BAB II
LANDASAN TEORI

2.1 *Programmable Logic Control* (PLC)

Programmable Logic Control merupakan suatu bentuk khusus pengontrol berbasis *microprocessor* yang memanfaatkan memori yang dapat diprogram untuk menyimpan instruksi – instruksi dan untuk mengimplementasikan fungsi – fungsi semisal logika, *sequencing*, pewaktuan (*timing*), pencacahan (*counting*) dan aritmatika guna mengontrol mesin-mesin dan proses-proses serta dirancang untuk dioperasikan oleh para insinyur yang hanya memiliki sedikit pengetahuan mengenai komputer dan bahasa pemrograman (Bolton, 2004 : 3). Sebagian besar industri telah menerapkan sistem otomatis dalam proses produksi. Pada umumnya sistem otomatis yang diterapkan terdiri atas dua metode yaitu otomatisasi berbasis kontrol relay dan otomatisasi berbasis *Programmable Logic Control* (PLC). Otomatisasi berbasis relay banyak digunakan pada mesinmesin yang memiliki urutan-urutan (sekuens) yang sederhana, sedangkan otomatisasi PLC dapat memiliki sekuens yang lebih kompleks dari *relay*. Otomatisasi berbasis PLC dapat diintegrasikan dengan sistem monitoring. Sistem monitoring berbasis PLC adalah suatu sistem yang berguna untuk mengontrol proses suatu kerja tertentu., dimana parameter atau inputan data diambil dan diolah oleh *Personal Computer* (PC) dan melalui sebuah program tertentu

(Bolton, 2006 : 3)

Sesuai namanya, PLC dapat dengan mudah diprogram ulang. Keunggulan PLC dibandingkan dengan sistem konvensional antara lain :

a. Relatif mudah untuk melakukan perubahan pada strategi kendali yang akan

   diterapkan, karena logika kendali yang digunakan diwujudkan dalam bentuk

   perangkat lunak.

b. Jumlah *relay* yang diperlukan dapat dikurangi sesuai dengan jumlah *input* maupun *output* yang diperlukan. Lebih mudah untuk proses instalasinya karena pengkabelan lebih sederhana.

c. Lebih mudah dalam menemukan kesalahan dan kerusakan, karena memiliki

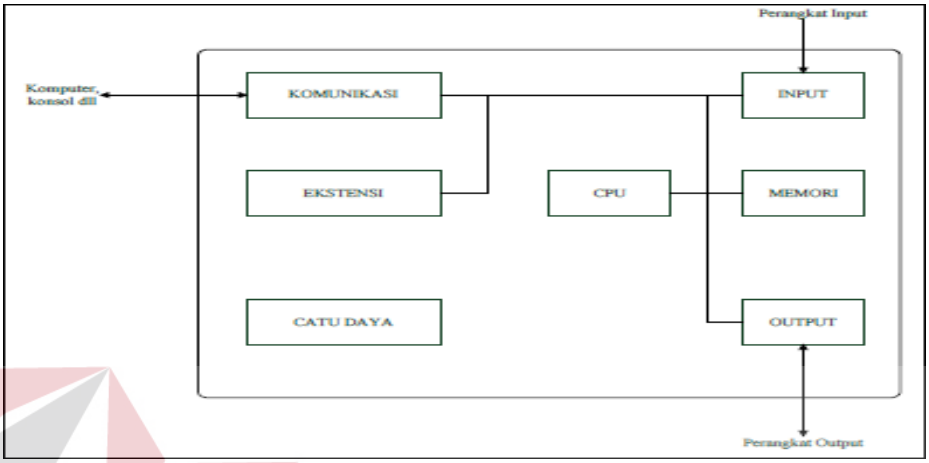
   fasilitas *self – diagnosis.*

d. Tahan terhadap temperature tinggi, tekanan tinggi dan kelembaban yang tinggi apabila dipakai secara terus - menerus, dan ini banyak di jumpai pada

   lingkungan industri.

2.1.1 Komponen Dasar PLC

PLC tersusun atas beberapa komponen dasar yang dapat dilihat pada Gambar 2.1 berupa diagram blok PLC.



Gambar 2.1 Diagram Blok PLC.

Dimana pada Gambar 2.1 dijelaskan beberapa komponen-komponen PLC yaitu :

a. **CPU (*Central Processing unit*)**, yaitu otak dari PLC yang mengerjakan berbagai operasi, antara lain mengeksekusi program, menyimpan dan mengambil data dari memori, membaca kondisi/nilai *input* serta mengatur nilai *output*, memeriksa adanya kerusakan (*self - diagnosis*), serta melakukan komunikasi dengan perangkat lain.
b. ***Input***, merupakan bagian PLC yang berhubungan dengan perangkat luar yang memberikan masukan kepada CPU. Perangkat luar *input* dapat berupa tombol, *switch*, sensor atau piranti lain.
c. ***Output***, merupakan bagian PLC yang berhubungan dengan perangkat luar yang memberikan keluaran dari CPU. Perangkat luar *output* dapat berupa lampu, katub (*valve*), motor dan perangkat – perangkat lain.

d. **Memori**, yaitu tempat untuk menyimpan program dan data yang akan dijalankan dan diolah oleh CPU. Dalam pembahasan PLC, memori sering disebut sebagai *file*. Dalam PLC memori terdiri atas memori program untuk menyimpan program yang akan dieksekusi, memori data untuk menyimpan nilai-nilai hasil operasi CPU, nilai *timer* dan *counter*, serta memori yang menyimpan nilai kondisi *input* dan *output*. Kebanyakan PLC sekarang memiliki satuan memori dalam *word* (16 bit).

e. **Fasilitas komunikasi**, yang membantu CPU dalam melakukan pertukaran data dengan perangkat lain, termasuk juga berkomunikasi dengan komputer untuk melakukan pemrograman dan pemantauan.

f. **Fasilitas ekstensi**, untuk menghubungkan modul PLC dengan modul pengembangan *input/output* sehingga jumlah terminal I/O dapat ditingkatkan.

g. **Catu daya**, untuk memberikan sumber tegangan kepada semua komponen dalam PLC. Biasanya sumber tegangan PLC adalah 220 V AC atau 24 V DC.

Pada dasarnya sinyal yang diterima/dibangkitkan oleh unit *input/output* PLC berupa sinyal digital, yang bernilai biner 0 atau 1. Perangkat *input/output* yang memiliki sinyal analog memerlukan piranti ADC (*Analog to Digital Converter*) atau DAC (*Digital to Analog Converter*) agar dapat dihubungkan ke PLC. Biasanya piranti ini terdapat dalam modul analog yang diproduksi pabrik pembuat PLC. Sinyal analog yang biasanya digunakan dalam PLC mengikuti standar industri, yaitu arus 4 – 20 mA untuk tegangan *input* digital bermacam-macam

mulai dari 5 V DC, 12 V DC atau 24 V DC, sedangkan terminal *output* dapat berupa *relay* atau transistor (Achmad, 2007 : 5-6).

2.1.2 Bahasa Pemrograman

Terdapat banyak pilihan bahasa untuk membuat program dalam PLC. Masing-masing bahasa mempunyai keuntungan dan kerugian sendiri-sendiri tergantung dari sudut pandang kita sebagai *user*.

A. *Ladder Diagram* (LDR)

1. Tipe Program

Ladder diagram mengGambarkan program dalam bentuk grafik. Diagram ini dikembangkan dari kontak-kontak *relay* yang terstruktur yang mengGambarkan aliran arus listrik. Dalam *ladder diagram* ini terdapat dua buah garis vertikal. Garis vertikal sebelah kiri dihubungkan dengan sumber tegangan positif/rel catu daya aktif sedangkan garis sebelah kanan dengan sumber tegangan negatif/rel catu daya pasif.

2. Elemen Program

Diantara dua garis ini dipasang kontak-kontak yang mengGambarkan kontrol dari *switch*, sensor atau *output*. Satu baris dari diagram disebut dengan satu rung. *Input* menggunakan symbol “| |” (kontak, normal *open*) dan “|/|” (negasi kontak, normal *closed*). *Output* mempunyai simbol “( )” yang terletak paling kanan menempel garis vertikal kanan.

Selama pemrograman setiap simbol yang diberikan adalah alamat PLC sesungguhnya atau merupakan alamat simbolic (misalnya S1, S2, S3, H). Dapat dilihat pada Gambar 2.2.

Gambar 2.2 *Ladder Diagram*.

B. *Statement List* (STL)

Struktur *statement list*

PROGRAM

STEP

KALIMAT

BAGIAN KONDISI

BAGIAN PELAKSANAAN

A. Kalimat

Kalimat merupakan pembentukan dasar dari organisasi program. Masingmasing kalimat terdiri dari bagian kondisi dibagian pelaksanaan. Bagian kondisi mengandung satu atau beberapa buah kondisi yang akan diuji (benar atau salah) pada saat program berjalan.

Bagian kondisi selalu dimulai dengan kata *IF* (jika). Jika kondisi berjalan benar maka instruksi yang ditulis pada bagian pelaksanaan akan dijalankan. Awal bagian pelaksanaan dimulai dengan kata *THEN* (maka).

Contoh :

```
IF              I5    (Jika input I5 memberikan sinyal
THEN      SET   O2    maka nyalakan output O2)
IF              I5    (Jika input I5 memberikan sinyal
          AND   I6    dan input I6 memberikan sinyal
THEN RESET      O4    jika ya, matikan output O4,
          SET   O6    nyalakan output O6)
```

B. STEP

Program yang tidak menggunakan instruksi STEP dapat diproses dengan cara *scanning*. Tetapi STL menyediakan instruksi STEP yang membagikan program menjadi bagian-bagian yang lebih kecil.

Aturan pelaksanaan STEP :

1. Jika kondisi dari sebuah kalimat terpenuhi maka bagian pelaksana akan dijalankan.
2. Jika kondisi dari kalimat terakhir dalam suatu STEP terpenuhi maka bagian pelaksana akan dijalankan dan program berlanjut ke STEP berikutnya.
3. Jika kondisi dari sebuah kalimat dalam suatu STEP tidak terpenuhi maka program akan berpindah ke kalimat berikutnya dalam STEP tersebut.
4. Jika kondisi dari kalimat terakhir dalam suatu STEP tidak terpenuhi maka program akan kembali ke kalimat pertama dari STEP yang sekarang.

C. **Instruksi NOP (*No Operation*)**

Instruksi NOP dapat diletakkan pada bagian kondisi atau bagian pelaksanaan dari sebuah kalimat. Bila digunakan dalam kondisi, instruksi *NOP* selalu bernilai benar. Dengan kata lain NOP menyebabkan pelaksanaan tanpa suatu kondisi.

Jika digunakan dalam bagian pelaksanaan pengertian NOP adalah "**Tidak melakukan sesuatu"**. Hal ini sering digunakan pada saat program harus menunggu untuk kondisi tertentu lalu pindah ke *STEP* berikutnya.

```
IF        I5    Jika input I5 aktif
THEN NOP        Jangan lakukan apa-apa, pergi
                ke STEP selanjutnya.
```

D. **Instruksi *JUMP***

Instruksi *JUMP* digunakan mempengaruhi jalannya suatu program. Pengertian yang mudah dimengerti dari instruksi *JUMP* adalah "LOMPAT".

```
STEP 1
IF              I5    Jika input I5 aktif
THEN      SET   O2    Aktifkan output O2
          JMP TO 4    Lompat ke STEP 4
IF              I6    Jika input I6 aktif
          AND   I8    dan input I8 aktif
THEN RESET      O4    Matikan output O4
          JMP TO 8    Lompat ke STEP 8
```

E. **Instruksi *LOAD…TO***

Instruksi *LOAD...TO* dapat digunakan untuk melakukan perhitungan-perhitungan aritmatika atau logika yang rumit mengcopy (dengan cara memanggilnya dalam *Multi bit Accumulator*). Instruksi *LOAD...TO* paling banyak digunakan dalam *multioperand*.

F. **Instruksi OTHRW**

Instruksi OTHRW dapat digunakan untuk mempengaruhi aliran program. Instruksi OTHRW ini dijalankan pada saat *IF* **terakhir yang dapat dijumpai bernilai salah**.

```
IF              I5
THEN SET        O2
OTHRW SET       O4
```

2.1.3 *Timer* dan *Counter*

A. *Timer*

Instruksi *timer* ini digunakan sebagai pewaktu sehingga dapat ditentukan nilainya. Pada tiap-tiap *timer* terdiri dari beberapa bagian, yaitu :

1. ***Timer* Status *Bit***

Untuk penulisannya adalah "Tn". *Timer* status bit ini berfungsi untuk mengetahui status *timer*, yakni aktif atau tidak aktif. Nilai bit akan berubah

menjadi “1” (aktif) pada saat diberi instruksi SET. Pada saat periode waktu yang diprogram telah selesai atau *timer* dihentikan dengan cara RESET, maka status bit akan berubah menjadi “0” (tidak aktif).

2. Timer Preset

Untuk penulisannya adalah “TPn”. *Timer preset* merupakan *operand multibit* yang berisi nilai interval *timer*. *Timer preset* diberi nilai saat melakukan inisialisasi atau pengaktifan terhadap *timer*.

3. Timer Word

Untuk penulisannya adalah “TWn”. *Timer word* berfungsi sebagai pencacah yang nilainya berkurang secara otomatis pada interval tertentu dan teratur.

A. *Counter*

Instruksi *counter* ini berfungsi untuk menghitung sampai suatu batas tertentu dan mempunyai kondisi ON dan OFF.

B. *Counter* Status *Bit*

Untuk penulisannya adalah “Cn”. *Counter* status bit ini berfungsi untuk mengetahui status *Counter*, yakni aktif atau tidak aktif. Nilai bit akan berubah menjadi “1” (aktif) pada saat diberi instruksi SET. Pada saat jumlah yang diprogram telah selesai atau *Counter* dihentikan dengan cara RESET, maka status bit akan berubah menjadi “0” (tidak aktif).

C. Counter Preset

Untuk penulisannya adalah “CPn”. *Counter preset* merupakan *operand multibit* yang berisi batas perhitungan *counter*. *Counter preset* diberi nilai saat melakukan inisialisasi atau pengaktifan terhadap *counter*.

D. *Counter Word*

Untuk penulisannya adalah "CWn". *Counter word* berisi nilai actual perhitungan yang diakibatkan oleh perintah INC maupun DEC.

E. Perbedaan dan Persamaan Antara *Counter* /*Timer*.

Counter dan *timer* memiliki perbedaan dan persamaan, seperti pada keterangan Tabel 2.1.

Tabel 2.1. Perbedaan dan Persamaan *Counter/Timer*.

Uraian	Perbandingan	
	Counter	***Timer***
Akan OFF jika terjadi, sebagai berikut :	• Belum digunakan sama sekali. • Di RESET. • CW berubah dari 1 menjadi 0 akibat perintah DEC. • CW berubah dari CP-1 menjadi CP akibat perintah INC.	• Belum digunakan sama sekali. • Di RESET. • TW berubah dari 1 menjadi 0 akibat perintah AUTO-DEC (perintah untuk mengurangi satu dan terjadi setiap 10ms setelah *timer* berstatus ON).
Mempunyai karakteristik khusus saat di SET	• *Counter* status berubah menjadi ON. • Nilai CW berubah menjadi nol.	• *Timer* status berubah menjadi ON. • Nilai TW (*Timer Word*) dibuat sama dengan TP (*Timer Preset*). • Selanjutnya terjadi AUTO-DEC setiap 10ms terhadap TW sampai TW sama dengan nol dan keadaan OFF terjadi.

2.1.4 *Flag*

Flag merupakan memori 1 bit internal. Dengan demikian *flag* hanya bernilai "1" dan "0". *Flag* berfungsi sebagai :

1. Mengganti *input* atau *output*.

2. Membuat sistem *latch*. *Latch* adalah suatu teknik / mekanisme untuk menyimpan suatu informasi / data, baik *input* maupun *output*.
3. Mengendalikan proses *sequensial*.

2.1.5 *Multitasking*

Multitasking adalah kemampuan PLC untuk menjalankan beberapa program secara serentak. Istilah *multitasking* ini biasanya digunakan pada sistem operasi yang mengorganisasikan program menjadi beberapa bagian-bagian program lagi dengan menggunakan teknik modular. Dalam PLC ini dimungkinkan terjadi proses *multitasking* karena sebuah program PLC dapat terdiri dari beberapa *module* dimana tiap modul tersebut juga merupakan suatu program yang utuh.

Keuntungan dari *multitasking* adalah program menjadi lebih pendek sehingga program menjadi lebih jelas.

Dalam FST, ada beberapa hal yang perlu diketahui yang berkaitan dengan *multitasking*, yaitu *project*, program dan *module*.

Project adalah suatu ruang atau direktori yang berisi beberapa program dan *module*. *Project* merupakan kumpulan dari program dan *module* baik yang saling berhubungan maupun yang berdiri sendiri.

Program adalah kumpulan-kumpulan instruksi yang membentuk suatu tugas tertentu. Program merupakan bagian utama dari suatu *project*. Dalam sebuah *project* terdapat 64 (0…63) program dan 9 (1…9) versi. Program dapat mengaktifkan suatu program lain dalam versi yang sama dan suatu *module*. Pada *multitasking* program berfungsi sebagai koordinator yang mengatur programprogram lain dan *module-module* yang digunakan.

Module adalah suatu kumpulan instruksi yang membentuk suatu tugas tertentu. *Module* sama halnya dengan program, hanya saja *module* tidak dapat langsung aktif begitu di transfer ke CCU. *Module* diaktifkan oleh program terlebih dahulu agar dapat bekerja. *Module* tidak dapat mengaktifkan program, tetapi *module* dapat mengaktifkan *module* lain, bahkan *module* dapat mengaktifkan dirinya sendiri. Dalam FST terdapat 2 *module call* yaitu *Calling Module Programs* (CMP) dan *Calling Function Module* (CFM). Dalam sebuah *project* terdapat 100 (0…99) *module* dan 9 (1…9) versi baik CMP maupun CFM. Adapun perbedaan antara CMP dan CFM, perbedaan itu terlihat seperti pada Tabel 2.2.

Property	CMP	CFM
Step structure	√	–
Task change at selection	√	–
Call of another CFM	√	√
Call of another CMP	–	–

Tabel 2.2. Perbedaan Antara CMP dan CFM.

FST mempunyai instruksi yang sama untuk memanggil / mengaktifkan program dan *module*, baik melalui program maupun *module* itu sendiri.

1) Memanggil program

```
STEP 1
    IF          NOP         Jika tidak ada sinyal inputan
    THEN SET    P1          Aktifkan program number 1
```

2) Memanggil *module* (CMP)

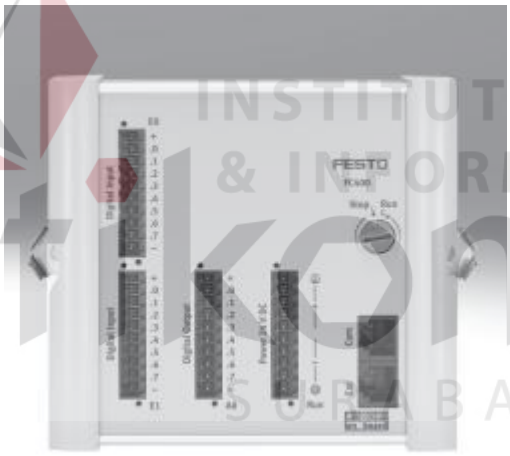
```
STEP 2
    IF          START       Jika input START memberikan
sinyal
    THEN CMP 2              Panggil/aktifkan CMP number 2
    JMP TO 1                Lompat ke STEP 1
```

3) Memanggil *module* (CFM)

```
STEP 3
    IF          NOP         Jika tidak ada sinyal inputan
    THEN CFM 1              Panggil/aktifkan CFM number 1
```

2.1.6 PLC FESTO FC-440

FC-440 adalah termasuk FEC standar yang mana memiliki berbagai macam tipe, misalnya FC-440, FC-600, FC-640, FC-660. FEC standar bukan *controller* mini yang baru, hal ini ditunjukkan dengan masih adanya ruang untuk inovasi (Festo, 2010 : 2). Pada *controller* mini seperti Gambar 2.3 sistem konektor FEC dapat diakses dari depan, dibuat agar tidak memakan ruang yang banyak dan ruang kontrolnya dikotak-kotakkan sesuai dengan I/O dan *supply*.



Gambar 2.3 PLC Festo FC-440.

Jumlah I/O pada FC-440 terdapat 16 digital input dan 8 digital output (Festo,2010 : 6). Pada FC-440 ini mempunyai *FST programming* dimana terdapat dua pemrograman yang dapat digunakan yaitu *Ladder Diagram* (LDR) dan *Statement List* (STL).

A. *Hardware FEC Standard*

FEC standar telah memiliki sebuah clip untuk menekan rel tersebut keatas dan lubang dipojokkan untuk tempat menaruhnya baut. Semua pengaksesan dilakukan dari depan dan tidak perlu adanya ruang pengaksesan dari belakang, bawah, atau samping (Festo, 2010 : 3) seperti pada Gambar 2.4.

Gambar 2.4 *Hardware* FEC Standar.

B. *Power Supply*

FEC standar memiliki tegangan eksklusif sebesar 24 V DC per kontrol yang dikotak-kotakkan. 24 V DC (+25%/-15%) supply untuk *controller* sendiri. 24 V DC (+/–25%) *power supply* untuk sinyal inputan, *positive switching*, 24 V DC sinyal keluaran 400 mA, tahan terhadap arus pendek and *lowresistance loads*. Analog *inputs/outputs* adalah 0(4) ... 20 mA I/Os, 12 *bit resolution* (Festo, 2010 :3).

C. *Serial Interfaces*

Setiap FEC standar dilengkapi dengan 2 interface serial yaitu COM dan EXT. Dalam interface serial terdapat *TTL(time to live*) dengan data transmisi maksimum rata-rata 115 kbits/s. Interface FEC dapat digunakan sebagai RS232c (SM14 or SM15) atau RS485 (SM35) sesuai dengan kebutuhan. COM interface umumnya digunakan bersamaan dengan SM14 untuk pemrograman, lain halnya

interface EXT dimana dapat digunakan untuk sebuah alat MMI, modem atau alatalat lainnya dengan sebuah interface serial (Festo, 2010 : 3).

D. *Ethernet Interfaces*

Versi FEC standar yang memiliki sebuah *Ethernet Interfaces* menggabungkan antara *Ethernet* 10*baseT interface* dengan sebuah koneksi RJ45 dan sebuah transmisi data rata-rata sebesar 10 Mbits/s. Sebuah led akan mengindikasikan aktif tidaknya status koneksi. FEC standar dapat mensupport komunikasi data dan *troubleshooting* dengan melalui *Ethernet interfaces* (Festo, 2010 : 3). Untuk Gambar FEC standar dengan *ethernet* seperti pada Gambar 2.5.

Gambar 2.5 *Hardware* FEC Standar Dengan *Ethernet.*

E. *Programming*

FEC standar telah diprogram dengan menggunakan *FST*. *FST* adalah bahasa pemrograman yang unik dan kaya seperti tradisinya dan sangat mudah untuk digunakan. Perintah seperti IF...THEN...ELSE tersedia juga di FST dan didukung dengan operasi STEP untuk program sekuensial. Pemrograman FST juga dapat diakses melalui *ethernet* (*sebuah* webserver juga tersedia) (Festo, 2010 : 3). Tampilan program FST pada computer seperti pada Gambar 2.6.

Gambar 2.6 Tampilan FST.

2.2 Winshock

Windows Socket yang biasa disebut *Winsock* berfungsi sebagai antarmuka pemrograman jaringan untuk Microsoft Windows yang berdasarkan pada "socket" yang populer pada BSD Unix. Winsock mencakup model Berkeley dan Windows. Aplikasi Winsock 1 bisa meminta Winsock untuk mengirim notifikasi pada jendela pesan. Ini memungkinkan program untuk menangani jaringan, masalah UI, proses background secara bersamaan. Winsock 2 menambahkan banyak fitur.Untuk memudahkan pemrograman dengan kontrol ActiveX, Microsoft telah membuat kontrol Winsock yang diimplementasikan pada file MSWINSCK.OCX. Untuk mempermudah pengertian dan pembahasan, contoh-contoh program pada modul ini akan menggunakan kontrol Winsock yang diimplementasikan pada bahasa pemrograman Microsoft Visual Basic versi 6.0 merupakan inti dari pemrograman jaringan. *Winsock* juga merupakan standar API jaringan pada semua varian dari sistem operasi Microsoft Windows. Ada beberapa fungsi Winsock yang hanya bekerja dengan TCP/IP, tetapi ada versi generik yang lebih baru dari semua fungsi pada Winsock 2 yang memungkinkan Anda menggunakan transpor lain.

Pemograman *winsock* biasanya digunakan untuk kegiatan jaringan. Untuk memperbaiki kenapa *system* jaringan tidak dapat terkoneksi meskipun konfigurasi jaringan *hardware*nya benar, mengirimkan file via tcp dan mengirimkan perintah antara satu *system* computer dengan *system* komputer yang lain. Tidak hanya sebatas jaringan LAN namun juga Internet.

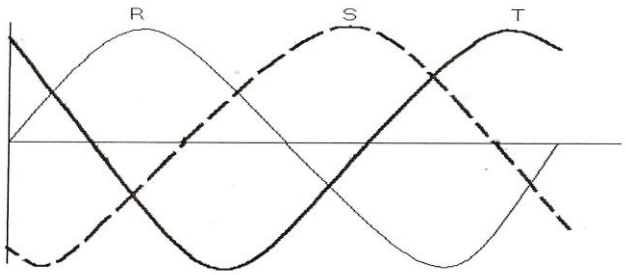
Pada Windows, *winsock* benbentuk file Winsock.dll dan pada file bawaan Visual Basic 6.0 biasanya berbentuk Winsock.ocx. Walaupun dalam penggunaan winsock tujuannya adalah untuk jaringan, namun kita bisa menggukannya dengan satu PC atau dengan *virtual manager*. Jika dengan menggunakan *virtual manager*, kita menginstal system operasi *windows* dengan menggunakan *harddisk virtual*, dan *setting Ip Address* dilakukakan seperti layaknya 2 PC.

Jika kita menggunakan satu PC saja, kita bisa menggunakan alamat Ip Address Loopback. Alamat Ip Address Loopback biasanya disebut localhost atau 127.0.0.1.

2.3 Motor Tiga Fasa

Motor tiga fasa adalah suatu motor AC yang menggunakan suplay tegangan tiga fasa. Dimana tegangan AC tiga fasa memiliki 4 hantaran dimana dari keempat hantaran tersebut memiliki tiga fasa yang diberi nama R, S, T, dan satu hantaran netral.

Pada tegangan tiga fasa memiliki beda fasa dari R, S, maupun pada T yang dapat mengakibatkan perputaran pada motor (Ardiono, 2004 : 17). Untuk memperjelas dapat dilihat pada Gambar 2.7.

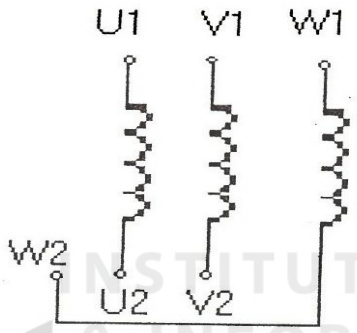


Gambar 2.7 Beda Fasa pada RST.

Tujuan penggunaan motor tiga fasa ini yaitu agar sistem lebih stabil, karena motor tiga fasa lebih stabil jika di bandingkan dengan motor AC satu fasa, motor tiga fasa juga memiliki torsi yang lebih besar.

2.2.1 Struktur Motor Tiga Fasa

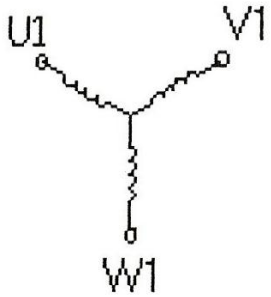
Di dalam motor tiga fasa terdapat tiga lilitan dimana dari ketiga lilitan tadi terdapat 6 buah hantaran yang dijadikan dua *group* yaitu U1, V1, W1 dan U2, V2, W2 untuk lebih jelasnya lihat Gambar 2.8.



Gambar 2.8 Struktur Motor Tiga Fasa.

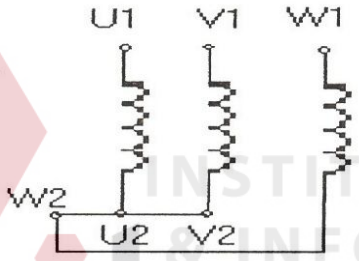
Pada motor tiga fasa terdapat beberapa cara untuk mengendalikan motor tiga fasa diantaranya dengan menggunakan struktur *star* dan delta. Keduanya memiliki kelebihan dan kekurangan, diantaranya :

a. Struktur *star* memiliki kelebihan arusnya lebih kecil jika dibandingkan dengan delta. Namun pada struktur *star*, motor akan lebih lama untuk mencapai kestabilan. Oleh karena itu, star banyak digunakan pada sistem yang membutuhkan kestabilan dalam waktu yang relatif cepat. Jika suplay tegangan yang digunakan adalah 220/380 maka struktur *star* akan dipakai untuk tegangan 380 volt. Hubungan dari struktur *star* adalah seperti Gambar 2.9.



Gambar 2.9. Struktur *Star*.

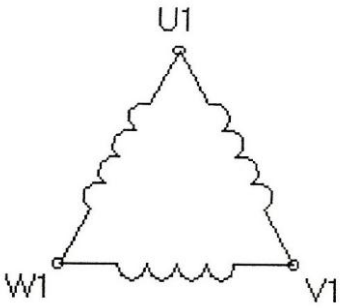
Jadi hubungan untuk struktur *star* pada motor tiga fasa tampak seperti pada Gambar 2.10.



Gambar 2.10. Hubungan Struktur *Star* pada Motor Tiga Fasa.

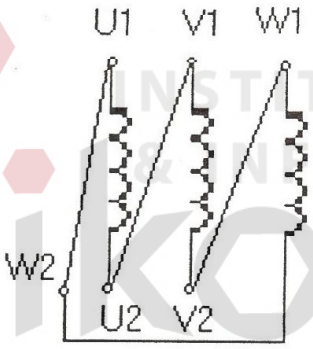
b. Sedangkan pada struktur delta biasanya digunakan untuk motor-motor dengan arus besar. Dengan menggunakan struktur delta, start pada motor lebih cepat sehingga kestabilan pada putaran motor akan lebih cepat dicapai dibandingkan dengan struktur *star*, namun pada struktur delta dibutuhkan arus yang lebih besar daripada struktur *star*. Oleh karena itu, pada sebagian motor digunakan keduanya yaitu delta untuk mendapatkan kestabilan motor yang cepat, kemudian *star* agar arus yang terpakai kecil. Jika suplay tegangan yang digunakan adalah 220/380 maka struktur delta

akan dipakai untuk tegangan 220 volt. Hubungan dari struktur delta adalah seperti pada Gambar 2.11.



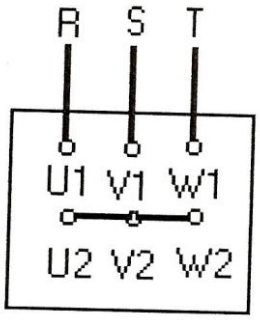
Gambar 2.11. Struktur Delta.

Jadi hubungan untuk struktur delta pada motor tiga fasa tampak seperti pada Gambar 2.12.



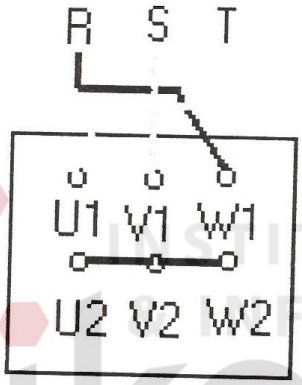
Gambar 2.12 Hubungan Struktur Delta pada Motor Tiga Fasa.

Pada motor AC tiga fasa arah putaran motor (rotasi) dapat diubah dengan mengubah posisi R dan T. Jika R dihubungkan dengan U1 dan S dengan V1 dan T dengan W1 maka motor akan berputar ke kanan. Hubungan listriknya tampak pada Gambar 2.13.



Gambar 2.13. Rangkaian Pembalik Arah Ke Kanan.

Namun jika posisi R dan T dibalik seperti pada Gambar 2.14, dimana R dihubungkan dengan W1 dan T dengan U1 maka motor akan berputar ke kiri.



Gambar 2.14 Rangkaian Pembalik Arah Ke Kiri.

Sumber : Ardiono, Rachman, dkk. 2004. *Laporan Kerja Praktek Di PT. FESTO Surabaya*.

2.3 *Inverter* VF-S11

Inverter adalah peralatan elektronik yang melakukan konversi arus dari arus searah ke arus bolak-balik. Tegangan dan frekuensi arus hasil konfersi tergantung dari transformator dan alat kontrol yang digunakan. Penggunaan *inverter* antara lain sebagai berikut :

1.DC *power utilitation.*

2. *Uninterrutpible Power Supplies* (UPS).

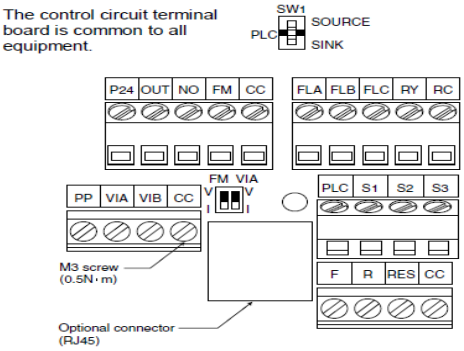
4. *Air Conditioning.*
5. Menjalankan motor AC, dll.

Pada tugas akhir ini aplikasi *inverter* yang diterapkan adalah untuk menjalankan motor AC. Pengontrolan motor AC tidak sama dengan pengontrolan motor DC. Pada motor DC untuk mengubah arah putaran motor kita cukup merubah polaritas tegangan pada motor. Sedangkan untuk mengatur kecepatan motor DC dapat dilakukan dengan cara memberi sinyal *Pulse Width Modulation* (PWM), yaitu suatu sinyal yang mempunyai lebar data *high* dan *low* yang berbeda. Metode pengontrolan pada motor DC tidak dapat digunakan pada motor AC. Pengaturan arah putaran motor AC tiga fasa dapat dilakukan seperti yang telah dibahas pada subbab 2.3.1, sedangkan untuk pengaturan kecepatannya dilakukan dengan dengan cara yang lebih rumit, dibandingkan dengan memberikan sinyal PWM untuk motor DC. Dengan menggunakan *inverter*, kita dapat merubah arah putaran, merubah kecepatan, melakukan *emergency break*, dan sebagainya. Pada motor AC, hanya dengan memberikan sinyal inputan yang sesuai pada *inverter*.

2.3.1 Koneksi

Pada *terminal board* terdapat *terminal-terminal* yang digunakan untuk memberikan sinyal masukan yang diperlukan untuk pengontrolan *inverter*.

Gambar dari *terminal board* seperti Gambar 2.15.



Gambar 2.15 *Terminal Board*.

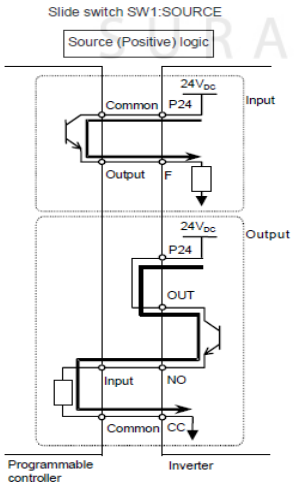
Sumber : Toshiba Schneider Inverter Corporation. 2003.

Kegunaan dari *terminal-terminal* yang ada dapat dilihat dalam lampiran 2. Kegunaan SW1 adalah untuk memilih mode yang digunakan pada saat mengoperasikan *inverter*, yaitu mode *source*, PLC, dan *sink*.

2.3.2 Mode Operasi *Inverter*

A. Mode *Source*

Pada mode operasi ini sinyal inputan diberikan pada *terminal* masukan dengan cara menghubungkannya dengan terminal P24. Koneksi mode ini dapat dilihat pada Gambar 2.16.

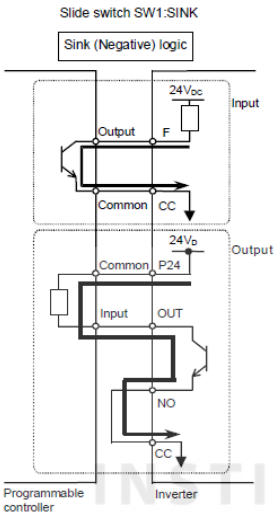


Gambar 2.16. Mode *Source*.

Sumber : Toshiba Schneider Inverter Corporation. 2003.

B. Mode *Sink*

Pada mode operasi ini sinyal inputan diberikan pada *terminal* masukan dengan cara menghubungkannya dengan terminal CC. Koneksi mode ini dapat dilihat pada Gambar 2.17.

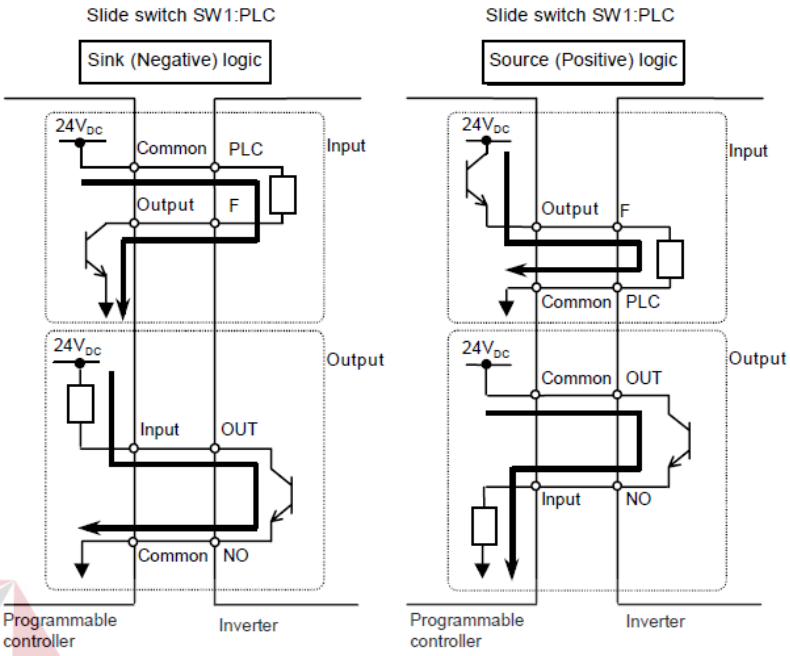


Gambar 2.17 Mode *Sink*.

Sumber : Toshiba Schneider Inverter Corporation. 2003

C. Mode PLC

Pada mode operasi ini sinyal inputan diberikan pada *terminal* masukan dengan cara menghubungkan *ground* PLC dengan terminal PLC, lalu menghubungkan terminal masukan dengan *output* dari PLC. Koneksi mode ini dapat dilihat pada Gambar 2.18.



Gambar 2.18 Mode PLC.

Sumber : Toshiba Schneider Inverter Corporation. 2003

2.3.3 Parameter *Inverter*

Parameter yang digunakan pada kerja praktek ini adalah sebagai berikut :

1. CNOD.
2. FNOD.
3. SR1-SR7.
4. F287-F294.

Untuk mode-mode yang lain dapat dilihat di Lampiran 3.

A. CNOD

CNOD adalah parameter yang digunakan sebagai *command mode selection*. Sinyal-sinyal masukan ke *inverter* dapat dikirimkan melalui dua cara yaitu, melalui *terminal board* atau melalui *operation panel*.

Untuk merubah arah putaran motor AC tidak dapat dilakukan melalui *operation panel*. Sedangkan untuk operasi melalui *terminal board* kita dapat merubah kecepatan motor, merubah arah putaran motor, mengatur control PID, dan lain sebagainya. Pada tabel 2.3 dijelaskan bagaimana mensetting inverter sesuai dengan yang sudah ditetapkan pada perangkat keras inverter Toshiba VF-S11.

Tabel 2.3 CNOD.

Title	Function	Adjustment range	Default setting
CN0d	Command mode selection	0 Terminal board 1 Panel	1
FN0d	Frequency setting mode	0 Internal potentiometer setting 1 VIA 2VIB 3 Operation panel 4 Serial communication 5 External contact up/down 6 VIA+VIB (Override)	0

Sumber : Toshiba Schneider Inverter Corporation. 2003.

B. FNOD

FNOD digunakan untuk menentukan *frequency setting mode*. Pengaturan frekuensi dapat dilakukan dengan tujuh cara yaitu :

a. Potensiometer pada *operation panel*.
b. VIA.
c. VIB.
d. Tombol atas dan bawah pada *operation panel*.
e. Komunikasi serial.
f. Saklar eksternal.
g. VIA dan VIB.

1. **Potensiometer Pada *Operation Panel***

Cara ini dapat dipilih dengan cara memberikan nilai 0 pada parameter FNOD. Pengaturan frekuensi dilakukan dengan cara memutar potensiometer yang terdapat pada *operation panel*.

2. VIA

Cara ini dapat dipilih dengan cara memberikan nilai 1 pada parameter FNOD. Pengaturan frekuensi dilakukan dengan cara memberikan tegangan 0-10Vdc atau 4-20m Adc pada *terminal* VIA.

3. VIB

Cara ini dapat dipilih dengan cara memberikan nilai 2 pada parameter FNOD. Pengaturan frekuensi dilakukan dengan cara memberikan tegangan 0-10Vdc pada *terminal* VIB.

4. Tombol Atas dan Bawah Pada *Operation Panel*

Cara ini dapat dipilih dengan cara memberikan nilai 3 pada parameter FNOD. Pengaturan frekuensi dapat dilakukan dengan cara menekan tombol atas dan bawah yang terdapat pada *operation panel*.

5. Komunikasi Serial

Cara ini dapat dipilih dengan cara memberikan nilai 4 pada parameter FNOD. Pengaturan frekuensi dapat dilakukan dengan cara komunikasi serial. Komunikasi serial dapat dilakukan dengan komputer ataupun dengan mikrokontroler.

6. Saklar Eksternal

Cara ini dapat dipilih dengan cara memberikan nilai 5 pada parameter FNOD. Pengaturan dapat dilakukan dengan cara memberi inputan pada *terminal* S1, S2,

S3, dan RES. Terminal-terminal tersebut digunakan sebagai selektor kecepatan (total ada 16 *presets* kecepatan). S1 berfungsi sebagai LSB dan RES berfungsi sebagai MSB.

7. VIA dan VIB

Cara ini dapat dipilih dengan cara memberikan nilai 6 pada parameter FNOD. Pengaturan frekuensi dilakukan dengan cara menjumlah nilai yang masuk pada *terminal* VIA dan VIB.

C. SR1-SR7

Parameter ini digunakan untuk memberi nilai pada *presets* kecepatan operasi frekuensi satu sampai tujuh.

D. F287-F294

Parameter ini digukanan untuk memberi nilai pada *presets* kecepatan operasi frekuensi delapan sampai lima belas. Untuk dapat memakai *presets* delapan sampai lima belas, kita harus memberi nilai 9 pada parameter F113.

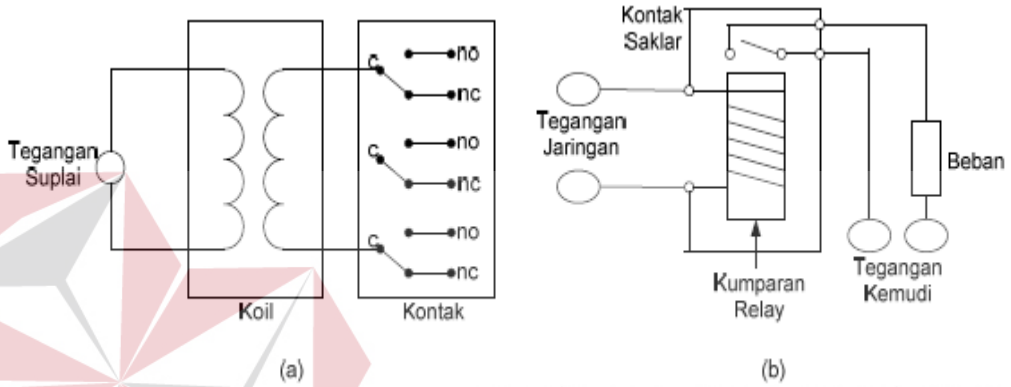
Sumber : Toshiba Schneider Inverter Corporation. 2003. *Instruction Manual TOSVERT TM VF-S11*. Toshiba Corporation : Japan.

2.4 *Relay*

Relay merupakan suatu alat yang berfungsi sebagai *switch* elektronik dimana penggeraknya terbuat dari lilitan kawat tembaga. Pada dasarnya sebuah lilitan tembaga pada sebuah inti besi yang mana bila kedua ujungnya dihubungkan dengan sumber tegangan, maka akan timbul medan magnet pada inti besi tersebut (Omega, 2005 : 26). Untuk lebih jelasnya dapat dilihat pada Gambar 2.19.

Sedangkan kontak yang merupakan saklar terdapat dua macam kondisi dari kontak tersebut, yaitu :

a. Normally Open (NO), yaitu kontak akan aktif pada saat koil di suplai tegangan.
b. Normally Closed (NC), yaitu kontak akan aktif pada saat koil tidak di suplai tegangan.



Gambar 2.19. (a) *Relay*. (b) Bagian dalam *relay*.

Pada sebuah inti besi yang menimbulkan medan magnet akan menarik sebuah lempengan besi dari kontaktor, sehingga akan menyebabkan titik satu dengan titik lainnya akan tersambung.

NC (mengunci).

2.5 Optocoupler

Optocoupler merupakan komponen elektronika yang memanfaatkan sinar sebagai pemicu on/off-nya. Sumber cahaya yang digunakan antara lain led *infra-red*, dan penerimanya menggunakan *phototransistor*. Pada saat *optocoupler* terhalangi maka keluaran akan bernilai 1, dan jika tidak terhalangi apapun maka keluaran akan bernilai 0. Ada beberapa tipe dari *optocoupler*, antara lain 4N25 atau 4N33.