Pengenalan Pada PLC

V. Tujuan

- Mahasiswa dapat mengenal cara memprogram PLC menggunakan Do-more.
- Mahasiswa dapat menggunakan instruksi load dan load not pada konsep PLC.
- Mahasiswa dapat membuat program PLC sederhana.

VI. Peralatan

- *Software* Do-More Designer
- *Software* Factory I/O
- Komputer atau laptop
- Koneksi internet

VII. Dasar Teori

Do-more Designer adalah sebuah *software* pemrograman PLC berfitur lengkap untuk rangkaian *Programmable Logic Controller* (PLC). Do-more memiliki manajemen program yang fleksibel dan mendukung perpaduan antara *stage* dan *ladder logic* untuk pendekatan terbaik dari kedua bagian yang menyederhanakan pemrograman dan mempermudah pada saat *troubleshooting*. Yang dimaksud *stage* di sini sebenarnya adalah istilah lain dari sebuah *State* diagram, step program, STL program, ataupun juga SFC program. *State* diagram ini adalah suatu paradigma pemrograman di mana mesin hanya berada pada salah satu kondisi yang berbeda pada waktu tertentu. Lalu selanjutnya adalah *ladder logic* atau *rung* atau juga sering disebut *ladder* diagram adalah garis penghubung antara instruksi *input* dan *output*.

Gambar 19 tampilan *Rung* atau *ladder logic*

Pada Do-More memiliki *address input* dan *output* digital yang dilambangkan dengan variabel X untuk *input* dan Y untuk *output* dengan masing-masing memiliki total nomor alamat 2048 buah yang bernomor 0 hingga 2047. Pada Do-More *Input* terletak di sebelah kiri dan *Output* terletak di sebelah kanan seperti pada gambar di bawah.

Gambar 20 contoh *Input* dan *output* digital pada Do-More

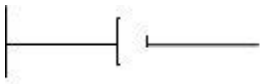
Selain *address input* dan *output* digital Do-More juga memiliki *input* dan *output* analog yang dilambangkan dengan variabel WX untuk *input* dan YX untuk *output* dengan total alamat 256 buah yang bernomor 0 hingga 255. Namun pada Do-More penggunaan *Input* dan *output* analog berbeda dengan digital. Pada *Input* dan *Output* analog membutuhkan instruksi tambahan seperti *Move*.

Dapat dilihat pada gambar di bawah ini cara penggunaan dari *input* dan *output* analog adalah menggunakan instruksi ST1 pada bagian *input*-nya. ST1 ini adalah *input* yang akan selalu *High* atau aktif saat program dimulai dan membuat *Move* ini akan selalu aktif, instruksi ini dapat diganti dengan instruksi lain sesuai kebutuhan. Lalu pada *input* dan *output* analog, *input* dan *output*-nya berada di dalam instruksi *Move* seperti yang terlihat pada contoh di bawah.

Gambar 21 contoh *Input* dan *output* analog pada Do-More

Pada Do-More ini juga memiliki 5 instruksi dasar yang biasa digunakan seperti *load*, *load not*, *counter*, *timer*, dan *compare*. Tetapi pada modul pertama ini kita hanya akan menggunakan *load* dan *load not* saja dan bagaimana cara menghubungkannya pada *software* simulator Factory I/O.

1. ***Load* atau *normally open*** adalah sebuah perintah *input* yang digunakan jika kita menginginkan suatu sistem bekerja pada kondisi logika *High*.



Gambar 22 Simbol instruksi *load*

2. ***Load not* atau *normally close*** adalah sebuah perintah *input* yang digunakan jika kita menginginkan suatu sistem bekerja pada kondisi logika *Low*.

Gambar 23 Simbol instruksi *load not*

Pada modul ini *software* simulasi yang akan digunakan adalah Factory I/O. Pada Factory I/O memiliki banyak *pin* yang dapat dimodifikasi jumlahnya pada menu *configuration* seperti yang dapat dilihat pada gambar di bawah dengan 4 jenis *pin* seperti *Coil* sebagai *output*, *Input* sebagai *input*, *Input* Register sebagai *Input* analog, dan *Holding* register yang biasa digunakan sebagai *output* analog tetapi sebenarnya *Holding* Register dapat digunakan sebagai *input* maupun *output* pada analog.

Gambar 24 tampak *Pin* pada drive simulasi Factory I/O

Untuk menghubungkan Do-More pada Factory I/O kita akan menggunakan jaringan Modbus. Untuk menggunakan Modbus pada Do-More, *address* atau alamat sumber yang digunakan untuk *input* digital adalah MI sedangkan *address* atau alamat yang digunakan sebagai *output* digital adalah MC. MI adalah singkatan dari Modbus *Input* yang cara menggunakannya sama dengan variabel Y pada Do-More tetapi fungsinya berbeda karena MI digunakan sebagai *input* untuk Factory I/O atau *output* dari Do-More sedangkan untuk MC adalah singkatan dari Modbus *Coil* dengan cara menggunakannya sama dengan variabel X pada Do-More tetapi fungsinya juga berbeda karena MC digunakan sebagai *Output* dari Factory I/O atau *input* yang akan masuk pada Do-More.

Gambar 25 contoh *input* dan *output* Digital Modbus pada Do-More

Sedangkan instruksi untuk penggunaan *input* dan *output* analog adalah MIR atau Modbus *Input* Register sebagai *input* pada Factory I/O maupun *output* dari Do-More dengan cara penggunaannya sama dengan variabel WY pada Do-More dan MHR atau Modbus *Holding* Register yang biasa digunakan sebagai *output* analog dari *software* Factory I/O dan *input* analog pada Do-More dengan cara penggunaannya sama dengan variabel WX, tetapi sebenarnya modbus *Holding* register dapat menjadi *output* maupun *input* pada Factory I/O maupun Do-More.

Gambar 26 contoh *input* dan *output* analog pada Do-More

VIII. Tugas

Gambar 27 tampilan sekilas skema pada modul ini

Download-lah skema Factory I/O dari link Google drive berikut: https://drive.google.com/u/3/uc?id=1fWKIRPRdBCLBsQ6h43MNW0GVDUxFnfd0&export=download. Kemudian buatlah agar masing-masing tombol dapat menyala ketika tombol telah ditekan dan mati saat ditekan kembali. Lalu buatlah agar ketika potensiometer diputar LCD dapat menunjukkan angka sesuai pergerakan potensiometer. **(petunjuk: gunakan *input output* digital untuk menyalakan tombol dan analog untuk potensiometer dan LCD yang telah dijelaskan sebelumnya).**