BAB II
TINJAUAN PUSTAKA

2.1. Short Message Service (S M S)

Yaitu fasilitas yang dimiliki oleh jaringan GSM (*Global System for Mobile Communikation)* yang memungkinkan pelanggan untuk mengirimkan dan menerima pesan-pesan singkat. SMS ditangani oleh jaringan melalui suatu pusat layanan atau SMS *Service Centre* (SMS SC) yang berfungsi menyimpan dan meneruskan pesan dari sisi pengirim ke sisi penerima. Format SMS yang dipakai oleh produsen MS (*Mobile Station)* adalah *Protocol Deskription Unit* (PDU). Format PDU akan mengubah septet kode ASCII (7bit) menjadi bentuk byte PDU (8bit) pada saat pengiriman data dan akan diubah kembali menjadi kode ASCII pada saat diterima oleh MS.

2.1.1. PDU (Protocol Data Unit) SMS

Dalam proses pengiriman atau penerimaan pesan pendek (SMS), data yang dikirim maupun diterima oleh stasiun bergerak menggunakan salah satu dari 2 mode yang ada, yaitu: mode teks, atau mode PDU (*Protocol Data Unit*) (Wavecom, 2000). Dalam mode PDU, pesan yang dikirim berupa informasi dalam bentuk data dengan beberapa kepala-kepala informasi. Hal ini akan memberikan kemudahan jika dalam pengiriman akan dilakukan kompresi data, atau akan dibentuk sistem penyandian data dari karakter dalam bentuk untaian bit-bit biner. Senarai PDU tidak hanya berisi pesan teks saja, tetapi terdapat beberapa meta-informasi yang lainnya, seperti nomor pengirim, nomor SMS Centre, waktu pengiriman, dan sebagainya. Semua informasi yang terdapat dalam PDU,

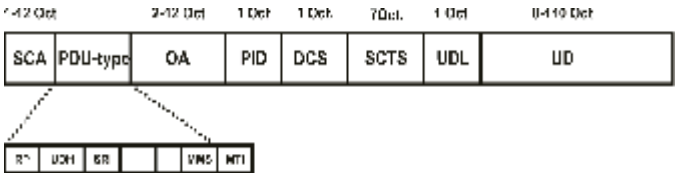
dituliskan dalam bentuk pasangan-pasangan bilangan heksadesimal yang disebut dengan pasangan oktet. Jenis PDU SMS yang akan digunakan adalah: SMS-Penerimaan (*SMS-DELIVER*) dan SMS-Pengiriman (*SMS SUBMIT*).

a. PDU Penerimaan (SMS-Deliver)

SMS Penerimaan (*SMS-Deliver*) adalah pesan yang diterima oleh terminal dari SMSC dalam bentuk PDU. PDU SMS-Penerimaan memiliki format seperti pada Gambar 2.1.

Pada PDU ini, terdapat beberapa meta-informasi yang dibawa, antara lain:

1. SCA (Service Centre Address), Berisi informasi SMS-centre.
2. Tipe PDU (PDU Type), Berisi informasi jenis dari PDU tersebut.
3. OA (Originating Address), Berisi informasi nomor pengirim.
4. PID (Protocol Identifier), Berisi informasi Identifikasi Protokol yang digunakan.
5. DCS (Data Coding Scheme), Berisi informasi skema pengkodean data yang digunakan.
6. SCTS (Service Center Time Stamp), Berisi informasi waktu
7. UDL (User Data Length), Berisi informasi panjang dari data yang dibawa.
8. UD (User Data), Berisi informasi data-data utama yang dibawa.

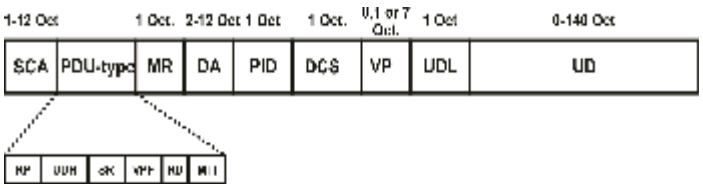


Gambar 2.1. Format PDU penerimaan

b. PDU Pengiriman (SMS-Submit)

PDU Pengiriman memiliki informasi-informasi yang sama dengan PDU Penerimaan, sementara yang berbeda adalah berupa informasi, antara lain :

1. MR (*Message Reference*), Parameter yang mengindikasikan nomor referensi SMS-Pengiriman.
2. DA (*Destination Address*), Berisi informasi nomor alamat yang dituju.
3. VP (*Validity Period*), Berisi informasi jangka waktu validitas pesan pada jaringan.



Gambar 2.2. Format PDU pengiriman

c. PDU untuk Kirim SMS ke SMS Centre.

1. Nomor SMS Center

a. Jumlah pasangan heksadesimal SMS *center* dalam bilangan heksa

b. Nasional / Internasional Code

Untuk Nasional, kode subheadernya yaitu 81

Untuk Internasional, kode subheadernya yaitu 91

c. No SMS Centernya sendiri, dalam pasangan heksa dibalik-balik. Jika tertinggal satu angka heksa yang tidak memiliki pasangan, angka tersebut akan dipasangkan dengan huruf F di depannnya.

Contoh:

Untuk no SMS Center Excelcom dapat ditulis dengan dua cara sebagai berikut:

Cara 1 :

0818445009 diubah menjadi :

a. 80-81-44-05-90----□ 5 pasang

b. 81 ----------□ 1 pasang

Total: 6 pasang

Digabung menjadi :**06818081440590**

Cara 2 :

62818445009 diubah menjadi :

a. 62-81-84-45-50-09□6 pasang

b. 91 ----------□ 1 pasang

Total: 7 pasang

Digabung menjadi : **07912618485400F921**

Tabel 2.1. Cara 1 Operator SMS Center

No	Operator Seluler	SMS Center NO	Kode PDU
1	Telkomsel	0811000000	06818011000000
2	Satelindo	0816125	0581806121F5
3	Excelcom	0818445009	06818081440590
4	Indo-M3	0855000000	06818055000000

Tabel 2.2. Cara 2 Operator Seluler

No	Operator Seluler	SMS Center NO	Kode PDU
1	Telkomsel	62811000000	07912618010000F0
2	Satelindo	62816125	059126181652
3	Excelcom	62818445009	07912618485400F9
4	Indo-M3	62855000000	07912658050000F0

Sumber : *http://brightside.wordpress.com/2006/01/20/tutorial-sms-gateway/*

2. Tipe SMS

Tipe SEND tipe SMS = 1. Jadi, bilangan heksanya adalah **01**.

3. Nomor Referensi SMS

Nomor referensi dibiarkan 0. Jadi, bilangan heksanya **00**. Selanjutnya akan diberikan sebuah nomor referensi otomatis oleh ponsel/alat SMS Gateway.

4. Nomor Ponsel penerima

Sama seperti cara menulis PDU Header untuk SMS-Centre, header ini juga terbagi atas tiga bagian, yaitu :

a. Jumlah bilangan decimal nomor ponsel yang dituju dalam bilangan heksa.

b. National/International Code.

Untuk national, kode subheader-nya adalah **81**.

Untuk international, kode subheader-nya adalah **91**.

c. Nomor ponsel yang dituju dalam pasangan heksa dibalik-balik.

Jika tertinggal satu angka heksa yang tidak memiliki pasangan, angka tersebut dipasangkan dengan huruf **F** di depannya.

Contoh :

Nomor ponsel yang dituju = 628129573337 dapat ditulis dengan dua cara sebagai berikut :

Cara 1 : 08129573337 diubah menjadi :

a. 0B : ada 11 angka

b. 81

c. 80-21-59-37-33-F7

Digabung menjadi : 0B818021593733F7

Cara 2 : 6281295733337 diubah menjadi :

d. 0C : ada 12 angka

e. 91

f. 26-81-92-75-33-73

Digabung menjadi : 0C91268192753373

5. Bentuk SMS

0 => 00 => dikirim sebagai SMS

1 => 01 => dikirim sebaga i telex

2 => 02 => dikirim sebagai fax

Dalam hal ini, pengiriman dalam bentuk SMS tentu saja memakai **00**.

6. Skema Encoding Data I/O

Ada dua skema, yaitu :

a. Skema 7 bit → ditandai dengan angka 0 → 00

b. Skema 8 bit → ditandai dengan angka lebih besar dari 0 → diubah ke heksa.

Kebanyakan ponsel/SMS Gateway yang ada di pasaran sekarang menggunakan skema 7 bit sehingga kita menggunakan kode **00**

7. Jangka Waktu Sebelum SMS Expired

Jika bagian ini di-skip, itu berarti kita tidak membatasi waktu berlakunya SMS. Sedangkan jika kita isi dengan suatu bilangan, angka yang kita berikan tersebut akan mewakili jumlah waktu validitas SMS tersebut.

8. Isi SMS

Header ini terdiri atas dua *subheader*, yaitu:

a. Panjang isi (jumlah huruf dari isi)

Misalnya : untuk kata "wahyu" → ada 5 huruf → **<u>05</u>**

b. Isi berupa pasangan bilangan heksa

Untuk ponsel/SMS Gateway berskema encoding 7 bit, jika kita mengetikkan suatu huruf dari keypad-nya, berarti kita telah membuat 7 angka I/O berturutan.

Ada dua langkah yang harus dilakukan untuk mengonversikan isi SMS, yaitu :

Langkah Pertama : mengubahnya menjadi kode 7 bit.

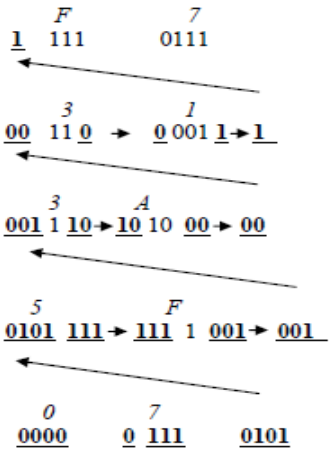
Langkah Kedua : mengubah kode 7 bit menjadi 8 bit yang diwakili oleh pasangan heksa.

Contoh : untuk kata **"wahyu**"

Langkah Pertama :

Bit	7	1
w	111	0111
a	110	0011
h	110	1000
y	111	1001
u	111	0101

Langkah Kedua :



Oleh karena total 7 bit x 5 huruf = 35 bit, sedangkan yang kita perlukan adalah 8 bit x 5 huruf = 40 bit, maka diperlukan 5 bit dummy yang diisi denganbilangan 0. Setiap 8 bit mewakili suatu angka heksa, tentu saja karena secara logika 2 pangkat 4 sama dengan 16.

Dengan demikian kata "wahyu" hasil konversinya menjadi **F7313A5F07**

Tabel 2.3. Standar Alphabet 7 Bit

				b7	0	0	0	0	1	1	1	1
				b6	0	0	1	1	0	0	1	1
				b5	0	1	0	1	0	1	0	1
b4	b3	b2	B1		0	1	2	3	4	5	6	7
0	0	0	0	0	@	Δ	SP	0	-	P	¨	p
0	0	0	1	1			!	1	A	Q	a	q
0	0	1	0	2	$	Φ	"	2	B	R	b	r
0	0	1	1	3		Γ	#	3	C	S	c	s
0	1	0	0	4		Λ		4	D	T	d	t
0	1	0	1	5		Ω	%	5	E	U	e	u
0	1	1	0	6		Π	&	6	F	V	f	v
0	1	1	1	7		Ψ	'	7	G	W	g	w
1	0	0	0	8		Σ	(	8	H	X	h	x
1	0	0	1	9		Θ	)	9	I	Y	i	y
1	0	1	0	10	LF	Ξ	*	:	J	Z	j	z
1	0	1	1	11			+	;	K	Ä	k	ä
1	1	0	0	12			,	<	L	Ö	l	ö
1	1	0	1	13	CR		-	=	M		m	
1	1	1	0	14		ß	.	>	N	Ü	n	ü
1	1	1	1	15			/	?	O		o	

Sumber : *http//www.bengkelprogram.com/cetak-artikel-175bps*

d. PDU untuk SMS terima dari SMS Centre.

Kebanyakan Header di bawah ini telah dibahas sebelumnya, kecuali beberapa yang berbeda, dijelaskan di bawah ini:

1. No. SMS – Centre

2. Tipe SMS → untuk SMS – terima = 4 → **04**

3. No Ponsel pengirim

4. Bentuk SMS

5. Skema Encoding

6. Tanggal dan waktu SMS di-stamp di SMS Centre. Di wakili oleh 12 bilangan heksa (6 pasangan) yang berarti : yy/mm/dd hh: mm: ss

Contoh : 207022512380 → 02/07/22 15:32:08 → 22 Juli 2002 15:32:08 WIB

7. batas waktu validasi→ jika tidak dibatasi dilambangkan dengan **00**

8. isi SMS Setelah mengupas satu demi satu header untuk SMS-Terima ini, maka untuk PDU dibawah ini :

07912658050000F00,04,0C91265816107398,00,00,207022512380,00,005F7313A5F07

Maka dapat diartikan sebagai berikut :

1. SMS tersebut dikirim lewat SMS-Centre: 62855000000.

2. SMS tersebut merupakan SMS terima.

3. SMS tersebut dikirim dari ponsel no.628561013789.

4. SMS tersebut diterima dalam bentuk SMS.

5. SMS tersebut memiliki skema encoding 7 bit.

6. SMS tersebut sampai di SMS-Centre pada tanggal : 22-07-02, pukul :15:32:08 WIB.

7. SMS tersebut tidak memiliki batas waktu valid.

8. SMS tersebut isinya adalah “wahyu”.

2.1.2. Perintah AT (AT *COMMAND*)

Perintah AT (*Hayes* AT *Command*) digunakan untuk berkomunikasi dengan terminal (*modem*) melalui gerbang serial pada komputer. Dengan penggunaan perintah AT, dapat diketahui atau dibaca kondisi dari terminal, seperti mengetahui kondisi sinyal, kondisi baterai, mengirim pesan, membaca pesan, menambah item pada daftar telepon, dan sebagainya. Pada tabel 1 diperlihatkan beberapa jenis perintah AT yang berhubungan dengan penanganan pesan-pesan SMS.

Tabel 2.4. Beberapa jenis perintah AT yang digunakan

AT Command	Fungsi
AT+CMGS	Mengirim pesan
AT+CMGR	Membaca pesan
AT+CMGF	Format pesan
AT+CMGD	Menghapus pesan
AT+CNMI	Prosedur indikasi pesan baru
AT+CPMS	Pemilihan target memori
AT+CSMS	Pemilihan layanan pesan

1. AT+CMGL=X Perintah AT+CMGL digunakan untuk memeriksa apakah ada pesan SMS pada handphone. Pesan SMS pada handphone antara lain :

a. X=0 untuk SMS baru pada kotak masuk (*inbox*)

b. X=1 untuk SMS lama pada kotak masuk (*inbox*)

c. X=2 untuk SMS tidak terkirim pada kotak keluar (*outbox*)

d. X=3 untuk SMS terkirim pada kotak keluar (*outbox*)

e. X=4 untuk semua pesan SMS yang ada

Misalnya untuk memeriksa pesan SMS baru pada kotak masuk (*inbox*) yaitu :

Com_cmgl :

db'AT+CMGL=0', 0

2.2. Inframerah

Sinar inframerah adalah termasuk cahaya monokromatis yang tidak tampak oleh mata manusia. Spektrum frekuensi cahaya secara umum dibagi menjadi tiga bagian yaitu :

a. Inframerah, mempunyai panjang gelombang 0,3 mm–0,7 μm.
b. Cahaya tampak, mempunyai panjang gelombang 0,7 μm – 0,4 μm.
c. *Ultra Violet*, mempunyai panjang gelombang 0,4 μm – 0,03 μm.

Gelombang elektromagnetik merupakan penyusun dari cahaya yang berada dalam spektrum elektromagnetik yang mempunyai jangkauan sangat lebar. Pada jarak yang sama, seluruh spektrum elektromagnetik tersebut mempunyai kecepatan yang sama tetapi frekuensinya berbeda sesuai dengan panjang gelombangnya.

Dalam hal ini berlaku: $e = \lambda.f$

dengan :

e = kecepatan cahaya (m/s)

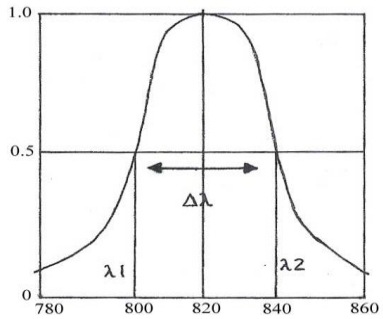
λ = panjang gelombang (m)

f = frekuensi (Hz)

Suatu spektrum frekuensi cahaya disebut inframerah jika panjang gelombangnya 0,78μm – 1000μm. Sedangkan spektrum frekuensi inframerah yang sering digunakan adalah $2{,}5.10^{14}$ Hz – $2{,}0.10^{14}$ Hz .

2.2.1. Spektrum LED (*Light Emitting Dioda*) Infra Merah

Dalam spesifikasinya sebuah LED akan memancarkan gelombang cahaya pada sebuah panjang gelombang tunggal, atau dapat dikatakan pada frekuensi tunggal. Namun sesungguhnya LED akan memancarkan gelombang cahaya pada suatu daerah panjang gelombang. Daerah ini disebut sebagai panjang gelombang. Pada umumnya LED mempunyai panjang gelombang sebesar 20 – 100 nm. Konversi antara panjang gelombang ($\Delta\lambda$) dan *bandwidth* (Δf) adalah : $\Delta f / f = \lambda / \Delta\lambda$

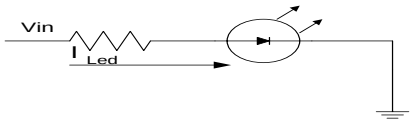


Gambar 2.3. Spektrum LED (*Light Emitting Dioda*) Infra Merah

Pada Gambar 2.3. diperlihatkan contoh spectrum sebuah LED infra merah yang beroperasi pada penjang gelombang tengah 820 nm panjang gelombang diambil pada titik separuh daya. Jadi dalam contoh tersebut diatas $\Delta\lambda = \lambda 2 - \lambda 1$ = 835 nm – 805 nm = 30 nm. Makin kecil panjang gelombang maka makin koherengelombang cahaya yang dipancarkan.

2.2.2. Tegangan dan Arus LED (*Light Emitting Dioda*)

LED menpunyai penurunan tegangan yang lazimnya 1,5 volt sampai 2,5 Volt untuk arus diantara 10mA sampai 150mA . Untuk LED dengan pancaran radiasi tampak arus yang lazim adalah 10mA sampai 50mA, sementara untuk LED infra merah arusnya sampai 150mA. Selain parameter tegangan parameter arus pada Led sangat menentukan kecerahan LED, karena LED mempunyai hambatan yang sangat kecil maka dipelukan sebuah tahanan seri terhadap LED untuk membatasi LED.



Gambar 2.4. Rangkaian LED (*Light Emitting Dioda*)
dengan Pembatas Arus R

2.3. Mikrokontroler AVR ATMega16

AVR adalah mikrokontroler RISC (Reduce Instruction Set Compute) 8 bit berdasarkan arsitektur Harvant yang dibuat oleh Atmel pada tahun 1996. AVR mempunyai kepanjangan *Advanced Versatile* RISC atau *Alf and Vegard's Risc Prosesor* yang berasal dari nama dua mahasiswa Norwergian institute of Technology (NTH), yaitu Alf-Egil Bogen dan vegard Wollan.

AVR memiliki keunggulan dibandingkan dengan mikrokontroler lain, yaitu AVR memiliki kecepatan eksekusi program yang lebih cepat, karena sebagian besar intruksi dieksekusi dalam 1 siklus *clock*, lebih cepat dibandingkan dengan mikrokontroler MCS51 yang memiliki arsitektur CISC (*Complex Instruction Set Compute*) dimana mikrokontroler MCS51 membutuhkan 12 siklus

clock untuk mengeksekusi 1 intruksi. Selain mikrokontroler AVR memiliki fitur yang lengkap antara lain :

1. ADC internal.
2. EEPROM internal.
3. *Timer/Counter*.
4. *Wancblog timer*.
5. *PWM, Port I/O*.
6. Komunikasi serial.
7. Komparator.
8. I2C, dll.

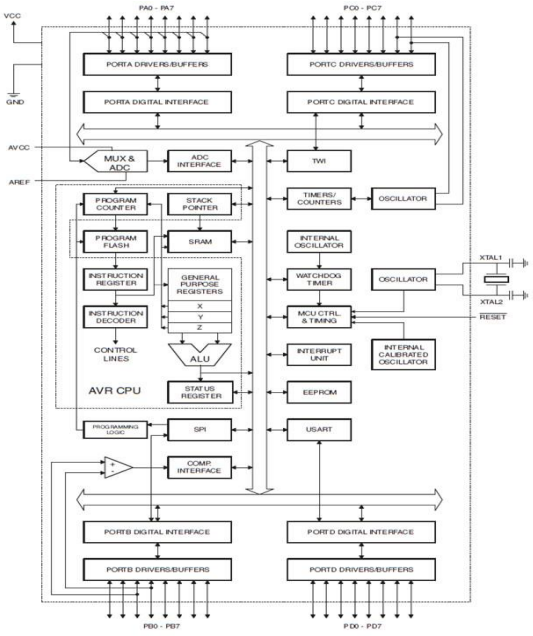
Sehingga mempermudah programer dan desainer menggunakannya untuk berbagai aplikasi sistem elektronik. Secara umum mikrokontroler AVR dapat dikelompokkan menjadi 3 kelompok, yaitu keluarga AT90Sxx, ATMega, dan ATTiny.

Tabel 2.5. Jenis mikrokontroler AVR

Mikrokontroler AVR		Memori		
Tipe	Jumlah *pin*	Flash	EEPROM	SRAM
TinyAVR	8–32	1–2K	64–128	0–128
AT90Sxx	20–44	1–8K	128–512	0–1K
ATMega	32–64	8-128K	512–4K	512–4K

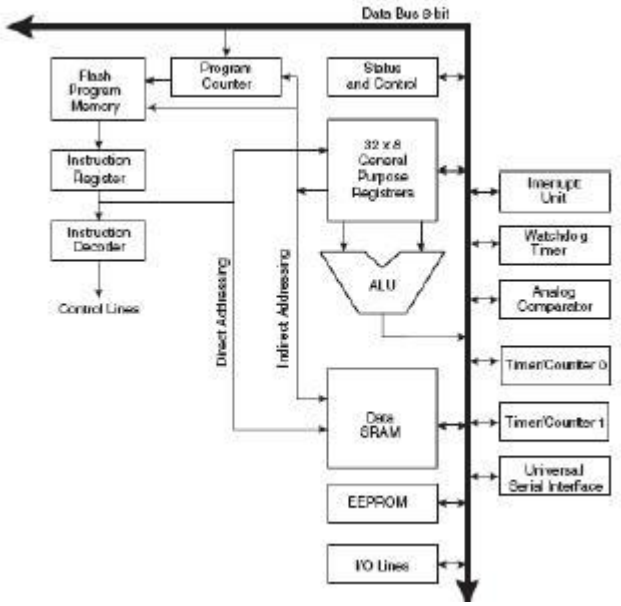
Menggunakan Bahasa C (CodeVision AVR Sumber: Heri Andrianto,2008 Pemrograman Mikrokontroler AVR ATMEGA16). Hal ,Informatika Bandung.

2.3.1. Blok Diagram



Gambar 2.5. Blok Diagram ATMega16

2.3.2. Arsitektur Mikrokontroler AVR RICS



Gambar 2.6. Arsitektur mikrokontroler AVR RICS

Dari gambar 2.6. AVR menggunakan arsitektur Harvard dengan memisahkan antara memori dan *bus* untuk program dan data sehingga memaksimalkan kemampuan dan kecepatan. Intruksi dalam memori program dieksekusi dengan *pipelining single level*. Dimana ketika satu intruksi dieksekusi, instruksi berikutnya diambil dari memori program. Konsep ini mengakibatkan intruksi dieksekusi setiap *clock cycle*.CPU terdiri dari 32x8-bit *general purpose register* yang dapat diakses dengan cepat dalam satu *clock cycle*, yang mengakibatkan operasi *Aritmatic Logic Unit* (ALU) dapat dilakukan dalam satu *cycle*. Pada operasi aritmatic dan logic pada ALU akan mengubah bit-bit yang terdapat pada suatu *Register* (SREG) secara paralel.

2.3.3. Fitur ATMega16

Fitur-fitur yang dimiliki ATMega16 adalah sebagai berikut :

1. Mikrokontroler AVR 8 *bit* yang memiliki kemampuan tinggi dengan daya rendah.
2. Arsitektur RISC dengan *thoughput* mencapai 16 MIPS pada frekuensi 16MHz.
3. Memiliki kapasitas *Flash* memori 16 *Kbyte*, EEPROM 512 *byte* dan SRAM 1 *Kbyte.*
4. Saluran I/O sebanyak 32 buah. Yaitu *Port* A, *Port* B, *Port* C, dan *Port* D.
5. CPU yang terdiri atas 32 buah *register*.
6. Unit interupsi internal dan eksternal.
7. *Port* USART untuk komunikasi serial.

8. Fitur *Peripheral*

a. Tiga buah *Timer/Counter* dengan kemampuan pembanding.

- 2 buah *Timer/Counter 8 bit* dengan *prescaler* terpisah *Mode Compare.*

- 1 buah *Timer/Counter* 16 *bit* dengan *Prescaler* terpisah, *Mode Compare*, dan *Mode capture.*

b. *Real Time Counter* dengan *Oscillator* tersendiri.

c. 4 *channel* PWM.

d. 8 *channel*, 10 bit ADC:

- 8 *Single-ended Channel*.

- 7 *Diffrential Channel* hanya pada kemasan TQFP.

- 2 *Diffrential Channel* dengan *Programmble Gain* 1x, 10x, atau 200x.

e. Byte-ortented Two-wire Serial interface.

f. *Programmable serial* USART.

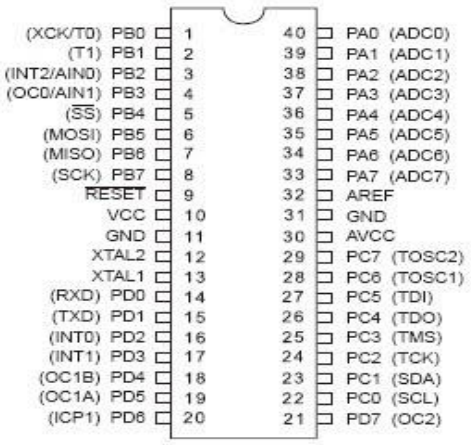
g. Antarmuka SPI.

h. *Watchdog Timer* dengan *oxcillator internal*

i. *One-chip Analog Comperator.*

2.3.4. Konfigurasi *Pin* AVR ATMega16

Konfigurasi *pin* ATMega16 dengan kemasan 40 *pin* DIP (*Dual Inline Package)* dapat dilihat pada gambar 2.7.



Gambar 2.7. konfigurasi kaki *(pin)* ATMega16

Dari gambar tersebut dapat dijelaskan fungsi dari masing-masing *pin* ATMega16 sebagai berikut :

1. VCC merupakan *pin* yang berfungsi masukan catu daya.
2. GND merupakan *pin Ground.*
3. *Port* A (PA0,PA7) merupakan pin input/output dua arah dan pin masukan ADC.
4. *Port* B (PB0,PB7), *Port* D (PD0,PD7)merupakan *Pin* input/output dua arah dan *Pin* fungsi khusus.
5. RESET merupakan *Pin* yang digunakan untuk me-reset mikrokontroler.
6. XTAL1 dan XTAL2 merupakan *Pin* masukan *clock* eksternal.
7. AVCC merupakan *Pin* masukan tegangan untuk ADC.
8. AREF merupakan *Pin* masukan tegangan refenssi ADC.

2.3.5. *Port* sebagai *input/output* digital

ATMega16 mempunyai empat buah *Port* yang bernama *Port*A, *Port*B, *Port*C, dan *Port*D. Keempat *Port* tersebut merupakan jalur bidirectional dengan pilihan internal pul-up. Tiap *Port* mempunyai tiga buah *register bit,* yaitu DDxn, PORTxn, dan PINxn. Huruf "x" mewakili nama huruf dari *Port* sebagian huruf "n" mewakili nomor *bit*. Bit DDxn terdapat pada I/O addres PINx. Bila DDxn diset 1 maka Px berfungsi sebagai *pin* output. Bila DDxn diset 0 maka Px berfungsi sebagai *pin* input. Bila PORTxn diset 1 pada saat *pin* berfungsi sebagai pin input, maka resistir *pull-up* akan diaktifkan. Untuk mematikan resistor *pull-up*, PORTxn harus diset 0 atau *pin* dikonfigurasi ssebagai *pin output. Pin port* adalah *tri-state* setelah kondisi reset. Bila PORTxn diset 1 pada saat *pin* terkonfigurasi sebagai *pin* output maka *pin port* akan berlogika 1. Dan bila PORTxn 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,PORTxn=0) kekondisi *output high* (DDxn=1,PORTxn=1) maka harus ada kondisi peralihan apakah itu kondisi *pull-up enabled* (DDxn=0. PORTxn=1) atau kondisi *output low* (DDxn=1, PORTxn=0). Biasanya, kondisi *pull-up enable* dapat diterima sepenuhnya, selama lingkungan impedansi tinggi tidak mempehatikan perbedaan antara sebuah *strong high driver* dengan sebuah *puul-up.* Jika ini bukan suatu masalah, maka *bit* PUD pada *register* SFTOR 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, PORTxn=0) atau kondisi *output high* (DDxn=1, PORTxn=0) sebagai kondisi transisi.

Tabel 2.6 Konfigurasi Pin Port

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 ifext. 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)

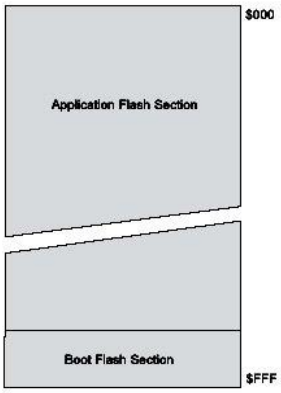
Sumber: http://ilmu computer.org/wp.content/uploads/2008/08/sholihul-atmega16.pdf...

Bit 2-PUD *pull-up Disable.* Bila *bi* diset bernilai 1 maka *pull-up* pada *port* I/O akan dimatikan walaupun *register* DDxn dan PORTxn dikonfigurasikan untuk menyalakan *pull-up* (DDxn=0, PORTxn=1).

2.3.6. Peta Memori AVR ATMega16

1. Memori Program

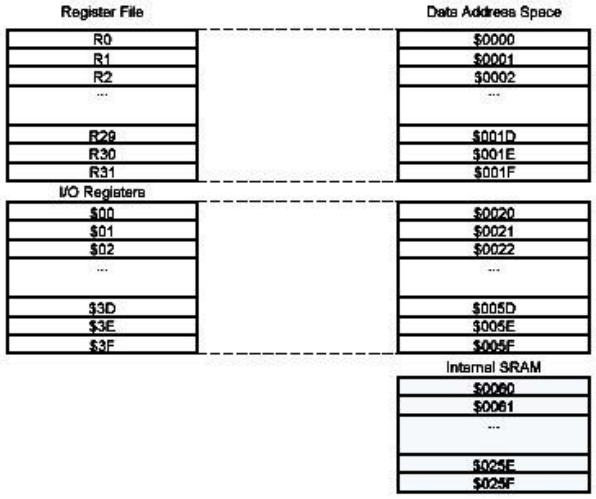
Arsitektur AVR mempunyai dua memori utama, yaitu memori data dan memori program. Selain itu, ATMega16 memiliki memori EEPROM untuk menyimpan data. ATMega16 memiliki 16k *byte on chip in-system Reprogammable Flash Memory* untuk menyimpan program. Karena semua intruksi AVR memiliki format 16 dan 32 *bit*, *Flash* diatur dalam 8K x 16 *bit.* Untuk keamanan program, memori program dibagi dalam dua bagian, yaitu bagian program *Boot* dan aplikasi. *Bootnoder* adalah program kecil yang bekerja pada saat star up time yang dapat memasukan seluruh program aplikasi dalam prosesor.



Gambar 2.8. Peta Memori Program AVR ATMega16

2. Memori Data (SRAM)

Memori data AVR ATMega16 terbagi menhjadi 3 bagian, yaitu 32 bua*h* *register* umum, 64 buah register I/O dan 1 Kbyte SRAM internal. *General purose register* menempati alamat data terbawa, yaitu $00 sampai $1F. sedaangkan memori I/O menempati 64 alamat berikutnya mulai dari $20 hingga $5F. memori I/O merupakan register yang khusus digunakan untuk mengatur fungsi terhadap berbagai peripheral mikrokontroler seperti control register, timer/counter, fungsi-fungsi I/O, dan sebagainya. 1024 alamat memori berikutnya mulai alamat $60 hingga $45 digunakan untuk SRAM internal.



Gambar 2.9. Peta Memori Data AVR ATMega16

3. Memori Data EEPROM

EEPROM *(electrically erasable programmable read only memory)* adalah memori yang masih dapat menyimpan data walaupun catu daya dimatikan. ATMega16 terdiri dari 512 *byte* memori data EEPROM 8 *bit*, data dapatditulis/dibaca dari memori ini, ketika catu daya dimatikan, data terakhir yang ditulis pada memori EEPROM masih tersimpan pada memori ini, atau dengan kata lain memori EEPROM bersifat *nonvolatile*. Alamat EEPROM mulai $000 sampai $1FF. untuk mengakses EEPROM dilakukan dengan cara menentukan EEPROM Address Register, EEPROM data register dan EEPROM kontrol Register.

2.3.7. Interupsi

Interupsi adalah kondisi dimana pada saat program utama dieksekusi atau dikerjakan oleh CPU kemudian tiba-tiba terhenti untuk sementara waktu karena ada rutin lain yang harus ditangani terlebih dahulu oleh CPU dan setelah selesai

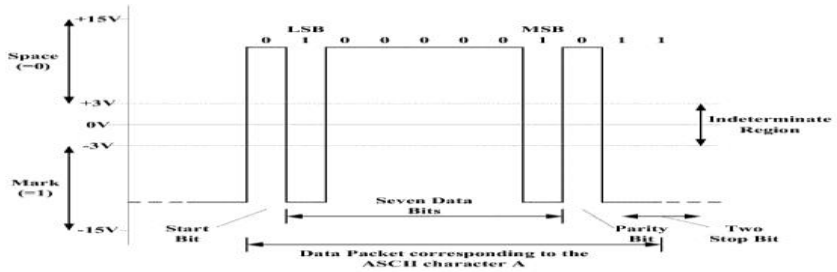
mengerjakan rutin instruksi pada program utama. ATMega16 memiliki 21 sumber interupsi seperti pada tabel.

Table 2.7. Interrupt Vector Pada ATMega16

Vector No	Program Address	Source	Interrupt Definition
1	0x0000	RESET	External Pin, Power-on Reset, Brown-out Reset and Watchdog Reset
2	0x0001	INT0	External Interrupt Request 0
3	0x0002	INT1	External Interrupt Request 1
4	0x0003	TIMER2 COMP	Timer/Counter2 Compare Match
5	0x0004	TIMER2 OVF	Timer/Counter2 Overflow
6	0x0005	TIMER1 CAPT	Timer/Counter1 Capture Event
7	0x0006	TIMER1 COMPA	Timer/Counter1 Compare Match A
8	0x0007	TIMER1 COMPB	Timer/Counter1 Compare Match B
9	0x0008	TIMER1 OVF	Timer/Counter1 Overflow
10	0x0009	TIMER0 OVF	Timer/Counter0 Overflow
11	0x000A	SPI, STC	Serial Transfer Complete
12	0x000B	USART, RXC	USART, Rx Complete
13	0x000C	USART, UDRE	USART Data Register Empty
14	0x000D	USART, TXC	USART, Tx Complete
15	0x000E	ADC	ADC Conversion Complete
16	0x000F	EE_RDY	EEPROM Ready
17	0x0010	ANA_COMP	Analog Comparator
18	0x0011	TWI	Two-wire Serial Interface
19	0x0012	INT2	External Interrupt Request 2
20	0x0013	TIMER0 COMP	Timer/Counter0 Compare Match
21	0x0014	SPM_RDY	Store Program Memory Ready

2.3.8. Komunikasi Serial USART

Komunikasi serial merupakan fitur yang penting dalam sistem *embended*, karena dengan komunikasi serial kita dapat dengan mudah menghubungkan mikrokontroler dengan peralatan lainnya. Port serial pada mikrokontroler terdiri atas dua pin, yaitu RxD dan TxD. RxD berfungsi untuk menerima data dari komputer atau peralatan lainnya, sedangkan TxD berfungsi untuk mengirim data ke komputer atau peralan lainnya. Pengiriman data serial dikirim satu per satu beserta format data serial yang umum.



Gambar 2.10. format serial

Pada mikrokontroler ATMega16, pin PD0 dan PD1 digunakan untuk komunikasi serial USART (*Universal Synchronous Asynchronous Receiver/Transmitter).* USART pada ATMega16 memiliki beberapa keuntungan dibandingkan dengan sistem UART (*Universal Asynchronous Receiver-Transmitter),* yaitu :

1. Operasi *full duplex* (mempunyai register receiver dan transmit yang terpisah).
2. Mendukung komunikasi multiprosessor.
3. Mode kecepatan transmisi bermode Mbps.

2.3.9. Inisialisasi USART

USART harus diinisialisasi sebelum komunikasi manapun dapat berlangsung. Untuk mengirim data serial menggunakan *Code VisionAVR,* untuk menggunakan fungsi *putcher, puts*, atau menggunakan I/O *register* UDR. Ada register yang perlu ditentukan nilainya, yaitu :

1. UBRR (USART *Based Rate Register*)

UBRR merupakan *register*16 bit yang berfungsi untuk menentukan kecepatan transmisi data yang akan digunakan. UBRR dibagi menjadi dua yaitu

UBRRH dan UBRRL. UBRR merupakan bit penyimpan konstanta kecepatan komunikasi serial. UBRRH menyimpan 4 bit tertinggi dan UBRRL menyimpan 8 bit sisanya. Data yang dimasukkan ke UBRRH dan UBRRL dihitung menggunakan rumus pada tabel 2.8.

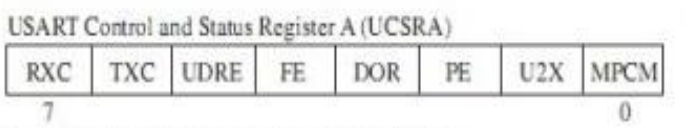
Tabel 2.8. Rumus Perhitungan UBBR

Operating Mode	Equation for Calculating Baud Rate[1]	Equation for Calculating UBRR Value
Asynchronous Normal Mode (U2X = 0)	$BAUD = \frac{f_{osc}}{16(UBRR+1)}$	$UBRR = \frac{f_{osc}}{16BAUD} - 1$
Asynchronous Double Speed Mode (U2X = 1)	$BAUD = \frac{f_{osc}}{8(UBRR+1)}$	$UBRR = \frac{f_{osc}}{8BAUD} - 1$
Synchronous Master Mode	$BAUD = \frac{f_{osc}}{2(UBRR+1)}$	$UBRR = \frac{f_{osc}}{2BAUD} - 1$

Sumber : Heri Adrianto, 2008, Pemrograman Mikrokontroler AVR ATMEGA16 Menggunakan Bahasa C (codeVision AVR). Informatika bandung.

U2X adalah *bit* pada *register* UCRA yang berfungsi untuk menggandakan *transfery rate* menjadi dua kalinya. Hanya berlaku untuk mode asinkron sedangkan untuk mode sinkron *bit* ini diset 0.

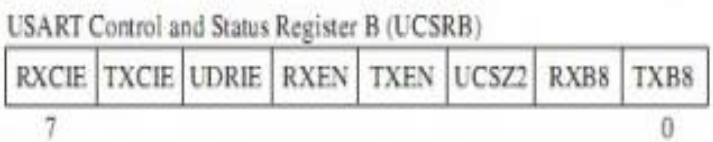
2. UCSRA (USART *Control And Status Register* A)



Gambar 2.11. UCSRA

3. USCSRB (USART *Control And Status Register* B)

USCRB merupakan register 8 bit pengatur aktivasi penerima dan pengirim USART, komposisinya sebagai berikut :



Gambar 2.12. UCSRB

Penjelasan bit pada UCSRB

RXCIE : *bit* pengatur aktivasi interupsi penerimaan data serial.

TXCIE : *bit* pengatur aktivasi interupsi pengiriman data serial.

UDRIE : *bit* pengatur aktivasi interupsi yang berhubungan dengan kondisi *bit* UDRE pada UCSRA.

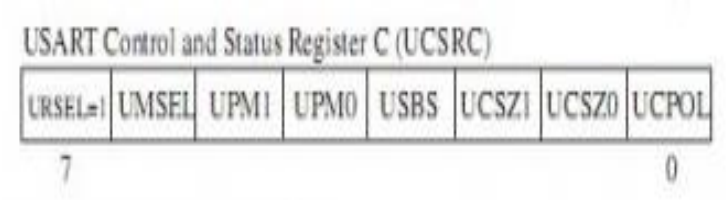
RXEN : *bit* pengatur aktivasi penerimaan serial ATMega16.

TXEN : *bit* pengatur aktivasi pengiriman serial ATMega16.

UCSZ2 bersama-sama dengan *bit* UCSZ1 dan UCSZ0 di *register* UCSZ0 menentukan ukuran karakter serial yang dikirimkan.

4. UCSRC (USART *Control And Status Register* C)

UCSRC merupakan *register* 8 *bit* yang digunakan untuk mengatur mode dan kecepatan komunikasi serial yang dilakukan. Komposisinya seperti gambar di bawah ini :



Gambar 2.13. UCSRC

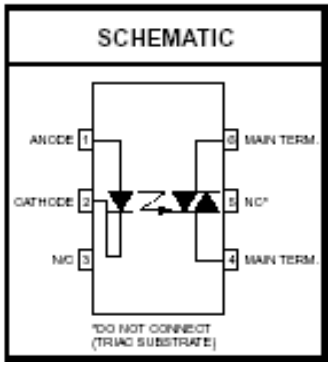
Penjelasan *bit* pada UCSRC

- URCEL : merupakan *bit* pemilih akses antara UCSRC dan UBRR.
- UMCEL : merupakan *bit* pemilih mode komunikasi serial antara sinkron dan asinkron.
- UPM : merupakan *bit* pengatur paritas.
- USBS : merupakan *bit* pemilih ukuran *bit* stop.

- UCSZ2 dan UCSZ0 merupakan *bit* pengatur jumlah karakter serial.
- UCPOL : merupakan *bit* pengater hubungan antara perubahan data keluaran dan data masukan serial dengan *clock* sinkronisasi yang hanya berlaku untuk mode sinkron dan untuk mode asinkron *bit* ini diset 0.

2.4. MOC 3021

MOC3021(optoisolator) adalah rangkaian yang menurut ilmu optik sebagai alat yang digunakan untuk atau sebagai driver triac. Alat ini berisi AlGaAs diode yang memancarkan infrared dan cahaya itu mengaktifkan silicon sebagai switch dari dua tempat, yang mana berfungsi seperti suatu triac. Alat ini dibuat atau dirancang untuk antarmuka antara kendali elektronik dan daya triac ke kendali beban resistive dan induktif untuk pengoperasian tegangan 115/240 VAC.



Gambar 2.14 Schematic MOC3021

Gambar 2.15 MOC3021 Triac Coupler Technical

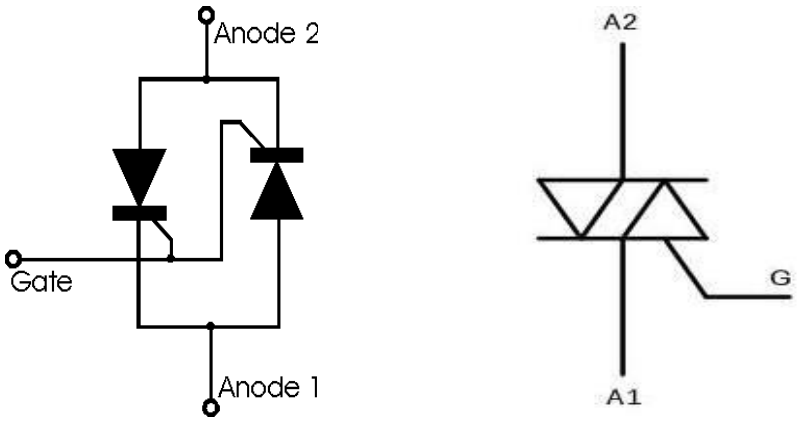
Prinsip kerja *optoisolator* adalah apabila diac yang ada didalamnya aktif maka diac akan mengalirkan arus listrik. *Optoisolator* ini aktif apabila *infrared* memberikan input pada diac. *Infrared* akan aktif apabila *infrared* mendapatkan

tegangan dan arus sebesar tegangan dan arus minimum yang dibutuhkan untuk menyalakan *infrared. Optoisolator* ini dibuat kompatibel dengan mikrokontroler karena arus yang dibutuhkan untuk mengaktifkan *infrared* dapat dipenuhi oleh mikrokontroler sehingga tidak memerlukan *driver* tambahan.

2.5. Triac

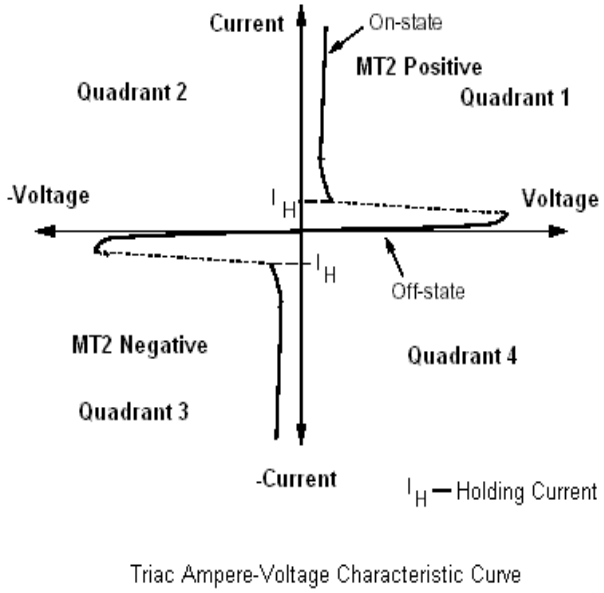
Triac adalah komponen 3 elektoda dari keluarga *thyristor*. Triac merupakan 2 buah SCR yang dihubungkan parallel berkebalikan dengan terminal *gate* bersama. Berbeda dengan SCR yang hanya melibatkan tegangan dengan polaritas positif saja, triac dapat dipicu dengan tegangan polaritas positif dan negative serta dapat dihidupkan dengan menggunakan tegangan bolak – balik yang ada pada *gate*. Triac banyak digunakan pada rangkaian pengendali dan pensaklaran.

Triac akan menghantar jika pada terminal gate diberi pemicuan yang berupa arus dengan tegangan positif dan negative. Triac akan tetap on bila arus *thermis* yang mengalir pada triac lebih besar dari arus penahan ($I_T > I_H$).



Gambar 2.16 Simbol TRIAC dan Ekivalensi

Sedangkan karekteristik TRIAC diperlihatkan pada gambar 2.17

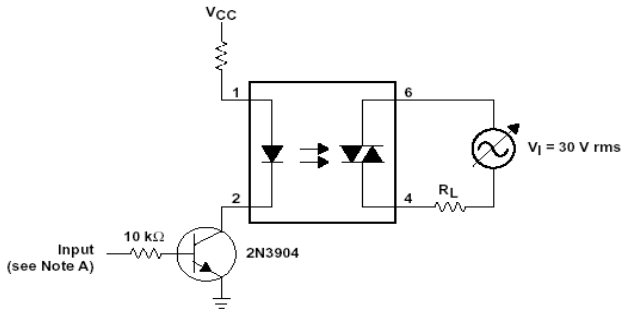


Gambar 2.17 Karakteristik TRIAC

Karena TRIAC merupakan komponen bidirectional, terminalnya tidak dapat ditentukan sebagai anoda/katode. Jika terminal MT2 positif terhadap MT1. TRIAC dapat dimatikan dengan memberikan sinyal gerbang positif antara gerbang G dan MT1. Jika terminal MT2 negatif terhadap MT1, maka TRIAC dapat dihidupkan dengan memberikan sinyal pulsa negative anrara gerbang dan terminal MT1. tidak perlu untuk memiliki kedua sinyal gerbang positif dan negative dan TRIAC dapat dihidupkan baik oleh sinyal gerbang positif maupun negative. Dalam prakteknya sensitivitas berfariasi antara satu kuadran dengan kuadran lain, dan TRIAC biasanya beroperasi dikuadran I atau kuadran III.

2.5.1. Rangkaian Driver TRIAC moc 3021

Rangkaian ini digunakan untuk mengaktifkan dan mematikan valve, driver triac yang terdiri dari IC MOC3021, Transistor 2N3904, dan resistor.



Gambar 2.18 Rangkaian driver triac

Tingkat yang kritis kenaikan off-state tegangan, dv/dt, terukur dengan masukkan pada O V. Frekwensi Vin ditingkatkan sampai phototriac menyala. Tingkat yang kritis kenaikan dari tegangan yang berganti-ganti, dv/dt©, di/terukur dengan menerapkan 5V sekali-kali berdenyut kepada terus meningkat dan masukan frekwensi Vin sampai phototriac terus tinggal setelah pulsa masukan telah berhenti. Dengan tidak ada pulsa masukan lebih lanjut, frekwensi Vin kemudian adalah secara berangsur-angsur berkurang sampai phototriac mematikan. Frekwensi dimana memadamkan terjadi boleh kemudian digunakan untuk menghitung dv/dt (c) menurut rumusan yang ditunjukkan diatas.

2.6. Bahasa C

Akar dari bahasa C adalah dari bahasa BCPL yang dikembangkan oleh Martin Richard pada tahun 1967. Bahasa ini memberikan ide kepada Ken Thomson yang kemudian mengembangkan bahasa yang disebut dengan bahasa B pada tahun 1970. Perkembangan selanjutnya dari bahasa B adalah bahasa C oleh Dennis Ricthie sekitar tahun 1970-an di Bell Telephone Laboratories Inc. (sekarang adalah AT dan T Bell Laboratories). Bahasa C pertama kali digunakan dikomputer Digital Equipment Corporation PDP-11 yang menggunakan sistem

operasi UNIX C adalah bahasa yang standar, artinya suatu program yang ditulis dengan bahasa C tertentu akan dapat dikonversi dengan bahasa C yang lain dengan sedikit modifikasi. Standar bahasa C yang asli adalah standar dari UNIX. Patokan dari standar UNIX ini diambil dari buku yang ditulis oleh Brian Kerningan dan Dennis Ritchie berjudul "The C Programming Language", diterbitkan oleh Prentice-Hall tahun 1978. Deskripsi C dari Kerninghan dan Ritchie ini kemudian dikenal secara umum sebagai "K dan RC".

2.6.1. Tipe Data

Didalam bahasa pemrograman computer, data yang digunakan umumnya dibedakan menjadi data nilai numerik dan nilai karakter. Nilai numerik dapat dibedakan lagi menjadi menjadi nilai numerik integer dan nilai numerik pecahan. Nilai numerik pecahan dapat dibedakan lagi menjadi nilai numerik pecahan ketetapan tunggal dan nilai numerik pecahan ketetapan ganda. Bahasa-bahasa pemrograman computer membedakan data kedalam beberapa tipe dengan tujuan supaya data menjadi efisien dan efektif. C menyediakan lima macam tipe data dasar, yaitu tipe data integer (niali numeric bulat yan dideklarasikan dengan int), floatingpoint (nilai numcomputer membedakan data kedalam beberapa tipe dengan tujuan supaya data menjadi efisien dan efektif. C menyediakan lima macam tipe data dasar, yaitu tipe data integer (niali numerik bulat yan dideklarasikan dengan int), floatingpoint (nilai numerik pecahan ketetapan tunggal yang dideklarasikan dengan float), double-precision (nilai numerik pecahan ketetapan ganda yang dideklarasikan dengan double).

Tabel 2.9. Tipe-tipe data dasar

Tipe	Lebar	Jangkauan Nilai	
		Dari	Sampai dengan
Int	***16 bit***	***-32768***	***32767***
Signed int			
Short int			
Signed int short in			
Unsigned int	***16 bit***	***0***	***65535***
Unsigned nt short int			
Long int	***32 bit***	***-2147483648***	***2147483649***
Signed int long int			
Unsigned int long int	***32 bit***	***0***	***4294967296***
Float	***32 bit***	***3.4E-38***	***3.4E+38***
Double	***64 bit***	***1.7E-308***	***1.7E+308***
Long Double	***80 bit***	***3.4E-4932***	***3.4E+4932***
Chart	***8 bit***	***-128***	***127***
Signed chart			
Unsigned chart	***8 bit***	***0***	***255***

Karakter (dideklarasikan dengan *char*), dan kosong (dideklarasikan dengan *void).* *Int, fload, double* dan *char* dapat dikombinasikan dengan pengubah (*modifier*) *signed, unsigned, long* dan *short,* hasil dari kombinasi tipe data ini dapat dilihat pada tabel.

2.6.2. Pengenalan Fungsi-Fungsi Dasar

Fungsi main () harus ada pada program, sebab fungsi inilah yang menjadi titik awal dan titik akhir eksekusi program. Tanda { di awal fungsi menyatakan awal tubuh fungsi dan sekaligus awal eksekusi program, sedangkan tanda } di akhir fungsi merupakan akhir tubuh fungsi dan sekaligus adalah akhir eksekusi program. Jika program terdiri atas lebih dari satu fungsi, fungsi main () biasa ditempatkan pada posisi yang paling atas dalam pendefinisian fungsi. Hal ini hanya merupakan kebiasaan. Tujuannya untuk memudahkan pencarian terhadap program utama bagi pemrograman. Jadi bukanlah merupakan suatu keharusan.

2.6.3. Pengenalan Praprosesor #*include*

include merupakan salah satu jenis pengarah praprosesor (*praprosesor directive*). Pengarah praprosesor ini dipakai untuk membaca file yang diantaranya berisi deklarasi fungsi dan definisi konstanta. Beberapa file judul disediakan dalam C. File-file ini mempunyai ciri yaitu namanya diakhiri dengan *eksekusi.h.* Misalnya pada program #*include* <*stdio.h*> menyatakan pada kompiler agar membaca file bernama stdio.h saat pelaksanaan kompilasi. Bentuk umum #*include* :

#*include"*namafile"

Bentuk pertama (#*include* <namafile>) mengisyaratkan bahwa pencarian file dilakukan pada direktori khusus, yaitu direktori file include. Sedangkan bentuk kedua (#*include*'namafile") menyatakan bahwa pencarian file dilakukan pertama kali pada direktori aktif tempat program sumber dan seandaianya tidak ditemukan pencarian akan dilanjutkan pada direktori lainnya yang sesuai dengan perintah pada sistem operasi.

2.6.4. Operasi Relasi

Operasi relasi biasa dipakai untuk membandingkan dua buah nilai. Hasil pembandingan berupa keadaan benar atau salah. Keseluruhan operasi pada C.

Tabel 2.10. Operasi Relasi

OPERATORA	MAKNA
>	Lebih dari
>=	Lebih dari atau sama dengan
<	Kurang dari
<=	Kurang dari atau sama dengan
= =	Sama dengan
! =	Tidak sama dengan

2.6.5. Operator Logika

Operator logika biasa dipakai untuk menghubungkan ekspresi relasi. Keseluruhan operator logika ditunjukkan pada tabel 2.11.

Tabel 2.11 Operator Logika

OPERATOR	MAKNA
&&	Dan (AND)
I I	Atau (OR)
!	Tidak (NOT)

2.6.6. Pernyataan Bahasa C

1. pernyataan *if*

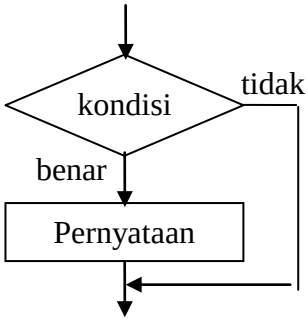
Pernyataan *if* mempunyai bentuk umum :

***If* (kondisi)**

Pernyataan;

Bentuk ini menyatakan :

a. Jika kondisi yang diseleksi adalah benar (bernilai logika = 1), maka pernyataan yang mengikutinya akan diproses.
b. Sebaliknya, jika kondisi yang diseleksi adalah tidak benar (bernilai logika = 0), maka pernyataan yang mengikutinya tidak akan diprosess. Mengenai kondisi harus ditulis diantara tanda kurung, sedangkan pernyataan dengan berupa sebuah pernyataan tunggal, pernyataan majemuk atau pernyataan kosong. Diagram alir dapat dilihat seperti gambar 2.19



Gambar 2.19 Diagram alir *if*

2. Pernyataan If-else

Pernyataan *if-else* memiliki bentuk :

***If* (kondisi)**

Pernyataan-1;

Else

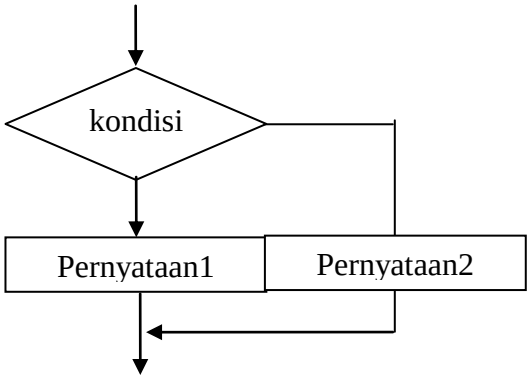
Pernyataan-2:

Arti dari pernyataan *if-else* :

a. Jika kondisi benar, maka pernyatan-1 dijalankan.

b. Sedangkan bila kondisi bernilai salah, maka pernyataan-2 yang dijalankan.

Masing-masing pernyataan-1 dan pernyataan-2 dapat berupa sebuah pernyataan tunggal, pernyatan majemuk ataupun pernyataan kosong.



Gambar 2.20. Diagram alir *if-else* Pernyataan-2

3. Pernyataan *Switch*

a. Pernyataan Switc Tunggal

Bentuk sintak dari statement *switch* tunggal adalah sebagai berikut:

Switch(kondisi)

{

Case konstanta1;

Statemen-statemen;

break;

Casekonstanta2;

Statemen-statemen;

break;

…….

Default;

Statemen-statemen;

}

b. Perulangan FOR

Pada pemrograman proses perulangan dibagi 2 bagian utama yaitu :

- Perulangan yang sudah diketahui jumlah perulangan sebelumnya perulangan tersebut dilakukan.
- Perulangan yang belum diketahui jumlah perulangannya sebelum perulangan tersebut dilakukan. Dalam hal ini dibagi menjadi dua bagian yaitu : kondisi perulangan diperiksa diawal perulangan dan kondisi perulangan diperiksa diakhir perulangan.

Bentuk perulangaan for:

For {ungkapan1; ungkapan2; ungkapan3}

{

Pernyataan ;

……..;

}

Keterangan :

- Ungkapan 1 : Digunakan untuk memberikan inisialisasi terhaadap variabel pengendali loop.
- Ungkapan 2 : Dipakai untuk kondisi keluar dari loop.
- Ungkapan 3 : Dipakai sebagai pengatur kenaikan nilai variabel pengendali loop.

2.6.7. Perulangan *While* dan *DO while*

Perulangan yang belum diketahui berapa kali akan diulangi maka dapat menggunakan *while* atau *do while.* Pada pernyataan *while*, pemeriksaan terhadap *loop* dilakukan dibagian awal (sebelum tubuh *loop*). Pernyataan *while* akan diulanhgi terus menerus selama kondisi bernilai benar, jika kondisinya salah maka perulangan dianggap selesai.

2.6.8. Lompatan

Statemen goto dapat digunakan untuk melompati dari suatu proses ke bagian proses yang lainnya di dalam program. Bentuk umum dari *Statemen goto* adalah sebagai berikut:

goto label;

2.7. CodeVisionAVR

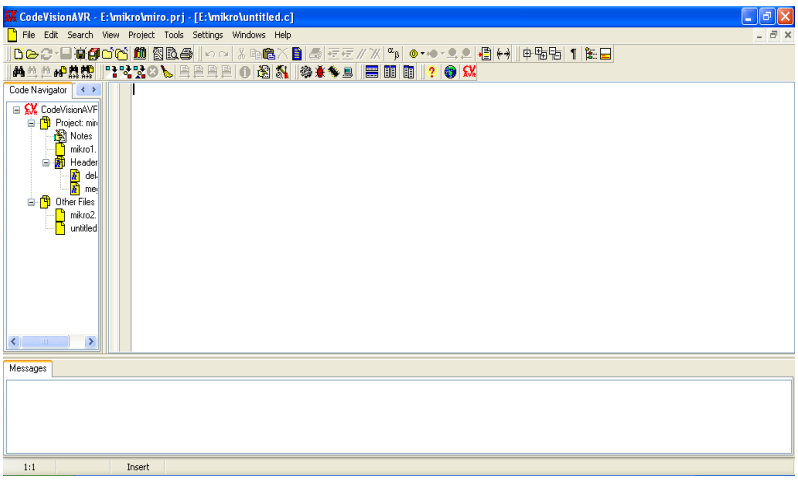
CodeVisionAVR pada dasarnya merupakan perangkat lunak pemrograman mikrokontroler keluarga AVR berbasis bahasa C. ada tiga komponen penting yang telah diintegrasikan dalam *software* ini adalah Compiler C, IDE (*Integrated Development Environment*) dan program generator.

Berdasarkan spesifikasi yang dikeluarkan oleh perusahaan pengembangannya, *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 *library* 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*).

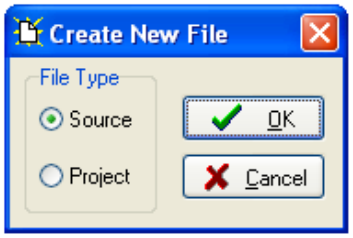
Khusus untuk library fungsi, disamping library standar (seperti fungsi-fungsi matematik, manipulasi *string*, pengaksesan memori dan sebagainya). *CodeVision AVR* juga menyediakan fungsi-fungsi tambahan yang sangat bermanfaat dalam pemrograman antarmuka AVR dengan perangkat luar yang umum digunakan dalam aplikasi kontrol. Beberapa fungsi library yang penting diantaranya adalah fungsi-fungsi mengakses LCD, komunikasi I2C, IC RTC (*Real Time Clock*), sensor suhu LM75, SPI (*Serial Peripheral Interface*) dan lain sebagainya.

Untuk memudahkan pengembangan program aplikasi, *CodeVisionAVR* juga dilengkapi IDE yang sangat mudah penggunaannya. Selain menu-menu pilihan

yang umum dijumpai pada setiap perangkat lunak berbasis Windows, *CodeVisionAVR* ini telah mengintegrasikan *software downloader* ISP (*In System Programmer*) yang dapat digunakan untuk mentransfer kode mesin hasil kompilasi kedalam sistem memori mikrokontroler AVR yang sedang diprogram.



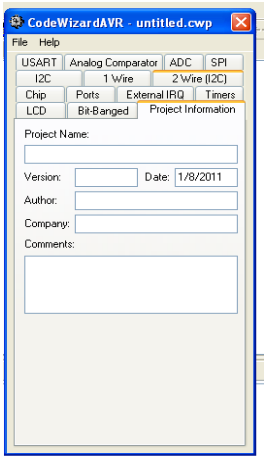
Gambar 2.21. Tampilan awal *CodeVisionAVR*



Gambar 2.22. Membuat file project baru

Selain itu, *CodeVisionAVR* juga menyediakan sebuah tool yang dinamakan dengan *Code Generator* atau C*odeWizardAVR*. Secara praktis, tool ini sangat bermanfaat membentuk sebuah kerangka program (*template*), dan juga memberi kemudahan bagi programmer dalam peng-inisialisasi register-register

yang terdapat pada mikrokontroler AVR yang sedang diprogram. Dinamakan *code generator*, karena perangkat lunak *codevision* ini akan membangkitkan kode-kode program secara otomatis setelah fase inisialisasi pada jendela *CodeWizardAVR* selesai dilakukan. Gambar 2.23. berikut memperlihatkan beberapa penggal garis kode program yang dibangkitkan secara otomatis oleh *CodeWizardAVR*. Secara teknis, penggunaan tool ini pada dasarnya hampir sama dengan *application wizard* pada bahasa-bahasa pemrograman visual untuk computer (seperti visual C, Borland Delphi, dan sebagainya).



Gambar 2.23. *Code Generator* yang dapat digunakan untuk menginisialisasi register-register pada mikrokontroler AVR