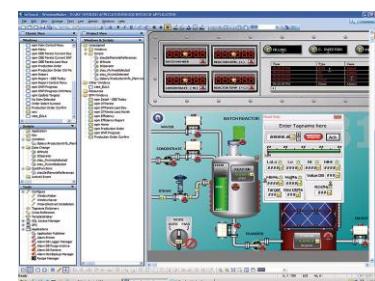




# MODUL PRAKTIKUM

## OTOMASI SISTEM

### SEMESTER GENAP 2014/2015



**LABORATORIUM OTOMASI DAN INFORMATIKA INDUSTRI  
JURUSAN TEKNIK ELEKTRO  
INSTITUT TEKNOLOGI SEPULUH NOPEMBER  
SURABAYA  
2015**



- **MODUL :**  
**DCS/SCADA SYSTEM – CS3000**

- **MODUL :**  
***HUMAN MACHINE INTERFACE (HMI)***  
**WONDERWARE DAN OPC**

- **MODUL :**  
**PLC SIEMENS S7 300**

- **MODUL :**  
**PLC OMRON CQM1**

- **MODUL :**  
**PLC- LG GLOFA GM4**

## DCS/SCADA SYSTEM – CS3000

### A. Pendahuluan

Perkembangan peradaban umat manusia dipengaruhi oleh kemajuan teknologi yang ditandai dengan sistem pembelajaran dan perkembangan metode yang diterapkan dalam kehidupan untuk mengolah sumber daya alam demi kesejahteraan umat manusia. Kontrol merupakan metode untuk menempatkan parameter-parameter lingkungan sesuai dengan keadaan yang diinginkan (*actual value*). Misalnya, membuat temperatur ruang tetap pada suhu 21 °C. Secara umum keterpaduan elemen-elemen yang mempunyai fungsi-fungsi tertentu untuk membentuk suatu fungsi kontrol disebut “Sistem Kontrol”.

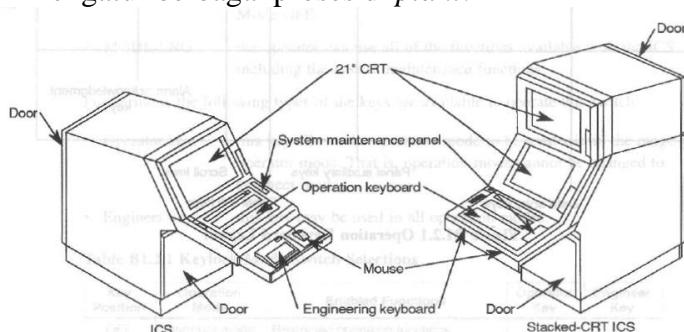
DCS Centum CS merupakan salah satu produk dari Yokogawa. Generasi sebelumnya adalah *Centum V*, dan *Centum XL*. *Centum CS* merupakan DCS untuk menangani *plant* berskala besar. Pada dasarnya komponen utama DCS dapat dibagi menjadi beberapa bagian. Secara umum sistem pengaturan DCS diawali dari *field instrument* yang akan mengirimkan sinyal ke FCS, selanjutnya FCS akan mengolah data dari *transmitter* dan mengirimkan hasil kalkulasi ke *field instrument*. Informasi mengenai pengolahan data dikirimkan secara terus menerus ke komputer di kontrol *room*. *Setting* parameter pada kontrol *valve* bisa dilakukan melalui komputer yang terkoneksi dengan *Field Control Station* (FCS). Secara umum DCS dapat digambarkan sebagai berikut.

#### a. Human Interface Station (HIS)

HIS merupakan suatu perangkat yang berfungsi untuk pengoperasian dan pengawasan / monitoring. HIS ini akan menampakkan variabel proses, parameter kontrol, dan alarm-alarm status segala kejadian proses / *plant*. Wujud HIS berupa perangkat komputer.

#### b. Komputer (EWS)

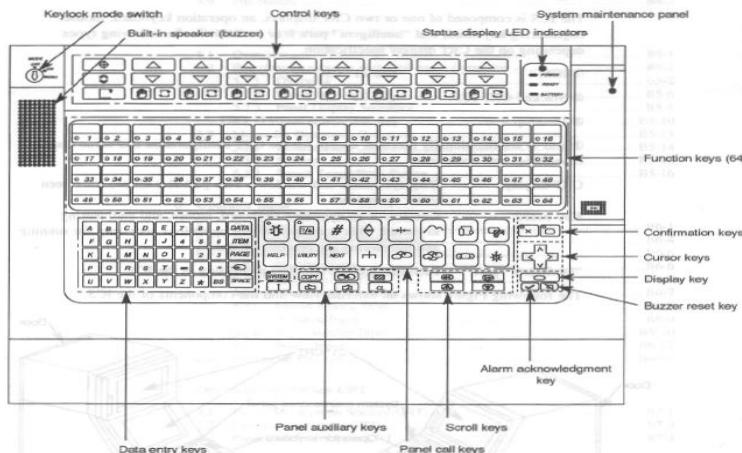
Komputer merupakan salah satu bagian yang tidak dapat dipisahkan dari DCS. Ada beberapa jenis komputer yang digunakan oleh *Centum CS*. Antara lain *Engineering Work Station* (EWS) dan server untuk menyimpan dan menganalisa data yang dihasilkan oleh sistem CS. EWS merupakan komputer yang umum kita jumpai di kontrol *room*. EWS digunakan untuk mengatur berbagai proses di *plant*.



Gambar 1.1, *Engineering Work Station* (EWS)

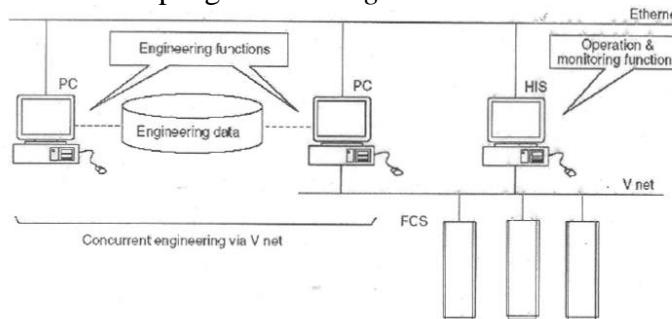
EWS biasanya diatur oleh operator. Komputer jenis ini sering kita jumpai pada kontrol *room*. *Console* computer biasanya tersimpan dalam boks casing. Pada EWS terdapat *Keyboard* khusus yang digunakan untuk mengatur berbagai proses di *plant*. *Keyboard*

tersebut memang didesain untuk menjalankan berbagai operasi di *plant* secara efisien. Oleh karena itu banyak dijumpai tombol *shortcut* untuk berbagai aplikasi kontrol.



**Gambar 1.2 , Keyboard EWS**

EWS mutlak diperlukan dalam sistem DCS. Setelah *plant* berjalan selama beberapa tahun pasti ada perubahan yang harus dilakukan, misalnya penambahan *instrument* baru. Setelah *hardware instrument* terinstall di *plant* maka dilakukan setting DCS melalui *software*. *Setting* ini diperlukan supaya perangkat baru tersebut dapat dikenali oleh DCS. *Setting* semacam inilah yang dilakukan oleh EWS. Komputer EWS adalah komputer biasa yang sering kita temukan dipasaran. Komputer tersebut biasa disebut personal komputer (PC). Hanya saja EWS mempunyai jaringan komunikasi dengan FCS, sehingga perubahan yang dilakukan oleh EWS akan mempengaruhi *setting* FCS.



**Gambar 1.3 , EWS ( Engineering Function)**

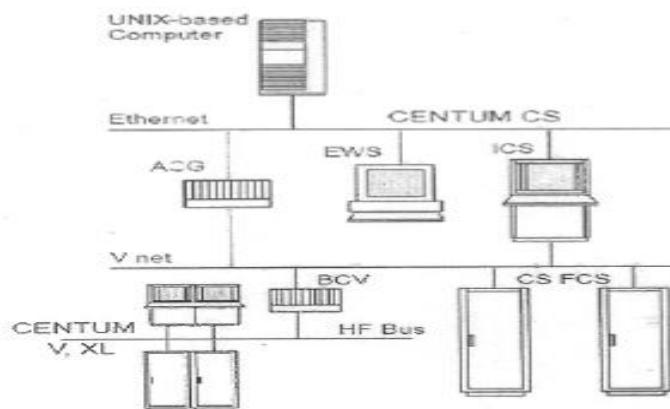
#### c. Field Control Station (FCS)

FCS berfungsi sebagai kontroler. Semua perhitungan mengenai kontrol di *plant* dilakukan di FCS. Dapat dikatakan bahwa FCS adalah otak pengaturan dari DCS. FCS befungsi melakukan pengontrolan *regulatory*, *sequence* atau *batch*. FCS merupakan bagian DCS yang terhubung langsung dengan berbagai *field instrument* di lapangan. Sinyal dari *field instrument* dikirim dengan menggunakan kabel biasa.

#### d. Field instrument

*Field instrument* adalah perangkat pendukung dalam sistem pengaturan otomatis. Contoh dari *field instrument* adalah *transmitter* dan *control valve*. *Transmitter* adalah perangkat yang memberikan informasi mengenai kondisi parameter yang dikontrol oleh *plant*. Sinyal

yang dikirimkan ke kontroler adalah sinyal *analog* dan *digital*. *Transmitter* dihubungkan dengan menggunakan kabel ke *node*. *Control valve* juga merupakan *instrument* pendukung yang bekerja memberikan aksi pengaturan pada proses. Semua *field instrument* yang terhubung pada DCS mempunyai fungsi yang spesifik. Kondisi semua *instrument* sangat menentukan kehandalan sistem kontrol.



**Gambar 1.4 , Konfigurasi DCS Centum CS**

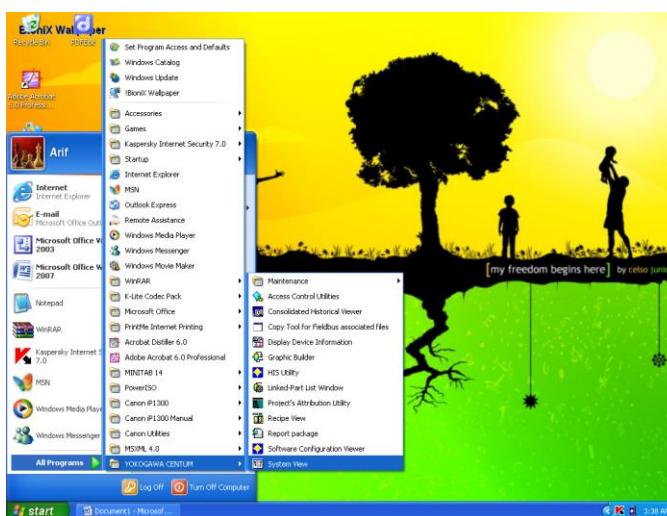
Sistem DCS lama dan baru bisa dikoneksikan dengan menggunakan perangkat *gateway*. Semua sistem dapat saling berkomunikasi melalui jaringan ethernet dan V-Net. Pada gambar tersebut dapat dilihat adanya *server*. *Server* digunakan untuk melakukan *data recording* dan memproses data sehingga dapat dianalisa oleh pihak-pihak terkait.

## B. Membuat Database Project

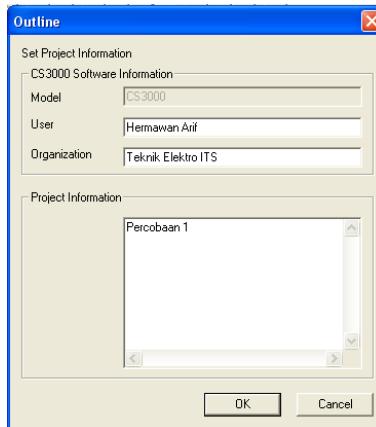
Sebelum membuat program untuk aplikasi kontrol terlebih dahulu harus dibangun *database project* untuk menempatkan program aplikasi yang akan dibuat. Database dibangun melalui menu [System View] pada program [YOKOGAWA CENTUM]. Setiap *project* akan mempunyai 4 folder dengan fungsi yang berbeda-beda. Keempat folder tersebut antara lain COMMON, BATCH, FCSXXXX, dan HISXXXX.

Prosedur untuk membuat *database* projek adalah sebagai berikut:

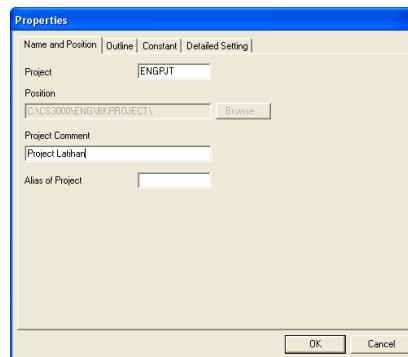
1. Aktifkan System View dengan memilih [Start] ---> [Program]---> [YOKOGAWA CENTUM]---> [System View]



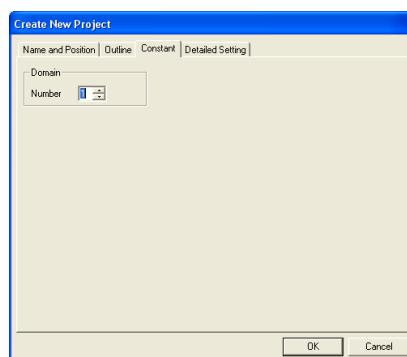
2. Pilih [File]-->[create New]-->[Project]
3. Jendela [ Outline ] akan muncul. Masukkan data pada kolom “User, Organization, dan Project information”



4. Jendela [Create New Project] akan muncul. Masukkan data sebagai berikut :



Setelah mengisi data, Klik tab [Constan] dan definisikan domain number.



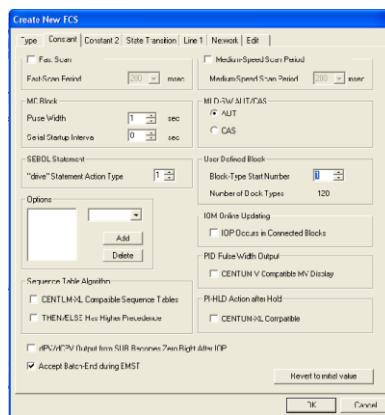
Klik tombol [OK]

5. Jendela [Create New FCS] akan muncul. Set data sebagai berikut :

Station Type	:AFS20D Duplexed Field Control Unit (for RIO, with Cabinet)
Database Type	:General Purpose
Domain Number	:1
Station Number	:1

Yang lainnya biarkan kosong

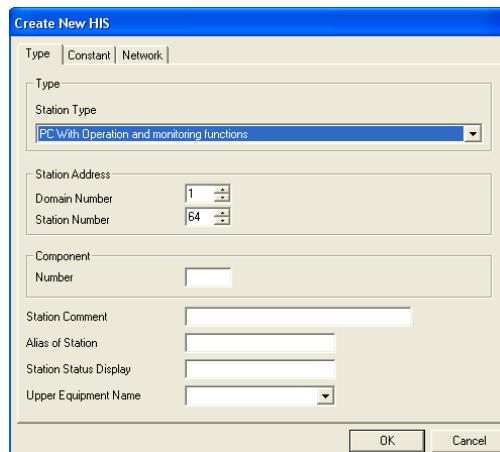
Lalu Klik tab [Constan] dan definisikan:  
 User-Defined Block / Block-Type Start Number :1  
 Dan yang lain biarkan *default*.



Klik tombol [OK]

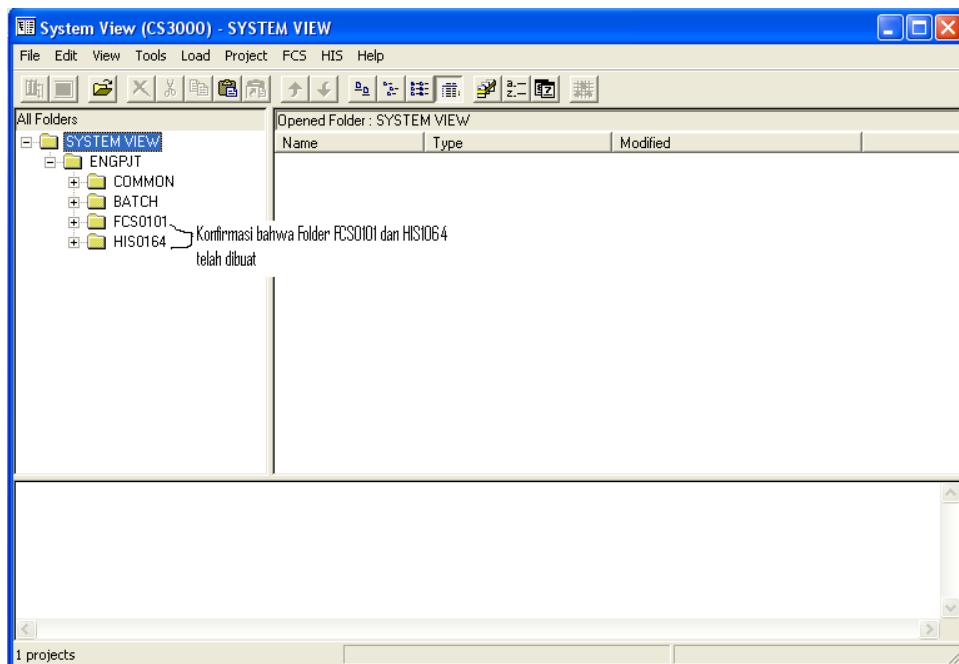
6. Jendela [Create New HIS] akan muncul

Station Type : PC with Operation and monitoring functions  
 Station Address / Domain Number : 1  
 Station Address / Station Number: 64  
 Yang lain biarkan *default*



Klik tombol [OK]

7. Konfirmasi bahwa Project ENGPJT telah dibuat dengan [System View]. Buka ENGPJT untuk memastikan bahwa folder FCS0101 dan HIS0164 telah dibuat



### B.1. Tipe-Tipe *Project* Terdiri Atas Tiga Macam, Yaitu :

#### 1. Default Project

Project pertama kali dibuat statusnya adalah *default project*

Project tipe ini mempunyai karakteristik antara lain :

- Project bisa di download ke FCS
- Project bisa di simulasi dengan FCS simulator dengan mode *virtual test*
- Project bisa di download ke HIS
- Project bisa di download secara off-line ke FCS

#### 2. Current Project

Setelah default project sukses di download, status project berubah menjadi current project.

Karakteristik current project adalah :

- Tidak bisa membuat current project yang ganda
- Project bisa dilakukan *target test*
- Project bisa di download ke HIS
- Project bisa di download ke FCS

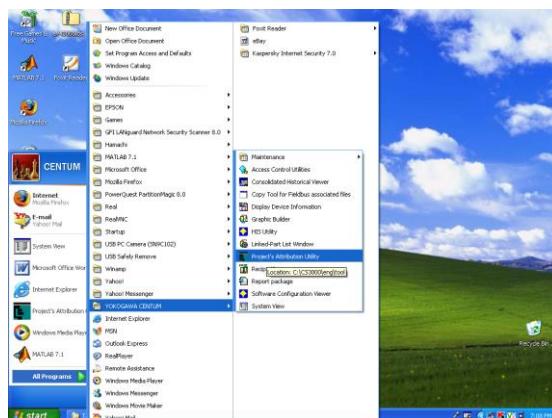
#### 3. User-Defined Project

Merupakan status project yang di salin (*copy*) dari default project dan Current Project . Project yang berstatus *User-Defined Project* tidak bisa di load ke HIS. Project hanya bisa digunakan sebagai *backup* dan *virtual test*. Karakteristik current project adalah :

- Pembuatan project ganda berstatus User-Defined Project bisa dilakukan melalui System View
- Project bisa di test dengan *virtual test* dengan FCS simulator
- Project tidak bisa di download ke FCS atau HIS

Cara mengubah tipe *project* yaitu dengan menggunakan fasilitas *Project's Attribution Utility* seperti ditunjukkan pada prosedur berikut:

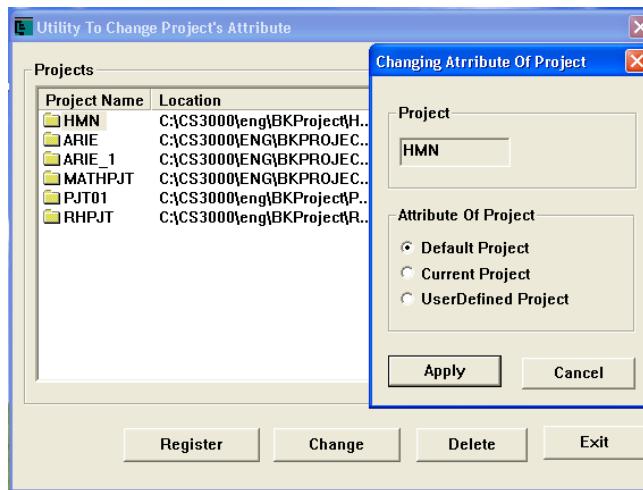
1. Aktifkan *Project's Attribution Utility* dengan memilih [Start] ---> [Program]---> [YOKOGAWA CENTUM]---> [*Project's Attribution Utility*]



2. Maka akan muncul peringatan pada jendela Utility To Change Project's Attribute. Klik [OK] pada jendela konfirmasi.



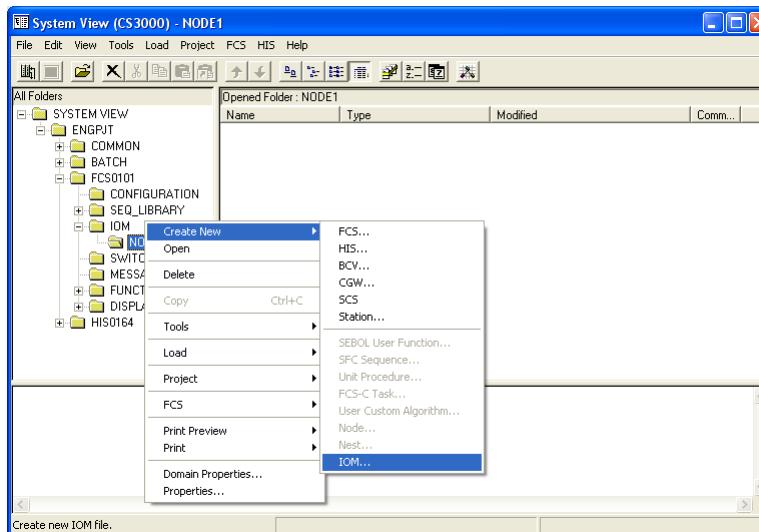
3. Selanjutnya pada jendela Utility To Change Project's Attribute pilih project yang akan diubah attributnya dengan cara klik kiri pada project--> [Change], pilih attribute --> [Apply]. Lalu keluar jendela dengan cara klik [Exit]



### C. Inisialisasi Input/Output (I/O) Proses

Pada bagian ini didefinisikan unit *analog I/O* pada slot 1unit 1 FCS0101. Prosedur inisialisasi input/output analog adalah sebagai berikut :

1. Aktifkan [System View] kemudian pilih [ENGPJT]-->[FCS0101]--> folder [IOM]

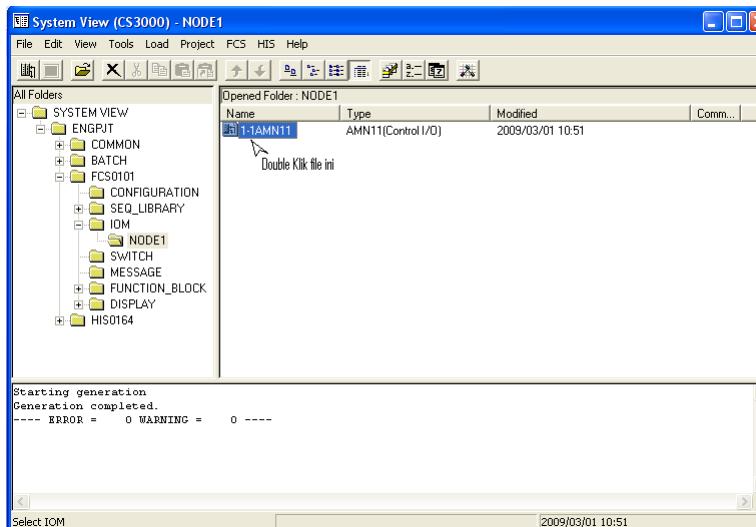


2. Jendela [Create New IOM] akan muncul. Definisikan sebagai berikut :

IOM Type / Category	:AMN 11/AMN 12 [Control I/O]
IOM Type / Type	:AMN 11[Control I/O]
Installation Position/Unit	: 1
Installation Position/Slot	: 1

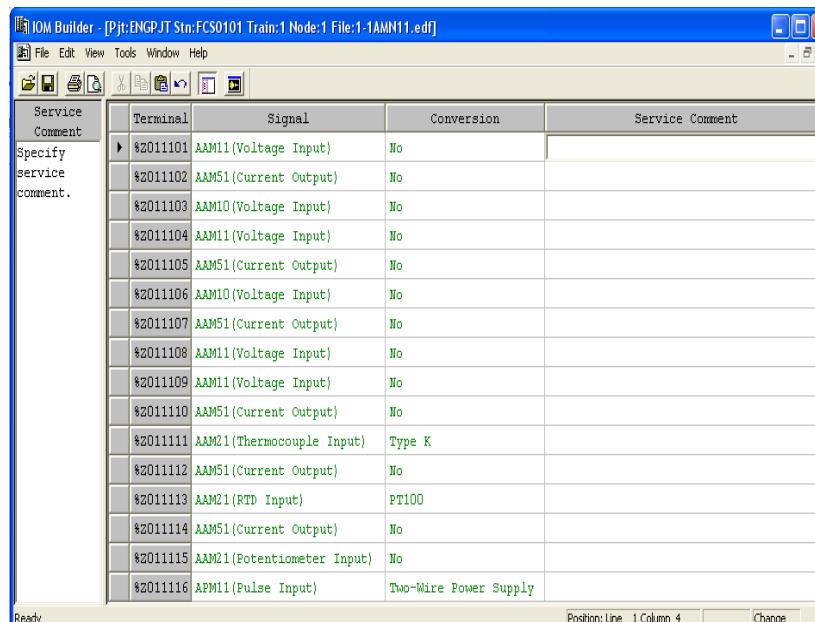
Klik [OK]

3. Gambar dibawah menunjukkan file 1-1AMN 11 yang telah dibuat pada folder [NODE1]

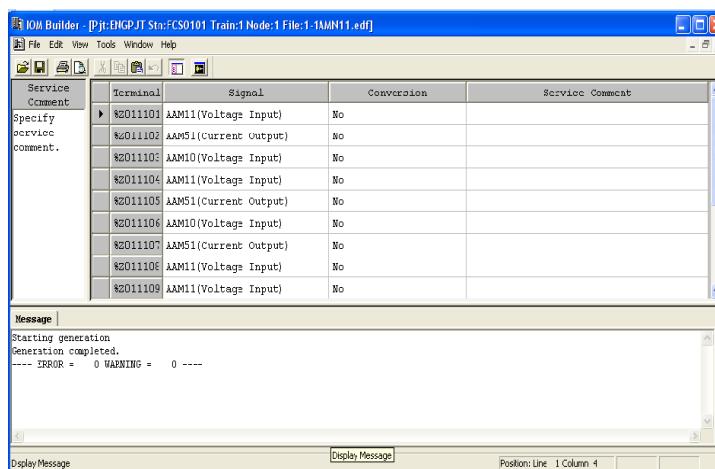


Aktifkan [IOM Builder] dengan klik dua kali file 1-1AMN 11

4. Jendela [IOM Bulder] akan muncul. Tentukan modul I/O untuk masing-masing terminal pada kolom Signal dan Conversion seperti pada gambar berikut :



5. Simpan file IOM Bulder yang telah kita buat, dengan memilih [File]-->[Save]



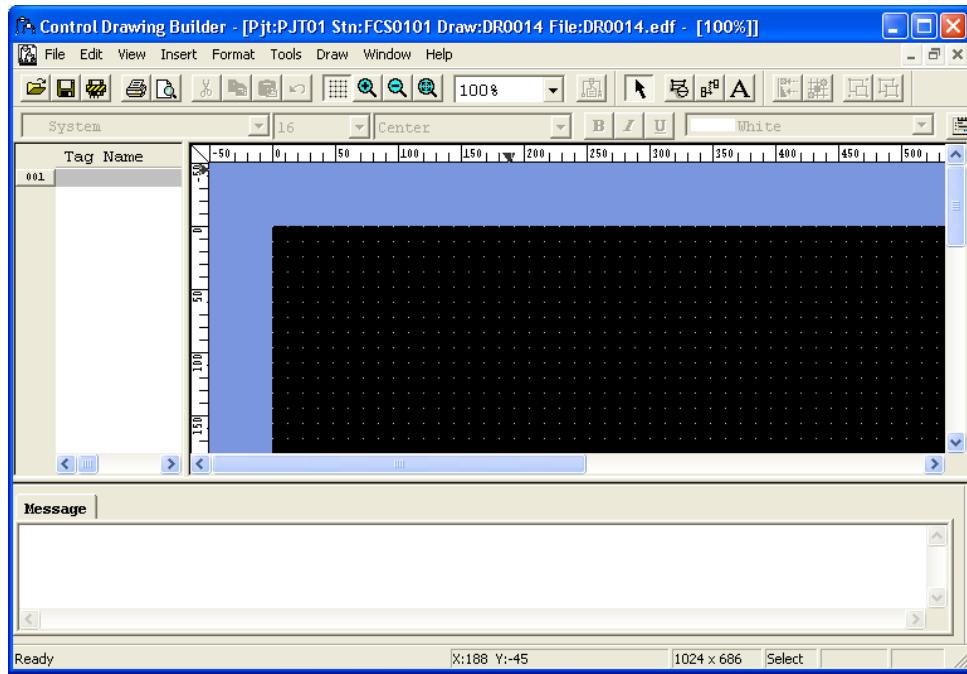
Jika tidak terdapat error pada [Message] area, keluar dari IOM Bilder. Dari Tollbar, Pilih [File] -- > [Exit IOM Builder]

#### D. Control Drawing Builder

*Control Drawing Builder* digunakan untuk menyusun fungsi dasar kontrol dari FCS. Dengan *Control Drawing Builder*, operasi seperti mendaftarkan blok fungsi ke dalam file *drawing* dan menentukan aliran data antar blok fungsi dapat dilakukan secara grafis.

CS 3000 atau Centum 3000 berisi 200 *Control Drawing* yang mana lebih banyak dibanding CS 1000 yang hanya berisi 50 *Control Drawing*. *Control Drawing Builder* dapat dimulai dengan memilih file dari aplikasi *System View*.

[FCS0101] → [Function Block] → klik dua kali pada [DR0001]

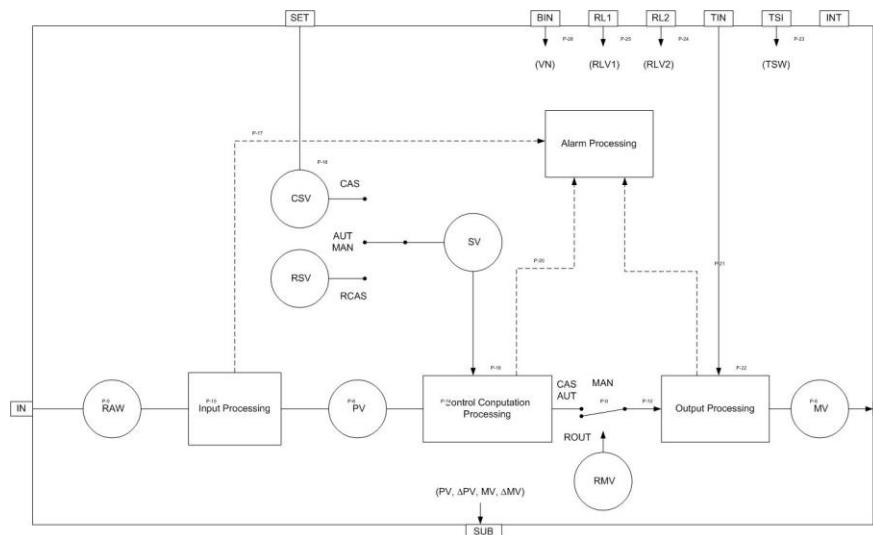


### D.1. Fungsi Control

Ada beberapa fungsi blok yang sering digunakan di dalam dunia teknik, diantaranya adalah sebagai berikut :

- Fungsi Blok Regulatory

Fungsi kontrol regulatory melaksanakan proses perhitungan kontrol menggunakan nilai proses analog untuk monitor proses dan kontrol proses. Blok fungsi yang menyediakan fungsi kontrol regulatory disebut “blok kontrol regulatory”. Blok kontrol regulatory mendukung mengikuti tipe proses : *input processing, control computation processing, output processing* dan *alarm processing*.



## Blok regulator

Keterangan :

IN	: Input Terminal	PV	: Process Variable
SET	: Setpoint value input terminal	SV	: Setpoint value
BIN	: Compensation input terminal	CSV	: Cascade setpoint value
RLn	: Reset signal input terminal	RSV	: Remote setpoint value
TIN	: Tracking signal input terminal	VN	: Compensated value input
TSI	: Tracking switch input terminal	RMV	: Remote manipulated output value
INT	: Interlock switch input terminal	RLVn	: Reset signal
SUB	: Auxiliary output terminal	MV	: Manipulated output value
OUT	: Output terminal	TSW	: Tracking switch
RAW	: Raw data input signal		

Blok kontrol *regulatory* harus mengikuti empat fungsi *processing* :

### **Input Processing**

Menerima sinyal dari terminal input dan output *Process Variable* (PV).

### **Control Computation Processing**

Melakukan *Control Computation Processing* dengan membaca *Process Variable* (PV) dan *Manipulated output Value* (MV).

### **Output Processing**

Membaca *Manipulated output Value* (MV) dan output hasil dari *Control Computation Processing* dari terminal output sebagai sinyal output.

### **Alarm Processing**

Mendeteksi ketidaknormalan dalam *Process Variable* (PV) atau *Manipulated output Value* (MV) dan memberitahu fungsi operasi dan monitoring.

#### - Fungsi Kontrol Sequence

Blok kontrol *sequence* yang mana mengeksekusi kontrol sequence meliputi blok tabel sequence, blok logic chart, blok SFC, blok switch instrument, blok sequence element, dan blok valve monitoring.

#### 1). Blok *Tabel Sequence*

Blok tabel sequence (ST 16, ST 16E) merupakan tabel keputusan, tipe blok fungsi yang mendeskripsikan hubungan antara sinyal input dan sinyal output dalam tampilan Y/N (Yes/No). Melalui membuat koneksi sequence dengan blok fungsi yang lain, mereka mengatur monitoring dari proses dan fase step sequences.

Kondisi dan operasi disusun dalam tabel format dan spesifies yang mana operasi dilakukan dengan kombinasi beberapa kondisi. Hal ini sesuai untuk mendeskripsikan dari semua sequence diantaranya pararel operation, inter lock operation dan seqence operation.

Di bawah ini dua model blok yang dikategorikan sebagai blok tabel sequence.

##### • Blok Tabel Sequence (ST 16)

Memiliki fungsi kontrol sequence yang menangani 64 sinyal EO, 32 rules.

##### • Blok Rule Extension (ST 16E)

Blok fungsi ini digunakan untuk Rule Extension dari blok tabel sequence (ST 16). Blok fungsi ini terhubung pada suatu extending sequence blok (ST 16) sebagai suatu tabel sequence yang diperluas hingga ke bentuk suatu group tabel sequence. Selain tipe step blok, tabel sequence (ST 16) tidak dapat terkoneksi.

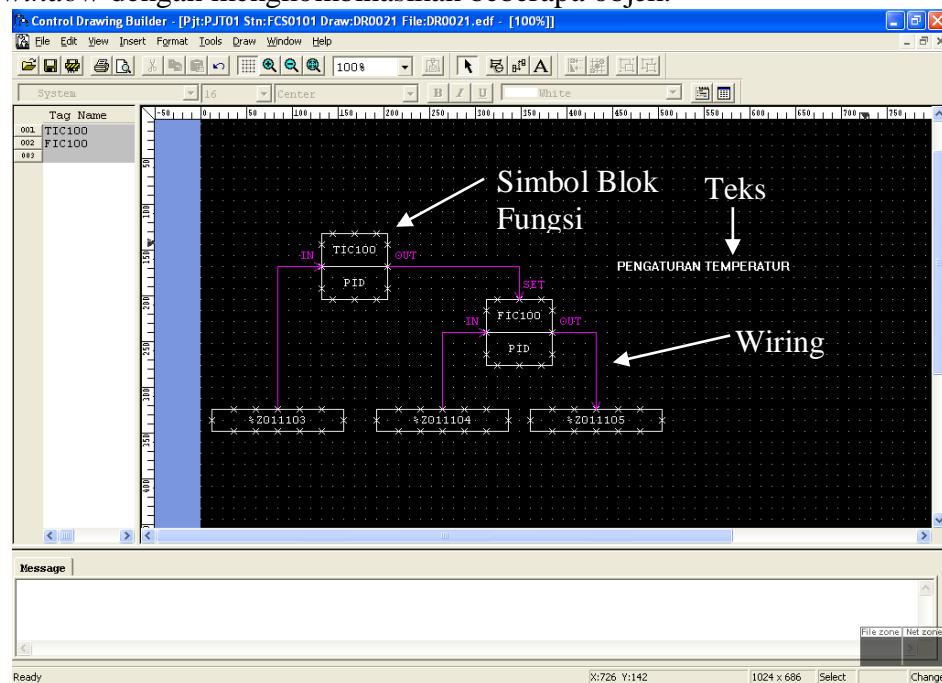
#### 2). Logic Chart

Blok *logic chart* (LC64) bisa mengkombinasikan atau menyusun sinyal dari blok fungsi yang lain, I/O proses dan I/O software ke dalam suatu aplikasi untuk menyambungkan satu sama lain kontrol *sequence*.

*Logic chart* terdiri dari sinyal kondisi, sinyal aksi dan operator logika. Blok LC64 adalah blok fungsi kontrol sequence dengan 32 *channels* sinyal input dan sinyal output serta mampu menangani 64 operator logika. (untuk lebih lengkapnya dapat dibaca di buku panduan “Centum CS 1000/3000 R3 Engineering Course”)

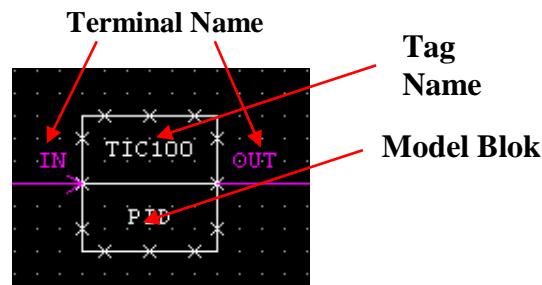
## D.2. Bagian dari *control drawing*

Dengan *control drawing builder*, fungsi dasar kontrol dari FCS dapat disusun di dalam *window* dengan mengkombinasikan beberapa objek.



### a. Blok Fungsi

Objek ini merepresentasikan blok fungsi yang disebut simbol blok fungsi.



### b. Teks

Objek ini merepresentasikan *arbitrary character string*.

KONTROL KASKADE

c. *Wiring*

Objek ini merepresentasikan aliran data antar blok fungsi atau antara blok fungsi dengan blok *data link*.

d. *Blok Data Link*

Objek ini merepresentasikan blok *data link* yang disebut simbol blok *data link*. Blok *data link* digunakan ketika menkoneksi blok fungsi dengan blok fungsi lain atau mengkoneksi suatu I/O proses atau I/O *software*.

Dua blok *data link* yang berlaku, yaitu : blok *data link* I/O dan blok *data link* eksternal. Blok *data link* I/O dapat koneksi dengan suatu I/O proses atau I/O *software*. Sedangkan blok *data link* eksternal dapat koneksi dengan blok fungsi yang didefinisikan dalam *control drawing* yang lain atau blok fungsi yang didefinisikan di *control station* yang lain.



Blok *Data Link*

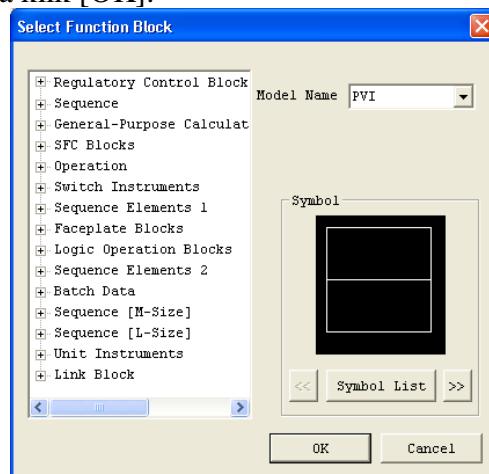
### D.3. Fungsi Toolbar

#### D.3.1. Drawing toolbar



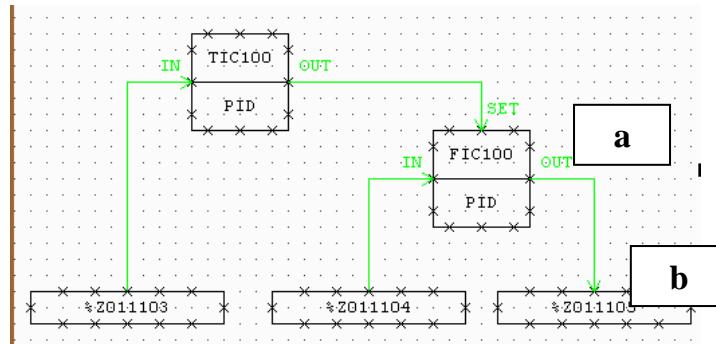
- Select Mode
- Teks
- Blok Fungsi

Objek ini digunakan untuk menampilkan blok fungsi ke *window control drawing buider*. Blok fungsi dapat dipilih dengan cara memilih salah satu model blok yang disediakan pada kotak sebelah kiri atau dengan cara mengetikkan nama model blok yang diinginkan dan selanjutnya klik [OK].



- Wiring*

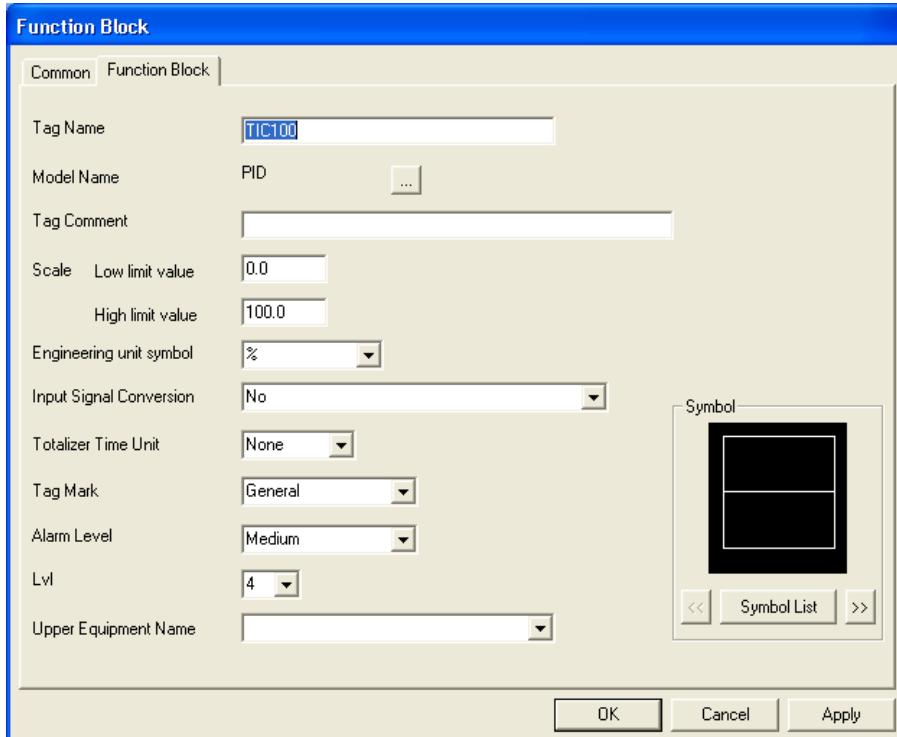
Objek ini merepresentasikan aliran data antar blok fungsi atau antara blok fungsi dengan blok *data link*.



Klik pada titik a kemudian klik dua kali pada titik b. *Wiring* akan menarik secara otomatis dengan garis tebal warna hijau.

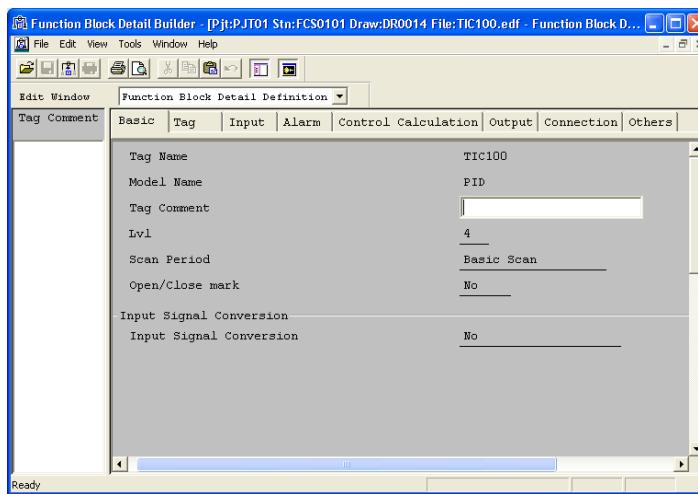
### D.3.2. Properties blok fungsi

Di dalam *control drawing builder*, kita dapat lebih spesifik merubah parameter dari blok fungsi melalui *window properties*. Yaitu dengan meng-klik blok fungsi yang kita inginkan kemudian klik kanan dan pilih [properties], sehingga akan muncul *window* seperti berikut.



### D.3.3. Spesifikasi detail blok fungsi (*Edit Function Block Detail*)

Di dalam *control drawing builder*, dengan *menu* ini kita juga dapat merubah spesifikasi blok fungsi dengan lebih detail dibanding *menu properties*, yaitu dengan cara meng-klik blok fungsi yang diinginkan dan selanjutnya pada *menu toolbar* dipilih [Window] → [Edit Function Block Detail], sehingga akan muncul jendela seperti berikut:



#### D.4. Contoh Pembuatan *Control Drawing*

Ada beberapa contoh pembuatan *control drawing* dari buku panduan “Engineering Course (Laboratory Excercise)”, yaitu sebagai berikut :

##### A. Control Cascade

Langkah pembuatan *Control Drawing* untuk *control cascade* adalah sebagai berikut :

###### 1. Setting Ukuran Panel

Pilih [properties] pada menu file control drawing builder. Pilih tab [attribute] konfirmasikan ukuran window adalah [1024x686]. Klik [OK]

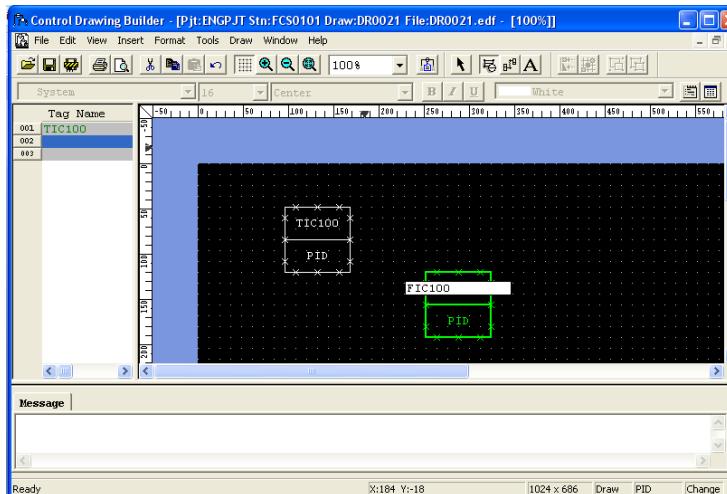
###### 2. Setting Grid

Pada jendela “Grid Option” set “Grid Type” berupa point, “grid color” gray 75, beri tanda centang pada “Display Grid” serta set “Grid Size” 12

###### 3. Membuat blok fungsi

Primary temperature controller [PID] :TIC100

Secondary flow controller [PID] : FIC100



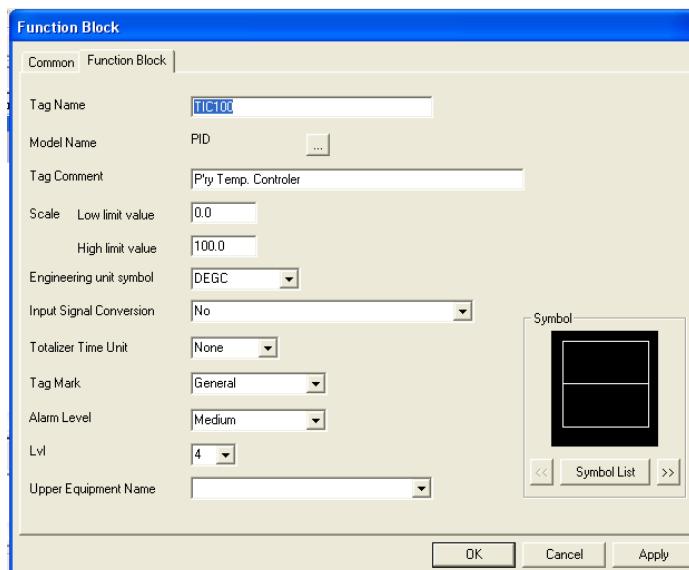
Atur menu properties kedua blok fungsi di atas sesuai data bi bawah ini.

TIC100

Tag Comment : P'ry Temp. Controller  
Scale/ High limit value : 100.0  
Engineering unit symbol : DEGC

FIC100

Tag Comment : S'ry Flow. Controller  
Scale/ High limit value : 50.0  
Engineering unit symbol : L/M



#### 4. Membuat blok Link I/O

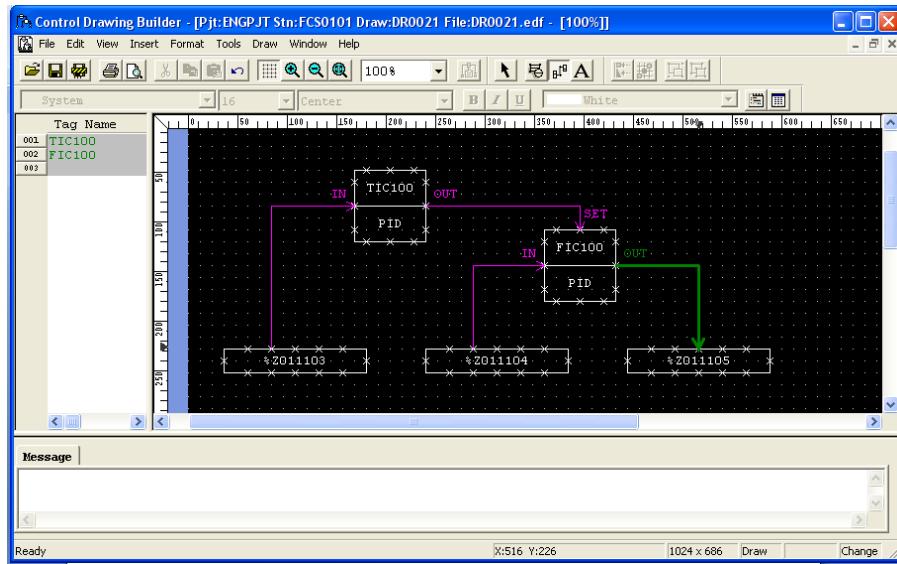
- Buat blok fungsi PIO yang berfungsi sebagai *input* untuk TIC100 dan beri nama %Z011103
- Dengan cara yang sama, buatlah:

*modul input* FIC100 : %Z011104  
*modul output* FIC100 : %Z011105

#### 5. Membuat sambungan/ pengkabelan

- Dari *toolbar*, pilih [Insert] -->[Wiring]
- Sambungkan [%Z011103] ke terminal [IN] TIC100.
- Sambungkan [OUT] TIC100 ke [SET] FIC100
- Ubah nama terminal FIC100 dari [IN] ke [SET]  
Klik kiri karakter [IN], lalu klik kanan pada mouse. Pilih [Terminal Name]--> [IOI] --> [SET]
- Hubungkan [%Z011104] ke [IN] FIC100 dan [OUT] terminal FIC100 ke [%Z011105]

Gambar yang sudah dibuat akan tampak seperti dibawah :



Pilih blok simbol [FIC100] dengan cara klik kiri blok kemudian pada toolbar [window] pilih [Edit Function Block detail].

- f) Maka jendela [Function Block Detail Bulder] akan muncul. Melalui sidebar spesifikasi Measurement Tracking.

Measurement Tracking / MAN : 1 :YES

Jika diinginkan spesifikasi yang lebih detail, maka pada toolbar [View] → [Detail Setting Items]. Maka tab untuk bermacam-macam setting akan muncul. (*Basic, Tag, Input, Alarm, Control Calculation* dll).

Setelah selesai penyetelan, update item dengan cara pada menu toolbar [File] → [Update].

- g) Setelah di update, keluar dari Function Block Detail Bulder. Pada toolbar [File] → [Exit Function Block Detail Bulder].

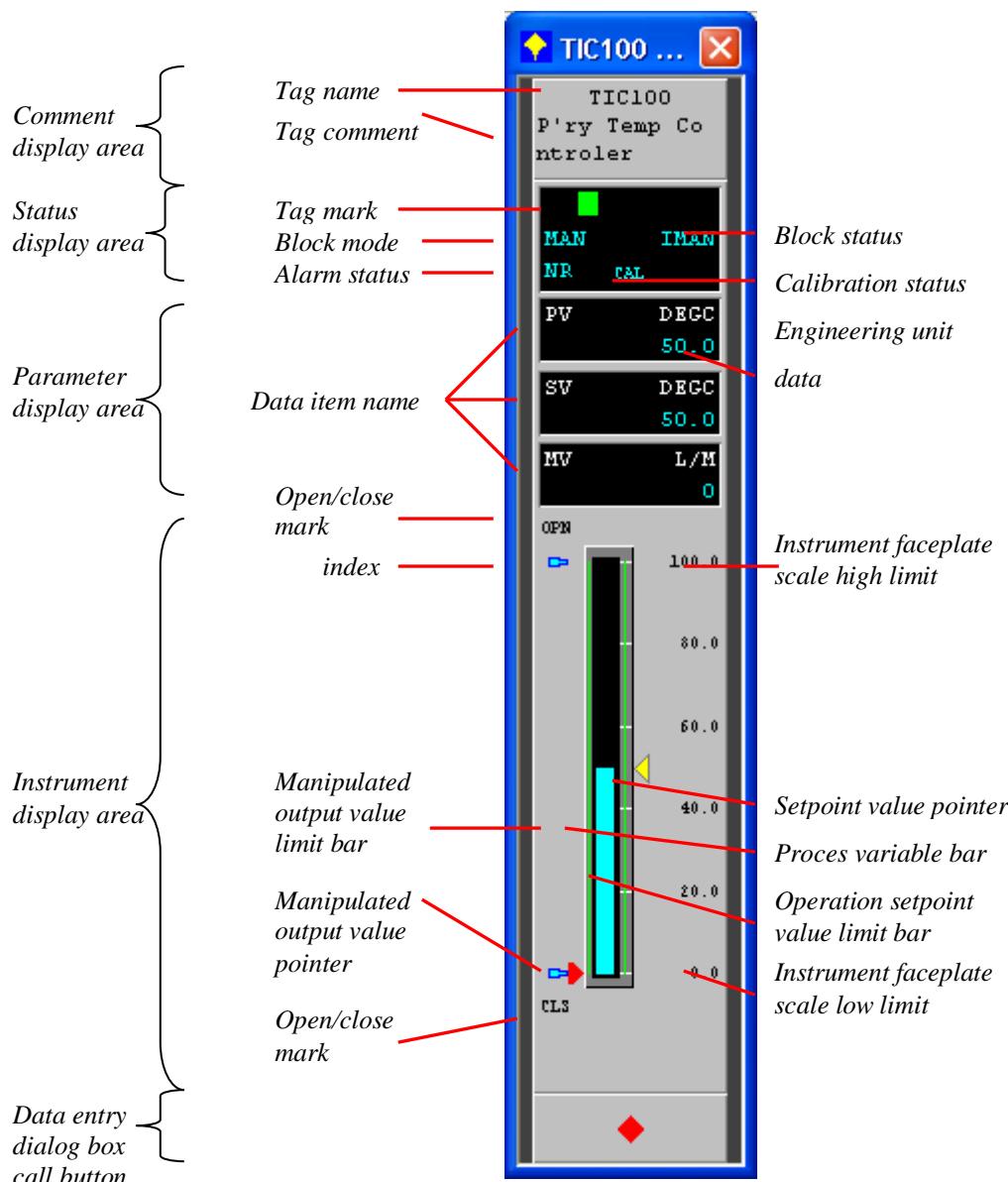
- h) Simpan item-item yang sudah didefinisikan. [File] → [Save].

Jika terdapat error, maka error message akan muncul. Perbaiklah terlebih dahulu dengan meruntut pada pesan yang ditampilkan.

Jika tidak terdapat error, keluar dari [Control Drawing Builder]. Dari toolbar [File] → [Exit control drawing builder].

## E. Instrument Faceplate

*Instrument faceplate* menampilkan status dan data blok fungsi, instrumen atau *contact I/O* secara grafis dan ditampilkan pada satu *window*. *Faceplate window* tidak hanya digunakan untuk memonitor tapi juga untuk mengubah-ubah dari parameter-parameter yang digunakan serta mode operasi yang ada. *Instrument faceplate* dalam suatu *faceplate window* atau *graphic window*. Gambar dibawah ini menampilkan salah satu jenis *faceplate window*.



Secara umum setiap faceplate instrument terdiri dari beberapa komponen berikut ini:

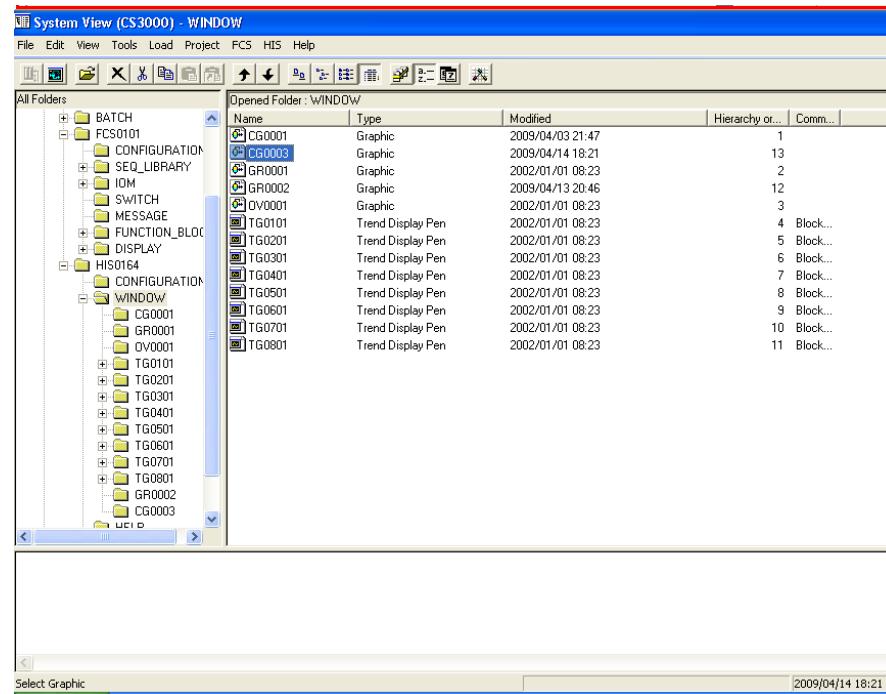
- Comment display area
- Status display area
- Parameter display area
- Instrument display area
- Operation mark
- Data input dialog box call button

## E.1. Pembuatan Control Window

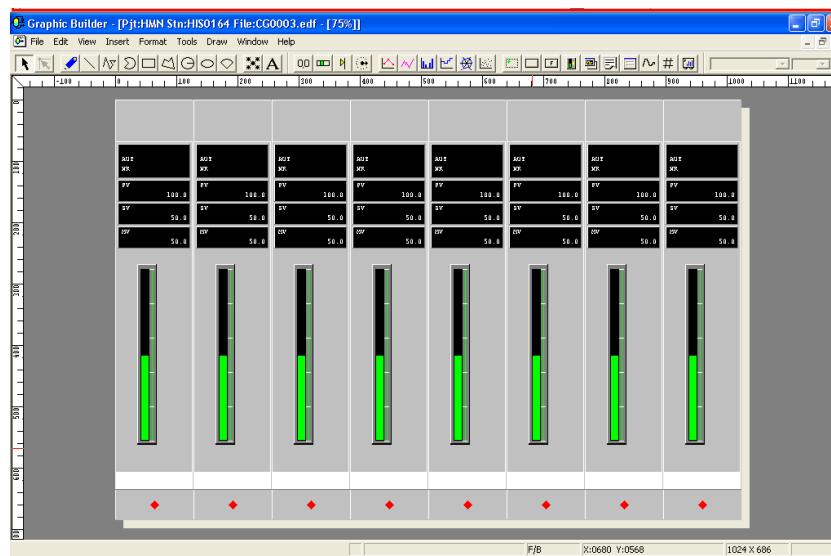
Prosedur :

1. Dari System View pilih project, [HIS0164] → [WINDOW]

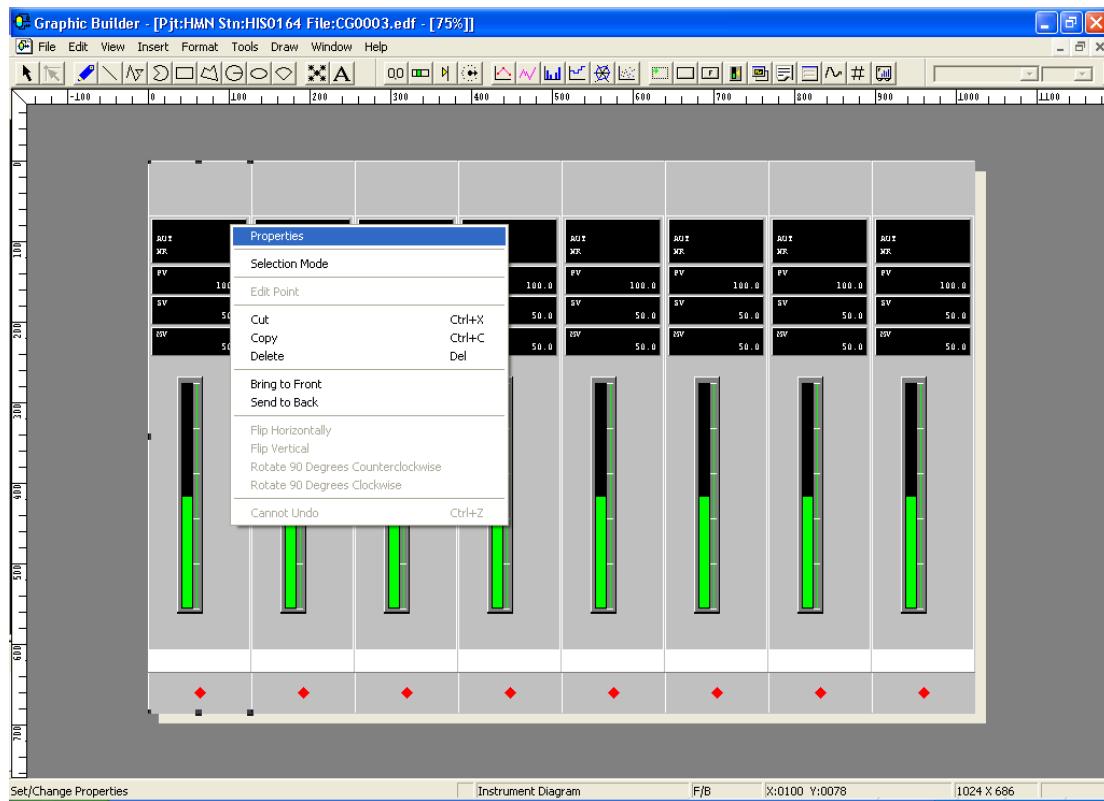
Klik ganda [CG0003] untuk membuka [Graphic Builder]



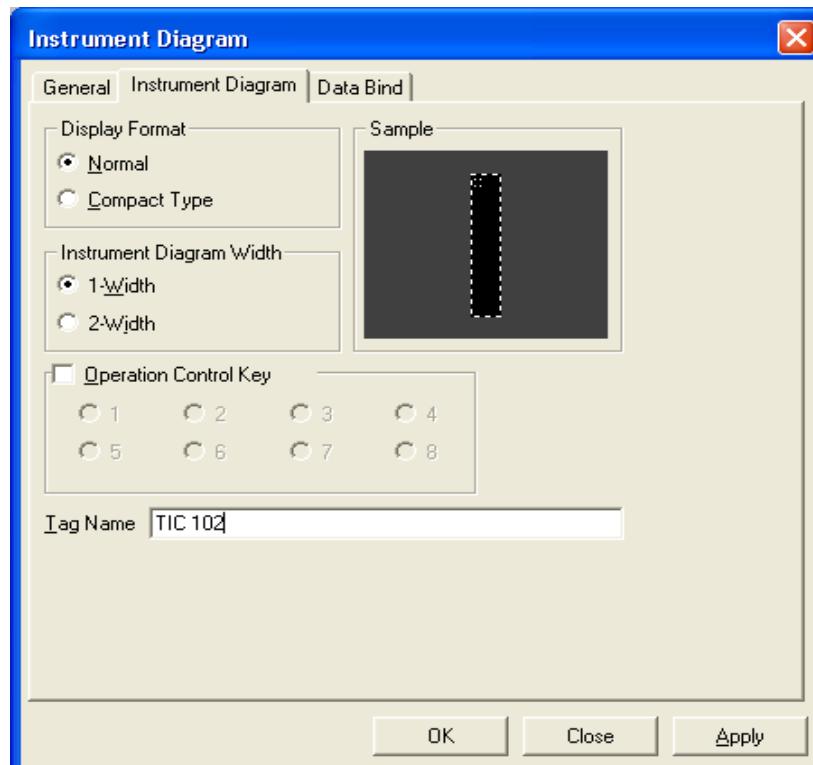
2. Panel *graphic builder* akan muncul seperti dibawah ini.



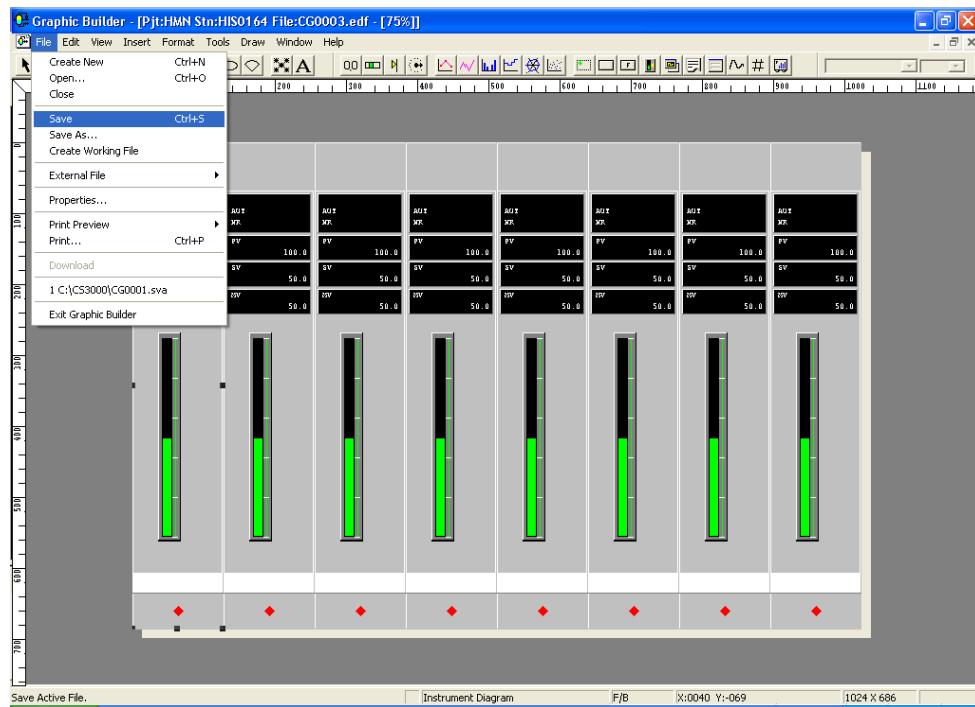
3. Klik kanan pada salah satu diagram instrumen dan kemudian pilihlah [Properties] dari menu yang ada.



- Setelah [Instrument Diagram] window muncul, isilah nama yang diinginkan pada kolom [Tag Name]. Pada contoh ini misalkan TIC102. dan kemudian klik [Apply] dan [OK].



- Lakukan langkah serupa pada langkah ke-4 untuk mendefenisikan diagram instrumen yang dibutuhkan.
- Setelah defenisi nama dari diagram instrumen selesai, pilihlah [File] → [Save] dari toolbar..

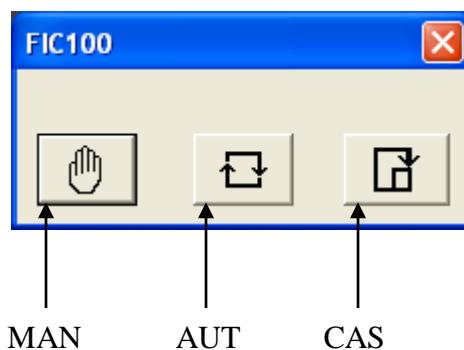


## E.2. Operasi pada Instrument Faceplate

Pada instrument faceplate terdapat beberapa operasi yang dapat digunakan, yaitu:

- Block mode change Operation

Ada tiga macam mode dasar untuk kontrol proses pada instrument faceplate, yaitu:



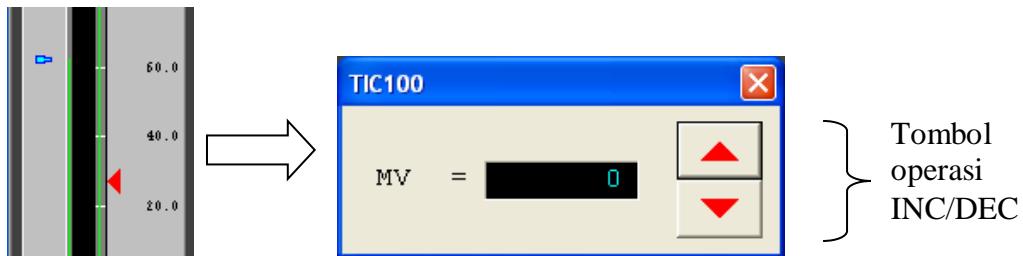
MAN : Manual

AUT ; Automatic

CAS : Cascade

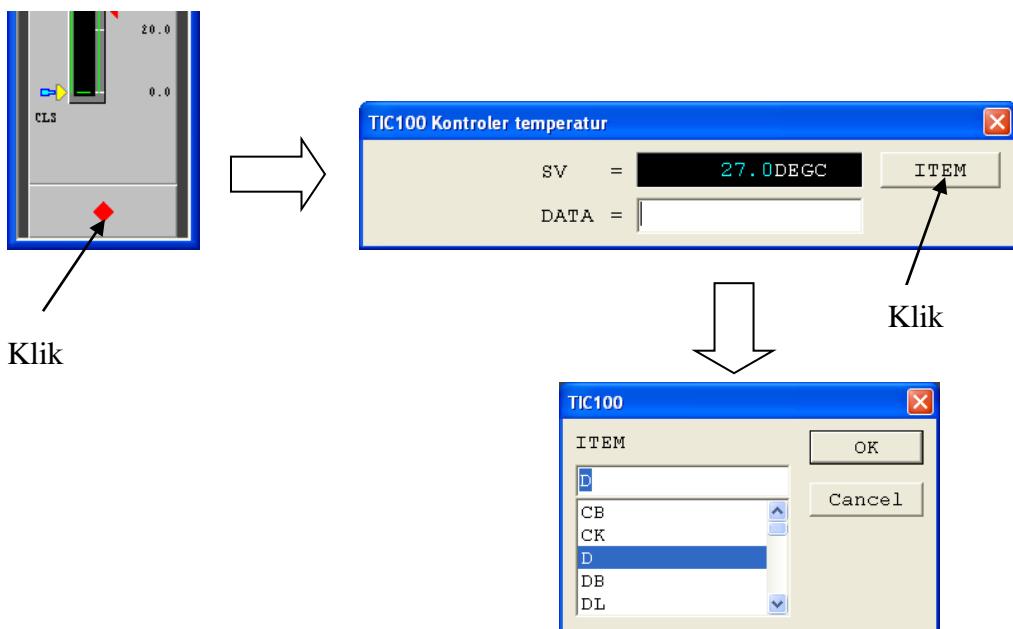
- Data INC/DEC Operation

Dengan mengklik kotak dialog operasi INC/DEC akan didapatkan suatu cara untuk mengganti nilai suatu parameter.

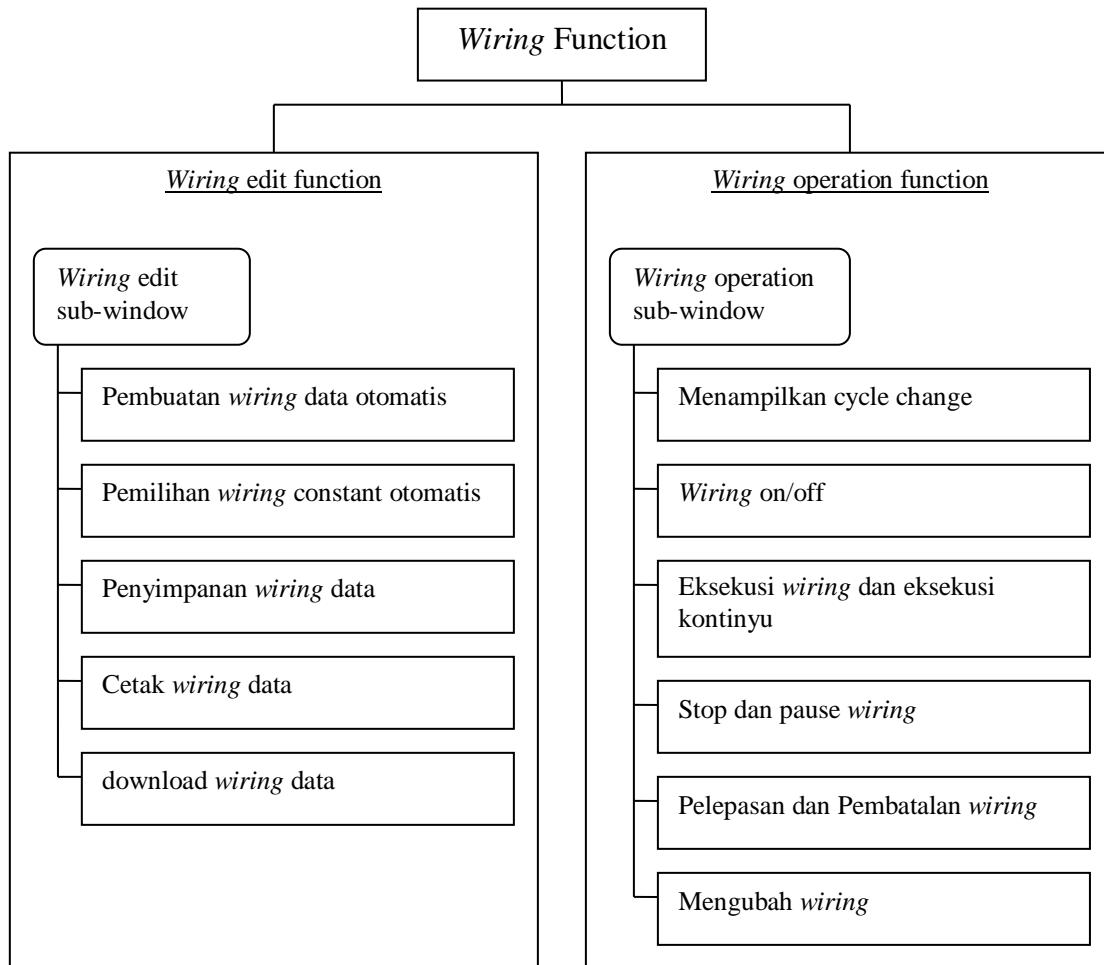


c. Data Entry Operation

Operasi ini berfungsi mengubah-ubah parameter apa yang akan digunakan. Prosedurnya adalah dengan mengklik tombol [CLICK] pada kotak dialog dan kemudian item data yang ingin digunakan dapat dipilih sesuai dengan proses yang dibuat.



## F. Wiring Function

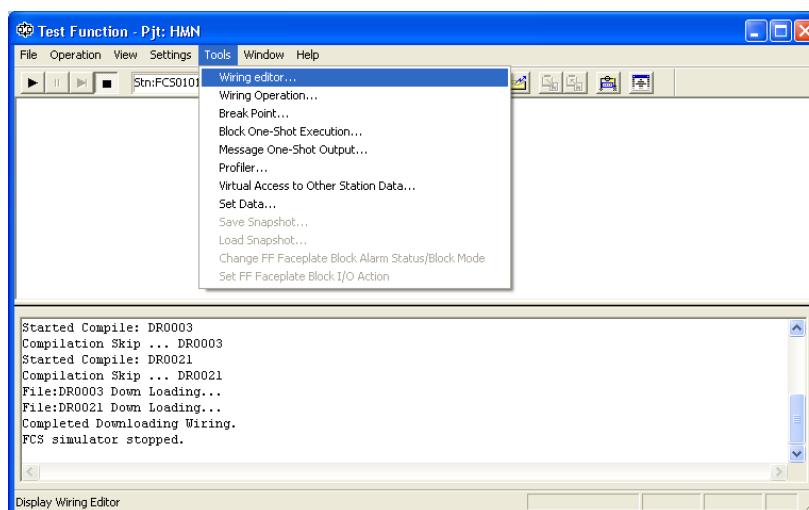


Gambar skematik jenis-jenis *wiring function*

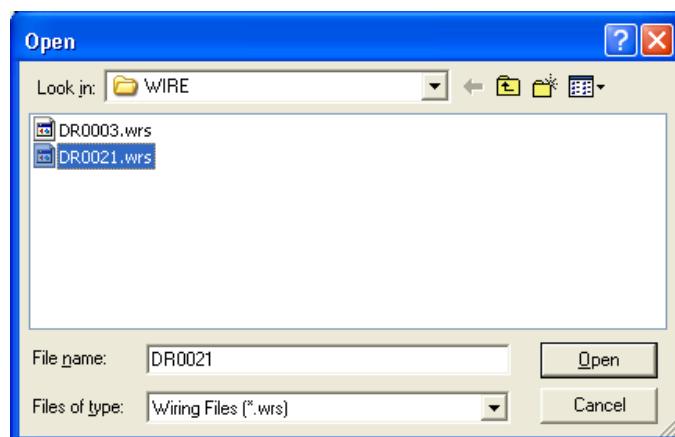
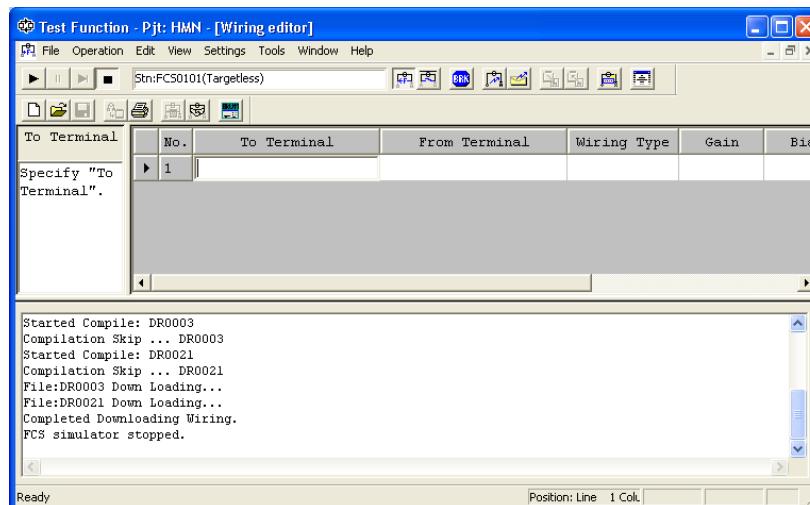
### F.1. Mengubah wiring data

Prosedur :

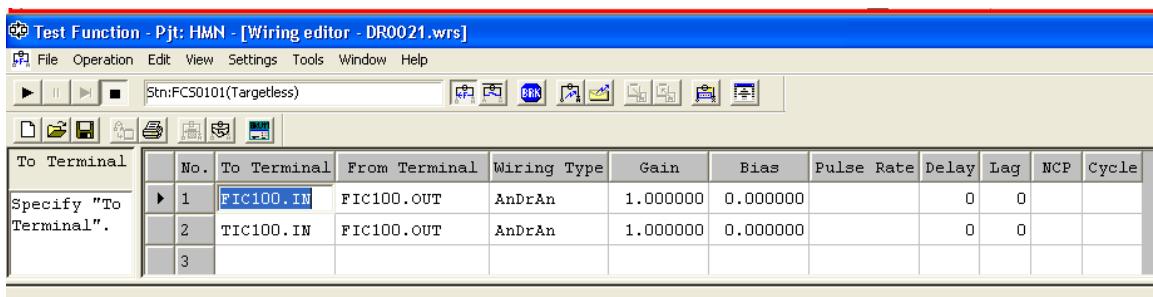
Dari window [Test Function], pilih [Tools] → [Wiring Editor]



Window [Wiring Editor] muncul seperti berikut:

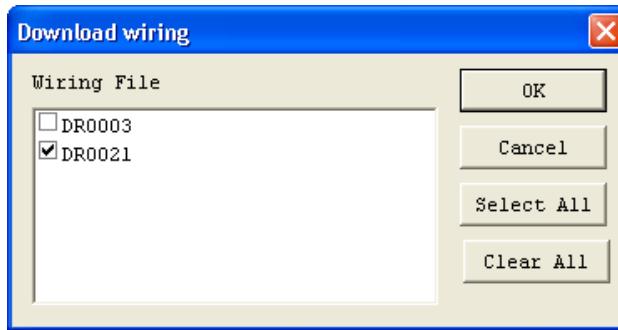


Buka file *wiring data* dengan [File] → [Open]



Pilih file yang akan dibuka, lalu klik [Open], misalnya file DR0021.wrs. setelah itu muncul seperti berikut:

Pastikan *test function* dalam keadaan *run*, lalu pilih [File] → [Download] untuk *download wiring*. Beri tanda centang *file* yang akan didownload, lalu klik [OK].



## No.

Menyatakan record no. dari file wiring untuk diubah.

## To Terminal

Menyatakan terminal koneksi tujuan dalam tag name.

## From Terminal

Menyatakan terminal koneksi asal dalam tag name.

## Wiring Type

Menyatakan tipe wiring, bila wiring source merupakan analog output dan *wiring destination* merupakan analog input, maka gunakan AnDrAn atau AnRvAn.

## Gain

Menyatakan *gain wiring* antara *terminal wiring* dan *simple wiring*, juga batas atas *time-series data* untuk *time-series data wiring*. Gain dapat diubah antara -999999.0 ~ 999999.00

## Bias

Menyatakan bias ketika *wiring* antar terminal dan *simple wiring*, juga batas bawah *time-series data* untuk *time series data wiring*. Bias dapat diubah antara -999999.0 ~ 999999.0

## Pulse Rate

*Pulse rate* dapat diubah antara 1 ~ 99999 HZ.

## Delay

Menyatakan dead time. *Delay* dapat diubah antara 1 ~ 99detik

## Lag

Menyatakan *lag* orde pertama. *Lag* dapat diubah antara 1~ 99 detik.

## NCP (Number of Continuous Wiring)

Menyatakan banyaknya *wiring* ketika *wiring* yang sama dinyatakan secara kontinu dari *current input terminal*.

## Cycle

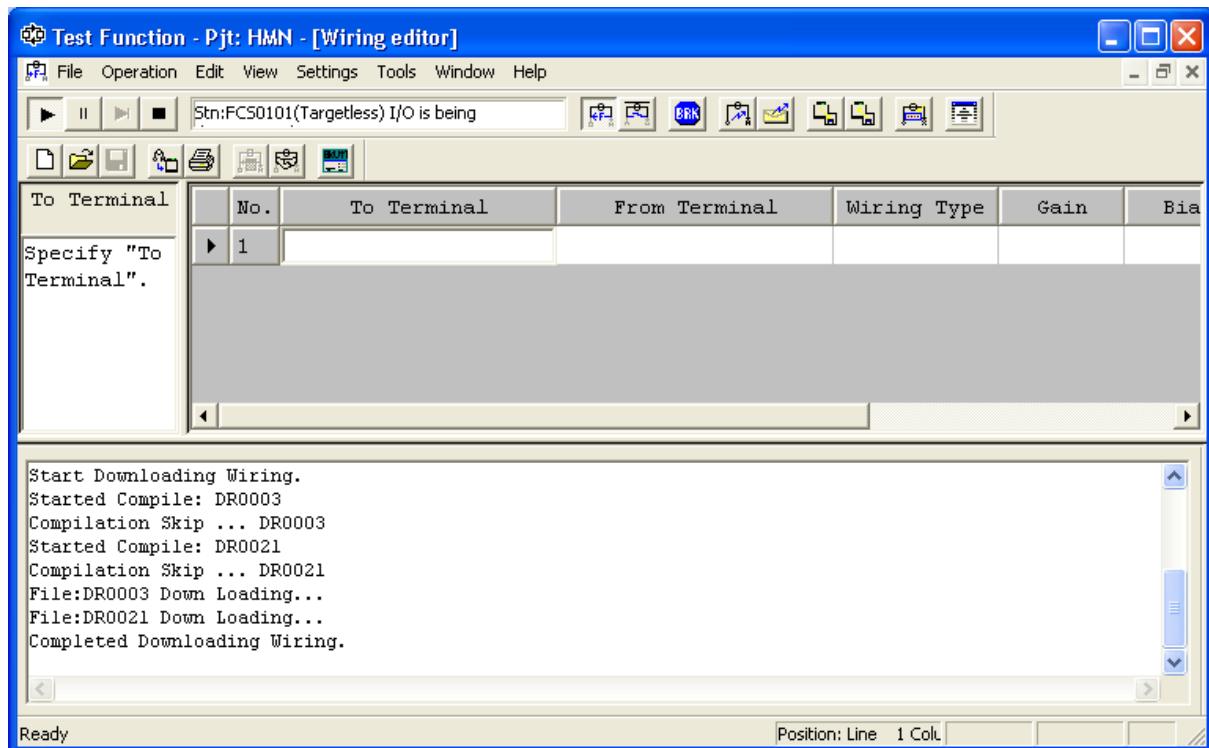
Menyatakan *time-series data wiring cycle time*. *Cycle* dapat diubah antara 1 ~99999 detik.

## 6.2.2 Wiring Operation Functions

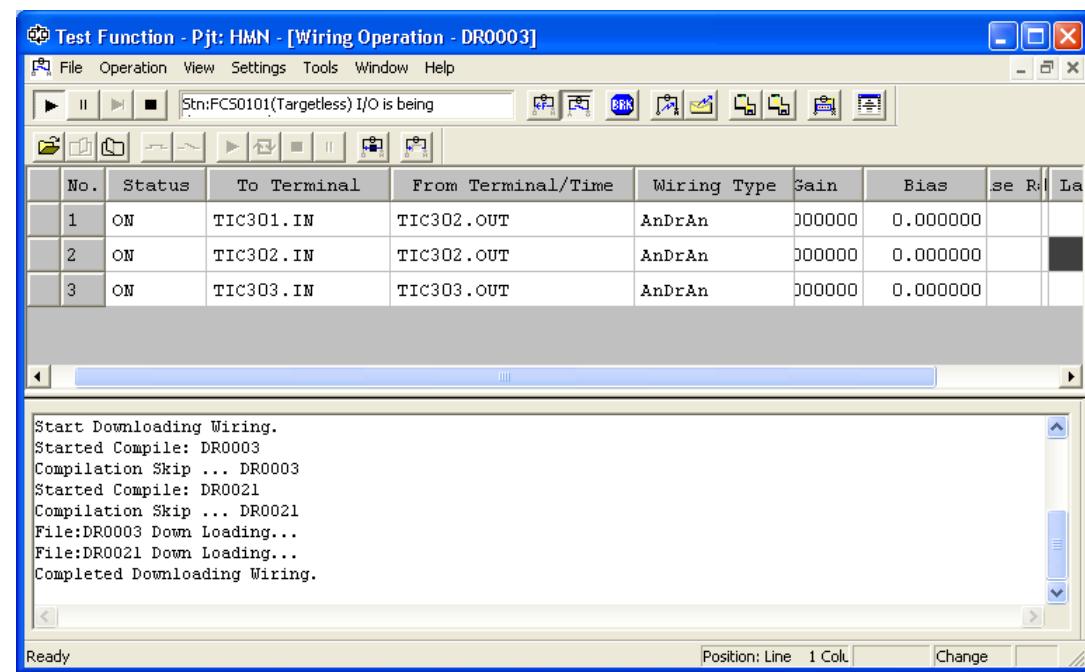
*Wiring operation functions* menampilkan status wiring FCS/FCS simulator yang terakhir diuji dan melakukan operasi yang berhubungan dengan *wiring* pada saat eksekusi tes.

Prosedur :

1. Pilih [Wiring Operation] dari [Tools] pada window [Test Function].



2. Setelah [Wiring Operation] window muncul, pastikan kolom [Status] pada setiap data dalam kondisi ON dan kolom [Lag] sudah terisi dengan yang diinginkan.

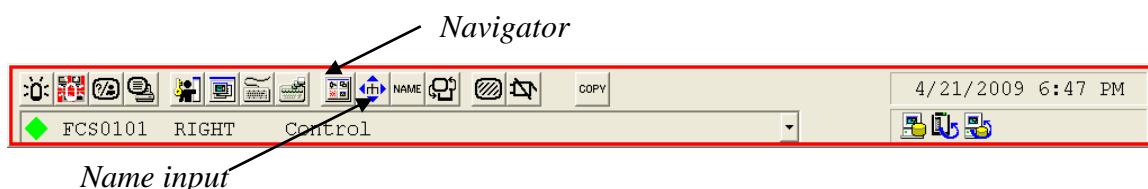


## G. Trend Window

*Trend window* menghimpun data proses yang berbeda-beda dan menampilkannya dalam bentuk perubahan deretan waktu pada suatu grafik. Perubahan deretan waktu untuk data yang telah dikumpulkan disebut sebagai *trend data*. Pada *trend window*, data dapat ditampilkan ketika berupan data sekarang dan data sebelumnya. Pada *trend window*, maksimum *trend data* yang dapat ditampilkan sebanyak delapan kanal.

### G.1. Prosedur untuk menjalankan trend window

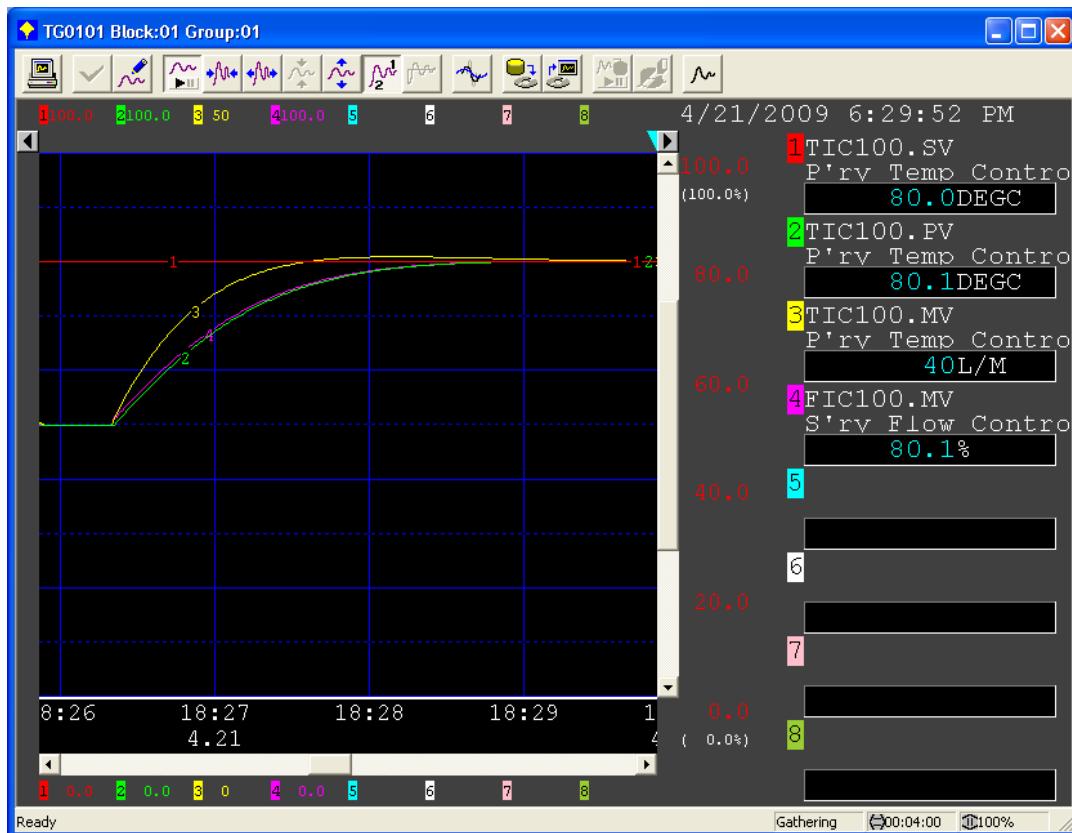
- Panggil *trend window* dengan mengetik NAME yang telah didefinisikan pada input window name dari *Navigator Window*. Misalkan namanya adalah TG0101.



The screenshot shows the "Navigator USER" window. On the left is a tree view with nodes for USER, SYSTEM, and PRODUCT. The main area is a table with columns: Name, Comments, and Type. The data is as follows:

Name	Comments	Type
CG0001		Control
GR0001		Graphic
OV0001		Overview
TG0101	Block:01 Group:01	Trend
TG0201	Block:02 Group:01	Trend
TG0301	Block:03 Group:01	Trend
TG0401	Block:04 Group:01	Trend
TG0501	Block:05 Group:01	Trend
TG0601	Block:06 Group:01	Trend
TG0701	Block:07 Group:01	Trend
TG0801	Block:08 Group:01	Trend
GR0002		Graphic
CG0003		Control

- Setelah itu akan muncul gambar sebagai berikut:



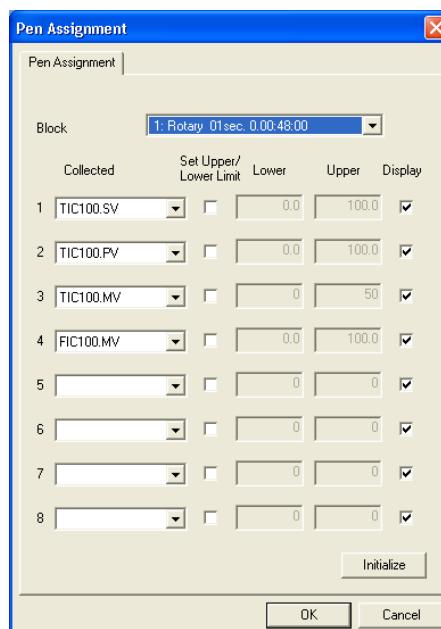
Berikut penjelasan tentang *toolbar* dari *trend window*:

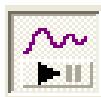


Tombol ini mengeluarkan gambar dari *trend window* yang ditampilkan terakhir kali.



Tombol ini digunakan untuk membuka kotak dialog dari *Trend Pen Assignment*





Ketika tombol ini ditekan, tampilan *trend* akan berhenti dan menekan sekali lagi tombol ini akan menampilkan *trend* kembali dari posisi sekarang.



Ketika tombol ini ditekan, *trend graph* dapat diperbesar ataupun diperkecil dalam arah horizontal( arah dari sumbu waktu) dengan waktu terakhir sebagai titik referensi.



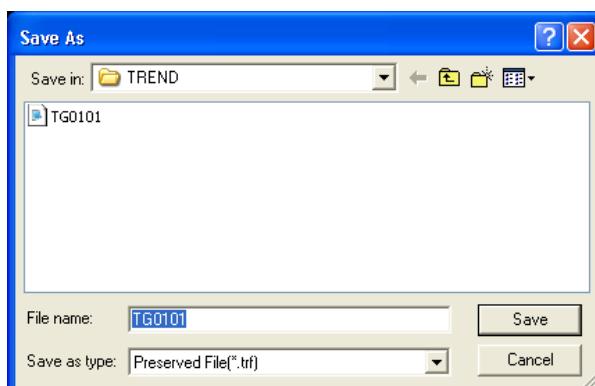
Ketika tombol ini ditekan, *pen number* akan ditampilkan pada *trend graph*.



Jika tombol ini ditekan akan mengembalikan seluruh data dan perubahan waktu kembali pada posisi *default*.



Tombol ini berfungsi untuk menyimpan *trend data* yang terakhir kali muncul dalam format .tf pada harddisk dengan posisi C:\CS3000\his\save\TREND



Ketika tombol ini ditekan, *trend data* yang telah disimpan akan ditampilkan sebagai grafik



Ketika tombol ini ditekan akan mengubah tampilan ke data waktu real.

## H. Test Function

*Test function* adalah suatu cara untuk mengecek program yang telah dibuat.

Jenis *test function* :

1. *Virtual test*

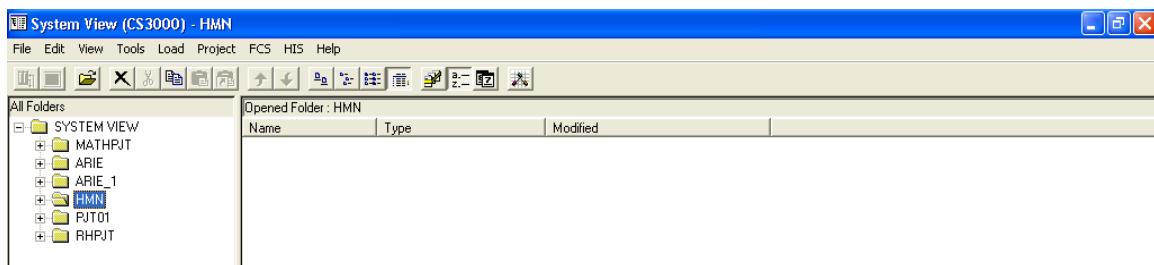
*Virtual test* menggunakan simulator FCS. Fungsi simulator ini mensimulasikan fungsi dan operasi pada FCS itu sendiri dalam suatu HIS tertentu.

2. *Target test*

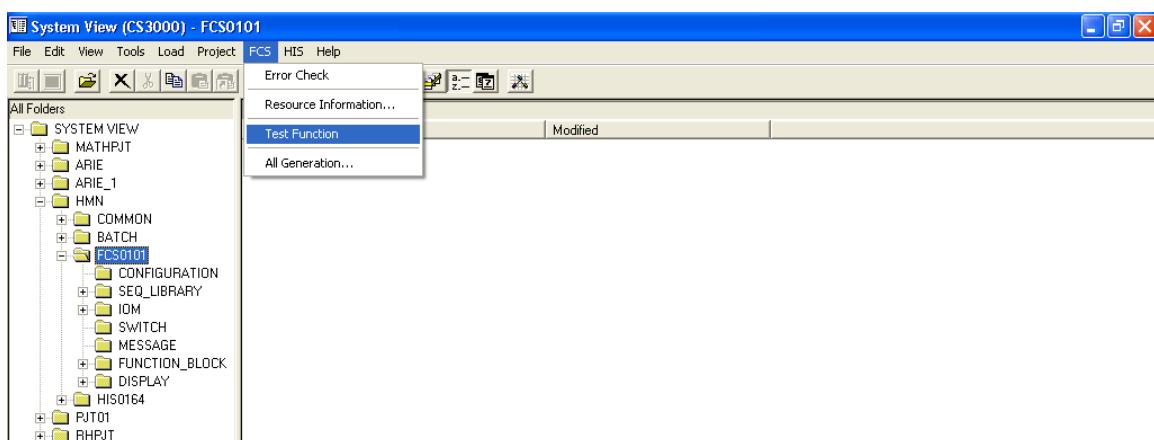
*Target test* menggunakan FCS sebenarnya. Jika tidak ada modul dan *test device* I/O, maka input dan output FCS dapat disimulasikan menggunakan I/O *disconnect* dan *wiring function*.

### H.1. Prosedur *Test Function*

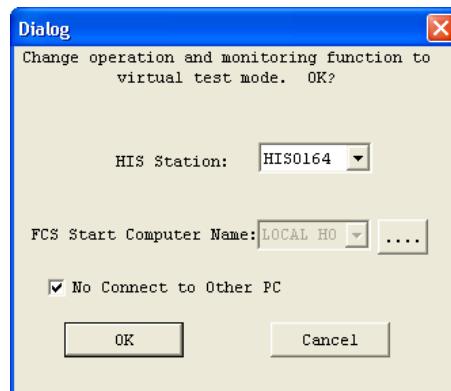
1. Pilihlah *target test* FCS dengan *System view*



2. Mulai *test function* melalui *System view*



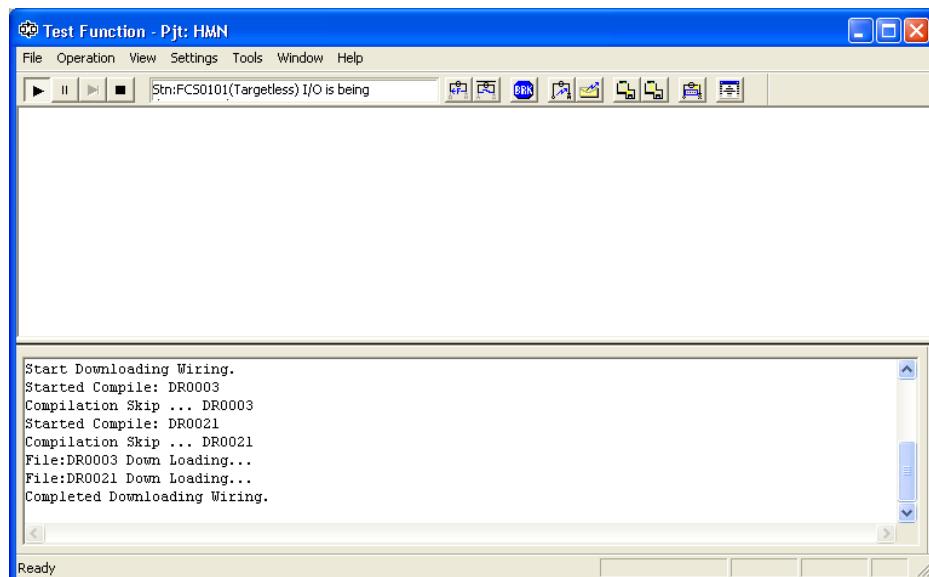
3. Pastikan pada kotak dialog, posisi HIS Station sesuai dengan proyek yang telah dibuat dan tidak ada koneksi dengan komputer lain.



4. Setelah *Test Function* mulai, akan terdapat beberapa operasi berikut:

- Fungsi operasi dan monitoring akan masuk ke mode *virtual test*
- FCS simulator mulai berjalan
- Pembuatan data *wiring* secara otomatis dan akan segera didownload ke FCS

Tunggu sampai pesan “*completed downloading wiring*” muncul.



Window operasi dan monitoring *virtual Test Function* HIS siap digunakan.

Operasi-operasi yang ada di *test function* antara lain:

1. FCS *operation* (*run, stop, pause*)

- FCS *Start* (*run*)

Fungsi ini memulai dan mengatur FCS simulator pada keadaan eksekusi kontrol.

- FCS *Pause*

Fungsi ini mengubah FCS simulator dari keadaan eksekusi kontrol atau eksekusi sekuen ke keadaan *pause*.

- *Stop*

Fungsi ini mengubah dari keadaan eksekusi kontrol, eksekusi sekuen, atau pause ke dalam kondisi berhenti.

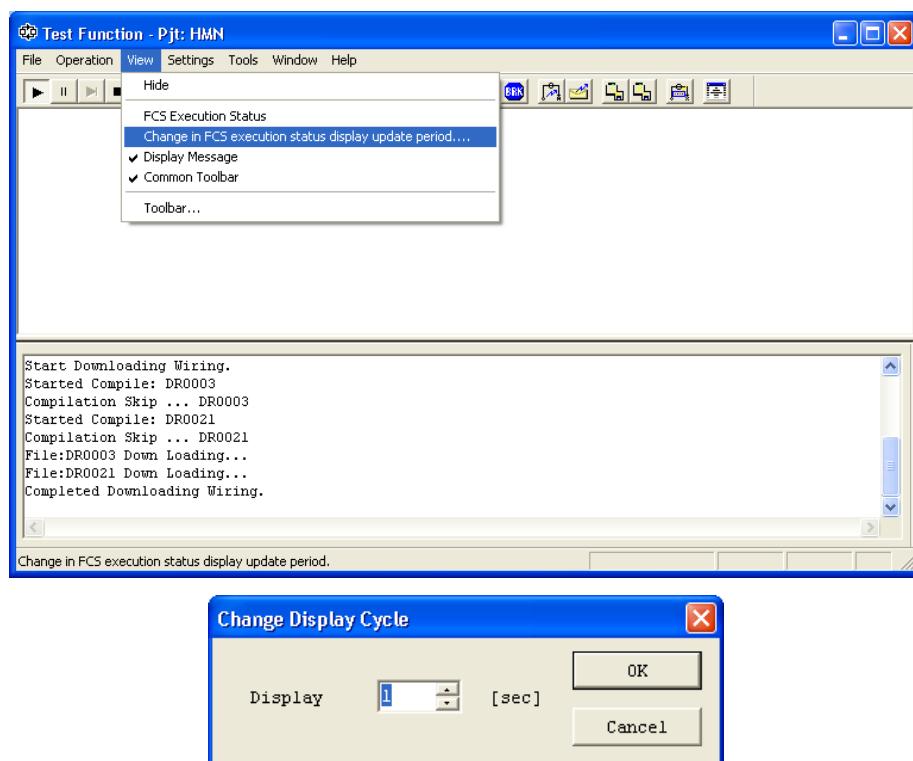
### 2. Step Execution

Fungsi *step execution* dapat dilakukan pada FCS simulator jika sudah dalam kondisi pause.

Fungsi ini bertujuan untuk menentukan berapa kali suatu kontrol dijalankan (1-100).

Langkah-langkahnya adalah:

- Dari window [Test Function], pilih [Settings] → [Change Number of Steps to be Executed]



### 3. Wiring Operation

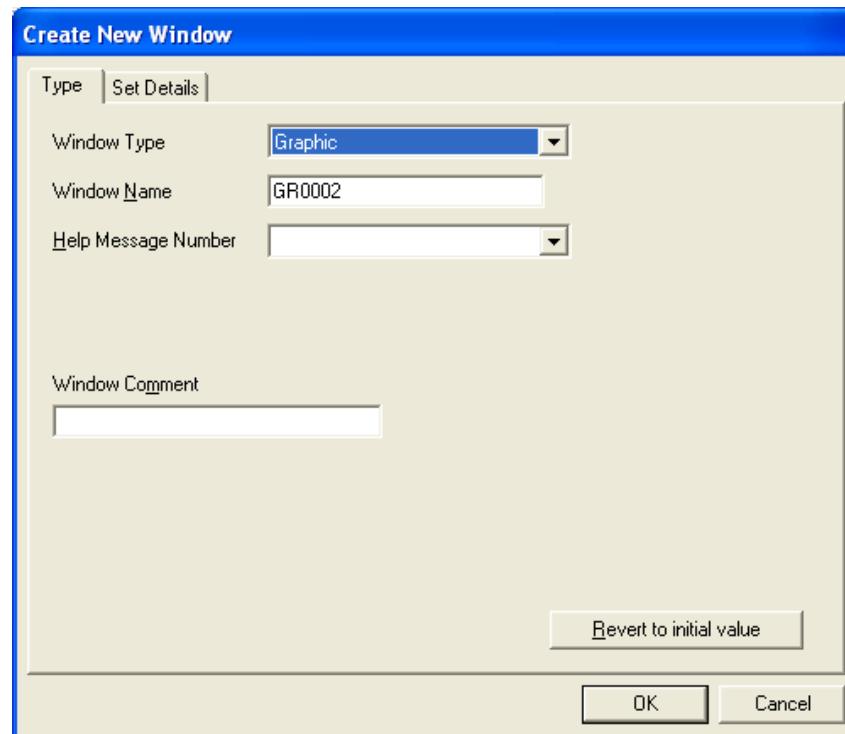
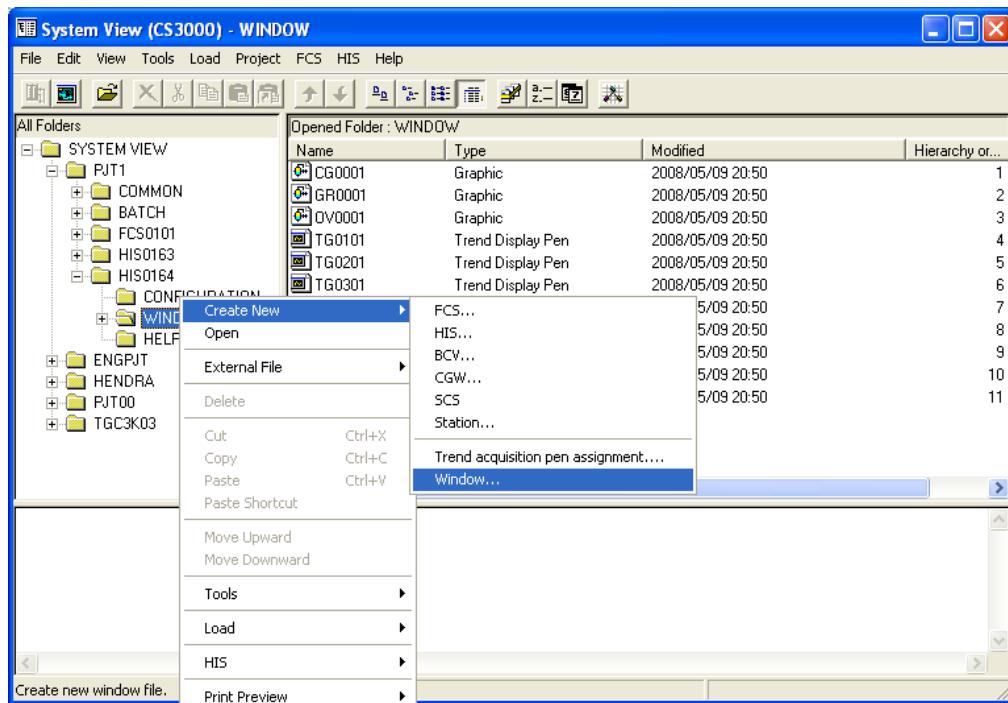
Fungsi yang melakukan *virtual wiring* pada proses I/O terminal dari I/O modul disebut *wiring function*. Fungsi-fungsi ini memperbolehkan untuk menguji operasi kontrol FCS/FCS simulator tanpa menggunakan I/O unit.

## I. Graphic Builder

Graphics Builder digunakan untuk membuat dan meng-edit *graphic windows* dalam pengoperasian dan monitoring.

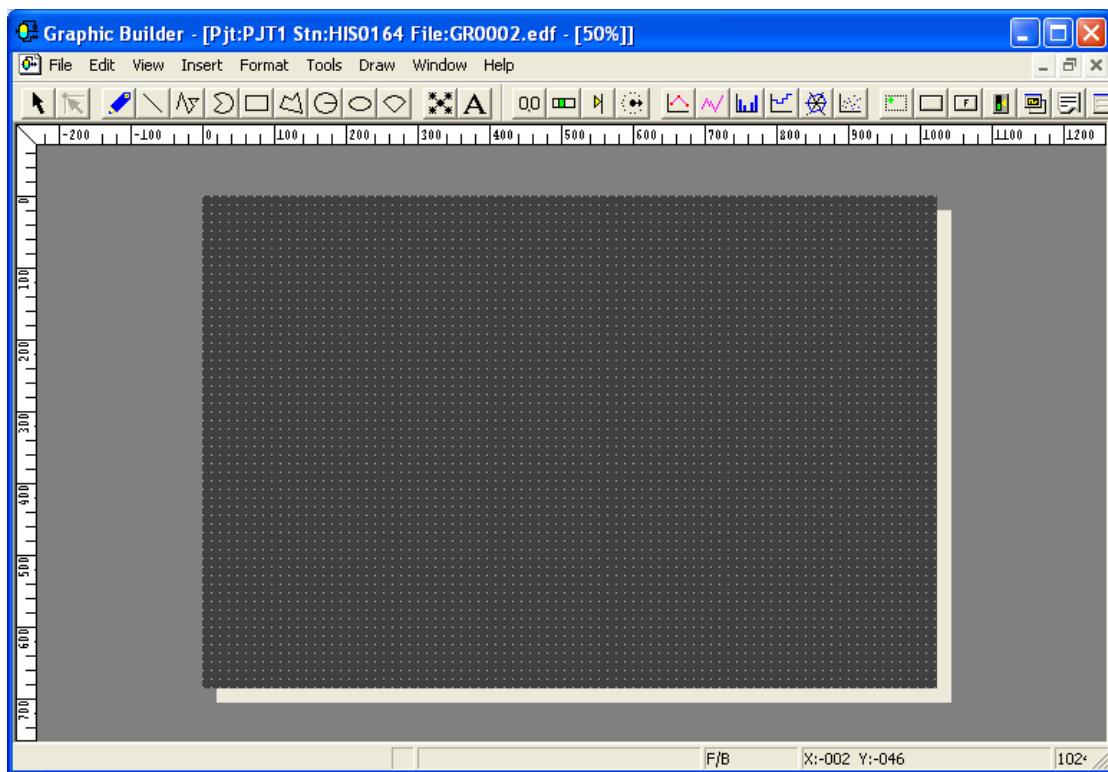
### A. Lingkungan *Graphics Builder*

Dari dalam *system view*, pilih *project* → [HIS0164], klik kanan [WINDOW] → [Create New] → [Window].



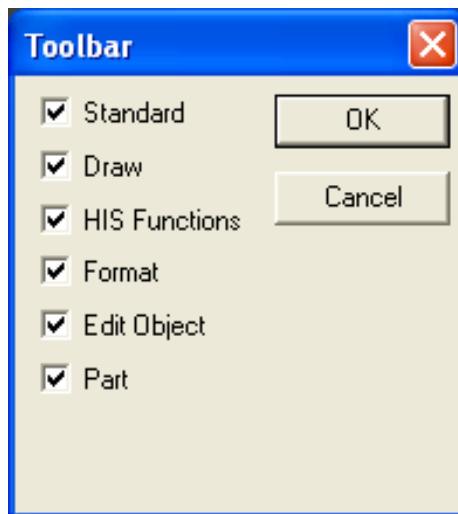
## B. Graphics

Dari system view, pilih project, [HIS0164], [WINDOW] → klik dua kali [GR0001]



### C. Toolbar

*Graphic Builder* menyediakan tool khusus untuk membuat dan meng-edit objek grafis yang menyusun *Graphic Window*.



### D. Toolbar Draw

Tabel Daftar tool dari toolbar *Draw*

Tombol	Tool	Fungsi
	<i>Select Mode</i>	Mengatur mode yang mana objek dapat dipilih
	<i>Point Correction</i>	Memasukkan mode pengkoreksian titik
	<i>Pen</i>	Memilih tool pen

	<i>Straight Line</i>	Menggambar garis lurus
	<i>Polyline</i>	Menggambar <i>polyline</i>
	<i>Arc</i>	Menggambar busur
	<i>Rectangle</i>	Menggambar persegi
	<i>Fill Area</i>	Menggambar area tertutup
	<i>Circle</i>	Menggambar lingkaran dengan radius yang terspesifikasi
	<i>Ellipse</i>	Menggambar elips
	<i>Sector</i>	Menggambar sector
	<i>Marker</i>	Menggambar <i>marker</i>
	<i>Text</i>	Menulis teks

## E. Toolbar HIS FUNCTION

Tabel daftar tool dari toolbar *HIS Function*

Tombol	Tool	Fungsi
	<i>Process Data-Character</i>	Menggambar suatu display objek dalam bentuk data karakter
	<i>Process Data-Bar</i>	Menggambar data bar persegi suatu objek
	<i>Process Data-Arrow</i>	Menggambar data bar objek berbentuk panah
	<i>Process Data-Circle</i>	Menggambar data bar objek berbentuk lingkaran
	<i>Line-Segment Graph</i>	Menggambar objek dengan segmen garis
	<i>User-defined Line-segment Graph Object</i>	Menggambar objek grafis berbentuk segmen garis yang dapat didefinisikan
	<i>Bar Graph</i>	Menggambar objek grafis bar
	<i>Step Graph</i>	Menggambar objek grafis berbentuk step
	<i>Radar Chart</i>	Menggambar objek grafis berbentuk radar
	<i>Two-dimensional Graph</i>	Menggambar objek grafis dua dimensi
	<i>Touch Target</i>	Menggambar objek target sentuh

## F. Properties Window objek

### 1). General Tab

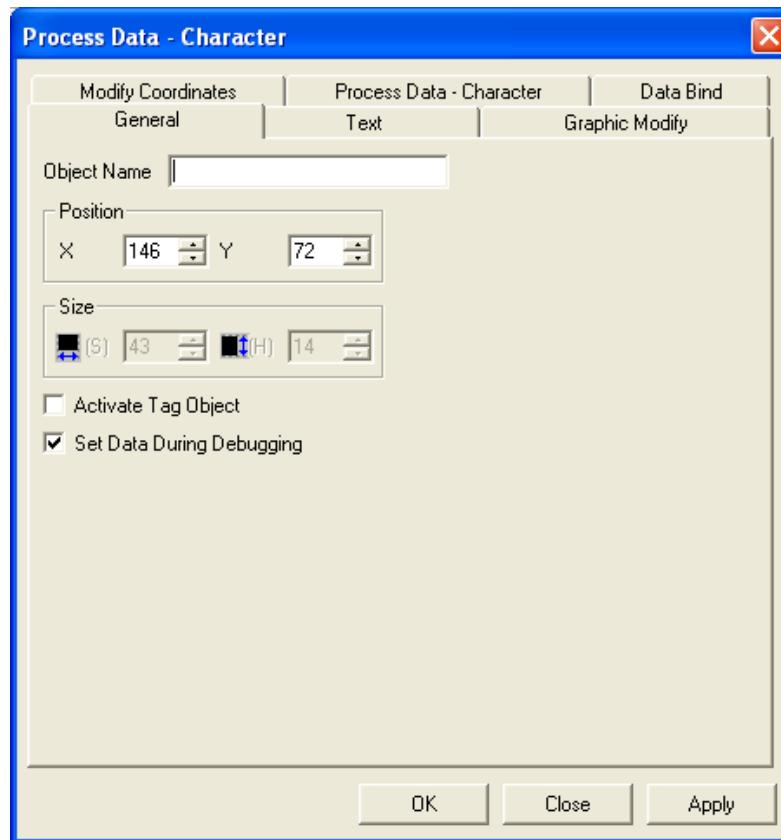
Tab ini digunakan untuk mengatur atribut yang umum pada semua objek grafik yang dibuat. Tab mengatur nama objek, posisi, dan ukuran serta apakah objek dapat digunakan sebagai tab object dan apakah pengaturan data mungkin dilakukan selama proses *debug*.

### Object Name

Ini mengatur nama objek. Teks yang dapat dimasukkan sebagai nama objek hingga 15 karakter.

### Position

Mengatur koordinat suatu objek.



### Object Size

Mengatur panjang dan lebar objek.

### Activate Tag Object

Salah satu metode untuk memanggil *window* dari *graphic window* adalah dengan menekan tombol *window calling* setelah memilih objek. Bagaimanapun *tag objects* dibatasi terhadap objek yang mengikutinya ke *tag name* yang dituju.

### Set Data During Debugging

Tampilan dan aksi objek grafis dapat dicek dengan menggunakan fungsi *debug* dari *graphic window*. Kehadiran atau tidaknya simulasi *data setting* selama proses *debug* dapat diatur untuk objek grafis.

### 2). Text Tab

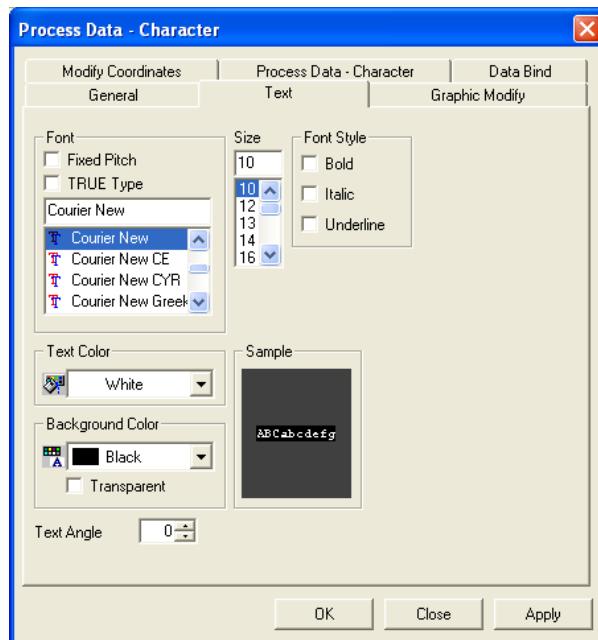
Tab ini digunakan untuk mengatur format teks. Warna teks dan latar belakangnya dapat diatur.

### Fixed Pitch

Jika Fixed Pitch dicontreng, daftar font akan diseleksi untuk menampilkan font fixed-pitch saja.

### True Type

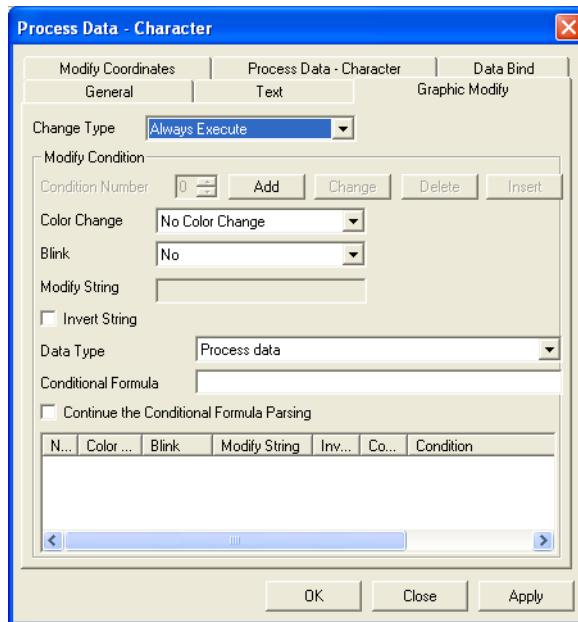
Jika True Type dicontreng, maka daftar font akan diseleksi untuk menampilkan font tipe True saja.



### 3). Graphic Modify Tab

Tab ini digunakan untuk mengatur kondisi atribut seperti warna, bentuk, dan status pencahayaan objek grafik.

Graphic modify adalah fungsi yang mengubah warna objek, status kedip, dan pattern berdasarkan data proses dan formula kondisional.



#### Change Type

Digunakan untuk mengatur ketika melakukan test pada kondisi graphic modifier. Waktu eksekusi untuk tes kondisional dapat dipilih dari [Execute Always], atau [Execute First Time Only].

#### Always Execute



Selama *conditional formula* benar, aksi perubahan seperti perubahan warna, dan kedipan dilakukan setiap waktu.

### **Execute First Time Only**

Jika formula kondisional benar, maka ketika Graphic window dipanggil, aksi perubahan seperti perubahan warna atau kedipan tampil sekali saja.

### **Conditional Formula**

Digunakan untuk mengatur graphic modifier conditional formula untuk digunakan sebagai criteria tes yang dilakukan ketika merubah grafik.

Berikut ini formula kondisional yang dapat digunakan :

#### **Data proses**

Contoh : PIC300.SV

#### **Konstanta bilangan bulat**

Contoh : 100

#### **Bilangan real**

Contoh : 0,5

#### **Konstanta heksadesimal**

Contoh 0x0001

#### **Konstanta teks**

Contoh : MAN, AUT

### **Continue the Conditional Formula Parsing**

Digunakan untuk mengatur apakah pengujian berkelanjutan dari formula kondisional pengubah grafik atau tidak. Dalam pengaturan secara default, pengujian dilakukan di dalam sekuen, memulai formula kondisional pengubah grafik dari kondisi pertama. Jika kondisi tersebut sesuai, akan dilakukan perubahan aksi dan pengujian formula kondisional pengubah grafik dinyatakan sukses.

Ketika pengelanjutan pengujian dipilih, pengujian formula kondisional pengubah grafik dilanjutkan setelah kondisi sesuai. Dalam kasus ini, semua perubahan aksi dengan kondisi sesuai telah dilakukan.

### **Modify Condition**

#### ➤ **Condition Number**

Urutan tampilan dari kondisi pengubah grafik yang sedang ditampilkan.

#### ➤ **Tombol [Add]**

Digunakan untuk penambahan kondisi pengubah grafik yang baru. Tombol [Add] tidak dapat digunakan setelah terdapat 8 formula kondisional.

#### ➤ **Tombol [Change]**

Digunakan untuk menggambarkan konten perubahan kondisi pengubah grafik ke data saat ini.

#### ➤ **Tombol [Delete]**

Digunakan untuk menghapus kondisi pengubah grafik yang sedang ditampilkan. Jika terdapat beberapa kondisi yang bebas, maka sisanya secara otomatis akan mengisi urutan dari kondisi yang telah dihapus.

## Change Action

- **Color Change**

- Normal Change Color

Digunakan untuk menampilkan objek dalam warna khusus. Warna dapat dipilih dari palet warna yang ada pada [Change Color].

- Change Alarm-Specific Color

Digunakan untuk menampilkan objek dalam warna alarm dari blok fungsi. Untuk mengaktifkannya dengan cara mengatur *tag name* blok fungsi ke dalam Conditional Formula.

- Overview Color

Digunakan untuk menampilkan objek dalam warna alarm yang mana subjek untuk mengawasi dispesifikasi dengan *tag name* atau *window name*. *Tag name* atau *window name* diatur dalam *Conditional Formula*.

- **Blink**

- Yes

Digunakan untuk mengaktifkan objek menjadi blink.

- Alarm Specific Blinking

Digunakan untuk membuat suatu objek menjadi blink sesuai status alarm diagram instrumen. Menggunakan kunci pernyataan dapat menyatakan pesan alarm.

- Screen Blinking

Digunakan untuk membuat blink objek. Jika option ini dipilih, maka objek akan berhenti saat *acknowledgment operation* dijalankan. Jika status Conditional Formula berubah setelah *acknowledgment operation*, proses blink akan berlanjut.

- Overview Blinking

*Tag name* atau *window name* dapat didefinisikan dalam formula untuk menggambarkan kondisi pengujian proses blinking. Penggambaran tersebut akan melakukan proses blink seperti *Tag name* atau *window name* yang didefinisikan oleh formula tersebut.

Ketika *tag name* didefinisikan dalam suatu formula, objek grafik menunjukkan proses blink ON/OFF mengikuti proses blink ON/OFF alarm dari *instrument faceplate* yang memiliki *tag name* tersebut.

- No

Untuk Meng-non-aktifkan blink suatu objek.

- **Modify String**

Hanya berlaku untuk objek teks. Aksi ini mengubah suatu string karakter menjadi string karakter yang lain. Hingga sebanyak 16 karakter *alphanumeric* untuk memodifikasi string dapat dimasukkan.

- **Invert String**

Digunakan untuk mengatur string karakter untuk ditampilkan dalam video cadangan.

- **Data Type**

Data type meliputi :

- Process Data

Process Data dapat digunakan dalam formula pengujian kondisional untuk proses modifikasi grafik. Formula yang menggunakan dalam process data adalah dalam format sintaks “*TagName*.*DataItem*”. Operator unary, operator pembanding dan tanda kurung besar dapat digunakan untuk conditional formula.

▪ Operator unary yang dapat diaplikasikan adalah sebagai berikut :

+, -, \*, /, & (simbol AND), l (simbol OR), % (remainder)

▪ Operator pembanding yang dapat diaplikasikan adalah sebagai berikut :

=, <> (tidak sama dengan), >, <, >=, <=, dan (conditional formula AND), atau (conditional formula OR).

- Contoh notasi pengisian conditional formula adalah sebagai berikut:

FIC100.PV>50.0

FIC100.PV+ FIC300.PV<= FIC400.PV

FIC100.PV>50.0 AND FIC200.PV<20.0

FIC100.ALRM = "HI" or "HH"

%CI0001.PV & 0x00FF <> 0

- Recipe Data Unit Name Specification

Ketika *recipe data* (spesifikasi nama unit) dipilih data type untuk proses modifikasi objek grafik, penghitungan grafik dapat digunakan. *Syntax*-nya adalah sebagai berikut.

NamaUnit.NamaBlokPerintah.DataItem[XY]

(X, Y adalah data array)

- Recipe Data Batch ID Specification

Ketika *recipe data* (spesifikasi Batch ID) dipilih data type untuk proses modifikasi objek grafik, penghitungan grafik dapat digunakan. *Syntax*-nya adalah sebagai berikut.

BatchID.NamaBlokPerintah.DataItem[X,Y]

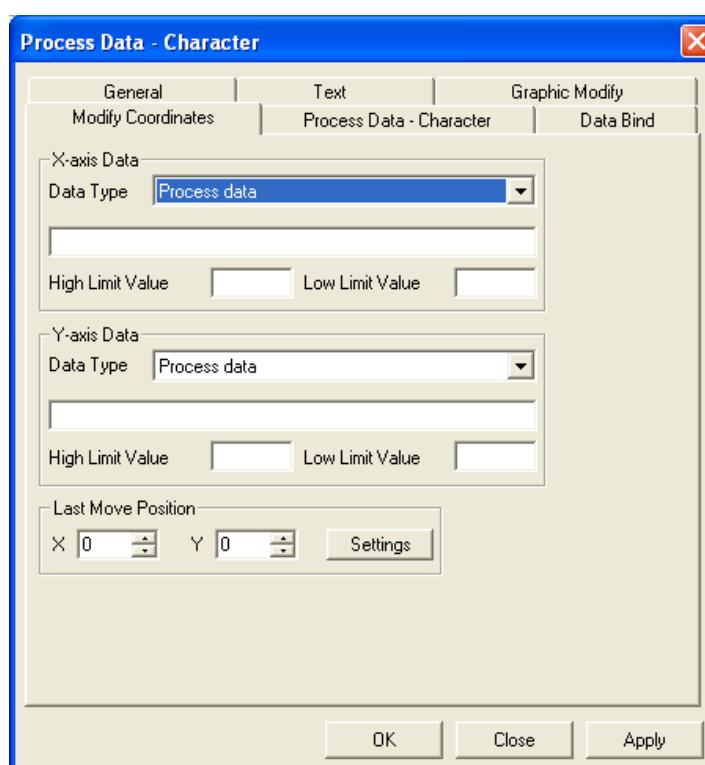
(X, Y adalah data array)

#### 4). Modify Coordinates Tab

Tab ini digunakan untuk mengatur perubahan posisi dari objek grafik. Modifikator koordinat berfungsi untuk menampilkan pergerakan posisi dari objek grafik mengikuti perubahan nilai data proses.

#### Data Koordinat dari Modifikator Koordinat

Data proses dan conditional formula dapat diatur untuk data koordinat. Pada penambahan, batas atas dan bawah koordinat referensi juga dapat digunakan untuk mengubah tiap koordinat.



Sedikitnya data X-axis atau Y-axis harus diatur. Objek grafik membuat gerakan 2 dimensi ketika kedua axis telah diatur, dan gerakan 1 dimensi saat hanya salah satu data koordinat yang diatur. Berikut ini dapat digunakan untuk mengisi data koordinat.

- Process data  
Contoh : PIC300.SV
- Integer constant  
Contoh : 100, 0
- Real number constant  
Contoh : 50.0, 0.50
- Hexadecimal constant  
Contoh : 0x0001
- Recipe data (Spesifikasi Nama Unit)  
Contoh : UNIT01.SYSRCM.STATUS
- Recipe data (Spesifikasi Batch ID)  
Contoh : 01-0041.SYSRCM.STATUS

#### 5). Process Data-Character Tab

Tab ini digunakan untuk mengatur data proses untuk ditampilkan sebagai nilai numerik atau string karakter. Tab ini juga digunakan untuk mengatur tipe data, format data tampilan dan data tampilan untuk menampilkan data proses atau data array sebagai nilai numerik atau string karakter.

#### Display Format

##### Type

Digunakan untuk mengatur format, jumlah digit dalam string dan titik desimal data yang ditampilkan.

Tabel berikut menunjukkan jumlah digit dalam integer dan titik desimal untuk masing-masing format.

Format Data Tampilan	Jumlah digit dalam Integer	Jumlah digit dalam titik desimal
Numeric Value	1 hingga 16	0 hingga 16
Percentage	1 hingga 16	0 hingga 16
Character String	1 hingga 16	-
Hexadecimal	1 hingga 16	-
Date	1 hingga 16	-
Time	1 hingga 99	-
Tag List	1 hingga 99	-

##### Jumlah digit untuk integer

Dari digit 1 hingga 99 dapat diatur untuk jumlah digit. Bagaimanapun item ini tidak berlaku ketika *tag list* dipilih.

##### Jumlah digit setelah titik desimal

Pengaturan ini berlaku saat *numeric value* atau *percent* dipilih sebagai format tampilan.

##### Specify Position – Jastifikasi Tampilan Data

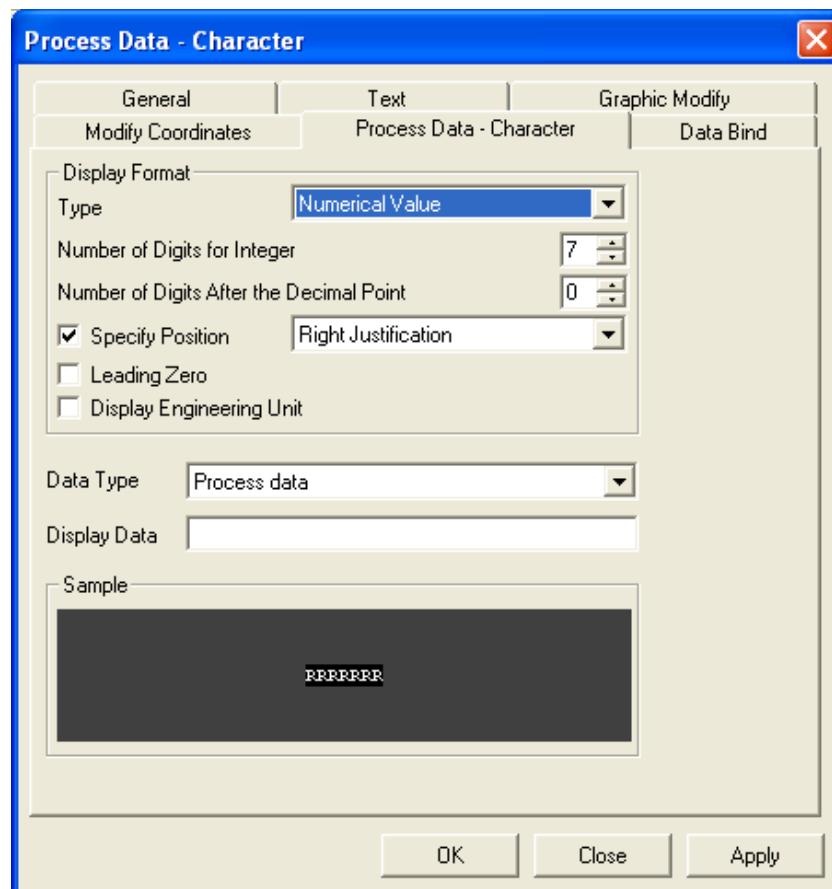
Posisi dasar dari data yang ditampilkan. Ada beberapa pilihan diantaranya Right (kanan), Left (kiri), atau center (Tengah).

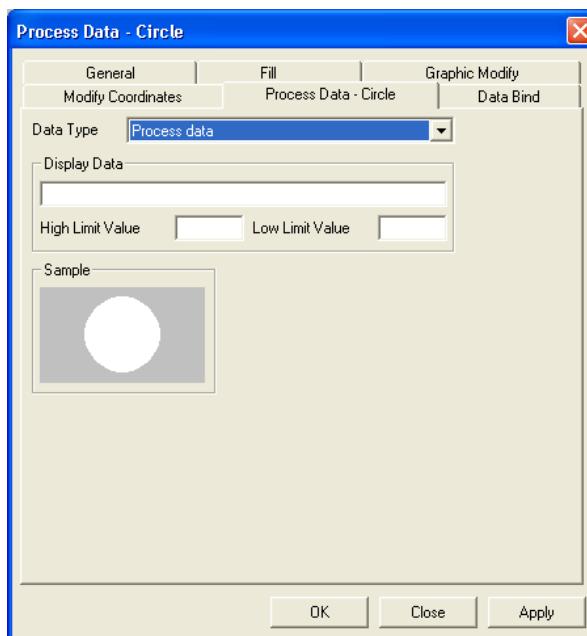
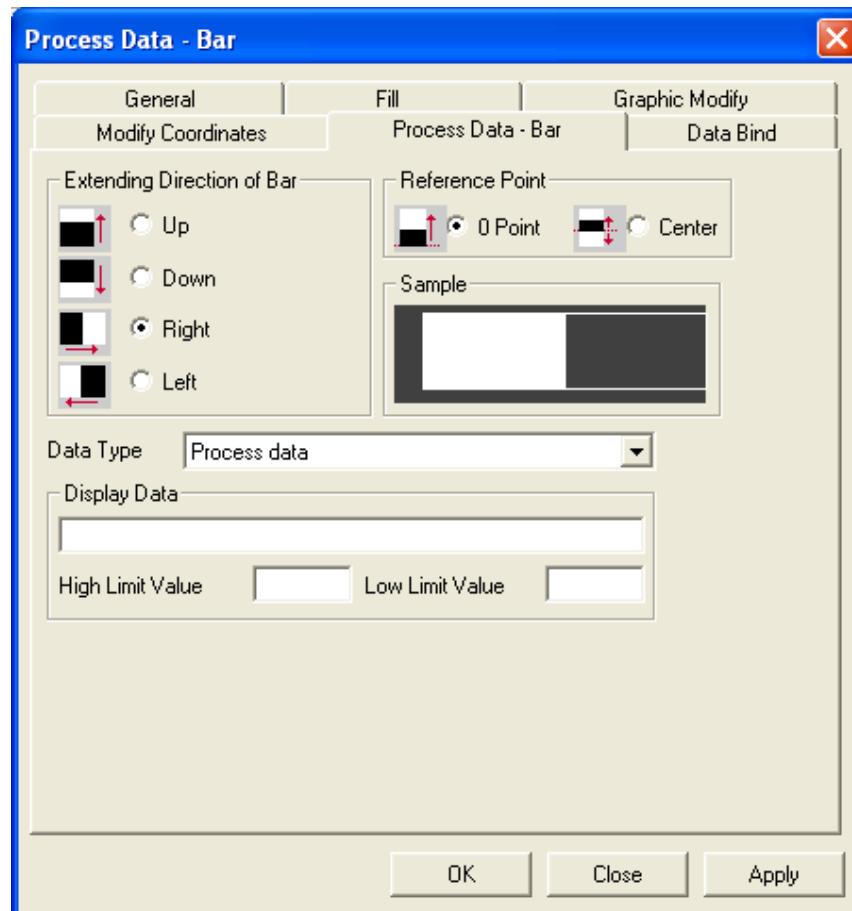
### Leading Zero

Ketika digit data kurang dari spesifikasi jumlah digit, *leading zero* dimungkinkan untuk ditempatkan sebelum jumlah yang valid. *Leading zero* hanya dapat dispesifikasikan untuk data dengan tipe format [Number], [Percent], atau [Hexadecimal].

### Display Engineering Unit

Digunakan untuk menampilkan atau menyembunyikan simbol dari unit *engineering*.





### Data Type

Digunakan untuk mengatur tipe data yang akan ditampilkan.

- Process data

Berikut ini contoh pengaturan data proses.

Untuk menampilkan nilai setpoint (SV) dari TIC300, dengan "TIC300.SV"

Untuk menampilkan status alarm TIG100, dengan "TIC100.ALRM"

Untuk menampilkan teks *pushbutton*, dengan “SW100.SWLB[2]”

Untuk menampilkan data dalam array, dengan “FIC100.IJ[01.02]”

- Recipe data (spesifikasi nama unit)
- Recipe data (spesifikasi *Batch ID*)
- Tag Comment

Pengaturan format ketika [Tag Comment] dipilih sebagai tipe data adalah dengan “Tag name”

- Graphic generic name

*Syntax* pengaturan data saat [Graphic generic name] dipilih sebagai tipe data yaitu dengan “Nama umum grafik”. *Syntax* yang sama dapat digunakan pada formula perhitungan.

## Display Data

Digunakan untuk mengatur data yang ditampilkan dalam bentuk data karakter.

- Process data  
Contoh : PIC300.SV
- Integer constant  
Contoh : 100, 0
- Real number constant  
Contoh : 50.0, 0.50
- Hexadecimal constant  
Contoh : 0x0001
- Text Constant  
Contoh : MAN, AUT
- Recipe data (Batch ID)  
Contoh : 01-0041.SYSRCM.RECIPE

## 6). Data Bind Tab

Tab ini digunakan untuk mengikat varian yang berbeda menjadi nama umum grafik.

## Set No., Set Name, Comment

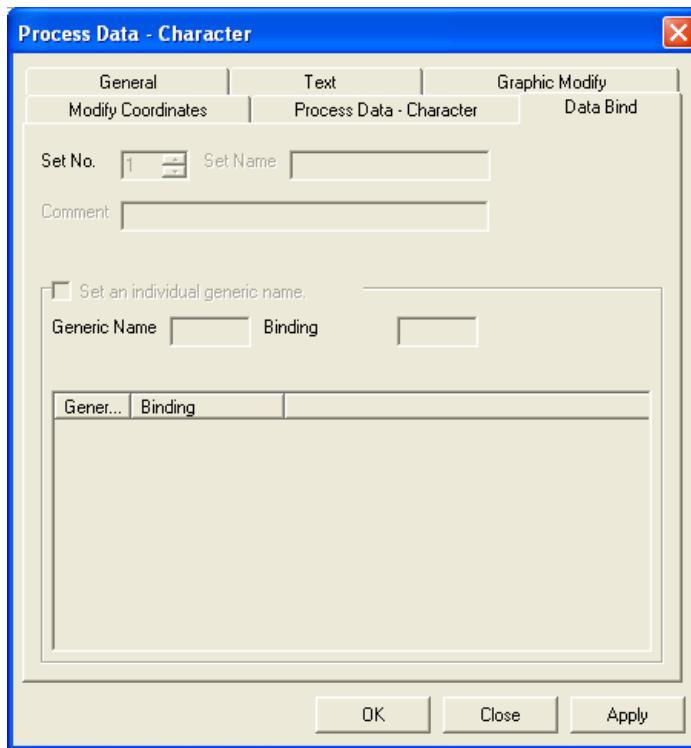
Digunakan untuk mengatur nama, mengatur nomor dan komentar dari pengaturan nama umum grafik. Pengaturan nomor dapat dilakukan pada *range* 1 hingga 200. Untuk mengatur nama, dapat dilakukan hingga 16 karakter huruf. Dan untuk mengatur komentar, dapat digunakan hingga 30 huruf atau 15 karakter *double-byte*.

## Set an individual generic name

Centang item ini, nama umum grafik dapat didefinisikan. Dan ketika tidak dicentang, proses pengikatan (*Binding*) tidak dapat dilakukan dan nama umum juga tidak dapat dirubah.

## Generic Name

*Generic Name* dipilih dari *list* yang akan ditampilkan. Nama umum grafik adalah nama lain dari *tag name*, item atau batas nilai objek grafik.



### Binding

Varian diikat untuk memilih nama umum yang akan ditampilkan. Varian dapat diikat pada suatu nama umum grafik dengan 4 tipe data, yaitu : nilai numerik, *string* teks, variabel proses dan data *recipe*. Untuk proses pengikatan (*Binding*) nilai numerik atau *string* teks, dapat digunakan secara langsung sedang untuk variabel proses dan data *recipe* dibutuhkan suatu fungsi.

- Fungsi untuk Variabel proses  
`@ProcessData()`  
 Contoh : `@ProcessData(FIC100.ALRM)`
- Fungsi untuk data *recipe* (Mengikuti nama unit)  
`@RecipeUnit()`  
 Contoh : `@RecipeUnit(UT0100.DATA01.ITEM01)`
- Fungsi untuk data *recipe* (Mengikuti Batch ID)  
`@RecipeBatchID()`  
 Contoh : `@RecipeBatchID(01-0010.DATA01.ITEM01)`

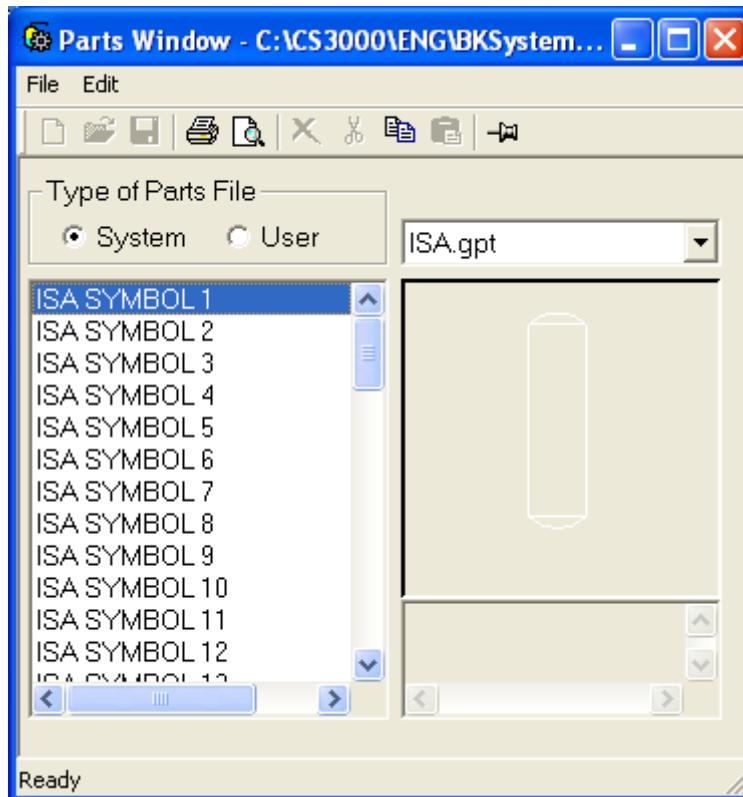
### G. Bagian-bagian Toolbar

Tabel daftar bagian-bagian toolbar

Tombol	Tool	Fungsi
	Parts	Memulai <i>Parts Window</i>
	Linked Part	Memulai <i>Linked Parts Window</i>

#### 1). Parts

Untuk memanggil menu parts klik tombol parts (lihat tabel diatas) pada jendela *graphic builder* atau bisa juga melalui menu [insert] → [Parts]. Jendela Parts Window akan muncul seperti gambar dibawah :

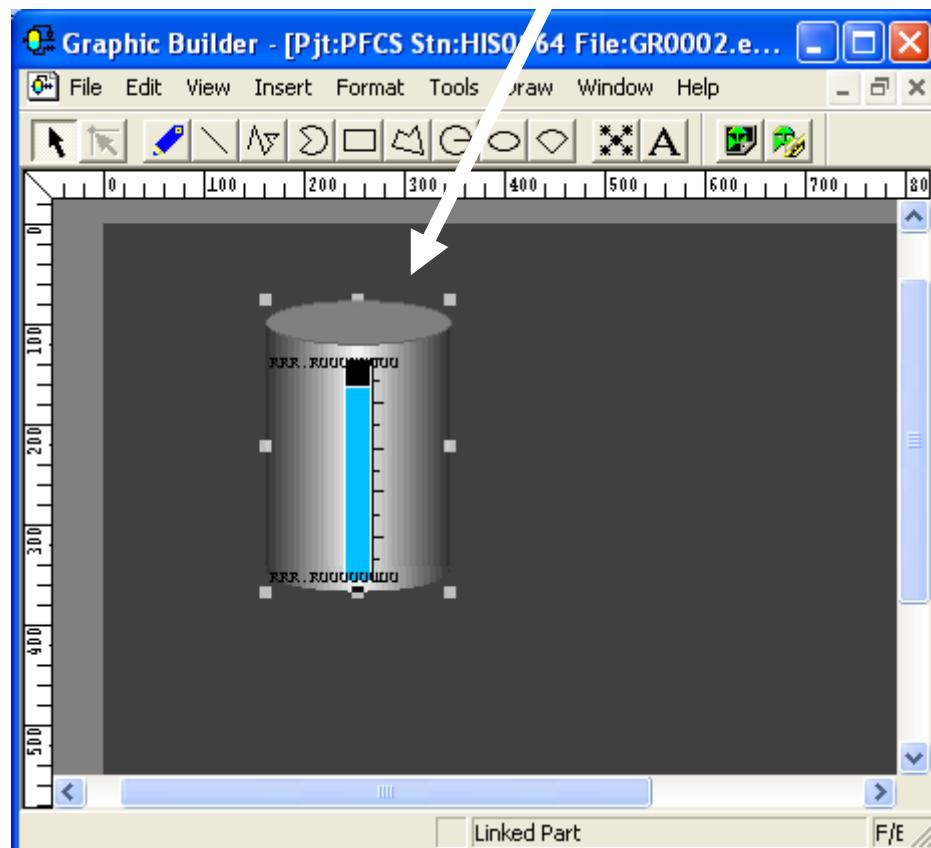
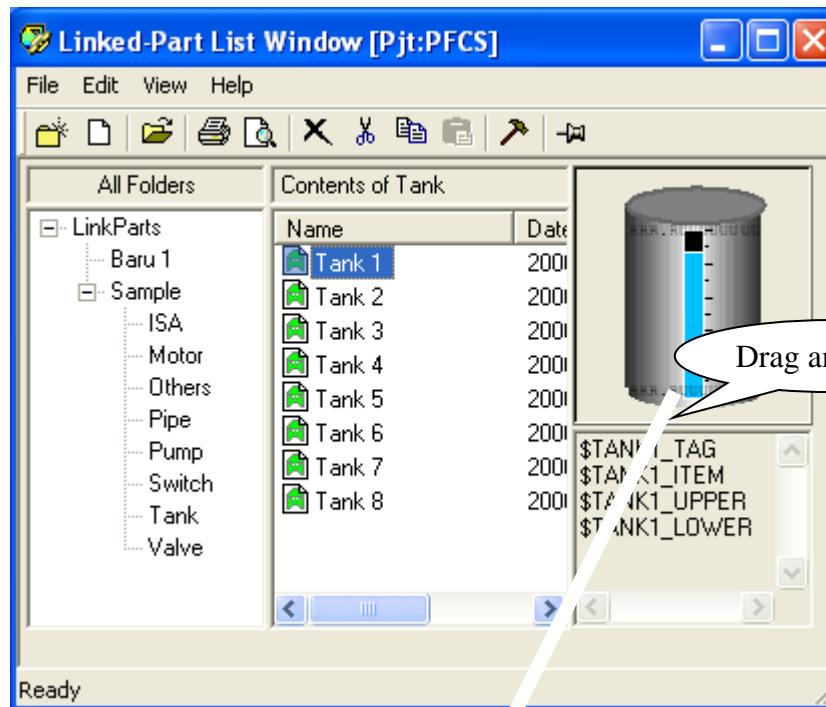


Tipe-tipe parts :

- System  
Secara *default* tipe system diberikan oleh CS3000.
- User  
Digunakan untuk membuka file *parts* yang sudah ada, dengan klik [open] pada menu [File] atau bisa juga untuk membuat file *parts* baru.

### Menyisipkan sebuah Linked Part ke jendela Graphic Builder

Pada menu toolbar pilih [insert] → [Linked Part] atau dengan klik tombol simbol linked part:



## HUMAN MACHINE INTERFACE (HMI) WONDERWARE DAN OPC

### A. Pendahuluan

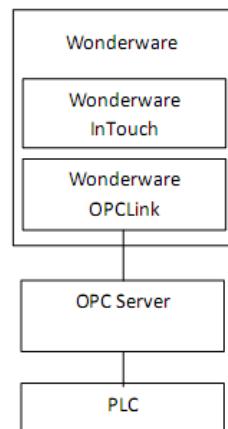
*Wonderware InTouch* adalah salah satu *software SCADA* yang dapat merepresentasikan keadaan real dari suatu proses plant tertentu dalam bentuk Gambar 2 dan animasi. Dengan adanya *software* ini, operator akan lebih mudah untuk mengamati, mengawasi serta mengontrol suatu proses real dimana jarak antar suatu plant dengan stasiun kontrol utama relatif jauh. Bisa juga kondisi di mana kondisi plant tersebut membahayakan keselamatan operator, dalam arti bahwa suatu proses tertentu peran manusia secara langsung harus dihilangkan.

Di sini akan diberikan gambaran umum mengenai cara meng-*online*-kan program HMI yang telah dibuat sebelumnya dengan suatu PLC (dalam percobaan ini menggunakan OMRON CQM 1).

Wonderware InTouch sendiri merupakan program HMI yang dapat dikomunikasikan dengan berbagai jenis PLC. Pada dasarnya protokol komunikasi yang digunakan oleh PLC dan Wonderware berbeda (tiap jenis PLC memiliki protokol komunikasinya sendiri). Untuk itu dibutuhkan *software* yang dapat menjembatani perbedaan protokol komunikasi tersebut. Dalam hal ini dibutuhkan OPC (*OLE for Process Control*) untuk menjembatani komunikasi, di mana OLE adalah *Object Linking Embedding*.

OPC yang dibutuhkan agar Wonderware InTouch dan PLC dapat berkomunikasi terdiri dari *OPC Client* dan *OPC Server*. Mula-mula Wonderware InTouch dihubungkan terlebih dahulu dengan Wonderware OPC Link sebagai *OPC Client*, kemudian dihubungkan lagi dengan *OPC Server* yang berfungsi sebagai driver dari PLC tertentu yang digunakan.

Wonderware OPC Link berfungsi sebagai *converter* protokol komunikasi. OPC Link terhubung dengan *OPC Server*, mengubah perintah *client* ke protokol OPC dan mengirimkannya kembali ke *client* menggunakan *DDE*, *FastDDE*, atau *SuiteLink* (protokol komunikasi Wonderware). Secara umum, komunikasi antara Wonderware dengan PLC dapat dilihat pada Gambar 1.



Gambar 1 Komunikasi antara Wonderware dengan PLC

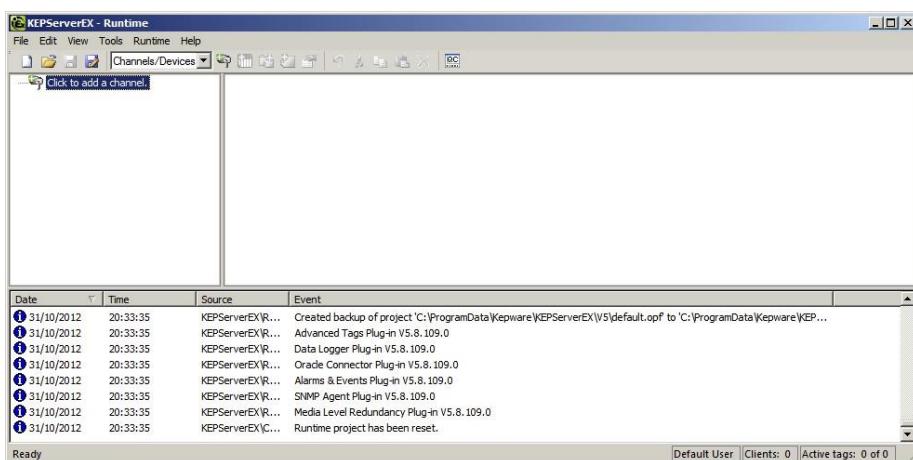
Namun *OPC Server* keluaran terakhir seperti *KEPServerEX5* sudah dapat berkomunikasi dalam protokol *SuiteLink* sehingga tidak dibutuhkan *OPC Client* lagi karena *OPC Server* dapat langsung berkomunikasi dengan Wonderware InTouch. Wonderware terdiri atas beberapa bagian:

- a. **Application Manager:** Membuat *project* baru.
- b. **Window Maker:** Membuat *window*, animasi, dalam suatu *project*.
- c. **Window Viewer:** Menjalankan *project* yang telah dibuat.

## B. Koneksi Wonderware dengan PLC OMRON CQM 1 Menggunakan OPC KEPServerEX

### - Setting KEPServerEX5:

1. Pastikan PC telah terinstal oleh CX-One, Wonderware InTouch, KEPServerEX5.
2. Buat program ladder *self holding* pada CX-Programmer kemudian **download** ke PLC OMRON CQM 1 (cara koneksi CX-Programmer dengan PLC Omron dapat dilihat di modul PLC OMRON).
3. Klik kanan icon *KEPServerEX5* pada *taskbar windows* pilih *Setting*.
4. Pada tab **Runtime Process** pilih **Interactive** pada **Process Mode**, jika perlu pilih **High Priority** untuk menjalankan *service* dengan kategori prioritas tinggi, jika sudah klik **OK**.
5. Buka *KEPServerEX5 Configuration* sehingga akan muncul *window* seperti ditunjukkan Gambar 2

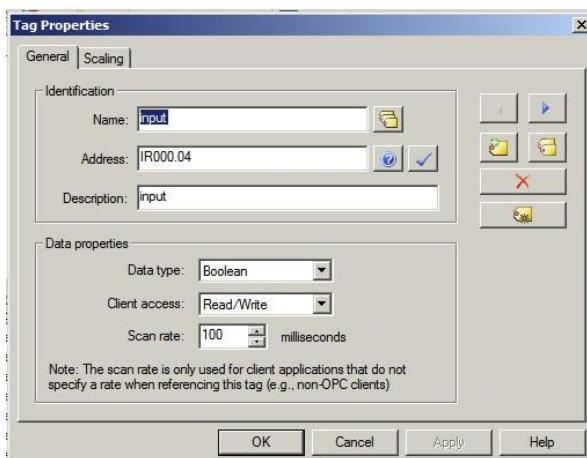


Gambar 2 Window KEPserverEX Configuration

6. Pada *toolbar* klik **File-Project Properties**, pada tab **FastDDE/SuiteLink** centang **Enable FastDDE/SuiteLink connection to the server**. SuiteLink merupakan protokol komunikasi dari Wonderware InTouch. Jika sudah klik **Ok**
7. Pada *toolbar* pilih **File-New** kemudian klik pada tulisan **Click to add a channel** yang terdapat pada bagian kiri panel untuk membuat *channel* baru. *Channel* ini akan memuat sejumlah *device* yang sejenis.
8. Pada *window* yang muncul isikan nama *channel* yang dikehendaki kemudian klik *next* untuk memilih *Device Driver* yang akan digunakan.
9. *Device Driver* disesuaikan dengan perangkat yang digunakan. Pada percobaan ini pilih *Omron Host Link* lalu klik *next* untuk mengatur *port* yang digunakan.
10. Pada **Connection type** pilih **COM Port** karena komunikasi PC dengan PLC menggunakan **COM Port**. Isikan nomor *port* yang digunakan pada **COM ID** (lihat di *device manager* untuk melihat PLC OMRON terhubung ke *port* berapa). Pastikan *port* tidak diakses oleh *software* lainnya. Biarkan isian lainnya **default** kemudian klik *next* terus hingga muncul tombol *finish*.
11. Setelah pengaturan *channel* selesai, klik tulisan **Click to add a device** pada bagian kiri panel untuk membuat *device* baru. Tiap *channel* dapat memuat 32 *device*. Pada *window* yang muncul isikan **nama device** yang diinginkan kemudian klik *next*.
12. Pilih jenis PLC yang digunakan, kali ini pilih **PLC CQM 1**. Untuk **Device ID** isikan **0** kemudian klik *next*. Pada **Scan Mode** pilih **Respect client specified scan rate** kemudian klik *next*. Pada **autodemotion** biarkan **default** atau centang

untuk men-*demote device* jika terjadi kegagalan dalam komunikasi kemudian klik **next**. Pada *Intercharacter Delay* biarkan **default** klik **next** kemudian klik **finish**. Untuk *timing parameter* biarkan **default** kemudian klik **next**.

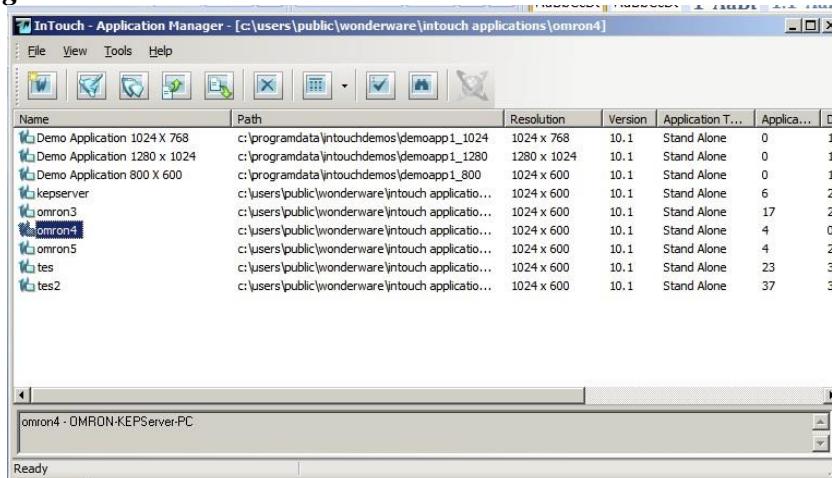
13. Langkah selanjutnya memberikan *tag* pada objek. *Tag* ini berisi alamat dari I/O PLC yang akan diakses oleh *Wonderware InTouch*. Pertama klik tulisan *Click to add static tag* pada bagian kanan panel.



**Gambar 3 Tag properties**

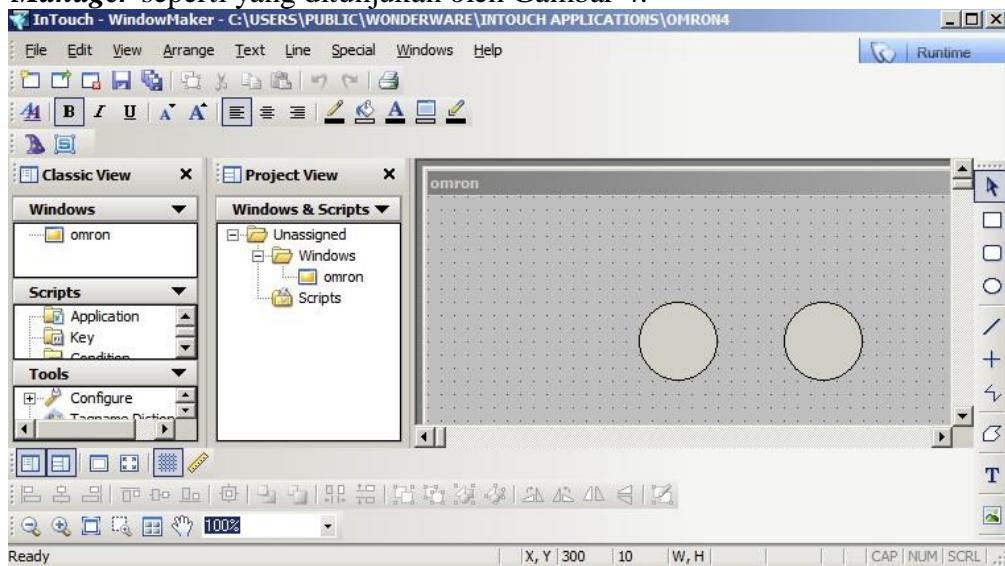
14. Di sini kita akan membuat dua *tag* yang mewakili *input* dan *output* PLC. Penulisan **address** menggunakan **HR** ditujukan untuk *input* dan **IR** ditujukan untuk output, yang keduanya berarti *internal relay*. Pada window **Tag Properties** yang muncul, seperti pada Gambar 3, isikan nama yang dikehendaki beserta deskripsinya jika diperlukan. Pada **Address** isikan **HR000.04** kemudian klik apply jika sudah. Sedangkan penulisan 04 menunjukkan alamatnya. Untuk membuat *tag* lagi klik tombol **new tag** di sebelah kanan. Kali ini isikan **IR100.07** pada **address**. Alamat ini menunjukkan alamat output dari PLC. Untuk alamat lainnya atau pengalamanan *device* tipe lain dapat dilihat di **menu help** dari **device driver Omron Host Link**.
15. Selanjutnya akan dibuat nama yang mewakili PLC yang digunakan. Pada toolbar klik **Edit-Alias Map...** kemudian klik **New Alias**. Pada jendela yang muncul beri nama *alias* sesuai keinginan dan pada *Map to* pilih **Channel1.Device1**

#### - Setting Wonderware InTouch



**Gambar 4 Application Manager**

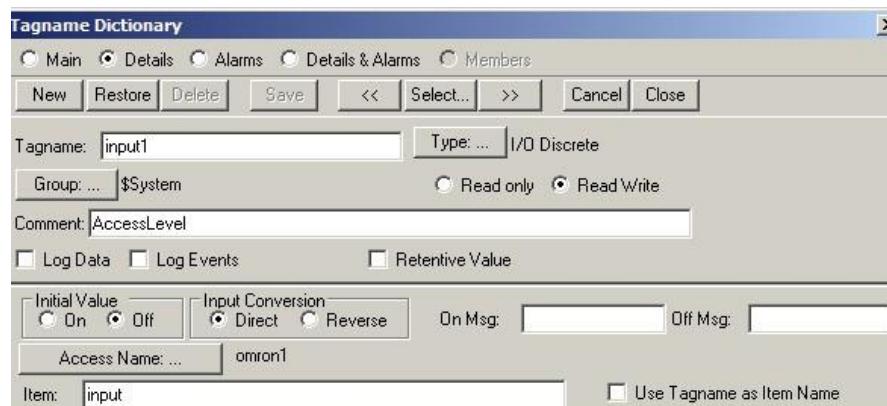
- Buka *Wonderware InTouch* dan pertama kali akan muncul **Window Application Manager** seperti yang ditunjukkan oleh Gambar 4.



**Gambar 5 Window maker**

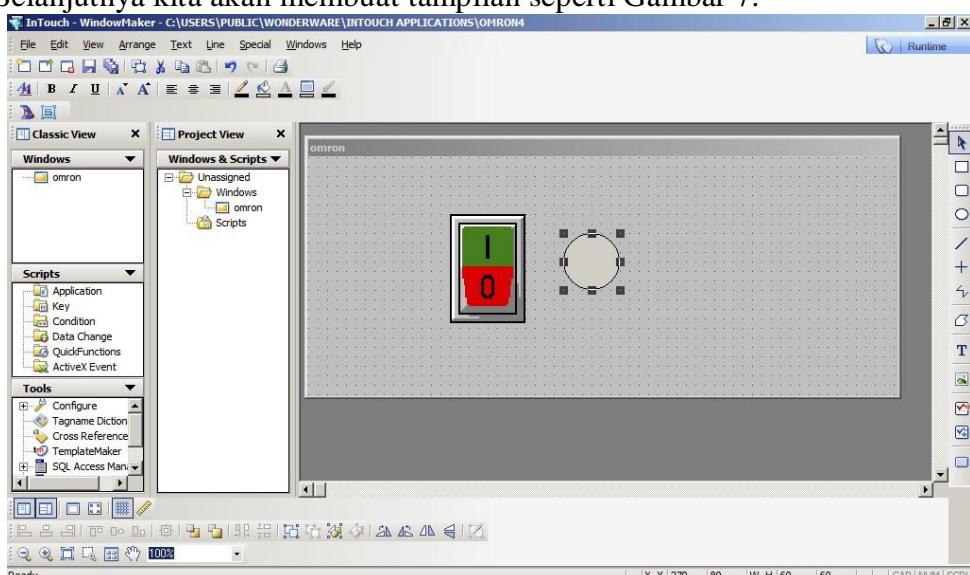
- Buat *project* baru dan isikan nama proyek beserta deskripsi proyek jika diperlukan untuk membedakan satu proyek dengan lainnya. Jika sudah klik dua kali pada nama project yang baru dibuat sehingga akan muncul *window maker* seperti ditunjukkan oleh Gambar 5.
- Pada Toolbar Klik **Special-Acces Name** klik **add** untuk membuat *access name* baru. Pada *window* yang muncul isikan informasi berikut:
  - Access:** Isikan nama yang dikehendaki.
  - Application Name:** *server\_runtime* (nama ini tidak boleh diubah karena merujuk pada aplikasi *KEPServerEX5*, jika menggunakan OPC Server namanya harus ikut disesuaikan).
  - Topic Name:** Isikan sesuai dengan yang telah dibuat sebelumnya pada *Alias Map* pada *KEPServerEX5*.
  - Which Protocol to use:** pilih *SuiteLink*
 Jika sudah klik Ok.
- Selanjutnya akan dibuat *tag* pada *wonderware*. Tag ini akan terhubung dengan tag yang telah dibuat di *KEPServerEX5*. Pada toolbar klik **Special - Tagname Dictionary**. Pada jendela yang muncul klik **New** untuk membuat *tag* baru. Isikan informasi berikut
  - Tagname:** isikan nama *tag* yang dikehendaki,
  - Type:** I/O discrete (I/O karena dihubungkan dengan I/O dari luar, *discrete* menunjukkan tipe *tag*)
  - Access Name:** pilih sesuai dengan *Access name* yang telah dibuat sebelumnya
  - Item:** pilih sesuai nama *tag* pada *KEPServerEX5*

Jika sudah klik **Save**. Untuk membuat *tagname* baru klik **New** dan ulangi langkah di atas. Lakukan ini untuk membuat satu *tagname* lagi karena ada dua *tag* yang dibuat pada *KEPServerEX5* sebelumnya. Gambar 6 merupakan *Window Tagname Dictionary*.



Gambar 6 Tagname Dictionary

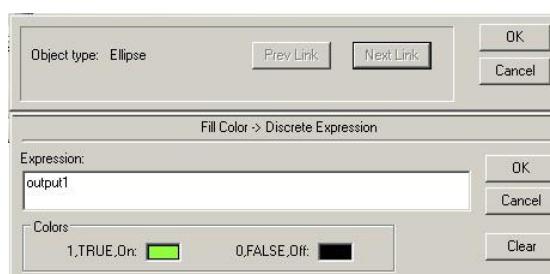
5. Selanjutnya kita akan membuat tampilan seperti Gambar 7.



Gambar 7 Switch and circle

Pertama buat **New Window** - isikan nama yang diinginkan - dan pilih resolusi yang diinginkan dengan catatan tidak melebihi resolusi monitor. Untuk membuat **switch** pertama klik **wizard** (ikon: topi) pada toolbar pilih **Switches - Rocker Switch - Ok**. Letakan **switch** pada tempat yang diinginkan di **window** yang telah dibuat. Untuk membuat lingkaran klik Gambar 2 linkaran di bagian kanan panel kemudian buat lingkaran pada **window**.

6. **Switch** selanjutnya akan dihubungkan dengan **tag input (00004)** dan lingkaran akan dihubungkan dengan **tag output (10007)**. Klik dua kali pada **switch** kemudian isikan **tagname** yang telah dibuat sebelumnya di **Tagname Dictionary**

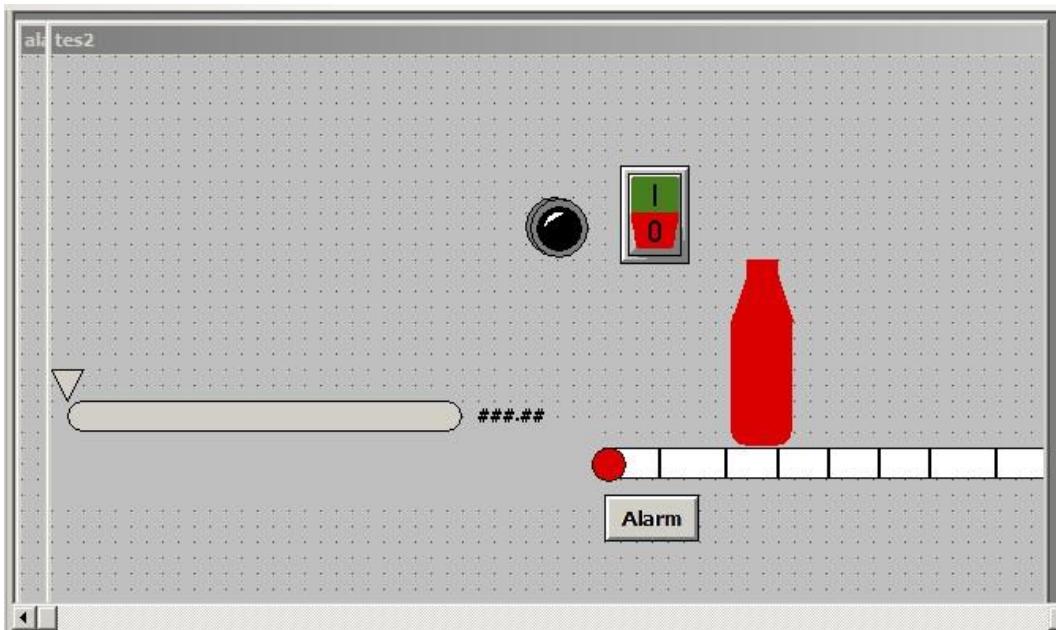


Gambar 8 Animation Link

7. Untuk lingkaran klik 2 kali sehingga akan muncul *window* dengan berbagai pilihan animasi. Dikehendaki lingkaran akan berubah berwarna hijau jika *output* aktif dan hitam jika *output* nonaktif. Klik **Fill Color - Discrete**, isikan *tagname* yang telah dibuat sebelumnya. Pada **color** pilih sesuai yang diinginkan. Informasi yang diisikan ditunjukkan oleh Gambar 8. Jika sudah klik **Ok**.
8. Sebelum menjalankan program yang telah dibuat simpan terlebih dahulu *window* yang telah dibuat, **file - save**. Untuk menjalankan program klik tombol **runtime** yang terletak di pojok kanan atas. Tekan dan lepas tombol input di PLC kemudian perhatikan perubahan yang terjadi pada *switch* serta lingkaran.
- 9.

### 2.1. Percobaan 2: Pengenalan fungsi dasar Wonderware InTouch

Pada percobaan kedua ini akan dibuat sebuah animasi sederhana seperti ditunjukkan Gambar 9.



Gambar 9 Pengisian botol di konveyor

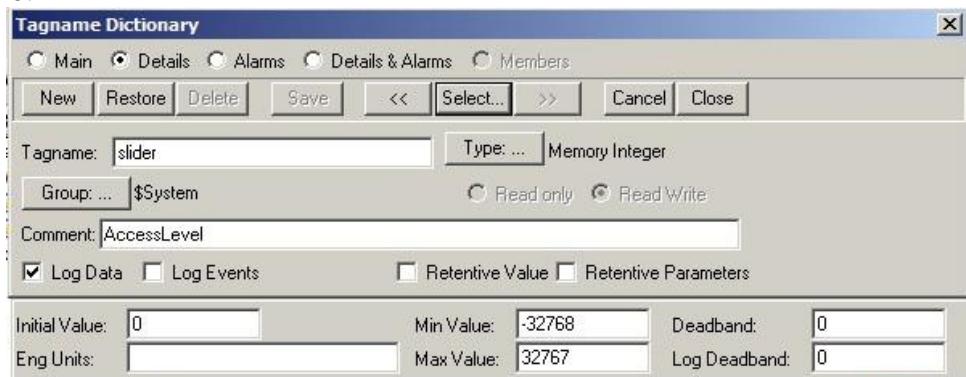
#### Membuat bentuk dasar

1. Buka **application manager** dan buat **project** baru, isikan informasi yang dibutuhkan.
2. Buka **Window Maker** (klik dua kali pada nama **project**). Ketika **Window Maker** terbuka, pilih **File/NewWindow**. Isikan informasi yang diperlukan.
3. Kemudian pilih **Rounded Rectangle** dari **Toolbox**. Gerakan kursor menuju posisi **40, 110** dan tahan tombol kiri **mouse**. **Drag mouse** ke posisi **250, 20** dan kemudian lepaskan.
4. Sekarang pilih **Polygon** dari **Toolbox**. Gerakan **mouse** menuju **30, 90** dan klik kiri (jangan tahan). Gerakan **mouse** ke posisi **50, 90** dan klik lagi. Terakhir gerakan **mouse** ke posisi **40, 110**. Pada posisi ini lakukan **double click**. Ini akan menutup polygon dan mengisinya dengan warna abu-abu.

#### Membuat Tagname

*Tagname* serupa dengan variabel dalam bahasa pemrograman. *Tagname* digunakan untuk menyimpan informasi/nilai. Pilih **Special/Tagname**

**Dictionary.** Klik **New** pada *dialog box* dan isi seperti yang ditunjukan oleh Gambar 10.

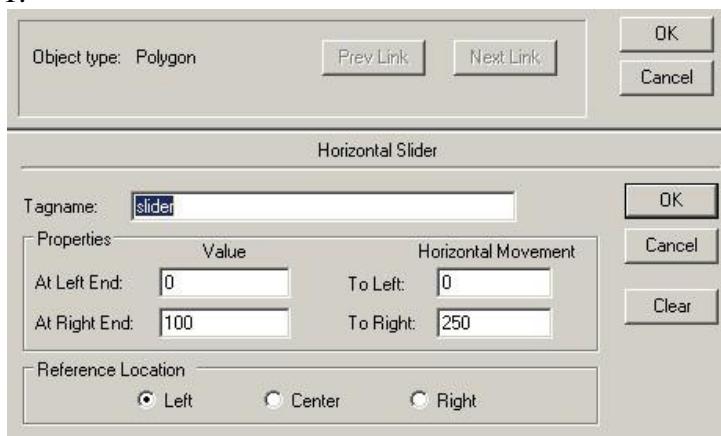


Gambar 10 Tagname Dictionary

Pastikan mengaktifkan **Log Data**. Klik **Save** jika sudah selesai kemudian **Close**.

### Membuat Animasi Objek Dasar

1. Karena *tagname* sudah dibuat, maka kita bisa menggunakannya untuk membuat animasi dari objek dasar yang telah dibuat sebelumnya. *Double click* segitiga yang telah dibuat. Pilih tombol **Horizontal Slider**. Isikan data seperti yang ditunjukan Gambar 11.



Gambar 11 Horizontal Slider

2. **Double click Rounded Rectangle.** Pilih **Percent Fill/Horizontal**. Isikan *slider* sebagai *Tagname*. Klik **Ok**.
3. Sekarang jalankan **Runtime**. Gerakan segitiga ke kanan dan kiri dengan *mouse*. Segitiga berfungsi sebagai *slider* dan *rectangle* terisi dengan warna sembari segitiga digerakan.

### Membuat text indikator

Nilai dari *tagname* “*slider*” bisa ditunjukan pada *window* melalui *text*. Caranya adalah

1. Pilih **text** dari *Toolbox* dan tempatkan pada kordinat **100, 170**. Ketikan: *Slider = #####.##* Tulisan # akan diganti oleh nilai *tagname* saat *runtime* dijalankan.
2. *Double click* pada *text* yang dibuat. Pilih **Value Display/Analog**. Masukan *slider* untuk *Expression*, kemudian klik **Ok**.
3. Jalankan **Runtime**. Tulisan **#####.##** akan digantikan oleh nilai *slider*. Gerakan *slider* ke kanan dan ke kiri kemudian amati efeknya pada *text*.

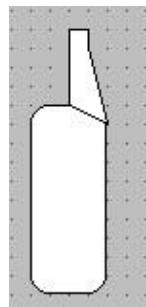
### Membuat Operasi Pengisian Botol

1. Gambar 2 **rounded rectangle** di tengah *window* seperti ditunjukan oleh Gambar 12.



Gambar 12 Rounded Rectangle

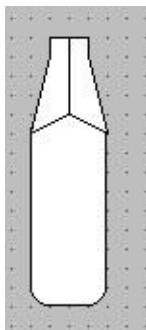
2. Kemudian buat **polygon** di atas dari **rectangle** sehingga didapat bentuk seperti ditunjukan oleh Gambar 2.13



Gambar 13 Polygon di atas rectangle

3. Klik **polygon** memilihnya dan klik **Duplicate Selection** dari *Toolbox*. Kemudian klik **Flip Horizontal**. Pindahkan **polygon** yang telah dibalik disebelah bentuk aslinya hingga keseluruhan bangun membentuk botol seperti ditunjukan oleh Gambar 14.

4.



Gambar 14 Botol

5. Botol yang dibuat terdiri dari tiga bangun, **rounded rectangle** dan **2 polygon**. Ketiga bangun ini bisa dibuat menjadi satu objek dengan tombol **Make Symbol** pada *Toolbox*. Pilih ketiga objek tersebut dengan *mouse* dan klik tombol **Make Symbol**.
6. Kemudian klik botol dan pilih **Percent/Vertical**. Masukan **slider** sebagai **Expression**. Ubah **background color** dengan memilih warna pada **Background Color**. Klik **OK** jika sudah. Sekarang ubah warna isi botol dengan mengklik botol, klik **Fill (on the Toolbox)** dan pilih warna yang dikehendaki.
7. Perhatikan bahwa botol memiliki garis dari **polygons** dan **rectangle**. Hilangkan garis ini dengan memilih **Lines** dari top menu dan **No Lines**.
8. Tekan **Runtime**. Amati proses pengisian botol dengan menggeser **slider**.

### Mengontrol gerakan Botol

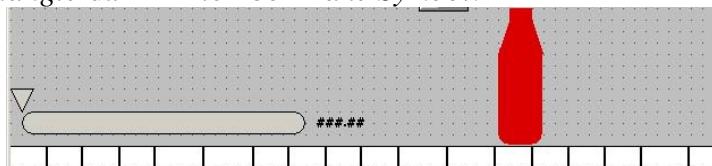
1. Buat *tagname* baru dan isikan parameter berikut:
  - *Name* : bottle
  - *Type* : Memory Real
  - *Minimum Value* : 0
  - *Maximum Value* : 500
  - *Log Data* : Enabled
  - *Log Events* : Disabled
2. Double click botol. Klik **Location/Horizontal button**. Isikan parameter berikut:
  - *Expression* : bottle
  - *Value* : Horizontal Movement
  - *Left End* : 0 Left: 0
  - *Right End* : 400 Right: 400
 Click **OK**.
3. Selanjutnya buat *window script* dengan memilih *Special/Scripts/Window Scripts* dari *top menu*. Isikan *script* berikut pada layar **While Showing**:
 

```
bottle = bottle+ 10;
IF bottle == 100
THEN
IF slider< 100 THEN
slider=slider+l;
bottle=90;
ENDIF;
ENDIF;
IF bottle> 320
THEN
bottle=0;
slider=0;
ENDIF;
```
4. Selanjutnya pada *On Show* dari menu *Window Scripts*, masukan *script* berikut:
 

```
bottle=0;
slider=0;
```
5. Jalankan **Runtime**. Jika benar botol akan bergerak dari kiri ke kanan dan berhenti untuk melakukan pengisian kemudian jalan kembali.

### Membuat konveyor

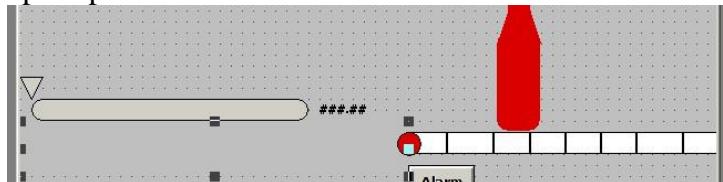
1. Buat *rectangle* berukuran kecil letakan tepat di bawah botol. Duplikasikan hingga membentang dari kiri hingga kanan layar seperti ditunjukkan oleh Gambar 15. Pilih semua *rectangle* dan klik tombol *Make Symbol*.



Gambar 15 Konveyor

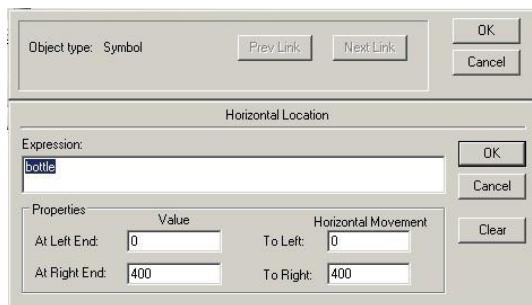
2. Buat kembali *rectangle* kali ini dengan ukuran lebih besar dan letakan di atas *rectangle* kecil yang telah dibuat sebelumnya. Ganti warnanya agar sama dengan background window dan hilangkan garis tepinya. Buat elipse berukuran kecil

dengan diameter sama dengan tinggi rectangle kecil yang dibuat sebelumnya. Hasilnya seperti pada Gambar 16.



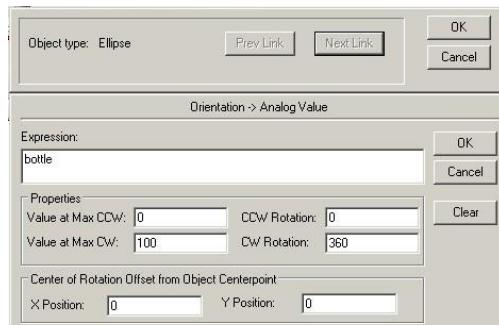
Gambar 16 Hasil akhir konveyor

- Untuk membuat konveyor bergerak *double click* pada rel konveyor. Pilih **Location - Horizontal**. Isikan parameter seperti ditunjukkan Gambar 17 berikut. Jika sudah klik **OK**



Gambar 17 Pengtagan konveyor

- Untuk membuat lingkaran berputar *double click* pada lingkaran. Pilih **Miscelaneous - Orientation**. Isikan parameter seperti ditunjukkan Gambar 18 berikut. Jika sudah klik **OK**.



Gambar 18 Pengetagan agar lingkaran berputar

- Jika sudah klik *runtime* dan amati yang terjadi.

### Membuat Alarm

*Alarm* dikondisikan untuk menunjukkan suatu *event* sedang terjadi pada proses. Wonderware menyediakan fungsi *monitoring* dan pembuatan *alarms*. *Alarm* proses ini akan terjadi ketika sebuah *switch stop* ditekan sehingga proses akan berhenti.

- Buat *tagname* baru dengan nama **ConvStop**. **Type Memory Discrete**, dengan **Log Events Enabled**. Klik **Details and Alarms** kemudian set **Alarm State On**.
- Sekarang klik tombol **Wizard** pada **Toolbox**. Pilih **Switches - On/Off Rocker Discrete Switch**. Letakan tombol ini di atas konveyor. *Double click* pada switch dan set **Tagname** nya **ConvStop**.
- Lagi, pilih **Wizards-Lights-Tube Lights**. Letakan lampu di sebelah switch. *Double click* dan set nama pada **Expression ConvStop.Alarm**.



4. Kemudian pada **Scripts - Window Scripts** sebelumnya menjadi seperti di bawah:  
*bottle = bottle+ 10;  
IF ConvStop == 1 THEN bottle = bottle - 10;  
ENDIF;  
IF bottle == 100 THEN  
slider=slider+1 Bottle = 90;  
ENDIF;  
ENDIF;  
IF bottle > 320 THEN Bottle = 0;  
Slider = 0;  
ENDIF;*

5. Jalankan **Runtime**. Amati perubahan yang terjadi ketika **switch** diaktifkan.

#### Membuat window berisi alarm panel

1. Buat window baru lewat **File - New Window**. Klik **Yes** ketika ditanya “**Copy Window Scripts?**”. Beri nama **window Alarm**, dan buat ukurannya sama dengan **window** yang awal.
2. Pilih **Wizards** dari **Toolbox** menu. Kemudian pilih **Alarm Display - Std. Alarm Display**. Atur agar **alarm display** dapat menampilkan 10 baris,
3. Klik dua kali pada tampilan **alarm**. Karena tidak akan dibuat prioritas, maka klik **Format Alarm Message** dan **unselect Priority**. Lalu pilih **Alarm History** pada tampilan. Tekan **OK**.
4. Karena **window Alarm** sudah dibuat, maka harus dibuat cara untuk berpindah dari **window bottle** ke **alarm** dan sebaliknya. Buka **window Bottle**. Pilih **Button** dari **Toolbox**. Pilih **Special - Substitute Strings** dari top menu. Tulis **Alarm** untuk **string**. **Double click** pada **button** dan pilih **Show Window**. Pilih **Alarm** (hanya **alarm** yang dicentang).
5. Sekarang buka **window alarm**, ulangi proses di atas. Kali beri nama tombol dengan **Bottle** dan pilih **Bottle** pada **Show Window**.
6. Jalankan **Runtime**. Jika benar sekarang memungkinkan berpindah **window** dari **alarm** ke **bottle** dan sebaliknya. Dan ketika terjadi perubahan pada proses, **switch** diaktifkan, maka akan muncul **alarm** yang bersangkutan pada **Alarm Displap Panel**.



# MODUL : PLC SIEMENS S7 300



## Modul Programmable Logic Controller (PLC) Siemens S7-300

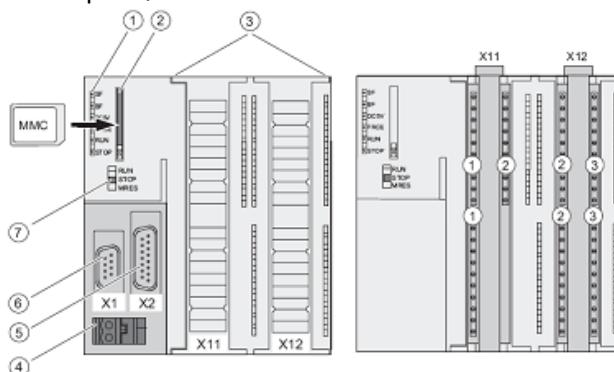
### 1. PENDAHULUAN

*Programmable Logic Controllers (PLC)* adalah komputer elektronik yang memiliki fungsi kendali untuk berbagai tipe dan tingkat kesulitan yang beraneka ragam. Definisi *Programmable Logic Controller* menurut Capiel (1982) adalah : “*Sistem elektronik yang beroperasi secara digital dan didisain untuk pemakaian di lingkungan industri, dimana sistem ini menggunakan memori yang dapat diprogram untuk penyimpanan secara internal instruksi-instruksi yang mengimplementasikan fungsi-fungsi spesifik seperti logika, urutan, perwaktuan, pencacahan dan operasi aritmatik untuk mengontrol mesin atau proses melalui modul-modul I/O digital maupun analog.*”

PLC memiliki bahasa pemrograman yang mudah dipahami dan dapat dioperasikan bila program yang telah dibuat dengan menggunakan *software* yang sesuai dengan jenis PLC yang digunakan sudah dimasukkan. Suatu PLC mempunyai komponen-komponen *hardware* yang berupa *power supply*, *central processing unit* (CPU), memori, dan komponen *input-output* (I/O).

PLC Siemens S7-300 merupakan produk PLC dari vendor Siemens. PLC S7-300 mempunyai spesifikasi *hardware* sebagai berikut:

1. Power Supply
2. CPU 312-5BD01-0AB0
3. Modul *input* I124.0 sampai I124.7 dan I125.0 sampai I125.1
4. Modul *output* Q124.0 sampai Q124.5



Gambar 1. Komponen PLC Siemens S7-300 dan modul *input-output* pada PLC Siemens S7-300

### 2. PENGALAMATAN PADA PLC SIEMENS S7 300

#### A. Alamat Input

Input pada PLC dimulai dari alamat I0.0 sampai I65535.7. Akan tetapi pada PLC Siemens S7-300, alamat yang berhubungan langsung dengan peripheral dimulai dari I124.0 sampai I124.7 dan I125.0 sampai dengan I125.1. Alamat-alamat yang tidak berhubungan dengan peripheral tersebut dapat digunakan sebagai alamat perantara.



### B. Alamat Output

Output pada PLC Siemens S7-300 dimulai dari alamat Q0.0 sampai dengan Q65535.7. Dan yang terhubung langsung dengan peripheral dimulai dari alamat Q124.0 sampai dengan Q124.5

### C. Alamat Memory

Selain alamat input dan output, S7-300 PLC Siemens ini menyediakan lokasi memori yang berbeda – beda, dengan pengalamatan yang sangat unik. Kita dapat memilih memori mana yang akan kita pakai dengan terlebih dahulu memilih spesifikasi alamat, yang meliputi Memory area, address Byte-nya dan Bit numbernya. Memory area pada PLC ada 5 macam yaitu : I, Q, V dan M yang semuanya itu dapat diakses sebagai Byte, Word ataupun Double Word.

**Contoh penulisan pengalamatan baik input/output maupun memory address :**

\* **Addressing Input Register (I) :**

Format :

Bit I[alamat byte].[alamat bit]	= I124.0
Byte, Word, Double Word I[tipe][awal alamat byte]	= IB4

\* **Addressing Output Register (Q) :**

Format :

Bit Q[alamat byte].[alamat bit]	= Q124.0
Byte, Word, Double Word Q[tipe][awal alamat byte]	= QW4

\* **Addressing Variabel Memory area (V) :**

Format :

Bit V[alamat byte].[alamat bit]	= V124.0
Byte, Word, Double Word V[tipe][awal alamat byte]	= VDW4

\* **Addressing Bit Memory area (M) :**

Format :

Bit M[alamat byte].[alamat bit]	= M25.7
Byte, Word, Double Word M[tipe][awal alamat byte]	= MD20

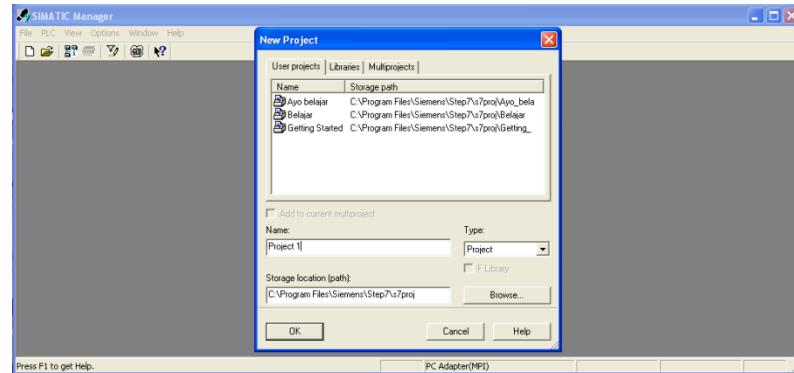
### 3. KONFIGURASI PLC SIEMENS S7-300 DENGAN STEP 7

1. Jalankan program SIMATIC Manager.



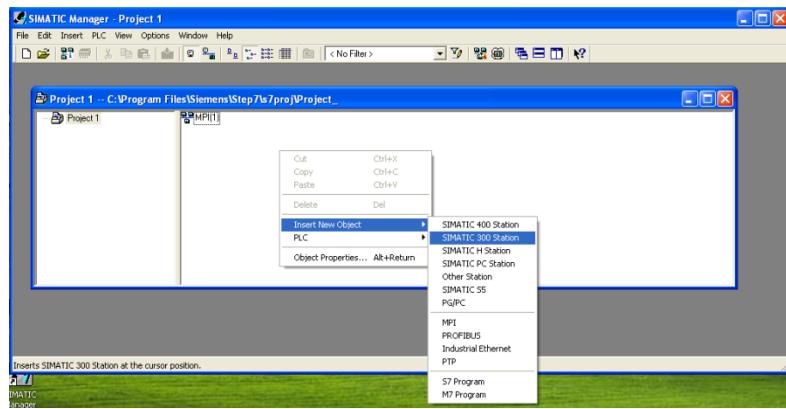
Gambar 2. Simbol Step 7 SIMATIC Manager

2. Membuat project baru dengan memilih **File → New**, dan beri nama *project* yang diinginkan, misal “Project 1”, lalu klik **OK**.



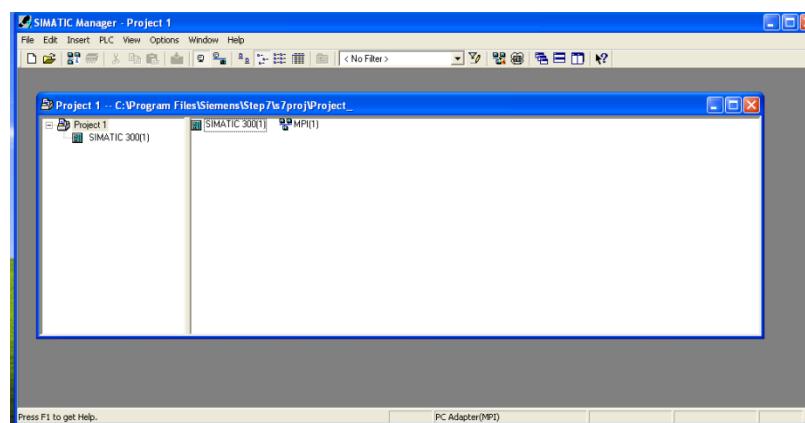
Gambar 3. Tampilan menu pada saat membuat project baru

3. Memilih PLC yang akan digunakan, dalam hal ini PLC Siemens S7-300, dengan cara klik kanan → insert new object → Simatic 300 Station



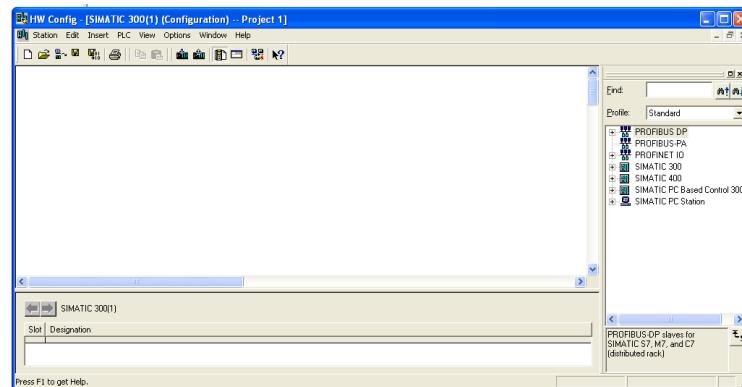
Gambar 4. Tampilan untuk memilih jenis PLC yang digunakan

Setelah memilih insert SIMATIC 300 station, maka akan muncul tampilan seperti pada Gambar 5.



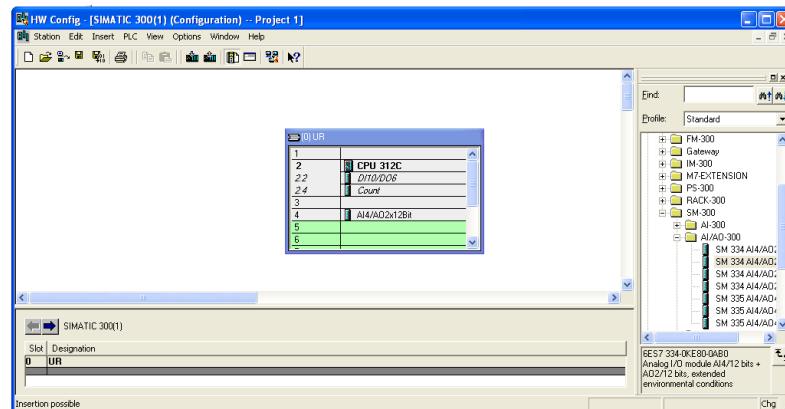
Gambar 5. Tampilan setelah memilih jenis PLC yang digunakan.

4. Melakukan konfigurasi *hardware* dengan cara memilih **SIMATIC 300(1)** → **Hardware**



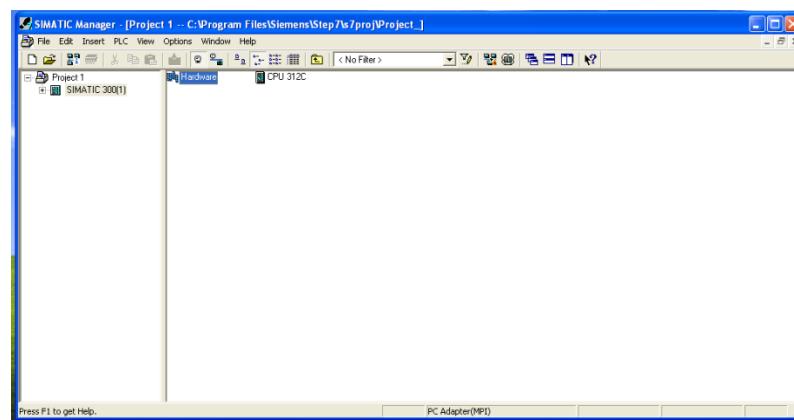
Gambar 6. Tampilan untuk melakukan konfigurasi *hardware*

5. Memilih jenis CPU dan modul *input-output* yang akan digunakan dengan memilih bagian RACK-300, CPU-300 → CPU 312C, dan SM-300 → AI/AO-300.



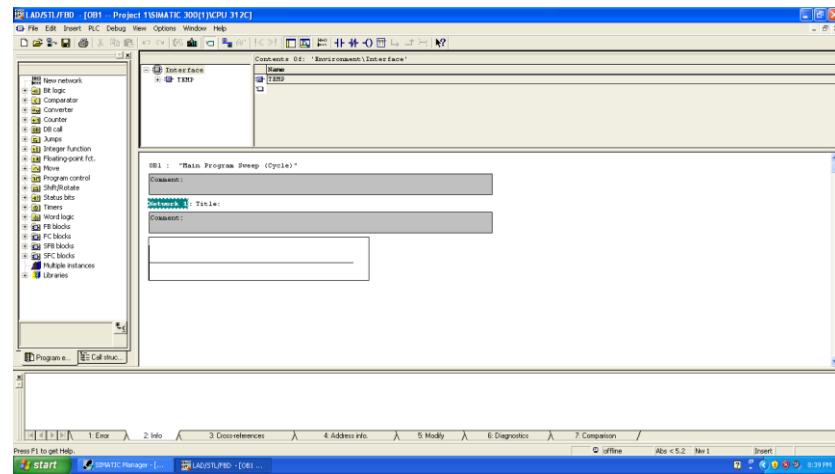
Gambar 7. Tampilan untuk memilih komponen pada konfigurasi *hardware*

Setelah memilih CPU dan modul I/O yang sesuai, klik **save and compile**, maka akan muncul tampilan seperti pada Gambar 8.



Gambar 8. Tampilan setelah melakukan konfigurasi *hardware*

6. Memulai tampilan pemrograman dengan memilih **CPU 312C** → **S7 Program(1)** → **Blocks** → **OB1**, lalu pilih jenis pemrograman yang akan digunakan (dalam hal ini *ladder (LAD)* diagram), maka akan muncul tampilan seperti pada Gambar 9.

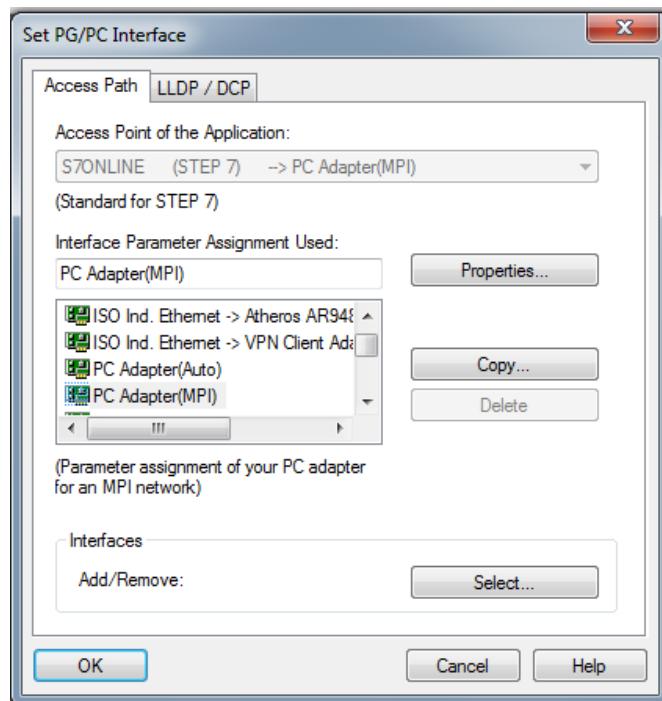


Gambar 9. Tampilan untuk melakukan pemrograman pada PLC

Setelah melakukan konfigurasi, maka pemrograman sudah dapat dilakukan. Untuk mememasukkan program yang sudah dibuat ke PLC, maka pilih menu

#### 4. KOMUNIKASI PLC SIEMENS S-300 DENGAN STEP 7

1. Pada window SIMATIC Manager, pilih menu option → Set PG/PC Interface, maka akan muncul tampilan seperti pada Gambar 10.



Gambar 10. Tampilan untuk set PG/PC Interface

2. Pada bagian Interface **Parameter Assignment Used**, pilih jenis yang akan digunakan, dan yang digunakan pada percobaan ini adalah **PC Adapter (MPI)**, lalu klik **OK**.



## 5. INSTRUKSI-INSTRUKSI PEMROGRAMAN STEP 7

### A. Bit Logic

1. Normally Open Contact ---| |---

Bit ini digunakan untuk memasukkan input yang keadaan normalnya terbuka.

2. Normally Closed Contact ---| / |---

Bit ini digunakan untuk memasukkan input yang keadaan normalnya tertutup.

3. Invert Power Flow --|NOT|--

Bit ini digunakan untuk membalikkan nilai hasil operasi logika sebelumnya.

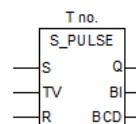
4. Output Coil ---( )

Bit ini digunakan untuk menunjukkan nilai hasil operasi logika.

### B. Timer

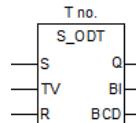
*Timer* merupakan instruksi yang berfungsi memberikan waktu tunda (delay). *Timer* dapat digunakan untuk mengatur lamanya waktu nyala atau waktu mati dari sebuah *output*. Pada pemrograman Step 7, terdapat berbagai macam *timer*, antara lain *pulse timer* (S\_PULSE), *on delay timer* (S\_ODT), dan *off delay timer* (S\_OFFDT).

1. *Pulse Timer* (S\_PULSE)



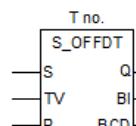
*Timer* ini digunakan untuk menentukan waktu tunda suatu operasi logika.

2. *On delay timer* (S\_ODT)



*Timer* ini digunakan untuk menunda waktu nyala suatu operasi logika.

3. *Off delay timer* (S\_OFFDT)

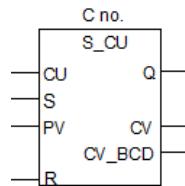


*Timer* ini digunakan untuk menunda waktu mati suatu operasi logika.

### C. Counter

*Counter* merupakan instruksi yang digunakan untuk operasi perhitungan. Terdapat beberapa jenis *counter* pada pemrograman Step 7, yaitu *counter up* (S\_CU), *counter down* (S\_CD), dan *counter up/down* (S\_CUD).

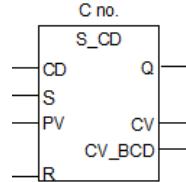
1. *Counter up* (S\_CU)



*Counter up* digunakan untuk menghitung urutan naik.

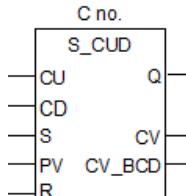


## 2. Counter down (S\_CD)



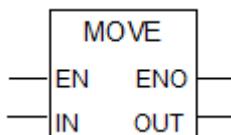
Counter down digunakan untuk menghitung urutan turun.

## 3. Counter up/down (S\_CUD)



Counter up/down digunakan untuk menghitung urutan naik atau turun.

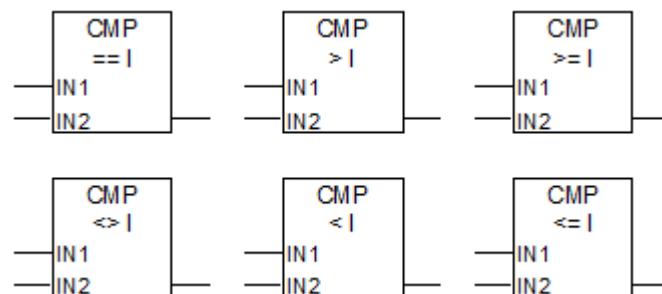
## D. Move



Instruksi move diaktifkan oleh input enable (EN), dan memindahkan nilai dari alamat input (IN) ke output (OUT).

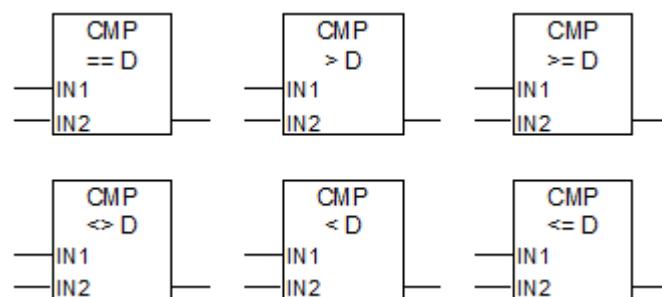
## E. Comparator

### 1. Compare integer



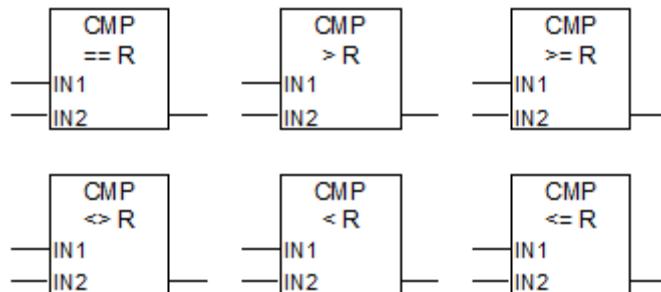
Instruksi compare integer akan membandingkan nilai pada input 1 (IN1) dan input 2 (IN2), dan hasil operasi logikanya akan diletakkan di output. Tipe data pada operasi ini adalah integer.

### 2. Compare double integer



Instruksi compare double integer akan membandingkan nilai pada input 1 (IN1) dan input 2 (IN2), dan hasil operasi logikanya akan diletakkan di output. Tipe data pada operasi ini adalah double integer.

#### 1. Compare real

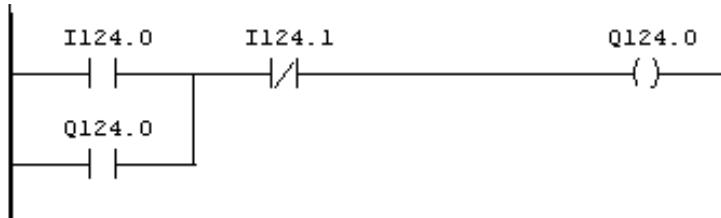


Instruksi compare double integer akan membandingkan nilai pada input 1 (IN1) dan input 2 (IN2), dan hasil operasi logikanya akan diletakkan di output. Tipe data pada operasi ini adalah double integer.

### 6. APLIKASI INSTRUKSI

#### A. Rangkaian Self Holding pada Ladder Diagram

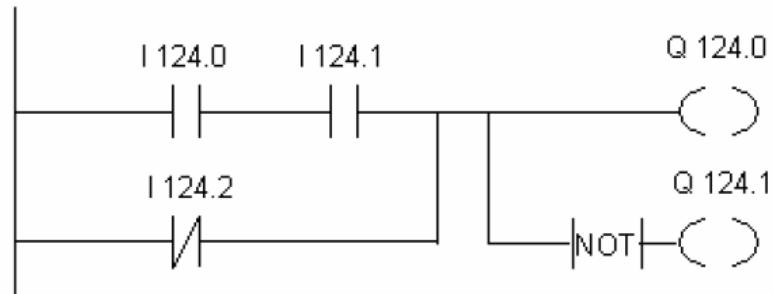
Rangkaian self holding adalah rangkaian yang dapat mempertahankan kondisi nilai suatu operasi logika.



Gambar 11. Rangkaian self holding pada ladder diagram

#### B. Rangkaian Logika dengan Ladder Diagram

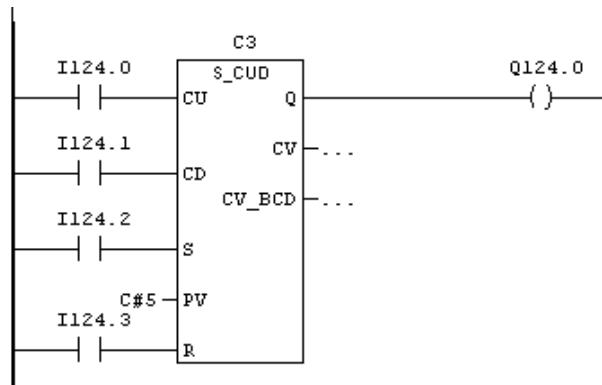
Rangkaian logika seperti *or*, *and*, dan *not* dapat diaplikasikan menggunakan ladder diagram. Pada Gambar 12, dengan mengubah-ubah kondisi input, maka kedua output akan selalu mempunyai nilai logika yang berbeda.



Gambar 12. Aplikasi rangkaian gerbang logika pada ladder diagram

### C. Rangkaian Counter

Pada Gambar 13, ketika I124.0 ditekan, maka *counter* akan menghitung naik, dan ketika I124.1 ditekan, counter akan menghitung turun. Output akan menyala ketika nilai hitungan counter lebih dari 0.

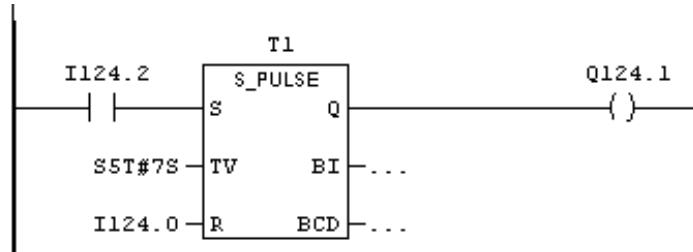


Gambar 13. Aplikasi counter pada ladder diagram

### D. Rangkaian Timer

#### 1. Pulse Timer

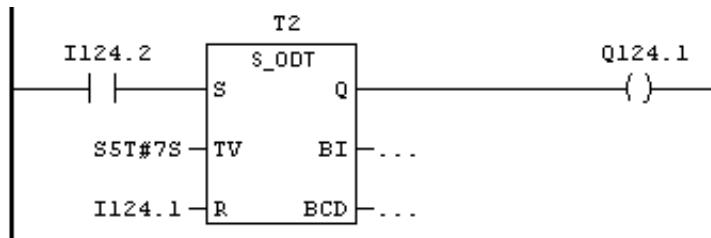
Pada Gambar 14, dengan menekan tombol I124.2, maka karakteristik dari sebuah *pulse timer* dapat diamati.



Gambar 14. Aplikasi pulse timer pada ladder diagram

#### 2. On Delay Timer

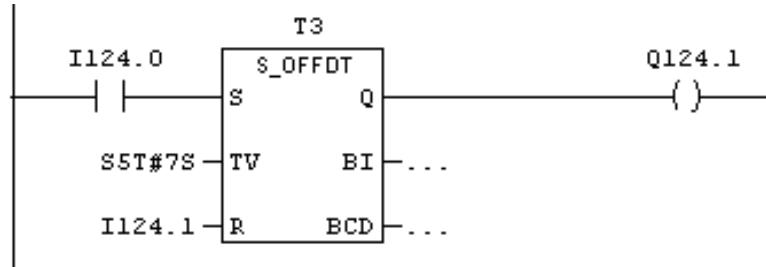
Pada Gambar 15, dengan menekan tombol I124.2, maka karakteristik dari sebuah *on delay timer* dapat diamati.



Gambar 15. Aplikasi on delay timer pada ladder diagram

### 3. Off Delay Timer

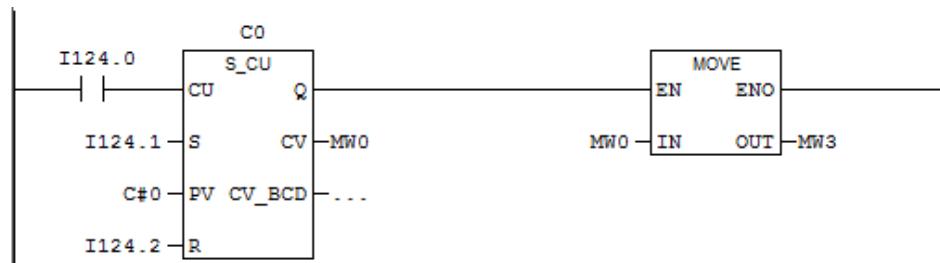
Pada Gambar 16, dengan menekan tombol I124.2, maka karakteristik dari sebuah *off delay timer* dapat diamati.



Gambar 16. Aplikasi *off delay timer* pada ladder diagram

### E. Aplikasi Instruksi Move

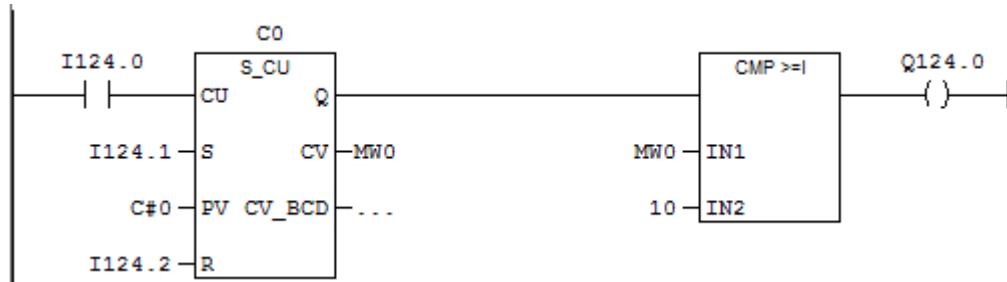
Pada Gambar 17, dapat diamati bahwa nilai dari counter yang terdapat pada alamat memori MW0 akan dipindah ke alamat MW3 oleh instruksi move.



Gambar 17. Aplikasi dari instruksi move

### F. Aplikasi Instruksi Compare

Pada Gambar 18, dapat diamati bahwa instruksi compare akan membandingkan nilai pada Input 1 (IN1) dan Input 2 (IN2). Hasil dari perbandingan tersebut akan mempengaruhi nilai logika output.



Gambar 18. Aplikasi dari instruksi comparator



# MODUL : Programmable Logic Controller Omron CQM1





## Programmable Logic Controller Omron CQM1

### 1.1 Introduction

Automation system is a process by which the system is expected to work on the effectiveness and efficiency of the value of time, human labor and so on, easy in operation and produce a good outcome process. To realize the above, it is used a so-called control programming PLC (Programmable Logic Controller). PLC is define as a digital electronic device that uses programmable memory to store instructions and to implement specific functions such as logic, sequence, timing, counting, and arithmetic to control various machines and processes. PLC consists of power supply, CPU(Central Processing Unit), input modules, output modules, memory, programming devices.

In this experiment, we used Omron PLC. There are different types of omron PLC such CQM1 , CPM1, and C200H. However, for this practical, we use CQM1 with CX Programmer as programming software. For modification or loading of new program, a programming is used. Various programming devices are available like Hand-Held Communicator (HHC), PC, and memory chip. Programming devices can be connected to PLC via the peripheral port/serial port using RS-232 cable. PLC CQM1 configuration can be seen in Figure 1.1.

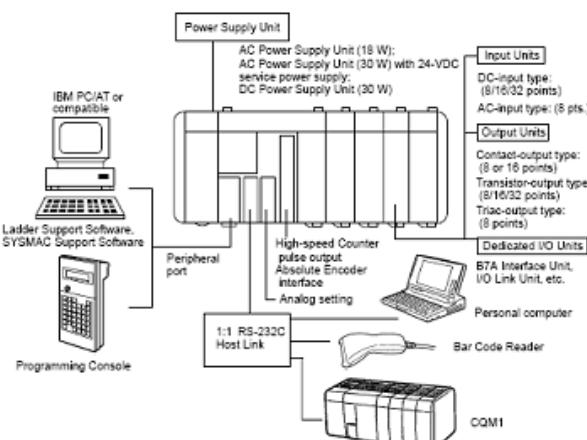
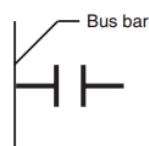


Figure 1.1 Configuration System

### 1.2 Basic Instructions

#### 1.2.1 Ladder Instructions

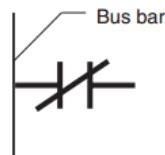
- Load



Description

LD is used for the first normally open bit from the bus bar or for the first normally open bit of a logic block.

- Load Not



Description

LD NOT is used for the first normally closed bit from the bus bar, or for the first normally closed bit of a logic block.

- And



Description

AND is used for a normally open bit connected in series.

- And Not



Description

AND NOT is used for a normally closed bit connected in series.

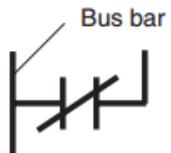
- Or



Description

OR is used for a normally open bit connected in parallel.

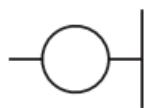
- Or Not



Description

OR NOT is used for a normally closed bit connected in parallel.

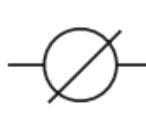
- Output



Description

OUT is used for a normally open coil.

- Output Not

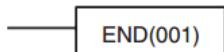




#### Description

OUT is used for a normally closed coil.

- End(001)

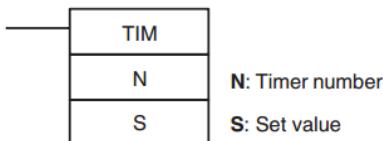


#### Description

END ( 01 ) is required as the last instruction in any program . There is no written instructions after END ( 01 ) will be executed . If there is no END ( 01 ) in the program , then no instructions are executed and the message " NO END INST "

### 1.2.2 Timer/Counter Instructions

- TIM



N: Timer number  
S: Set value

#### Description

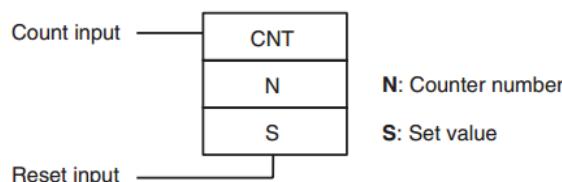
The timer will be activated when the previous RLO value is " 1 " and the value of SV will be reset when the previous RLO value is " 0 " . The output of the TIM will be active when the value of SV = 0 and will hold it until the timer is reset .

N= TC Number (000 - 511)

SV= Set Value (in 100 ms)

- CNT

#### Simbol



N: Counter number  
S: Set value

#### Description

When the RLO before the instruction CNT value changes from " 0 " to " 1 " then the value of PV ( present value ) will be reduced by 1. When the value PV = 0 then, output the CNT would be= " 1 " . When input R changes from " 0 " to " 1 " then the PV value will be reset to the value of the SV .

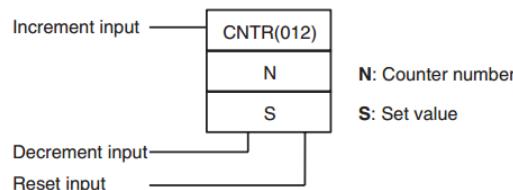
N= TC Number (000 - 511)

SV= Set Value

PV=Present Value

- CNTR(12)

#### Simbol



#### Description

When the value RLO before instruction II ( Increment Input ) changed from " 0 " to " 1 " then the value of PV ( present value ) would be a increment by 1. When the value RLO before instruction DI ( Decrement Input ) changed from " 0 " to " 1 " then PV value (present value) will be reduced by 1. When the value PV = 0 then output CNTR would be = " 1 ". When input R changes from " 0 " to " 1 " then the PV value will be reset to the value of the SV .

N= TC Number (000 - 511)

SV= Set Value

PV=Present Value

### 1.3 Programming Console

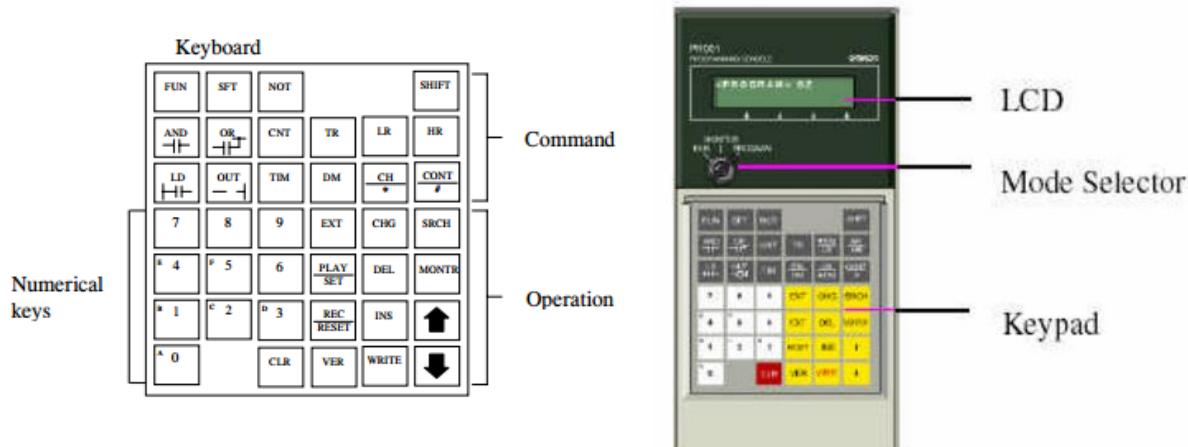


Figure 1.2 Programming Console CQM1-PRO01-E

- Program Mode  
The CPU Unit is stopped. Programming operations, such as writing or changing programs, clearing memory, and checking the program, can be performed.
- Monitor Mode  
The CPU Unit is operating and I/O processing is being performed. In this mode, CPU Unit operation can be monitored and functions such as forcing bits ON/OFF, changing timer/counter SV/PC, changing word data PVs, and online editing can be used. This mode is often used for making program adjustments and for trial system operations
- Run Mode  
Used for normal operation of the CPU Unit. The operating status of the CPU Unit can



be monitored in this mode, but functions such as forcing bits ON/OFF and changing PVs and SVs cannot be performed.

- Password program

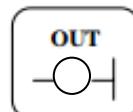
*CLR-> MONTR -> CLR -> Program*

- Clear all program

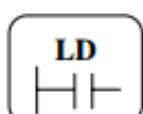
*CLR-> SET -> NOT -> RESET -> MONTR -> CLR*



Add Function



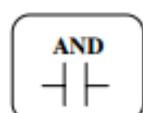
Output



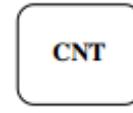
Load



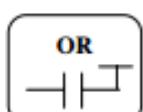
Timer



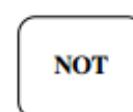
AND logic



Counter



OR Logic



Not Logic

## 1.4 References

- Passen, David W., Industrial Automation : Circuit Design and Component, Haifa : John Wiley & Sons, Inc : 1989.
- Liptak, Bela G., Instrument Engineers' Handbook, Fourth Edition, Volume Two: Process Control and Optimization
- Liptak, Bela G., Instrument Engineers' Handbook, Volume 3: Process Software and Digital Networks, Fourth Edition
- PLC Omron User manual

## 1.5 Addressing CX-Programmer

PLC OMRON have many configuration, there are CPM1, CQM1, C200H etc. This addressing in PLC OMRON CQM1:

- Input Address

Input address start from 00000 to the big number in input of PLC. In this experiment use PLC OMRON CQM1 with 16 input. So have address input from 00000 to 00015.

- Output Address

Output address in this PLC type start from 10000 to 10015

- Internal Relay Address

Internal relay address start from 20000 to the biggest number who PLC can use.



## 1.6 CX-Programmer Online Procedures and Omron PLC CQM1

1. Make sure the PC is connected to the PLC using RS-232 cable.
2. Click the **Start Menu**→**All Programs**→**OMRON**→**CX-One**→**CX-Programmer**. Click **File menu**→**New** or by pressing **Ctrl+N** so Change PLC window will appear as shown in Figure 1.3.

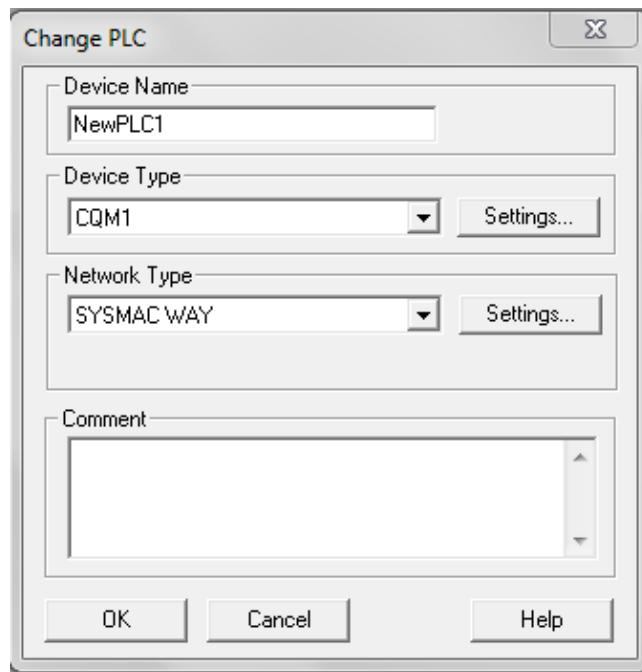


Figure 1.3 Change PLC

3. Fill the **Device Name** with your own word as project name. In **Device Type** select **CQM1** (in this experiment CQM1 type is used). Then press the **Settings** button so that **Device Test Setting (CQM1)** window will appears, Figure 1.4.

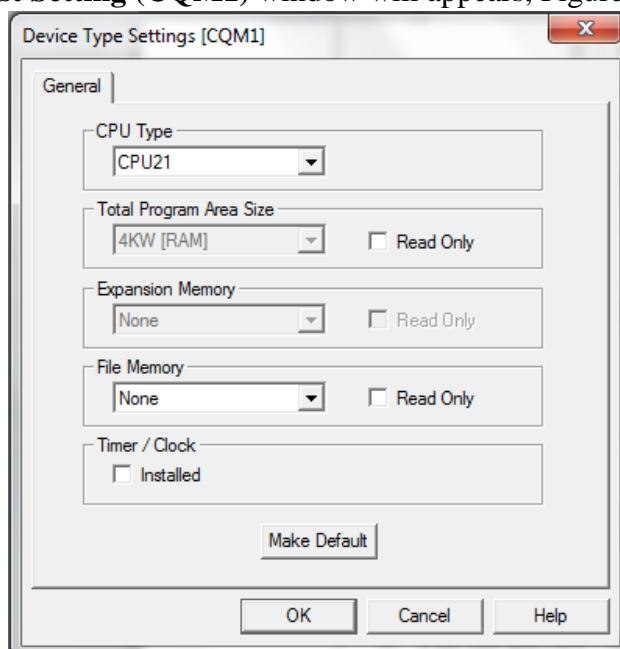


Figure 1.4 Device Type Setting



4. Choose **CPU21** as CPU Type. In **Memory File** choose **None**, and click **OK**.
5. Return to the previous window, in **Network Type** choose **SYSMAC WAY**. Then click **Settings**, **Network Setting (SYSMAC WAY)** window will appear as in Figure 1.5.

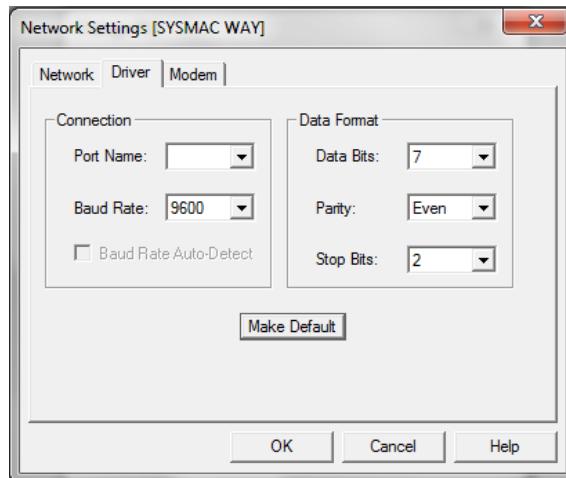


Figure 1.5 Network Setting

6. On the **Driver** tab, enter the appropriate **Port Name** on the CPU you are running (see the **Device Manager**), then select **9600, 7, Even, 2** in **Baud Rate, Data Bits, Parity, and Stop Bits**. Then click **OK** it will be return to the previous window, and click **OK** in Change PLC window. Now you can program your PLC.
7. To **download** a program that has been created, **compile** your program in order to determine whether there exists an error or not in the program by clicking **Program Menu → Compile** or by pressing **Ctrl+F7**, see Figure 1.6. If no error appears , then make sure the CX-Programmer online with the PLC by clicking **PLC Menu → Work Online** or by pressing **Ctrl+W**, see figure 1.7.

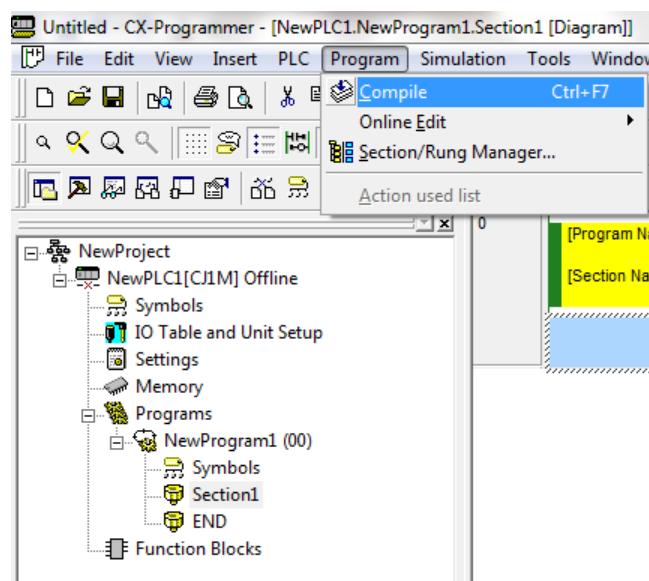


Figure 1.6 Compile program

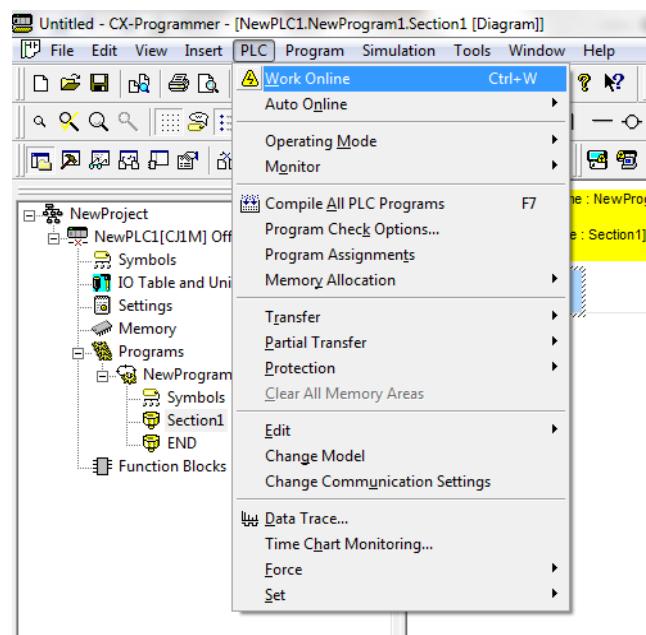


Figure 1.7 Work Online

8. After being connected make sure the PLC is in program condition, to set it click **PLC Menu→Operating Mode→Program** or by pressing **Ctrl+1**, see figure 1.8.

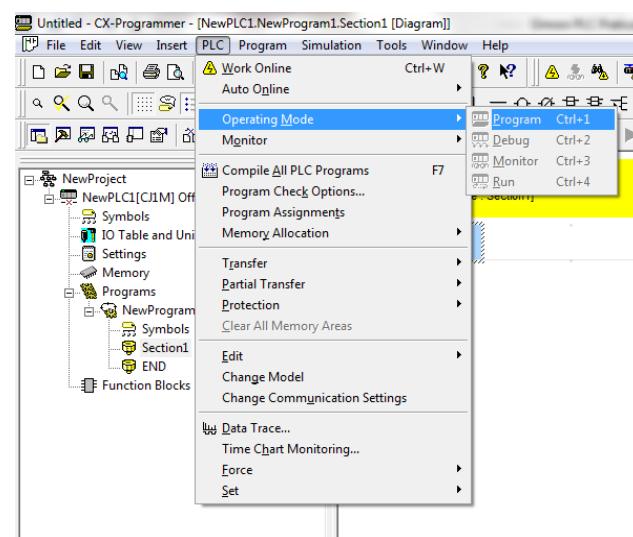


Figure 1.8 Change to program mode

9. Click **PLC Menu → Transfer→To PLC** or by pressing **Ctrl+T**. If the transfer is completed set PLC to run mode by clicking **PLC Menu→Operating Modes→Run** or by pressing **Ctrl+4**.



# MODUL : PLC- LG GLOFA GM4



## PLC LG ( GLOFA GM4 )

### 1. Pendahuluan

Otomasi sistem adalah proses membuat suatu sistem bekerja efektif dan efisien dalam hal waktu dan energy yang dibutuhkan, langkah yang mudah , dan tentunya keluaran yang sesuai. Untuk mewujudkan hal tersebut, sebuah alat pengaturan yang sering digunakan adalah *Programmable Logic Control (PLC)*



Gambar 1. PLC LG GLOFA GM4

Pada praktikum ini akan digunakan PLC LG dengan menggunakan *software* GMWIN 4.0. PLC LG mempunyai banyak tipe seperti GLOFA GM4, GM6, dan GM7. Untuk praktikum kali ini digunakan tipe GM4. Pada Gambar 1 merupakan bentuk fisik dari PLC LG GLOFA GM4 ,sedangkan untuk spesifikasi input/output akan dijelaskan pada Gambar 2.

Input type		DC Input				
Part number		G4I-D22A/C	G4I-D22B	G4I-D24A/C	G4I-D24B	G4I-D28A
Input point		16 points		32 points		64 points
Rated input voltage				DC 12/24V <sup>*1</sup>		
Rated input current		5/11mA		3/7mA		3/6mA
On voltage/current		DC 9.5V or more/4mA or more		DC 9.5V or more/3mA or more		
Off voltage/current				DC 6V or less/1.0mA or less		
Response time	Off → On			10ms or less		
	On → Off			10ms or less		
Common		8 points/1COM		32 points/1COM		
Type		Source/Sink	Source (+COM)	Source/Sink	Source (+COM)	Source/Sink
Operating Indicator				LED		
Insulation method				Photocoupler insulation		
Current consumption (DC 5V)		70mA		75mA		250mA
G4I-D22/		G4I-D22B	G4I-D24/	G4I-D24B	G4I-D28A <sup>*2</sup>	
 Terminal block No.		 Terminal block No.	 Pin No.	 Pin No.	 Pin No. <sup>Expt</sup>	 Pin No. <sup>Expt</sup>

Gambar 2. Data sheet input dari PLC LG

Output type	Relay output		Triac output			
Part number	G4Q-RY2A		G4Q-SS2A	G4Q-SS2B		
Output point	16 points		16 points			
Rated load voltage	DC 12/24V, AC 110/220V (50/60Hz)		AC 100~240V (50/60Hz)			
Rated load current	1 Point 1 Common	2A 4A/1COM	1A 5A/1COM	0.6A 2.4A/1COM		
Response time	Off → On On → Off	10ms or less 12ms or less	0.5cycle +1ms or less 0.5cycle +1ms or less			
Common	8 points/1COM					
Operating indicator	LED					
Type	-					
Insulation method	Photocoupler insulation					
Surge absorber	-		Varistor, CR absorber			
Current consumption (DC 5V)	100mA		330mA			
External power supply	DC24V					

G4Q-RY2A *2	G4Q-SS2A	G4Q-SS2B
Terminal block No.	Terminal block No.	Terminal block No.

**Gambar 3. Data sheet output dari PLC LG**

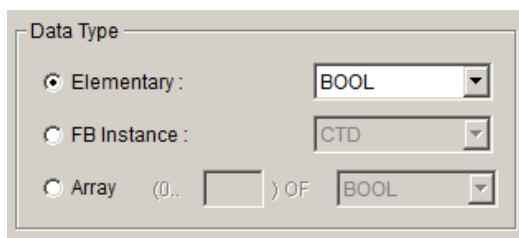
### 1.1. Data type

Tipe data yang digunakan pada *software* GMWIN 4.0 dapat dilihat pada Tabel 1.

**Tabel 1. Tipe data yang mendukung GLOFA PLC**

No	Simbol	Tipe Data	Ukuran (bits)	Interval
1	SINT	Short Integer	8	-128 ~ 127
2	INT	Integer	16	-32768 ~ 32767
3	DINT	Double Integer	32	-2147483648 ~ 2147483647
4	LINT	Long Integer	64	-2^63 ~ 2^63 - 1
5	USINT	Unsigned Short Integer	8	0 ~ 255
6	UINT	Unsigned Integer	16	0 ~ 65535
7	UDINT	Unsigned Double Integer	32	0 ~ 4294967295
8	ULINT	Unsigned Long Integer	64	0 ~ 2^64 - 1
9	REAL	Real Numbers	32	-3.402823E38 ~ -1.401298E-45 4.9406564E-324 ~ 1.7976931E308
10	LREAL	Long Real Numbers	64	-1.7976931E308 ~ -4.9406564E-324 4.9406564E-324 ~ 1.7976931E308
11	TIME	Duration	32	T#0S ~ T#49D17H2M47S295MS
12	DATE	Date	16	D#1984-01-01 ~ D#2163-6-6
13	TIME_OF_DAY	Time of Day	32	TOD#00:00:00 ~ TOD#23:59:59:999
14	DATE_AND_TIME	Date and Time	64	DT#1984-01-01-00:00:00 ~ DT#2163-6-6-23:59:59:999

15	STRING	Character String	30*8	Limited within 30 letters
16	BOOL	Boolean	1	0, 1
17	BYTE	Bit String of Length 8	8	16#0 ~ 16#FF
18	WORD	Bit String of Length 16	16	16#0 ~ 16#FFFF
19	DWORD	Bit String of Length 32	32	16#0 ~ 16#FFFFFFFF
20	LWORD	Bit String of Length 64	64	16#0 ~ 16#FFFFFFFFFFFFFFF



Gambar 4. Tampilan pengaturan tipe data

## 1.2. *Initial value*

Jika *initial value* dari data tidak diisi, maka akan otomatis berisi seperti Tabel 2.

Tabel 2. *Initial value* dari berbagai jenis data

Tipe Data	Initial Value
SINT, INT, DINT, LINT	0
USINT, UINT, UDINT, ULINT	0
BOOL, BYTE, WORD, DWORD, LWORD	0
REAL, LREAL	0.0
TIME	T#0S
DATE	D#1984-01-01
TIME_OF_DAY	TOD#00:00:00
DATE_AND_TIME	DT#1984-01-01-00:00:00
STRING	“(empty string)”



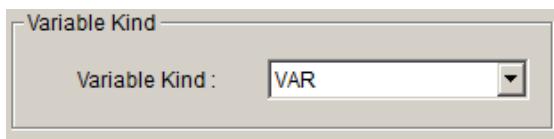
Gambar 5. Tampilan pengaturan *initial value*

## 1.3. *Variable declaration*

Berfungsi mendeklarasikan bagian untuk diubah pada sebuah elemen program. Pada Tabel 3. berikut adalah tipe dari pendeklarasian variable

Tabel 3. Tipe dari variabel

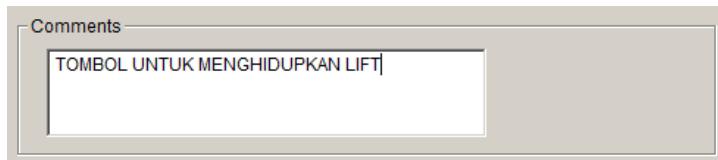
Tipe Variabel	Deskripsi
VAR	Variabel umum yang tersedia untuk <i>read/write</i>
VAR_RETAIN	Menjaga data dari variabel
VAR_CONSTANT	Variabel yang hanya bisa di <i>read</i>
VAR_EXTERNAL	Mendeklarasikan variabel sebagai VAR_GLOBAL



Gambar 6. Tampilan pada pengaturan tipe variabel

#### 1.4. Comment

Memberi comment bertujuan untuk menambahkan penjelasan pada fungsi, kontak, fungsi blok bila diperlukan



Gambar 7. Tampilan pada pengaturan comment

#### 1.5. Addressing

Addressing merupakan sesuatu hal yang sangat penting yang dilakukan untuk menghubungkan PLC dengan software tertentu. Setiap PLC memiliki karakteristik tersendiri dalam cara memberikan alamat. Berikut pada Tabel.2 adalah contoh pemberian alamat pada PLC LG GLOFA

Tabel 4. Data sheet pemberian alamat

Prefix	Comments
I	<i>Input</i>
Q	<i>Output</i>
M	<i>Internal Memory</i>
X, none	1 bit
B	1 byte ( 8 bit )
W	1 word ( 16 bit )
D	2 word ( 32 bit )
L	4 word ( 64 bit )

Contoh ) %QX3.1.4 or %Q3.1.4 : Bit ke 4 dari slot 1 pada rak 3  
 %IW2.4.2 : word ke 2 dari slot 4 pada rak 2  
 %MD48 : double word ke 48 dari internal memory

%MW10.3

: Bit ke 3 dari word ke 10 di internal memory

## 1.6. Pengenalan Hardware

### a. Power Supply Module



Merupakan modul elektrik yang berfungsi menyuplai tegangan.

- Port merah(paling atas) berfungsi sebagai masukan + 24 volt
- Port hitam (paling bawah)berfungsi sebagai masukan 0 volt

### b. Switch Module



Merupakan modul elektrik yang berfungsi sebagai modul masukan.

- Port merah(paling atas) berfungsi sebagai masukan + 24 volt
- Port hitam (paling bawah) berfungsi sebagai masukan 0 volt
- Port kuning berfungsi sebagai com (referensi)

### c. Buzzer and Lamp Module

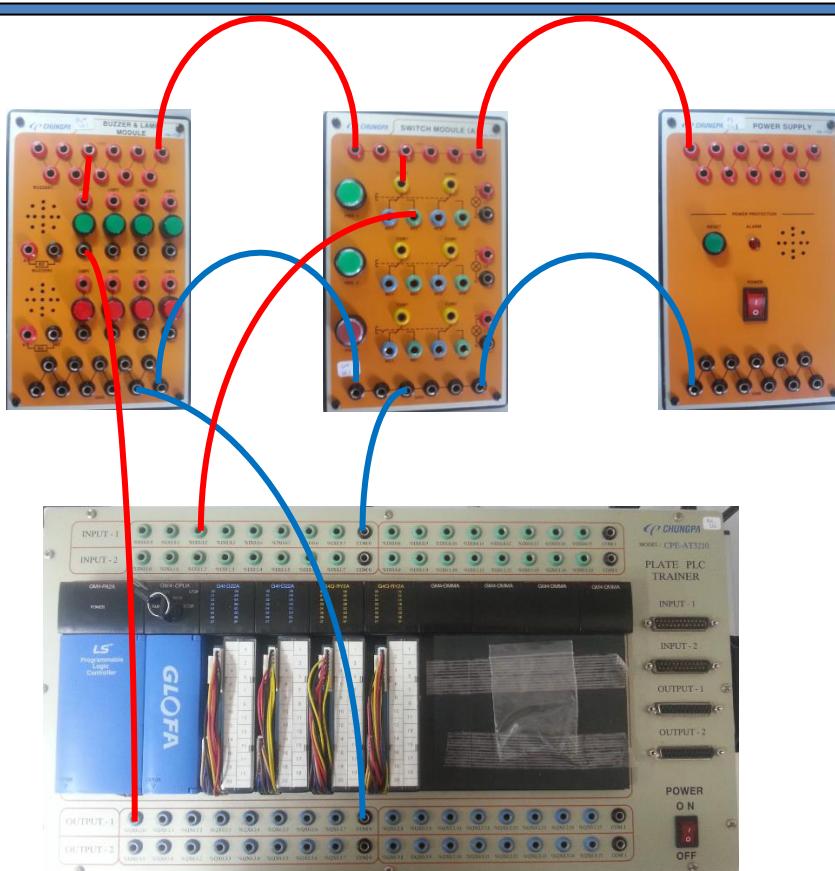


Merupakan modul elektrik yang berfungsi sebagai relay.

- Port merah(paling atas) berfungsi sebagai masukan + 24 volt
- Port hitam(paling bawah) berfungsi sebagai masukan 0 volt

### d. Wiring

Pemasangan kabel *wiring* antara PLC dan juga modul input/output secara sederhana dapat dilihat pada Gambar 8.

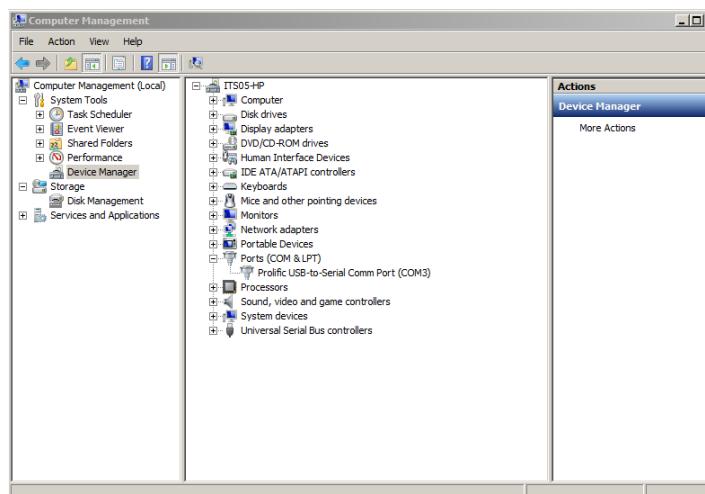


Gambar 8. Tampilan pada pengaturan tipe variabel

## 2. Langkah-langkah percobaan

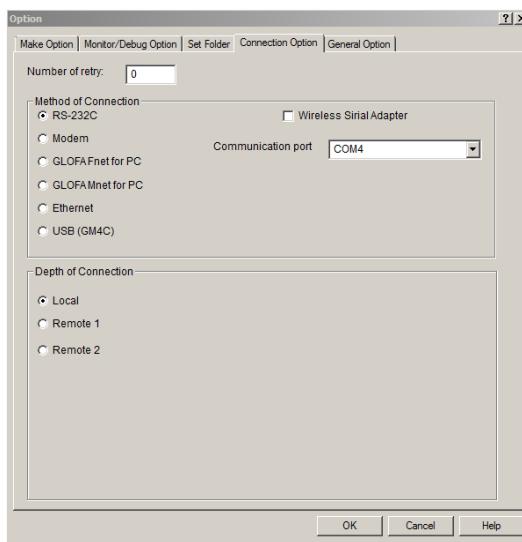
### 2.1 Konfigurasi PLC LG dengan software GMWIN 4.0

- Pastikan kabel USB to serial sudah terhubung dengan PLC LG.
- Cek sambungan USB to Serial (com 1~4) pada komputer anda. Klik kanan computer -> manage -> device manager -> ports. Sehingga akan tampak tampilan window seperti Gambar 9.



Gambar 9. Tampilan window untuk memeriksa sambungan USB to Serial.

- Buka software GMWIN 4.0 , pilih menu project -> option -> connection option. Sehingga akan tampak seperti pada Gambar 10.

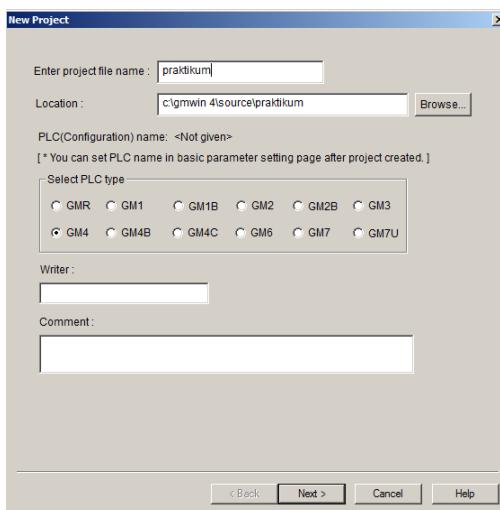


Gambar 10. Tampilan *option* untuk melakukan pengaturan koneksi.

- Set RS.232C pada metode koneksi
- Set (com 1 ~ 4) pada communication port
- Set local pada *depth of connection*

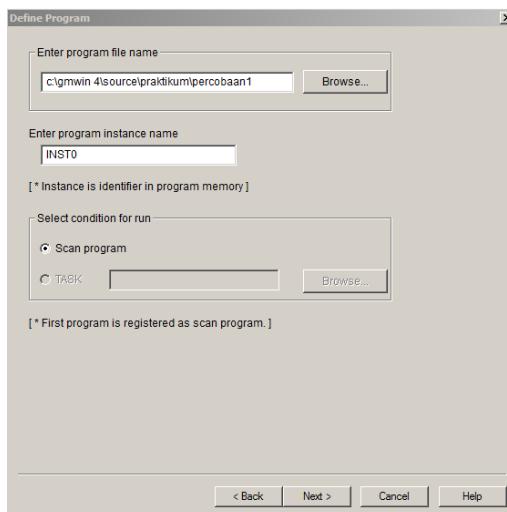
Jika sudah klik **Ok**.

4. Membuat project baru. Project -> new project kemudian beri nama project tersebut dan pilih jenis PLC yang akan anda gunakan seperti pada Gambar 11. Untuk melanjutkan klik **Next**.



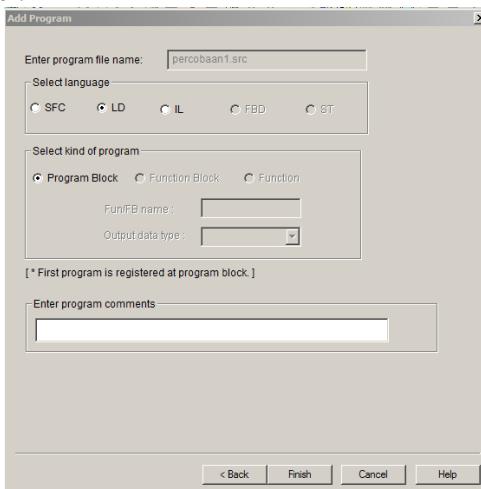
Gambar 11. Tampilan *menu* untuk memberi nama *project* dan memilih jenis PLC

Lanjutkan dengan memberi nama program , untuk melanjutkan klik **Next**.



Gambar 12. Tampilan menu untuk memberi nama program

Langkah terakhir memilih bahasa pemrograman yang digunakan,pilih LD (ladder) kemudian klik **Finish**



Gambar 13. Tampilan menu untuk memilih bahasa program

5. Nyalakan PLC , *Online -> Connect* . Jika kata *offline* pada status bar di kanan bawah window berubah menjadi GM4 stop, maka PLC anda sudah terhubung dengan PC.

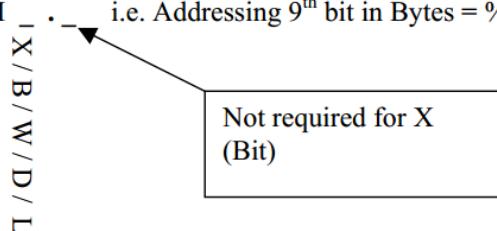
## 2.2 Instruksi dasar pada software GMWIN 4.0

1. Addressing
- Referensi *Input/Output*

**%\_ \_ \_ . \_ \_ \_ i.e. 1<sup>st</sup> bit in Input area = %IX0.0.0**

- Referensi Memori

%M . . i.e. Addressing 9<sup>th</sup> bit in Bytes = %MB1.2, in Words = %MW0.9



Simbol	Deskripsi
X	Bits
B	Byte
W	Word
D	Double Word
L	Long word
M	Memory
I	Input
Q	Output

## 2. Instruksi dasar

- Normally open contact

Simbol



Deskripsi

Instruksi NO akan bernilai “1” ketika dia aktif dan akan bernilai “0” ketika dia tidak aktif

- Normally close contact

Simbol



Deskripsi

Instruksi NC akan bernilai “0” ketika dia aktif dan akan bernilai “1” ketika dia tidak aktif

- Positive edge triggered contact

Simbol



Deskripsi

Positive edge triggered contact mendeteksi perubahan kondisi dari “0” ke “1” sehingga RLO = “1” setelah instruksi ini.

- Negative edge triggered contact



Deskripsi

Negative edge triggered contact mendeteksi perubahan kondisi dari “1” ke “0” sehingga RLO = “1” setelah instruksi ini.

- Normally open coil

Simbol



Deskripsi

Ketika RLO bernilai “1” maka NO coil akan bernilai “1”. Ketika RLO bernilai “0” maka NO coil akan bernilai “0”.

- Normally close coil



Deskripsi

Ketika RLO bernilai “1” maka NO coil akan bernilai “0”. Ketika RLO bernilai “0” maka NO coil akan bernilai “1”.

- Set coil

Simbol



Deskripsi

Jika RLO bernilai “1” maka <address> akan dibah nilainya menjadi “1”. Ketika RLO bernilai “0” maka tidak akan memberikan efek apapun pada <address>.

- Reset coil

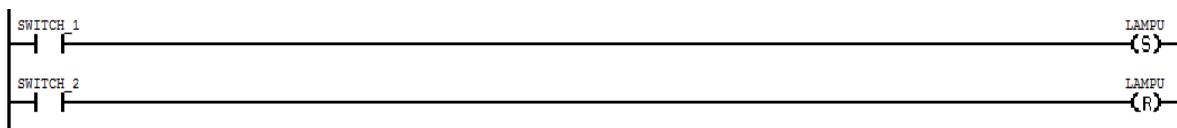
Simbol



Deskripsi

Jika RLO bernilai “1” maka <address> akan dibah nilainya menjadi “0”. Ketika RLO bernilai “0” maka tidak akan memberikan efek apapun pada <address>.

- Contoh



3. Timer

IN = Kondisi timer

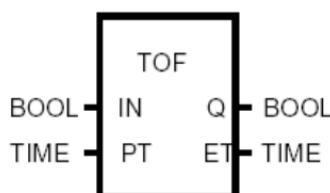
PT = Waktu Preset

Q = Keluaran

ET = Waktu yang terhitung

- TOF (Timer off delay)

Simbol

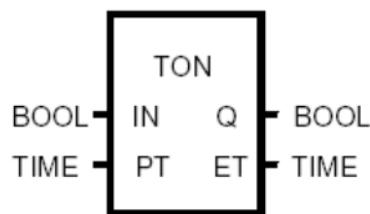


Deskripsi

Ketika masukan IN bernilai “1” maka keluaran Q bernilai “1” kemudian ketika masukan IN bernilai “0” maka timer akan mulai menghitung sampai nilai PT. Ketika perhitungan selesai maka keluaran Q akan bernilai “0”

- TON(Timer on delay)

Simbol

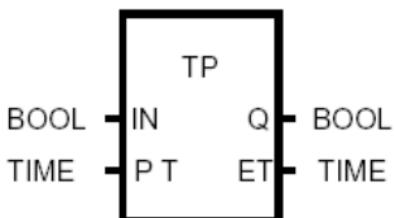


Deskripsi

Ketika masukan IN bernilai “1” maka timer akan mulai menghitung sampai nilai PT. Ketika perhitungan selesai maka keluaran Q akan bernilai “1”

- TP(Timer pulse)

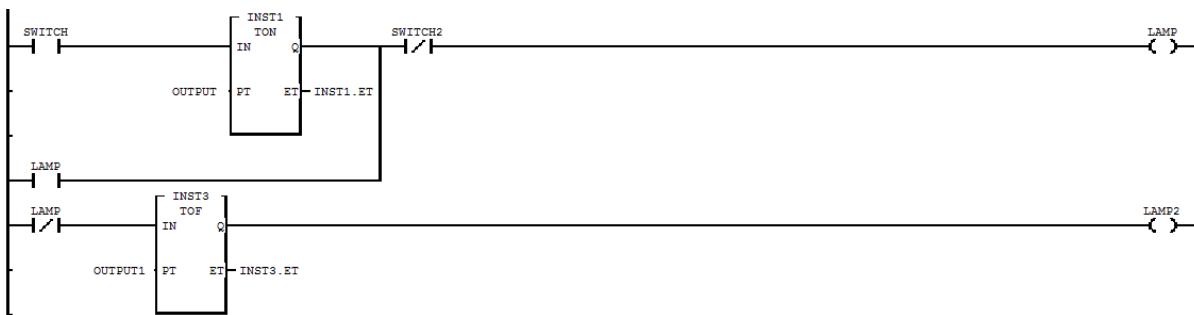
Simbol



Deskripsi

Ketika IN bernilai "1" maka keluaran Q bernilai "1" dan timer akan memulai perhitungan. Ketika perhitungan mencapai nilai PT maka Q akan bernilai "0" dan timer akan kembali menghitung. Timer akan mengubah keluaran dari "1" ke "0" terus menerus sampai IN bernilai "0".

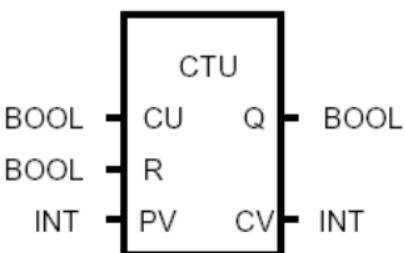
- Contoh



#### 4. Counter

- Counter UP

Simbol

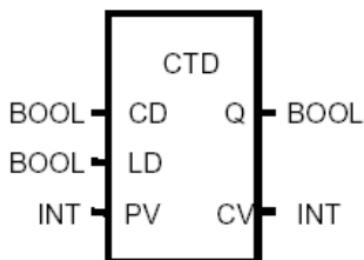


Deskripsi

Ketika masukan CU berubah dari "0" ke "1" maka nilai CV akan ditambah 1. Keluaran Q akan bernilai "1" ketika nilai CV=PV. Q akan bernilai "1" hingga masukan R bernilai "1". Ketika masukan R bernilai "1" maka CV akan direset menjadi 0.

- Counter Down

Simbol

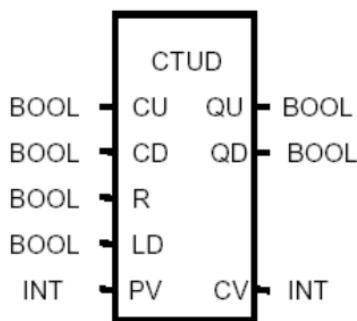


#### Deskripsi

Ketika masukan LD bernilai "1" maka nilai CV akan sama dengan PV. Ketika masukan CD berubah dari "0" ke "1" maka nilai CV akan dikurangi 1. Keluaran Q akan bernilai "1" ketika nilai CV=0. Q akan bernilai "1" hingga masukan LD bernilai "1" lagi.

- Counter UP/Down

Simbol



#### Deskripsi

Merupakan gabungan dari CTU dan CTD. Ketika nilai CV=PV maka keluaran QU akan bernilai "1" sedangkan jika CV=0 maka keluaran QD akan bernilai "1"

- Contoh

