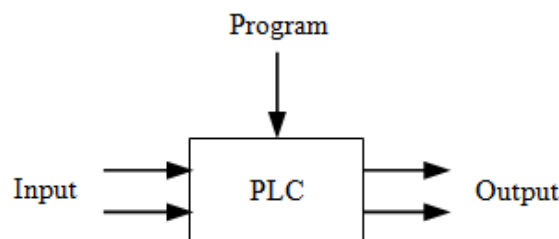


BAB II TINJAUAN PUSTAKA

2.1 PLC (*Programmable Logic Controller*)

Programmable Logic Controller atau PLC pada awalnya dikenal sebagai *Programmable Controller* (PLC) yang lahir sebagai produk yang kompak, dapat diprogram dan direprogram seperti komputer, tidak memakan tempat dan energi yang besar, berbasis teknologi digital, yang dapat menggantikan rangkaian *relay* dan *hardwire*. *Programmable logic controller* singkatnya PLC merupakan suatu bentuk khusus pengontrolan berbasis mikroprosesor yang memanfaatkan memori yang dapat diprogram untuk menyimpan intruksi-intruksi dan untuk mengimplementasikan fungsi – fungsi semisal logika, *sequencing*, pewaktuan (*timing*), pencacahan (*counting*) dan aritmetika guna mengontrol mesin-mesin dan proses-proses seperti (gambar 2.1) dan dirancang untuk dioperasikan oleh para insinyur yang memiliki pengetahuan mengenai bahasa pemrograman. Piranti ini dirancang sedemikian rupa agar tidak hanya programer komputer saja yang dapat membuat dan mengubah program- programnya. Oleh karena itu, para perancang PLC telah menempatkan program awal di dalam piranti ini (pre-program) yang memungkinkan program-program kendali dimasukkan dengan menggunakan suatu bentuk bahasa pemrograman yang sederhana dan intuitif.



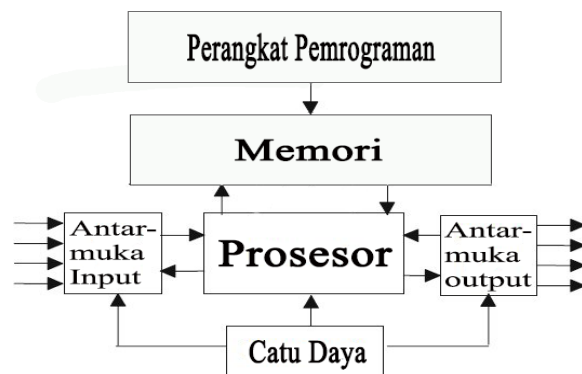
Gambar 2.1 Logika PLC (*Programmable Logic Controller*)
(Erlangga, 2003)

Istilah logika (*logic*) dipergunakan karena pemrograman yang harus dilakukan sebagian besar berkaitan dengan pengimplementasian operasi- operasi

logika dan penyambungan saklar. Perangkat-perangkat *input*, yaitu sensor-sensor semisal saklar dan perangkat-perangkat *Output* di dalam sistem yang di kendali, misalnya motor, katup, dll yang disambungkan ke PLC. Sang operator kemudian memasukan serangkaian intruksi, yaitu sebuah program ke dalam *memory* PLC. Perangkat pengontrol tersebut kemudian memantau input-input dan output-output sesuai dengan intruksi-intruksi di dalam program dan melaksanakan aturan-aturan kendali yang telah diprogramkan.

PLC serupa dengan komputer namun, bedanya komputer dioptimalkan untuk tugas-tugas perhitungan dan penyajian data, sedangkan PLC dioptimalkan untuk tugas-tugas pengontrolan dan pengoperasian di dalam lingkungan industri. Dengan demikian PLC memiliki karakteristik :

1. Kokoh dan dirancang untuk tahan terhadap getaran, suhu, kelembaban, dan kebisingan.
2. Antarmuka input dan output telah tersedia secara *built-in* didalamnya.
3. Mudah diprogram dan menggunakan sebuah bahasa pemrograman yang mudah dipahami, yang sebagian besar berkaitan dengan operasi- operasi logika dan penyambungan.



Gambar 2.2 Sistem PLC (*Programmable Logic Controller*)
(Erlangga, 2003)



2.1.1 Hardware

Umumnya, sebuah sistem PLC memiliki lima komponen dasar. Komponen-komponen ini adalah unit *processor*, memori, unit catu daya, bagian antarmuka *input/output*, dan perangkat pemrograman.

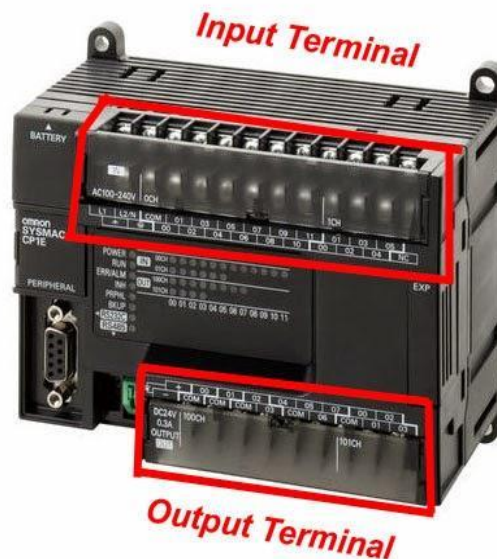
1. *Unit processor* atau *central processing unit* (unit pengolahan pusat) (*CPU*) adalah unit yang berisi mikroprocessor yang menginterpretasikan sinyal-sinyal input dan melaksanakan tindakan-tindakan pengontrolan, sesuai dengan program yang tersimpan di dalam memori, lalu mengkomunikasikan keputusan-keputusan yang diambilnya sebagai sinyal kontrol ke antarmuka output.
2. *Unit catu daya* diperlukan untuk mengkonversikan tegangan AC sumber menjadi tegangan rendah DC (5V) yang dibutuhkan oleh *processor* dan rangkaian-rangkaian di dalam modul-modul antarmuka input dan output.
3. *Perangkat pemrograman* dipergunakan untuk memasukkan program yang dibutuhkan ke dalam memori. Program tersebut dibuat dengan menggunakan perangkat ini dan kemudian dipindahkan ke dalam unit memori PLC.
4. *Unit memori* adalah tempat dimana program yang digunakan untuk melaksanakan tindakan-tindakan pengontrolan oleh mikroprocessor disimpan.
5. *Bagian input dan output* adalah antarmuka di mana prosesor menerima informasi dari dan mengkomunikasikan informasi kontrol ke perangkat-perangkat eksternal. Sinyal-sinyal input, oleh karenanya, dapat berasal dari saklar-saklar.

Tahap dasar untuk penyiapan awal untuk memudahkan dan memasukkan program dalam PLC dengan mempersiapkan daftar seluruh peralatan *input* dan *output* beserta lokasi I/O bit, penempatan lokasi *word* dalam penulisan data. Untuk pemrograman sebuah PLC dahulu kita harus mengenal atau mengetahui tentang organisasi dan memorinya. Ilustrasi dari organisasi memori adalah sebagai peta memori (memori *map*), yang spacenya terdiri dari kategori *User Programmable* dan Data Tabel. *User Program* adalah dimana program *Logic Ladder* dimasukkan dan disimpan yang berupa instruksi – instruksi dalam format

Logic Ladder. Setiap instruksi memerlukan satu word didalam memori.

2.1.2 PLC Omron CP1E-E20 SDR-A

Merupakan Jenis dari PLC Omron seri CP1E, sedangkan arti dari E20 merupakan jumlah dari output dan input yang terdapat pada PLC. PLC jenis ini dapat di implementasikan pada penggerak mekanisme alat industri, alat rumah tangga, dan tugas teknik lainnya, yang mana bersifat logika elektronika.



Gambar 2.3 PLC Omron CP1E-E20 SDR-A
(User's Manual Omron, 2017)

PLC Omron seri CP1E memiliki I/O sebanyak 20 yang dimana 12 input bisa diubah menjadi *analog*, yaitu bekerja dengan tegangan 5 sampai 24 volt dan memiliki *output* sebanyak 8 yang dimana masing-masing *output* tersebut juga memiliki *internal relay* yang bekerja dengan arus hingga 10 A. PLC Omron seri CP1E bekerja dengan tegangan yang bisa diubah 100 sampai 240 VAC, Program *memory*: 2Ksteps (EEPROM), Data *memory* DM: 2Kwords. Dan memiliki minimal tegangan kerja 5 VDC dan maksimum tegangan kerja 24 VDC pada input PLC. Kemudian pada masing-masing output PLC memiliki internal relay yang memiliki maksimum arus kerja sebesar 10 A. PLC Omron seri CP1E



memiliki sistem program dengan menggunakan *software* pemrograman CX-Programmer.

2.1.3 *Software CX-Programmer*

CX-Programmer merupakan *software* khusus untuk memprogram PLC buatan OMRON. CX Programmer ini sendiri merupakan salah satu *software* bagian dari CX-One. Dengan CX-Programmer ini kita bisa memprogram aneka PLC buatan omron dan salah satu fitur yang saya suka yaitu adanya fitur simulasi tanpa harus terhubung dengan PLC, sehingga kita bisa mensimulasikan *ladder* yang kita buat, dan simulasi ini juga bisa kita hubungkan dengan HMI PLC Omron yang telah kita buat dengan menggunakan CX-Designer (bagian dari CX-One).

Software ini beroperasi di bawah sistem operasi *Windows*, oleh sebab itu pemakai *software* ini diharapkan sudah familier dengan sistem operasi *Windows* antara lain untuk menjalankan *software* program aplikasi, membuat *file*, menyimpan *file*, mencetak *file*, menutup *file*, membuka *file*, dan keluar dari (menutup) *software* program. Ada beberapa persyaratan minimum yang harus dipenuhi untuk bisa mengoperasikan CX Programmer secara optimal yaitu:

Komputer IBM PC/AT

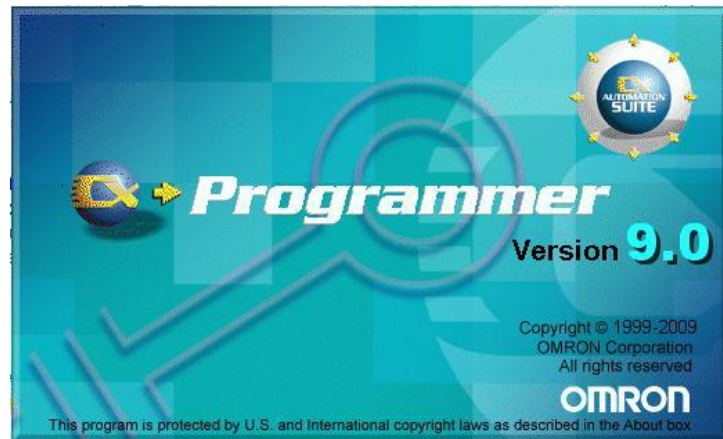
kompatibel CPU Pentium I

minimal 133 MHz RAM

32 Mega bytes

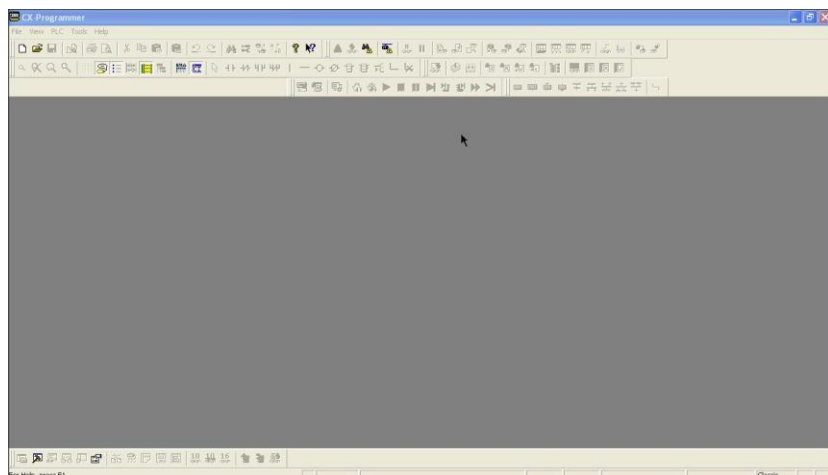
Hard disk dengan ruang kosong kurang lebih 100 MB

Monitor SVGA dengan resolusi 800 x 600



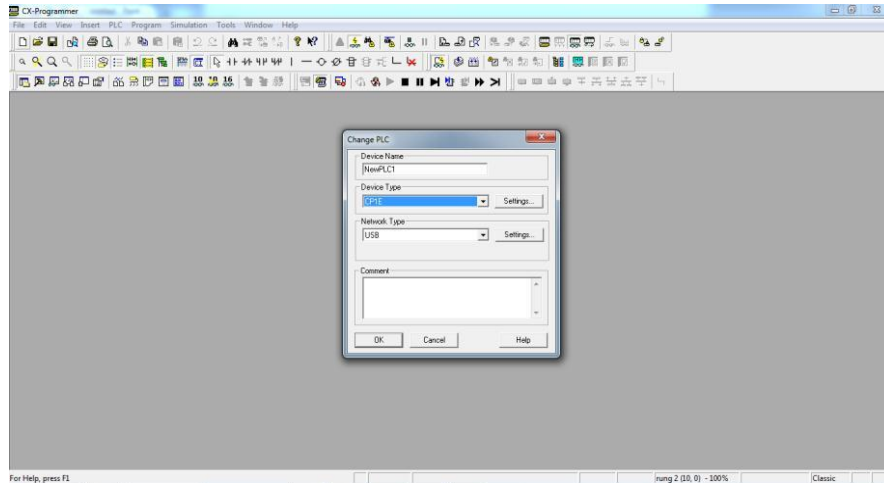
Gambar 2.4 CX-Programmer Version 9.0 Omron (PLCDROID, 2018)

Berikut tampilan dari CX-Programmer saat pertama kali dibuka :



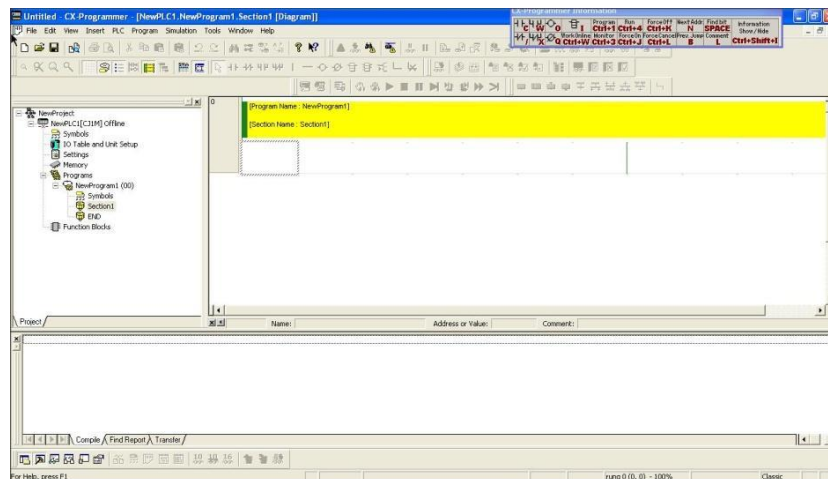
Gambar 2.5 Tampilan Pertama Program CX-Programmer Version 9.0 Omron (PLCDROID, 2018)

Untuk memulai menggunakan CX-Programmer ini yaitu pada menu pilih *file* -> *new* atau bisa langsung pada *toolbar* klik gambar kertas putih untuk memulai membuat *project* baru, kalo untuk membuka *file project* yang sudah dibuat sebelumnya yaitu pilih *file* -> *open* atau pada *toolbar* pilih gambar disamping kertas putih maka akan muncul tampilan berikut :



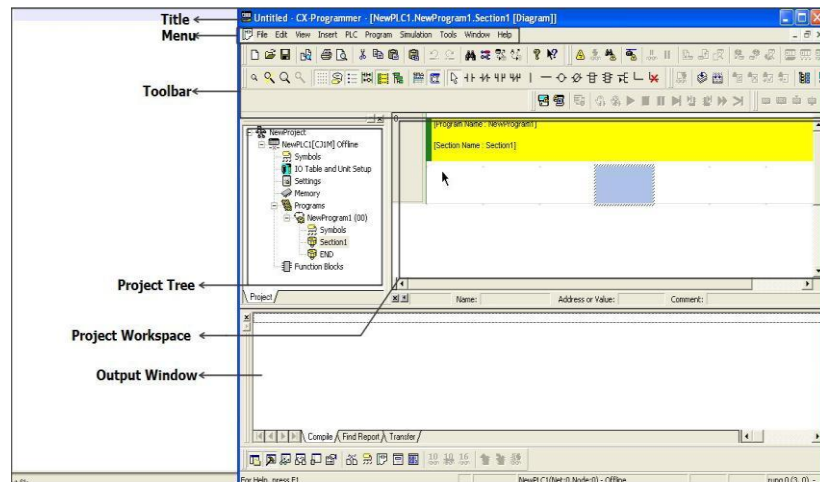
Gambar 2.6 Tampilan Pemilihan Device PLC Pada Program CX Programmer Version 9.0 Omron (PLCDROID,2018)

Setelah memilih tipe PLC yang akan digunakan, misalnya PLC CP1E dan *Network type* yang akan digunakan yaitu USB untuk setting lebih dalam bisa diklik *setting*, kemudian klik OK maka akan tampil tampilan berikut :



Gambar 2.7 Tampilan Project Program CX-Programmer Version 9.0 Omron (PLCDROID, 2018)

Keterangan detail untuk tampilan tersebut yaitu :



Gambar 2.8 Tampilan Keterangan Project Program CX-Programmer Version 9.0 Omron (PLCDROID,2018)

Title Bar :

Menunjukkan nama file atau data tersimpan dan dibuat pada CX- Programmer

Menus :

Pilihan Untuk memilih Menu

Toolbar :

Pilihan untuk memilih fungsi dengan menekan tombol.

Select[view] Toolbar, Kemudian dapat memilih toolbar yang ingin ditampilkan.

Section :

Dapat membagi program kedalam beberapa blok. Masing-masing blok dapat dibuat atau ditampilkan.

Project WorkSpace Project Tree :

Mengatur program dan data. Dapat membuat duplikat dari setiap elemen dengan melakukan Drag dan Drop diantara proyek yang berbeda atau melalui suatu proyek.

Ladder Window :

Layar sebagai tampilan atau membuat diagram tangga.



Output Window :

Menunjukkan informasi error saat melakukan compile (error check).
Menunjukkan hasil dari pencarian kontak / koil didalam list form.
Menunjukkan detail dari error yang ada pada saat loading suatu proyek.

Status Bar :

Menunjukkan suatu informasi seperti nama PLC, status on line/offline, lokasi dari cell yang sedang aktif.

Information Window :

Menampilkan window yang menunjukkan shortcut key yang digunakan pada CX – Programmer.

Symbol Bar :

Menampilkan nama, alamat atau nilai dan comment dari simbol yang sedang dipilih cursor

2.1.4 Program PLC

Suatu *software* yang berfungsi sebagai pengontrol otomatis yang berupa *softcontact* yang diimplementasikan kedalam suatu bentuk bilangan logika. Sehingga dapat mengatur sistem suatu alat industri elektronika dan mekanik.

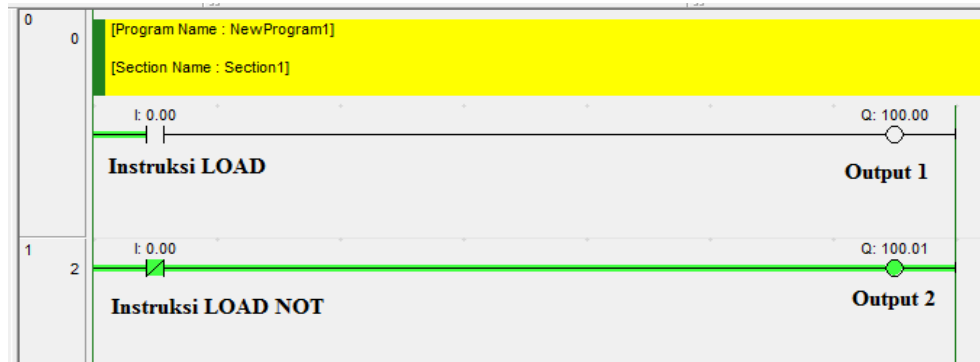
Ada 2 sistem pemrograman pada PLC Omron CP1E-E20SDR-A :

1. *Function Block Diagram* : Jenis Teknik Pemrograman *Logic* yang tersusun dari *block-block diagram* dalam fungsi blok diagram khusus.
2. *Ladder Diagram* : Jenis Teknik Pemrograman *Logic* yang disusun dalam satuan-satuan kontak untuk menghasilkan fungsi tertentu dalam menghasilkan logika yang terdiri dari kontak NC, NO, *Timer*, *Counter*, dan lain-lain.

2.1.5 Instruksi Dasar Pada PLC

1. *LOAD* (LD) dan *LOAD NOT* (LD NOT)

Kondisi pertama yang mengawali sembarang blok logika di dalam diagram tangga berkaitan dengan instruksi *LOAD* (LD) atau *LOAD NOT* (LD NOT). Contoh instruksi ini ditunjukkan pada gambar 2.9



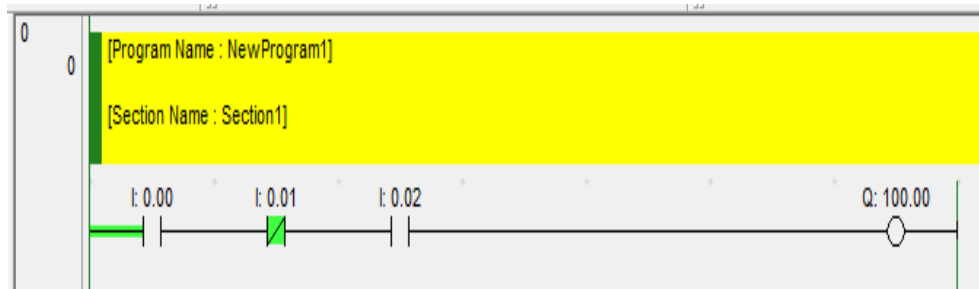
Gambar 2.9 Contoh Penggunaan Instruksi LD dan LD NOT (PLCDROID, 2018)

Tabel 2.1 Kode Mnemonik Instruksi LD dan LD NOT

Alamat	Instruksi	Operan
00000	LD	0.00
00001	OUT	100.00
00002	LD NOT	0.00
00003	OUT	100.01

2. AND dan AND NOT

Jika terdapat dua atau lebih kondisi yang dihubungkan seri pada garis instruksi yang sama maka kondisi pertama menggunakan instruksi LD atau LD NOT, dan sisanya menggunakan instruksi AND atau AND NOT. Gambar menunjukkan suatu penggalan diagram tangga yang mengandung tiga kondisi yang dihubungkan secara seri pada garis instruksi yang sama dan berkaitan dengan instruksi LD, AND NOT, dan AND. Masing-masing instruksi tersebut membutuhkan satu baris kode mnemonic.



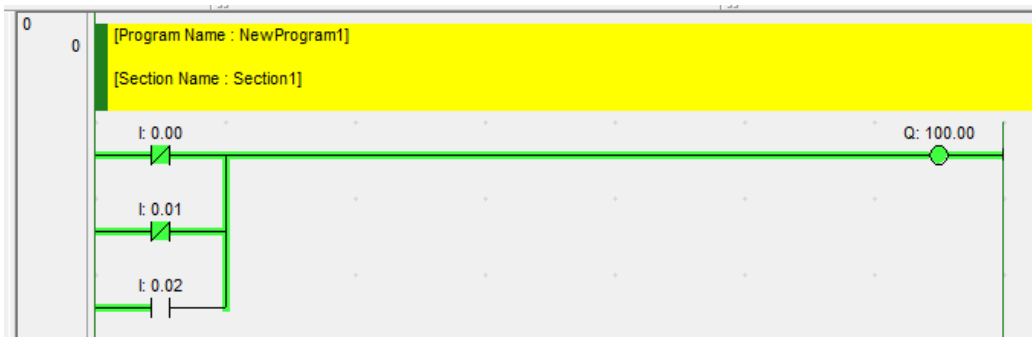
Gambar 2.10 Contoh Penggunaan Instruksi AND dan AND NOT (PLCDROID, 2018)

Tabel 2.2 Kode Mnemonik Instruksi AND dan AND NOT

Alamat	Instruksi	Operan
00000	LD	0.00
00001	AND NOT	0.01
00002	AND	0.02
00003	OUT	100.00

3. OR dan OR NOT

Jika dua atau lebih kondisi yang dihubungkan paralel, artinya dalam garis instruksi yang berbeda kemudian bergabung lagi dalam satu garis instruksi yang sama maka kondisi pertama terkait dengan instruksi LD dan LD NOT dan sisanya berkaitan dengan instruksi OR dan OR NOT. Gambar menunjukkan tiga buah instruksi yang berkaitan dengan instruksi LD NOT, OR NOT, dan OR. Masing-masing instruksi tersebut membutuhkan satu baris kode mnemonik.



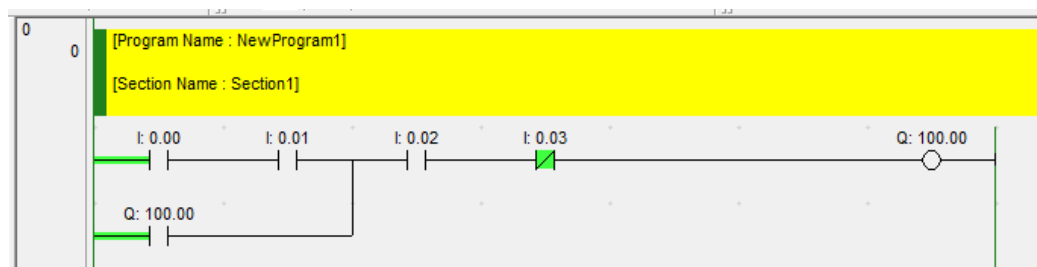
Gambar 2.11 Contoh Penggunaan Instruksi OR dan OR NOT (PLCDROID, 2018)

Tabel 2.3 Kode Mnemonik Instruksi OR dan OR NOT

Alamat	Instruksi	Operan
00000	LD NOT	0.00
00001	OR NOT	0.01
00002	OR	0.02
00003	OUT	100.00

4. Kombinasi instruksi AND dan OR

Jika instruksi AND dan OR digabung atau dikombinasikan dalam suatu rangkaian tangga yang kompleks maka bisa dipandang satu persatu, artinya bisa dilihat masing-masing hasil gabungan dua kondisi menggunakan instruksi AND atau OR secara sendiri-sendiri kemudian menggabungkannya menjadi satu kondisi menggunakan instruksi AND atau OR yang terakhir. Gambar 2.12 menunjukkan contoh diagram tangga yang mengimplentasikan cara seperti tersebut di atas.



Gambar 2.12 Contoh Penggabungan Instruksi AND dan OR (PLCDROID, 2018)

Tabel 2.4 Kode Mnemonik Instruksi AND dan OR

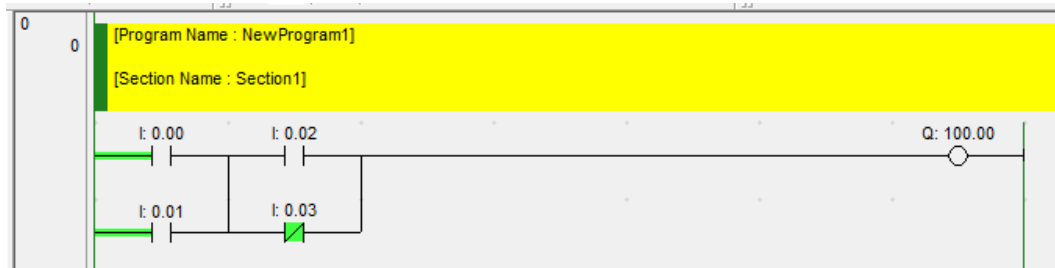
Alamat	Instruksi	Operan
00000	LD	0.00
00001	AND	0.01
00002	OR	100.00
00003	AND	0.02
00004	AND NOT	0.03
00005	OUT	100.00

5. Instruksi-instruksi Blok Logika

Instruksi-instruksi blok logika tidak berhubungan dengan suatu kondisi tertentu pada diagram tangga, melainkan untuk menyatakan hubungan antar blok-blok logika, misalnya instruksi AND LD akan meng-AND-logik-kan kondisi eksekusi yang dihasilkan oleh dua blok logika, demikian juga dengan OR LD untuk meng-OR logikkan kondisi eksekusi yang dihasilkan dua blok logika.

a. AND LOAD (AND LD)

Gambar 2.13 menunjukkan contoh penggunaan blok logika AND LD yang terdiri atas dua blok logika, yang akan menghasilkan kondisi ON jika blok logika kiri dalam kondisi ON (salah satu dari 0.00 atau 0.01 yang ON) dan blok logika kanan juga dalam keadaan ON (0.02 dalam kondisi ON atau 0.03 dalam kondisi OFF).



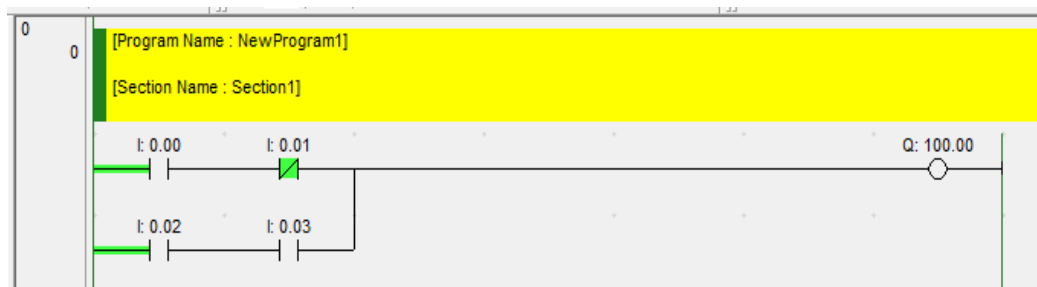
Gambar 2.13 Contoh Penggunaan Instruksi Blok Logika AND LD (PLCDROID, 2018)

Tabel 2.5 Kode Mnemonik Instruksi Blok Logika AND LD

Alamat	Instruksi	Operan
00000	LD	0.00
00001	OR	0.01
00002	LD	0.02
00003	OR NOT	0.03
00004	AND LD	-
00005	OUT	100.00

b. OR LOAD (OR LD)

Instruksi ini digunakan untuk meng-OR-logik-kan dua blok logika. Gambar 2.14 menunjukkan contoh penggunaan blok logika OR LD yang terdiri atas dua blok logika. Kondisi eksekusi ON akan dihasilkan jika blok logika atas atau blok logika bawah dalam kondisi ON. Artinya, 0.00 dan kondisi ON dan 0.01 dalam kondisi OFF atau 0.02 dan 0.03 dalam kondisi ON).



Gambar 2.14 Contoh Penggunaan Instruksi Blok Logika OR LD (PLCDROID, 2018)

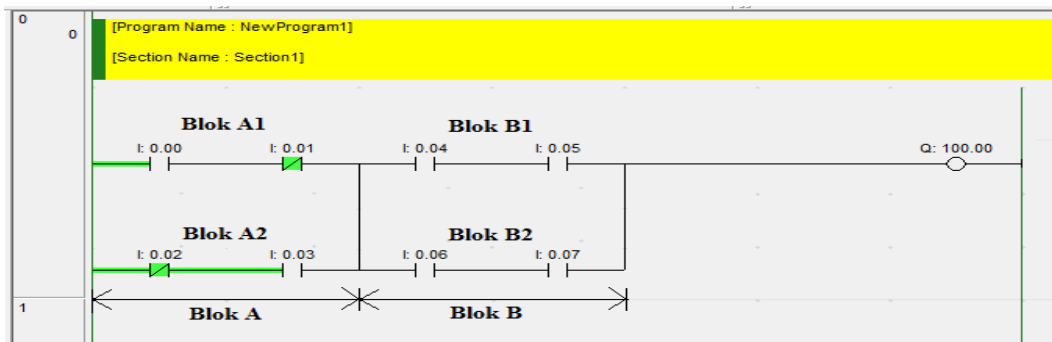
Tabel 2.6 Kode Mnemonik Instruksi Blok Logika OR LD

Alamat	Instruksi	Operan
00000	LD	0.00
00001	AND NOT	0.01
00002	LD	0.02
00003	AND	0.03
00004	OR LD	-
00005	OUT	100.00

c. Logika Kompleks

Untuk membuat kode mnemonik diagram tangga yang kompleks, caranya dengan cara membagi membagi diagram tersebut ke dalam blok- blok logika yang besar, kemudian membagi lagi blok yang besar tersebut menjadi blok- blok logika yang lebih kecil, demikian seterusnya hingga tidak perlu lagi dibuat blok yang lebih kecil lagi. Blok-blok ini kemudian masing-masing dikodekan, mulai dari yang kecil, dan digabungkan satu per satu hingga membentuk diagram tangga yang asli. Instruksi blok logika AND LD dan OR LD hanya digunakan untuk

menggabungkan dua blok logika saja (blok logika yang digabungkan berupa hasil penggabungan sebelumnya, atau hanya sebuah kondisi tunggal). Gambar 2.15 memperlihatkan suatu contoh diagram tangga yang kompleks, yang dapat dibagi dua blok besar (blok A dan B). Blok A dapat dibagi lagi menjadi dua blok yang lebih kecil (blok A1 dan A2), dan blok B dibagi menjadi dua blok yang lebih kecil, yaitu blok B1 dan B2. Kemudian blok- blok logika yang kecil ini ditulis terlebih dahulu, diawali dengan menuliskan blok A1 (alamat 00000 dan 00001) dan blok A2 (alamat 00002 dan 00003), kemudian digabung menggunakan instruksi blok logik OR LD (alamat 00004). Selanjutnya blok B1 dituliskan (alamat 00005 dan 00006) dilanjutkan dengan blok B2 (alamat 00007 dan 00008) dan digabung dengan instruksi blok logik OR LD (alamat 00009). Hasilnya berupa blok A dan blok B yang kemudian juga digabung menggunakan blok logika AND LD (alamat 00010).



Gambar 2.15 Contoh Penggunaan Instruksi Blok Logika Kompleks (PLCDROID, 2018)



Tabel 2.7 Kode Mnemonik Instruksi Blok Logika Kompleks

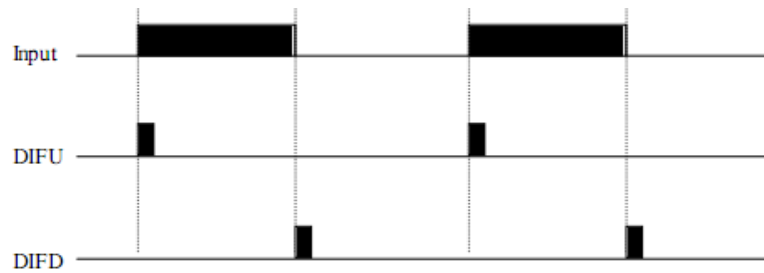
Alamat	Instruksi	Operan
00000	LD	0.00
00001	AND NOT	0.01
00002	LD NOT	0.02
00003	AND	0.03
00004	OR LD	-
00005	LD	0.04
00006	AND	0.05
00007	LD	0.06
00008	AND	0.07
00009	OR LD	-
00010	AND LD	-
00011	OUT	100.00

6. Instruksi Kendali Bit

Terdapat instruksi dasar yang dapat digunakan untuk mengontrol status bit secara *individual*, yaitu *DIFFERENTIATE UP* (DIFU), *DIFFERENTIATE DOWN* (DIFD) instruksi ini dituliskan di sisi paling kanan diagram tangga dan membutuhkan sebuah alamat bit sebagai operan. Selain instruksi-instruksi ini digunakan untuk membuat bit-bit keluaran ON atau OFF dalam area IR (ke piranti eksternal), ini juga digunakan untuk mengontrol bit-bit lainnya.

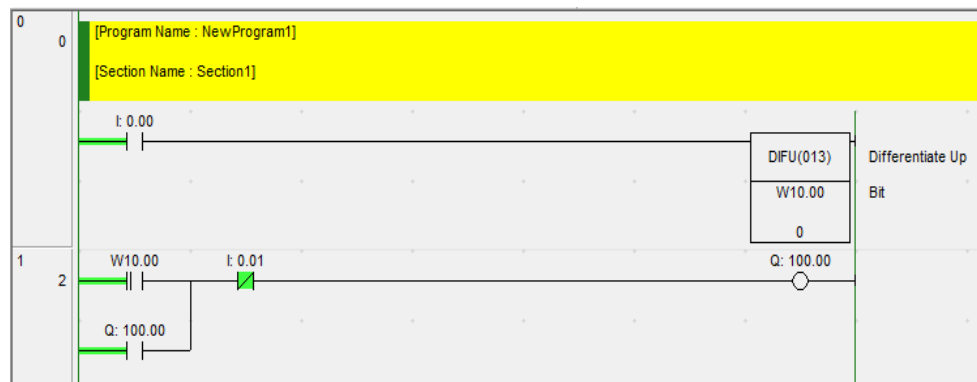
Kedua instruksi ini sangat sering sekali digunakan dalam pemrograman PLC. Kedua instruksi ini masuk ke dalam jenis *ladder instructions*, pada sub kategori *bit control instructions*.

Untuk penjelasan mengenai instruksi DIFU dan DIFD lihat gambar berikut:



Gambar 2.16 Prinsip Kerja Instruksi Kendali Bit DIFU dan DIFD (PLCDROID, 2018)

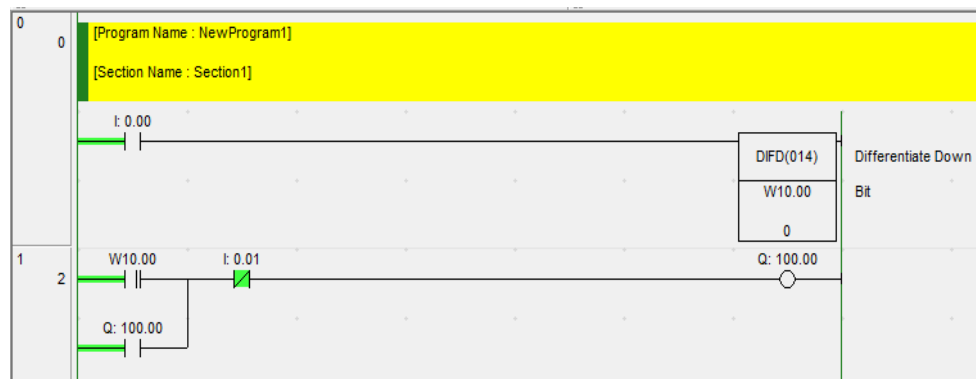
Jadi seperti terlihat pada gambar di atas, baik instruksi DIFU maupun instruksi DIFD output ON nya (warna hitam pada gambar) hanya sekali dan dalam waktu yang singkat saja, atau biasa disebut *one scan only*. sedangkan perbedaan dari instruksi DIFU dan DIFD, bahwa instruksi DIFU ini akan ON (tentunya dalam waktu singkat saja) saat input baru saja mengalami perubahan dari OFF ke ON. Sedangkan pada instruksi DIFD, akan ON (dalam waktu singkat saja) saat input baru saja mengalami perubahan dari ON ke OFF.



Gambar 2.17 Contoh Penggunaan Instruksi Kendali Bit *DIFFERENTIATE UP* (DIFU) (PLCDROID, 2018)

Tabel 2.8 Kode Mnemonik Instruksi Kendali Bit *DIFFERENTIATE UP* (DIFU)

Alamat	Instruksi	Operan
00000	LD	0.00
00001	DIFU	W10.00
00002	LD	W10.00
00003	OR	100.00
00004	AND NOT	0.01
00005	OUT	100.00



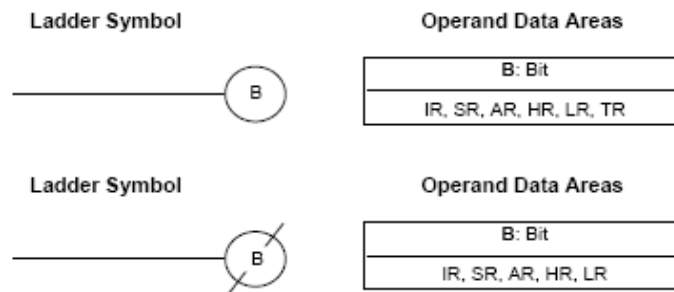
Gambar 2.18 Contoh Penggunaan Instruksi Kendali Bit *DIFFERENTIATE DOWN* (DIFD) (PLCDROID, 2018)

Tabel 2.9 Kode Mnemonik Instruksi Kendali Bit *DIFFERENTIATE DOWN* (DIFD)

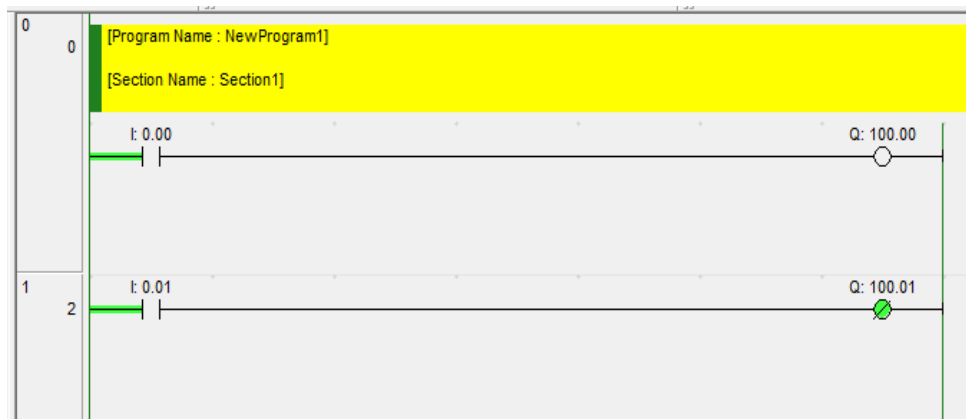
Alamat	Instruksi	Operan
00000	LD	0.00
00001	DIFD	W10.00
00002	LD	W10.00
00003	OR	100.00
00004	AND NOT	0.01
00005	OUT	100.00

7. Instruksi *OUTPUT* (OUT) dan *OUTPUT NOT* (OUT NOT)

Instruksi ini digunakan untuk mengontrol operan yang berkaitan dengan kondisieksekusi (apakah ON atau OFF). Dengan menggunakan instruksi OUT, maka bit operanakan menjadi ON jika kondisi eksekusinya juga ON, sedangkan OUT NOT akanmenyebabkan bit operan menjadi ON jika kondisi eksekusinya OFF. Gambar 2.19 memperlihatkan simbol tangga dan area data operan dari instruksi OUT dan OUT NOT, sedangkan Gambar 2.20 memperlihatkan contoh implementasi kedua instruksi tersebut.



Gambar 2.19 Simbol Tangga Dan Area Data Operan Instruksi OUT dan OUT NOT (PLCDROID, 2018)



Gambar 2.20 Contoh Penggunaan Instruksi OUT dan OUT NOT (PLCDROID, 2018)

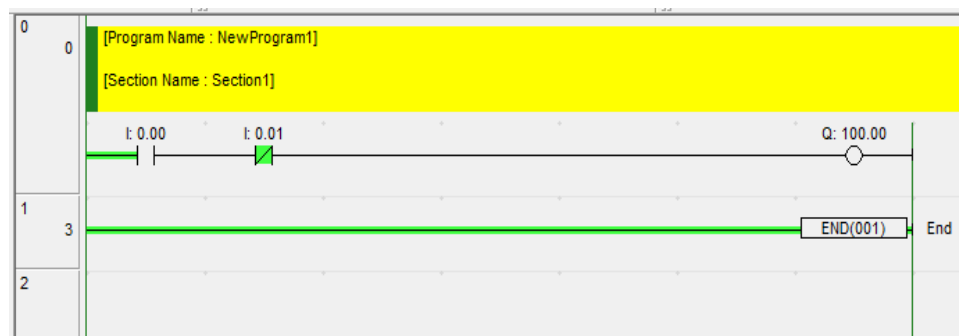
Tabel 2.10 Kode Mnemonik Instruksi OUT dan OUT NOT

Alamat	Instruksi	Operan
00000	LD	0.00
00001	OUT	100.00
00002	LD	0.01
00003	OUT NOT	100.01

8. Instruksi END

Instruksi END merupakan instruksi terakhir yang harus dituliskan atau digambarkan dalam diagram tangga. CPU pada PLC akan mengerjakan semua instruksi dalam program dari awal (baris pertama) sampai ditemui instruksi END yang pertama, sebelum kembali lagi mengerjakan instruksi dalam program dari awal (artinya instruksi-instruksi yang ada di bawah instruksi END akan diabaikan). Instruksi END tidak memerlukan operan dan tidak boleh diawali dengan suatu kondisi seperti pada instruksi lainnya.

Suatu diagram tangga atau program PLC harus diakhiri dengan instruksi END, jika tidak maka program tidak dijalankan sama sekali. Angka yang dituliskan pada instruksi END pada kode mnemonik merupakan kode fungsinya. Gambar 2.21 memperlihatkan contoh penggunaan instruksi END.

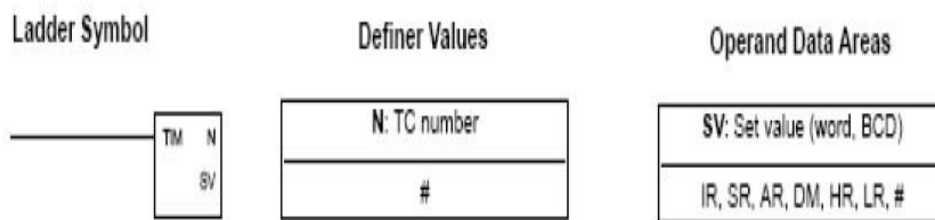
**Gambar 2.21** Contoh Penggunaan Instruksi END (PLCDROID, 2018)

Tabel 2.11 Kode Mnemonik Instruksi END

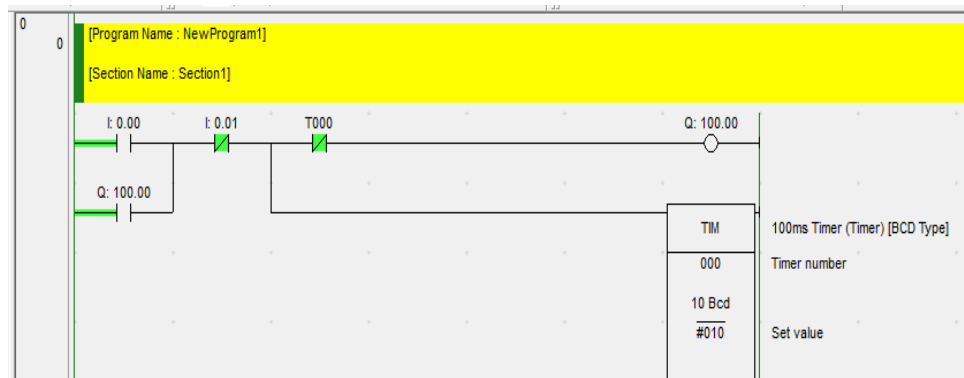
Alamat	Instruksi	Operan
00000	LD	0.00
00001	AND NOT	0.01
00002	OUT	100.00
00003	END(001)	-

9. Instruksi *TIMER* (TIM)

Instruksi TIM dapat digunakan sebagai *timer* (pewaktu) ON-delay pada rangkaian relai. Gambar 2.22 memperlihatkan simbol tangga dan area data operan dari instruksi TIM. Instruksi TIM membutuhkan angka timer (N), dan nilai set (SV) antara 0000 sampai 9999 (artinya 000,0 sampai 999,9 detik).



Gambar 2.22 Simbol Tangga Dan Area Data Operan Dari Instruksi *TIMER* (TIM) (PLCDROID, 2018)



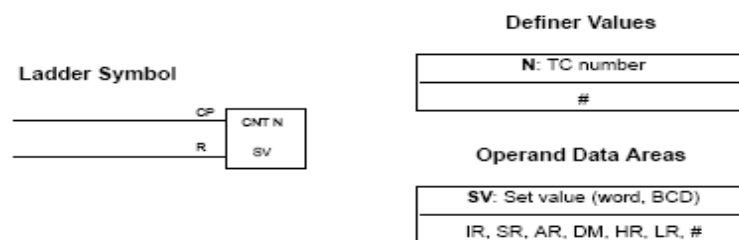
Gambar 2.23 Contoh Penggunaan Instruksi *TIMER* (TIM) (PLCDROID, 2018)

Tabel 2.12 Kode Mnemonik Instruksi *TIMER* (TIM)

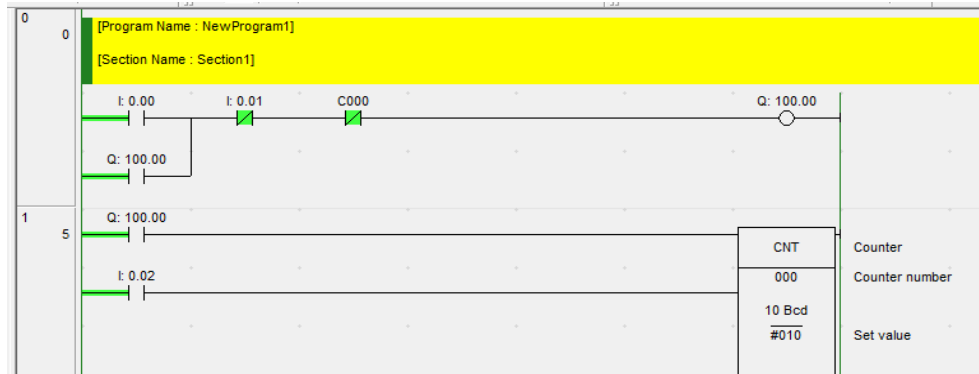
Alamat	Instruksi	Operan
00000	LD	0.00
00001	OR	100.00
00002	AND NOT	0.01
00003	TIM	000 #010
00004	AND NOT	T000
00005	OUT	100.00

10. Instruksi *COUNTER* (CNT)

CNT yang digunakan di sini adalah *counter* penurunan yang diset awal. Penurunan satu hitungan setiap kali saat sebuah sinyal berubah dari OFF ke ON. *Counter* harus diprogram dengan input hitung, input reset, angka *counter*, dan nilai set (SV) Nilai set ini adalah 0000 sampai 9999. Gambar 2.24 memperlihatkan simbol tangga dan area data operan dari instruksi CNT. Dan Gambar 2.25 memperlihatkan contoh penggunaan instruksi END.



Gambar 2.24 Simbol Tangga Dan Area Data Operan Dari Instruksi *COUNTER* (CNT) (PLCDROID, 2018)



Gambar 2.25 Contoh Penggunaan Instruksi *COUNTER*(CNT) (PLCDROID, 2018)

Tabel 2.13 Kode Mnemonik Instruksi *COUNTER*(CNT)

Alamat	Instruksi	Operan
00000	LD	0.00
00001	OR	100.00
00002	AND NOT	0.01
00003	AND NOT	C000
00004	OUT	100.00
00005	LD	100.00
00006	LD	0.02
00006	CNT	000 #010

2.2 Perangkat – Perangkat *Input*

Pada dasarnya PLC memerlukan sebuah input untuk melakukan perintah yang sudah diprogram dan di download ke PLC. Umumnya PLC hanya memerlukan input oleh sebuah saklar mekanis dan sensor. Sebuah saklar mekanis menghasilkan sinyal (atau sinyal-sinyal) hidup / mati sebagai akibat dari tertutup atau terbukanya saklar oleh suatu *input* mekanis. Dipasaran tersedia saklar-saklar dengan kontak *normally open* (normal terbuka) (NO) atau *normally closed* (normal tertutup) (NC) atau kontak yang dapat diatur sesuai kebutuhan dengan memilih kontak-kontak yang tepat. Berikut ini adalah beberapa contoh dari *input* PLC.



2.2.1 Sensor Proximity

Sensor *proximity* adalah sensor yang berfungsi untuk mendeteksi ada atau tidaknya suatu objek. Sensor *proximity* dapat mendeteksi keberadaan benda disekitarnya tanpa ada kontak fisik dengan benda tersebut. Cara kerja sensor *proximity* ini yaitu dengan memancarkan medan elektromagnetik dan mencari perubahan bentuk medan elektromagnetik pada saat benda dideteksi. Contoh medan elektromagnetik yang sering digunakan yaitu sinar infra merah. Jika benda telah terdeteksi maka sinyal infra merah tersebut akan merubah bentuk sinyal dan mengirimkan sinyal kembali ke sensor dan memberitahukan bahwa didepan sensor terdapat benda. Ada pun jenis-jenis *proximity* sensor yaitu :

1. *Inductive Sensor Proximity* (Sensor Jarak Induktif)

Sensor Jarak Induktif atau *Inductive Proximity Sensor* adalah Sensor Jarak yang digunakan untuk mendeteksi keberadaan logam baik logam jenis Ferrous maupun logam jenis non-ferrous. Sensor ini dapat digunakan untuk mendeteksi keberadaan (ada atau tidak adanya objek logam), menghitung objek logam dan aplikasi pemosisian. Sensor induktif sering digunakan sebagai pengganti saklar mekanis karena kemampuannya yang dapat beroperasi pada kecepatan yang lebih tinggi dari sakelar mekanis biasa. Sensor Jarak Induktif ini juga lebih andal dan lebih kuat.

Sensor Proximity Induktif pada umumnya terbuat dari kumparan/koil dengan inti ferit sehingga dapat menghasilkan medan elektromagnetik frekuensi tinggi. *Output* dari sensor jarak jenis induktif ini dapat berupa analog maupun digital. Versi *Analog* dapat berupa tegangan (biasanya sekitar 0 – 36VDC) atau arus (0 – 30mA). Jarak pengukurannya bisa mencapai hingga 2 inci. Sedangkan versi Digital biasanya digunakan pada rangkaian DC saja ataupun rangkaian AC/DC. Sebagian besar Sensor Induktif Digital dikonfigurasi dengan *Output* “*NORMALLY – OPEN*” namun ada juga yang dikonfigurasi dengan *Output* “*NORMALLY – CLOSE*”. Sensor Induktif ini

sangat cocok untuk mendeteksi benda-benda logam di mesin dan di peralatan otomatisasi

Inductive Proximity Sensor ini pada dasarnya terdiri dari sebuah osilator, sebuah koil dengan inti ferit, rangkaian detektor, rangkaian *output*, kabel dan konektor. Osilator pada Sensor Jarak ini akan membangkitkan gelombang sinus dengan frekuensi yang tetap. Sinyal ini digunakan untuk menggerakkan kumparan atau koil. Koil dengan Inti Ferit ini akan menginduksi medan elektromagnetik. Ketika garis-garis medan elektromagnetik ini ter-interupsi oleh objek logam, tegangan osilator akan berkurang sebanding dengan ukuran dan jarak objek dari kumparan/koil. Dengan demikian, Sensor Proksimitas ini dapat mendeteksi adanya objek yang sedang mendekatinya. Pengurangan tegangan osilator ini disebabkan oleh arus Eddy yang diinduksi pada logam yang meng-interupsi garis-garis logam.

2. *Capacitive Proximity Sensor*

Sensor Jarak Kapasitif atau *Capacitive Proximity Sensor* adalah Sensor Jarak yang dapat mendeteksi gerakan, komposisi kimia, tingkat dan komposisi cairan maupun tekanan. Sensor Jarak Kapasitif dapat mendeteksi bahan-bahan dielektrik rendah seperti plastik atau kaca dan bahan-bahan dielektrik yang lebih tinggi seperti cairan sehingga memungkinkan sensor jenis ini untuk mendeteksi tingkat banyak bahan melalui kaca, plastik maupun komposisi kontainer lainnya.

Sensor Jarak Kapasitif ini pada dasarnya mirip dengan Sensor Jarak Induktif, perbedaannya adalah sensor kapasitif menghasilkan medan elektrostatik sedangkan sensor induktif menghasilkan medan elektromagnetik. Sensor Jarak Kapasitif ini dapat digerakan oleh bahan konduktif dan bahan non-konduktif. Elemen aktif Sensor Jarak Kapasitif dibentuk oleh dua elektroda logam yang diposisikan untuk membentuk ekuivalen (sama dengan) dengan Kapasitor Terbuka. Elektroda ini ditempatkan di rangkaian osilasi yang berfrekuensi tinggi. Ketika objek



mendekati permukaan sensor jarak kapasitif ini, medan elektrostatik pelat logam akan terinterupsi sehingga mengubah kapasitansi sensor jarak. Perubahan ini akan mengubah kondisi dalam pengoperasian sensor jarak sehingga dapat mendeteksi keberadaan objek tersebut.

3. *Ultrasonic Proximity Sensor*

Ultrasonic yang dipancarkan atau dikeluarkan oleh *transmitter* atau alat pemancar gelombang *ultrasonic*. *Transmitter* mengeluarkan gelombang *ultrasonic* yang dihasilkan dari frekuensi diatas normal dari gelombang suara. Cara kerjanya sebenarnya sangatlah simpel, pada awalnya *transmitter* akan mengeluarkan gelombang *ultrasonic* yang biasanya dikeluarkan secara berkala dalam beberapa detik sekali

Pancaran gelombang *ultrasonic* tersebut akan terus dipancarkan menyeluruh dan meluas dalam jangkauannya. Kemudian ketika pancaran gelombang *ultrasonic* tersebut menabrak sebuah objek tertentu, maka pancaran gelombang *ultrasonic* tersebut akan berhenti dan dengan kemudian berbalik arah menuju alat penerima sinyal *ultrasonic* atau lebih dikenal dengan istilah *receiver* yang terdapat pada sensor jarak. Pada saat itu juga *receiver* akan memberikan data dari hasil tangkapan gelombang *ultrasonic* tadi kepada mikro kontroler yang kemudian oleh mikro kontroler akan diproses menjadi sebuah data mengenai bentuk objek dan jarak dari objek yang tersentuh gelombang *ultrasonic* tadi. Jaraknya gelombang yang dipancarkan oleh *transmitter* tergantung pada alat yang digunakan.

4. *Photoelectric Proximity Sensor* (Sensor Jarak Fotolistrik)

Sensor Jarak Fotolistrik atau *Photoelectric Proximity Sensor* adalah sensor jarak yang menggunakan elemen peka cahaya untuk mendeteksi obyek. Sensor Proximity Fotolistrik terdiri sumber cahaya (atau disebut dengan Emitor) dan Penerima (*Receiver*).

2.2.2 Proximity Sensor LJ12A3-4-Z

Proximity Sensor LJ12A3-4-Z adalah sensor yang menggunakan medan magnet untuk mendeteksi objek yang berada dihadapannya. Objek yang dapat dideteksi adalah yang berjarak 4mm.

Sensor ini memiliki tiga kabel dengan rincian seperti berikut :

1. Kabel coklat perlu dihubungkan ke tegangan 6 - 36 V.
2. Kabel biru dihubungkan ke *ground*.
3. Kabel hitam dihubungkan ke *input* PLC dan akan memberikan nilai *HIGH* atau *LOW*. Apabila didepan sensor tidak terdapat halangan, nilainya berupa *HIGH*. Kalau ada halangan di depan sensor, hasilnya *LOW*.



Gambar 2.26 Proximity Sensor LJ12A3-4-Z/BX

Prinsip kerja Sensor *proximity* LJ12A3-4-Z/BX Bila objek berada di depan sensor dan dapat terjangkau oleh sensor maka *output* rangkaian sensor akan berlogika “1” atau “*high*” yang berarti objek “ada”. Sebaliknya jika objek berada pada posisi yang tidak terjangkau oleh sensor maka *output* rangkaian sensor akan bernilai “0” atau “*low*” yang berarti objek “tidak ada”.

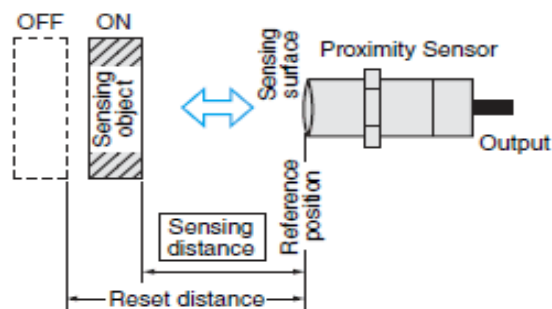
Berikut ini adalah spesifikasi dari *Proximity* Sensor LJ12A3-4-Z.

Spesifikasi Proximity Sensor LJ12A3-4-Z yaitu :

- *Work Input* : 6 - 36 Volt
- *Output Current* : 300 mA (max)
- *Size* : 12 mm (diameter) x 66 mm (lebar)

- Panjang kabel : 100 cm
- Objek yang dideteksi : Metal (tembaga, besi, aluminium, dll)
- Jarak deteksi : 4 mm
- NPN Output

2.2.3 Jarak Deteksi Sensor *Proximity*

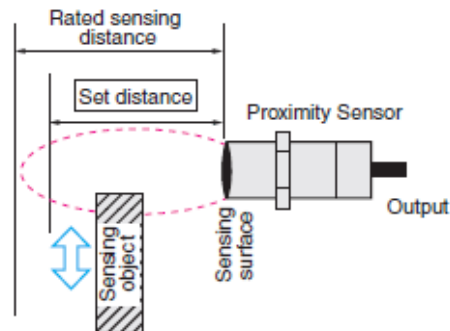


Gambar 2.27 Skematik Jarak Deteksi Sensor *Proximity* (Omron, 2018)

Jarak deteksi atau *sensing distance* adalah jarak dari posisi yang terbaca dan tidak terbaca sensor untuk operasi kerjanya berkisar satu milimeter sampai beberapa sentimeter, ketika objek benda digerakkan oleh metode tertentu.

Reset distance adalah jarak dimana sensor dalam keadaan *reset*, sehingga objek tidak akan terdeteksi dan sensor tidak aktif (OFF). *Sensing distance* adalah jarak toleransi dimana sensor aktif bekerja mendeteksi jikalau ada benda yang melewati permukaan sensor (*sensing surface*). Saat objek melewati permukaan sensor maka objek tersebut (*sensing object*) akan mengaktifkan sensor (ON) sehingga *Proximity* sensor akan menghasilkan output. Baik *reset distance* maupun *sensing distance* dihitung mulai dari *reference position* atau posisi referensi yang sama dengan posisi *sensing surface*.

2.2.4 Pengaturan Jarak Deteksi Sensor *Proximity*



Gambar 2.28 Skematik Pengaturan Jarak Deteksi Sensor *Proximity* (Omron, 2018)

Mengatur jarak dari permukaan sensor memungkinkan sensor lebih stabil dalam pengoperasiannya. Posisi objek *sensing* transit ini adalah sekitar 70% – 80% dari jarak normal *sensing*.

Jarak sensing ini biasanya tertera pada spesifikasi semua sensor *proximity* yang harus dipenuhi karena jika tidak dipenuhi maka sensor tidak akan mensensing atau mendeteksi jika ada barang yang lewat. Jarak *sensing* biasa di singkat dengan Sn.

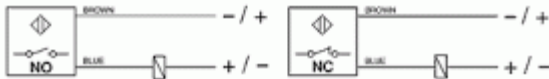
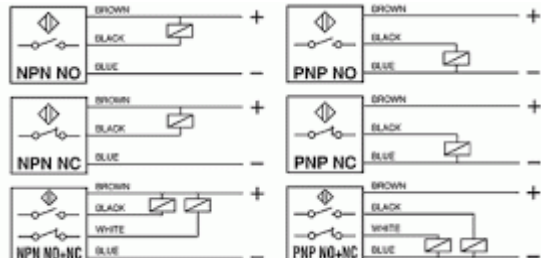
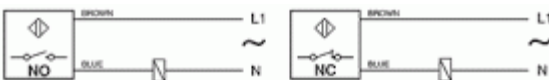
2.2.5 Faktor yang Mempengaruhi Jarak Deteksi Sensor *Proximity*

- Bentuk Objek yang dideteksi disekitar area deteksi sensor
- Bahan penyanggah sensor
- Kecepatan objek yang melewati sensor

2.2.6 Tipe Output Sensor Proximity

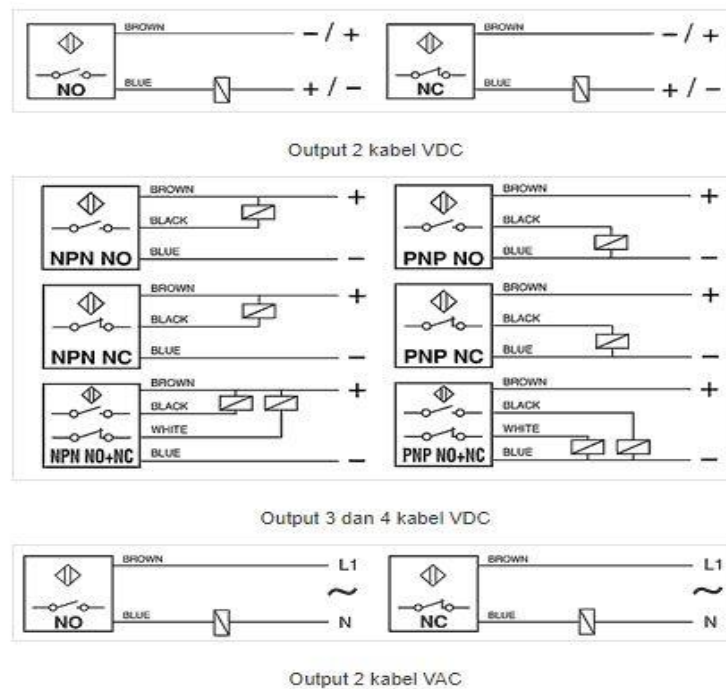
Sementara itu untuk nilai *output* dari *proximity switch* ada 3, yaitu seperti tabel dibawah ini:

Tabel 2.14 Tipe Output Sensor Proximity.

No.	Jenis	Simbol
1.	Output 2 Kabel VDC	
2.	Output 3 dan 4 Kabel VDC	
3.	Output 2 Kabel VAC	

(Fargo, 2017)

Persis seperti fungsi tombol, atau secara spesifik menyerupai fungsi *limit switch* dalam suatu sistem kerja rangkaian yang membutuhkan suatu perangkat pembaca dalam sistem kerja kontinue mesin.



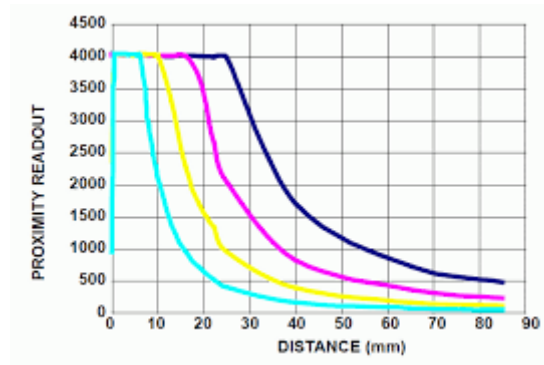
Gambar 2.29 3 Macam *Output Sensor Proximity* (electric mechanic, 2014)

Dengan melihat gambar diatas kita dapat mengenali *type* sensor Proximity Switch ini, yaitu *type* NPN dan *type* PNP. Tipe inilah yang nanti bisa dikoneksikan dengan berbagai macam peralatan kontrol semi digital yang membutuhkan nilai nilai logika sebagai *input* untuk proses kerjanya.

Beberapa jenis *proximity* sensor ini hanya bisa dikoneksikan dengan perangkat PLC tergantung *type* dan jenisnya. Sensor ini juga bisa dikoneksikan langsung dengan berbagai macam peralatan kontrol semi digital seperti *Sensor Controller* dan *Counter relay* digital.

Pada prinsipnya fungsi *Proximity Sensor* ini dalam suatu rangkaian pengendali adalah sebagai kontrol untuk memati-hidupkan suatu *interlock* dengan

peralatan semi digital untuk sistem kerja berurutan dalam rangkaian kontrol.



Gambar 2.30 Grafik perbandingan jarak (mm) terhadap outputnya (mV) (Edy, 2011)

Berdasarkan dari gambar diatas menunjukkan semakin jauh jarak antara sensor dengan objek maka semakin kecil tegangan keluarannya.

2.3 Perangkat-perangkat *Output*

Port-port output sebuah PLC dapat berupa tipe relay atau tipe isolator-optik dengan transistor atau tipe triac, bergantung pada perangkat yang tersambung padanya, yang akan dikontrol. Secara umum, sinyal digital dari salah satu kanal output sebuah PLC digunakan untuk mengontrol sebuah *actuator* yang pada gilirannya mengontrol sebuah proses. Istilah *actuator* digunakan untuk perangkat yang mengubah sinyal listrik menjadi gerakan-gerakan mekanik yang kemudian digunakan untuk mengontrol proses. Berikut ini adalah beberapa contoh dari output PLC.

2.3.1 Tombol Tekan

Prinsip kerja tombol tekan hampir sama dengan saklar tekan yang digunakan pada instalasi penerangan, bedanya jika saklar tekan jenis yang mempunyai togel akan langsung mengikat/mengunci, sedangkan pada tombol tekan tidak ada. Jadi tombol tekan setelah ditekan tidak akan mengunci, tetapi kembali keadaannya semula.



Gambar 2.31 Tombol Tekan (chevy, 2018)

Jadi tombol tekan setelah ditekan tidak akan mengunci, tetapi kembali keadaannya semula. Ada dua kontak yang dapat dilakukan oleh tombol tombol tekan, yaitu :

1. Kontak NO (*Normally Open*)
2. Kontak NC (*Normally Close*)

2.3.2 Lampu Tanda

Lampu tanda atau biasa disebut juga *pilot lamp* digunakan pada peralatan kontrol untuk menandai bekerja atau tidaknya suatu peralatan atau rangkaian, dapat juga sebagai kondisi/keadaan beban. Jika lampu tanda dipergunakan untuk menandai suatu peralatan yang sedang bekerja, maka lampu tanda dipasang seri pada kontak NO, sedangkan apabila lampu tanda digunakan untuk menandai tidak bekerjanya suatu peralatan, maka lampu tanda dipasang paralel pada kontak NC pada rangkaian yang mengontrol peralatan tersebut. Jika lampu tanda dipergunakan untuk menandai keadaan suatu peralatan/beban, maka lampu tanda mempergunakan warna-warna yang berbeda-beda bergantung pada kondisi peralatan/beban yang ditandai. Tabel 2.1 dibawah ini merupakan warna- warna yang menunjukkan fungsi dari lampu tanda.

Tabel 2.15 Fungsi Warna Lampu Tanda

Kondisi peralatan/Beban	Warna lampu
Kondisi tidak normal, beban lebih bahaya.	Merah
Hati-hati, perhatian	Kuning
Posisi siap, mulai beroperasi	Hijau
Beroperasi normal	Putih

Jika lampu tanda dipergunakan untuk menandai keadaan suatu peralatan/beban, maka lampu tanda mempergunakan warna-warna yang berbeda-beda bergantung pada kondisi peralatan/beban yang ditandai. Tabel 2.14 diatas ini merupakan warna- warna yang menunjukkan fungsi dari lampu tanda.



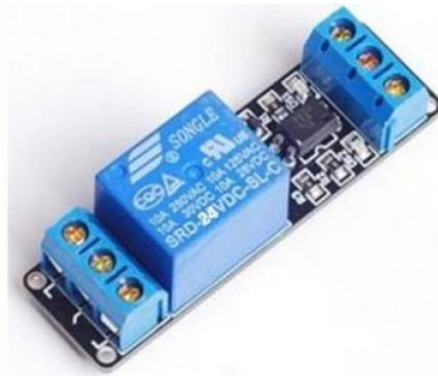
Gambar 2.32 Lampu Tanda (sufijaya, 2019)

Lampu tanda tidak jauh berbeda dengan lampu penerangan biasa, biasanya lampu ini mempunyai tahanan dalam yang besar sehingga dayanya rata-rata kecil. Lampu tanda juga sama seperti lampu penerangan biasa yang mempunyai bentuk bermacam-macam yang biasa dilihat pada gambar 2.31 diatas.

2.3.3 *module relay*

module relay adalah suatu piranti yang bekerja berdasarkan elektromagnetik untuk menggerakkan sejumlah kontaktor (saklar) yang tersusun. *Module relay* akan tertutup (On) atau terbuka (Off) karena efek induksi magnet yang dihasilkan kumparan (induktor) ketika dialiri arus listrik. Berbeda dengan saklar dimana pergerakan *module relay* (On/Off) dilakukan manual tanpa perlu arus listrik.

Sebagai komponen elektronika, *module relay* mempunyai peran penting dalam sebuah sistem rangkaian elektronika dan rangkaian listrik untuk menggerakkan sebuah perangkat yang memerlukan arus besar tanpa terhubung langsung dengan perangkat pengendali yang mempunyai arus kecil. Dengan demikian *module relay* dapat berfungsi sebagai pengaman.



Gambar 2.33 *module relay* (Amazon, 2017)

Ada beberapa jenis *relay* berdasarkan cara kerjanya yaitu:

- *Normaly On* : Kondisi awal kontaktor tertutup (On) dan akan terbuka (Off) jika *relay* diaktifkan dengan cara memberi arus yang sesuai pada kumparan (*coil*) *relay*. Istilah lain kondisi ini adalah *Normaly Close* (NC).
- *Normaly Off* : Kondisi awal kontaktor terbuka (Off) dan akan tertutup jika *relay* diaktifkan dengan cara memberi arus yang sesuai pada kumparan (*coil*) *relay*. Istilah lain kondisi ini adalah *Normaly Open* (NO).
- *Change-Over* (CO) atau *Double-Throw* (DT) : *Relay* jenis ini memiliki dua pasang terminal dengan dua kondisi yaitu *Normaly Open* (NO) dan *Normaly Close*

