BAB II
TINJAUAN PUSTAKA

2.1 RFID (*Radio Frequency Identification*)

RFID (*Radio Frequency Identification*) adalah teknologi identifikasi berbasis gelombang radio. Teknologi ini mampu mengidentifikasi berbagai objek secara simultan tanpa diperlukan kontak langsung (atau dalam jarak pendek). RFID dikembangkan sebagai pengganti atau penerus teknologi *barcode*. RFID bekerja pada HF *(High Frequency)* untuk aplikasi jarak dekat (*proximity*) dan bekerja pada UHF *(Ultra High Frequency)* untuk aplikasi jarak jauh (*vicinity*).

Sensor RFID adalah sensor yang mengidentifikasi suatu barang dengan menggunakan frekuensi radio. Sensor ini terdiri dari dua bagian penting: *transceiver* (*reader*) dan *transponder* (*tag*). Setiap *tag* tersimpan data yang berbeda. Data tersebut merupakan data identitas *tag*. *Reader* akan membaca data dari *tag* dengan perantara gelombang radio. Pada *reader* biasanya berhubungan dengan suatu mikrokontroler. Mikrokontroler ini berfungsi untuk mengolah data yang didapat *reader*.

Gambar 2.1 *Reader* RFID (*Radio Frequency Identification*)

2.1.1 Pembaca RFID

Pembaca RFID adalah merupakan penghubung antara *software* aplikasi dengan antena yang akan meradiasikan gelombang radio ke *tag* RFID. Gelombang radio yang diemisikan oleh antena berpropagasi pada ruangan di sekitarnya. Akibatnya data dapat berpindah secara *wireless* ke *tag* RFID yang berada berdekatan dengan antena.

Sebuah pembaca RFID harus menyelesaikan dua buah tugas, yaitu:

a.Menerima perintah dari *software* aplikasi

b.Berkomunikasi dengan *tag* RFID

2.1.2 *Tag* RFID

Tag RFID adalah perangkat yang dibuat dari rangkaian elektronika dan antena yang terintegrasi di dalam rangkaian tersebut. Rangkaian elektronik dari *tag* RFID umumnya memiliki memori sehingga *tag* ini mempunyai kemampuan untuk menyimpan data. Memori pada *tag* secara dibagi menjadi sel-sel. Beberapa sel menyimpan data *Read Only*, misalnya *serial number* yang unik yang disimpan pada saat *tag* tersebut diproduksi. Selain pada RFID mungkin juga dapat ditulis dan dibaca secara berulang. (*sumber:http://iwanjuda.com/2013/12/20/identifikasi-frekuensi-radio-atau-rfid/*)

Gambar 2.2 Macam-Macam RFID

(*sumber:http://www.marcodesalvo.it/en/287-swinxs/*)

Berdasarkan catu daya *tag*, *tag* RFID dapat digolongkan menjadi:

a. *Tag* Aktif: yaitu *tag* yang catu dayanya diperoleh dari baterai, sehingga akan mengurangi daya yang diperlukan oleh pembaca RFID dan *tag* dapat mengirimkan informasi dalam jarak yang lebih jauh. Kelemahan dari tipe *tag* ini adalah harganya yang mahal dan ukurannya yang lebih besar karena lebih komplek. Semakin banyak fungsi yang dapat dilakukan oleh *tag* RFID maka rangkaiannya akan semakin komplek dan ukurannya akan semakin besar.
b. *Tag* Pasif: yaitu *tag* yang catu dayanya diperoleh dari medan yang dihasilkan oleh pembaca RFID. Rangkaiannya lebih sederhana, harganya jauh lebih murah, ukurannya kecil, dan lebih ringan. Kelemahannya adalah *tag* hanya dapat mengirimkan informasi dalam jarak yang dekat dan pembaca RFID harus menyediakan daya tambahan untuk *tag* RFID.

2.1.3 Frekuensi Kerja RFID

Faktor penting yang harus diperhatikan dalam RFID adalah frekuensi kerja dari sistem RFID. Ini adalah frekuensi yang digunakan untuk komunikasi *wireless* antara pembaca RFID dengan *tag* RFID.

Ada beberapa band frekuensi yang digunakan untuk sistem RFID yaitu:

Low Frequency (LF) : 125 - 134 KHz
High Frequency (HF) : 13.56 MHz
Ultra High Frequency (UHF) : 868 - 956 MHz
Microwave : 2.45 GHz
(*sumber://prisekip.ugm.ac.id/files/2009/08/11.pdf*)

Pemilihan dari frekuensi kerja sistem RFID akan mempengaruhi jarak komunikasi, interferensi dengan frekuensi sistem radio lain, kecepatan komunikasi data, dan ukuran antena. Untuk frekuensi yang rendah (*Low Frequency* (LF) : 125 - 134 KHz) umumnya digunakan *tag* pasif (tidak memiliki sumber energy sendiri tanpa *battery,* Modulasi akan aktif setelah *tag*

menerima gelombang elektromagnetik dari *reader*) dan untuk frekuensi tinggi (*High Frequency* (HF) : 13.56 MHz - *Microwave* : 2.45 GHz) digunakan *tag* aktif (memiliki sumber energi sendiri, modulasi aktif langsung dari *tag* sendiri). Pada frekuensi rendah, *tag* pasif tidak dapat mentransmisikan data dengan jarak yang jauh, karena keterbatasan daya yang diperoleh dari medan elektromagnetik. Akan tetapi komunikasi tetap dapat dilakukan tanpa kontak langsung.

Pada frekuensi tinggi, jarak komunikasi antara *tag* aktif dengan pembaca RFID dapat lebih jauh, tetapi masih terbatas oleh daya yang ada. Sinyal elektromagnetik pada frekuensi tinggi juga mendapatkan pelemahan ketika *tag* tertutupi oleh es atau air. Pada kondisi terburuk, *tag* yang tertutup oleh logam tidak terdeteksi oleh pembaca RFID.

2.1.4 Akurasi RFID

Akurasi RFID dapat didefinisikan sebagai tingkat keberhasilan pembaca RFID melakukan identifikasi sebuah *tag* yang berada pada area kerjanya. Keberhasilan dari proses identifikasi sangat dipengaruhi oleh beberapa batasan fisik, yaitu:

a. Posisi antena pada pembaca RFID
b. Karakteristik dari material lingkungan yang mencakup sistem RFID
c. Batasan *catu* daya
d. Frekuensi kerja sistem RFID

Sensor RFID yang digunakan adalah RFID 125KHz. Sensor ini digunakan karena RFID dengan frekuensi 125 KHz merupakan model yang banyak dijumpai dipasaran dan memiliki harga relatif terjangkau.

2.2 Mikrokontroler ATMega8

Mikrokontroler jenis AVR adalah prosesor yang sekarang ini paling banyak

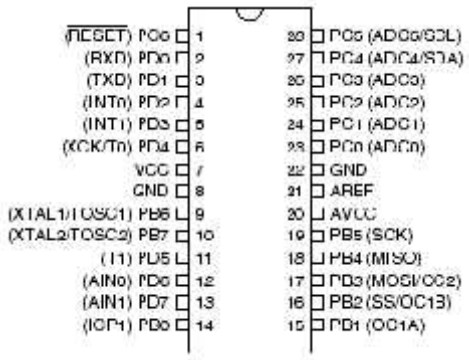
digunakan dalam membuat aplikasi sistem kendali bidang instrumentasi, dibandingkan dengan mikrokontroler keluarga MCS51 seperti AT 89C51/52.

Mikrokontroler seri AVR pertama kali diperkenalkan ke pasaran sekitar tahun 1997 oleh perusahaan Atmel, yaitu sebuah perusahaan yang sangat terkenal dengan produk mikrokontroler seri AT89S51/52- nya yang sampai sekarang masih banyak digunakan di lapangan. Keterbatasan pada mikrokontroler tersebut (resolusi, memori, dan kecepatan) menyebabkan banyak orang beralih ke mikrokontroler AVR.

Mikrokontroler AVR standar memiliki arsitektur 8 bit, dimana semua instruksi dikemas dalam kode 16 bit dan sebagian besar instruksi dieksekusi dalam satu situs *clock*, berbeda dengan instruksi MCS51 yang membutuhkan 12 situs *clock* (Widodo Budiharto dan Gamayel Rizal, 2007:28).

Hal ini karena kedua jenis mikrokontroler tersebut memiliki arsitektur yang berbeda. AVR berteknologi RISC (*Reduce Insruction Set Computing*), sedangkan seri MCS51 berteknologi CISC (*Complex Instruction Set Computing*). AVR dapat dikelompokkan menjadi empat kelas yaitu keluarga AT*tiny*, keluarga AT90Sxx, keluarga ATmega, dan AT86RFFxx. Perbedaan dari masing- masing keluarga AVR tersebut adalah memori, peripheral, dan fungsinya.

2.2.1 Konfigurasi Pin ATMega8



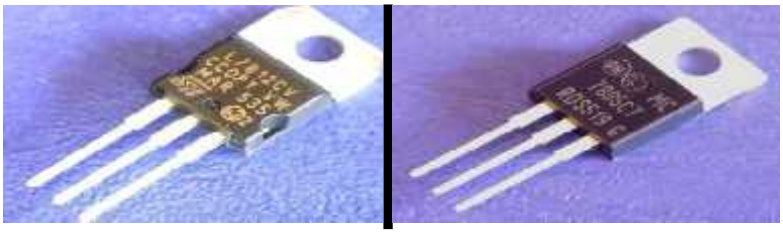
Gambar 2.3 Susunan pin ATMega8

Berikut ini adalah susunan pin/kaki dari ATmega8 :

1. VCC adalah merupakan pin masukan positif catu daya.
2. GND sebagai pin *Ground*.
3. PORT B (B.0-B.5) merupakan pin I/O dua arah dan pin fungsi khusus yaitu *Timer/Counter*, dan SPI.
4. PORT C (C.0-C.6) merupakan pin I/O dua arah dan dapat diprogram sebagai pin ADC.
5. PORT D (D.0-D.4) merupakan pin I/O dua arah dan pin fungsi khusus yaitu interupsi *eksternal* dan komunikasi serial.
6. *Reset* merupakan pin yang digunakan untuk me-*reset* mikrokontroler.
7. XTAL1 dan XTAL2 sebagai pin masukan *clock* eksternal. Suatu mikrokontroler membutuhkan sumber detak (*clock*) agar dapat mengeksekusi instruksi yang ada di memori. Semakin tinggi kristalnya, semakin cepat kerja mikrokontroler tersebut.
8. AVCC sebagai pin *supply* tegangan untuk ADC.
9. AREF sebagi pin masukan tegangan referensi untuk ADC.

2.3 IC Regulator

Untuk menstabilkan tegangan DC (+) dan tegangan DC (-) dari catu daya utama sebelum mensuplay rangkaian maka perlu digunakan regulator dengan memasang IC regulator tipe 78xx dan 79xx agar tegangan *output*-nya sesuai dengan kebutuhan rangkaian.



Gambar 2.4 Bentuk IC Regulator
(*sumber: http://www.linksukses.com/2012/05/regulator-5-volt.html*)

IC regulator memiliki berbagai macam jenis yang tergantung dari besar *output* keluarannya. Tidak semua nilai tegangan dapat diwujudkan dengan menggunakan IC regulator. Produsen IC regulator sudah menetapkan berbagai jenis IC regulator berdasarkan *output* nya yang sampai sekarang ini banyak digunakan dalam rangkaian elektronik. Berbagai tipe IC regulator beserta hasil keluaran *output*-nya dapat dilihat pada Tabel 2.1.

Tabel 2.1 Tipe IC Regulator

No	Tipe	Output	Tipe	Output
1	L7805C	5V	L7905C	-5V
2	L7852C	5.2V	L7952C	-5.2V
3	L7806C	6V	L7906C	-6V
4	L7808C	8V	L7908C	-8V
5	L7809C	9V	L7909C	-9V
6	L7812C	12V	L7912C	-12V
7	L7815C	15V	L7915C	-15V
8	L7818C	18V	L7918C	-18V
9	L7820C	20V	L7920C	-20V
10	L7822C	22V	L7922C	-22V
11	L7824C	24V	L7924C	-24V

Jika dibandingkan dengan regulator tegangan lain, seri 78XX ini mempunyai keunggulan diantaranya :

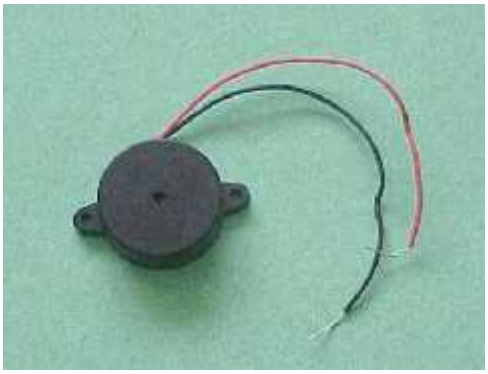
1. Untuk regulasi tegangan DC, tidak memerlukan komponen elektronik tambahan.
2. Aplikasi mudah dan hemat ruang.
3. Memiliki proteksi terhadap *overload* (beban lebih), *overheat* (panas lebih), dan hubungsingkat.
4. Dalam keadaan tertentu, kemampuan pembatasan arus peranti 78XX tidak hanya melindunginya sendiri, tetapi juga melindungi rangkaian yang ditopangnya.

Kekurangan :

1. Tegangan *input* harus lebih tinggi 2-3 Volt dari tegangan *output* sehingga IC 7805 kurang tepat jika digunakan untuk menstabilkan tegangan baterai 6 Volt menjadi 5 Volt.
2. Seperti halnya regulator linier lain, arus input sama dengan arus *output*. Karena tegangan *input* harus lebih tinggi dari tegangan *output* maka akan terjadi panas pada IC regulator 7805 sehingga diperlukan pendingin yang cukup.

2.4 Buzzer

Buzzer adalah sebuah komponen elektronika yang berfungsi untuk mengubah getaran listrik menjadi getaran suara. Pada dasarnya prinsip kerja *buzzer* hampir sama dengan *loud speaker*, jadi *buzzer* juga terdiri dari kumparan yang terpasang pada diafragma dan kemudian kumparan tersebut dialiri arus sehingga menjadi elektromagnet, kumparan tadi akan tertarik ke dalam atau keluar, tergantung dari arah arus dan polaritas magnetnya, karena kumparan dipasang pada diafragma maka setiap gerakan kumparan akan menggerakkan diafragma secara bolak-balik sehingga membuat udara bergetar yang akan menghasilkan suara.



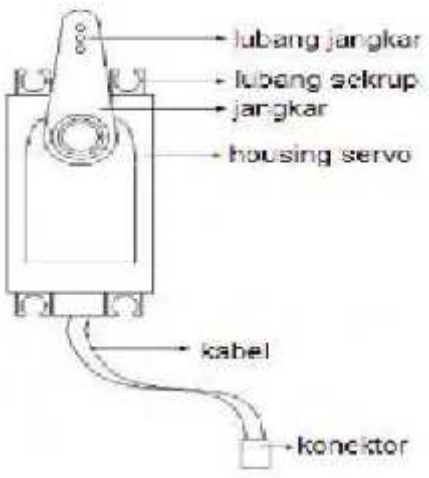
Gambar 2.5 Bentuk Fisik *Buzzer*

2.5 Motor Servo

Motor Servo adalah sebuah motor DC yang dilengkapi rangkaian kendali dengan sistem *closed feedback* yang terintegrasi dalam motor tersebut. Pada motor servo posisi putaran sumbu (*axis*) dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. (*sumber: http://www.scribd.com/doc/156131684/Makalah-motor-servo*)

Gambar 2.6 Motor Servo

Motor servo disusun dari sebuah motor DC, *gearbox*, *variabel* resistor (VR) atau potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas maksimum putaran sumbu (*axis*) motor servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang pada pin kontrol motor servo.



Gambar 2.7 Komponen Luar Motor Servo

Motor servo ini terdiri dari beberapa bagian sebagai berikut:

1. Jangkar untuk menghubungkan motor servo dengan objek-objek yang akan digerakkan.
2. Lubang Jangkar bagian ini berfungsi untuk menempatkan sekrup yang mengaitkan jangkar ke objek-objek yang akan digerakkan. Pada gambar tampak lubang jangkar dihubungkan ke obyek dengan sekrup untuk gerakan memutar.
3. Lubang Sekrup yang berfungsi untuk mengaitkan motor servo dengan tubuh robot.
4. *Housing* Servo, di dalam bagian ini terdapat motor DC, *gearbox* dan rangkaian pengatur sudut servo.
5. Kabel, kabel yang menghubungkan rangkaian servo dengan pengendali servo.
6. Konektor, konektor 3 pin yang terdiri dari *input* tegangan positif (+), *input* tegangan negatif (GND) dan *input* pulsa (*signal*). Cara yang paling mudah untuk menentukan posisi kabel *signal* adalah dengan mengingat kabel merah adalah (+), kabel hitam adalah (-), warna lain selain merah dan hitam adalah kabel *signal*.

2.6 Baterai *Recharge* 9 Volt

Dengan meningkatnya perangkat *portable* seperti laptop, ponsel, MP3 player, dan alat-alat listrik tanpa kabel lainnya, kebutuhan akan baterai isi ulang meningkat dengan pesat dalam beberapa tahun terakhir. Baterai isi ulang telah ada sejak tahun 1859, ketika fisikawan Perancis, Gaston Plante menemukan sel asam timbal. Dengan timbal anoda, timbal dioksida katoda, dan elektrolit asam sulfat, baterai Plante adalah pendahulu dari aki mobil saat ini.

Baterai *non-rechargeable*, atau sel primer, dan baterai isi ulang, atau sel sekunder, menghasilkan arus dengan cara yang sama: melalui reaksi elektrokimia melibatkan anoda, katoda, dan elektrolit. Dalam baterai isi ulang, reaksinya dapat dibalik. Ketika energi listrik dari sumber luar diterapkan pada sel sekunder, aliran

elektron negatif ke positif yang terjadi selama pelepasan dibalik, dan pengisian muatan sel dikembalikan. Baterai isi ulang yang paling umum di pasar saat ini adalah *lithium-ion* (LiOn) , meskipun *nikel-metal hidrida* (NiMH) dan *nikel – kadmium* (NiCd) merupakan baterai yang juga pernah sangat umum digunakan.

Dalam hal baterai isi ulang, tidak semua baterai dibuat sama. Baterai NiCd termasuk yang pertama sel sekunder yang tersedia secara luas, tetapi mereka mengalami masalah sulit yang dikenal sebagai efek memori. Pada dasarnya, jika baterai ini tidak sepenuhnya habis setiap kali mereka digunakan, mereka akan cepat kehilangan kapasitasnya. Baterai NiCd sebagian besar ditinggalkan karena ada baterai NiMH. Sel-sel sekunder membanggakan kapasitas yang lebih tinggi dan hanya sedikit dipengaruhi oleh efek memori, tetapi mereka tidak memiliki umur simpan yang sangat baik. Seperti baterai NiMH, baterai LiOn memiliki umur panjang, tetapi mereka menyimpan muatan lebih baik, beroperasi pada tegangan yang lebih tinggi, dan tersedia dalam kemasan yang jauh lebih kecil dan ringan. Pada dasarnya semua teknologi portabel berkualitas tinggi yang diproduksi hari ini mengambil keuntungan dari teknologi ini. Namun, baterai LiOn saat ini tidak tersedia dalam ukuran standar seperti AAA, AA, C atau D, dan jauh lebih mahal dari rekan-rekannya yang lebih dulu.

Dengan baterai NiCd dan NiMH, pengisian dapat menjadi rumit. Anda harus berhati-hati untuk tidak men-*charge* terlalu berlebihan, karena hal ini dapat menyebabkan penurunan kapasitas. Untuk mencegah hal ini terjadi, beberapa pengisi beralih ke *trickle charge* atau cukup matikan pengisian saat pengisian selesai. Baterai NiCd dan NiMH juga harus direkondisi, berarti Anda sesekali harus benar-benar melakukan pengosongan dan pengisian lagi untuk meminimalkan kehilangan kapasitas. Di sisi lain, baterai LiOn, memiliki pengisian daya canggih yang mencegah pengisian yang berlebihan dan tidak perlu direkondisi.

Bahkan pada akhirnya baterai isi ulang pun akan mati, meskipun mungkin membutuhkan ratusan kali pengisian ulang sebelum hal itu terjadi. Ketika mereka akhirnya benar-benar mati, pastikan untuk membuang mereka di fasilitas daur ulang.

Gambar 2.8 Baterai *Charge* 9volt

(*sumber:http://www.amazine.co/28270/tembaga-cu-fakta-sifat/*)

2.7 Serat Tembaga

Tembaga merupakan logam kemerahan dengan struktur kristal kubus. Tembaga memantulkan sinar merah dan oranye dan menyerap frekuensi lain dalam spektrum cahaya terlihat. Logam ini mudah ditempa, ulet, dan merupakan konduktor panas dan listrik yang baik. Tembaga lebih lunak dari seng, dapat dipoles, dan memiliki reaktivitas kimia rendah. Dalam udara lembab, tembaga perlahan-lahan membentuk selaput permukaan kehijauan yang disebut patina. Lapisan ini melindungi dari serangan korosi lebih lanjut. Tembaga merupakan unsur yang banyak terdapat di alam. Manusia tercatat juga banyak menggunakan tembaga. Kebanyakan tembaga digunakan untuk peralatan listrik (60 %); konstruksi, seperti atap dan pipa (20%); mesin industri, seperti penukar panas (15 %); dan paduan logam (5 %). Tembaga ideal digunakan sebagai kabel jaringan listrik karena mudah ditangani, dapat ditarik menjadi kawat halus, dan memiliki konduktivitas listrik tinggi.

Gambar 2.9 Serat Tembaga

2.8 Bahasa C

Dikembangkan pertama kali oleh Dennis Ritchie dan Ken Thomson pada tahun 1972, Bahasa C merupakan salah satu bahasa pemrograman yang paling populer untuk pengembangan program- program aplikasi yang berjalan pada sistem mikroprosesor (komputer). Karena kepopulerannya, vendor-vendor perangkat lunak kemudian mengembangkan *compiler* C sehingga menjadi beberapa varian berikut: *Turbo* C, *Borland* C, *Microsof*t C, *Power* C, *Zortech* C dan lain sebagainya. Untuk menjaga portabilitas, *compiler-compiler* C tersebut menerapkan ANSI C (ANSI: *American National Standards Institute*) sebagai standar bakunya. Perbedaan antara *compiler-compiler* tersebut umumnya hanya terletak pada pengembangan fungsi-fungsi pustaka serta fasilitas IDE (*Integrated Development Environment*)-nya saja.

Relatif dibandingkan dengan bahasa aras tinggi lain, bahasa C merupakan bahasa pemrograman yang sangat fleksibel dan tidak terlalu terikat dengan berbagai aturan yang sifatnya kaku. Satu- satunya hal yang membatasi penggunaan bahasa C dalam sebuah aplikasi adalah semata-mata kemampuan imaginasi programmer-nya saja. Sebagai ilustrasi, dalam program C kita dapat saja secara bebas menjumlahkan karakter huruf (misal „A") dengan sebuah bilangan bulat (misal „2"), dimana hal yang sama tidak mungkin dapat dilakukan dengan menggunakan bahasa aras tinggi lainnya. Karena sifatnya ini, seringkali bahasa C dikategorikan sebagai bahasa aras menengah (*mid level language*).

Dalam kaitannya dengan pemrograman mikrokontroler, bahasa C sekarang mulai menggeser bahasa yang lebih dulu digunakan untuk pemrograman mikrokontroler yaitu bahasa *assembler*. Penggunaan bahasa C akan sangat efisien terutama untuk program mikrokontroler yang berukuran relatif besar. Dibandingkan dengan bahasa *assembler*, penggunaan bahasa C dalam pemrograman memiliki beberapa kelebihan berikut: Mempercepat waktu pengembangan, bersifat modular dan terstruktur, sedangkan kelemahannya adalah kode program hasil kompilasi akan relatif lebih besar dan sebagai konsekuensinya hal ini terkadang akan mengurangi kecepatan eksekusi.

Khusus pada mikrokontroler AVR, untuk mereduksi konsekuensi negatif diatas, perusahaan Atmel merancang sedemikian sehingga arsitektur AVR ini efisien dalam mendekode serta mengeksekusi instruksi-instruksi yang umum dibangkitkan oleh *compiler* C (Dalam kenyataanya, pengembangan arsitektur AVR ini tidak dilakukan sendiri oleh perusahaan Atmel tetapi ada kerja sama dengan salah satu vendor pemasok *compiler* C untuk mikrokontroler tersebut, yaitu IAR C).

Tabel 2.2 Beberapa *Compiler* C untuk mikrokontroler AVR

Compiler C	Platform	Keterangan
IAR C	-DOS -Windows	Komersil
CodeVisionAVR	-Windows	Komersil,
ImageCraft's C	-DOS Windows -Linux	Komersil
AVR-GCC	DOS -Windows	General Public Licence
C-AVR	-Windows	Komersil
Small C for AVR	-DOS	Komersil
GNU C for AVR	Linux	General Public Licence
LCC-AVR	-Linux, -Windows	Free
Dunfields AVR	-Windows	Komersil

Struktur penulisan bahasa C secara umum terdiri atas empat blok, yaitu :

a. *Header*.
b. Deklarasi konstanta global atau variabel.
c. Fungsi atau prosedur.
d. Program utama.

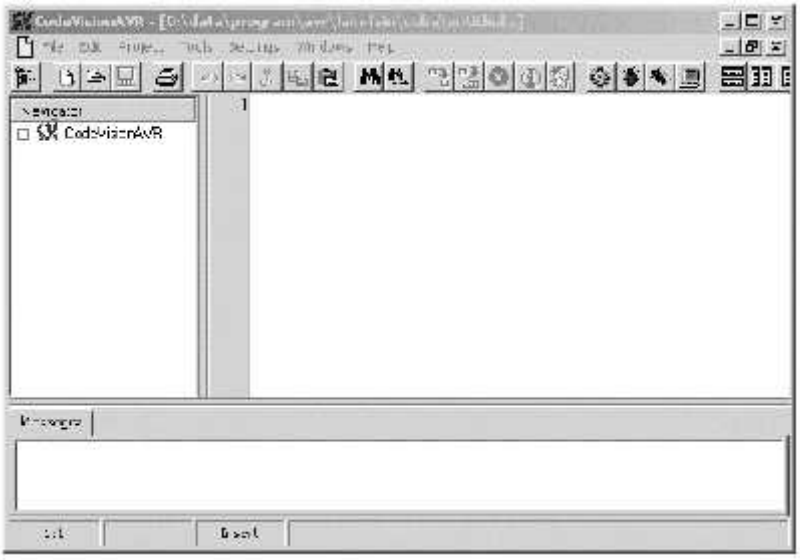
2.9 Perangkat Lunak *CodeVisionAVR* (CVAVR)

CodeVisionAVR pada dasarnya merupakan perangkat lunak pemrograman mikrokontroler keluarga AVR berbasis bahasa C. Ada tiga komponen penting yang telah diintegrasikan dalam perangkat lunak ini: *compiler* C, IDE dan Program generator.

Berdasarkan spesifikasi yang dikeluarkan oleh perusahaan pengembangnya,

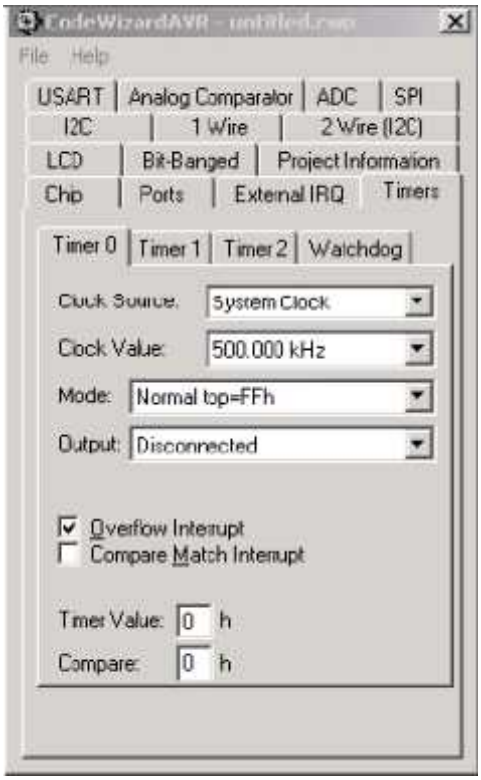
compiler C yang digunakan hampir mengimplementasikan semua komponen standar yang ada pada bahasa C standar ANSI (seperti struktur program, jenis tipe data, jenis operator, dan pustaka fungsi standar-berikut penamaannya). Tetapi walaupun demikian, dibandingkan bahasa C untuk aplikasi komputer, *compiler* C untuk mikrokontroler ini memiliki sedikit perbedaan yang disesuaikan dengan arsitektur AVR tempat program C tersebut ditanamkan (*embedded*).

Khusus untuk pustaka fungsi, disamping pustaka standar (seperti fungsi-fungsi matematik, manipulasi *string*, pengaksesan memori dan sebagainya), *CodeVisionAVR* juga menyediakan fungsi-fungsi tambahan yang sangat bermanfaat dalam pemrograman antarmuka AVR dengan perangkat luar yang umum digunakan dalam aplikasi kontrol. Beberapa fungsi pustaka yang penting diantaranya adalah fungsi-fungsi untuk pengaksesan LCD, komunikasi I C, IC RTC (*Real Time Clock*), sensor suhu LM35, SPI (*Serial Peripheral Interface*) dan lain sebagainya. Untuk memudahkan pengembangan program aplikasi, *CodeVisionAVR* juga dilengkapi IDE yang sangat *user friendly*. Selain menu-menu pilihan yang umum dijumpai pada setiap perangkat lunak berbasis *Windows*, *CodeVisionAVR* ini telah mengintegrasikan perangkat lunak *downloader* (*in system programmer*) yang dapat digunakan untuk mentransfer kode mesin hasil kompilasi kedalam sistem memori mikrokontroler AVR yang sedang diprogram.



Gambar 2.10 IDE Perangkat Lunak *CodeVisionAVR*

Selain itu, *CodeVisionAVR* juga menyediakan sebuah *tool* yang dinamakan dengan *Code Generator* atau *CodeWizardAVR* . Secara praktis, *tool* ini sangat bermanfaat membentuk sebuah kerangka program (*template*), dan juga memberi kemudahan bagi programmer dalam peng-inisialisasian register-register yang terdapat pada mikrokontroler AVR yang sedang diprogram. Dinamakan kode generator, karena perangkat lunak *CodeVision* ini akan membangkitkan kode-kode program secara otomatis setelah fase inisialisasi pada jendela *CodeWizardAVR* selesai dilakukan. Secara teknis, penggunaan *tool* ini pada dasarnya hampir sama dengan *application wizard* pada bahasa-bahasa pemrograman visual untuk komputer (seperti *Visual C*, *Borland Delphi*, dan sebagainya).



Gambar 2.11 Kode Generator Untuk Inisialisasi Register Pada Mikrokontroler AVR

2.10 Flowchart

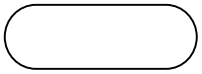
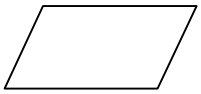
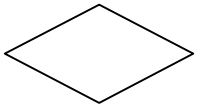
Menurut Indrajani (2011:22), *flowchart* merupakan penggambaran secara grafik dari langkah-langkah dan urutan prosedur program yang biasanya mempermudah penyelesaian masalah.

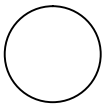
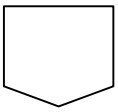
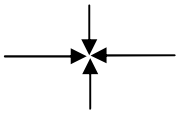
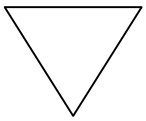
Flowchart atau diagram alir merupakan sebuah diagram dengan simbol-simbol grafis yang menyatakan aliran algoritma atau proses yang menampilkan langkah-langkah yang disimbolkan dalam bentuk kotak, beserta urutannya dengan menghubungkan masing-masing langkah tersebut menggunakan tanda panah. Diagram ini bisa memberi solusi selangkah demi selangkah untuk penyelesaian masalah yang ada di dalam proses atau algoritma tersebut.

2.10.1 Simbol – Simbol Flowchart

Flowchart disusun dengan simbol-simbol. Simbol-simbol ini dipakai sebagai alat bantu menggambarkan proses di dalam program. Simbol - simbol *flowchart* beserta fungsinya dapat ditunjukkan pada Tabel 2.3.

Tabel 2.3 Simbol - Simbol Flowchart

No	Simbol	Fungsi
1	Terminal 	Simbol untuk memulai dan mengakhiri suatu program
2	Proses 	Simbol untuk menyatakan suatu tindakan (proses) yang dilakukan oleh komputer
3	*Manual Operator* 	Simbol untuk menyatakan suatu tindakan (proses) yang tidak dilakukan oleh komputer
4	*Input – Output* 	Simbol untuk menyatakan proses *input* atau *output* tanpa tergantung jenis peralatannya
5	*Decision* 	Simbol untuk menunjukkan suatu kondisi tertentu yang akan menghasilkan dua kemungkinan jawaban : ya / tidak
6	*Predefined Process* 	Simbol untuk menyatakan penyediaan tempat penyimpanan suatu pengolahan didalam storage

7	*Connector* 	Simbol untuk menyatakan sambungan dari proses ke proses lainnya dalam halaman yang sama
8	*Off Line Connector* 	Simbol untuk menyatakan sambungan dari proses ke proses lainnya dalam halaman yang berbeda
9	Arus atau *Flow* 	Garis untuk menghubungkan arah tujuan simbol *flowchart* yang satu dengan yang lainnya
10	*Off-line Storage* 	Simbol untuk menunjukkan bahwa data di dalam simbol ini akan disimpan ke suatu media tertentu
11	*Manual Input* 	Simbol untuk memasukkan data secara manual dengan menggunakan *on-line keyboard*
12	*Punched Card* 	Simbol untuk menyatakan input berasal dari kartu atau output ditulis ke kartu