BAB II

LANDASAN TEORI

2.1. Kontroler PID (Proporsional, Integral, Derivatif)

Pengendali PID (Proporsional Integral Derivatif), merupakan gabungan dari tiga sistem kendali yang bertujuan untuk mendapatkan keluaran dengan *risetime* yang tinggi dan galat yang kecil. Seperti yang kita ketahui bahwa sistem kendali Proporsional memiliki keunggulan yaitu *risetime* yang cepat tetapi sangat rentan dengan *overshot/undershot*, sistem kendali *integral* memiliki keunggulan untuk meredam galat, sedangkan sistem kendali Derivatif memiliki keunggulan untuk memperkecil *delta error* atau meredam *overshot/undershot*. PID berdasarkan implementasinya dibedakan menjadi analog dan digital, PID analog diimplementasikan dengan komponen elektronika *resistor*, *capacitor*, dan *operational amplifier*, sedangkan PID digital diimplementasikan secara program.

PID digital pada dasarnya merupakan suatu proses dari suatu program yang dijalankan dengan menggunakan komputer. Dalam prosesnya nilai yang kita masukkan *(setting point)*, dan nilai hasil pembacaan sensor saat ini *(present value)* diproses sehingga galat yang didapatkan sama dengan 0 (nol), atau nilai *setting point* sama dengan *present value*. Untuk dapat mengimplementasikan kendali PID pada sistem digital, maka PID harus diubah kedalam persamaan diskrit.

Berikut ini formula PID saat menggunakan e(t) sebagai *error function*, untuk kontroler PID penuh adalah:

$$R(t) = K_p \cdot \left[e(t) + 1/T_I \,.\, \int_0^t e(t)dt + T_D \cdot de(t)/dt \right] \qquad (2.1)$$

Kemudian, kita tulis ulang dengan mesubstitusikan T_D dan T_I, jadi kita mendapatkan P, I dan D. Ini sangat penting, untuk menyesuaikan secara eksperimen untuk mencapai nilai relatif dari ketiganya yaitu P, I dan D. Kita tulis kembali formula dengan mensubstitusikan $Q_I = K_p / T_I$ dan $Q_D = K_p \cdot T_D$ sehingga persamaan menjadi

$$R(t) = K_p \cdot e(t) + Q_I . \int_0^t e(t)dt + Q_D \cdot de(t)/dt \qquad (2.2)$$

Menggunakan diskritisasi yang sama sebagai kontroler PI, kita akan mendapatkan:

$$R_n = K_p \cdot e_n + Q_I . t_{delta} . \sum_{i=1}^{n} \frac{e_i + e_{i-1}}{2} + Q_D / t_{delta} . (e_n - e_{n-1}) \qquad (2.3)$$

Kemudian, dengan menggunakan perbedaan antara output kontroler berikutnya, akan menghasilkan :

$$R_n - R_{n-1} = K_p \cdot (e_n - e_{n-1}) + Q_I . t_{delta} . (e_n - e_{n-1})/2 + Q_D . t_{delta} . (e_n - 2e_{n-1} + e_{n-2}) \qquad (2.4)$$

Akhirnya (mensubstitusikan $Q_1 . T_{delta}$ dengan K_I dan $Q_D / . T_{delta}$ dengan K_D).

$$R_n = R_{n-1} + K_p \cdot (e_n - e_{n-1}) + K_I . (e_n - e_{n-1})/2 + K_D (e_n - e_{n-1} + e_{n-2}) \qquad (2.5)$$

Dimana :

R_n : *Output*
R_{n-1} : Output sebelumnya
K_p : konstanta P
K_I : konstanta I
K_D : konstanta D
e_n : error sekarang
e_{n-1} : error sebelumnya
e_{n-2} : error dua kali sebelumnya

Program 1 menunjukkan bagian program untuk kontroler PD, sedangkan Program 2 menunjukkan program kontroler PID keseluruhan.

a. Program 1. Kerangka program kontroler PD

```
1 static int e_old=0;
2 ...
3 e_func = v_des - v_act;                    /* error function */
4 deriv  = e_old - e_func;                   /* diff. of error fct. */
5 e_old  = e_func; /* store error function */
6 r_mot  = Kp*e_func + Kd*deriv;             /* motor output */
7 r_mot = min(r_mot, +100);                  /* limit output */
8   r_mot = max(r_mot, -100);                /* limit output */
```

b. Program 2. Kerangka program kontroler PID

```
1 static int r_old=0, e_old=0, e_old2=0;
2 ...
3 e_func = v_des - v_act;
4 r_mot  = r_old + Kp*(e_func-e_old) + Ki*(e_func+e_old)/2
5                + Kd*(e_func - 2* e_old + e_old2);
6 r_mot = min(r_mot, +100);   /* limit output */
7 r_mot = max(r_mot, -100);   /* limit output */
8 r_old = r_mot;
9 e_old2 = e_old;
10 e_old  = e_func;
```

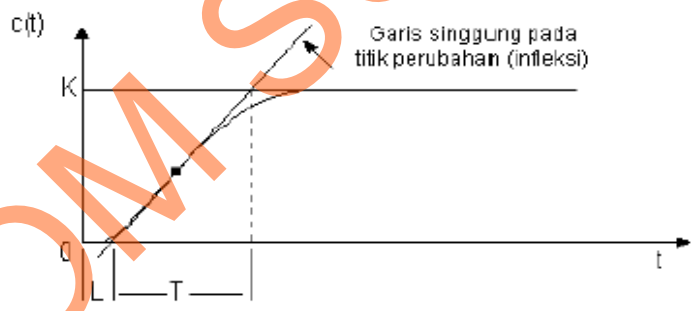
Sumber : Embedded Robotic 2006

2.1.1. *Tuning* PID

Aspek yang sangat penting dalam desain kendali PID ialah penentuan parameter kendali PID supaya sistem kalang tertutup memenuhi kriteria performansi yang diinginkan (Wicaksono, 2004). Adapun metode *tuning* kendali PID yang sudah banyak dan sering digunakan adalah Ziegler-Nichols dan Cohen-Coon.

a). Metode Ziegler-Nichols

Ziegler-Nichols pertama kali memperkenalkan metodenya pada tahun 1942. Metode ini memiliki dua cara yaitu metode osilasi dan kurva reaksi. Kedua metode ditujukan untuk menghasilkan respon sistem dengan lonjakan maksimum sebesar 25%. Metode kurva reaksi didasarkan terhadap reaksi sistem kalang terbuka. Plant sebagai kalang terbuka dikenai sinyal *step function*. Kalau plant minimal tidak mengandung unsur *integrator* ataupun *pole-pole* kompleks, reaksi sistem akan berbentuk S. Gambar 1 menunjukkan kurva berbentuk S tersebut. Kelemahan metode ini terletak pada ketidakmampuannya untuk menangani plant *integrator* maupun plant yang memiliki *pole* kompleks. Kurva berbentuk S mempunyai dua konstanta, waktu mati (*dead time*) L dan waktu tunda T. Dari Gambar 2.1, terlihat bahwa kurva reaksi berubah naik setelah selang waktu L.



Gambar 2.1. Kurva respon berbentuk S

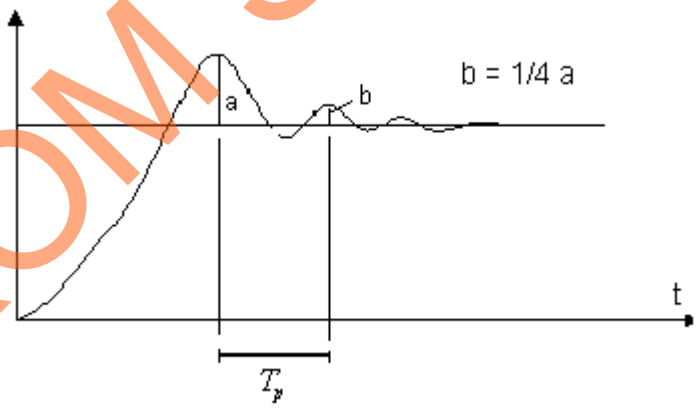
Sedangkan waktu tunda menggambarkan perubahan kurva setelah mencapai 66% dari keadaan mantapnya. Pada kurva dibuat suatu garis yang bersinggungan dengan garis kurva. Garis singgung itu akan memotong dengan sumbu absis dan garis maksimum. Perpotongan garis singgung dengan sumbu absis merupakan ukuran waktu mati, dan perpotongan dengan garis maksimum merupakan waktu tunda yang diukur dari titik waktu L. Tabel 2.1 merupakan rumusan penalaan parameter PID berdasarkan cara kurva reaksi.

Tabel 2.1. Penalaan parameter PID dengan metode kurva reaksi

Tipe Kendali	Kp	Ti	Td
P	T/L	~	0
PI	0,9 T/L	L/0,3	0
PD			
PID	1,2 T/L	2L	0,5L

b). Metode Cohen-Coon

Karena tidak semua proses dapat mentolerir keadaan osilasi dengan amplitudo tetap, Cohen-Coon berupaya memperbaiki metode osilasi dengan menggunakan metode quarter amplitude decay. Respon loop tertutup sistem, pada metode ini, dibuat sehingga respon berbentuk *quarter amplitude decay*. *Quarter amplitude decay* didefinisikan sebagai respon transien yang amplitudonya dalam periode pertama memiliki perbandingan sebesar seperempat (1/4), untuk lebih jelasnya dapat dilihat pada Gambar 2.2.



Gambar 2.2. Kurva respon *quarter amplitude decay*

Pada kendali Proporsional Kp ditala hingga diperoleh tanggapan *quarter amplitude decay*, periode pada saat tanggapan ini disebut Tp dan parameter Ti dan Td dihitung dari hubungan KP dengan TP. Sedangkan penalaan parameter kendali PID adalah sama dengan yang digunakan pada metode Ziegler-Nichols. Selain

cara tersebut, metode Cohen-Coon ini bisa dihitung dengan aturan praktis yang parameter-parameter plantnya diambil dari kurva reaksi yang terdapat pada tabel 2.2.

Tabel 2.2 Penalaan parameter PID dengan metode Cohen-Coon

Tipe Kendali	Kp	Ti	Td
P	$\frac{1}{K}\left(\frac{T}{L}\right)\left[1+\frac{1}{3}\left(\frac{L}{T}\right)\right]$	-	-
PI	$\frac{1}{K}\left(\frac{T}{L}\right)\left[0{,}9+\frac{1}{12}\left(\frac{L}{T}\right)\right]$	$L\left[\frac{30+3\left(\frac{L}{T}\right)}{9+20\left(\frac{L}{T}\right)}\right]$	-
PD	$\frac{1}{K}\left(\frac{T}{L}\right)\left[\frac{5}{4}+\frac{1}{6}\left(\frac{L}{T}\right)\right]$	-	$L\left[\frac{6-2\left(\frac{L}{T}\right)}{22+3\left(\frac{L}{T}\right)}\right]$
PID	$\frac{1}{K}\left(\frac{T}{L}\right)\left[\frac{4}{3}+\frac{1}{4}\left(\frac{L}{T}\right)\right]$	$L\left[\frac{32+6\left(\frac{L}{T}\right)}{13+8\left(\frac{L}{T}\right)}\right]$	$L\left[\frac{4}{11+2\left(\frac{L}{T}\right)}\right]$

2.2. Mikrokontroller ATmega32

Mikrokontroller dan *microprocessor* mempunyai beberapa perbedaan. *Microprocessor* yang terdapat pada komputer seperti Intel Pentium, hanya dapat bekerja apabila terdapat komponen pendukung seperti RAM (*Random Access Memory*), *hard disk*, *motherboard*, perangkat I/O, dll. Komponen-komponen tersebut diperlukan karena *microprocessor* hanya dapat melakukan pengolahan data, namun tidak dapat menyimpan data, menyimpan program, menerima

masukan dari *user* secara langsung, ataupun menyampaikan data hasil pemrosesan ke keluaran. Berbeda dengan *microprocessor*, mikrokontroller sudah dilengkapi dengan komponen-komponen yang dikemas dalam satu *chip* seperti memori, perangkat I/O, *timer*, ADC (*Analog to Digital Converter*), dll. Hal ini membuat mikrokontroller lebih tepat untuk digunakan pada aplikasi *embedded system*. (Husanto, 2008)

Mikrokontroller yang digunakan pada proyek ini adalah mikrokontroller keluarga AVR yang mempunyai arsitektur 8-bit RISC (*Reduce Instruction Set Compute*) produksi ATMEL yaitu ATmega32. Salah satu kelebihan arsitektur RISC dari arsitektur CISC (*Complex Instruction Set Compute*) adalah kecepatan waktu eksekusi tiap instruksi. Sebagian besar instruksi RISC dieksekusi dalam waktu satu *clock cycle*, sedangkan pada CISC sebagian besar instruksi dieksekusi dalam waktu dua belas *clock cycle*.

Beberapa fitur yang dimiliki ATmega32 adalah sebagai berikut (ATMEL, 2011):

a. Mempunyai kinerja tinggi dengan konsumsi daya yang rendah
b. *Fully static operation*
c. Kinerja mencapai 16 MIPS (*Millions Instruction per Seconds*) pada osilator dengan nilai frekuensi 16 MHz
d. Memiliki kapasitas memori *Flash* sebesar 32 kByte, EEPROM (*Electrically Erasable Programmable Read-Only Memory*) sebesar 1024 Byte, dan SRAM (*Static Random-Access Memory*) sebesar 2 kByte
e. Memiliki 32 jalur I/O
f. Memiliki 2 buah *Timer*/*Counter* 8-bit dan 1 buah *Timer*/*Counter* 16-bit

g. Memiliki 4 kanal PWM (*Pulse Width Modulation*)

h. Memiliki 8 kanal ADC 10-bit

i. Memiliki antarmuka: *Two-wire Serial Interface*, USART (*Universal Synchronous Asynchronous Receiver/Transmitter*), SPI (*Serial Peripheral Interface Bus*)

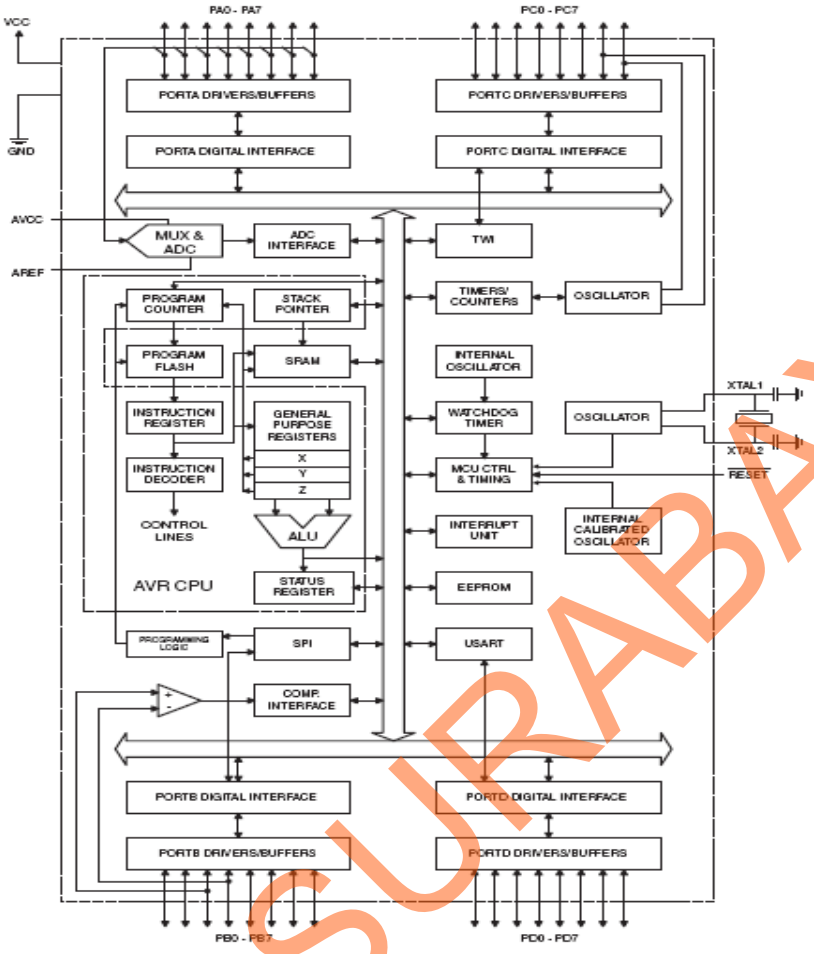
j. Memiliki *Watchdog Timer* dengan osilator internal yang terpisah

k. Memiliki *Comparator* tegangan analog

l. Memiliki unit interupsi eksternal dan internal

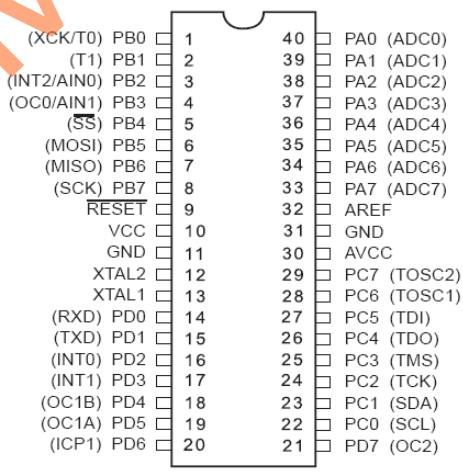
m. Bekerja pada tegangan 4.5 V – 5.5 V dengan konsumsi arus maksimal 15 mA (dengan osilator 8 MHz, tegangan 5 V dan suhu pada rentang -40 °C - 85 °C).

Proses pemrograman ATmega32 dilakukan menggunakan fitur ISP (*In-System Programmable*) melalui antarmuka SPI (*Serial Peripheral Interface*). Fitur ISP memungkinkan untuk melakukan proses *download* program ke dalam mikrokontroller tanpa bantuan mikrokontroller *master* seperti proses *download* program pada mikrokontroller AT89C51. *File* dengan ekstensi ".hex", yaitu kode program yang telah di-*compile* akan dikirimkan secara serial ke mikrokontroller untuk ditulis ke dalam memori *Flash* melalui jalur SPI yaitu *pin* MISO (*Master In Slave Out*), MOSI (*Master Out Slave In*), dan SCK (*Serial Clock*) yang digunakan sebagai sinyal sinkronasi komunikasi. Diagram blok ATmega32 terdapat pada Gambar 2.3, sedangkan konfigurasi pin ATmega32 terdapat pada Gambar 2.4.



Sumber : ATMEL 2011

Gambar 2.3. Blok diagram ATmega32



Sumber : ATMEL 2011

Gambar 2.4. Konfigurasi pin ATmega32 (ATMEL, 2011)

2.2.1. Fungsi–fungsi Pin pada ATmega32

a. VCC : Sumber tegangan +5V DC (*Direct Current*). (pin 10)

b. GND : Pin yang dihubungkan dengan *ground* sebagai referensi untuk VCC. (pin 11 dan pin 31)

c. *Port* A (PA0..PA7) merupakan pin I/O dua arah dan pin masukan tegangan analog untuk ADC

d. *Port* B (PB0..PB7) merupakan pin I/O dua arah dengan fungsi alternatif, seperti yang terlihat pada Tabel 2.1 di bawah

Tabel 2.3. Fungsi alternatif *Port* B

Pin	*Alternate Functions*
PB7	SCK (SPI *Bus Serial Clock*)
PB6	MISO (SPI *Bus Master Input/Slave Output*)
PB5	MOSI (SPI *Bus Master Output/Slave Input*)
PB4	$\overline{SS}$ (SPI *Slave Select Input*)
PB3	AIN1 (*Analog Comparator Negative Input*) OC0 (*Timer/Counter*0 *Output Compare Match Output*)
PB2	AIN0 (*Analog Comparator Positive Input*) INT2 (*External Interrupt* 2 *Input*)
PB1	T1 (*Timer/Counter*1 *External Counter Input*)
PB0	T0 (*Timer/Counter*0 *External counter Input*) XCK (USART *External Clock Input/Output*)

Sumber: ATMEL (2011)

e. *Port* C (PC0..PC7) merupakan pin I/O dua arah dengan fungsi alternatif, seperti yang terlihat pada Tabel 2.4 di bawah

Tabel 2.4. Fungsi alternatif *Port* C

Pin	*Alternate Functions*
PC7	TOSC2 (*Timer Oscillator Pin* 2)
PC6	TOSC1 (*Timer Oscillator Pin* 1)
PC5	TDI (JTAG *Test Data In*)
PC4	TDO (JTAG *Test Data Out*)
PC3	TMS (JTAG *Test Mode Select*)
PC2	TCK (JTAG *Test Clock*)
PC1	SDA (*Two-wire Serial Bus Data Input/Output Line*)
PC0	SCL (*Two-wire Serial Bus Clock Line*)

Sumber: ATMEL (2011)

f. *Port* D (PD0..PD7) merupakan pin I/O dua arah dengan fungsi alternatif, seperti yang terlihat pada Tabel 2.5 di bawah

Tabel 2.5 Fungsi alternatif *Port* D

Pin	*Alternate Functions*
PD7	OC2 (*Timer/Counter*2 *Output Compare Match output*)
PD6	ICP (*Timer/Counter*1 *Input Capture*)
PD5	OC1A (*Timer/Counter*1 *Output Compare* A *Match Output*)
PD4	OC1B (*Timer/Counter*1 *Output Compare* B *Match Output*)
PD3	INT1 (*External Interrupt* 1 *Input*)
PD2	INT0 (*External Interrupt* 0 *Input*)
PD1	TXD (USART *Output Pin*)
PD0	RXD (USART *Input Pin*)

Sumber: ATMEL (2011)

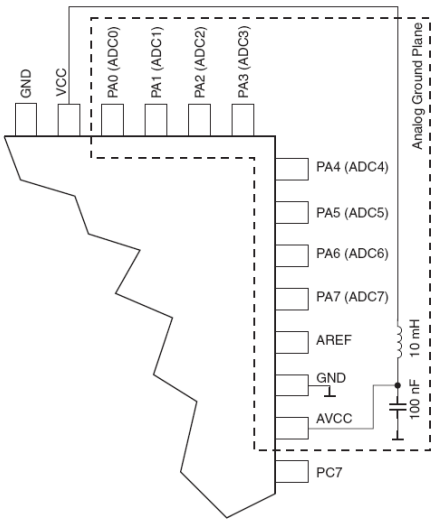
g. $\overline{RESET}$: Masukan untuk *reset* (*active low*). Jika diberikan kondisi *low* paling tidak selama 1.5 µS akan menghasilkan kondisi *reset* pada mikrokontroller meskipun mikrokontroller tidak mendapat *clock* dari osilator. (pin 9)

h. XTAL1 : Masukan ke penguat osilator. Pin ini dihubungkan dengan kristal atau sumber osilator yang lain. (pin 13)

i. XTAL2 : Keluaran dari penguat osilator. Pin ini dihubungkan dengan kristal atau *ground*. (pin 12)

j. AVCC : Pin yang digunakan untuk memberikan sumber tegangan pada *Port* A. Pin ini harus tetap dihubungkan dengan VCC meskipun fitur ADC tidak digunakan. Apabila fitur ADC digunakan, maka pin AVCC harus dihubungkan dengan VCC melalui *low-pass filter* seperti yang terlihat pada Gambar 2.4. (pin 30)

k. AREF : Pin yang digunakan sebagai masukan tegangan referensi untuk ADC. (pin 32).



Sumber : ATMEL 2011

Gambar 2.5 Koneksi AVCC dengan VCC melalui *low-pass filter*

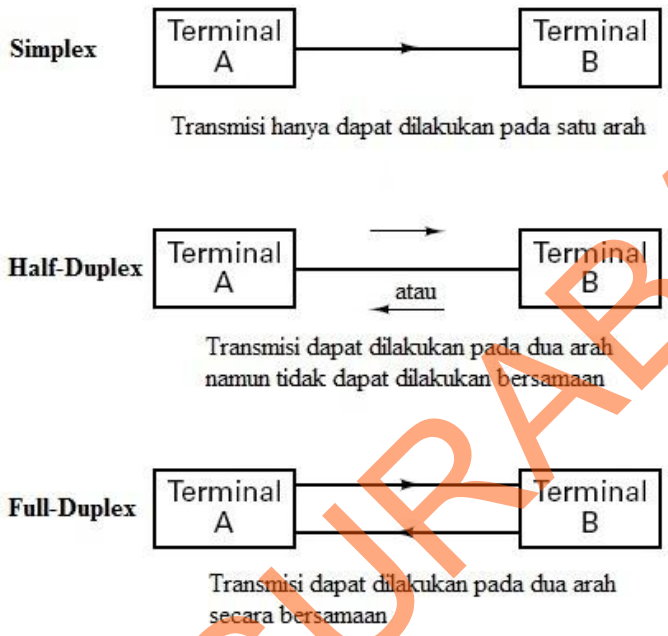
2.2.2. USART

Menurut Winoto (2008) USART dapat difungsikan sebagai transmisi data sinkron dan asinkron. Sinkron berarti *transmitter* dan *receiver* mempunyai satu sumber *clock* yang sama. Sedangkan asinkron berarti *transmitter* dan *receiver* yang mempunyai sumber *clock* yang berbeda.

Menurut Mazidi (2000) transmisi data secara serial adalah transmisi data dimana data tersebut akan dikirimkan sebanyak satu bit dalam satu satuan waktu. Terdapat dua cara dalam mentransmisikan data secara serial, yaitu secara *synchronous* dan *asynchronous*. Perbedaan dari kedua cara tersebut adalah sinyal *clock* yang dipakai sebagai sinkronisasi pengiriman data.

Transmisi secara *synchronous* yaitu pengiriman data serial yang disertai dengan sinyal *clock*, sedangkan *asynchronous* yaitu pengiriman data serial yang tidak disertai sinyal *clock* sehingga *receiver* harus membangkitkan sinyal *clock* sendiri (tidak memerlukan sinkronisasi). (Nalwan, 2003)

Pengiriman data secara serial dapat dibagi menjadi tiga menurut arah datanya, yaitu *Simplex*, *Half-Duplex* dan *Full-Duplex*. Ketiga mode tersebut diilustrasikan pada Gambar 2.6. (Mazidi, 2000)

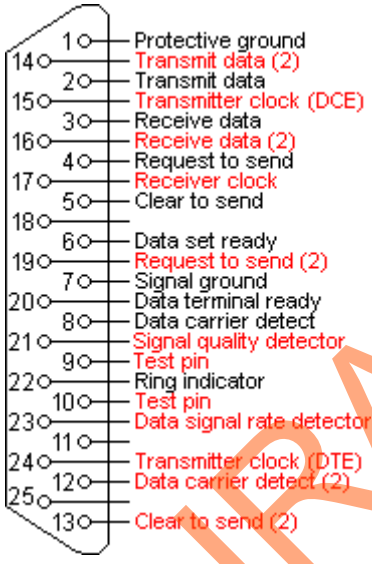


Sumber : Lohala, 2011

Gambar 2.6. Arah komunikasi serial

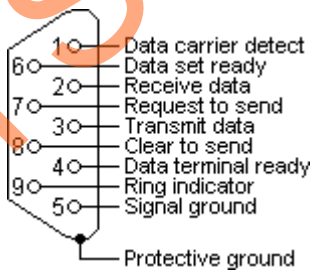
Satuan kecepatan transfer data (*baud rate*) pada komunikasi serial adalah bps (*bits per second*). Untuk menjaga kompatibilitas dari beberapa peralatan komunikasi data yang dibuat oleh beberapa pabrik, pada tahun 1960 EIA (*Electronics Industries Association*) melakukan standarisasi antarmuka serial dengan nama RS232. Keluaran yang dihasilkan oleh RS232 tidak sesuai dengan keluaran TTL (*Transistor-Transistor Logic*) yang sudah ada. Dalam RS232, logika 1 direpresentasikan dengan tegangan -3 V sampai dengan -25 V sedangkan logika 0 direpresentasikan dengan tegangan +3 V sampai dengan +25 V. Hasil tak terdefinisi jika berada diantara tegangan -3 V sampai dengan +3 V. IBM PC atau komputer yang berbasis x86 (8086, 286, 386, 486, dan Pentium) secara umum *processor* yang digunakan memiliki dua *port* COM. Keduanya merupakan

konektor jenis RS232 yaitu DB25 dan DB9. Ilustrasi DB25 dan keterangan *pinout*-nya terdapat pada Gambar 2.7, sedangkan ilustrasi DB9 dan keterangan *pinout*-nya terdapat pada Gambar 2.8.



Sumber : Bies, 2011

Gambar 2.7. *Pinout* konektor DB25



Sumber : Bies, 2011

Gambar 2.8 *Pinout* konektor DB9

Sumber : ATMEL 2011

2.3. Modul komunikasi *Wireless* 802.15.4 Xbee-Pro

Modul komunikasi *wireless* yang penulis gunakan adalah Xbee-Pro. Modul Xbee-Pro direkayasa untuk memenuhi ZigBee / IEEE 802.15.4 dan merupakan untuk standarisasi pengalamatan unik dengan harga yang murah,serta jaringan nirkabel ini hanya membutuhkan daya yang rendah. Modul

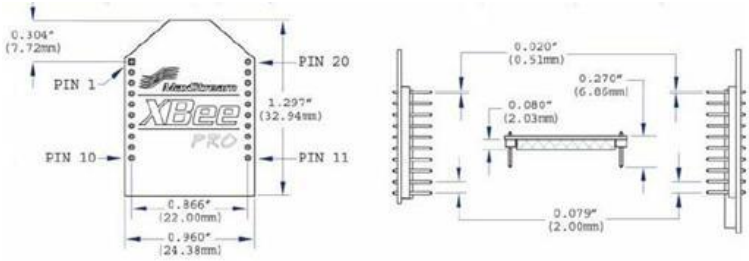
membutuhkan daya minim dan disisi pengiriman dapat mengandalkan data penting antar perangkat. Modul ini beroperasi dalam frekuensi 2,4 GHz ISM.

Modul komunikasi wireless ini mempunyai fitur yaitu :

1. Pengontrolan jarak jauh dalam ruangan bisa mencapai 100 meter dan jika diluar ruangan dapat mencapai 300 meter.
2. Modul ini mampu mentransmisikan daya hingga: 100 mW (20 dBm)
3. Mempunyai sensitivitas penerima data mencapai: -100 dBm
4. Mempunyai kecepatan transfer data: 250,000 bps
5. Paket dapat dikirimkan dan diterima menggunakan data 16-bit atau sebuah alamat 64-bit (protokol 802.15.4).
6. Setiap modul akan menerima paket memiliki alamat *broadcast*. Ketika dikonfigurasi untuk beroperasi di *Broadcast Mode*, modul penerima tidak mengirim ACK (*Acknowledgement*) dan Transmitting.

Xbee-Pro ini mempunyai 20 kaki, diantaranya 4 pin sebagai input adalah port 3, port 5, port 9, dan port 14, serta ada 4 pin sebagai output adalah port 2, port 4, port 6, dan port 13. Dan 4 pin yang digunakan, yaitu VCC dan GND untuk tegangan suplay, DOUT merupakan pin transmit (TX), DIN merupakan pin *receive* (RX). Modul X-Bee Pro dapat dilihat pada gambar 2.9. dan dimensi Xbee-Pro dapat dilihat pada gambar 2.10. dibawah ini.

Sumber : Manual Xbee-Pro, 2011
Gambar 2.9. Modul X-Bee Pro



Sumber : Manual Xbee-Pro, 2011

Gambar 2.10. Dimensi Xbee-Pro

Tabel 2.6. Spesifikasi Xbee-Pro

Performance	
Indoor Urban-Range	up to 300‟ (100 m)
Outdoor RF line-of-sight Range	up to 1 mile (1500 m)
Transmit Power Output	60 mW (18 dBm) conducted,
(software selectable)	100 mW (20 dBm) EIRP
RF Data Rate	250,000 bps
Serial Interface Data Rate	1200 – 115200 bps
(software selectable)	(non-standard baud rates also supported)
Receiver Sensitivity	- 100 dBm (1% packet error rate)
Power Requirements	
Supply Voltage	2.8 – 3.4 V
Idle / Receive Current (typical)	55 mA (@3.3 V)
Power-down Current	< 10 µA
General	
Operating Frequency	ISM 2.4 GHz
Frequency Band	2.4 - 2.4835 GHz
Modulation	OQPSK
Dimensions	0.960" x 1.297" (2.438cm x 3.294cm)
Operating Temperature	-40 to 85° C (industrial)
Antenna Options	Integrated Whip, Chip or U.FL
Connector	
Networking & Security	
Supported Network Topologies	
lanjutan	
	Point-to-point, Point-to multipoint & Peer-to-peer
Number of Channels	12 Direct Sequence Channels
(softw are selectable)	

Sumber : Manual Xbee-Pro, 2011

2.3.1. Command Mode

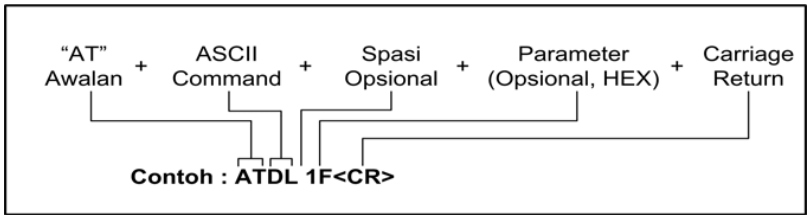
Untuk memodifikasi atau membaca parameter Xbee-Pro, langkah pertama yang harus dilakukan adalah masuk ke command mode, yaitu command yang menafsirkan karakter yang datang. AT Command ini dilakukan dengan memprogram modul.

2.3.2. AT Command

Untuk masuk ke mode AT Command, kirim tiga rangkaian karakter ”+++” kemudian amati guard time sebelum dan sesudah karakter perintah (command) yang menunjuk pada mode rangkaian dari default AT Command. Berikut adalah mode rangkaian default AT Command untuk masa transisi ke mode command :

a. Tidak ada karakter yang dikirim selama satu detik [GT (Command Guard Time) parameter = 0x3E8]
b. Input tiga karakter plus (”+++”) dalam satu detik [CC (Command Sequence Character) parameter = 0x2B]
c. Tidak ada karakter yang dikirim selama satu detik [GT (Command Guard Time) parameter = 0x3E8]

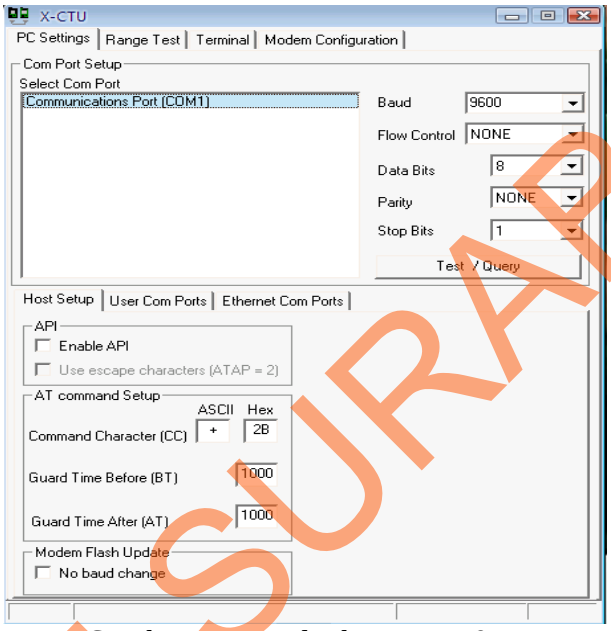
Untuk mengirim AT Command dan parameter, digunakan syntax seperti gambar 2.11 di bawah ini berikut :



Sumber : Manual Xbee-Pro, 2011

Gambar 2.11. Struktur pemrograman pada AT Command

Langkah pertama yang harus dilakukan dalam menggunakan Xbee- PRO agar dapat melakukan komunikasi point to point atau point to multipoint adalah melakukan seting konfigurasi alamat (address). Proses konfigurasi ini dapat dilakukan melaui software X-CTU yang merupakan software aplikasi khusus untuk Xbee-Pro ditunjukkan pada gambar 2.12 dibawah ini.



Sumber : Manual Xbee-Pro, 2011

Gambar 2.12. Tampilan untuk setting konfigurasi alamat pada X-CTU

Cara lain untuk melakukan setting dapat dilakukan melaui hiperterminal. Untuk melakukan setting konfigurasi address melalui hiperterminal ada dua metode. Metode pertama disebut *one line per command* dan metode kedua disebut *multiple command on one line.*

1. **Metode 1** *(One line per command)*

Tabel 2.7 Konfigurasi Pemrograman X-CTU Metode 1

+++ OK<CR> (Enter into Command mode)
ATDL<Enter> {Current Value} <CR> (Read Destination Address Low)
ATDL1A0D <Enter> OK <CR> (Modify Destination

Address Low)
ATWR <Enter> OK <CR> (Write to non volatile memory)
ATCN
<Enter> OK <CR> (Exit Command Mode)

Sumber : Manual Xbee-Pro, 2011

2. **Metode 2** *(Multiple commands on one line)*

Tabel 2.8. Konfigurasi Pemrograman X-CTU Metode 2

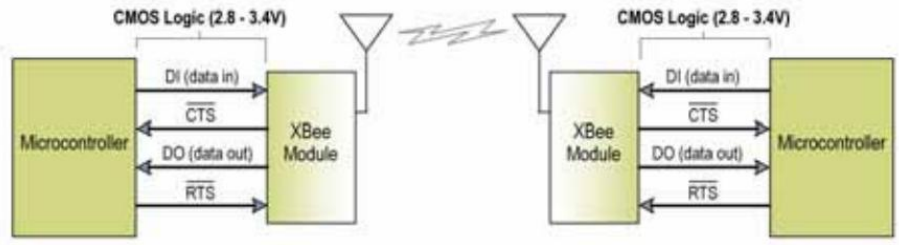
+++ OK<CR> (Enter into Command mode)
ATDL<Enter> {Current Value} <CR> (Read Destination Address Low)
ATDL1A0D,WR<CN<>Cr>OK,OK,OK<CR>

Sumber : Manual Xbee-Pro, 2011

Setelah melakukan setting konfigurasi ini maka modul Xbee- PRO siap digunakan untuk melakukan komunikasi point to point, dengan baud rate 9600 bps.

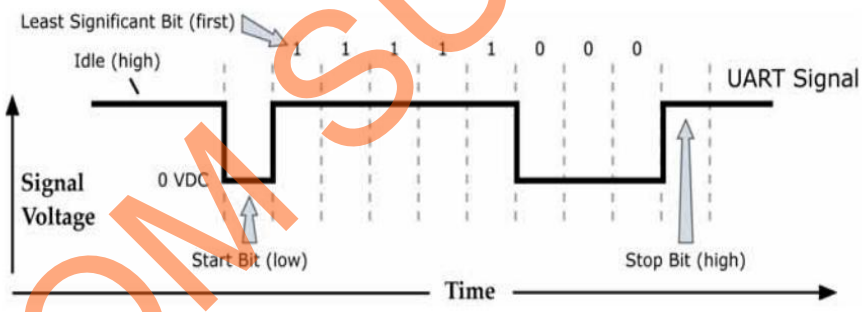
2.3.3. Pengoperasian Xbee-Pro

Modul Xbee-Pro dihubungkan dengan host melalui level logika port serial asinkron. Melalui port serial ini, modul Xbee-Pro dapat berkomunikasi dengan logika dan tegangan yang kompatibel dengan UART atau melalui level translator ke sembarang device serial, seperti RS-232/485/422 atau USB. Device yang mempunyai UART interface dapat langsung dihubungkan secara langsung dengan pin-pin modul XBee seperti di tunjukkan gambar 2.13.



Gambar 2.13 Diagram sistem aliran data pada XBee

Data diterima oleh modul Xbee-Pro melalui pin DI (pin 3) sebagai sinyal serial asinkron. Sinyal harus berada pada kondisi idle high ketika tidak ada data yang harus ditransmisikan. Setiap byte data terdiri dari satu bit start (low), 8 bit data (dengan LSB terlebih dahulu), dan satu bit stop (high). Gambar 3.20 mengilustrasikan pola bit data serial dari Xbee-Pro. Paket data 0x1F (bilangan desimal "31") yang ditransmisikan melalui Xbee-Pro. Contoh format data adalah 8-N-1 (bits – parity – jumlah bit stop).



Gambar 2.14. Contoh format pengiriman data

Gambar 2.14 menunjukkan diagram data flow internal dengan lima buah pin yang paling sering digunakan. Ada 2 mode operasi dari XBee, yaitu mode Transparent (AT) dan mode API (Packet). Mode transparent (AT) digunakan jika diinginkan konfigurasi point-to-point yang sederhana, dimana XBee bertindak sebagai modem serial *wireless* antara komputer atau mikrokontroler dengan remote device. Mode *transparent* (AT) menggunakan komunikasi serial yang sederhana. Fitur dari mode *transparent* adalah sebagai berikut:

- Sederhana
- Kompatibel dengan semua peralatan yang menggunakan komunikasi serial
- Terbatas hanya untuk komunikasi point-to-point antara 2 XBee

Mode operasi API (packet) mempunyai kemampuan yang lebih baik namun lebih kompleks dari dari mode transparent. Dengan mode API, memungkinkan untuk membuat jaringan yang terdiri dari beberapa XBee dan antar XBee yang satu dengan yang lainnya dapat saling berkomunikasi secara individual. Fitur dari mode API adalah sebagai berikut:

- I/O line passing, yaitu menerima data dari remote XBee yang berdiri sendiri (stand-alone remote Xbee)
- Memungkinkan untuk komunikasi *broadcast* dan komunikasi dengan lebih dari satu XBee
- Menerima acknowledgement bahwa paket telah dikirim dengan baik
- Memungkinkan konfigurasi jarak jauh

2.3.4. Pengalamatan (*Addressing*) Xbee-Pro

Pengalamatan digunakan untuk membedakan satu Xbee-Pro dengan XBee lainnya dan mencegah duplikasi paket data. Setiap modul Xbee-Pro mempunyai source address (alamat asal) untuk mencegah agar pesan non-duplikat tidak dianggap sebagai pesan duplikat.

Xbee-Pro mempunyai dua bentuk dasar pengalamatan, yaitu *Broadcast* dan *Unicast*. Pesan *Broadcast* adalah sebuah pesan yang akan diterima oleh semua modul yang mempunyai PAN ID (Personal Area Network) yang sama. Pesan *Broadcast* dikirim hanya sekali dan tidak

diulang, sehingga tidak ada jaminan bahwa node-node yang dikirimi akan menerima pesan tersebut. Agar Xbee-Pro bisa mengirim pesan *Broadcast*, set DH=0x0 dan DL=0xFFFF.

Dengan setingan tersebut, semua modul Xbee-Pro yang berada dalam jangkauan jaringan akan menerima pesan. Pesan *Unicast* merupakan metode yang lebih handal dalam pengiriman data. Pesan *Unicast* dikirim dari satu modul ke modul yang lain berdasarkan pengalamatan modul-modul tersebut. Jika pesan diterima dengan baik, XBee pene-Pro rima akan mengirim balik sebuah acknowlegdement atau ACK. Jika Xbee-Pro pengirim tidak menerima ACK, Xbee-Pro pengirim akan mengirim ulang data tersebut (maksimal 3 kali) sampai ACK diterima. Hal ini akan meningkatkan kemungkinan pengiriman data sampai ke tujuan.

Ada 2 metode pengiriman data pada pesan *Unicast*, yaitu menggunakan pengalamatan 16 bit dan pengalamatan 64 bit. Satu atau kedua metode tersebut dapat digunakan untuk mengkomunikasikan Xbee-Pro, akan tetapi, pengalamatan 16 bit bisa di-*disable* sedangkan pengalamatan 64 bit tidak bisa di-*disable*.

Pengalamatan 16 bit lebih cocok digunakan untuk jaringan yang kecil atau jaringan yang mempunyai jumlah node yang tetap. Pengalamatan 16 bit menggunakan 16 bit bilangan heksa untuk menentukan source address atau *destination address* dari setiap modul Xbee-Pro. Ketika membangun suatu jaringan, setiap modul Xbee-Pro harus mempunyai *source address* yang unik. Parameter MY (gambar 3.22) secara default di-set 0, maka ketika menggunakan pengalamatan 16 bit, nilai tersebut harus diubah menjadi nomor

yang unik. Karena menggunakan pengalamatan 16 bit, maka jaringan akan mempunyai 2 16 atau 65536 alamat unik. Pada pengalamatan 16 bit, destination address (DL) Xbee-Pro pengirim harus sesuai dengan source address (MY) dari Xbee-Pro penerima, sedangkan parameter DH harus di-set 0.

Sumber : Manual Xbee-Pro 2011

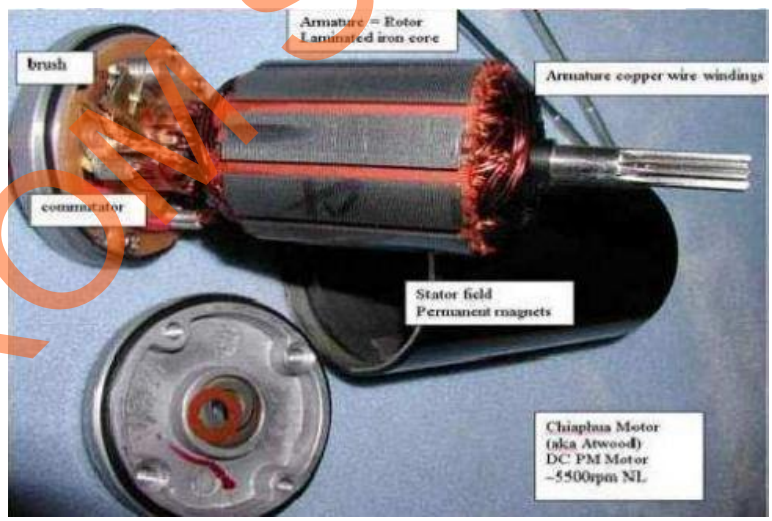
2.4. Motor DC

Motor DC merupakan alat penggerak dari robot. Secara garis besar motor dapat dibedakan dalam tiga kategori yaitu :

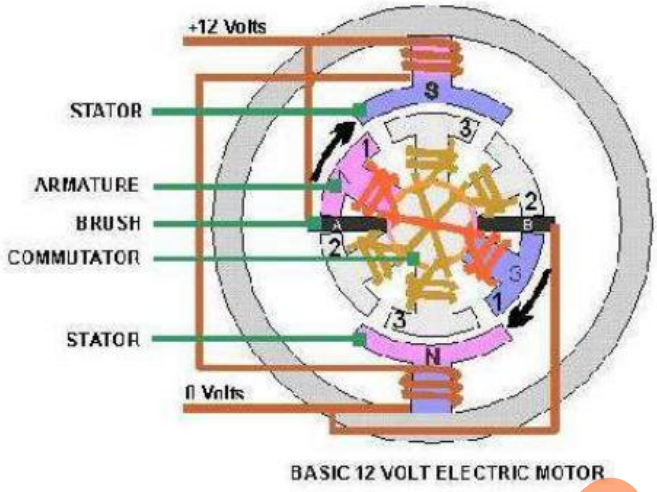
- Motor AC adalah motor yang digerakkan dengan jaringan satu fasa atau tiga fasa dengan frekuensi 60 atau 50 Hz.
- Motor DC Konvensional adalah motor yang mempunyai dua terminal yang dihubungkan dengan dua kutub *battery*. Biasanya motor DC dioperasikan dengan *supply* DC yang dikonversikan dari jaringan AC. Secara struktural motor DC adalah motor yang mempunyai *copper commutator* dan karbon atau *metal brushes.*
- *Electrically controlled precision motor*, yang termasuk di dalammnya adalah *brushles dc motor* dan *stepping motor.*

Motor DC merupakan motor yang paling banyak digunakan dalam kehidupan. Motor DC adalah motor yang penggeraknya berupa sumber tegangan searah. Kebanyakan motor yang digunakan dimainan, mobil dan *radio-controlled* adalah motor DC. Hal ini menyebabkan produksi motor DC lebih besar daripada motor-motor lainnya.

Sebuah motor DC memiliki kumparan-kumparan kawat yang dipancangkan didalam slot-slot sebuah silinder yang terbuat dari bahan feromagnetik. Silinder ini diberi nama *armature* dipasang pada suatu bentuk dudukan (*bearing*) dan bebas putar. Dudukan *armature* adalah sebuah medan magnet yang dihasilkan oleh magnet-magnet permanen atau yang dialirkan melalui kumparan-kumparan kawat yang dinamakan kumparan medan. Kedua magnet ini, magnet permanen maupun *electromagnet,* disebut sebagai *stator* (bagian yang diam). Ketika arus mengalir melalui kumparan *armature,* sebuah konduktor berarus yang berada tegak lurus terhadap sebuah medan magnet akan mengalami gaya. Gaya-gaya akan bekerja pada kumparan tersebut dan mengakibatkan putaran. Bagian-bagian dari motor DC dapat dilihat pada gambar 2.15, kemudian untuk melihat isi detail dari motor DC dapat dilihat pada gambar 2.16. dibawah ini.



Gambar 2.15. Bagian-bagian motor DC



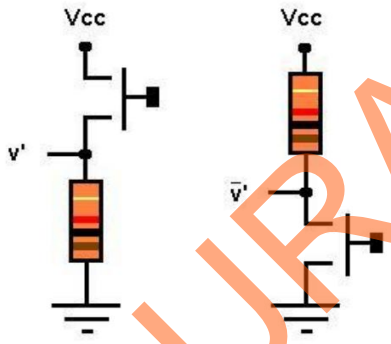
Gambar 2.16. Detail Motor DC

Kecepatan putaran dapat diubah dengan cara mengubah besar arus pada kumparan *armature*. Akan tetapi, karena sumber tegangan tetap biasanya digunakan sebagai input ke kumparan perubahan arus yang diperlukan seringkali diperoleh melalui penggunaan sebuah rangkaian elektronik. Rangkaian ini dapat mengontrol nilai rata-rata tegangan, dengan cara mengubah-ubah interval waktu untuk menghasilkan tegangan DC yang bervariasi, yang dalam pembuatan tugas akhir ini menggunakan rangkaian PWM (*pulse with modulation*) yang sudah terdapat pada mikrokontroller ATMega 32 dari Atmel.

2.5. Joystick

Joystick merupakan alat yang digunakan untuk mengendalikan, mengontrol serta menjalankan robot. Dalam joystick ini terdapat beberapa tombol yang berguna untuk mengirimkan inputan-inputan untuk menjalankan salah satu motor pada robot. Masing-masing tombol ini mempunyai fungsi yang berbeda-beda. Dalam joystick ini terdapat enam buah *push button* dan dua buah potensio terdiri dari 2 buah push button yang berfungsi untuk menggerakan tangan robot

naik dan turun, dua buah push button yang berfungsi untuk menggerakkan tangan robot maju dan mundur, 2 buah push button yang berfungsi untuk menggerakkan tangan robot untuk menjepit dan melepaskan benda, serta 2 buah potensio yang berfungsi untuk mengontrol motor kanan dan motor kiri yang berguna untuk menjalankan robot. Sebagai tombol pada *joystick* guna mengontrol pergerakan robot, penulis menggunakan *push button* pada rangkaian robot ini. Berikut adalah cara kerja *push button* sebagai *switch* pada *joystick* ditunjukkan pada gambar 2.17



Gambar 2.17 Cara kerja *push button*

Pada gambar 2.17 diatas, VCC adalah sebagai sumber tegangan (baterai) dihubungkan dengan suatu tahanan(load) dan *push button.* Arus akan melewati tahanan jika *push button* ditekan sehingga rangkaian tersebut menjadi *close up*(arus mencapai *ground*). Pada *joystick* robot manual, tahanan (*load)* adalah H-Bridge yang nantinya akan dialiri arus sehingga mampu menggerakkan motor DC.