. Tampilan menu kotak dialog "File -> New"

c. Pilih "Project" lalu "OK" selanjutnya akan muncul jendela konfirmasi seperti gambar dibawah



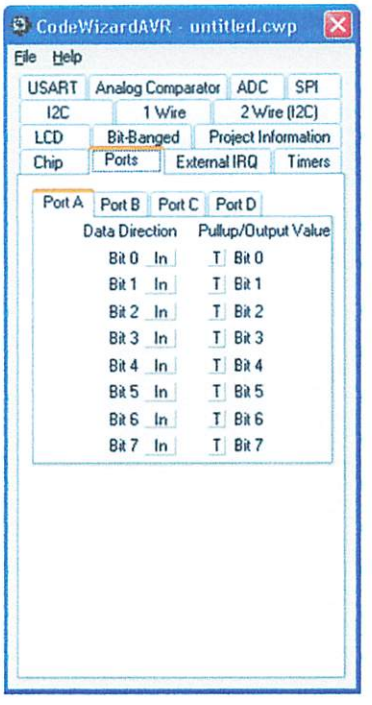
Gambar 10. Tampilan menu kotak dialog konfirmasi CodeWizar

d. Pilih "Yes" lalu akan muncul CodeWizardAVR yang dimana disini merupakan pengaturan awal program seperti apa yang akan dibuat yang nantinya akan digenerate menjadi script program bahasa C mikrokontroler

1. Jendela untuk pengaturan Chip mikrokontroler dan frekuensi kristal yang dipakai

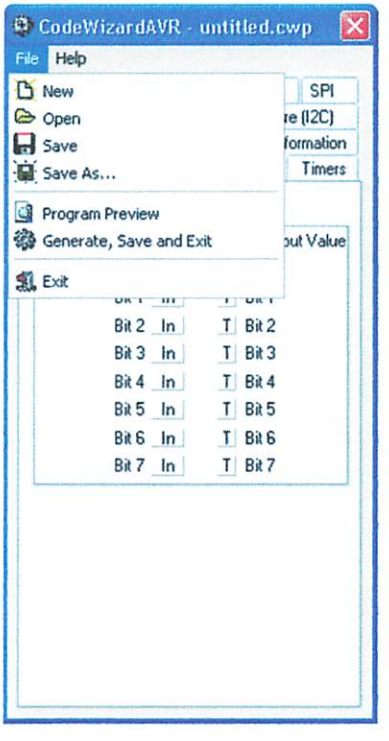
Gambar 11. Tampilan jendela pengaturan frekuensi kristal

2. Pengaturan PORT I/O untuk dijadikan sebgai keluaran atau masukan serta pengaturan lainnya tergantung program apa yang akan dibuat



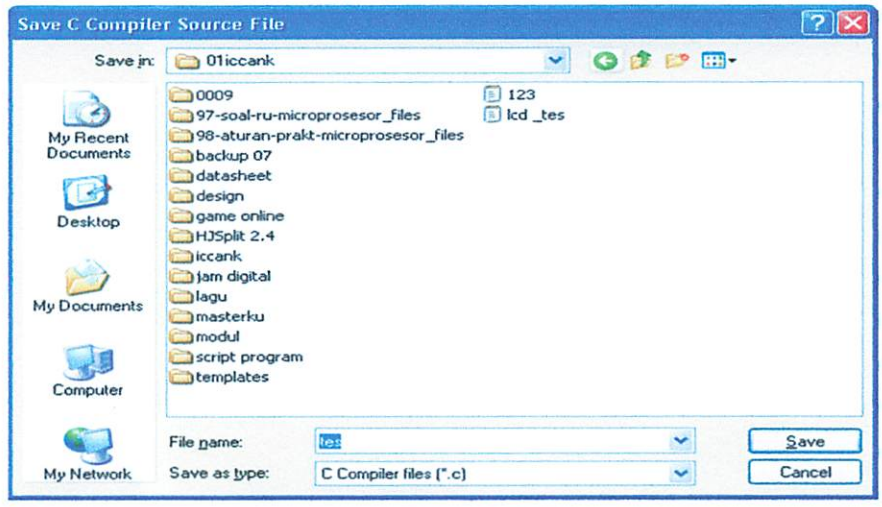
Gambar 12. Tampilan gambar pengaturan Port I/O

e. Setelah semuanya pengaturan beres silahkan pilih "File -> Generate, Save and Exit" untuk menyimpan tiga file program utama Project



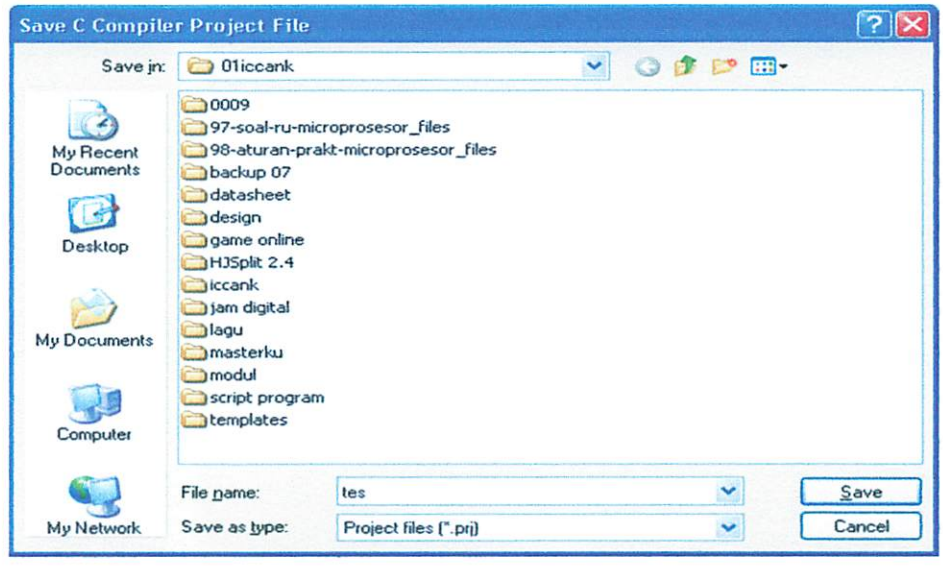
Gambar 13. Tampilan menu "File"

1. Pertama File script program dalam bahasa C dengan Extention ".C"



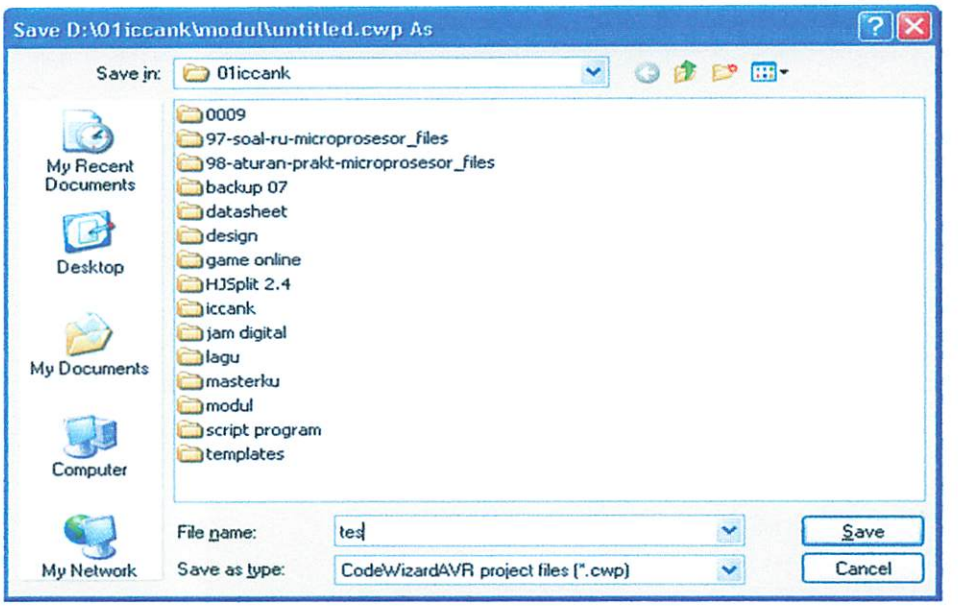
Gambar 14. Tampilan save pada file .C

2. Kemudian File Project CodeVision AVR itu sendiri dengan Extention ".prj"



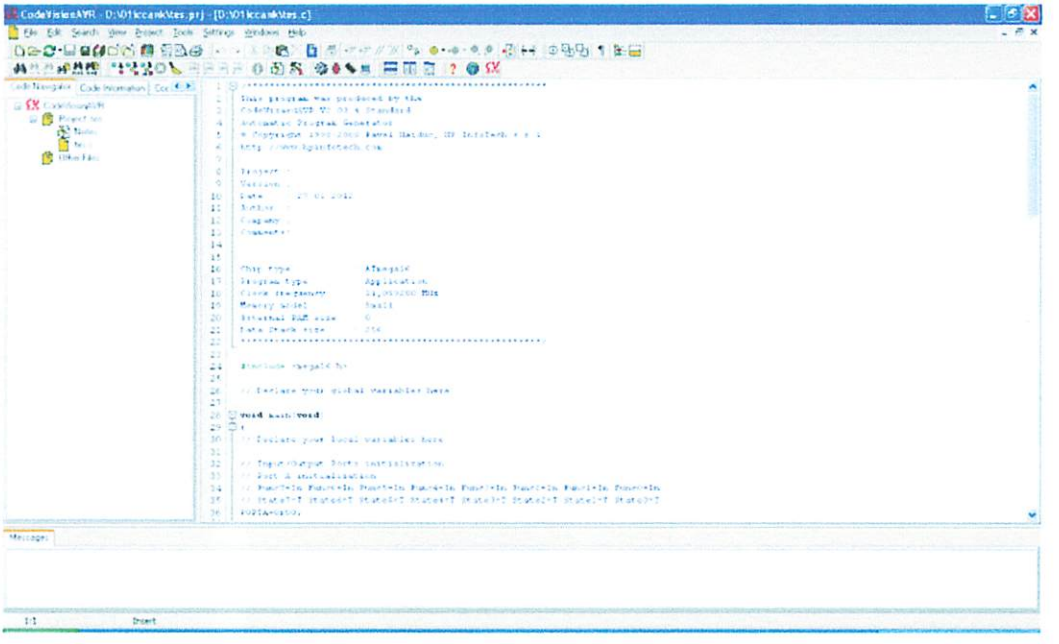
Gambar 15. Tampilan save pada file .prj

3. Dan yang terakhir File CodeWizardAVR berupa file pengaturan tadi dengan Extention ".cwp".



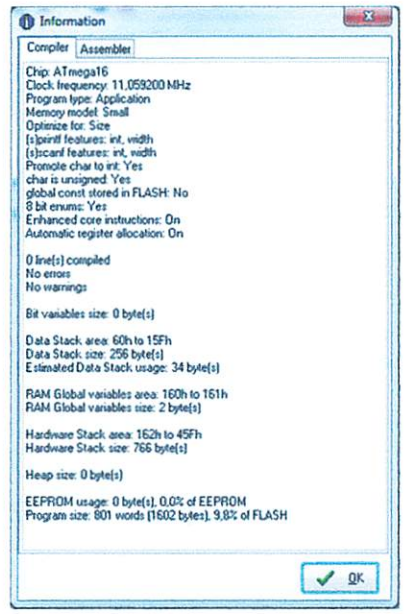
Gambar 16. Tampilan save pada file .cwp

f. Buat program bahasa C pada jendela hasil pengaturan CodeWizard tadi



Gambar 17. Jendela editor program hasil pengaturan CodeWizard

g. Untuk mengompile program tekan "Ctrl + F9" maka akan muncul jendela pemberitahuan bahwa program yang dibuat telah dicompile dan siap untuk dimasukkan ke dalam chip mirokontroler. Pada jendela ini juga dapat dilacak apakah terjadi kesalahan kode pemrograman atau tidak.



Gambar 18. Hasil kompilasi program

3. INPUT/OUTPUT DASAR DENGAN PORT MIKROKONTROLER

3.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program input dan output menggunakan port mikrokontroler dari minimum sistem AVR dengan cara mengakses portD dan port B secara per *bit* dan per *byte*

3.2 DASAR TEORI

Input/output merupakan dasar dari pemahaman mikrokontroler. Setiap kita mengakses port mikrokontroler baik dalam mode pemrograman apapun, hal ini tak dapat dipisahkan dari input/output. Input berarti menginisialisasi port mikrokontroler sebagai port yang menerima masukan dari piranti luar. Output berarti menginisialisasi port mikrokontroler sebagai keluaran hasil kode pemrograman. Akses data pada input/output dapat berupa per bit ataupun per byte tergantung kode pemrograman yang dibuat.

3.3 PROSEDUR PERCOBAAN

a) Percobaan I : PortD sebagai Output

Persiapan :

- Hubungkan portD mikrokontroler dengan output_LED port pada peripheral I/O
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan PortC sebagai output dengan keadaan awal keluaran low "0"
- Ketikan program berikut pada program utama

I. Program 1:

```
while (1)
        {
        // Place your code here
        PORTC=0xFF; //lampu menyala
        }
```

Pada program diatas apabila dijalankan dan diunduh pada mikrokontroler maka akan menyebabkan lampu menyala. Perintah "PORTC=0xFF;"

berfungsi mengakses port C mikrokontroler secara 8 bit data sekaligus. Karena itu data perintah yang dikeluarkan berupa data biner 8 bit yang disusun menjadi heksadesimal

II. Program 2:

```
while (1)
      {
      // Place your code here
      PORTC.0=1;
      PORTC.1=0;
      PORTC.2=1;
      PORTC.3=1;
      PORTC.4=1;
      PORTC.5=0;
      PORTC.6=0;
      PORTC.7=1;
      }
```

Pada program ke 2 apabila dijalankan dan diunduh ke mikrokontroler akan menyebabkan lampu menyala dengan kondisi "10011101" mulai dari MSB ke LSB. Program ini mengakses port c yang diinisialisasikan sebagai output per bit. Jadi setiap bitnya ditandai dengan PORTC.0 sebagai bit 0/LSB(Less Significant Bit) hingga ke bit PORTC.7 sebagai bit 7/MSB(Most Significant Bit)

Kesimpulan yang dapat diambil adalah berdasarkan program diatas maka perintah PORTX digunakan untuk mengontrol outputan yang akan diberikan kepada mikrokontroler pada setiap port nya. PORTX diikuti dengan data per byte sehingga bilangannya dapat berupa kombinasi biner atau dapat juga berupa heksadesimal. Sedangkan untuk mengakses per bit port mikrokontroler digunakan perintah PORTX.n dengan n sebagai urutan bit - mulai terkecil(0)-hingga terbesar(7) – dan diikuti pemberian data berupa logika yang ingin dikeluarkan apakah itu HIGH(1) atau LOW(0).

b) Percobaan II : PortB sebagai Input

- Hubungkan portB mikrokontroler dengan input_switch port dan portD mikrokontroler dengan output_LED port pada peripheral I/O
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port A sebagai input dengan keadaan awal inputan low "0"(togle), dan Port C sebagai output dengan keadaan awal outputan low "0"
- Ketikan program berikut pada program utama:

I. Program I

```
#include <mega32a.h>
#include <delay.h>
#define Saklar  PINA
#define Saklar0 PINA.0
#define Saklar1 PINA.1
#define Saklar2 PINA.2
#define Saklar3 PINA.3
#define Saklar4 PINA.4
#define Saklar5 PINA.5
#define Saklar6 PINA.6
#define Saklar7 PINA.7

// Declare your global variables here
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
DDRC=0xFF;
```

```
PORTD=0x00;
DDRD=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;
ADCSRA=0x00;
SPCR=0x00;
TWCR=0x00;
while (1)
    {
    // Place your code here
    PORTC = Saklar;
    }
}
```

Pada program pertama, input yang dibaca secara 8 bit data/1 control words, kemudian hasilnya dikeluarkan pada portC mikrokontroler secara 8 bit juga. Jadi setiap perubahan bit inputan pada portA akan mempengaruhi keadaan output mikrokontroler pada portC

II. Percobaan Kedua

```
#include <mega32a.h>
#include <delay.h>
#define Saklar  PINA
#define Saklar0 PINA.0
#define Saklar1 PINA.1
#define Saklar2 PINA.2
#define Saklar3 PINA.3
#define Saklar4 PINA.4
#define Saklar5 PINA.5
#define Saklar6 PINA.6
#define Saklar7 PINA.7

// Declare your global variables here
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
DDRC=0xFF;
PORTD=0x00;
DDRD=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
```

```
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;
ADCSRA=0x00;
SPCR=0x00;
TWCR=0x00;
while (1)
    {
    // Place your code here
    If (Saklar0==1); delay_ms(200); PortC.0=1;
    If (Saklar1==1); delay_ms(200); PortC.1=1;
    If (Saklar2==1); delay_ms(200); PortC.2=1;
    If (Saklar3==1); delay_ms(200); PortC.3=1;
    If (Saklar4==1); delay_ms(200); PortC.4=1;
    If (Saklar5==1); delay_ms(200); PortC.5=1;
    If (Saklar6==1); delay_ms(200); PortC.6=1;
    If (Saklar7==1); delay_ms(200); PortC.7=1;
    }
}
```

Pada program kedua ini, inputan dibaca secara per bit dan masing – masing dikeluarkan pada bit output yang diinginkan. Jadi setiap bit inputan akan berpengaruh terhadap bit output yang diinginkan.

Kesimpulan yang dapat diambil adalah berdasarkan program diatas maka perintah PINX digunakan untuk mengontrol inputan yang akan dibaca mikrokontroler pada setiap port nya. PINX diikuti dengan pembacaan data per byte sehingga bilangannya dapat berupa kombinasi biner atau dapat juga berupa heksadesimal. Sedangkan untuk mengakses per bit port mikrokontroler digunakan perintah PINX.n dengan n sebagai urutan bit - mulai terkecil(0)-hingga terbesar(7) – dan diikuti pemberian logika pembacaan data yang ingin dibaca apakah itu HIGH(1) atau LOW(0).

4. *INTERRUPT*

4.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program *interrupt* eksternal(INT0, INT1, INT2) menggunakan port mikrokontroler dari minimum sistem AVR

4.2 DASAR TEORI

Mikrokontroler AVR mempunyai masukan interupsi eksternal sebanayk 3 buah yaitu INT0, INT1, dan INT2. Interupsi akan menyulut, sekalipun pin INT0..2 dikonfigurasikan sebagai keluaran. Fitur ini menyediakan suatu cara membangkitkan interupsi pada perangkat lunak. Interupsi eksternal dapat disulut oleh transisi naik atau turun atau level rendah(kecualai INT2 yang hanya dapat disulut dengan transisi naik atau turun). Fitur ini diseting oleh MCU Control Register dan MCU Control and Status Register. Saat interupsi eksternal diaktifkan dan dikonfigurasikan sebagai level trigger(hanya pada INT0 dan INT1), interupsi akan menyulut selama pin ditahan dalam logika rendah.

4.3 PROSEDUR PERCOBAAN

c) Percobaan III : INTERUPSI EKSTERNAL

Persiapan :

- Hubungkan portC mikrokontroler dengan output_LED port pada peripheral I/O dan portD mikrokontroler dengan input_switch
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port D sebagai input dengan keadaan awal inputan low "0". Setting INT0 pada keadaan terpilih dengan pilihan transisi yang diinginkan low level.
- Ketikan program berikut pada program utama :

```
#include <mega32a.h>
#include <delay.h>
interrupt [EXT_INT0] void ext_int0_isr(void)
{
// Place your code here
PORTC=0x55;
```

```
delay_ms(500);
PORTC=0x00;
delay_ms(500);
PORTC=0xFF;
delay_ms(500);
PORTC=0x00;
}
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
DDRC=0xFF;
PORTD=0x00;
DDRD=0x00;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

// External Interrupt(s) initialization
// INT0: On
```

```
// INT0 Mode: Low level
GICR|=0x40;
MCUCR=0x00;
MCUCSR=0x00;
GIFR=0x40;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;
ADCSRA=0x00;
SPCR=0x00;
TWCR=0x00;
#asm("sei")
while (1)
   {
   // Place your code here

   }
}
```

Setelah program di kompilasi dan diunduh pada mikrokontroler, berikanlah suatu keadaan saklar pada PORTD.2 dari high ke low. Hal ini akan mengakibatkan suatu interupsi pada program berupa nyala lampu bervariasi sesuai dengan urutan data dari inti program interupsi INT0. Hal ini karena pada saat setting INT0 tersebut kita memilih mode sulut Low Level, dimana mode ini akan mendeteksi suatu keadaan pada PORTD.2(INT0) dari High ke Low. Akibat daripada kondisi tersebut adalah suat program interupsi sesuai dengan konteks program yang diberikan pada kolom program INT0 yang kita buat.

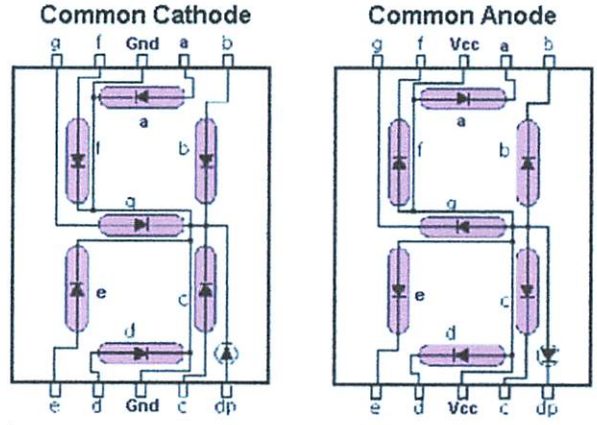
5. SEVEN SEGMENT DISPLAY

5.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program untuk menghidupkan seven segment display serta memahami pembentukan karakter pada seven segment dispay

5.2 DASAR TEORI

- Hardware seven segment display terdiri dari 2 seven segment common anoda yang setiap datanya dikontrol secara array oleh ULN2003A dan kontrol seven segmentnya dikontrol secara bit melalui pin control port.
- Data yang diberikan pada seven segment dibedakan menjadi dua jenis seperti dibawah ini.



- Berdasarkan data pada gambar masing masing data berupa urutan huruf a hingga g. Huruf ini masing – masing mewakili data LSB hingga MSB dan terdiri dari 8 bit secara keseluruhan. Masing – masing a hingga g bila dinyalakan susuai posisinya masing – masing akan membentuk kombinasi tertentu sesuai keterangan gambar.
- Pada Seven Segment Common Katode data yang diberikan adalah active high, ini artinya bahwa untuk menghidupkan masing – masing bit data diperlukan inputan 1(high). Sedangkan pada Seven Segment Common Anode data yang diberikan adalah active low, ini artinya untuk menghidupkan masing – masing bit data diperlukan inputan low.
- Sifat daripada ULN2003A adalah sebagai penguat output dengan membalik keadaan inputannya. Jadi apabila kita memerlukan data 1 maka kita memberikan inputan pada ULN2003A berupa data 0, begitu pula sebaliknya.

5.3 PROSEDUR PERCOBAAN

d) Percobaan IV : Seven Segment Display

Persiapan :

- Hubungkan portC mikrokontroler dengan control port pada peripheral I/O dan portD mikrokontroler dengan 7segment_PIN port pada peripheral I/O
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port D serta Port C sebagai output dengan keadaan awal outputan low "0".
- Ketikan program berikut pada program utama :

a.) Program Dasar I

```
#include <mega32a.h>
#include <delay.h>
// Alphanumeric LCD functions
#include <alcd.h>
// Declare your global variables here
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
DDRC=0xFF;
PORTD=0x00;
DDRD=0xFF;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
```

```
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;
ADCSRA=0x00;
SPCR=0x00;
TWCR=0x00;
lcd_init(8);

while (1)
    {
    // Place your code here
    PORTD = 0b00000110;
    PORTC = 0x00;
    }
}
```

Program tersebut diatas merupakan program yang berfungsi menampilkan karakter angka '1' pada tampilan seven segment. PORTD merupakan kontrol inputan data terhadap nilai seven segment yang akan diberikan, sedangkan PORT C merupakan kontrol Vcc seven segment melalui transistor PNP. Nilai '0' menyebabkan transistor dalam keadaan saturasi sehingga mengalirkan Vcc pada sumber seven segment, sebaliknya nilai '1' menyebabkan transistor pada keadaan cut off sehingga Vcc tidak mengalir pada sumber tegangan seven

segment. Pada percobaan selanjutnya nilai PORTD dapat diatur sedemikian rupa sehingga membentuk karakter tampilan yang diinginkan.

Berikut nilai terdaftar pada tabel sebagai hasil percobaan yang telah dilakukan untuk menghasilkan tampilan karakter pada seven segment.

No	Data 7 Segment(PORTD)								Data Heksadesimal	Hasil tampilan karakter
	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0		
1	0	0	1	1	1	1	1	1	3F H	Angka 0
2	0	0	0	0	0	1	1	0	06 H	Angka 1
3	0	1	0	1	0	1	1	1	57 H	Angka 2
4	0	1	0	0	1	1	1	1	4F H	Angka 3
5	0	1	1	0	0	1	1	0	66 H	Angka 4
6	0	1	1	0	1	1	0	1	6D H	Angka 5
7	0	1	0	0	1	1	0	1	4D H	Angka 6
8	0	0	0	0	0	1	1	1	07 H	Angka 7
9	0	1	1	1	1	1	1	1	7F H	Angka 8
10	0	1	1	0	1	1	1	1	6F H	Angka 9

Berikut program menghitung naik tampilan angka 0 hingga 9 pada pengujian seven segment display :

b.) Program II

```
#include <mega32.h>
#include <delay.h>
// Alphanumeric LCD Module functions
#asm
  .equ __lcd_port=0x18 ;PORTB
#endasm
#include <lcd.h>

// Declare your global variables here
unsigned char
bilangan1[10]=
{0x3f, 0x06, 0x57, 0x4f, 0x66, 0x6d, 0x4d, 0x07, 0x7f, 0x6f}; //deklarasi urutan data
heksadesimal pada portD
```

```
int geser, i;

void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0xFF;

PORTD=0x00;
DDRD=0xFF;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
ACSR=0x80;
```

```
SFIOR=0x00;

// LCD module initialization
while (1)
    {
    geser= 0;
    for(geser=0;geser<10;geser++)
    {
    PORTC.0=0;
    PORTC.1=0;
    PORTD=bilangan1[geser];
    Delay_ms(200);
    }
    };
}
```

6. LED DOT MATRIKS DISPLAY

6.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program untuk menghidupkan dot matriks 5x7 membentuk sebuah karakter dan memahami pemrograman dotmatriks tersebut.

6.2 DASAR TEORI

- Dot matriks display merupakan susunan LED yang membentuk dimensi beberapa kolom x beberapa baris. Dengan susunan seperti ini dot matriks dapat menghasilkan tampilan karakter dari kombinasi nyala LED di setiap dot(titik)nya.
- Dalam hardware ini digunakan Led Dot Matriks berdimensi 5x7.
- Teknik yang digunakan dalam menghidupkan dot matriks sering disebut dengan scanning. Yang dimaksud dengan scanning disini sendiri adalah mengendalikan nyala dot matriks dengan mengontrol kolom/baris yang dinyalakan secara bergeser per satuan waktu dengan kasat mata.

6.3 PROSEDUR PERCOBAAN

e) Percobaan V : Dot Matriks Display

Persiapan :

- Hubungkan portC mikrokontroler dengan IC4017 port pada peripheral I/O dan portD mikrokontroler dengan DotMatriks port pada peripheral I/O
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port D serta Port C sebagai output dengan keadaan awal outputan low "0".
- Ketikan program berikut pada program utama :

```
#include <mega32a.h>
#include <delay.h>
// Alphanumeric LCD functions
#include <alcd.h>
#define FRAME 5
```

```
// Declare your global variables here
char angka0[4]=
{
0b0111110
0b1000111
0b1001001
0b1110001
0b0111110
};
void resetIC()
{
PORTC=0x02;
Delay_ms(1);
PORTC=0x00;
Delay_ms(1);
}
void main(void)
{
// Declare your local variables here
Int i, j, k;
PORTA=0x00;
DDRA=0x00;

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0xFF;

PORTD=0x00;
DDRD=0xFF;

TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

TCCR1A=0x00;
TCCR1B=0x00;
```

```
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

MCUCR=0x00;
MCUCSR=0x00;

TIMSK=0x00;

UCSRB=0x00;

ACSR=0x80;
SFIOR=0x00;

ADCSRA=0x00;

SPCR=0x00;

TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTB Bit 0
// RD - PORTB Bit 1
// EN - PORTB Bit 2
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
```

```
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 8
lcd_init(8);

while (1)
{
// Place your code here
for(k=0 ; k<50 ; k++)
{
for(i=0 ; i<FRAME ; i++)
{
// Write the value from the array to the port
PORTC = 0x00;
PORTD = animation2[i];
delay_ms(2);
PORTC = 0x04;
}
}
Reset();
   }
}
```

Pada program diatas kita melakukan pemrograman tampilan dot matriks display dengan teknik scanning kolom. Pada sebuah definisi array angka0 berisikan sekumpulan data array yang nantinya mengisi PORTD sebagai keluaran pada baris dot matriks display. Hal ini bila disusun akan menjadi sebuah karakter angka 0 yang akan tampil pada dot matriks display. Secara kasat mata kita membuat pengisian alamat pada kaki – kaki kolom dot matriks sesuai data array yang telah kita definisikan.

7. LCD DISPLAY

7.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program untuk membentuk karakter angka maupun huruf pada LCD 16 x 2 melalui port mikrokontroler.

7.2 DASAR TEORI

- LCD display module adalah suatu embedded sistem yang dapat berkomunikasi melalui port mikrokontroler dengan instruksi tertentu sesuai library pemrograman yang menerjemahkan kode ASCII tertentu ke tampilan berupa huruf, angka, karakter pada papan tampilan.
- Untuk mengakses LCD, CodeVisionAVR telah menyediakan library LCD Display dengan pilihan karakter sebesar 16x2 hingga 20x2. Pada setiap percobaan ini digunakan LCD dengan dimensi karakter 16x2.

7.3 PROSEDUR PERCOBAAN

f) Percobaan V : LCD DISPLAY

Persiapan :

- Hubungkan portB mikrokontroler dengan LCD PORT pada peripheral I/O.
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port B sebagai port LCD Display. Perhatikan konfigurasi yang dibentuk, jangan sampai port yang dinisialisasikan tertukar.
- Ketikan program berikut pada program utama :

```
#include <mega32a.h>
#include <delay.h>

// Alphanumeric LCD functions
#include <alcd.h>
// Declare your global variables here
void main(void)
{
// Declare your local variables here

PORTA=0x00;
```

```
DDRA=0x00;

PORTB=0x00;
DDRB=0x00;

PORTC=0x00;
DDRC=0xFF;

PORTD=0x00;
DDRD=0xFF;

TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;

TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;

ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;

MCUCR=0x00;
MCUCSR=0x00;

TIMSK=0x00;

UCSRB=0x00;
```

```
ACSR=0x80;
SFIOR=0x00;

ADCSRA=0x00;

SPCR=0x00;

TWCR=0x00;

// Alphanumeric LCD initialization
// Connections are specified in the
// Project|Configure|C Compiler|Libraries|Alphanumeric LCD menu:
// RS - PORTB Bit 0
// RD - PORTB Bit 1
// EN - PORTB Bit 2
// D4 - PORTB Bit 4
// D5 - PORTB Bit 5
// D6 - PORTB Bit 6
// D7 - PORTB Bit 7
// Characters/line: 8
lcd_init(8);
lcd_clear();
lcd_gotoxy(0,1);
lcd_putsf("MODUL PRAKTIKUM");
lcd_gotoxy(1,1);
lcd_putsf("MIKROKONTROLER");
delay_ms(100);
while (1)
   {
   // Place your code here

   }
}
```

Sesuai program yang diketikan, fungsi daripada masing – masing perintah pada baris program adalah :

- "lcd_clear()" adalah fungsi pada library untuk menghapus tampilan karakter sebelumnya secara keseluruhan pada 16x2 dimensi lcd.

- "lcd_gotoxy(x,y)" adalah fungsi library yang menempatkan letak karakter awal pada barisx dan kolom y.
- "lcd_putsf("karakter")" adalah fungsi library yang menampilkan karakter berupa huruf dan atau angka yang ditulis dalam fungsi tersebut untuk ditampilkan pada tampilan lcd.

8. MOTOR STEPPER

8.1 TUJUAN PRAKTIKUM

- Praktikan mampu membuat program untuk menjalankan sebuah motor stepper dengan instruksi yang dituliskan pada mikrokontroler

8.2 DASAR TEORI

- Motor steeper merupakan aktuator yang memanfaatkan gaya tarik magnet. Secara elektronika penyusunnya terdiri dari masing – masing transistor dengan urutan yang biasanya terdiri dar A, B, C, dan D. Hal ini mempunyai tujuan untuk menunjukan urutan fasa gerak motor stepper. Sehingga pada saat pembuatan driver serta pemrograman pada mikrokontroler, motor stepper dapat dijalankan dengan benar.
- Motor yang digunakan merupakan pabrikan Mitsumi.Pin konektornya terdiri dari 4 konektor yang dapat diprogram secara *half* step maupun *full* step. Untuk pola full step, Motor stepper akan berputar sebesar 360^0 dengan banyaknya step sebanyak 200 step atau 1 stepnya mewakili 1.8^0. Untuk pola Half step, Motor stepper akan berputar sebesar 360^0 dengan banyaknya step sebanyak 400 step atau 1 stepnya mewakili 0.9^0. Selain itu mode *full step* akan menghasilkan sebuah putaran dengan kecepatan tinggi dan torsi kecil, sebaliknya mode *half step* akan menghasilkan putaran dengan kecepatan rendah dan torsi yang besar.

8.3 PROSEDUR PERCOBAAN

g) Percobaan V : MOTOR STEPPER

Persiapan :

- Hubungkan portD mikrokontroler dengan Pin stepper motor port pada peripheral I/O.
- Hubungkan sumber tegangan dengan minimum sistem AVR
- Hubungkan port USB ISP AVR Programer ke PC dan port ISP minimum sistem AVR
- Konfigurasi pada CodeWizard AVR dengan ketentuan chip pemrograman menggunakan ATMega32, frek. Xtall 11.0592 MHz, dan Port D sebagai port output dengan bit keluaran awal 0.
- Ketikan program berikut pada program utama :

a.) Program I :

```
#include <mega32a.h>
#include delay.h
// Declare your global variables here
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
DDRC=0x00;
PORTD=0x00;
DDRD=0xFF;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;
```

```
// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;

while (1)
    {
    // Place your code here
    //keadaan ClockWise
    PORTD=0x08;
    delay_ms(100);
    PORTD=0x04;
    delay_ms(100);
    PORTD=0x02;
    delay_ms(100);
    PORTD=0x01;
    }
```

b.) Program II :

```
#include <mega32a.h>
#include delay.h
// Declare your global variables here
void main(void)
{
// Declare your local variables here
PORTA=0x00;
DDRA=0x00;
PORTB=0x00;
DDRB=0x00;
PORTC=0x00;
```

```
DDRC=0x00;
PORTD=0x00;
DDRD=0xFF;
TCCR0=0x00;
TCNT0=0x00;
OCR0=0x00;
TCCR1A=0x00;
TCCR1B=0x00;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H=0x00;
ICR1L=0x00;
OCR1AH=0x00;
OCR1AL=0x00;
OCR1BH=0x00;
OCR1BL=0x00;
ASSR=0x00;
TCCR2=0x00;
TCNT2=0x00;
OCR2=0x00;
MCUCR=0x00;
MCUCSR=0x00;
TIMSK=0x00;
UCSRB=0x00;
ACSR=0x80;
SFIOR=0x00;

// ADC initialization
// ADC disabled
ADCSRA=0x00;

// SPI initialization
// SPI disabled
SPCR=0x00;

// TWI initialization
// TWI disabled
TWCR=0x00;
```

```
while (1)
  {
  // Place your code here
   //keadaan Counter Clock Wise
  PORTD=0x01;
  delay_ms(100);
  PORTD=0x02;
  delay_ms(100);
  PORTD=0x04;
  delay_ms(100);
  PORTD=0x08;
  }
```

- Pada program I, motor diberikan urutan data masing – masing 08H, 04H, 02H, dan 01H, hal ini merepresentasikan urutan data pada kutub transistor D ke C ke B dan terakhir ke A. Dengan urutan data tersebut maka motor akan berputar full step dengan arah clock wise (searah putaran jarum jam).
- Pada program II, motor diberikan urutan data masing – masing 01H, 02H, 04H, dan 08H, hal ini merepresentasikan urutan data pada kutub transistor A ke B ke C dan terakhir ke D. Dengan urutan data tersebut maka motor akan berputar full step dengan arah counter clock wise (berlawanan putaran jarum jam).