# BAB II
# LANDASAN TEORI

## 2.1 Mikrofon Elektret

Mikrofon adalah tranducer yang mengubah energi-energi akustik (gelombang suara) menjadi enegi listrik. Mikrofon Elektret adalah jenis khusus mikrofon kapasitor yang telah memiliki sumber muatan tersendiri sehingga tidak membutuhkan pencatu daya dari luar. Sumber muatan berasal dari suatu alat penyimpan muatan yang terbuat dari bahan *teflon*. Lapisan tipis teflon dilekatkan pada pelat logam statis dan mengandung muatan-muatan negatif dalam jumlah besar. Muatan-muatan tersebut terperangkap pada satu sisi yang kemudian menimbulkan medan listrik pada celah yang berbentuk kapasitor. Getaran suara yang ada mengubah tekanan udara di dalamnya sehingga membuat jarak antara diafragma dan pelat logam statis juga berubah-ubah. Akibatnya, nilai kapasitansi berubah dan tegangan terminal mikrofon pun juga berubah. Bentuk dan simbol mikrofon elektret ditunjukkan pada Gambar 2.1.

**Gambar 2.1** Bentuk Mikrofon Elektret[1]

Mikropon elektret memiliki 2 pin yaitu pin positif sebagai keluaran yang akan diukur dan pin negatif sebagai referensi pengukuran. Pin negatif juga terhubung dengan badan mikrofon.

### 2.1.1 Spesifikasi Mikrofon Elektret

Spesifikasi Mikrofon elektret ditunjukkan pada tabel 2.1

[1] https://en.wikipedia.org/wiki/Electret_microphone,diakses tanggal:5Maret2017

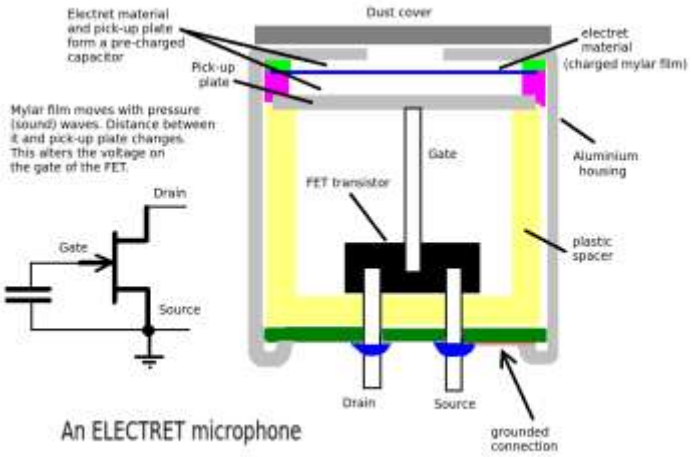
**Tabel 2.1** Tabel spesifikasi mikrofon elektret[2]

**SPECIFICATIONS**

| parameter | conditions/description | min | typ | max | units |
|---|---|---|---|---|---|
| directivity | omnidirectional | | | | |
| sensitivity (S) | f = 1 kHz, 1 Pa, 0 dB = 1 V/1 Pa | -46 | -44 | -42 | dB |
| operating voltage | | | 3 | 10 | Vdc |
| output impedance (Zout) | f = 1 kHz, 1 Pa | | 2.2 | | KΩ |
| sensitivity reduction (ΔS-Vs) | f = 1 kHz, 1 Pa, Vs = 3.0 to 2.0 Vdc | | -3 | | dB |
| frequency (f) | | 20 | | 20,000 | Hz |
| current consumption (IDSS) | Vs = 3.0 Vdc, RL = 2.2 KΩ | | | 0.5 | mA |
| signal to noise ratio (S/N) | f = 1 kHz, 1 Pa, A-weighted | | 60 | | dBA |
| operating temperature | | -20 | | 70 | °C |
| storage temperature | | -20 | | 70 | °C |

### 2.1.2 Struktur dan Bagian-Bagian Mikrofon Elektret

Mikrofon elektret tersusun dari tabung alumunium, material elektret (mylar film) sebagai diafragma, dan lapisan teflon yang melekat pada plat statis dan terhubung pada gate transistor FET internal. Struktur mikrofon elektret ditunjukkan gambar 2.2.

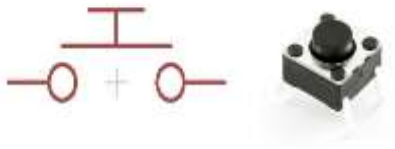



**Gambar 2.2** Struktur dan konfigurasi Mikrofon Elektret[3]

[2] http://www.cui.com/product/resource/cma-4544pf-w.pdf ,diakses tanggal:5Maret2017

## 2.2 Tombol

Tombol adalah saklar sederhana yang berfungsi untuk menghubungkan atau memutuskan aliran arus listrik dengan sistem kerja tekan tidak mengunci. Sistem kerja tidak mengunci disini berarti tombol akan bekerja sebagai perangkat penghubung atau pemutus aliran arus listrik saat tombol ditekan, lalu saat tombol tidak ditekan (dilepas), maka saklar akan kembali pada kondisi normal. Secara sederhana, tombol terdiri dari dua bilah logam yang disusun dengan konfigurasi tertentu di dalam wadah dan memiliki knop. Ketika knop ditekan, maka kedua bilah logam akan saling menempel atau terpisah (sesuai dengan konfigurasi yang digunakan), lalu saat knop dilepas kedua bilah logam akan kembali ke posisi awal. Simbol dan bentuk tombol ditunjukan pada Gambar 2.9.



**Gambar 2.3** Simbol dan bentuk tombol

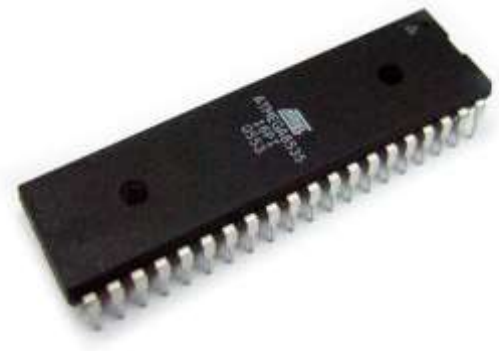
## 2.3 Mikrokontroler ATMEGA8535

Mikrokontroler adalah sebuah sistem komputer lengkap dalam satu serpih (*chip*). Mikrokontroler lebih dari sekedar sebuah mikroprosesor karena sudah memiliki ROM (*Read-Only Memory*), RAM (*Read-Write Memory*), beberapa port masukan maupun keluaran, dan beberapa *peripheral* seperti pencacah/pewaktu, ADC (*Analog to Digital converter*), DAC (*Digital to Analog converter*) dan serial komunikasi.

Salah satu mikrokontroller yang banyak digunakan saat ini yaitu mikrokontroler AVR. AVR adalah mikrokontroller RISC (*Reduce Instuction Set Compute*) 8 bit berdasarkan arsitektur Harvard. Secara umum mikrokontroler AVR dapat dikelompokkan menjadi 3 kelompok, yaitu keluarga AT90Sxx, ATMega dan ATtiny. Pada dasarnya yang membedakan masing-masing kelas adalah memori, *peripheral*, dan fiturnya. Seperti mikroprosessor pada umumnya, secara internal mikrokontroller ATMega8535 terdiri atas unit-unit fungsionalnya

[3] https://electronics.stackexchange.com/fet-impedance- electret-mic,diakses tanggal:5Maret2017

*Arithmetic and Logical Unit* (ALU), himpunan register kerja, register dan dekoder instruksi, dan pewaktu beserta komponen kendali lainnya. Berbeda dengan mikroprosessor, mikrokontroler menyediakan memori dalam serpih yang sama dengen prosesornya (*in chip*).Bentuk fisik IC mikrokontroller ATmega8535 di tunjukkan pada gambar 2.4.



**Gambar 2.4** ATMEGA8535[4]

### 2.3.1 Arsitektur ATMEGA8535

Mikrokontroller ini menggunakan arsitektur Harvard yang memisahkan memori program dari memori data, baik bus alamat maupun bus data, sehingga pengaksesan program dan data dapat dilakukan secara bersamaan (*concurrent*).
Secara garis besar mikrokontroller ATMega8535 terdiri dari :
Mikrokontroller AVR 8 bit yang memiliki kemampuan tinggi, dengan daya rendah

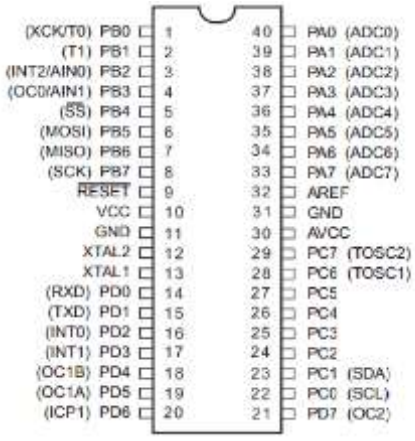
1. Arsitektur RISC dengan throughput mencapai 16 MIPS pada frekuensi 16Mhz.
2. Memiliki kapasitas Flash memori 8 Kbyte, EEPROM 512 Byte, dan SRAM 512 byte
3. Saluran I/O 32 buah, yaitu Port A, Port B, Port C, dan Port D.
4. CPU yang terdiri dari 32 buah register.
5. User interupsi internal dan eksternal
6. Port antarmuka SPI dan Port USART sebagai komunikasi serial

[4] http://store.nerokas.co.ke/index.php?route=product/category&path=135_136, diakses tanggal:5Maret2017

7. Fitur Peripheral
    - Dua buah 8-bit *timer/counter* dengan prescaler terpisah dan mode *compare*
    - Satu buah 16-bit timer/counter dengan prescaler terpisah, mode *compare*, dan mode *capture*
    - *Real time counter* dengan osilator tersendiri
    - Empat kanal PWM dan Antarmuka komparator analog
    - 8 kanal, 10 bit ADC
    - *Byte-oriented Two-wire Serial Interface*
    - *Watchdog timer* dengan osilator internal

### 2.3.2 Konfigurasi Pin ATMEGA8535

Konfigurasi *pin* mikrokontroler Atmega8535 dengan kemasan 40 pin dapat dilihat pada Gambar 2.5. Dari gambar tersebut dapat terlihat ATMega8535 memiliki 8 pin untuk masing-masing *Port A*, *Port B*, *Port C*, dan *Port D*.




**Gambar 2.5** Port-port Atmega8535[5]

### 2.3.3 Deskripsi Mikrokontroler ATMEGA8535

- VCC (*Power Supply*) dan GND (*Ground*)

[5] http://www.atmel.com/images/doc2466.pdf, diakses tanggal:5Maret2017

VCC merupakan pin yang berfungsi sebagai masukan catu daya terminal positif

- GND (*Ground*)

  GND merupakan pin *ground*

- Port A (PA7..PA0)

Port A berfungsi sebagai *input* analog pada konverter A/D. Port A juga sebagai suatu port I/O 8-bit dua arah, jika A/D konverter tidak digunakan. Pin-pin port dapat menyediakan resistor *internal pull-up* (yang dapat dipilih untuk masing-masing bit). Port A *output* buffer mempunyai karakteristik simetris antara kemampuan sink dan kemampuan source. Ketika pin PA0 ke PA7 digunakan sebagai input maka secara eksternal ditarik rendah jika resistor *internal pull-up* diaktifkan. Pin port A menjadi kondisi tri-stated ketika reset menjadi aktif.

- Port B (PB7..PB0)

Port B adalah suatu port I/O 8-bit dua arah dengan resistor *internal pull-up* (dapat diset per bit). Port B output buffer mempunyai karakteristik simetris antara kemampuan sink dan kemampuan source. Sebagai input, pin port B secara eksternal ditarik rendah jika resistor *pull-up* diaktifkan. Pin port B adalah tri-stated ketika reset aktif.

- Port C (PC7..PC0)

Port C adalah suatu port I/O 8-bit dua arah dengan resistor *internal pull-up* (dapat diset per bit). Port C output buffer mempunyai karakteristik simetris antara kemampuan sink dan kemampuan source. Sebagai input, pin port C secara eksternal ditarik rendah jika resistor *pull-up* diaktifkan. Pin port C adalah tri-stated ketika reset aktif.

- Port D (PD7..PD0)

Port D adalah suatu port I/O 8-bit dua arah dengan resistor *internal pull-up* (dapat diset per bit). Port D output buffer mempunyai karakteristik simetris antara kemampuan sink dan kemampuan source. Sebagai input, pin port D secara eksternal ditarik rendah jika resistor *pull-up* diaktifkan. Pin port D adalah tri-stated ketika reset aktif.

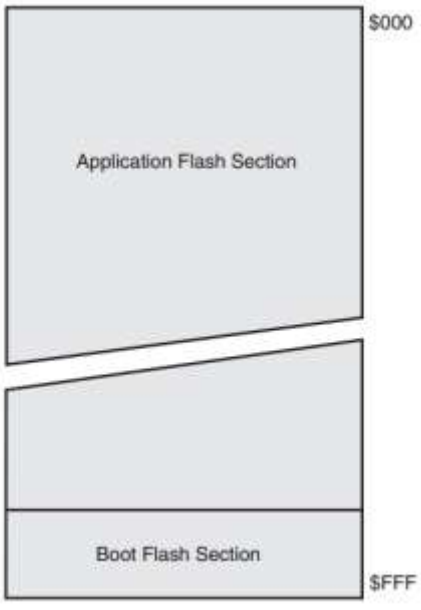
- RESET (*Reset input*)
- XTAL1 (*Input Oscillator*)

- XTAL2 (*Output Oscillator*)
- AVCC adalah pin penyedia tegangan untuk Port A dan Konverter A/D.
- AREF adalah pin referensi analog untuk konverter A/D.

### 2.3.4 Struktur Memori ATMega8535

#### 2.3.4.1 Memori Program

Arsitektur ATMega8535 mempunyai dua memori utama, yaitu memori data dan memori program. Selain itu, ATMega8535 memiliki memori EEPROM untuk menyimpan data. ATMega8535 memiliki 8K byte *On-chip In-System Reprogrammable Flash Memory* untuk menyimpan program. Instruksi ATMega8535 semuanya memiliki format 16 atau 32 bit, maka memori *flash* diatur dalam 4K x 16 bit. Memori *flash* dibagi kedalam dua bagian, yaitu bagian program *boot* dan aplikasi seperti terlihat pada Gambar 2.6. *Bootloader* adalah program kecil yang bekerja pada saat sistem dimulai yang dapat memasukkan seluruh program aplikasi ke dalam memori prosesor. Gambar peta memori program ATmega8535 AVR ditunjukkan pada gambar 2.6.

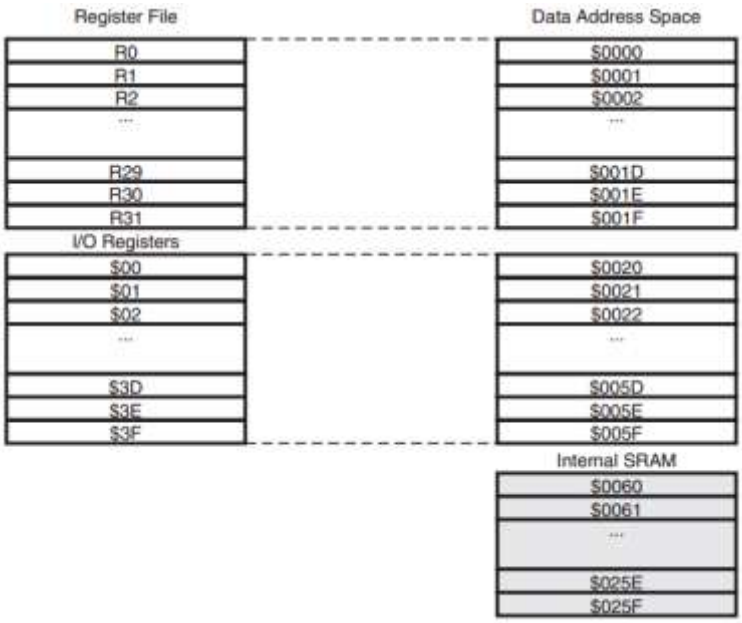



**Gambar 2.6** Peta Memori Program AVR ATMEGA8535[6]

[6] http://www.atmel.com/images/doc2466.pdf, diakses tanggal:5Maret2017

### 2.3.4.2 Memori Data (SRAM)

Memori data AVR ATMega8535 terbagi menjadi 3 bagian, yaitu 32 register umum, 64 buah register I/O dan 512 Kbyte SRAM internal. *General purpose Register* menempati alamat data terbawah, yaitu $00 sampai $1F. Sedangkan 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 fitur mikrokontroler seperti kontrol register, *timer/counter*, fungsi-fungsi I/O, dan sebagainya. 512 alamat berikutnya mulai dari $60 hingga $25f digunakan untuk SRAM internal. Gambar peta memori data ATmega8535 ditunjukkan pada gambar 2.7.




**Gambar 2.7** Peta Memori Data AVR ATMEGA8535[7]

### 2.3.4.3 Memori Data EEPROM

ATmega8535 terdiri dari 512 byte memori data EEPROM 8 bit,data dapat tulis/baca 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 dari $000 sampai $1FF.

[7] http://www.atmel.com/images/doc2466.pdf, diakses tanggal:5Maret2017

## 2.4 SRAM Eksternal 62256

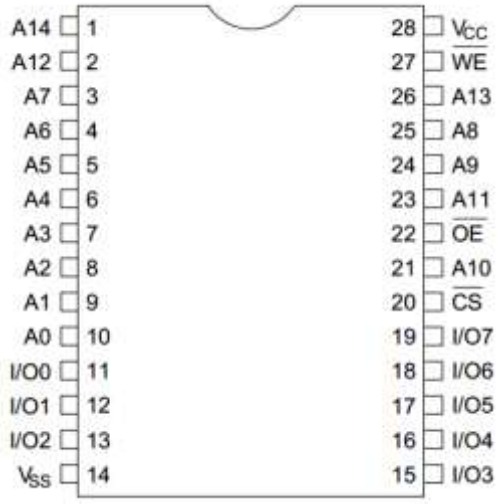
Static Random Acces Memory (SRAM) adalah suatu perangkat elektrik penyimpan data yang bersifat sementara. Data yang tersimpan pada SRAM akan hilang jika SRAM terputus dari sambungan catu daya. SRAM adalah salah satu jenis RAM yang memiliki sifat statis. Berbeda dengan Dynamic Random Acces Memory (DRAM) yang memerlukan siklus penyegaran dalam interval waktu tertentu, isi dari SRAM tidak akan berubah selama catu daya tetap terhubung dan tidak ada perubahan input data. SRAM juga memiliki kecepatan akses yang sangat tinggi dibandingkan dengan jenis RAM lainnya, oleh karena itu SRAM sering digunakan sebagai chace memory atau buffer sebelum data dimasukkan ke dalam ROM.

SRAM yang digunakan pada sistem ini bertipe HM62256ALP keluaran Hitachi. HM62256ALP memiliki fitur sebagai berikut :

1. Kapasitas memory : 32Kbyte
2. Kecepatan akses tinggi 70nS (max)
3. Konsumsi daya rendah.Keadaan standby: 1µW. Keadaan beropersai: 25mW (frekuensi=1 Mhz)
4. Catu daya tunggal : 5V
5. Tidak memerlukan osilator
6. Satu jalur data input dan output dengan 3 kondisi output
7. Kompatibel dengan level tegangan TTL

### 2.4.1 Konfigurasi Pin HM62256ALP

Konfigurasi *pin* SRAM HM62256ALP dengan kemasan 28pin dapat dilihat pada Gambar 2.8. Dari gambar tersebut dapat terlihat HM62256ALP memiliki 15 pin jalur alamat (A0-A14) dan 8 pin jalur data (I/O0-I/O7).




**Gambar 2.8** konfigurasi pin HM62256ALP

### 2.4.2 Deskripsi pin SRAM HM62256ALP

- VCC (*Power Supply*)

  Berfungsi sebagai masukan catu daya positif 5VDC.
- VSS (ground)

  Berfungsi sebagai masukan ground.
- A0~A14

  Berfungsi sebagai jalur alamat 15 bit (32Kbyte).
- I/O0~I/O7

  Berfungsi sebagai jalur data 8 bit.
- /WE

  Berfungsi sebagai input write enable pada proses penulisan memory.
- /OE

  Berfungsi sebagai pin control output pada proses pembacaan maupun penulisan memory.
- /CS

  Berfungsi sebagai input pemilih chip, yaitu jika pin tersebut berlogika 0, maka memory beroperasi secara normal, namun jika pin tersebut berlogika 1, maka memory dalam kondisi standby.

## 2.5 EEPROM Eksternal W27E040

Electrically Erasable Programmable Read Only Memory (EEPROM) adalah suatu perangkat elektrik penyimpan data yang bersifat permanen. Data

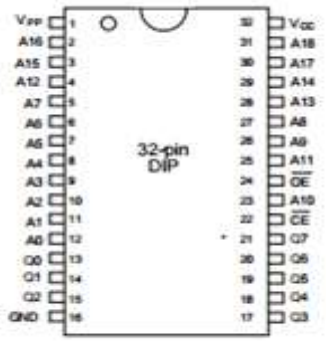
yang tersimpan pada EEPROM tidak akan hilang jika SRAM terputus dari sambungan catu daya.

EEPROM yang digunakan pada sistem ini bertipe W27E040-12 keluaran Windbond. W27E040-12 memiliki fitur sebagai berikut :

1. Kapasitas memory : 512Kbyte
2. Dapat hapus dan deprogram dengan algoritma yang sederhana.
3. Kecepatan akses tinggi 120nS (max)
4. Konsumsi daya rendah.Keadaan standby: 5μA. Keadaan beropersai: 30mA
5. Catu daya tunggal : 5V
6. Memerlukan variasi tegangan Vpp 14VDC pada mode erase dan 12 VDC pada mode program
7. Tidak memerlukan osilator
8. Satu jalur data input dan output dengan 3 kondisi output
9. Kompatibel dengan level tegangan TTL

### 2.5.1 Konfigurasi Pin W27E040-12

Konfigurasi *pin* EEPROM W27E040-12 dengan kemasan 32pin dapat dilihat pada Gambar 2.9. Dari gambar tersebut dapat terlihat W27E040-12 memiliki 19 pin jalur alamat (A0-A18) dan 8 pin jalur data (Q0-Q7).




**Gambar 2.9** konfigurasi pin W27E040-12

### 2.5.2 Deskripsi pin SRAM HM62256ALP

- VCC (*Power Supply*)

  Berfungsi sebagai masukan catu daya positif 5VDC.

- GND (ground)

Berfungsi sebagai masukan ground.

- A0~A18

  Berfungsi sebagai jalur alamat 18 bit (512Kbyte).

- Q0~Q7

  Berfungsi sebagai jalur data 8 bit.

- Vpp

  Berfungsi sebagai input catu daya pada proses pemrograman atau penghapusan memory. Vpp memerlukan catu daya 14VDC pada mode ERASE, 12VDC pada mode PROGRAM, dan 5VDC pada mode normal.

- /OE

  Berfungsi sebagai pin control output pada proses pembacaan maupun penulisan memory.

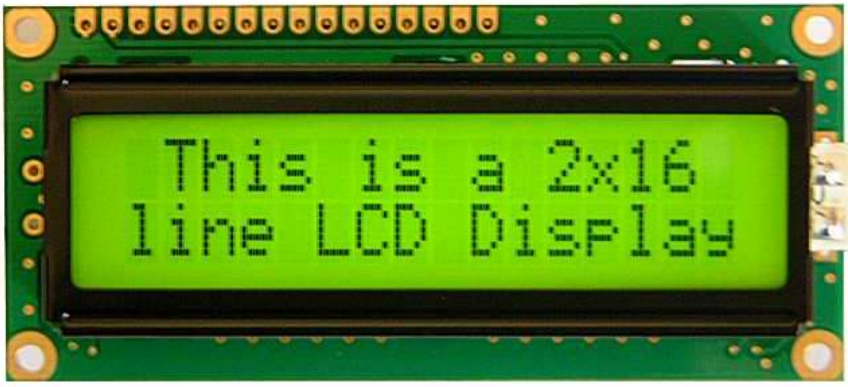
- /CE

  Berfungsi sebagai input pemilih chip, yaitu jika pin tersebut berlogika 0, maka memory beroperasi secara normal, namun jika pin tersebut berlogika 1, maka memory dalam kondisi standby.

## 2.6 LCD Karakter 2x16

Liquid Crystal Display (LCD) adalah suatu perangkat elektronika yang dirancang sedemikan rupa, sehingga dapat menampilkan tulisan maupun gambar. LCD banyak digunakan sebagai layar tampilan pada berbagai jenis aplikasi elektronika, seperti monitor komputer, televisi, seluler dan lain-lain sebagainya. LCD yang khusus hanya untuk menampilkan tulisan disebut LCD karakter. LCD 16X2 terdiri dari 16 kolom dan 2 baris.

LCD memiliki suatu *controller* yang berfungsi untuk mengontrol tampilan layar LCD. *Controller* pada modul LCD menerima instruksi dan data dari suatu prosesor atau mikrokontroller untuk menentukan karakter apa yang akan ditampilkan pada layar LCD. Pada umumnya LCD 16X2 mampu mengerjakan seluruh instruksi yang didukung oleh *controller* jenis HD44780. Instruksi-intruksi untuk modul LCD tersebut dapat dilihat di datasheet yang disediakan oleh pabrik pembuatan. Tampilan fisik LCD 16 x 2 ditunjukkan pada gambar 2.10.




**Gambar 2.10** LCD 16 x 2[8]

Modul LCD pada umumnya terdiri dari 14 pin, tetapi LCD yang memiliki *backlight* mempunyai 16 pin, yaitu 2 pin tambahan untuk menyalakan LED *backlight*. Fungsi pin LCD 16x2, ditunjukkan seperti pada tabel 2.2.

**Tabel 2.2** Fungsi Pin LCD Karakter 16x2[9]

| PIN | Nama | Fungsi |
|---|---|---|
| 1 | VSS | Ground |
| 2 | VCC | +5V |
| 3 | VEE | Contrast Voltage |
| 4 | RS | Register Select<br>0 = Send Instruction<br>1 = Send Data |
| 5 | R/W | Read/Write, to choose write or read mode<br>0 = Write Mode<br>1 = Read Mode |
| 6 | EN | Enable Signal<br>0 = start to lacht data to LCD character<br>1 = disable |
| 7 | DB0 | Data bit ke-0 H/L (LSB) |
| 8 | DB1 | Data bit ke-1 H/L |
| 9 | DB2 | Data bit ke-2 H/L |

[8] http://maxembedded.com/2011/06/lcd-interfacing-with-avr/, diakses tanggal:5Maret2017
[9] Heri Andrianto: *Pemrograman Mikrokontroler AVR Atmega16 Menggunakan Bahasa C (CodeVisionAVR)* (Bandung: Informatika Bandung, 2013), 79

| | | |
|---|---|---|
| 10 | DB3 | Data bit ke-3 H/L |
| 11 | DB4 | Data bit ke-4 H/L |
| 12 | DB5 | Data bit ke-5 H/L |
| 13 | DB6 | Data bit ke-6 H/L |
| 14 | DB7 | Data bit ke-7 H/L (MSB) |
| 15 | ANODE | Backlight (+) |
| 16 | KATODE | Backlight (-) |

Cara mengirimkan instruksi untuk dieksekusi oleh *controller* LCD:

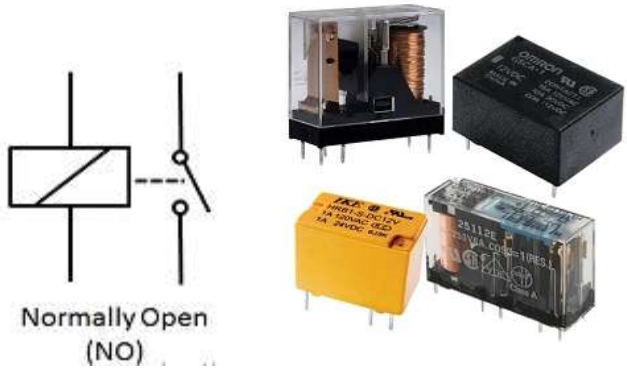
1. Set supaya pin RS = 0, R/W = 0, EN = 1.
2. Kemudian kirim data berupa instruksi untuk dieksekusi *controller* pada LCD melalui DB0 - DB7 (pin 7 – pin 14).
3. Set supaya pin EN = 0, kemudian berikan *delay* sesaat, dan set kembali pin EN = 1

Cara mengirimkan karakter atau data untuk dicetak pada layar LCD:

1. Set supaya pin RS = 1, R/W = 0, EN = 1.
2. Kemudian kirimkan data berupa ASCII dari karakter yang ingin ditampilkan pada layar LCD melalui jalur DB0 – DB7 (pin7 - pin14).
3. Set supaya pin EN = 0, kemudian berikan *delay* sesaat, dan set kembali pin EN = 1.

## 2.7 Relay

Relay adalah suatu peranti yang menggunakan prinsipelektromagnet untuk mengoperasikan seperangkat kontak sakelar. Susunan paling sederhana terdiri dari kumparan kawat penghantar yang dililit pada inti besi. Bila kumparan ini dienergikan, medan magnet yang terbentuk menarik tuas berporos yang digunakan sebagai pengungkit mekanisme sakelar magnet. Relay digunakan untuk menggerakkan beban dengan daya yang besar dengan daya pengendali yang lebih kecil. Relay juga berfungsi sebagai isolator antara jaringan beban dengan jaringan pengendali. Symbol dan bentuk relay ditunjukan pada Gambar 2.11.




**Gambar 2.11** Simbol relay dan bentuk relay[10]

## 2.8 Bahasa Pemrograman Mikrokontroler AVR

Pengembangan sebuah sistem menggunakan mikrokontroler AVR buatan ATMEL menggunakan *software* AVR STUDIO dan CodeVision AVR. AVR STUDIO merupakan *software* khusus untuk bahasa *assembly* yang mempunyai fungsi sangat lengkap, yaitu digunakan untuk menulis program, kompilasi, simulasi dan *download* program ke IC mikrokontroler AVR. Sedangkan CodeVision AVR merupakan *software C-cross compiler*, dimana program dapat ditulis dalam bahasa C, CodeVision memiliki IDE (Integreted Development Environment) yang lengkap, dimana penulisan program, *compile*, *link*, pembuatan kode mesin (*assembler*) dan *download* program ke *chip* AVR dapat dilakukan pada CodeVision, selain itu ada fasilitas terminal, yaitu untuk melakukan komunikasi serial dengan mikrokontroler yang sudah diprogram. Proses *download* program ke IC mikrokontroler AVR dapat menggunakan *downloader* secara ISP *(In-System Programming)*. *In-System Programming flash on-chip* mengizinkan memori program untuk diprogram ulang dalam sistem menggunakan hubungan serial SPI.

### 2.8.1 Bahasa C

Berikut ini penjelasan aturan penulisan program dalam bahasa C. Untuk seterusnya, pemrograman mikrokontroler AVR menggunakan bahasa C dengan penjelasan sebagai berikut:

Penulisan program dalam bahasa C

[10] http://teknikelektronika.com/pengertian-relay-fungsi-relay,diakses tanggal:5Maret2017

```
#include<mega16.h>
#include <delay.h>
#define        IRsensor        PINA.0
#define        pompa           PORTB.0

//variable global
unsigned int I,j;

void main(void)
{
//variable local
Chart data_rx;
DDRA=0x00;
PORTA=0xFF;
DDRB=0xFF;
PORTB=0x00;
….
….
While(1)
{
…….
…….
};
}
```

Penjelasan:

*Preprocessor* (#) : digunakan untuk memasukan (*include*) *text* dan *file* lain, mendefinisikan macro yang dapat mengurangi beban kerja pemrograman dan meningkatkan *legibility source code* (mudah dibaca).

#define : digunakan untuk mendefinisikan *macro*

| | | | |
|---|---|---|---|
| Contoh | #define | ALFA | 0xff |
| | #define | SUM(a,b) | a+b |
| | #define | sensor | PINA.2 |
| | #define | pompa | PORTB.0 |

Komentar :

penulisan komentar untuk beberapa baris komentar sekaligus

/*….komentar….*/

Penulisan untuk satu baris saja

//….komentar….

### 2.8.1.1 *Identifiers*

*Identifier* adalah nama yang diberikan pada *variable***,** fungsi, label, atau objek lain. *Identifier* dapat mengandung huruf (A …Z, a …z) danangka (0 … 9) dan karakter ( _ ). *Identifiers* bersifat *Case sensitive*. *Indetifier* dapat mencapai maksimal 32 karakter.

### 2.8.1.2 Konstanta

Konstanta *integer* dan *long integer* ditulis dalam bentuk *decimal* (1234), dalam bentuk biner (0b101001), *hexadecimal* mempunyai awalan 0x (0xff), atau Dalam *octal* dengan awalan 0 (0777).

*Unsigned integer* mempunyai akhiran U(10000U)

*Long integer* mempunyai akhiran L(99L)

*Unsigned long integer* mempunyai akhiran UL(99UL)

*Floating point* mempunyai akhiran F(1.234F)

Konstanta karakter harus di lingkungi oleh tanda kutip (‘a’)

### 2.8.1.3 Tipe Data

**Tabel 2.3** Tipe data[11]

| Tipe | Ukuran (*bit*) | Rentang (*Range)* |
|---|---|---|
| Bit | 1 | 0, 1 (tipe data bit hanya dapat digunakan untuk variable global) |
| Char | 8 | -128 to 127 |
| Unsigned char | 8 | 0 to 255 |
| signed char | 8 | -128 to 127 |
| Int | 16 | -32768 to 32767 |

[11] Heri Andrianto, Op.Cit., 25

| | | |
|---|---|---|
| Short int | 16 | -32768 to 32767 |
| Unsigned int | 16 | 0 to 65535 |
| Signed int | 16 | -32768 to 32767 |
| Long int | 32 | -2147463648 to 2147483647 |
| Unsigned long int | 32 | 0 to 4294967295 |
| Signed long int | 32 | -2147483648 to 2147483647 |
| Float | 32 | ±1.175e-38 to ±3.402e38 |
| Double | 32 | ±1.175e-38 to ±3.402e38 |

### 2.8.1.4 *Operator*

**Tabel 2.4** Daftar Operator Kondisi[12]

| Operator Kondisi | Keterangan |
|---|---|
| < | Lebih kecil |
| <= | Lebih kecil atau sama dengan |
| > | Lebih besar |
| >= | Lebih besar sama dengan |
| = = | Sama dengan |
| ! = | Tidak sama dengan |

**Tabel 2.5** Daftar Operator Aritmatika[13]

| Operator Aritmatika | Keterangan |
|---|---|
| + | Penjumlahan |
| - | Pengurangan |
| * | Perkalian |
| / | Pembagian |
| % | Sisa bagi(modulus) |

**Tabel 2.6** Daftar Operator Logika[14]

| Operator Logika | Keterangan |
|---|---|
| ! | *Boolean* NOT |
| && | *Boolean* AND |
| \| \| | *Boolean* OR |

[12] Ibid., 26
[13] Loc.Cit.
[14] Loc.Cit.

**Tebel 2.7** Daftar Operator *Bitwise*[15]

| Operatot ***Bitwise*** | **Keterangan** |
|---|---|
| ~ | Komplemen *Bitwise* |
| & | *Bitwise* AND |
| \| | *Bitwise* OR |
| ^ | *Bitwise Exclusive* OR |
| >> | *Right Shift* |
| << | *Left Shift* |

**Tabel 2.8** Daftar Operator *Assignment*[16]

| Operator ***Assignment*** | **Keterangan** |
|---|---|
| = | Untuk memasukkan nilai |
| += | Untuk menambah nilai dari keaadan semula |
| -= | Untuk mengurangi nilai dari keadaan semula |
| *= | Untuk mengalikan nilai dari keadaan semula |
| /= | Untuk melakukan pembagian terhadap bilangan semula |
| %= | Untuk memasukkan nilai sisa bagi dari pembagin bilangan semula |
| <<= | Untuk memasukkan *shift left* |
| >>= | Untuk memasukkan *shift Right* |
| &= | Untuk memasukkan nilai *bitwise* AND |
| ^= | Untuk memasukkan nilai *bitwise* XOR |
| \= | Untuk memasukkan nilai *bitwise* OR |

### 2.8.1.5 Program Control

**a. Percabangan**

Perintah *if* dan *if … else …*

Perintah if dan *if … else …* digunakan untuk melakukan operasi percabangan bersyarat. Fungsi-fungsi untuk menetapkan kondisi dapat dilihat dalam tabel.

Sintaks penulisan *if* dapat ditulis sebagai berikut:

***if(<expression>)<statement>;***

Sintak perintah *if ... else ...* dapat ditulis sebagai berikut:

---

[15] Loc.Cit.
[16] Ibid., 27

***if(<expression>)<statement>;***

***else<statement2>;***

jika hasil *testing expression* memberikan hasil tidak nol *statement1* akan dilaksanakan. Pada keadaan sebaliknya *statement2* yang akan dilaksanakan. Sebaiknya pemanfaatan perinta *if* untuk beberapa kondisi dilakukan dengan menggunakan blok-blok

**Percabangan *switch***

Perintah percabangan ***if … else …*** dapat digantikan dengan perintah ***switch.***

Dalam pernyataan ***switch***, sebuah variable secara berurutan diuji oleh beberapa konstanta bilangan bulat atau konstanta karakter. Sintaks perintah ***switch*** dapat ditulis sebagai berikut:

```
Switch(variable)
{
 Case konstanta_1:  statement;
                    Break;
 Case konstanta_2:  statement;
                    Break;
 Case konstanta_n:  statement;
                    Break;
 Default:           statement;
}
```

Hal-hal yang perlu diperhatikan:

1. *Switch* hanya dapat memeriksa *variable* terhadap sebuah konstanta, sedangkan *if* dapat memeriksa persyaratan perbandingan (lebih besar, lebih kecil, dan seterusnya).
2. Tidak ada dua konstanta yang sama di dalam sebuah *switch.*
3. Perintah *switch* jika dimanfaatkan dengan tepat dapat memberikan hasil yang lebih baik daipada perintah *if ... else ...* yang membentuk tangga dan/atau bersarang.

**b. *Looping* (Pengulangan)**

*Looping* adalah pengulangan satu atau beberapa perintah sampai mencapai keadaan tertentu. Ada tiga perintah *looping,* yaitu: *for…, while…,* dan *do…while….* Sintaks *loop for* dapat dituliskan sebagai berikut:

***for***

untuk pengulangan yang melakukan proses *increment*

```
for(nama_variable= nilai_awal;syarat_loop;nama_variable + +)
{Statement_yang_diulang;}
//untuk pengulangan yang melakukan proses decrement
for(nama_variable=nilai_awal;syarat_loop;nama_variable - -)
{Statement_yang_diulang;}
```

Syarat_loop adalah pernyataan rasional yang menyatakan syarat berhentinya pengulangan, biasanya berkaitan dengan *variable control*, *nama_variable*++ dan *nama_variable*--, menyatakan proses *increment* dan proses *decrement* pada *variable* kontrol.

***While***

Perintah ***while*** dapat melakukan ***looping*** apabila persyaratannya benar. Sintaks perintah ***while*** dapat dituliskan sebagai berikut:

```
Nama_variable = nilai_awal;
while(syarat_loop)
{Statement_yang_akan_diulang;
Nama_variable++;}
```

***do …while***

Perintah *while* terlebih dahulu melakukan pengujian persyaratan sebelum melakukan *looping.* Kadang-kadang hal ini menimbulkan keropotan-kerepotan yang tidak perlu, misalnya inisialisasi *variable control.* Salah satu solusi adalah dengan menggunakan loop *do …while.*

**Nama_variable = nilai_awal;**
***do***
**{statement_yang _akan_diulang;**
**nama_variable + +;}**
**while(syarat_loop)**

### 2.8.1.6 *Array*

***Array*** adalah deretan variable yang berjenis sama dan mempunyai nama yang sama. Setiap anggota deretan (elemen) diberi nomor yang disebut indeks, dimulai dari indeks nol. ***Array*** diatur agar mempunyai lokasi memori yang bersebelahan dengan alamat terkecil menujuk elemen ***array*** pertama dan alamat terbesar menunjukkan elemen terakhir. Elemen ***array*** dapat diakses dengan menggunakan indeksnya. Bentuk deklarasi array adalah:

**Tipe nama_array[ukuran]**
**Int nilai[100];**
**Nilai[1]=10;**
**Niali[2]=3;**

### 2.8.1.7 Fungsi

Sebuah program yang besar dapat dipecah menjadi bebepara subprogram yang terpisah yang melakukan fungsi tertentu. Subprogram yang seperti itu disebut fungsi. Sebagai contoh, sebuah program yang melakukan proses pengisian data berulang kali dapat dilengkapi dengan sebuah ***fungsi*** yang bertugas untuk melakukan proses pengisian data. Apabila program hendak melakukan proses pengisian data, program dapat melakukan pemanggilan ***fungsi*** tersebut.

***Fungsi*** adalah sebuah blok yang melingkupi beberapa perintah. Deklarasi ***fungsi*** dapat dilakukan dengan cara:

**Tipe nama_fungsi(argumen)**

Parameter dalam ***fungsi*** dijelaskan sebagai berikut:

**Tipe** adalah nilai yang dihasilkan oleh ***fungsi***, jika tidak dinyatakan, hasil ***fungsi*** dianggap bertipe **integer.** Deklarasi tipe ***void*** dapat dimanfaatkan untuk menghindari terjadinya nilai balik.

**Argumen** adalah deklarasi variable apa saja yang dibutuhkan ***fungsi*** dan bersifat ***optional***.

**a. Fungsi dengan nilai balik**

Fungsi ini memberikan hasil yang berupa nilai

Fungsi dengan nilai *balik (return value)*.

Contoh:

```
long luas( )
{int sisi=10;
return (sisi*sisi);}
```

**b. Fungsi tanpa nilai balik**

Fungsi ini tidak memberikan hasil yang berupa nilai melainkan berupa sebuah proses. Fungsi ini bertipe **void**.

Contoh:

```
void kedip( )
{PORTD=0;
Delay_ms(500);        //delay 500 ms
PORTD=255;
delay_ms(500);
PORTD=0;
delay_ms(500);
return;}
```

**Pernyataan *return***

Pernyataan ***return*** dapat menyatakan dua hal:

1. ***return*** mengakhiri jalanan ***fungsi*** dan kembali ke program utama.

2. Mengirim nilai balik.

***Fungsi*** dapat ditulis pada akhir program dengan membuat sebuah ***prototype function*** dibagian awal program.Cara menulis ***fungsi*** yang seperti itu memberikan kemudahan kepada ***programmer*** untuk memeriksa dan menbaca ulang sebuah program yang besar.

**c. Parameter dalam sebuah fungsi**

Parameter dalam subuah fungsi dibagi dua yaitu ***parameter formal*** dan ***parameter actual. Parameter formal*** adalah parameter pada saat fungsi itu dibuat, sedangkan ***parameter actual*** adalah parameter yang terdapat pada saat pemanggilan fungsi.

Contoh:

```
//mendeklarasikan fungsi
long func(int param_1, int param_2);
//mendefinisikan fungsi,param_1 dan param_2 disebut parameter formal
long func(int param_1, int param_2)
{return param_1*param_2;}
//memanggil fungsi dan mengisikan nilai yang dihasilkan ke sebuah variabel x, nilai 20 dan 30 disebut parameter actual
x=func(20, 30); // mengisikan parameter _1=20 dan parameter_2=30
//hasil perkaliannya disimpan ke variable x
```