BAB II

LANDASAN TEORI

2.1. Pakan Ikan Buatan

Pakan ikan buatan adalah pakan ikan yang dibuat dengan mengkombinasikan bahan-bahan tertentu kemudian dicampurkan sesuai dengan ketentuan-ketentuan yang ada. Biasannya pakan ikan dibuat dalam berberapa bentuk. Berikut adalah macam-macam bentuk ikan buatan yang biasanya ada dipasaran.

2.1.1. *Crumble*

Crumble adalah pakan ikan buatan yang biasanya dibuat dalam bentuk remah-remah. Pakan semacam ini biasanya dibuat untuk ikan yang masih dalam bentuk anakan (kecil), karena pakan semacam ini lebih mudah dicerna. Kelemahan pakan semacam ini adalah karena ukurannya yang halus maka mudah larut dalam air sehingga tingkat pencemaran air jadi meningkat.

2.1.2. Pelet

Pelet ikan adalah salah satu jenis pakan ikan buatan yang paling banyak dipergunakan petani ikan dalam proses pembudidayaan ikan (Gambar 2.1). Pelet ikan terdiri dari berberapa campuran komposisi pakan ikan yang telah diformulasikan terlebih dahulu. Komposisi ini biasanya terdiri dari bahan-bahan

yang memiliki gizi yang dibutuhkan oleh ikan seperti karbohidrat, protein, lipid dan sebagainya.

Berberapa komposisi bahan tidaklah sama untuk setiap jenis ikan tergantung kepada jenis kelompok ikan (herbivora, karnivora, omnivora), habitat, serta ukuran dan umur dari budi daya ikan itu sendiri (*http://www.o-fish.com, 2008)*. Bahan-bahan komposisi diatas kemudian dicampurkan dengan binder (pasta). Pasta di sini berfungsi untuk sebagai lem perekat antar bahan-bahan campuran. Pasta dapat berupa galatin atau tepung tapioka.

Pelet biasanya diberikan untuk ikan yang telah besar atau dewasa. Biasanya diberikan ketika ikan sudah berumur lebih dari 120 hari masa perkembanganya (*http://www.iptek.net.id, 2008*). Pelet yang telah jadi adalah pelet yang dapat mengapung berberapa saat dan memiliki tingkat ketahanan yang bagus ketika berada dalam air (tidak mudah hancur). Pelet biasanya dicetak dalam bentuk padat dan kering serta memiliki ukuran yang berbeda–beda, hal ini dimaksudkan agar pakan ikan yang diberikan dapatlah dicerna oleh ikan dengan baik dan dapat mengurangi tingkat pencemaran air kolam yang disebabkan adanya pakan ikan yang berlebihan.

Berdasarkan ukuran, ikan dapat dikompokkan atas 3 macam : ikan ukuran kecil (biasanya ikan semacam ini adalah ikan-ikan hias yang berukuran kecil seperti ikan mas koki, ikan nila, dan lainnya), ikan jenis ini memakan pelet dengan ukuran kecil. Ikan ukuran sedang (biasanya sejenis ikan gurami, ikan mas, ikan lele dan lain-lain), ikan jenis ini memakan pelet dengan ukuran sedang dan ukuran kecil. Ikan ukuran besar (biasanya sejenis ikan kerapu, ikan patin dan

sejenisnya), ikan ini biasanya memakan pelet dengan ukuran yang besar, sedang dan kecil.

2.1.3. *Flake*

Flake adalah pakan ikan buatan yang dibuat dalam bentuk lembaran-lembaran (Gambar 2.1). Pakan ini biasanya dipergunakan sebagai pakan ikan tambahan untuk ikan hias. Pakan semacam ini memiliki kelemahan yakni mudah larut dalam air.

Gambar 2.1 Macam-macam Pakan Ikan

2.2. *Programmable Logic Controller* (PLC)

PLC pertama kali dikembangakan oleh para insinyur dari General Motor pada tahun 1968 yaitu pada saat perusahaan tersebut ingin mengganti sistem kontrol relay yang kompleks. Sistem pengganti tersebut harus mampu dalam :

1. Pemprograman yang sederhana.
2. Pengubahan program tanpa harus mengubah sistem secara keseluruhan.
3. Lebih kecil dalam ukuran, lebih murah dan dapat diandalkan.

4. Perawatan yang mudah dan biaya perawatan yang rendah.

Dimana akhirnya didapatkan sebuah sistem yang hanya menggunakan sinyal biner. Keuntungannya adalah bahwa sinyal ini dapat digunakan dalam kontrol program dan dapat di proses secara digital serta disimpan didalam memori elektronik. Sinyal-sinyal ini selanjutnya dapat digunakan untuk menggerakkan motor atau silinder sebab pada prinsipnya sebuah output hanya memerlukan dua status keadaan saja yaitu “0” dan “1” (Indrijono, 1).

Programmable Logic Controller (PLC) adalah suatu bentuk khusus pengontrolan berbasis mikroprosesor yang memanfaatkan memori, dimana memori tersebut dapat diprogram untuk menyimpan instruksi-instruksi dan mengimplementasikan fungsi-fungsi semisal logika, *sequencing*, pewaktuan (*timing*), pencacahan (*counting*) dan aritmatika guna mengontrol mesin-mesin dan proses-proses. PLC dirancang untuk dioperasikan oleh para insinyur yang hanya memiliki sedikit pengetahuan mengenai komputer dan bahasa pemprograman. Piranti ini dirancang sedemikian rupa agar tidak hanya programmer komputer saja yang dapat membuat atau mengubah program-programnya (Bolton W, 3).

PLC memiliki keunggulan yang signifikan, karena sebuah perangkat pengontrolan yang sama dapat dipergunakan di dalam beraneka ragam sistem kontrol. Untuk memodifikasi sebuah sistem kontrol dan aturan-aturan pengontrolan yang dijalankannya, yang harus dilakukan oleh seorang operator hanyalah memasukkan seperangkat intruksi yang berbeda dari yang digunakan sebelumnya.

Dewasa ini PLC secara luas digunakan dan dikembangkan dari unit-unit kecil yang berdiri sendiri (*self contained*) yang hanya memiliki input output

terbatas, sampai unit-unit besar dengan jumlah input output yang banyak, sehingga mampu meng-*handle* atau melakukan perngontrolan secara besar.

2.3. Konsep dan Fungsi Umum PLC

2.3.1. Konsep PLC

Sesuai dengan namanya, konsep dari PLC adalah sebagai berikut :

Menunjukkan kemampuan PLC yang dapat dengan mudah diubah-ubah sesuai program yang dibuat dan kemampuannya dalam memproses input secara aritmetik (membandingkan, menjumlah, membagi dan sebagainya) serta kemampuan PLC dalam mengontrol dan mengatur proses sehingga menghasilkan output yang diinginkan.

2.3.2. Fungsi PLC

Fungsi PLC secara umum adalah:

1. *Control Sequence*

   Kemampuan PLC memproses input sinyal menjadi sinyal output yang untuk kemudian digunakan sebagai keperluan pemprosesan teknik secara berurutan (*sequence*) dan PLC menjaga agar semua langkah (STEP) dalam proses *sequence* dapat berlangsung dalam urutan yang tepat.

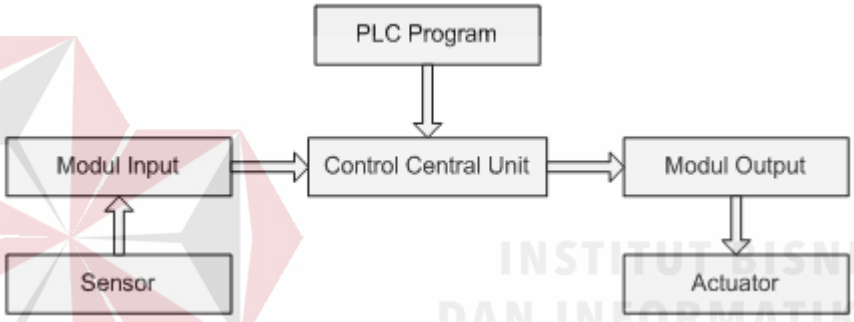
2. *Monitoring Plant*

   Monitoring plant adalah seperangkat peralatan yang digunakan untuk menjalankan suatu operasi tertentu. PLC secara terus-menerus memonitor suatu sistem, dan mengambil tindakan sesuai program

sehubungan dengan proses yang dikontrol atau bisa saja dengan menampilkannya pada operator.

2.4. Perangkat Keras PLC

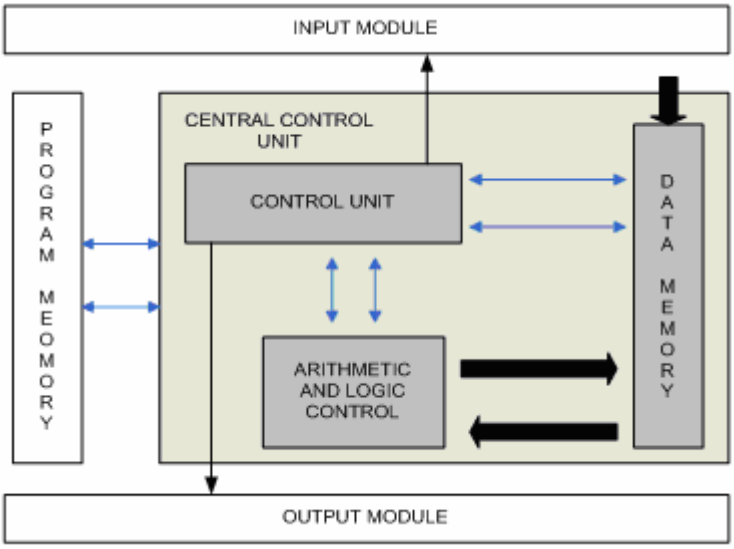
PLC sederhana mempunyai komponen utama berupa *Central Control Unit* (CCU), unit I/O, *Programming Console*, dan catu daya. Secara umum sistem komponen dari sebuah PLC adalah seperti pada Gambar 2.2.



Gambar 2.2 Sistem komponen dari PLC

2.4.1. *Central Control Unit* (CCU)

Central Control Unit (CCU) merupakan unit yang berisi mikroprossesor yang menginterpretasikan sinyal-sinyal input dan melaksanakan tindakan-tindakan pengontrolan, sesuai dengan program yang tersimpan dalam memori, lalu mengkomunikasikan keputusan-keputusan yang diambil sebagai sinyal-sinyal kontrol ke antarmuka output (Bolton W, 4). CCU dari sebuah PLC dapat diuraikan seperti pada Gambar 2.3.



Gambar 2.3 *Central Control Unit* dari sebuah PLC

Struktur internal CCU tergantung pada mikroprosessor yang bersangkutan. Pada umumnya komponen-komponen struktur tersebut adalah:

1. Unit aritmetika dan logika (*Aritmetic and logic unit*) (ALU).

   Bertugas untuk menangani manipulasi data dan melaksanakan operasi aritmetika seperti penjumlahan dan pengurangan serta operasi-operasi logika AND, OR, NOT, dan OR-EKSKLUSIF.

2. Memori.

   Dinamakan juga register, terletak dalam mikroprosessor dan dipergunakan untuk menyimpan informasi yang terlibat dalam pengeksekusian program.

3. Sebuah unit kontrol (*Control Unit*).

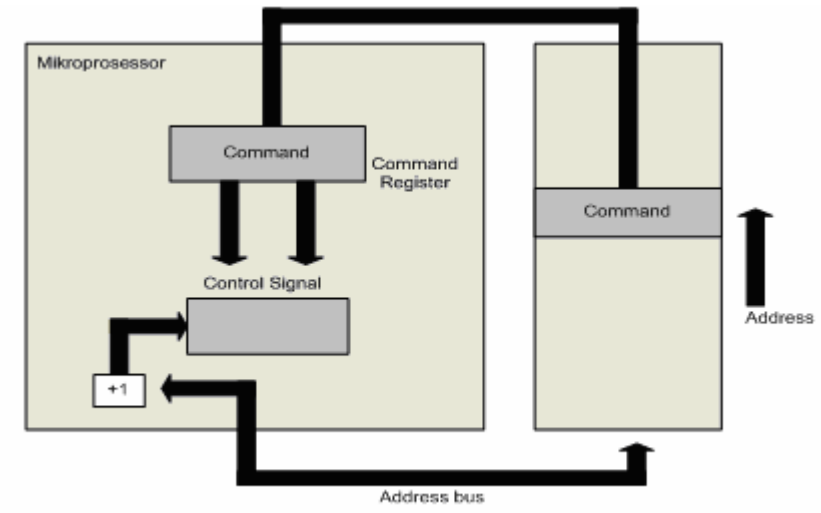
   Digunakan untuk pengontrolan pemwaktuan operasi-operasi.

Pemprosesan sebuah program oleh CCU secara sederhana dapat dijelaskan sebagai berikut :

1. Program memori berisi perintah-perintah yang harus dikerjakan, CCU mempunyai akses ke program memori.
2. Unit kontrol menghendaki data input melalui modul input dan memprosesnya di ALU yang juga terhubung dengan perintah yang tersimpan dalam memori program.
3. Output data dikeluarkan oleh *control unit* melalui modul output ke sistem yang dikontrol.

PLC memproses program-program dengan cara baris demi baris, artinya setiap baris program dari seluruh program diproses secara berurutan (Gambar 2.4). Sebagai patokan setiap baris program, seperti sebuah perintah yang diproses dalam dua langkah :

1. mengambil perintah (*fetch*) dari program memori.
2. mengeksekusi perintah tersebut.



Gambar 2.4 *Command Sequence*

Isi dari program *counter* ditransfer ke bus alamat. *Control unit* kemudian menyebabkan perintah yang sesuai dengan alamat dalam program memori ditempatkan dalam *bus* data. Perintah ini kemudian dibaca dan diletakkan di *instruction register*. Pada saat perintah telah diterjemahkan, *control unit* kemudian akan menghasilkan urutan sinyal kontrol untuk proses eksekusi. Selama eksekusi program, perintah di *fetch* secara berurutan. Ini disebabkan dengan adanya *increment* alamat secara otomatis yang dilakukan oleh program *counter* untuk mendapatkan alamat program berikutnya yang akan dikerjakan (Indrijono, 9).

Program untuk pemprosesan data PLC tidak seperti dengan program untuk pemrosesan data secara biasa. Dalam PLC setelah program selesai pada baris terbawah dari program, maka secara otomatis program akan dijalankan kembali dari awal program (pemprosesan sirkular). Demikian hal ini dilakukan secara terus-menerus.

Karakteristik dari pemrosesan sirkular adalah sebagai berikut :

1. Segera setelah suatu program dieksekusi sekali, secara otomatis akan meloncat (*jump*) awal dari program dan proses diulangi lagi.
2. Sebelum baris pertama program diproses, seperti pada awal dari *cycle*, status dari semua input disimpan dalam tabel *image*. Proses *image* adalah lokasi memori yang dapat diakses dalam sebuah *cycle*. Status dari suatu input dipertahankan tetap selama masih dalam sebuah *cycle*, walaupun fisik dari *cycle* tersebut telah diubah.
3. Mirip dengan input, semua output juga disimpan dalam tabel output. Hanya akhir dari suatu *cycle* semua outputnya secara fisik diubah sesuai dengan status logika yang disimpan dalam memori.

Waktu yang dibutuhkan oleh PLC selama satu baris perintah dikerjakan termasuk aktualisasi dan pemrosesan *image* disebut *cycle time*. Realistisnya periode waktu ini mendekati antara 1 dan 100 millidetik.

Adapun jenis mikroprocessor yang dipergunakan pada PLC tergantung dari *merk* dan tipe PLC-nya. Misalnya pada PLC OMRON tipe C200HS menggunakan mikroprocessor Z-80. PLC MODICON tipe 984/e menggunakan mikroprocessor intel 8088 sedangkan PLC FESTO DIDATIC seri FPC 100 menggunakan mikrokontroler 8031.

2.4.2. Memori

Memori yang terdapat dalam PLC berfungsi sebagai tempat dimana program yang akan digunakan untuk melaksanakan tindakan-tindakan pengontrolan oleh mikroprosessor disimpan (Bolton W, 7). Memori terdiri dari *Read-only Memory* (*ROM*), *Random-Access Memory* (*RAM*), *Random-Access Memory* (*RAM*) untuk data dan *Erasable and Programmable Read-Only-Memory* (*EPROM*). *Read-only Memory* (*ROM*) adalah sistem yang menyediakan fasilitas penyimpanan permanen untuk sistem operasi dan data tetap yang digunakan oleh CPU, *Random-Access Memory* (*RAM*) untuk program sang pengguna sedangkan *Random-Access Memory* (*RAM*) untuk data merupakan tempat disimpannya informasi mengenai status perangkat-perangkat input dan output dan nilai-nilai *timer* (piranti pewaktuan) dan *counter* (piranti pencacah) dan perangkat internal lainnya. RAM data kadangkala disebut sebagai *table* data atau *table* register. Sebagian dari memori ini, yaitu blok, alamat, diperuntukkan bagi alamat-alamat input dan output dan status masing-masing input dan output tersebut. Sebagian

lainnya disisihkan untuk menyimpan data telah ditetapkan sebelumnya (*preset*) dan sisanya untuk menyimpan nilai-nilai *counter*, nilai *timer* dan lain-lainnya. Sedangkan *Erasable and Programmable Read-Only-Memory* (*EPROM)* yaitu ROM yang dapat diprogram dan setelah itu progam tersebut secara permanen tersimpan di dalamnya.

2.4.3. Modul Input dan Modul Output (I/O)

Unit input ouput menyediakan antarmuka yang menghubungkan sistem dengan dunia luar, memungkinkan dibuatnya sambungan-sambungan (atau koneksi) antara perangkat-perangkat input, semisal sensor dengan perangkat-perangkat output, semisal motor dan solenoid, melalui kanal-kanal input dan output. Setiap titik input output memiliki sebuah alamat yang unik yang dapat digunakan oleh CCU. (Bolton W, 8)

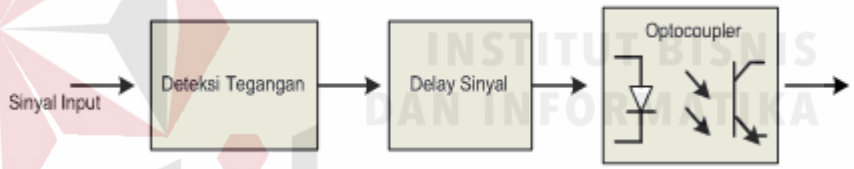
Setiap input output memiliki nomor dan alamat urutan khusus yang digunakan selama membuat program untuk memonitor satu persatu aktivitas input dan output didalam program. Untuk mempermudah penggunaan I/O maka digunakan terminal I/O berupa kanal. Tiap-tiap kanal I/O memiliki suatu alamat tersendiri yang digunakan pada program. Kanal-kanal input output menyediakan fungsi-fungsi isolasi dan pengkondisian sinyal sehingga sensor-sensor dan aktuator-aktuator seringkali dapat disambungkan padanya tanpa membutuhkan rangkaian tambahan apapun.

A. Modul Input

Adalah modul tempat menghubungkan sensor-sensor dengan modul itu sendiri. Sinyal sensor tersebut selanjutnya akan di teruskan ke CCU. Fungsi terpenting dari sebuah modul input adalah sebagai berikut :

1. Mendeteksi sinyal masukan.
2. Mengatur tegangan kontrol untuk batas tegangan logika masukan yang diijinkan.
3. Melindungi peralatan elektronik yang sensitif terhadap tegangan luar.
4. Menampilkan sinyal masukan tersebut

Blok diagram dari modul input dapat dilihat pada Gambar 2.5.



Gambar 2.5 Blok diagram modul input

Deteksi tegangan error meyakinkan bahwa tegangan masuk masih dalam batas yang diijinkan atau tidak. Bila tegangannya terlalu tinggi akan diturunkan melalui *diode breakdown.*

Delay sinyal meyakinkan apakah tegangan yang diterima sudah merupakan input yang sebenarnya atau bukan. Rangkaian ini mempertahankan tegangan input sesaat (1-20 ms) untuk membedakannya dengan sinyal-sinyal lain seperti tegangan interferensi.

Optocoupler mengirimkan informasi sensor berupa cahaya dan menciptakan isolasi elektronik antara kontrol dan rangkaian logika, selanjutnya

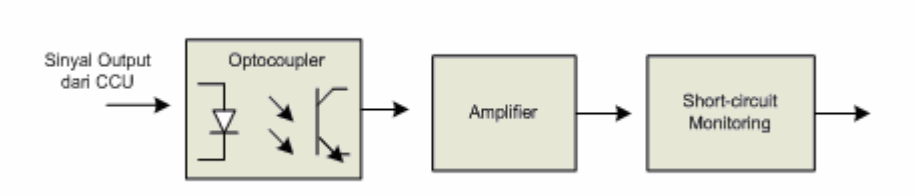
melindungi komponen elektronik yang sensitif dari naiknya tegangan luar secara tiba-tiba. Terdapat *optocoupler* yang mampu memberikan perlindungan terhadap tegangan sampai dengan 5 kv (5000 Volt) yang sesuai dengan aplikasi untuk industri

B. Modul Output

Modul output mengeluarkan sinyal dari CCU ke kontrol elemen yang diperlukan untuk menggerakkan aktuator sesuai dengan tugas yang telah diberikan. Fungsi terpenting dari modul output adalah sebagai berikut :

1. Mengatur tegangan kontrol untuk batas tegangan logika keluaran yang diijinkan.
2. Melindungi peralatan elektronik yang sensitif terhadap tegangan luar.
3. Memberikan penguatan sinyal output sebelum dikeluarkan sehingga cukup kuat untuk menggerakkan aktuator.
4. Memberikan perlindungan terhadap arus hubung singkat (*short-circuit*) dan pembebanan lebih (*overload).*

Blok diagram dari modul output dapat dilihat pada Gambar 2.6.



Gambar 2.6 Blok diagram modul output

Optocoupler adalah bentuk dasar dari power secara elektronik yang memberikan perlindungan terhadap komponen elektronik dan juga berfungsi

untuk pengatur tegangan output. Saat ini perlindungan terhadap *short-circuit* dan *overload* serta *power amplification* telah dikemas dalam satu rangkaian terpadu berupa modul-modul melalui hubungan transistor secara Darlington atau lainnya.

Amplifier berguna untuk menguatkan arus listrik output sehingga nantinya cukup kuat untuk menggerakkan aktuator.

Short circuit monitoring memonitor jika terjadi arus hubung singkat pada rangkaian luar dan memutuskan hubungan antara modul output dengan rangkaian luar.

Kisaran sinyal input yang mungkin tersedian pada sebuah PLC berskala besar adalah sinyal-sinyal digital/diskrit (yaitu sinyal ‘hidup’ atau ‘mati’) 5 volt, 24 volt, 110 volt dan 220 volt, sedangkan untuk PLC berukuran kecil kemungkinan memiliki satu bentuk input 24 volt.

2.4.4. Unit Catu Daya

Unit ini dipergunakan untuk mengkonversikan tegangan arus a.c sebagai sumber menjadi tegangan rendah d.c yang dibutuhkan oleh prosesor dan rangkaian didalam modul-modul antarmuka input dan output PLC.

2.5. Perangkat Lunak PLC

Pada PLC terdapat berberapa pilihan atau alternatif bahasa pemprograman yang dapat dibuat. Masing-masing bahasa memiliki struktur program dan penulisan yang berbeda-beda. *Ladder diagram* misalnya adalah salah satu bahasa program yang dimiliki oleh setiap PLC. Tetapi pada PLC FESTO DIDATIC seri FPC 101-B LED selain menggunakan bahasa pemprograman

ladder diagram juga terdapat bahasa pemprograman lain yang dapat dipergunakan yakni *statement list.*

2.5.1. *Ladder Diagram* (LDR)

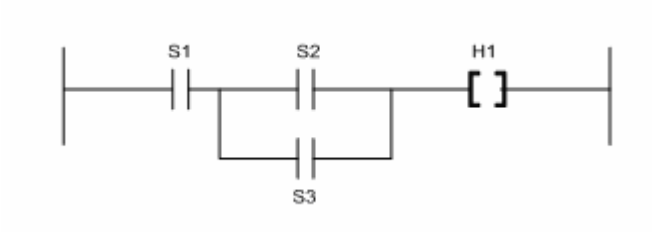
Merupakan salah satu bahasa pemprograman pada PLC dimana ladder diagram memilki tipe dan elemen program yang membedakannya dengan bahasa pemprograman PLC lainnya.

A. Tipe Program

Ladder diagram menggambarkan program dalam bentuk grafik. Diagram ini dikembangakan 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 tegangan positif atau rel catu daya aktif sedangkan garis sebelah kanan dengan sumber tegangan negatif atau rel catu daya pasif.

B. Elemen Program

Diantara dua garis ini dipasang kontak-kontak yang menggambarkan kontrol dari *switch*, sensor atau output. Satu baris diagram tersebut disebut juga satu RUNG. Input menggunakan “[ ]” (kontak, normal *open*) dan “[/]” (negasi kontak, normal *closed*). Output mempunyai simbol “( )” yang terletak paling kanan menempel garis vertikal kanan. Selama pemprograman, setiap simbol yang diberikan adalah alamat PLC sesungguhnya atau merupakan alamat *symbolic* (misalkan S1, S2, S3, H1). Salah satu contoh bentuk penulisan suatu program dalam *ladder diagram* dapat dilihat pada Gambar 2.7.



Gambar 2.7 *Ladder Diagram*

2.5.2. *Statement List* (STL)

Merupakan bahasa pemprograman yang menggunakan kalimat dalam penulisannya. Sama halnya denga *ladder diagram*, pada *statement list* juga memiliki tipe dan struktur yang berbeda dengan *ladder diagram*.

A. Tipe Program

Selain menggunakan *ladder diagram* tedapat cara lain untuk proses pemprograman pada PLC, yakni dengan menggunakan *statement list*. *Statement List* adalah salah satu bahasa pemprograman untuk PLC dimana semua hubungan logika dan *control sequence* dapat diprogram dengan menggunakan perintah dalam bahasa ini (indrijono, 24).

B. Struktur program

Struktur dari *statement list* secara umum dapat dituliskan sebagai berikut :

PROGRAM

STEP

KALIMAT

BAGIAN KONDISI

BAGIAN PELAKSANAAN

Struktur dari *statement list* dapat dijelaskan sebagai berikut :

1. STEP

Program yang tidak menggunakan instruksi STEP dapat diproses dengan cara paralel (*scanning*). Tetapi STL menyediakan instruksi STEP yang membagi program menjadi bagian-bagian yang lebih kecil. Dalam sebuah program dapat berisi sampai 256 STEP (0 sampai 255). Dimana setiap STEP dapat diberi label atau tidak, dan hanya dibutuhkan jika setiap STEP tersebut merupakan target dari intruksi JUMP. Setiap STEP harus setidaknya berisi satu kalimat dan dapat digabungkan dengan beberapa kalimat.

Contoh :

```
STEP (label)
IF                i2
THEN    SET       o3

IF      N         i3
THEN    RESET     o4
```

Jika kondisi IF pertama (IF i2 (input)) salah, maka program tidak melompat ke STEP berikutnya tetapi akan melakukan pengecekan terhadap kondisi IF yang kedua (IF i3 (input)). Dan jika kedua tidak terpenuhi maka program akan kembali ke kalimat pertama. Dengan kata lain program akan menunggu sampai salah satu kondisi terpenuhi.

Terdapat beberapa aturan pelaksanaan STEP :

a. Jika kondisi dari sebuah kalimat terpenuhi maka bagian pelaksana akan dijalankan.

b. Jika kondisi dari kalimat terakhir dalam suatu STEP terpenuhi maka bagian pelaksana akan menjalankan dan program berlanjut ke STEP berikutnya.

c. Jika kondisi dari sebuah kalimat dalam suatu STEP tidak terpenuhi maka program akan berpindah ke kalimat berikutnya dalam STEP tersebut.

d. Jika kondisi dari kalimat terakhir dalam suatu STEP tidak terpenuhi maka program akan kembali ke kalimat pertama dari STEP yang sekarang.

2. Kalimat

Kalimat merupakan pembentukan dasar dari organisasi program. Masing-masing kalimat terdiri dari bagian kondisi dan bagian pelaksanaan.

3. Bagian Kondisi

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).

Contoh :

```
STEP
IF                  i1          ”Jika input 1
       AND          i2          ”dan input 2
```

4. Bagian Pelaksana

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

Contoh :

```
STEP
```

```
IF                  i6          “(Jika input 6 memberikan sinyal
THEN                o1          “maka nyalakan output 1)
IF                  i6          “(Jika input 6 memberikan sinyal
        AND         i2          “dan input 2 memberikan sinyal
THEN    RESET       o5          “Jika ya, matikan output 5,
        SET         o4          “nyalakan output 4).
```

2.6. Intruksi

Terdapat beberapa intruksi yang dapat mempengaruhi jalannya eksekusi dari sebuah kontrol program.

2.6.1. Intruksi NOP

Intruksi ini biasanya diletakkan pada bagian pelaksana pada akhir dari sebuah STEP. Bila diletakkan pada bagian kondisi maka NOP akan selalu bernilai benar. Jika NOP digunakan pada bagian pelaksana, maka itu artinya jika kondisi diatas terpenuhi maka STEP tidak melakukan apa-apa dan menuju ke STEP berikutnya. Penggunaan NOP ini biasanya digunakan pada saat program harus menunggu kondisi tertentu lalu pindah ke STEP berikutnya.

Contoh :

```
IF      i1
THEN    NOP
```

Jika kondisi (IF i1 (input)) terpenuhi maka STEP tidak melakukan apa-apa dan loncat ke STEP berikutnya.

2.6.2. Intruksi JUMP

Intruksi JUMP biasanya digunakan untuk merujuk ke STEP tertentu. Penggunaan JUMP biasanya dipakai untuk melompati suatu STEP berikutnya

atau kembali ke STEP sebelumnya. Bahkan pada kondisi tertentu intruksi JUMP dapat merujuk ke STEP-nya sendiri.

Contoh :

```
STEP 1
IF                      i1
THEN      NOP
IF        N             i1
THEN      JUMP TO       1
```

Jika kondisi pertama terpenuhi (IF i1 (input)) maka STEP tidak melakukan apa-apa dan loncat ke STEP berikutnya. Dan jika kondisi pertama tidak terpenuhi maka lakukan pengecekan kondisi kedua. Jika kondisi kedua terpenuhi (IF N i1 (input)) maka loncat ke 1 (Kembali ke STEP 1 (awal STEP)).

2.6.3. Intruksi OTHRW

Intruksi ini dijalankan pada saat IF terakhir yang dapat dijumpai bernilai salah.

Contoh :

```
STEP
IF                      i2
THEN      SET           o1
          OTHRW SET     o2
```

Jika kondisi terpenuhi (IF i2 (input2)) maka SET output 1 (o1), tetapi jika kondisi tak terpenuhi maka SET output 2 (o2).

2.6.4. Intruksi PSE

Digunakan untuk menandai akhir dari sebuah program dan menyebabkan perubahan program (*Program Selection End*). Dapat juga digunakan untuk men-swap *virtual processor* (berpindah). Pada saat kembali ke program yang menjalankan intruksi PSE, program tersebut akan kembali ke

kalimat pertama dari STEP sekarang atau kalimat pertama dalam program jika tidak ada STEP.

Contoh :

```
STEP 10
IF                    i1    “Jika tombol ditekan
THEN     SET          o1    “Maka silinder maju

STEP 11
IF                    i2    “Jika silinder sudah maju
THEN     RESET        o1    “Maka silinder mundur
         PSE                “Kembali ke kalimat pertama STEP 11
```

2.7. *Operand*

Operand adalah pengenal dalam PLC. *Operand* dapat di interogasi atau di manipulasi dengan menggunakan intruksi-intruksi program dan operator. Beberapa *operand* penting yang sering dipakai dalam penulisan program dapat dilihat pada Tabel 2.1.

Tabel 2.1 *Operand-Operand*

Nama	Penulisan	Contoh	AbsoluteOperand	Symbolic Operand
Input	In	I1	I1.1	Sensor
Output	On	O2	O2.2	Motor
Flag	Fn	F0.0.1	F0.0	Flag1
Timer	Tn	T3	T3	Timer3
Counter	Cn	C4	C4	Counter

Pada *statement list, operand* tersebut dapat dituliskan berdasarkan simbol *operand*-nya saja.

Contoh :

```
STEP
IF                    SENSOR
THEN     SET          MOTOR
         RESET        FLAG1
```

2.8. Perangkat Lunak (*Software*)

2.8.1. *Visual Basic*

Visual basic adalah salah satu dari berbagai bahasa pemprograman tingkat tinggi. *Visual basic* merupakan bahasa pemprograman yang sederhana dibandingkan dengan bahasa pemprograman yang lain karena *visual basic* membebaskan pemprogram dari penulisan perintah atau instruksi yang kompleks sehingga langkah pemprograman menjadi jauh lebih sederhana. Bahasa *visual basic* adalah salah satu bahasa yang dapat digunakan untuk berkomunikasi dengan PLC. Untuk melakukan komunikasi dengan PLC maka terdapat beberapa format penulisan yang dapat digunakan untuk melakukan pengecekan ataupun pengubahan status dari *operand* pada PLC. Beberapa *operand* pada PLC dapat dilakukan pengecekan dan perubahan terhadap statusnya, operand tersebut meliputi input, output, *flag*, *timer*, dan *counter* seperti pada Tabel 2.2.

Tabel 2.2 Tabel *Display-Modification* 1 bit operand
(Sumber : Festo, lampiran *display* dan *Modification operand*)

Display	Display	Modification
Input Format	Respond	Respon
Input DE<o>.<m> Output DA <p>.<m> Flag DM<n>.<n> Timer DT <X> Counter DZ<n>	DE<o>.<m>={0/1} DA<p>.<m>={0/1} DM<n>.<n>={0/1} DT<x>={0/1} DZ<n>={0/1}	 MA<p>.<m>={0/1} MM<n>.<n>={0/1} MT<x>={0/1} MZ<n>={0/1}
o={0 to 2 and 10 to 17} m={0 to 7}	p={0 to 1 and 10 to 17} n={0 to15}	X={0 to 31} Y={0 to 15}

Contoh bentuk penulisan status bit pada *visual basic* dapat dituliskan seperti contoh dibawah ini :

```
MSComm1.Output = "Dm7.7" + Chr$(13)
```

Contoh diatas *visual basic* melakukan pengecekan terhadap *flag* yang memiliki *absolute operand* F7.7 apakah bernilai “1”. Bentuk penulisan respon dari permintaan tersebut dapat ditulis dengan cara :

data = MSComm1.Input

Hasil respon untuk kemudian disimpan di variable data.

2.8.2. *Macromedia Flash*

Macromedia Flash adalah salah satu perangkat lunak (*software*) yang biasa dipergunakan dalam dunia disain dan grafis. Pada *software flash* ini dapat dilakukan proses pembuatan animasi yang interaktif. Dimana animasi yang dibuat dapat disisipkan perintah-perintah untuk mengontrol animasi tersebut. Pengontrolan tersebut dapat dilakukan karena pada *software* ini terdapat telah terdapat *actionscript* yang telah terintegrasi.

Untuk dapat menjalankan animasi *flash* pada *visual basic*, maka yang pertama dilakukan adalah menambahkan komponen pada *visual basic*. Komponen tersebut adalah *shockwaveflash* (flash.ocx). Komponen *shockwaveflash* tersebut kemudian dihubungkan dengan alamat dari link file *flash* yang akan ditampilkan berada. Misalkan alamat dari file flash yaitu G:\Flash Games\The Warriors di mana pada folder tersebut terdapat nama flash movie yang akan dijalankan serta nama formatnya yaitu swf, maka penulisannya pada *visual basic* menjadi G:\Flash Games\The Warrior\The Hero.swf. Untuk memindahkannya Anda tinggal meng-copy alamat file tersebut dan memunculkan dalam properties pada komponen *shockwaveFlash* (*www.ilmukomputer.org, 2008*).

Bentuk pemanggilan animasi flash dalam visual basic dapat dituliskan sebagai berikut : ShockwaveFlash1.Play