UNIVERSITAS INDONESIA

PERANCANGAN SISTEM PENGAWASAN DAN KENDALI ALIRAN

SKRIPSI

SKRIPSI INI DIAJUKAN UNTUK MELENGKAPI SEBAGIAN
PERSYARATAN MENJADI SARJANA TEKNIK

HERU CAHYONO
0706199413

FAKULTAS TEKNIK UNIVERSITAS INDONESIA
DEPARTEMEN TEKNIK ELEKTRO
DEPOK
MEI 2009

Skripsi ini adalah hasil karya saya sendiri,

dan semua sumber baik yang dikutip maupun dirujuk

telah saya nyatakan dengan benar.

Nama : Heru Cahyono
NPM : 0706199413

Tanda tangan:

Tanggal : 30 Juli 2009

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Heru Cahyono
NPM	: 0706199413
Program Studi	: Teknik Elektro
Judul Skripsi	: Perancangan Sistem Pengawasan dan Kendali Aliran

Telah berhasil dipertahankan di hadapan Dewan Penguji dan diterima sebagai bagian persyaratan yang diperlukan untuk memperoleh gelar Sarjana Teknik pada Program Studi Teknik Elektro Fakultas Teknik, Universitas Indonesia.

DEWAN PENGUJI

Dosen Pembimbing
Ir. Wahidin Wahab MSc. PhD ( )

Dosen Penguji 1
Dr. Abdul Muis ST, M.Eng ( )

Dosen Penguji 2
Dr. Ir. Feri Yusivar M.Eng ( )

Ditetapkan di : Depok

Tanggal : 6 Juli 2009

KATA PENGANTAR

Puji syukur saya panjatkan kepada Tuhan Yang Maha Esa, karena atas berkat dan rahmat-Nya, saya dapat menyelesaikan skripsi ini. Penulisan skripsi ini dilakukan dalam rangka memenuhi salah satu syarat untuk mencapai gelar Sarjana Teknik Jurusan Elektro pada Fakultas Teknik Universitas Indonesia. Saya menyadari bahwa, tanpa bantuan dan bimbingan dari berbagai pihak, sangatlah sulit bagi saya untuk menyelesaikan skripsi ini. Oleh karena itu, saya mengucapkan terima kasih kepada:

1. Bapak Ir. Wahidin Wahab MSc. PhD selaku dosen pembimbing yang telah menyediakan waktu, tenaga dan pikiran untuk mengarahkan saya dalam penyusunan skripsi ini,
2. Seluruh staf pada Departemen Elektro Fakultas Teknik Universitas Indonesia,
3. Orang tua dan keluarga saya yang telah memberikan bantuan dukungan material dan moral.

Akhir kata, saya berharap Tuhan Yang Maha Esa berkenan membalas segala kebaikan semua pihak yang telah membantu. Semoga skripsi ini membawa manfaat bagi pengembangan ilmu.

Depok, 17 Mei 2009

Heru Cahyono
0706199413

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai sivitas akademik Universitas Indonesia, saya yang bertanda tangan di bawah ini :

Nama : Heru Cahyono
NPM : 0706199413
Program Studi : Teknik Elektro
Departemen : Elektro
Fakultas : Teknik
Jenis Karya : Skripsi

Demi pengembangan ilmu pengetahuan, menyetujui untuk memberikan kepada Universitas Indonesia Hak Bebas Royalti Noneksklusif (Non-exclusive Royalty-Free Right) atas karya ilmiah saya yang berjudul:

Perancangan Sistem Pengawasan dan Kendali Aliran

Beserta perangkat yang ada (jika diperlukan). Dengan hak bebas royalti Noneksklusif ini Universitas Indonesia berhak menyimpan, mengalihmedia/formatkan, mengelola dalam bentuk pangkalan data (database), merawat dan mempublikasikan tugas akhir saya selama tetap mencantumkan nama saya sebagai penulis atau pencipta dan sebagai pemilik Hak Cipta.

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat di : Depok
Pada tanggal : 16 Juni 2009
Yang menyatakan

(Heru Cahyono)

ABSTRAK

Nama : Heru Cahyono
Program Studi : Teknik Elektro
Judul : Perancangan Sistem Pengawasan dan Kendali Aliran

Supervisory Control And Data Acquisition (SCADA) adalah perangkat lunak yang digunakan untuk pengawasan dan kendali suatu sistem yang dalam skripsi ini adalah sistem kendali aliran. Aliran baik fluida atau gas adalah salah satu parameter yang penting dalam industri untuk mendapatkan produk akhir yang dipersyaratkan. Sistem kendali aliran memanfaatkan *Programable Logic Controller* (PLC) sebagai pengendali dan satu atau beberapa komputer yang terpasang perangkat lunak *Human Machine Interface* (HMI) sebagai SCADA. Pemrograman PLC dengan menggunakan fungsi blok dan perancangan HMI dengan grafik yang merepresentasikan sistem kendali aliran dengan menampilkan semua parameter yang diperlukan untuk pengawasan dan kendali. Sistem kendali menggunakan metode Ziegler Zichols untuk penalaan parameter pengendali *Proportional Integral Derivative* (PID) sehingga didapatkan nilai parameter yang sesuai dengan kriteria desain untuk sistem kendali aliran. Dengan metode Ziegler Nichols masih harus dilakukan *fine tuning* untuk mendapatkan parameter pengendali sesuai kriteria desain.

Kata Kunci:
SCADA, PLC, HMI, pengendali PID, Metode Ziegler Nichols.

ABSTRACT

Name : Heru Cahyono
Study Program : electronics engineering
Title : Design for Flow Control and Monitoring System.

Supervisory Control And Data Acquisition (SCADA) is a software for monitoring and control of a system, in which this work a flow control system is used. Both fluid flow or gas flow are one of the crucial parameter in industries to achieve the requirement for final product. The flow control system used a Programable Logic Controller (PLC) as the controller and one or more computer which embedded with Human Machine Interface (HMI) software as SCADA. The PLC is programmed using function block and HMI configuration with graphic to represent flow control system by placing all text or figure interface that are needed for monitoring and control. The Ziegler Nichols tuning Method is used to tune the control parameters of the PID controller. After the parameters are set, the controller is retune using heuristic method to achieved the best response.

Key words:
SCADA, PLC, HMI, PID Controller, Ziegler Nichols Methode.

DAFTAR ISI

HALAMAN JUDUL……..…..i
HALAMAN PERNYATAAN ORISINALITAS……....…..ii
HALAMAN PENGESAHAN……..…....….iii
KATA PENGANTAR.....…….……...iv
LEMBAR PERSETUJUAN PUBLIKASI KARYA ILMIAH……..…..v
ABSTRAK/ABSTRACT..……...…..…vi
DAFTAR ISI……..vii
DAFTAR GAMBAR…….....ix
DAFTAR TABEL……...…..xiv
DAFTAR SINGKATAN……..……xv
DAFTAR ISTILAH.....……..…..xvii
DAFTAR LAMPIRAN…….....…xix

BAB 1 PENDAHULUAN.....1
1.1. LATAR BELAKANG……..…..1
1.2. PERUMUSAN MASALAH……...……..…..2
1.3. TUJUAN PENELITIAN……...……..3
1.4. PEMBATASAN MASALAH..…….....3
1.5. SISTEMATIKA PENELITIAN..……...…..3
1.6. SISTEMATIKA PENULISAN.……...…...4

BAB 2 LANDASAN TEORI……..……...5
2.1. AKUISISI DATA……..……..…….....5
2.2. SENSOR DAN TRANSDUSER……..…..6
2.2.1. Akurasi……..……..…..6
2.2.2. Sensitifitas……..……...……7
2.2.3. Repeatabilitas……..……..7
2.2.4. Range……..……...…7
2.2.5. Orifice Meter……..……...7
2.2.6. Katup Kendali (*Control Valve*)……...….12
2.2.7. Komunikasi Antar Peralatan……..……13
2.3. SISTEM KENDALI……...……...……....14
2.4. METODE KENDALI……...……..…16
2.8.1. Pengendali Proportional……..…18
2.8.2. Pengendali Proportional Derivative……...19
2.8.3. Pengendali Proportional Integral……....20
2.8.4. Pengendali Proportional Integral Derivative……..20
2.5. PIPING AND INSTRUMENTATION DIAGRAM (P&ID)……....21
2.6. PROGRAMABLE LOGIC CONTROLLER (PLC)…….22
2.6.1. Central Procesing Unit (CPU)……24
2.6.2. Memori……...25
2.6.3. Pemrograman PLC……26
2.6.4. Catu daya PLC……...26
2.6.5. Masukan-masukan PLC……27
2.6.6. Antarmuka Masukan……..28

2.6.7. Keluaran-keluaran PLC……..28
2.6.8. Antarmuka Keluaran……..29
2.6.9. Jalur Tambahan……..29
2.6.10. Menggunakan Piranti Masukan dan Keluaran……...29
2.7.10.1. Konsep Dasar……..30
2.7.10.2. Jalur-jalur Masukan…….31
2.7.10.3. Jalur-jalur Keluaran……32
2.6.11. Modul-modul PLC…….33
2.7. HUMAN MACHINE INTERFACE (HMI)……..37

BAB 3 PERANCANGAN SISTEM……38
3.1. PERANCANGAN PLANT……..……38
3.2. PERANCANGAN PERANGKAT LUNAK……...…….39
1.2.1. Desain Perangkat lunak HMI…….39
1.2.2. Pemrograman PLC……...……..…45

BAB 4 ANALISA SISTEM KENDALI……..74
4.1. TUNING PARAMETER PENGENDALI PID……74
4.2. METODE REAKSI PROSES……..…….……....74
4.2.1. Metode Tangen……..75
4.2.2. Metode Perbaikan Tangen.……...…….…78
4.2.3. Metode Smith……..…….…….…81
4.3. METODE OSILASI (CONTINUOUS CYCLING)……...…..83

BAB 5 KESIMPULAN……..…….…….87

DAFTAR REFERENSI.……..…..88
LAMPIRAN……..……...…....…89

DAFTAR GAMBAR

Gambar 2.1. Blok diagram sistem akuisisi data……..……..…..5

Gambar 2.2. Proses perbedaan tekanan sebagai flow meter pada *orifice*……8

Gambar 2.3. Plat *orifice*……...….11

Gambar 2.4. Katup kendali (*Control valve*).....13

Gambar 2.5. Blok diagram loop kendali proses……15

Gambar 2.6. Blok diagram sistem kendali loop tertutup dengan pengendali PID.16

Gambar 2.7. Blok diagram sistem kendali loop tertutup dengan pengendali P.....18

Gambar 2.8. Blok diagram sistem kendali loop tertutup dengan pengendali PD..19

Gambar 2.9. Blok diagram sistem kendali loop tertutup dengan pengendali PI....20

Gambar 2.10. Blok diagram sistem kendali loop tertutup dengan pengendali PID.....21

Gambar 2.11. Elemen-elemen dasar PLC.....24

Gambar 2.12. Rangkaian antarmuka masukan PLC.....28

Gambar 2.13. Rangkaian antarmuka keluaran PLC…….……..29

Gambar 2.14. Ilustrasi terminal COMM……....30

Gambar 2.15. Menghubungkan sensor keluaran *sinking* dengan masukan *sourcing*.....31

Gambar 2.16. Menghubungkan sensor keluaran *sourcing* dengan masukan *sinking*.....31

Gambar 2.17. Menghubungkan beban keluaran dengan keluaran PLC tipe *sinking*.....32

Gambar 2.18. Menghubungkan beban keluaran dengan keluaran PLC tipe *soucing*.....32

Gambar 2.19. Modul *power supply*.....33

Gambar 2.20. Modul CPU……..…..…..……34

Gambar 2.21. Modul digital input……..34

Gambar 2.22. Modul digital output……...….35

Gambar 2.23. Modul analog input.....35

Gambar 2.24. Modul analog output……..…….36

Gambar 2.25. *Base modul*…….....…36

Gambar 2.26. *Human Machine Interface* (HMI)…….…...…37

Gambar 3.1. P&ID pengendali aliran.....38

Gambar 3.2. Arsitektur SCADA pengendali aliran.....39

Gambar 3.3. (a) *Icon* OPC (b) Jendela utama OPC.....39

Gambar 3.4. Mengkonfigurasi OPC.....40

Gambar 3.5. Kotak dialog *Topic Definition*……...…....40

Gambar 3.6. Kotak dialog OPCLink Topic Definition…….....41

Gambar 3.7. Kotak dialog OPC *browser*.....42

Gambar 3.8. Kotak dialog *Create New Application*…….….43

Gambar 3.9. Jendela *Intouch Application Manager*…….….44

Gambar 3.10. Kotak dialog *window properties*……...…..44

Gambar 3.11. Kotak dialog OPC *tag Creator*……...…45

Gambar 3.12. Diagram alir Pembuatan program PLC……...46

Gambar 3.13. Jendela menu pilihan *project* baru……..……47

Gambar 3.14. Project *tree windows*……...……47

Gambar 3.15. Kotak dialog *target*……..48

Gambar 3.16. Device *label definition*……...….48

Gambar 3.17. Kotak dialog *insert*……..……..…..49

Gambar 3.18. Nama program pada *project tree*……...…..49

Gambar 3.19. Gambar lembar kerja (*worksheet*)……...50

Gambar 3.20. *Combo box Edit Wizard*…….....……50

Gambar 3.21. Memasukkan pengendali PID ke lembar kerja…….…..51

Gambar 3.22. Fungsi blok analog input……..…..……52

Gambar 3.23. Fungsi blok analog output.....52

Gambar 3.24. (a) *Icon* untuk menyambung object
(b) Analog input dan pengendali PID…..……...…...…52

Gambar 3.25. Hasil penyambungan analog input, pengendali dan analog output.53

Gambar 3.26. Penyambungan analog input dengan pengendali……..…….….53

Gambar 3.27. Penugasan analog input ke *hardware*…….....54

Gambar 3.28. (a) fungsi blok analog input (b) Fungsi blok pengendali……..…..54

Gambar 3.29. Penugasan pembacaan kembali nilai output……...….....55

Gambar 3.30. Hasil penempatan *variable* pembacaan kembali.....55

Gambar 3.31. Penugasan pembacaan kembali ke pengendali.....56

Gambar 3.32. Hasil dari penempatan *variable* pembacaan kembali…….....56

Gambar 3.33. Pemeriksaan *variable*……..……....56

Gambar 3.34. Penempatan *variable* untuk akses parameter dan *engineering* parameter……...…….57

Gambat 3.35. Letak pin akses parameter dan *engineering* parameter……..….…58

Gambar 3.36. Pembuatan *variable* untuk engineering parameter……...….58

Gambar 3.37. Memasukkan *variable* untuk *engineering* parameter…….....58

Gambar 3.38. *Variable engineering* parameter……..……....59

Gambar 3.39. *Variable comment* untuk *engineering* parameter……...….59

Gambar 3.40. Penempatan *variable comment* pada *worksheet*……..…...60

Gambar 3.41. Pemberian *comment* ‘Flow Control’ untuk *engineering* parameter.....60

Gambar 3.42. *Variable* akses parameter……..……..61

Gambar 3.43. Pembuatan v*ariable* untuk akses parameter……...…61

Gambar 3.44. Hasil akhir dari fungsi blok loop kendali.....62

Gambar 3.45. Fungsi blok untuk mode pengendali (Auto atau Manual).....62

Gambar 3.46. Penugasan program……...….63

Gambar 3.47. Pemberian nama *task*……...……63

Gambar 3.48. Penghapusan *task* yang tidak perlu……..…...…64

Gambar 3.49. Kotak dialog *software wiring*.....64

Gambar 3.50. Kotak dialog *software wiring wizard*.....65

Gambar 3.51. *Software wiring* dalam *project tree*……...…..….65

Gambar 3.52. Mengubah nilai GS_NFIO_DISCONNF.....66

Gambar 3.53. Proses kompilasi……..……...….67

Gambar 3.54. Dialog *project control*……...…….67

Gambar 3.55. Kotak dialog *download*……..…..68

Gambar 3.56. Mengubah status PLC…….……68

Gambar 3.57. Mode *debug* PLC……..…..69

Gambar 3.58. Menampilkan *watch window*……..……70

Gambar 3.59. Menambahkan akses parameter pengendali ke *watch window*…..71

Gambar 3.60. Akses parameter dalam *watch window*……...…....71

Gambar 3.61. Mengubah nilai *set value*…….……72

Gambar 3.62. Mengubah mode pengendali ke auto……...…72

Gambar 4.1. Kurva perubahan PV akibat pengubahan MV……...…74

Gambar 4.2. (a) Kurva garis singgung terhadap PV metode tangen…….....76
(b) Kurva karakteristik plant metode tangen.……...…….76

Gambar 4.3. Respon sistem pengendali PI metode tangen.……..….77

Gambar 4.4. Respon sistem pengendali PID metode tangen.……...…….78

Gambar 4.5. (a) Kurva garis singgung terhadap PV metode perbaikan tangent....78
(b) Kurva karakteristik plant metode perbaikan tangent……...……79

Gambar 4.6. Respon sistem pengendali PI metode perbaikan tangen.……..80

Gambar 4.7. Respon sistem pengendali PID metode perbaikan tangen……..…..81

Gambar 4.8. (a)Kurva garis singgung terhadap PV metode Smith……...….81
(b) Kurva karakteristik plant metode Smith……..……...82

Gambar 4.9. Respon sistem pengendali PI metode Smith……...…..83

Gambar 4.10. Respon sistem pengendali PID metode Smith……...…….83

Gambar 4.11. (a) Respon sistem pada saat tuning…….……..……..84
(b) Respon sistem pada SCADA saat tuning…..……..…..….84

Gambar 4.12. Respon sistem pengendali PI metode Osilasi…….....…...85

Gambar 4.13. Respon sistem pengendali PID metode Osilasi……...……...86

DAFTAR TABEL

Tabel 2.1. Respon pengendali PID terhadap perubahan konstanta.....17

Tabel 4.1. *Ziegler-Nichols* PID parameter menggunakan metode reaksi proses...75

Tabel 4.2. *Ziegler-Nichols* PID parameter menggunakan metode osilasi.....84

DAFTAR SINGKATAN

CPU	Central Processing Unit
DAQ	Data Acquisition
DDE	Dinamic Data Exchange
DDL	Dinamic Data Library
EFD	Engineering Flow Diagram
GND	Ground
FC	Flow Controller
FCV	Flow Control Valve
HMI/MMI	Human Machine Interface/Man Machine Interface
I/O	Input/Output
MV	Manipulated Value
Matlab	Matematika Laboratory
MFD	Mechanical Flow Diagram
LAN	Local Area Networks
ODBC	Open Data Based Connectivity
OPC	OLE for Process Control
P&ID	Piping & Instrumentation Diagram
OLE	Open Link Embedded
OS	Operational system
UFD	Utility Flow Diagram
PLC	Programable Logic Controller
PID	Proportional Integral Derivative
PC	Personal Computer
PV	Proses Variable
PFD	Process Flow Diagram
RTDB	Real Time Databased
RTD	Resistive Temperature Detector
SCI	Serial Communication Interface
SV	Set Value
VCC	Common-Collector Voltage

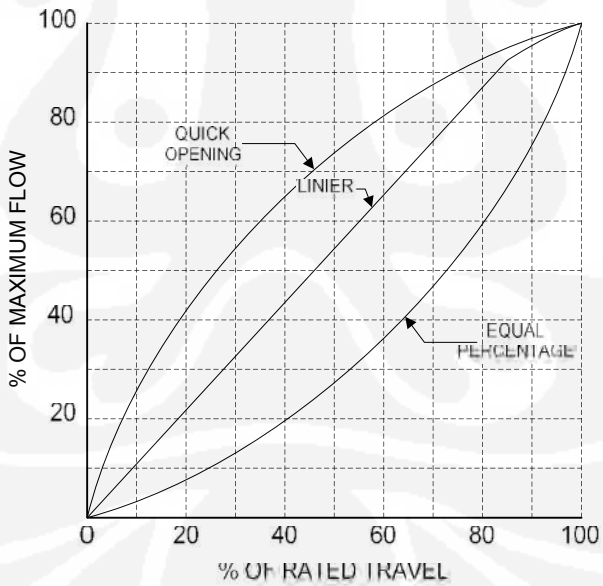
DAFTAR ISTILAH

Aktuator : Pneumatik, hydraulic, atau peralatan menggunakan listrik yang menyuplai daya dan menggerakkan buka tutup katup.

Pengendali : Peralatan yang beroperasi secara automatic dengan penggunaan algoritma tertentu untuk meregulasi variabel terkendali. Input pengendali menerima informasi tentang status variabel proses dan kemudian memberikan sinyal output yang tepat ke elemen kendali akhir.

Dead time : Interval waktu (td) dalam ketidak adaan respon sistem adalah didapatkan kecil (biasanya 0.25% - 5%) input step. Ini diukur dari waktu input step adalah inisialisasi pada respon pertama yang terdeteksi pada sistem untuk diuji. Dead time bisa diterapkan pada katup atau pada keseluruhan proses.

Karakteristik equal percentage : karakteristik aliran yang melekat untuk penambahan sama dari *rate travel*, akan secara ideal memberikan perubahan persentase sama dari koefisien aliran (Cv), seperti gambar kriteria katup dibawah ini



Gambar karakteristik katup

Elemen kendali Akhir : peralatan yang menerapkan strategi kendali yang ditentukan oleh output pengendali. Ketika elemen kendali akhir bisa diukur, variable kecepatan mengatur pompa atau peralatan sakelar on-off, elemen kendali

akhir yang paling umum pada proses control industri adalah katup kendali (*control valve)*. Katup kendali memanipulasi lairan fluida seperti gas, uap air, air atau senyawa kimia.

Orde pertama : istilah yang merujuk hubungan yang dinamis antara input dan output peralatan. Sistem orde pertama atau alat adalah satu yang hanya memiliki satu peralatan penyimpan tenaga dan hubungan *transient* dinamik antara input dan output ini di cirikan oleh sikap kuadratik.

Gain : adalah perbandingan *magnitude* pada perubahan output yang diberikan sistem atau peralatan untuk *magnitude* pada perubahan input yang menyebabkan perubahan output. Gain memiliki dua komponen : gain statik dan gain dinamik. Gain static adalah gain yang berhubungan antara input dan output dan adalah indikator ketepatan dengan input dapat melakukan perubahan pada output ketika sistem atau peralatan pada kondisi *steady state*. Sensitifitas terkadang digunakan untuk pengertian gain statik. Gain dinamik adalah gain hubungan antara input dan output ketika sistem pada kondisi pergerakan atau perubahan. Gain dinamik adalah fungsi dari frekuensi atau *rate* dari perubahan input.

Hysteresis : perbedaan maksimum dalam nilai output untuk setiap nilai input tunggal selama kalibrasi, meniadakan *error* karena *dead band*.

Inherent Characteristic : Hubungan antara koefisien aliran dan perjalanan penutup (disk) bergerak pada posisi tertutup pada *rate travel* dengan konstan *pressure drop* pada katup.

DAFTAR LAMPIRAN

Lampiran 1. Program fungsi blok pengendali PID.....89

Lampitan 2. Program *software wiring* sebagai pengganti plant.....91

BAB 1
PENDAHULUAN

1.1. LATAR BELAKANG

Untuk melakukan pengawasan dan kendali suatu *plant* tidak harus ke *plant* setiap saat. Dengan adanya sistem pengawasan dan kendali suatu *plant* secara keseluruhan pada area tertentu yang sering di kenal sebagai *Supervisory Control and Data Acquisition* (SCADA), bisa didapatkan semua status peralatan yang ada di *plant*, misalnya indikasi buka tutup katup, tekanan, suhu dan kecepatan aliran pada plant. Selain status peralatan yang ada di plant juga bisa dilakukan aksi kendali untuk memperoleh produk akhir sesuai dengan keinginan.

Dengan sistem akuisisi data bisa diambil data dari *plant* untuk di tampilkan pada komputer. Data bisa berupa bit 0 atau 1 sebagai representatif dari indikasi sebuah sakelar, atau sebuah sinyal analog 4-20mA. Sinyal analog pada umumnya digunakan untuk tekanan, level, kecepatan dan juga sinyal untuk mengendalikan katup. Bisa juga sinyal berupa lebar pulsa atau sinyal berupa frekuensi dari pulsa.

Supervisory Control and Data Acquisition (SCADA) bukanlah merupakan sistem kendali saja tetapi fungsinya juga difokuskan pada pengawasan atau *monitoring*. SCADA adalah murni paket perangkat lunak yang secara umum menggunakan *Programable Logic Controller* (PLC) sebagai pengendali ataupun menggunakan modul pengendali lainnya.

Sistem SCADA tidak hanya digunakan untuk proses industri seperti, pembuatan baja, pembangkit dan distribusi tenaga, industri kimia tetapi juga pada fasilitas eksperimen seperti peleburan nuklir. Kisaran ukuran *plant* 1000 sampai dengan 10.000 I/O. Sistem SCADA berkembang telah sangat pesat dan sekarang memasuki pasar industri dengan ukuran mencapai 100.000 I/O. Sistem SCADA beroperasi mengunakan DOS, VMS dan UNIX. Namun pada beberapa tahun terakhir ini semua pembuat SCADA telah beralih ke sistem operasi NT, Windows XP, Windows Server 2003 dan juga beberapa ke sistem operasi LINUX.

Ada dua layer lapisan dalam sistem SCADA yakni "*Layer Client*" yakni untuk interaksi *Human Machine Interface* (HMI) dan "*Data Server Layer*" yang

menangani hampir semua aktifitas proses pengendalian data. Server data berkomunikasi dengan peralatan dilapangan melalui pengendali proses. Pengendali proses seperti PLC tersambung dengan salah satu server data langsung melalui jaringan atau *fieldbuss*. Server data saling tersambung ke *station client* melalui *Ethernet LAN*.

SCADA bisa melakukan tugas yang banyak tergantung RTDB (*Real Time DataBased*) di tempatkan pada satu atau lebih server. Server bertanggung jawab untuk akuisisi data dan menangani (alarm, perhitungan, pengambilan dan pengarsipan) set parameter. Sangat mungkin untuk keberadaan server secara langsung untuk tugas-tugas khusus, contoh: *history*, *data logger*, *alarm*.

SCADA adalah sistem yang sangat diperlukan untuk pengawasan *plant* dengan resiko kecelakaan tinggi. SCADA adalah pilihan terbaik untuk masalah itu jadi semua permasalahan baik itu akuisisi data dan pengendalian *plant* dilakukan pada area yang dianggap cukup aman jika terjadi kecelakaan di *plant*. Untuk pengendali dalam skripsi ini akan di gunakan PLC dan untuk data akuisisi menggunakan perangkat lunak HMI.

1.2. PERUMUSAN MASALAH

Aliran baik fluida atau gas adalah salah satu dari parameter proses industri yang sangat penting untuk dikendalikan sehingga didapatkan produk akhir yang dipersyaratkan.

Sistem kendali menggunakan PLC sebagai pengendali dan perangkat lunak HMI sebagai SCADA. Pemrograman PLC sebagai pengendali aliran dengan metode kendali PID dengan menggunakan fungsi blok. Perancangan HMI dengan grafik yang merepresentasikan sistem kendali aliran dengan menampilkan semua parameter baik berupa text atau gambar yang diperlukan untuk pengawasan dan kendali aliran.

Metode Ziegler Nichols digunakan untuk penalaan pengendali PID sehingga didapatkan parameter pengendali sesuai dengan kriteria desain sistem kendali aliran.

1.3. TUJUAN PENELITIAN

Tujuan penelitian yang dilakukan pada skripsi ini adalah untuk merancang suatu sistem pengawasan dan kendali dengan memanfaatkan PLC sebagai pengendali dan sebuah komputer dengan perangkat lunak HMI berfungsi melakukan akuisisi data dengan detail sebagai berikut :

1. Dapat dikonfigurasi menjadi sistem kendali loop tertutup dengan memanfaatkan PLC sebagai pengendali,
2. Dapat mengkonfigurasi sistem pengawasan dan kendali dengan memanfaatkan perangkat lunak HMI,
3. Dapat dimanfaatkan untuk melakukan tuning pengendali sehingga didapatkan sistem kendali yang sesuai dengan kriteria desain.

1.4. PEMBATASAN MASALAH

Pembahasan dalam skripsi ini di batasi pada perancangan sistem pengawasan dan kendali sistem aliran, dengan detail sebagai berikut:

1. Pembahasan mengenai sistem kendali dengan memanfaatkan PLC *Stardom* produk *Yokogawa* sebagai pengendali
2. Pembahasan mengenai sistem akuisisi data dengan perangkat lunak HMI/MMI menggunakan *Intouch* produk *Wonderware*.
3. Pembahasan tentang metode *Ziegler-Nichols* tuning parameter pengendali.

1.5. SISTEMATIKA PENELITIAN

Penelitian yang didefinisikan pada skripsi ini dilakukan meliputi:

1. Pendekatan tinjauan pustaka yaitu melakukan studi literature dari buku-buku pustaka, sumber-sumber di internet, dan buku-buku manual dari perangkat yang digunakan,
2. Pendekatan diskusi dengan pembimbing skripsi yang berkaitan dengan topik bahasan skripsi,
3. Perancangan dan pembuatan perangkat keras dan perangkat lunak,
4. Pembuatan program pada komputer dan pada PLC, untuk melakukan pengujian terhadap batasan masalah yang dibahas pada skripsi ini.

1.6. SISTEMATIKA PENULISAN

Guna membuat pembahasan masalah dalam skripsi menjadi lebih sistematis, skripsi ini dibagi menjadi beberapa bab antara lain sebagai berikut:

Bab pertama, pendahuluan, memuat latar belakang masalah, tujuan, pembatasan masalah, metodologi penelitian dan sistematika penulisan. **Bab Kedua**, landasan teori, menggambarkan landasan pengetahuan mengenai sistem akuisisi data, sistem kendali, metode kendali, PLC dan hal-hal lain yang digunakan pada skripsi ini. **Bab Ketiga**, perancangan, menggambarkan langkah pengerjaan skripsi ini mulai dari perancangan sistem hingga pengaturan konfigurasi yang digunakan. **Bab Keempat**, analisa, berisi analisa dan pembahasan mengenai analisa sistem kendali yang menjadi fokus penelitian. **Bab Kelima**, kesimpulan, bab ini merangkum kesimpulan pembahasan skripsi ini.

BAB 2
LANDASAN TEORI

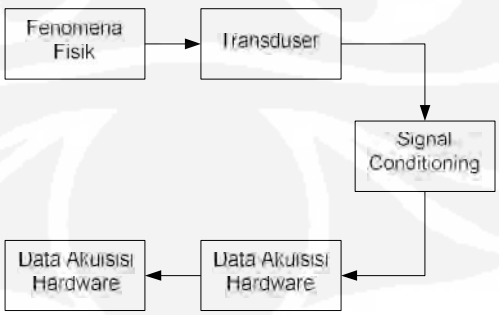
1.7. AKUISISI DATA

Akuisisi data adalah proses pembacaan besaran fisis alam (suhu, tekanan, kecepatan aliran, dll) yang di ubah ke bentuk sinyal listrik, diukur selanjutnya diubah dalam format digital untuk pemprosesan, analisa dan disimpan pada komputer. Dalam penggunaannya, sistem akuisisi data (DAQ) didesain bukan hanya untuk memperoleh data, tetapi juga untuk melakukan aksi kendali. Dalam definisi sistem DAQ perlu di kembangkan lagi meliputi aspek kendali pada sistem secara keseluruhan.

Sistem akuisisi data dan kendali mungkin terdiri dari banyak jenis perangkat keras dari vendor yang berbeda. Selanjutnya sistem integrator mengintegrasikan semua komponen kedalam satu sistem. Elemen dasar dari sistem akuisisi data seperti blok diagram pada gambar 2.1 sebagai berikut:

- Sensor dan Transduser
- Pengkabelan (*wiring*)
- Pengkondisi sinyal (*Signal conditioning*)
- Perangkat keras akuisisi data
- PC dan Sistem Operasi (*Operatingl System*)
- Perangkat lunak akuisisi data

Masing-masing Elemen dari keseluruhan sistem berperan dalam memperoleh pengukuran yang teliti dan pengumpulan data dari proses atau fenomena fisik yang akan di monitor.



Gambar 2.1. Blok Diagram Sistem Akuisisi Data

2.2. SENSOR DAN TRANSDUSER

Transduser adalah peralatan yang mengubah suatu bentuk energi kebentuk energi yang lain. Sensor dan transduser berfungsi sebagai antarmuka antara fenomena fisik dan sistem akuisisi data dengan mengubah fenomena fisik menjadi sinyal listrik kemudian melalui pengkondisi sinyal diteruskan kepada perangkat keras akuisisi data, agar bisa menerima sinyal tersebut. *Transduser* yang ada dapat membaca hampir semua pengukuran fisik dan menyediakan sinyal output listrik seperti contoh: *thermocouple*, *resistive temperature detector* (RTD), *thermistor* dan IC sensor merubah suhu ke bentuk sinyal digital.

Ada dua kategori dari transduser yakni:

- *Transduser Aktif* mengubah energi bukan listrik kedalam bentuk sinyal listrik. *Transduser* ini tidak memerlukan catu daya dari luar untuk beroperasi. *Thermocouple* adalah salah satu contohnya.
- *Transduser Pasif* mengubah nilai komponen rangkaian listrik, seperti resistor, induktor, kapasitor, merujuk pada perubahan kuantitas fisik untuk di ukur, seperti *strain gauge* dan *pressure transduser* mengukur gaya dan tekanan, tipe lain dari *transduser* yang berfungsi untuk mengukur perpindahan, baik linier atau sudut, kecepatan dan percepatan, cahaya, bahan kimia, tegangan, arus, hambatan atau pulsa. Dalam masing-masing kasus, sinyal listrik yang dihasilkan adalah proporsional dengan kuantitas fisik yang diukur.

Transduser dikelompokkan berdasarkan pada kuantitas fisik yang diukur, karakteristik *transduser* yang perlu diperhatikan dalam pemilihan tipe *transduser* adalah sebagai berikut:

- Akurasi
- Sensitifitas
- Repeatabilitas
- Range

2.2.1. Akurasi

Akurasi *transduser* mendiskripsikan seberapa dekat pengukuran dengan nilai sebenarnya pada proses variable yang diukur. Hal ini menunjukkan

maksimum *error* yang mungkin terjadi pada pengukuran setiap nilai pada *range transduser* yang digunakan. Vendor selalu menyediakan nilai akurasi transduser dengan nilai % error pada *operational range transduser*, seperti ± 1% antara 20°C sampai 120°C.

2.2.2. Sensitifitas

Sensitifitas didefinisikan sebagai jumlah perubahan sinyal output *transduser* untuk perubahan yang spesifik dalam variabel input yang diukur. Tingginya sensitifitas peralatan seperti *thermistor*, mungkin mengubah tahanan misalnya 5% per °C, saat peralatan dengan sensitifitas yang rendah seperti *thermocouple* mungkin menghasilkan tegangan keluaran itu berubah hanya 5µV per °C.

2.2.3. Repeatibility

Jika dua atau lebih pengukuran dibuat pada proses variabel pada keadaan yang sama, *repeatability transduser* mengindikasikan seberapa dekat pengulangan dari pengukuran dapat memberikan hasil ukur yang sama. Kemampuan untuk membangkitkan output yang hampir sama pada besaran input yang sama.

2.2.4. Range

Transduser selalu dibuat untuk beroperasi pada rentang ukur yang spesifik.

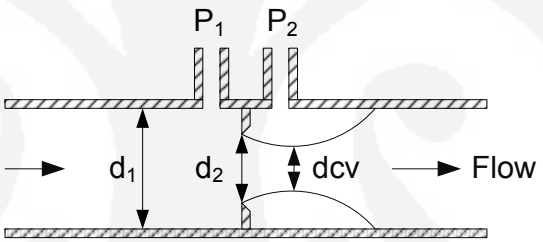
2.2.5. Orifice Meter.

Flow meter head beroperasi dengan prinsip penempatan pembatas pada jalur yang akan menyebabkan head perbedaan tekanan. Perbedaan tekanan yang disebabkan oleh *head* diukur dan diubah menjadi pengukuran aliran. Penerapan industri, *flow meter head* menggabungkan sistem pengiriman *pneumatik* atau listrik untuk pembacaan kecepatan aliran. Secara umum indikasi instrumentasi akar kuadrat perbedaan tekanan dan menampilkan kecepatan aliran pada indikator.

Ada dua elemen dalam *head flow meter*; elemen utama adalah pembatas dalam lintasan dan elemen kedua adalah alat pengukur perbedaan tekanan. Gambar 2.2. memperlihatkan operasi dasar dari *head flow meter*.

Perbedaan yang diakibatkan oleh pembatas pada line seperti *orifice* ini diukur oleh *manometer mercury* atau detektor perbedaan tekanan. Dari pengukuran ini kecepatan aliran ditentukan dengan persamaan hukum fisika.

Head flow meter pada kenyataannya cenderung mengukur kecepatan aliran volumetrik dari pada kecepatan aliran masa. Kecepatan aliran masa dapat dihitung dari kecepatan aliran volumetrik berdasarkan suhu atau tekanan. Suhu dan tekanan mengakibatkan berat jenis fluida sehingga masa fluida mengalir melewati titik tertentu. Jika sinyal kecepatan aliran volumetrik adalah mengkompensasi perubahan suhu atau tekanan, sinyal kecepatan aliran masa yang sesungguhnya bisa di dapatkan. Dalam termodinamika mendeskripsikan suhu, berat jenis adalah berbanding terbalik, ketika tekanan dan berat jenis adalah proporsional.



Gambar. 2.2. Proses perbedaan tekanan sebagai pengukur aliran pada *orifice*

$$P_1 + \frac{1}{2} \cdot \rho \cdot V_1^{\,2} = P_2 + \frac{1}{2} \cdot \rho \cdot V_2^{\,2} \qquad (1)$$

atau

$$P_1 - P_2 = \frac{1}{2} \cdot \rho \cdot V_2^{\,2} - \frac{1}{2} \cdot \rho \cdot V_1^{\,2} \qquad (2)$$

dengan persamaan kontinuity

$$Q = A_1 \cdot V_1 = A_2 \cdot V_2 \text{ atau } V_1 = \frac{Q}{A_1} \text{ dan } V_2 = \frac{Q}{A_2} \qquad (3)$$

$$P_1 - P_2 = \frac{1}{2} \cdot \rho \cdot \left(\frac{Q}{A_2}\right)^2 - \frac{1}{2} \cdot \rho \cdot \left(\frac{Q}{A_1}\right)^2 \qquad (4)$$

$$Q = A_2 \sqrt{\frac{2(P_1 - P_2)/\rho}{1-(A_2/A_1)^2}} \quad (5)$$

$$Q = A_2 \sqrt{\frac{1}{1-(d_2/d_1)^4}} \sqrt{2(P_1 - P_2)/\rho} \quad (6)$$

$$Q = C_d A_2 \sqrt{\frac{1}{1-\beta^2}} \sqrt{2(P_1 - P_2)/\rho} \text{ dimana } C = \frac{C_d}{\sqrt{1-\beta^4}} \quad (7)$$

Jadi untuk persamaan aliran volumetrik pada fluida adalah

$$Q = C \cdot A_2 \sqrt{2(P_1 - P_2)/\rho} \quad (8)$$

Persamaan untuk mass flow pada fluida adalah

$$\dot{m} = \rho \cdot Q = C \cdot A_2 \sqrt{2\rho(P_1 - P_2)} \quad (9)$$

Dimana :

Q = aliran volumetrik (m^3/s)

$\dot{m}$ = *mass flow* (kg/s)

C_d = *koefisien discharge*

C = koefisien aliran orifice

A_1 = luas penampang pipa (m^2)

A_2 = luas penampang lubang orifice (m^2)

d_1 = diameter pipa (m)

d_2 = diameter lubang orifice (m)

$$\beta = \frac{d_2}{d_1} \quad (10)$$

V_1 = kecepatan pada *upstream* (m/s)

V_2 = kecepatan pada *downstream* (m/s)

P_1 = tekanan pada *upstream* ($kg/(m \cdot s^2)$)

P_2 = tekanan pada *downstream* ($kg/(m \cdot s^2)$)

ρ = *density fluida* (kg/m^3)

Dan untuk persamaan aliran masa pada gas bertekanan adalah

$$\dot{m} = CA_2 P_1 \sqrt{\frac{2M}{ZRT_1}\left(\frac{k}{k-1}\right)\left[\left(\frac{P_2}{P_1}\right)^{2/k} - \left(\frac{P_2}{P_1}\right)^{(k+1)/k}\right]} \qquad (11)$$

Dimana :

k = *specific heat ratio* (c_p / c_v)

m = kecepatan aliran masa (kg/s)

C = koefisien aliran orifice

A_2 = luas penampang lubang orifice (m^2)

ρ_1 = berat jenis gas pada *upstream* (kg/m^2)

P_1 = tekanan gas pada *upstream* ($kg/m.s^2$)

P2 = tekanan *downstream* pada lubang orifice ($kg/m.s^2$)

M = masa molekul gas (kg/mol)

R = konstanta hukum gas = 8.3145 J/(mol.K)

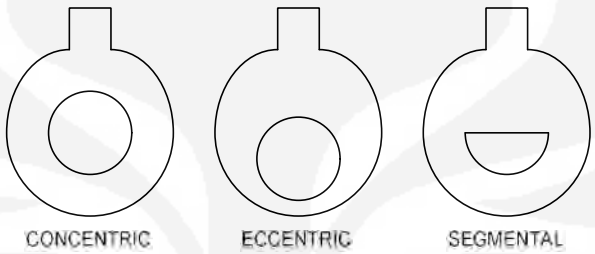
T_1 = suhu absolut gas *upstream* (K)

Z = faktor kempresible gas pada P_1 dan T_1

Persamaan diatas menggunakan luas penampang bukaan *orifice* hal ini tidak realistis menggunakan luas penampang terkecil pada *vena contracta*. Sebagai tambahan rugi-rugi akibat gesekan mungkin tidak bisa diabaikan dan *viskositas* dan afek *turbulence* akan ada. Sehingga koefisien *discharge* Cd diperkenalkan. Ada metode untuk menentukan koefisien tersebut. Parameter $\sqrt{1-\beta^4}$ sering dihubungkan sebagai faktor pendekatan kecepatan dan membagi koefisien *discharge* oleh parameter diatas dan menghasilkan koefisien aliran C. Ada juga metode untuk menentukan *koefisien* aliran sebagai fungsi β dan lokasi tapping sensor tekanan pada *downstream*. Sebagai perkiraan tingkat kekasaran *koefisien* aliran diasumsikan antara 0.60 dan 0.75. Untuk pendekatan pertama bisa menggunakan *koefisien* aliran 0.62. Orifice akan bekerja dengan baik bila dicapai dengan panjang *upstream* 20 s/d 40 diameter pipa.

Ada beberapa tipe dari *head flow meter* antara lain : *Orifice meter, Venturi tube, Dall flow tube* dan *pitot tube*. Pada pembahasan kali ini hanya dibahas mengenai *orifice meter*.

Plat *orifice* adalah cara termudah dalam pembatasan aliran yang digunakan dalam mendeteksi aliran yang paling ekonomis, dengan konstruksi plat datar dengan ketebalan 1/16 sampai ¼ inch. Itu secara normal dipasang antara sepasang *flange* dan di pasang langsung pada pipa lurus untuk menghilangkan gangguan pola aliran dari *fitting* dan katup.



Gambar 2..3. Plat Orifice

Tiga jenis plat *orifice* yang digunakan adalah *concentric*, *eccentric* dan *segmental*, seperti gambar 2.3. Plat *orifice concentric* adalah yang paling umum dari ketiga tipe itu. Seperti terlihat pada gambar 2.3, *orifice* adalah sama jaraknya (*concentric*) terhadap diameter dalam pipa. Aliran melalui tepi plat *orifice* dan akan terjadi perubahan kecepatan. Seperti zat cair/gas yang melalui *orifice*, zat tersebut terkumpul, dan kecepatan zat meningkat ke nilai maksimal. Pada titik ini, tekanan pada nilai minimum. Zat seakan terbagi di dalam pipa, kecepatan kembali turun pada nilai asal. Tekanan meningkat ± 60% sampai 80% dari nilai awal. Tekanan yang hilang tidak bisa didapatkan lagi oleh karena itu tekanan keluaran akan selalu kurang dari tekanan masukkan. Tekanan pada dua sisi *orifice* diukur, menghasilkan perbedaan tekanan yang sebanding dengan aliran.

Plat *Orifice Segmental* dan *eccentric* fungsinya identik dengan *orifice concentric*. Bagian lingkaran pada *orifice segmental* adalah *concentric* dengan pipa. Porsi *segmental* penahanan penghilangan *orifice* dari material asing pada sisi *upstream* pada *orifice* ketika di dipasang pada pipa horizontal. Tergantung pada tipe zat, bagian *segmental* di tempatkan pada salah satu di atas atau di bawah pada pipa horizontal untuk meningkatkan ketelitian pengukuran.

Plat *orifice eccentric* bergeser kedinding dalam pipa. Desain ini juga mencegah perusakan *upstream* dan ini digunakan dengan cara yang sama seperti

plat *orifice segmental*. Plat *orifice* memiliki dua daerah yang tidak menguntungkan yakni :

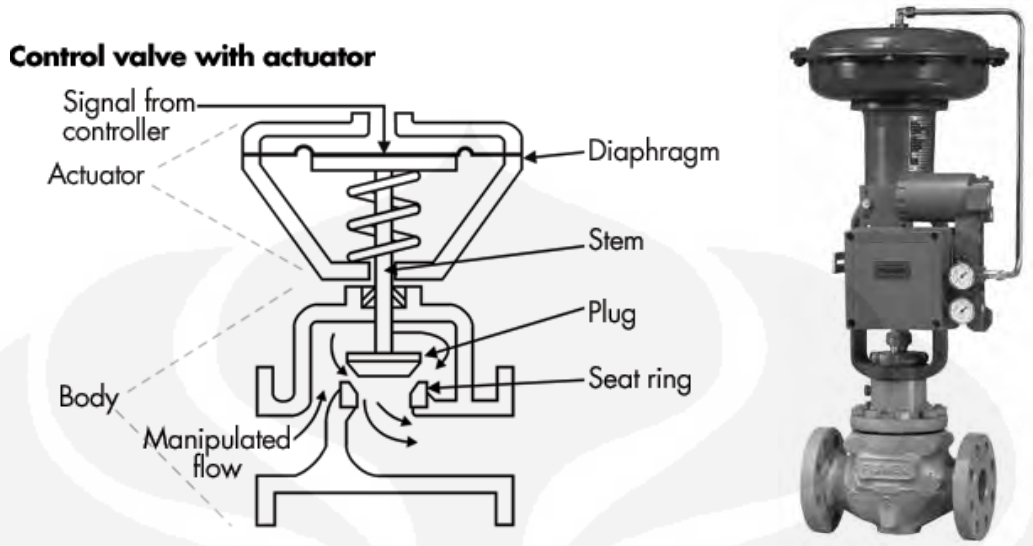
a. Menyebabkan penurunan tekanan yang permanen (tekanan keluar akan 60% sampai 80% dari tekanan masukkan),
b. *Orifice* adalah subyek untuk korosi yang akhirnya akan menyebabkan kurang telitinya pengukuran perbedaan tekanan.

2.2.6. Katup Kendali (*Control Valve*)

Plant-plant proses terdiri dari ratusan loop kendali semua tersambung menjadi satu untuk menghasilkan produk yang ditawarkan untuk dijual. Masing-masing loop kendali didesain untuk menjaga variabel proses yang penting, seperti aliran, tekanan, *level*, suhu dan sebagainya. Dengan kisaran operasi yang di perbolehkan untuk menentukan kualitas dari produk akhir. Masing-masing loop menerima dan secara internal tercipta gangguan yang mengakibatkan gangguan variabel proses dan interaksi dari loop yang lain dalam jaringan yang mengakibatkan gangguan yang mempengaruhi variabel proses.

Elemen kendali akhir yang paling umum dalam proses kendali pada industri adalah katup kendali (*control valve*). Katup kendali memanipulasi aliran fluida seperti gas, uap air, air atau senyawa kimia untuk mengkompensasi untuk gangguan dan tetap meregulasi variabel proses sedekat mungkin dengan *set point* yang diinginkan.

Untuk bisa menganalisa sistem kendali yang akan dibuat, harus bisa membuat fungsi alih dari *plant* yang akan dikendalikan, dalam kasus ini adalah katup kendali. Dengan elemen akhir berupa katup dan di gerakkan oleh aktuator, secara umum katup kendali dapat di lihat pada gambar 2.4.



Gambar 2.4. Katup kendali (*Control Valve*)

Model matematis untuk katup kendali bisa didapatkan dengan metode reaksi proses dari *Ziegler Nichols*, hal itu dibahas pada bab 4 analisa sistem kendali. Dari Metode *Ziegler Nichols* didapatkan persamaan matematis orde 1 sebagai berikut:

$$G(s) = \frac{1}{5s+1} \qquad (14)$$

Dengan kriteria perancangan sebagai berikut :

1. Memiliki *rise time* yang cepat
2. *Overshoot* sekecil mungkin
3. Tidak memiliki *steady stare error*

2.2.7. Komunikasi Antar Peralatan

Pengkabelan dilapangan merupakan perwujudan sambungan dari *transduser* dan sensor ke perangkat keras pengkondisi sinyal dan/atau perangkat keras akuisisi data. Ketika pengkondisi sinyal dan/atau perangkat keras akuisisi data berada jauh dari PC, kemudian pengkabelan diperlukan untuk menyediakan hubungan antara perangkat keras dan host komputer. Komunikasi antarmuka bisa menggunakan RS-323 atau RS-485, dan juga menggunakan *ethernet LAN*.

Pengkabelan sebagai komunikasi untuk sebuah sistem yang kompleks, itu sangat rentan terhadap efek *noise* dari luar. *Grounding* dan pelindung kabel pada

kabel komunikasi adalah hal yang sangat penting untuk mengurangi efek *noise*. Komponen pasif pada sistem akuisisi data dan kendali ini sering diabaikan sebagai komponen yang penting, hasilnya sistem menjadi tidak akurat atau tidak *reliable* karena teknik pengkabelan yang salah.

1.8. SISTEM KENDALI

Sistem kendali adalah suatu sistem yang bertujuan untuk mengendalikan suatu proses agar output yang dihasilkan dapat dikendalikan sehingga tidak terjadi kesalahan. Kendali loop tertutup mungkin digunakan pada beberapa peralatan untuk menyelesaikan kendali variabel proses. Loop akan menampung paling tidak empat elemen-elemen dasar antara lain :

1. Pengukuran variabel proses.

   Sensor, yang umum dikenal sebagai *transmitter*, mengukur beberapa variable pada proses seperti suhu, level cairan, tekanan atau kecepatan dan mengubah hasil pengukuran menjadi sinyal (4-20mA) untuk ditransmisikan ke pengendali atau sistem kendali.

2. Metode kendali (*algoritma control*)

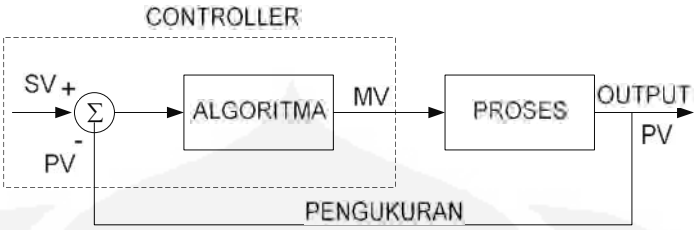
   Algoritma matematika dalam sistem kendali di jalankan pada periode waktu yang sama (setiap detik atau lebih cepat) untuk menghitung sinyal output untuk ditransmisikan ke elemen akhir.

3. Elemen kendali akhir

   Katup, Pengendali kecepatan motor atau peralatan lain yang menerima sinyal dari pengendali dan memanipulasi proses, biasanya dengan merubah kapasitas aliran suatu material.

4. Proses

   Respon proses untuk perubahan variabel yang dimanipulasi dengan hasil perubahan dalam variabel yang diukur. Dinamika respon proses adalah faktor utama dalam pemilihan parameter yang digunakan dalam algoritma kendali.



Gambar 2.5. Blok diagram loop pengendali proses

Tujuan pokok penggunaan kendali otomatis adalah tercapainya produksi lebih ekonomis. Penghematan dicapai dalam beberapa cara yang berbeda:

1. Menurunkan biaya buruh.
2. Mengeliminasi atau mengurangi kesalahan manusia.
3. *Improvisasi* kualitas proses.
4. Mengurangi ukuran dari peralatan proses dan jumlah area yang dibutuhkan.
5. Menyediakan tingkat keamanan yang tinggi dalam operasi.

Proses kendali adalah mengukur variabel proses (PV), pembandingan variabel dengan mengacu pada *setpoint/set value* (SV) dan untuk mendapatkan nilai manipulasi (MV) sehingga proses akan tetap merujuk pada nilai *set point*.

Pada kenyataannya proses menampung banyak variabel yang membutuhkan nilai pada *set point* (SV) dan banyak variabel yang bisa dimanipulasi. Biasanya masing-masing variabel kendali mungkin hasil pengaruh dari lebih dari satu variabel yang dimanipulasi dan masing-masing variabel yang dimanipulasi mungkin mengakibatkan lebih dari satu variabel yang terkendali. Oleh karena itu dalam sebagian besar sistem kendali proses, variabel yang dimanipulasi dan variabel kendali adalah pasangan bersama jadi pada manipulasi variabel digunakan untuk mengendalikan satu variabel kendali. Masing-masing pasang variabel terkendali dan variabel yang dimanipulasi(*manipulated value*) (MV), bersama-sama dengan algoritma kendali adalah merujuk sebagai loop kendali.

Pada beberapa kasus loop kendali meliputi banyak input dari proses dan banyak output ke proses. Ada beberapa algoritma yang bisa digunakan untuk mengendalikan proses, yang paling umum dan gampang adalah sakelar ON/OFF.

Sebagai contoh *thermostat* menyalakan pemanas ketika suhu turun di bawah *set point* dan kemudian mematikan *heater* ketika suhu *set point* tercapai.

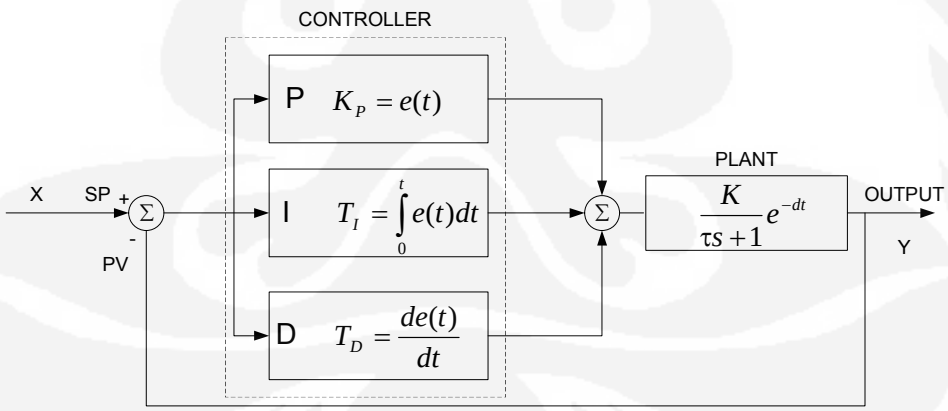
Untuk kendali yang lebih baik ada beberapa algoritma matematika yang menghitung perubahan berdasarkan output pada variable kendali. Sejauh ini metode kendali yang sangat umum digunakan adalah metoda PID (*Proportional Integral Derivative*).

2.4. METODE KENDALI

Sebuah sistem kendali yang di rancang, perlu di analisa terlebih dahulu untuk mendapatkan gambaran respon sistem. Gambaran tersebut meliputi :

1. Respon sistem terhadap berbagai macam input (*step function*, *ramp function*, dan *impulse function*, dll) termasuk jika ada gangguan dari luar.
2. Kestabilan sistem (metode: *root locus*, frekuensi respon, *state space*).
3. Respon sistem terhadap berbagai macam jenis pengendali (P, I, D dan/atau kombinasinya).

Namun demikian, ada kendala dalam menganalisa sistem yang akan dibuat, yaitu bagaimana mendapatkan fungsi alih dari sistem tersebut. Pada skripsi ini hanya di batasi pada sistem dengan *Unity Feedback System*, seperti ditunjukkan pada gambar 2.6.



Gambar 2.6. Blok diagram sistem loop tertutup dengan pengendali PID

Pengendali PID memiliki fungsi alih sebagai berikut :

$$H(s) = K_P + \frac{T_I}{s} + K_D s \qquad (15)$$

Pengendali PID sebenarnya terdiri dari 3 jenis cara pengaturan yang saling di kombinasikan, yaitu pengendali P (*Proportional*), Pengendali D (*Derivative*) dan pengendali I (*Integral*). Masing-masing memiliki parameter tertentu yang harus di set untuk dapat beroperasi dengan baik. Setiap jenis, memiliki kelebihan dan kekurangan masing-masing, hal ini dapat dilihat pada tabel 2.1.

Tabel 2.1. Respon pengendali PID terhadap perubahan konstanta

Closed-Loop Response	**Rise Time**	**Overshoot**	**Settling Time**	**SS Error**
Kp ↑	Menurun	Meningkat	Perubahan kecil	Menurun
Ti ↑	Menurun	Meningkat	Meningkat	Hilang
Td ↑	Perubahan kecil	Menurun	Menurun	Perubahan kecil

Untuk merancang sistem kendali PID, kebanyakan dilakukan dengan metoda coba-coba (*trial & error*). Hal ini disebabkan karena parameter Kp, Ti dan Td tidak *independent*. Untuk mendapatkan aksi kendali yang baik diperlukan langkah coba-coba dengan kombinasi antara P, I dan D sampai ditemukan nilai Kp, Ti dan Td seperti yang diiginkan. Tabel 2.1 hanya di pergunakan sebagai pedoman jika akan melakukan perubahan konstanta, tetapi dalam skripsi ini akan digunakan metode Ziegler Nichols untuk mendapatkan parameter-parameter pengendali.

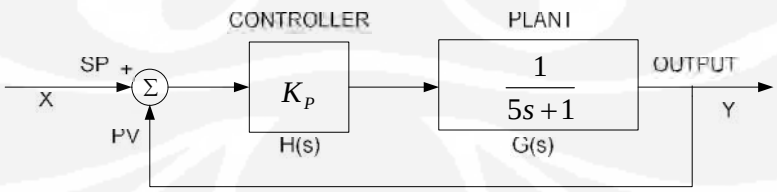
Desain sebuah sistem kendali, dimulai dengan membuat blok diagram sistem. Blok diagram (yang berisi fungsi alih) tersebut selanjutnya akan di analisa dengan menggunakan aksi pengendalian yang berbeda. Dengan perubahan sinyal input sehingga perancang dapat melihat respon sistem, jika mendapat sinyal input tertentu. Kombinasi antara sinyal input dan jenis aksi pengendalian ini akan menghasilkan respon yang berbeda-beda.

Tanggapan sistem dapat dilihat setelah sistem diberikan sinyal masukan yang berbeda. Kombinasi antara sinyal masukan dan aksi pengendalian ini akan menghasilkan tanggapan yang berbeda-beda. (Ogata, Katsuhiko, 1997) menjelaskan langkah-langkah yang harus dilakukan dalam perancangan sistem kendali sebagai berikut :

1. Memahami cara kerja sistem,
2. Mencari model sistem dinamik dalam persamaan differensial,
3. Mendapatkan fungsi alih sistem dengan transformasi Laplace,
4. Memberikan aksi pengendalian dengan menentukan konstanta Kp, Ti dan Td,
5. Menggabungkan fungsi alih yang sudah didapatkan dengan jenis aksi pengendalian,
6. Menguji sistem dengan sinyal masukan fungsi langkah, fungsi undak dan impuls ke dalam fungsi alih yang baru,
7. Melakukan transformasi Laplace balik untuk mendapatkan fungsi dalam domain waktu,
8. Menggambar tanggapan sistem dalam domain waktu.

2.4.1. Pengendali Proportional.

Karakteristik aksi pengendalian Proporsional adalah untuk mengurangi waktu naik, menambah *overshoot* (kenaikan cukup besar), dan mengurangi *steady state error*. Kenaikan *overshoot* ini sebanding dengan kenaikan nilai parameter Kp. Waktu turun juga menunjukkan kecenderungan yang membesar. Blok diagram sistem kendali loop tertutup adalah seperti pada gambar 2.7.



Gambar 2.7. Blok diagram sistem kendali loop tertutup dengan pengendali P

Fungsi alih sistem kendali loop tertutup dengan pengendali P adalah sebagai berikut:

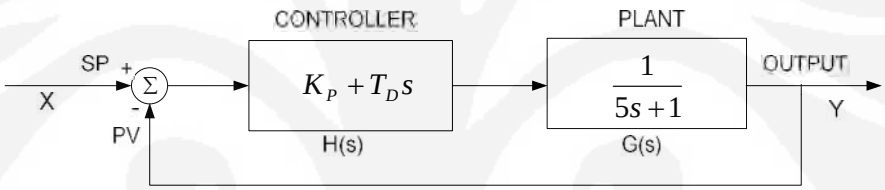
$$H(s)G(s) = \frac{K_P}{5s+1} \tag{16}$$

$$\frac{Y}{X} = \frac{H(s)G(s)}{1+H(s)G(s)} \tag{17}$$

$$\frac{Y}{X} = \frac{K_P}{5s+(1+Kp)} \tag{18}$$

2.4.2. Pengendali Proportional Derivative

Blok diagram sistem kendali loop tertutup dengan pengendali PD adalah seperti gambar 2.8.



Gambar 2.8.Blok diagram sistem kendali loop tertutup dengan pengendali PD

Fungsi alih sistem kendali loop tertutup dengan pengendali PD adalah sebagai berikut:

$$H(s)G(s) = \frac{K_P + T_D s}{5s+1} \tag{19}$$

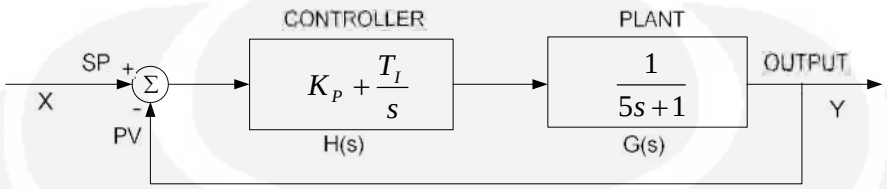
$$\frac{Y}{X} = \frac{H(s)G(s)}{1+H(s)G(s)} \tag{20}$$

$$\frac{Y}{X} = \frac{T_D s + K_P}{(5+T_D)s+(1+K_P)} \tag{21}$$

Penggunaan pengendali PD dapat mengurangi *over shoot* dan *settling time*, tetapi tidak memberikan dampak apapun terhadap *steady state error*.

2.4.3. Pengendali Proportional Integral

Blok diagram sistem kendali loop tertutup dengan pengendali PI seperti terlihat pada gambar 2.9.



Gambar 2.9. Blok diagram sistem kendali loop tertutup dengan pengendali PI

Fungsi alih sistem kendali loop tertutup dengan pengendali PI adalah sebagai berikut:

$$H(s)G(s) = \frac{K_P s + T_I}{5s^2 + s} \tag{22}$$

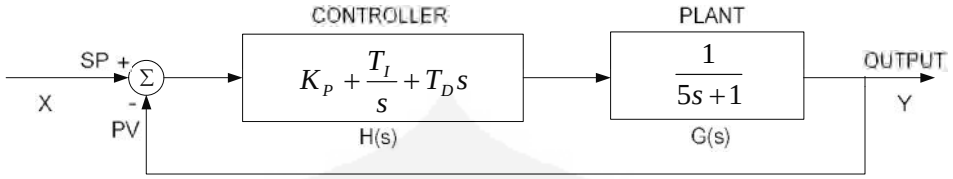
$$\frac{Y}{X} = \frac{H(s)G(s)}{1 + H(s)G(s)} \tag{23}$$

$$\frac{Y}{X} = \frac{K_P s + T_I}{5s^2 + (1 + K_P)s + T_I} \tag{24}$$

Pengendali Integral memiliki karakteristik mengurangi *rise time*, menambah *over shoot* dan *settling time* serta meghilangkan *steady state error* (karakteristik ini tidak dimiliki oleh jenis yang lain). Oleh karena itu, nilai Kp harus di kurangi untuk menghindari *overshoot* yang berlebihan. Nilai Ti diambil lebih besar dari Kp, karena di inginkan untuk meniadakan *steady state error*. Jika Kp > Ti, maka *steady state error*-nya tidak dapat dihilangkan.

2.4.4. Pengendali Proportional Integral Derivative

Blok diagram sistem kendali loop tertutup dengan pengendali PID adalah seperti gambar 2.10.



Gambar 2.10. Blok diagram sistem kendali loop tertutup dengan pengendali PID

Fungsi alih sistem kendali loop tertutup dengan pengendali PID adalah sebagai berikut:

$$H(s)G(s) = \frac{T_D s^2 + K_P s + T_I}{5s^2 + s} \quad (25)$$

$$\frac{Y}{X} = \frac{H(s)G(s)}{1 + H(s)G(s)} \quad (26)$$

$$\frac{Y}{X} = \frac{T_D s^2 + K_p s + T_I}{(5 + T_D)s^2 + (1 + K_P)s + T_I} \quad (27)$$

Dalam penerapan sistem kendali, sebenarnya tidak perlu menggunakan pengendali PID. Jika sistem sudah memberikan respon yang cukup baik hanya dengan menggunakan pengendali PI, maka tidak perlu menambahkan pengendali D ke dalamnya. Sehingga sistem menjadi lebih sederhana (kombinasi yang makin banyak membuat sistem menjadi makin komplek).

2.6. PIPING AND INSTRUMENTATION DIAGRAM (P&ID)

Piping and Instrumentation Diagram (P&ID) dapat diartikan sebagai sebuah alat bantu untuk menerangkan konsep desain dari suatu proses dan kebutuhan pabrik atau unit produksi yang perlu atau akan dibangun. Ada beberapa tahapan membuat suatu P&ID yang harus dilalui dan dikenal yang dimulai dengan mengenal tahapan proyek yang akan dilakukan dan juga mencakup semua simbol instrumen kendali ataupun *piping* yang perlu dinotasikan dalam sebuah P&ID.

P&ID sendiri adalah grafik detail yang merepresentasikan proses meliputi perangkat keras dan perangkat lunak (contoh, pemipaan, peralatan, instrumentasi)

yang perlu untuk didesain, membuat dan mengoperasikan fasilitas. Persamaan yang umum untuk P&ID meliputi EFD (*Engineering Flow Diagram*), UFD (*Utility Flow Diagram*) dan MFD (*Mechanical Flow Diagram*).

2.7. PROGRAMABLE LOGIC CONTROLLER (PLC)

PLC pertama kali diperkenalkan pada tahun 1960-an. Alasan utama perancangan PLC adalah untuk menghilangkan biaya perawatan dan menggantikan sistem kendali berbasis relai. Bedford Associates (Bedford, MA) mengajukan usulan yang diberi nama MODICON (*Modular Digital Controller*) untuk perusahaan-perusahaan mobil di Amerika. Sedangkan perusahaan lain mengajukan sistem berbasis komputer (PDP-8). MODICON 084 merupakan PLC pertama di dunia yang digunakan pada produk komersial.

Saat kebutuhan produksi berubah maka demikian juga dengan sistem kendalinya. Hal ini menjadi sangat mahal jika perubahannya terlalu sering. Karena relai merupakan alat mekanik, maka memiliki masa pengunaan yang yang terbatas yang akhirnya membutuhkan jadwal perawatan yang ketat. Pelacakan kerusakan atau kesalahan menjadi cukup membosankan jika banyak relai yang digunakan. Bayangkan saja sebuah panel kendali yang dilengkapi dengan monitor untuk ratusan hingga ribuan relai yang terdapat dalam sistem kendali tersebut, bagaimana kompleknya melakukan pengkabelan pada relai-relai tersebut.

Dengan adanya pengendali baru ini, harus mempermudah para teknisi perawatan dan teknisi lapangan melakukan pemrograman. Umur alat harus menjadi lebih panjang dan pemrograman proses harus dapat dimodifikasi atau diubah dengan lebih mudah. Pada pertengahan tahun 1970-an, teknologi PLC yang dominan adalah sekuenser mesin-kondisi dan CPU berbasis *bit-slice*. Prosesor AMD 2901 dan 2903 cukup populer digunakan dalam MODICON dan PLC A-B. Mikroprosesor konvensional kekurangan daya dalam menyelesaikan secara cepat logic PLC untuk semua PLC, kecuali PLC kecil. Setelah mikroprosesor konvensional mengalami perbaikan dan pengembangan, PLC yang besar-besar mulai menggunakannya. Bahkan sampai saat ini masih ada yang masih berbasis AMD 2903.

Kemampuan komunikasi pada PLC mulai muncul pada awal tahun 1973. Sistem yang pertama adalah *Modbus*-nya MODICON. Dengan demikian PLC bisa melakukan komunikasi dengan PLC yang lain dan dan bisa ditempatkan lebih jauh dari lokasi mesin sesungguhnya yang dikendalikan. Sekarang kemampuan komunikasi ini dapat digunakan untuk mengirimkan dan menerima berbagai macam tegangan untuk membolehkan dunia analog terlibat.

Pada tahun 1980-an dilakukan usaha untuk melakukan standarisasi komunikasi dengan protokol otomasi pabrik milik General Motor (*Genenal Motor's Manufacturing Automation Protocol* (MAP)). Juga merupakan waktu untuk memperkecil ukuran PLC dan membuat perangkat lunak pemrograman melalui pemrograman simbolik dengan komputer dari pada menggunakan terminal pemrograman. Sekarang PLC terkecil seukuran dengan sebuah relai.

Tahun 1990-an dilakukan reduksi protokol baru dengan modernisasi lapisan fisik dari protokol-protokol populer yang bertahan pada tahun 1980-an. Standar terakhir (IEC 1131-3, bisa diakses di http://www.plcopen.org/default.htm) berusaha untuk menggabungkan bahasa pemrograman PLC dibawah satu standar internasional. Sekarang bisa dijumpai PLC-PLC yang dapat diprogram dengan standar fungsi blok, daftar instruksi, C dan teks terstruktur pada saat bersamaan.

Sistem kendali proses terdiri atas sekumpulan piranti-piranti dan peralatan elektronik yang mampu menangani kestabilan, akurasi, dan mengeliminasi transisi status yang berbahaya dalam proses produksi. Masing-masing komponen dalam sistem kendali proses tersebut memegang peranan pentingnya masing-masing.

PLC (*Programable Logic Controller*) adalah sebuah alat yang digunakan untuk menggantikan rangkaian deretan relai yang di jumpai pada sistem kendali proses konvensional. PLC bekerja dengan cara mengamati masukkan kemudian melakukan proses dan melakukan tindakan sesuai dengan yang diprogramkan yang berupa menghidupkan atau mematikan keluaran.

Ada beberapa kelemahan dari sistem kendali konvensional tanpa penggunaan PLC sebagai pengendali antara lain:

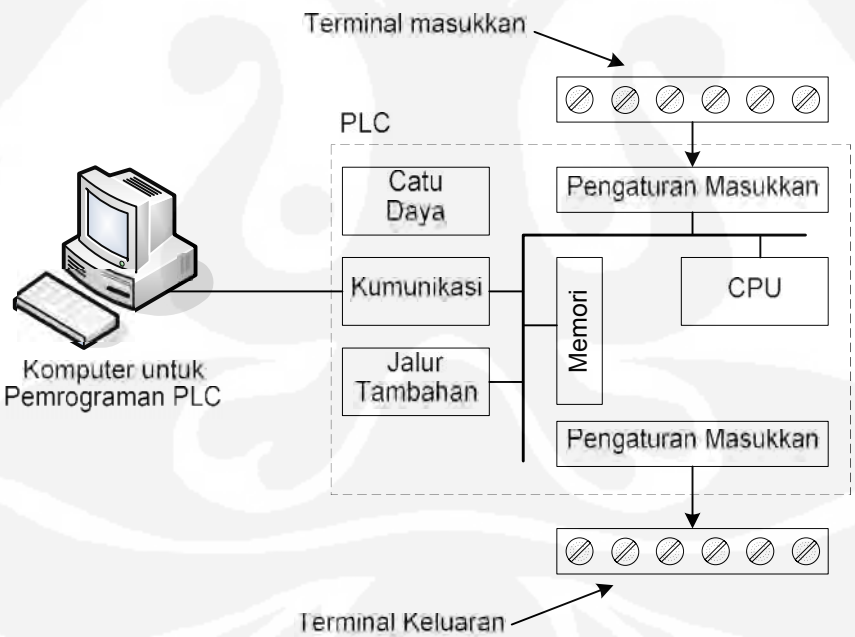
1. Perlu kerja keras saat dilakukan pengkabelan.
2. Kesulitan saat dilakukan penggantian dan/atau perubahan.
3. Kesulitan saat dilakukan pelacakan kesalahan.

4. Saat terjadi masalah, waktu tunggu tidak menentu dan biasanya lama.

Sedangkan penggunakan PLC sebagai pengendali memiliki beberapa kelebihan dibanding dengan sistem kendali konvesional, antara lain:

1. Jumlah kabel yang dibutuhkan bisa berkurang.
2. PLC mengkonsumsi daya lebih rendah dibanding dengan sistem kendali konvensional.
3. Fungsi diagnostik pada sebuah pengendali PLC membolehkan pendeteksian kesalahan yang mudah dan cepat.
4. Perubahan pada urutan operasional atau proses aplikasi dapat dilakukan dengan mudah, yaitu hanya dengan melakukan perubahan atau penggantian program.

PLC sesungguhnya merupakan sistem mikrokontroller khusus untuk industri, artinya seperangkat perangkat lunak dan keras yang diadaptasi untuk keperluan aplikasi dalam dunia industri. Elemen-elemen dasar dari PLC ditunjukkan pada gambar 2.11.



Gambar 2.11. Elemen-elemen dasar PLC

2.7.1. Central Processing Unit (CPU).

Unit pusat pengolahan atau CPU merupakan otak dari sebuah pengendali PLC. CPU itu sendiri biasanya merupakan sebuah mikrokontroller. Pada awalnya merupakan mikrokontroller 8-bit seperti 8051, namun saat ini bisa merupakan mikrokontroller 16 atau 32 bit. Biasanya untuk produk-produk PLC buatan Jepang, mikrokontrollernya adalah Hitachi dan Fujitsu, sedangkan untuk produk eropa banyak menggunakan Siemens dan Motorola untuk produk-produk Amerika. CPU ini juga menangani komunikasi dengan piranti ekternal, interkonektifitas antar bagian-bagian internal PLC, eksekusi program, manajemen memori, mengawasi atau mengamati masukan dan memberikan sinyal keluaran (sesuai dengan proses atau program yang dijalankan). Pengendali PLC memiliki suatu rutin kompleks yang digunakan untuk memeriksa memori agar dapat dipastikan memori PLC tidak rusak, hal ini dilakukan karena alasan keamanan. Hal ini bisa dijumpai dengan adanya indikator lampu pada badan PLC sebagai indikator terjadinya kesalahan atau kerusakan.

2.7.2. Memori

Memori sistem (saat ini banyak yang mengimplementasikan penggunaan teknologi *Flash*) digunakan oleh PLC untuk sistem kendali proses. Selain berfungsi untuk menyimpan "*system operasi*", juga digunakan untuk menyimpan program yang harus dijalankan, dalam bentuk biner, hasil terjemahan diagram tangga yang dibuat oleh pengguna atau pemrogram. Isi dari memori *flash* tersebut dapat berubah (bahkan dapat juga dikosongkan atau dihapus) jika memang dikehendaki seperti itu. Tetapi yang jelas, dengan penggunaan teknologi *flash*, proses penghapusan dan pengisian kembali memori dapat dilakukan dengan mudah dan cepat. Pemrograman PLC biasanya dilakukan melalui kanal serial komputer yang bersangkutan.

Memori pengguna dibagi menjadi beberapa blok yang memiliki fungsi khusus. Berupa bagian memori digunakan untuk menyimpan status masukan dan keluaran. Status yang sesungguhnya dari masukan maupun keluaran disimpan sebagai logika atau bilangan '0' dan '1' (dalam lokasi bit memori tertentu). Masing-masing masukan atau keluaran berkaitan dengan sebuah bit dalam

memori. Sedangkan bagian lain dari memori digunakan untuk menyimpan isi variabel-variabel yang digunakan dalam program yang dituliskan. Misal, nilai pewaktu atau nilai pencacah bisa disimpan dalam bagian memori ini.

2.7.3. Pemrograman PLC

PLC dapat diprogram melalui komputer, tetapi juga bisa diprogram melalui pemrograman manual, yang bisa disebut dengan konsol *(console)*. Untuk keperluan ini dibutuhkan perangkat lunak, yang biasanya juga bergantung pada produk PLC-nya.

Saat ini fasilitas transmisi PLC dengan komputer sangat penting sekali artinya dalam pemrograman-ulang PLC dalam dunia industri. Sekali sistem diperbaiki, program yang benar dan sesuai harus disimpan kedalam PLC lagi. Selain itu perlu dilakukan pemeriksaan program PLC, apakah selama disimpan tidak terjadi perubahan atau sebaliknya, apakah program sudah berjalan dengan benar atau tidak. Hal ini membantu untuk menghindari situasi berbahaya dalam ruang produksi (pabrik), dalam hal ini beberapa pabrik PLC telah membuat fasilitas dalam PLC-nya berupa dukungan terhadap jaringan komunikasi, yang mampu melakukan pemeriksaan program sekaligus pengawasan secara rutin apakah PLC bekerja dengan baik dan benar atau tidak.

Hampir semua produk perangkat lunak untuk memprogram PLC memberikan kebebasan berbagai macam pilihan seperti: memaksa suatu saklar (masukan atau keluaran) bernilai ON atau OFF, melakukan pengawasan program *(monitoring)* secara *real-time* termasuk pembuatan dokumentasi diagram tangga yang bersangkutan. Dokumentasi diagram tangga ini diperlukan untuk memahami program sekaligus dapat digunakan untuk pelacakan kesalahan. Pemrograman dapat memberikan nama pada piranti masukan maupun keluaran, komentar-komentar pada blok diagram dan lain sebagainya. Dengan pemberian dokumentasi maupun komentar pada program, maka akan mudah nantinya dilakukan pembenahan (perbaikan maupun modifikasi) program dan pemahaman terhadap kerja program diagram tangga tersebut.

2.7.4. Catu Daya PLC

Catu daya listrik digunakan untuk memberikan pasokan catu daya ke seluruh bagian PLC (termasuk CPU, memori dan lain-lain). Kebanyakan PLC bekerja dengan catu daya 24 VDC atau 220 VAC. Beberapa PLC catu dayanya terpisah (sebagai modul tersendiri). Yang demikian biasanya merupakan PLC besar, sedangkan yang medium atau kecil, catu dayanya sudah menyatu. Pengguna harus menentukan berapa besar arus yang diambil dari modul keluaran/ masukan untuk memastikan catu daya yang bersangkutan menyediakan sejumlah arus yang memang dibutuhkan

Catu daya listrik ini biasanya tidak digunakan untuk memberikan catu daya langsung ke masukan maupun keluaran, artinya masukan dan keluaran murni merupakan saklar (baik relay maupun *optoisolator*). Pengguna harus menyediakan sendiri satu catu daya terpisah untuk masukan dan keluaran PLC. Dengan cara demikian, maka lingkungan industri dimana PLC digunakan tidak akan merusak PLC-nya itu sendiri karena memiliki catu daya terpisah antara PLC dengan jalur-jalur masukan dan keluaran.

2.7.5. Masukan-masukan PLC

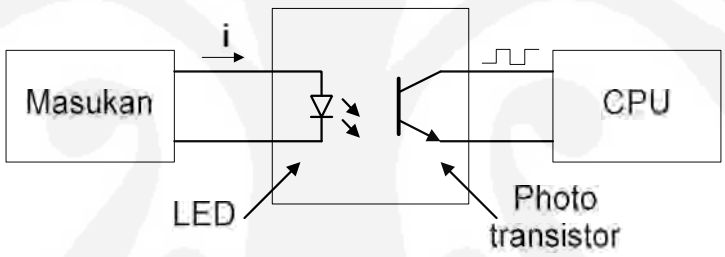
Kecerdasan sebuah sistem terotomasi sangat bergantung pada kemampuan sebuah PLC untuk membaca sinyal dari berbagai macam jenis sensor dan piranti-piranti masukan lainnya. Untuk mendeteksi proses atau kondisi atau status suatu keadaan atau proses yang sedang terjadi, misalnya, berapa cacah barang yang sudah diproduksi, ketinggian permukaan air, tekanan udara dan lain sebagainya, maka dibutuhkan sensor-sensor yang tepat untuk masing-masing kondisi atau keadaan yang akan dideteksi tersebut. Dengan kata lain, sinyal-sinyal masukan tersebut dapat berupa logik (ON atau OFF) maupun analog. PLC kecil biasanya hanya memiliki jalur masukan digital saja, sedangkan yang besar mampu menerima masukan analog melalui unit khusus yang terpadu dengan PLC-nya. Salah satu sinyal analog yang sering dijumpai adalah sinyal arus 4 hingga 20 mA (atau mV) yang diperoleh dari berbagai macam sensor.

Lebih canggih lagi, peralatan lain dapat dijadikan masukan untuk PLC, seperti citra dari kamera, robot (misalnya, robot bisa mengirimkan sinyal ke PLC

sebagai suatu informasi bahwa robot tersebut telah selesai memindahkan suatu objek dan lain sebagainya) dan lain-lain.

2.7.6. Antarmuka Masukan

Antarmuka masukan berada diantara jalur masukan yang sesungguhnya dengan unit CPU. Tujuannya adalah melindungi CPU dari sinyal-sinyal yang tidak dikehendaki yang bisa merusak CPU itu sendiri. Modul antarmuka masukan ini berfungsi untuk mengkonversi atau mengubah sinyal-sinyal masukan dari luar ke sinyal-sinyal yang sesuai dengan tegangan kerja CPU yang bersangkutan (misalnya, masukan dari sensor dengan tegangan kerja 24VDC harus dikonversikan menjadi tegangan 5 VDC agar sesuai dengan tegangan kerja CPU). Hal ini dengan mudah bisa dilakukan menggunakan rangkaian *opto-isolator* seperti yang ditunjukkan pada gambar 2.12.



Gambar 2.12. Rangkaian antarmuka masukan PLC

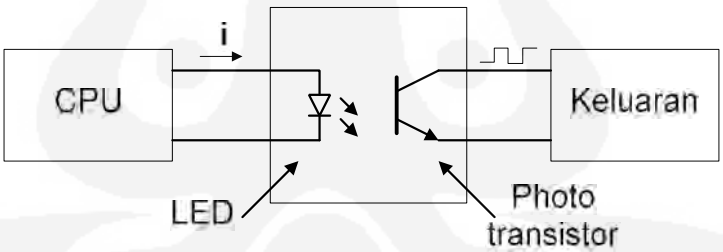
Penggunaan *opto-isolator* artinya tidak ada hubungan kabel sama sekali antara dunia luar dengan unit CPU. Secara '*optik*' dipisahkan (perhatikan gambar 2.12), atau dengan kata lain sinyal ditransmisikan melalui cahaya. Cara kerjanya sederhana, piranti ekternal akan memberikan sinyal untuk menghidupkan LED (dalam optoisolator), akibatnya *photo isolator* akan menerima cahaya dan akan menghantarkan arus (ON), CPU akan melihatnya sebagai logika nol (catu antara kolektor dan emitor drop dibawah 1 volt). Begitu juga sebaliknya, saat sinyal masukkan tidak ada lagi, maka LED akan mati dan photo transistor akan berhenti menghantar (OFF), CPU akan melihatnya sebagai logika satu.

2.7.7. Keluaran-keluaran PLC

Sistem terotomasi tidak akan lengkap jika tidak ada fasilitas keluaran atau fasilitas untuk menghubungkan dengan alat-alat eksternal (yang dikendalikan). Beberapa alat atau piranti yang banyak digunakan adalah motor, *solenoid*, relai, lampu indikator, katup kendali dan lain sebagainya. Keluaran ini dapat berupa analog atau digital. Keluaran digital bersikap seperti sakelar, menghubung dan memutuskan jalur. Keluaran analog digunakan untuk menghasilkan sinyal analog (misalnya, perubahan tegangan untuk mengendalikan motor secara regulasi linier sehingga diperoleh kecepatan putar tertentu).

2.7.8. Antarmuka Keluaran

Seperti pada antarmuka masukan, keluaran juga membutuhkan antarmuka yang sama yang digunakan untuk memberikan perlindungan antara CPU dengan peralatan eksternal, seperti yang ditunjukkan pada gambar 2.13. Cara kerjanya juga sama, yang menyalakan dan mematikan LED di dalam *optoisolator* sekarang adalah CPU, sedangkan yang membaca status *photo transistor*, apakah menghantarkan arus apa tidak adalah peralatan atau piranti eksternal.



Gambar 2.13. Rangkaian antarmuka keluaran PLC

2.7.9. Jalur Tambahan

Setiap PLC biasanya memiliki jalur masukan dan keluaran yang terbatas. Jika diinginkan jumlah ini dapat ditambah menggunakan sebuah modul keluaran atau masukan tambahan (I/O expansion atau I/O extension module)

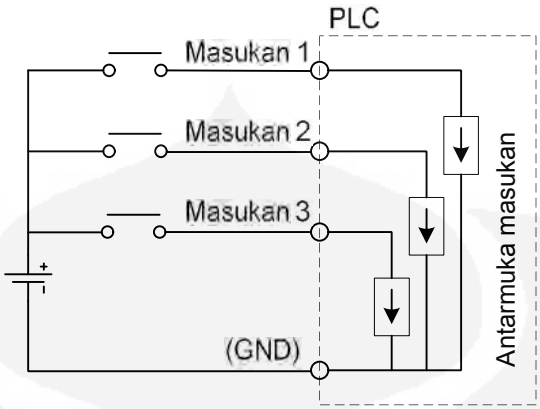
2.7.10. Menghubungkan Piranti Masukkan dan Keluaran

Seperti yang telah dibahas sebelumnya, PLC yang berdiri sendiri tidak ada artinya, agar berfungsi sebagaimana mestinya, PLC haruslah dilengkapi dengan piranti-piranti masukan maupun keluaran. Untuk masukkan, diperlukan sensor untuk memperoleh informasi yang dibutuhkan. Kemudian apa yang dikendalikan, inilah fungsi dari keluaran, dihubungkan dengan berbagai macam piranti yang akan dikendalikan seperti motor, *solenoid* dan sebagainya.

2.7.10.1. Konsep Dasar

Konsep dasar ini berkaitan dengan apa yang bisa dihubungkan dan bagaimana cara menghubungkannya ke masukan atau keluaran PLC. Ada dua istilah yang sudah lazim dikalangan elektronika maupun pengguna PLC, yaitu istilah "*sinking*" dan "*sourcing*". Istilah *singking* berkaitan dengan penarikan sejumlah arus dari piranti luar (eksternal), istilah ini berkaitan dengan tanda "-" (terminal negatif) atau GND (ground). Sedangkan istilah *sourcing*, yang berkaitan dengan terminal atau tanda "+" atau Vcc, berkaitan dengan pemberian sejumlah arus ke piranti luar.

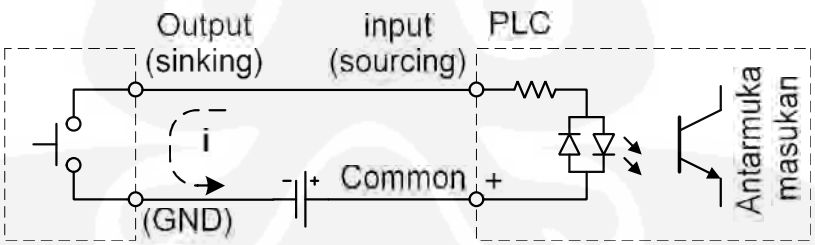
Masukan dan keluaran, baik yang bersifat *sinking* maupun *sourcing* hanya bisa menghantarkan arus listrik searah, artinya menggunakan catu daya DC. Dengan demikian, setiap jalur masukan atau keluaran memiliki terminal (+) dan (-), jika terdapat 5 masukan, maka akan terdapat 10(5x2 terminal) terminal masukan, yang masing-masing bertanda (+) dan (-). Namun hal ini kemudian dihindari dengan cara menyatukan terminal (-)nya, yang kemudian untuk beberapa masukan atau keluaran dijadikan satu dan disebut dengan jalur *common* (dalam PLC dengan tanda COMM). Pada gambar 2.14 ditunjukkan contoh 3 masukan dengan satu jalur tunggal terminal COMM dan masing-masing dihubungkan dengan sebuah sakelar.



Gambar 2.14. Ilustrasi terminal COMM

2.7.10.2. Jalur-jalur Masukan

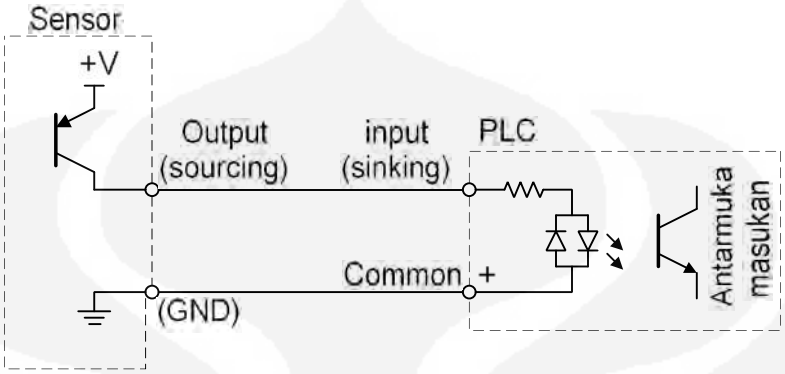
Yang perlu diperhatikan dalam menghubungkan piranti luar dengan jalur masukan, yang biasanya berupa sensor, adalah bahwa keluaran dari sensor bisa berbeda tergantung dari sensornya sendiri dan aplikasinya. Yang penting bagaimana caranya dibuat suatu rangkaian sensor yang dapat memberikan sinyal ke PLC sesuai dengan spesifikasi masukan PLC yang digunakan. Pada gambar 2.15 ditunjukkan suatu contoh cara menghubungkan sebuah sensor dengan tipe keluaran *sinking* dengan masukan PLC yang bersifat *sourcing*.



Gambar 2.15. Menghubungkan sensor keluaran *sinking* dengan masukan *sourcing*

Pada gambar 2.15, jenis sensor yang digunakan sebagaimana disebutkan sebelumnya merupakan jenis yang menarik arus (*sinking*), dengan demikian masukan atau hubungan yang cocok disisi lainnya (PLC) adalah memberikan arus (*sourcing*). Perhatikan penempatan tegangan DC-nya, terutama polaritas terminalnya. Dalam hal ini *COMMON* bersifat positif untuk tipe hubungan

semacam ini. Sedangkan pada gambar 2.16 ditunjukkan tipe hubungan kebalikan dari tipe yang sebelumnya.

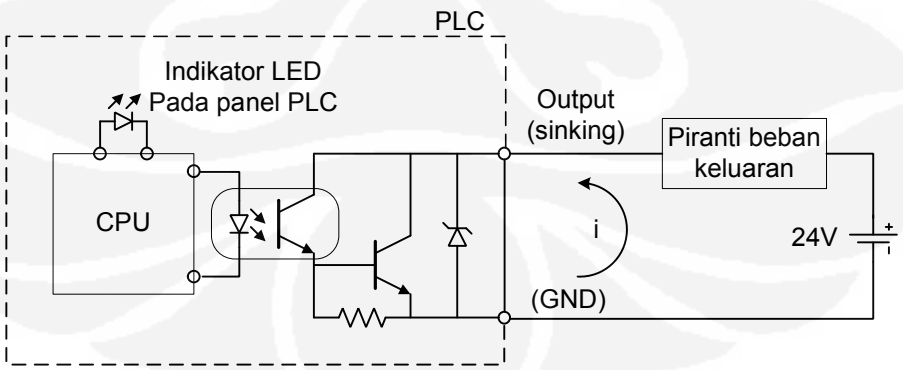


Gambar 2.16. Menghubungkan sensor keluaran *sourcing* dengan masukan *sinking*

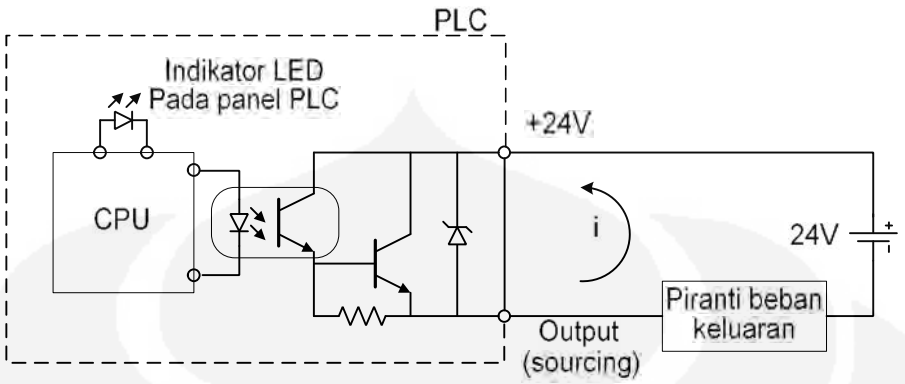
Pada gambar 2.16 tersebut terlihat bahwa sekarang sensor memiliki sumber arus tersendiri sehingga tipenya merupakan *sourcing*, pasangan terminalnya disisi yang lain (PLC) merupakan tipe *sinking*. Untuk tipe hubungan semacam ini, COMMON bersifat negatif atau GND. Secara garis besar dapat dikatakan bahwa harus dilakukan hubungan *sinking-sourcing* atau *sourcing-sinking* bukan hubungan *sinking-sinking* maupun *sourcing-sourcing*.

2.7.10.3. Jalur-jalur Keluaran

Karena dari PLC biasa dapat berupa transistor PNP, NPN maupun relai. Pada gambar 2.17 dan 2.18, masing-masing ditunjukkan bagaimana cara PLC mengatur piranti eksternal secara nyata.



Gambar 2.17. Menghubungkan beban keluaran dengan keluaran PLC tipe *sinking*



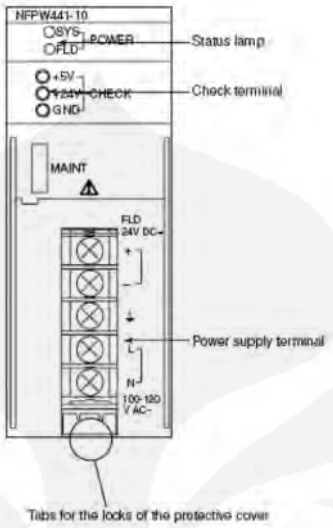
Gambar 2.18. Menghubungkan beban keluaran dengan keluaran PLC tipe *soucing*

Pada gambar 2.18 ditunjukkan bagaimana PLC menangani beban keluaran, jika PLCnya sendiri keluarannya tipe *sinking*. Beban diletakkan antara terminal masukkan *sinking* dengan terminal positif catu daya, yang digunakan untuk mengerakkan beban bukan untuk PLC-nya itu sendiri. Sedangkan pada gambar 2.18 adalah sebaliknya, keluaran PLC adalah *sourcing*, sehingga konfigurasinya, beban keluaran diletakkan antara keluaran *sourcing* dengan terminal negatif.

2.8.11. Modul-modul PLC

Pada PLC yang besar biasanya bagian-bagiannya terpisah atau yang sering dikenal dengan PLC modular. Modul PLC *stardom* yang akan digunakan dalam skripsi ini adalah sebagai berikut:

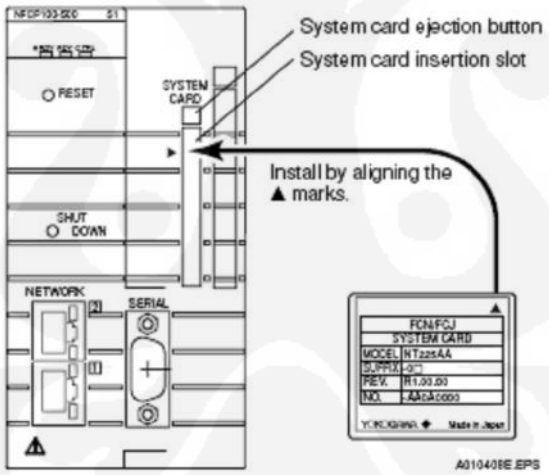
1. Modul Power Supply
   Model : NFPW 441-10



Gambar 2.19. Modul power supply

2. Modul CPU.

   Model : NFCP100-S



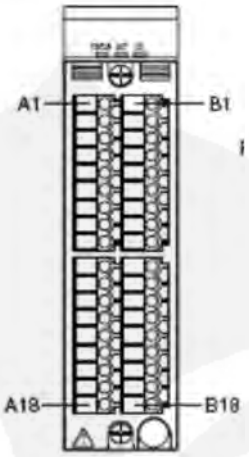
Gambar 2.20. Modul CPU

3. Modul I/O Digital

   Digital Modul terdiri dari dua tipe yaitu modul input dan modul output.

   a. Modul Digital Input

      Model : NFDV151



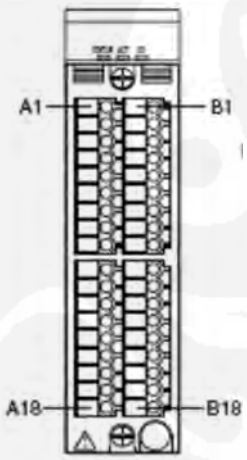
Gambar 2.21. Modul digital input

Data yang menggunakan input digital adalah:

- Indikator buka/tutup katup.

b. Modul Digital Output

Model : NFDV551



Gambar 2.22. Modul digital output

Data yang menggunakan output digital adalah:

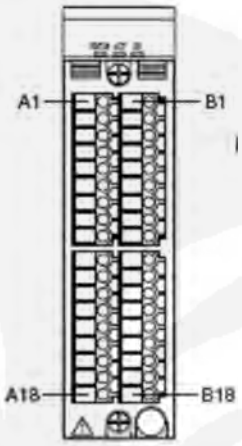
- Perintah buka/tutup katup
- Indikator tekanan yang terlalu tinggi.

4. Modul I/OAnalog

Modul analog terdiri dari dua tipe yaitu modul input dan modul output.

a. Modul analog Input

   Model : NFAI141



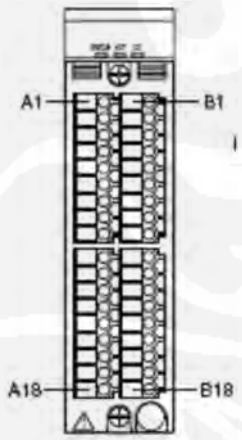
Gambar 2.23. Modul analog input

Data yang menggunakan Input Analog :

- Kecepatan aliran
- Perbedaan tekanan.

b. Modul analog output

   Model : NFAI543



Gambar 2.24. Modul analog ouput

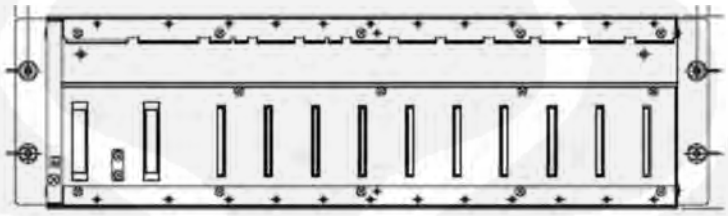
Data yang menggunakan output analog adalah:

- Katup kendali.

5. Base Modul.

   Base Modul berfungsi sebagai tempat untuk menempatkan modul yang akan digunakan.

   Model : NFBU 200-S0



Gambar 2.25. Base modul

Pada gambar 2.25 terlihat base modul dengan 2 slot untuk power supply dan 10 slot untuk I/O modul dan Modul komunikasi.

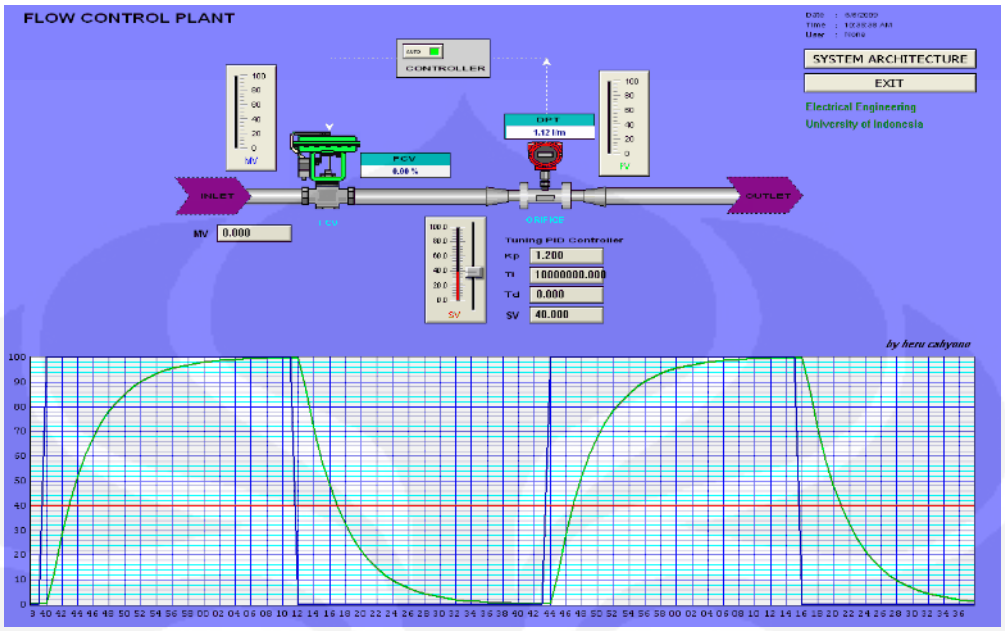
2.9. HMI (Human Machine Interface)

Human Machine Interface adalah perangkat lunak yang digunakan untuk menghubungkan antara mesin dengan manusia. Pada skripsi ini HMI yang digunakan adalah *Intouch Wonderware*. *Wonderware* menggunakan protocol TCP/IP. *Intouch* terdiri dari tiga program utama yakni *Intouch application Manager, WindowMaker* dan *WindowViewer*.

Intouch Application Manager mengorganisasi aplikasi yang dibuat, dan juga untuk konfigurasi *WindowViewer* sebagai *service*, untuk mengkonfigurasi *Network Application Development (NAD)* untuk arsitektur berdasarkan *client* atau *server*, untuk mengkonfigurasi *Dynamic Resolution Conversion (DRC)*.

WindowMaker digunakan untuk membangun sebuah perangkat lunak untuk aplikasi tertentu. Dimana grafik yang berorientasi pada obyek digunakan untuk membuat animasi. Aplikasi yang dibuat dapat disambungkan dengan sistem I/O industri.

WindowViewer adalah aplikasi yang digunakan untuk menampilkan HMI yang telah dibuat oleh *WindowMaker* seperti pada gambar 2.26.



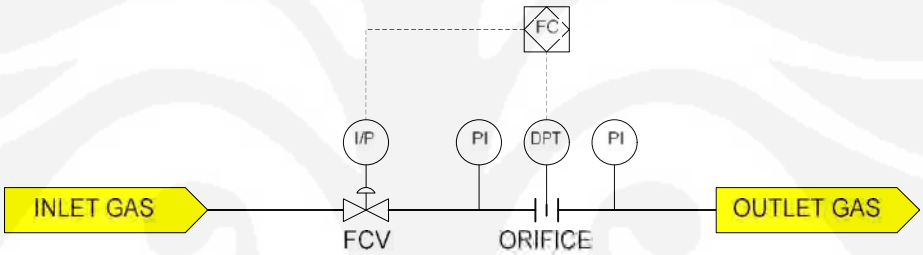
Gambar 2.26. Human Machine Interface (HMI)

BAB 3
PERANCANGAN

1.9. PERANCANGAN PLANT

Pada skripsi ini akan dibuat sebuah plant untuk mengendalikan aliran baik aliran gas atau cairan. Untuk flow meter menggunakan *orifice* meter yakni memanfaatkan perbedaan tekanan pada sisi *downstream* dan *upstream* untuk mendapatkan kecepatan aliran. Dan untuk aktuator menggunakan katup kendali (*control valve*). Pengendali menggunakan PLC seri Stardom Yokogawa.

Untuk membuat desain sistem kendali untuk plant pengendali aliran ini kita mulai dari desain *Piping dan Instrumentation diagram* (P&ID), yang di tunjukkan pada gambar 3.1.



Gambar 3.1. P&ID pengendali aliran

Dimana :

FC : *Flow Controller*

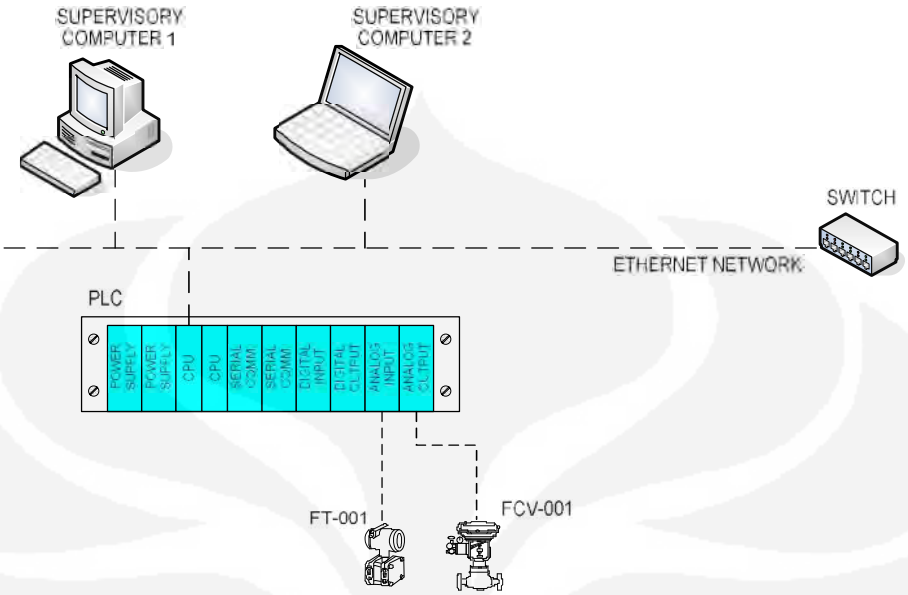
FCV : *Flow Control Valve*

I/P : *Current to Pneumatic Converter*

PI : *Pressure Indicator*

PT : *Pressure Transmitter*

Setelah itu diperlukan desain sistem SCADA untuk mendapatkan data dari Plant. Sistem arsitektur bisa dilihat pada gambar 3.2 di bawah ini.



Gambar 3.2. Arsitektur SCADA pengendali aliran

1.10. PERANCANGAN PERANGKAT LUNAK

Perangkat lunak yang digunakan pada skripsi ini terdiri dari dua perangkat yakni perangkat lunak untuk PLC dan perangkat lunak untuk HMI SCADA

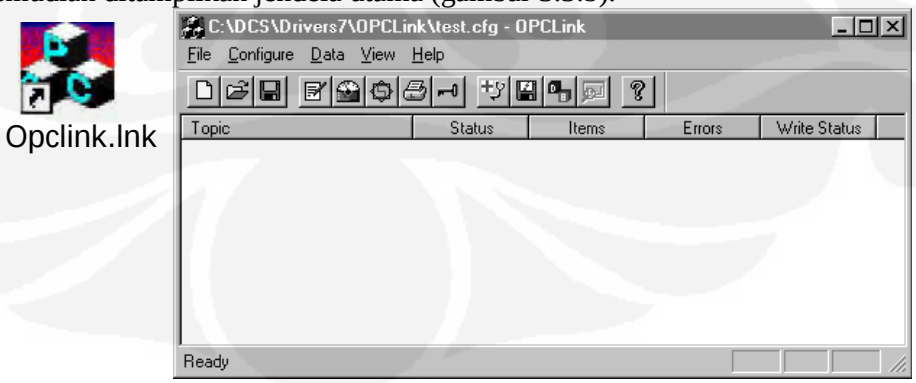
3.2.1. Desain perangkat lunak HMI.

HMI sistem SCADA menggunakan *Intouch* produk *Wonderware*. Langkah-langkah pembuatan HMI adalah sebagai berikut:

1. Mengkonfigurasi OPCLink.

1.1 Jendela utama OPCLink.

(1) Klik dua kali pada icon *OPCLink*(gambar 3.3.a) untuk memulai, kemudian ditampilkan jendela utama (gambar 3.3.b).



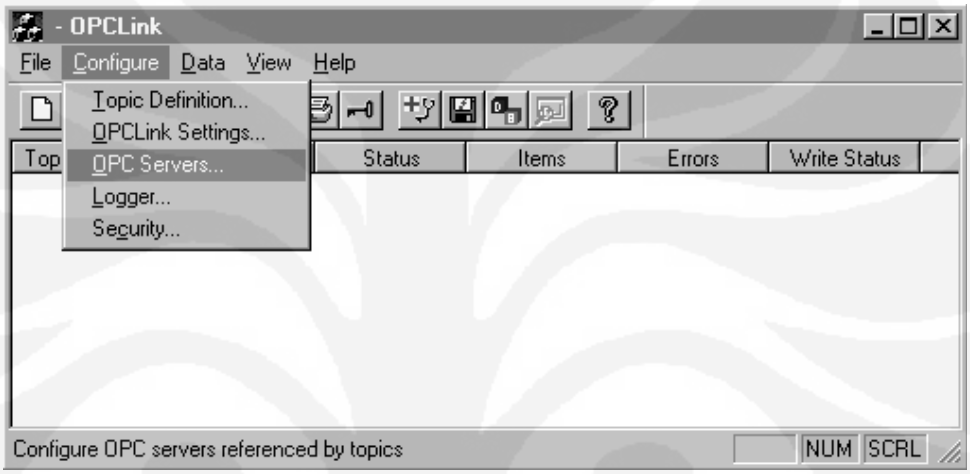
(a) (b)

Gambar 3.3. (a) *Icon* OPC (b) Jendela utama OPC

1.2 Mengkonfigurasi OPCLink.

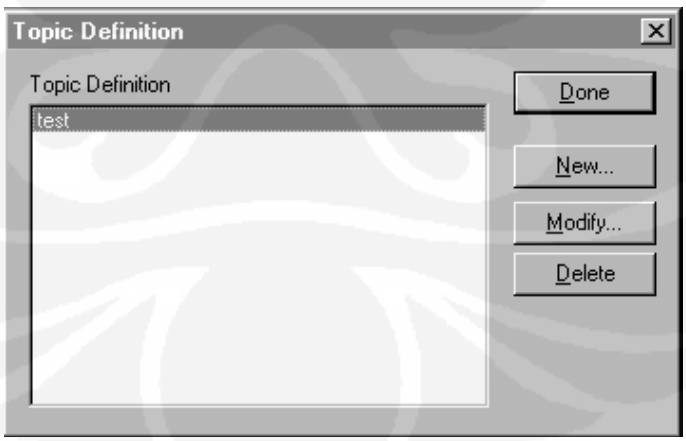
Sekali *OPCLink* terpasang pada komputer, harus dilakukan beberapa konfigurasi. Konfigurasi *OPCLink* secara otomatis menyimpan data dalam file konfigurasi. Jika tidak ada file konfigurasi yang dipilih pengguna harus memberi nama file.

(1) Untuk mengakses pilihan yang digunakan untuk bermacam-macam konfigurasi buka menu *Configure* (gambar 3.4).



Gambar 3.4. Mengkonfigurasi OPC

(2) Pilih *Topic Definition* dari menu *Configure* untuk membuat modifikasi baru atau menghapus *Topic Definition*. Kemudian akan ditampilkan kotak dialok *Topic definition*(gambar 3.5).

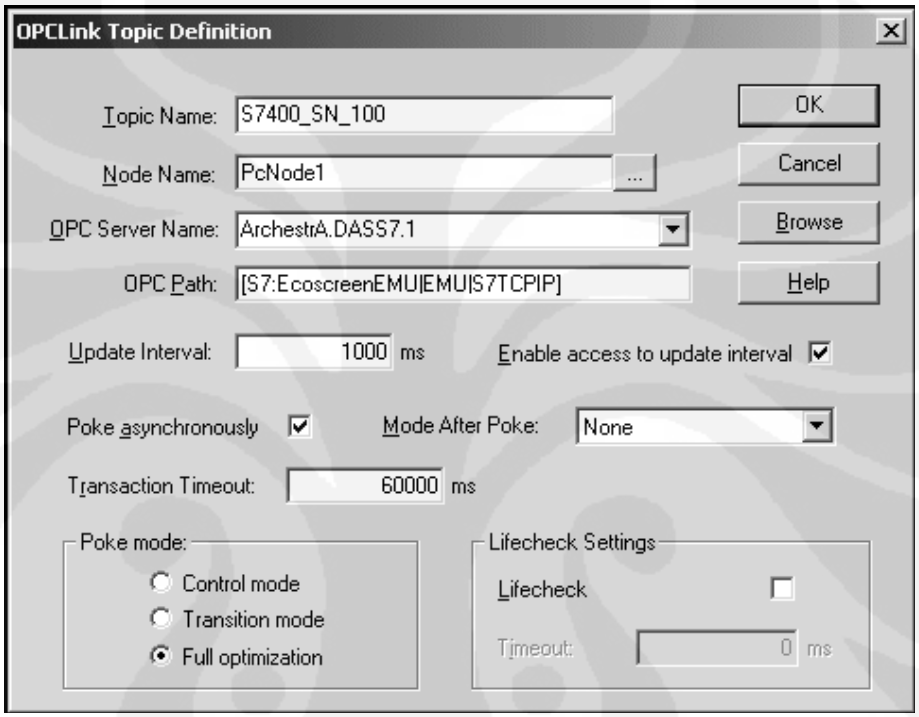


Gambar 3.5. Kotak dialog *Topic Definition*.

(3) Klik tombol Done untuk menutup kotak dialog ketika telah selesai membuat file baru atau menghapus.

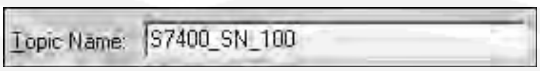
(4) Untuk memodifikasi atau melihat *topic definition* yang telah ada, pilih nama dalam daftar dan klik tombol *modify*.

(5) Ketika tombol *new* dipilih akan ditampilkan kotak dialog *OpenLink Topic Definition* (gambar 3.6)



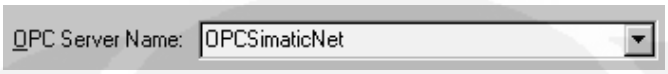
Gambar 3.6. Kotak dialog OPCLink *Topic Definition*

(6) Masukkan nama untuk PLC pada kotak text [Topic Name]. (Ketika komunikasi dengan *Intouch* nama ini digunakan sebagai [topic name] dalam [Access Name Definition].



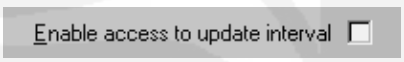
(7) Jika diinginkan penggunaan DCOM untuk penyambungan ke remote *OPC server*, harus diisi nama pada kotal text [Node Name], diisi dengan nama komputer dimana *OPC server* terpasang. (Harus dieksekusi DCOMCNFG dengan klik pada tombol start pada *taskbar* dan pilih Run remote OPC server komputer untuk mengizinkan *OPCLink* atau *OPC client* yang lain untuk mengakses).

(8) Pilih nama *OPC server* yang akan digunakan, daftar nama dalam daftar menampilkan *OPC server* yang ada pada sistem.

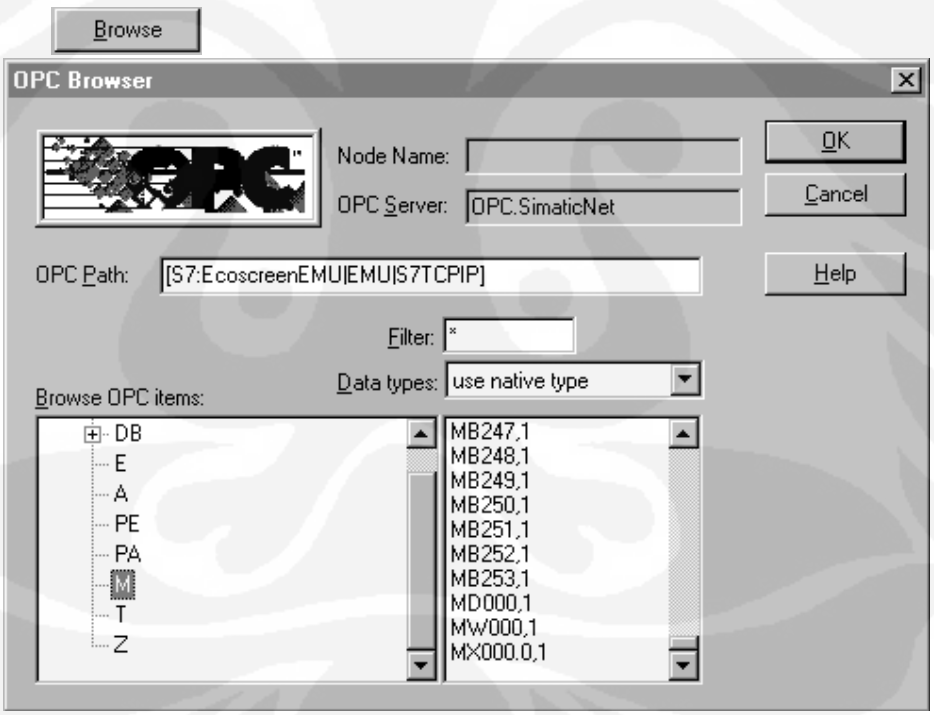


(9) Masukkan frekuensi (dalam milisecond) *OPCLink* akan mengakuisisi data (jika 0 *OPCLink* tidak akan mengakuisisi data dari *OPC server*).

(10) Pilih, pilihan ini untuk mengenable akses *client* untuk mengupdate jeda waktu.



(11) Klik *browse* untuk mengakses *OPC Browser*, kemudian ditampilkan kotak dialog *OPC Browser* (gambar 3.7).



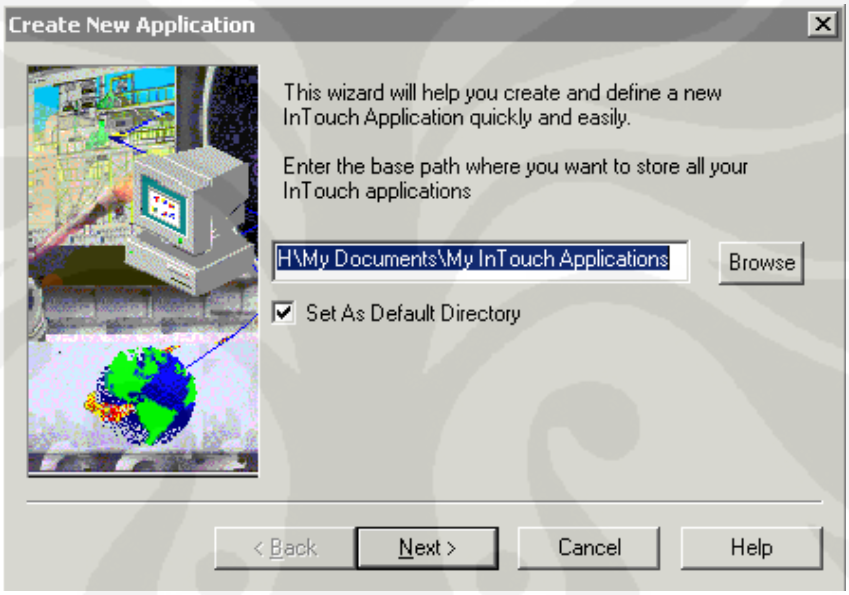
Gambar 3.7. Kotak dialog *OPC browser*

(12) Kemudian klik [OK], dan ditampilkan kotak dialog *Topic Definition* dan klik [Done]

2. Membuat HMI

(1) Setelah *OPC* berfungsi dengan benar jalankan *Intouch software* untuk membuat HMI. Klik dua kali icon *Intouch* pada *desktop*, kemudian akan ditampilkan jendela *Intouch Aplication Manager*

(2) Buat aplikasi baru dengan klik [New], kemudian ditampilkan kotak dialog *Create New Application* (gambar 3.8)



Gambar 3.8. Kotak dialog *Create New Application*

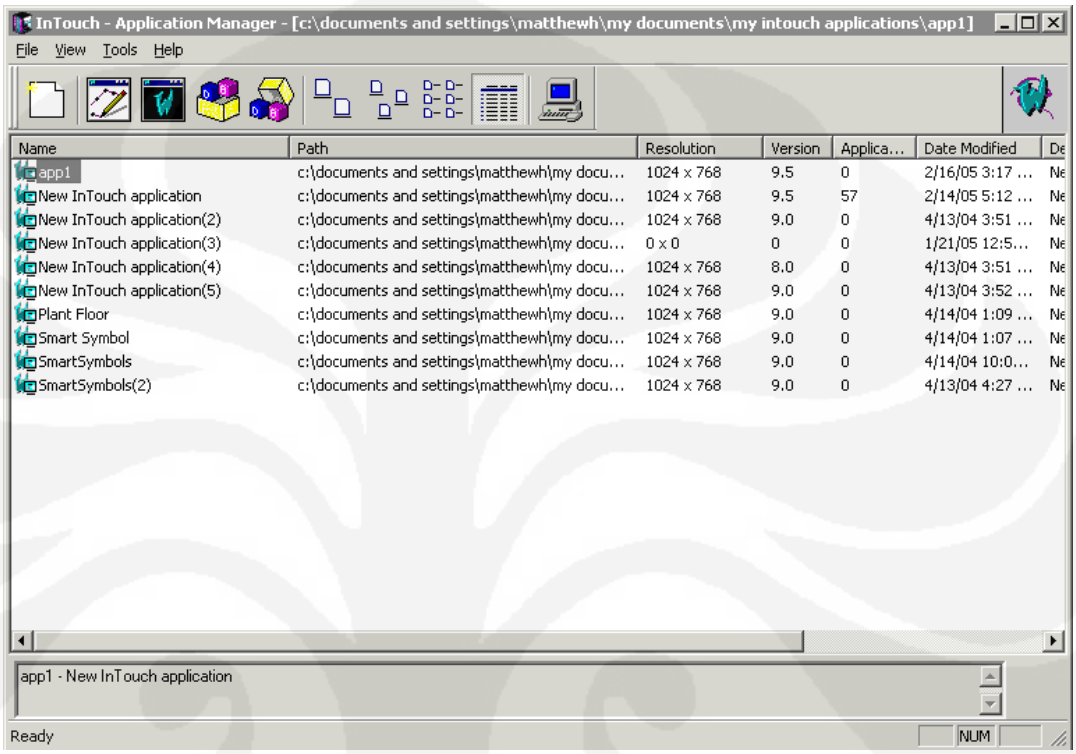
(3) Klik [Browse] untuk menentukan lokasi file akan disimpan.

(4) Klik [OK], kemudian klik [Next], kemudian ditampilkan halaman berikutnya dari kotak dialog *Create New Application*. Nama default untuk aplikasi baru adalah "NewApp", pakai nama ini atau masukkan nama yang lain.

(5) Klik [Next] kemudian ditamplikan halaman ketiga dari kotak dialog *Create New Application*.

(6) Pada kotak nama, masukan nama untuk nama aplikasi baru yang akan muncul ketika aplikasi terdaftar pada jendela *Intouch application Manager*.

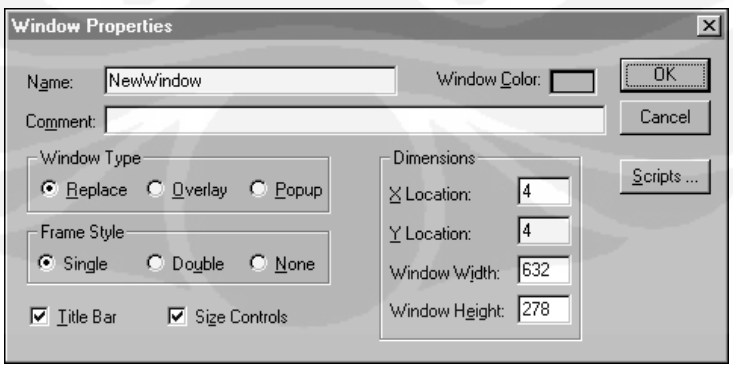
(7) Dalam kotak *Description*, ketik deskripsi aplikasi, kemudian klik [Finish] kemudian jendela *Intouch Application Manager* (gambar 3.9) muncul kembali dengan menampilkan icon aplikasi baru yang telah dibuat.



Gambar 3.9. Jendela *Intouch Application Manager*

(8) Untuk membuka aplikasi klik pada aplikasi yang akan dibuka kemudian klik [windows maker] pada *toolbar*, kemudian akan ditampilkan jendela *Intouch-WindowMakker*.

(9) Buat Jendela baru, klik [file menu], [new window], kemudian ditampilkan kotak dialog *window properties* (gambar 3.10)

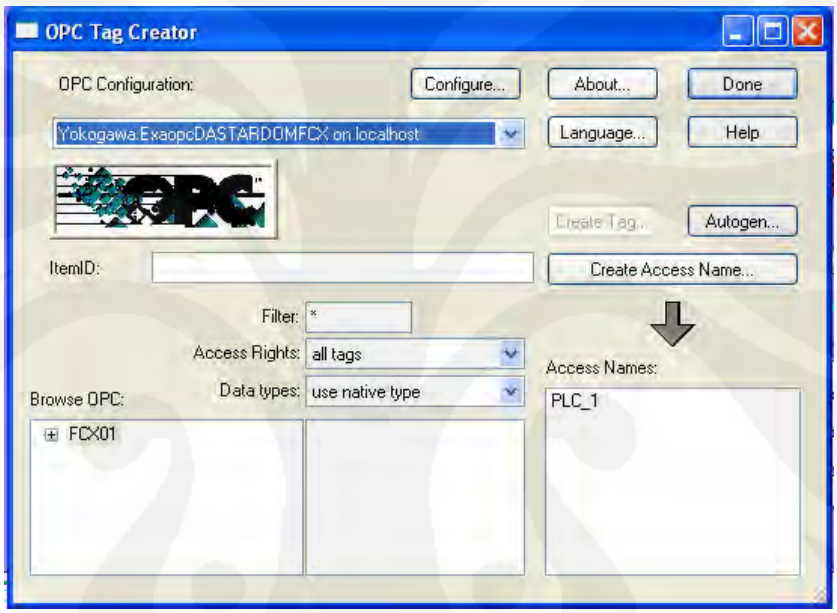


Gambar 3.10. Kotak dialog window properties

(10) Ketik nama pada kotak nama, maksimal 32 karakter, pada kotak *comment* ketik apa saja.

(11) Buka aplikasi baru yang telah dibuat pada window dan akan ditampilkan jendela kosong.

(12) Sebelum melakukan akuisisi data terlebih dahulu dibuat *tag* untuk nantinya dibaca oleh HMI. Klik dua kali [tag creator] kemudian akan ditampilkan kotak dialog *OPC Tag Creator* (gambar 3.11).



Gambar 3.11. Kotak dialog *OPC tag creator*

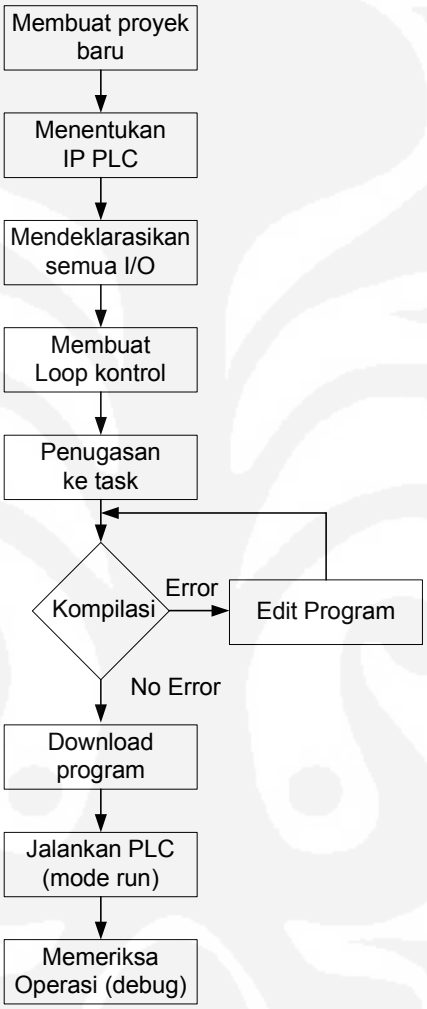
(13) *Browse tag* yang akan ditampilkan di HMI setelah itu klik [Create access name] dan beri nama *tag* dan klik *done*.

(14) Kemudian klik [Text] pada tool dan ketik ###.##, klik dua kali text tersebut, kemudian ditampilkan kotak dialog *object:text*

(15) Klik tombol [analog] pada kolom [value display] dan akan ditampilkan kotak dialog *analog expression*. Pada kolom expression klik dua kali kemudian klik nama *tag* yang telah dibuat sebelumnya menggunakan *tag creator*.

3.2.2. Pemrograman PLC

PLC yang digunakan adalah *Stardom Yokogawa*, untuk membuat program digunakan *Logic designer* dari *Yokogawa*. Diagram alir dari pembuatan program untuk PLC seperti gambar 3.12.



Gambar 3.12. Diagram alir pembuatan program PLC

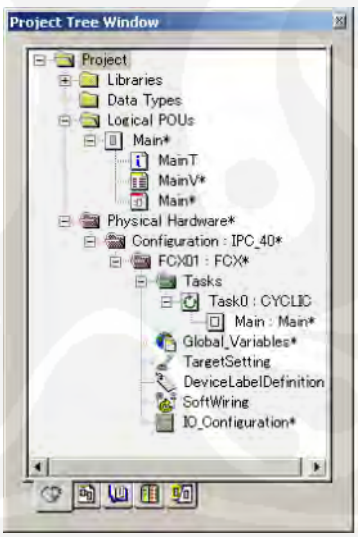
Langkah-langkah pembuatan single loop pengendali aliran dengan metode PID adalah sebagai berikut :

1. **Membuat *project* baru**
   (1) Pilih File-*New Project*
   (2) Akan muncul menu pilihan pada *template New Project*

Gambar 3.13. Jendela menu pilihan *project* baru.

(3) Pilih STARDOM NPAS kemudian klik OK.

(4) Kemudian akan dibuat *Project Tree Windows* (gambar 3.14) secara default.

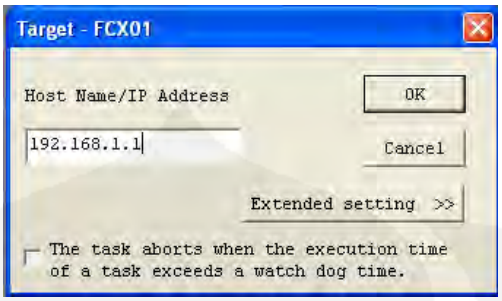


Gambar 3.14. *Project tree windows*

2. Menentukan target pengendali (FCN/FCJ)

(1) Klik dua kali pada [TargetSetting] pada *project tree* untuk menampilkan kotak dialog *Target* (gambar 3.15).

(2) Masukkan “192.168.1.1” untuk *IP address* dan klik [OK].



Gambar 3.15. Kotak dialog target

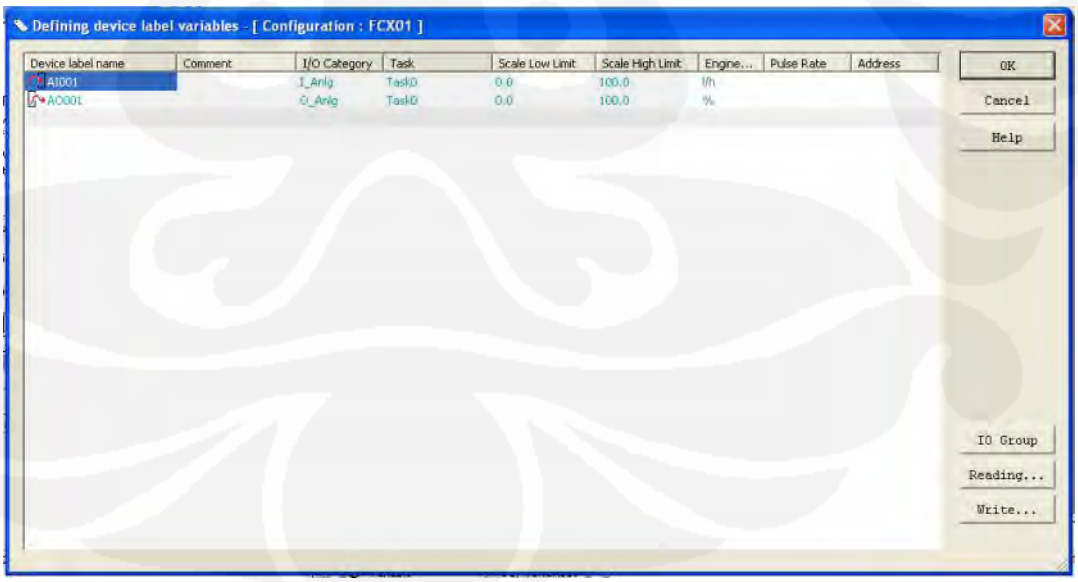
3. **Menetapkan *Device Label Variable***

(1) Klik dua kali [DeviceLabelDefinition] pada *project tree* untuk menampilkan [defining device label variable] (gambar 3.16).

(2) Definisikan dua device label variable seperti berikut ini:

[Analog input]
Device label name: AI001
I/O Category: I_Anlg
Task: Task0
Scale Low Limit: 0.0
Scale High Limit: 100.0
Engineering Unit: l/h

[Analog output]
Device label name: AO001
I/O Category: O_Anlg
Task: Task0
Scale Low Limit: 0.0
Scale High Limit: 100.0
Engineering Unit: %



Gambar 3.16. *Device label definition*

(3) Setelah selesai membuat *definisi device* klik [OK] kemudian klik [YES]

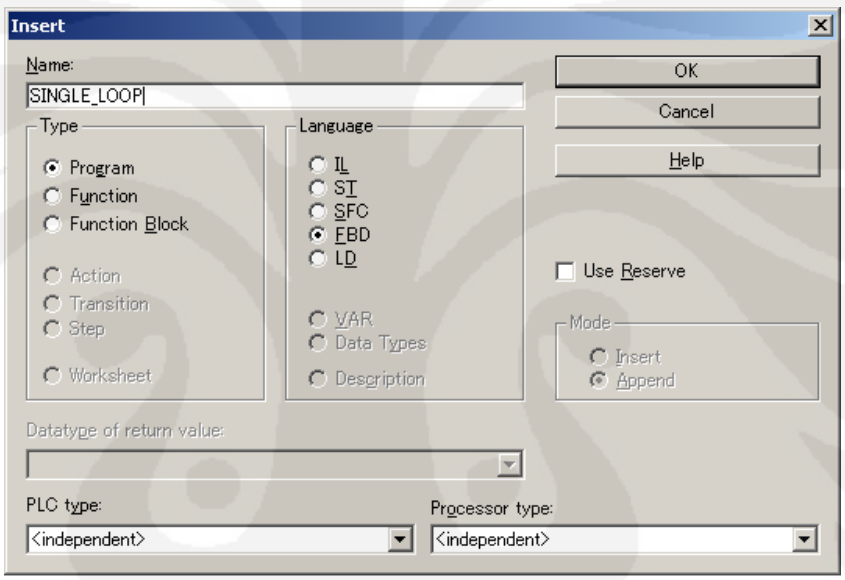
4. Membuat Program

Program ini yang nantinya di masukkan ke PLC sehingga PLC berfungsi sebagai pengendali dengan metode kendali PID.

4.1. Penambahan Program

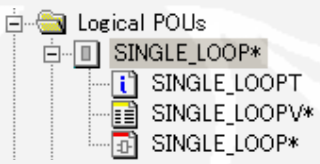
(1) Klik kanan [Logical POU] pada project tree dan kemudian pilih [Insert]-[Program] untuk menampilkan kotak dialog Insert (gambar 3.17).

(2) Masukkan "SINGLE LOOP" pada [Name], pilih [FBD] untuk [Language], kemudian klik [OK].



Gambar 3.17. Kotak dialog *insert*

Dari *project tree*, pastikan bahwa nama POU (program) "SINGLE LOOP" telah terbuat dengan 3 lembar kerja (gambar 3.18). Aplikasi kendali dapat dibuat oleh pemprograman pada 3 lembar kerja tersebut.

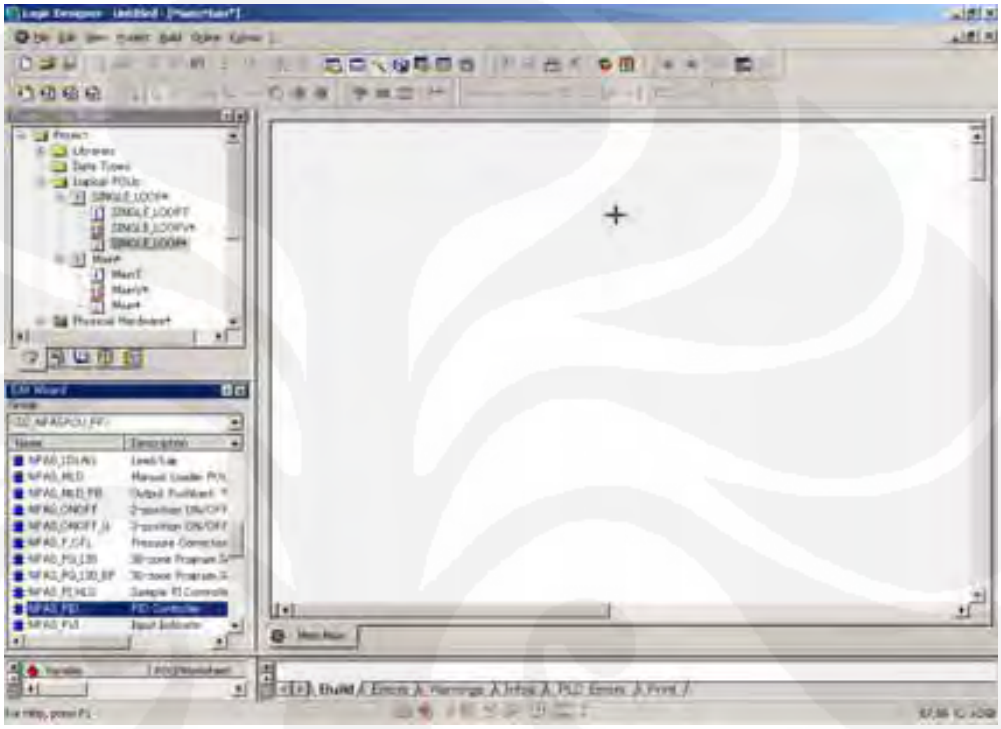


Gambar 3.18. Nama program pada *project tree*

4.2. Penempatan fungsi blok pengendali PID

(1) Klik dua kali [SINGLE_LOOP] *code worksheet* pada project tree untuk membuka *code worksheet*.

(2) Klik *worksheet* untuk menentukan posisi fungsi blok pengendali PID.



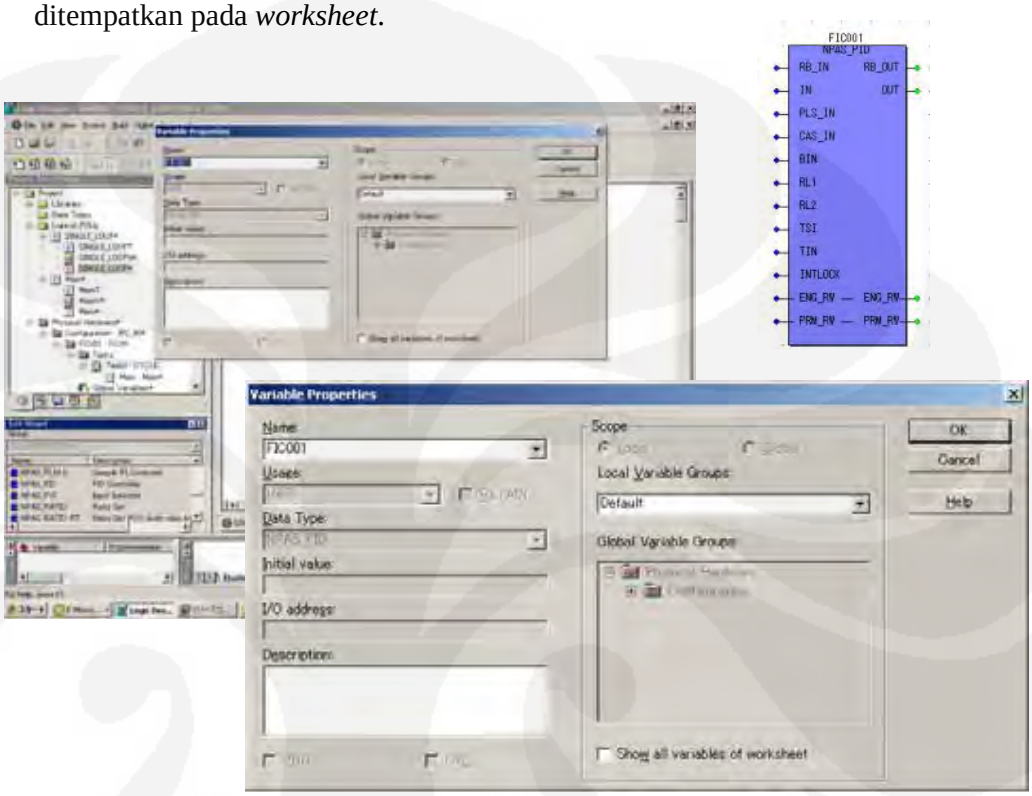
Gambar 3.19. Gambar lembar kerja (*worksheet*)

(3) Pilih "<SD_NPASPOU_PF>" (library) dari [Group] *combo box* [Edit Wizard] (gambar 3.20).

Gambar 3.20. *Combo box Edit Wizard*

(4) Klik dua kali [NPAS_PID (Pengendali PID)]. Kemudian akan di tampilkan dialog *Variable Properties*.

(5) Masukkan "FIC001" dalam [Nama] dan klik [OK]. Fungsi blok ditempatkan pada *worksheet*.



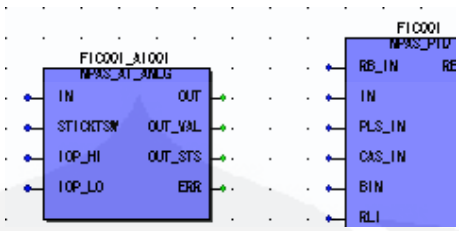
Gambar 3.21. Memasukkan pengendali PID ke lembar kerja

4.3. Membuat Fungsi blok standar analog input

(1) Klik lokasi disebelah kiri [FIC001] untuk menempatkan fungsi blok yang baru.

(2) Klik dua kali [NPAS_AI_ANLG (Standard Analog Input)], kemudian akan ditampilkan dialog *variable properties*. NPAS_AI_ANLG juga terdapat di dalam <SD_NPASPOU_PF> *library*.

(3) Masukkan "FIC001_AI001", dan klik [OK], kemudian fungsi blok ditempatkan pada *worksheet* (gambar 3.22).



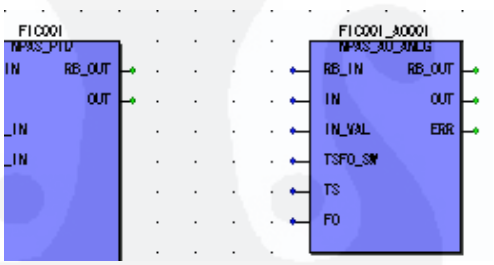
Gambar 3.22. Fungsi blok analog input

4.4. Membuat fungsi blok standar analog output

(1) Klik lokasi pada bagian kanan [FIC001] untuk menempatkan fungsi blok.

(2) Klik dua kali [NPAS_AO_ANLG (Standard Analog Output POU)], kemudian akan ditampilkan dialog *variable properties*. NPAS_AO_ANLG juga terdapat di dalam <SD_NPASPOU_PF> library.

(3) Masukkan "FIC001_AO001" dan klik [OK]. Fungsi blok di tempatkan pada *worksheet* (gambar 3.23).

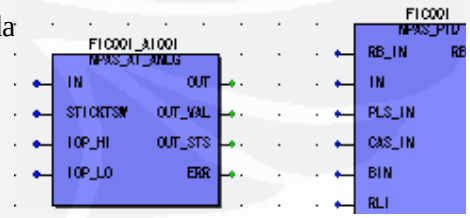


Gambar 3.23. Fungsi blok analog output

4.5. Menghubungkan fungsi blok

(1) Pilih Connect Objects icon pada toolbar (gambar 3.24(a)).

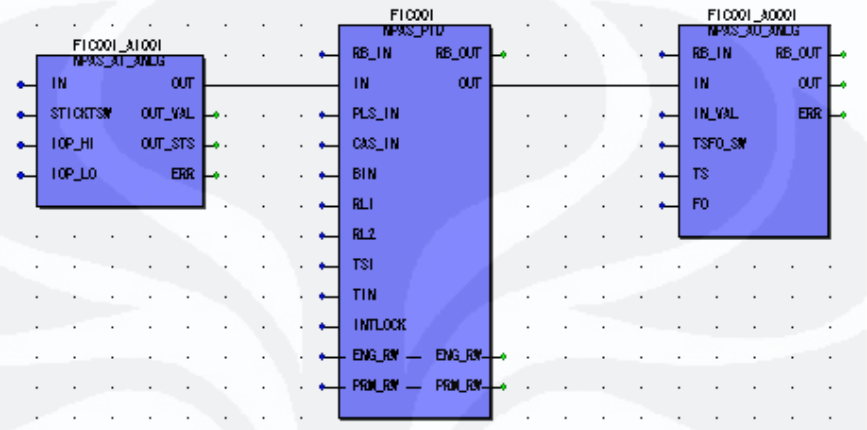
(2) Klik lingkaran hijau [OUT] pada [FIC001_AI001], diikuti oleh lingkaran biru untuk [IN] pada



(a) (b)

Gambar 3.24. (a) *Icon* untuk menyambung *object* (b) Analog input dan pengendali PID

(3) Sambungkan [OUT] pada [FIC001] dan [IN] pada [FIC001_AO001]



Gambar 3.25. Hasil penyambungan analog input, pengendali dan analog output

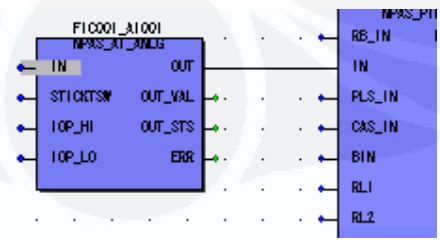
4.6. Penyambungan Device Label Variable.

(1) Klik dua kali lingkaran biru untuk [IN] pada [FIC001_AI001], kemudian akan di tampilkan dialog *variable properties* dialog.

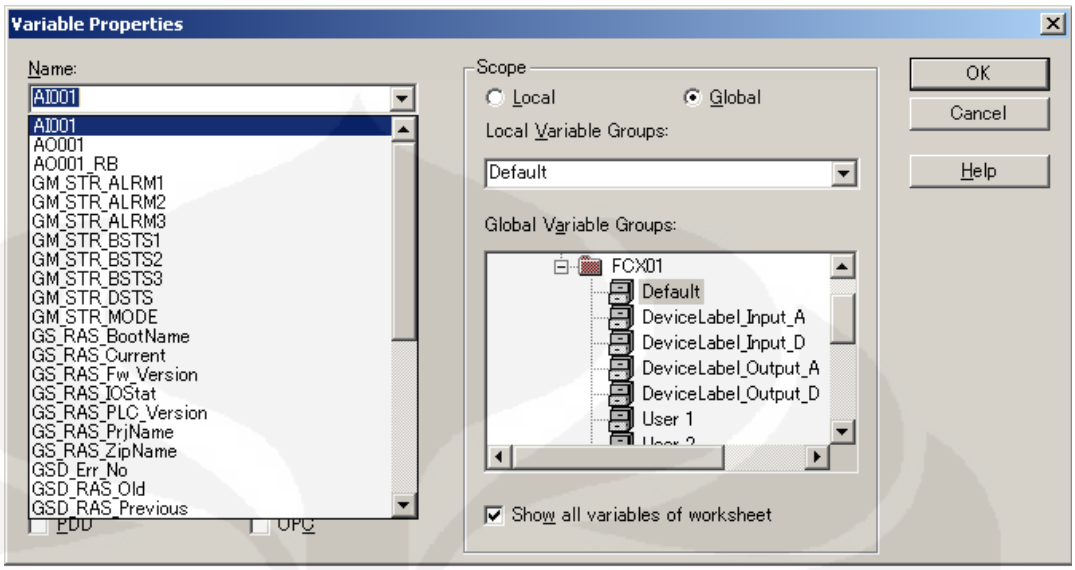
(2) Device label variable adalah global variable. Untuk menampilkan device label variable "AI001" pada daftar *variable*, yang sudah ditentukan *resource* (FCN/FCJ) dimana itu seharusnya.

Dibawah [Global Variable Group], pilih [Physical hardware]-[configuration]-[FCX01]-[Default]. Ubah [scope] ke [global], dan on kan [Show all variable of worksheet] *checkbox*.

(3) Klik [Name] *combo box*, dan pilih [AI001] dari tampilan daftar *global variable* pada *resource specified* pada tahapan di atas klik [OK].



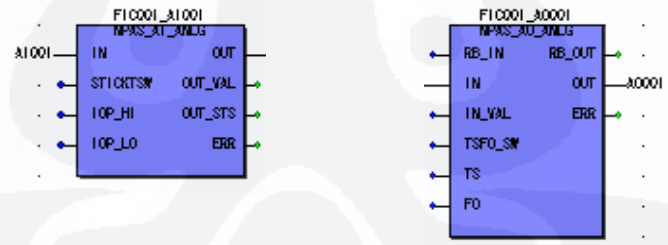
Gambar 3.26. Penyambungan analog input dan pengendali.



Gambar 3.27. Penugasan analog input ke *hardware*.

(4) Sambungkan device label variable "AO001" sebagai output "FIC001", klik dua kali lingkaran hijau untuk [OUT] pada [FIC001_AO001]. Ditampilkan dialog *variable properties*.

(5) Sejak scope dipilih pada tahapan diatas, jangan diubah. Pilih [AO001] dan klik [OK].



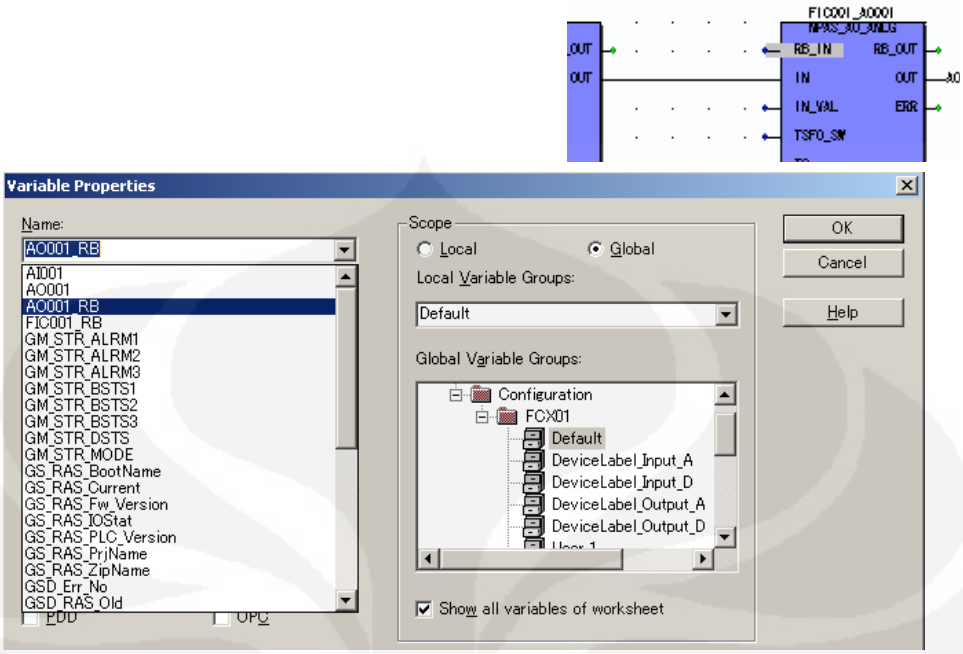
(a) (b)

Gambar 3.28. (a) fungsi blok analog input (b) Fungsi blok pengendali

4.7. Penyambungan Read Back Variable

(1) Klik dua kali lingkaran biru untuk [RB_IN] pada [FIC001_AO001], kemudian ditampilkan dialog *variable properties* (gambar 3.29).

(2) Ubah [scope] ke [global], klik [Name] *combo box* dan pilih [AO001_RB] dari daftar *global variable*, klik [OK].

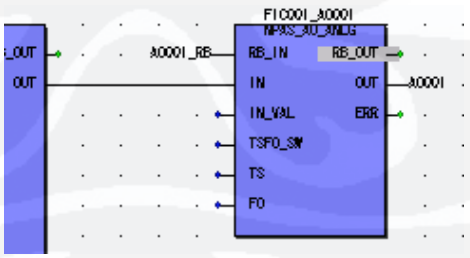


Gambar 3.29. Penugasan pembacaan kembali nilai output

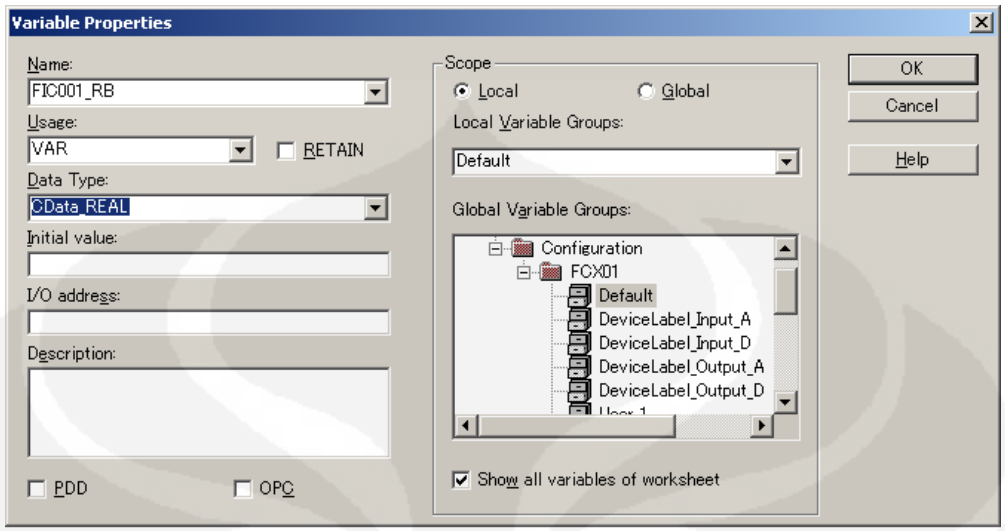
Saat ini sudah ditentukan pembacaan kembali untuk device *label variable* "AO001" ke "FIC001_AO001". Didapatkan *state read back* "FIC001_AO001" ke "FIC001".

(3) Klik dua kali [RB_OUT] pada [FIC001_AO001], kemudian akan ditampilkan dialog *variable properties* (gambar 3.30).

(4) Pilih [Local] untuk [Scope], masukkan [FIC001_RB] untuk [Name], pilih [CData_REAL] untuk [Data Type], dan klik [OK]. Kemudian ditambahkan *variable read back*.



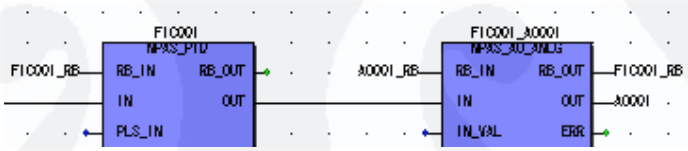
Gambar 3.30. Hasil penempatan variable pembacaan kembali



Gambar 3.31. Penugasan pembacaan kembali ke pengendali

(5) Selanjutnya klik dua kali [RB_IN] pada [FIC001], kemudian ditampilkan dialog *variable properties*.

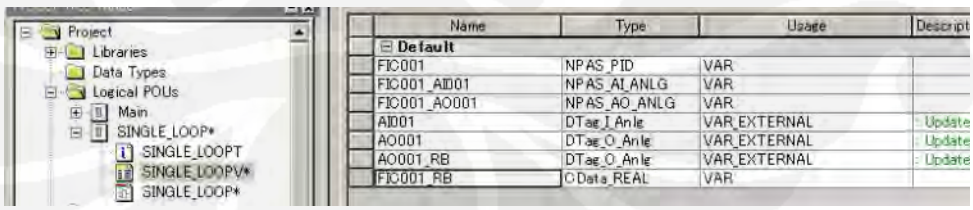
(6) Pilih [FIC001_RB] dalam [Name] combo box dan klik [OK]



Gambar 3.32. Hasil dari penempatan *variable* pembacaan kembali

4.8. Pemeriksaan *Variable*

(1) Klik dua kali *variable worksheet* [SINGLE_LOOPV] pada *project tree* untuk membuka *variable worksheet*. *Variable* ditampilkan dengan tipe [VAR_EXTERNAL] Pada variable worksheet adalah *global variable*.



Gambar 3.33. Pemeriksaan *variable*

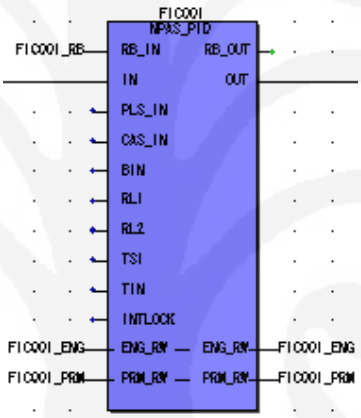
5. Penyimpanan Project

(1) Pilih [File]-[Save Project As/Zip Project As] dari menu.

(2) Masukkan “skripsi” untuk nama *project* kemudian klik [Save].

6. Memasukkan nilai ke Engineering Parameter.

Engineering parameter dapat digunakan untuk mengatur komponen, batas atas histeresis, batas bawah histeresis dan informasi lain untuk NPAS POU ketika pembuatan aplikasi kendali. Semua *engineering* parameter pada NPAS POU adalah didefinisikan sebelumnya dengan nilai *default* untuk kenyamanan. Nilai default itu dapat diubah sesuai keinginan menggunakan *user defined variable* mengacu sebagai “*Variable* untuk memberi nilai pada *engineering* parameter”.



Gambar 3.34. Penempatan *variable* untuk akses parameter dan *engineering* parameter.

6.1. Pendefinisian variable untuk memberi nilai pada engineering parameter.

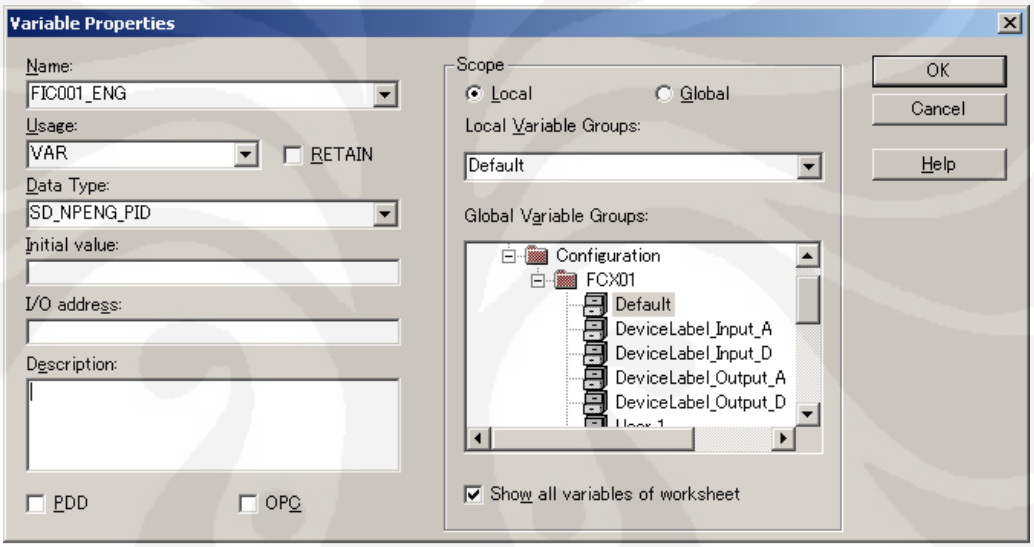
Seperti namanya, “*variable* untuk memberi nilai pada *engineering* parameter” ini digunakan untuk memberi nilai pada *engineering* parameter. Untuk mengubah nilai *default* pada engineering parameter, ditetapkan “*variable* untuk pemberian nilai pada *engineering* parameter”, dan kemudian mengubah nilai *variable* ini. Langkah-langkah nya sebagai berikut:

(1) Klik dua kali “SINGLE_LOOP” *code worksheet* pada *project tree* untuk membuka *code worksheet*.

(2) Klik dua kali [ENG_RW] pada [FIC001], kemudian ditampilkan dialog *variable properties* (gambar 3.36).

Gambar 3.35. Letak pin akses parameter dan *engineering* parameter.

(3) Pastikan *scope* adalah *local*, masukkan [FIC001_ENG] pada [Name], dan klik [OK]. Tipe data secara otomatis [SD_NPENG_PID].

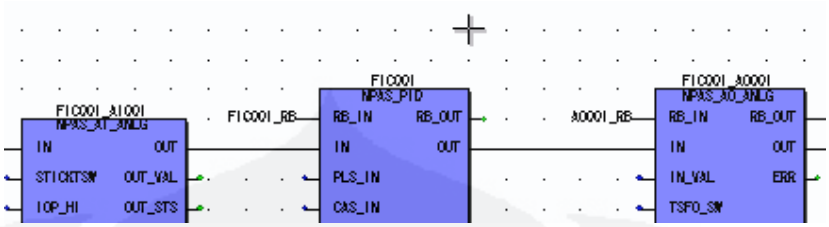


Gambar 3.36. Pembuatan *variable* untuk *engineering* parameter.

6.2. Pemberian nilai pada Engineering Parameter.

Beri nilai untuk *comment* pada *engineering* parameter. Untuk memberi nilai pada *engineering* parameter, masukkan data ke *variable* untuk memberi nilai pada engineering parameter.

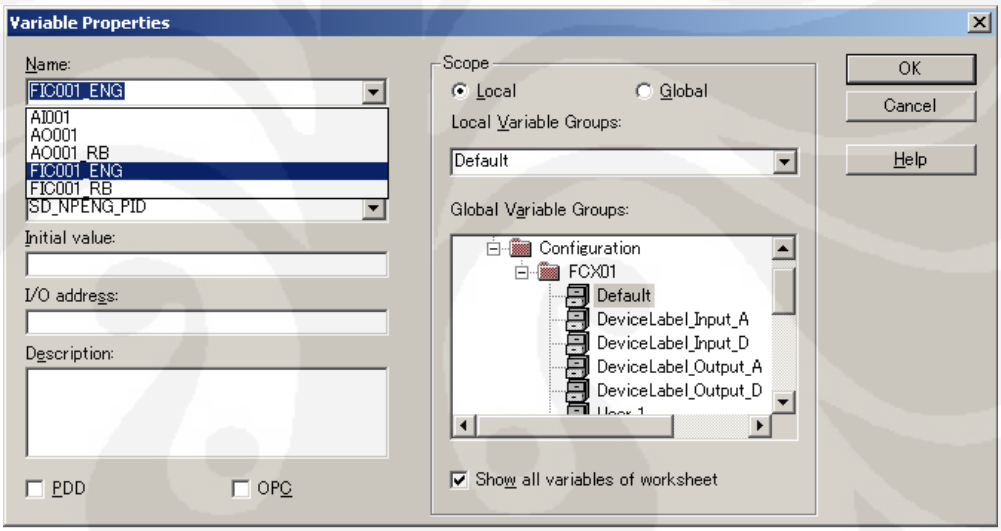
(1) Klik lokasi diatas [FIC001] untuk menentukan posisi untuk memasukkan *variable* untuk memberi nilai pada *engineering* parameter.



Gambar 3.37. Memasukkan *variable* untuk *engineering* parameter

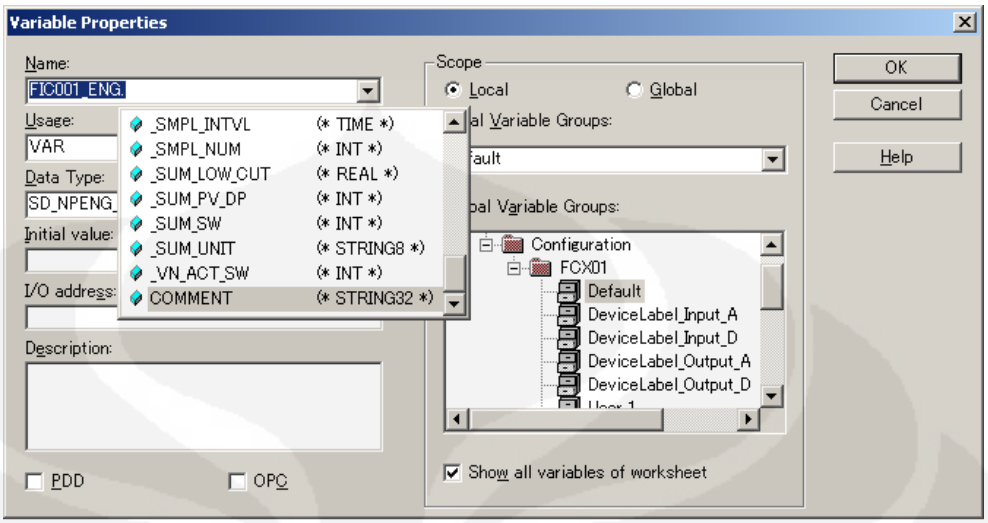
(2) Klik kanan dan pilih [Variable] dari display menu, kemudian ditampilkan dialog *variable properties*.

(3) Klik [Name] *combo box*, dan kemudian pilih [FIC001_ENG] dari tampilan daftar *local variable*.



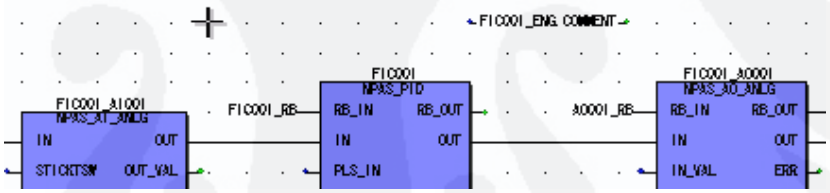
Gambar 3.38. *Variable engineering* parameter

(4) Masukkan karekter (‘.’) setelah “FIC001_ENG”, dan pilih [COMMENT] dari tampilan *menu variable* untuk pemberian nilai pada *engineering* parameter, klik [OK].



Gambar 3.39. *Variable comment* untuk *engineering* parameter

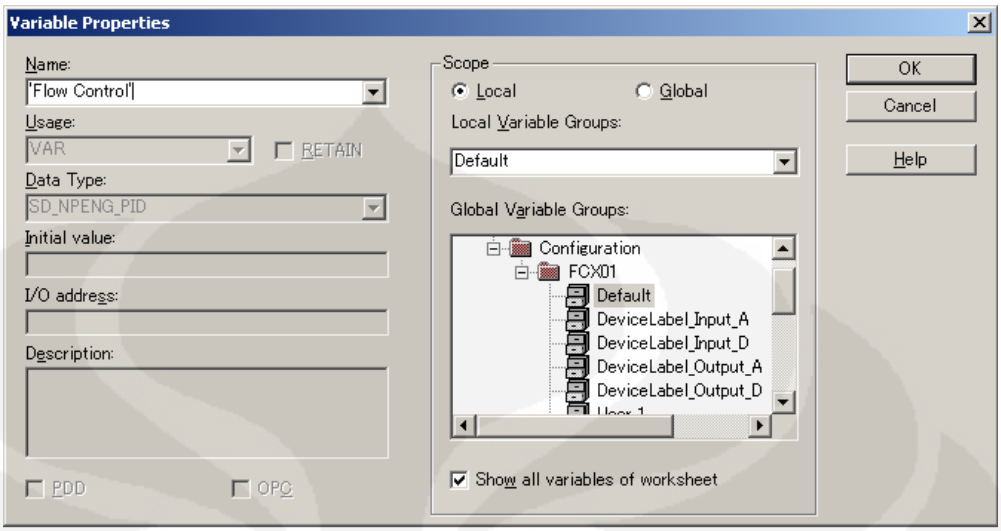
(5) Selanjutnya, klik lokasi pada *worksheet* pada sebelah kiri [FIC001_ENG.COMMENT] untuk menentukan lokasi untuk menempatkan nilai yang menjadi tugas untuk *user variable*.



Gambar 3.40. Penempatan *variable comment* pada *worksheet*

(6) Klik kanan, pilih [Variable] dari tampilan *menu*, kemudian ditampilkan dialog *variable properties* (gambar 3.41).

(7) Masukkan “Flow Control”, dan klik [OK]. Sebagai catatan jika memasukkan *string* harus di dalam tanda (‘).

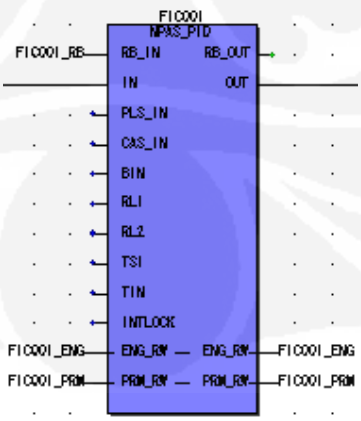


Gambar 3.41. Pemberian *comment* 'Flow Control' untuk *engineering* parameter.

(8) Pilih *Connect Objects icon*, dan sambungkan 'Flow Control' dengan FIC001_ENG.COMMENT.

7. Pendefinisian Variable untuk pembacaan dan penulisan Access Parameter.

Access parameter bisa digunakan untuk menampilkan atau mengeset PV, SV, MV, MODE, PH, PL, P, I, D dan parameter lain selama operasi. Untuk membaca dan menulis akses parameter ini dari aplikasi yang lain, *user-defined variable* digunakan mengacu sebagai "*variable* untuk membaca dan menulis akses parameter".

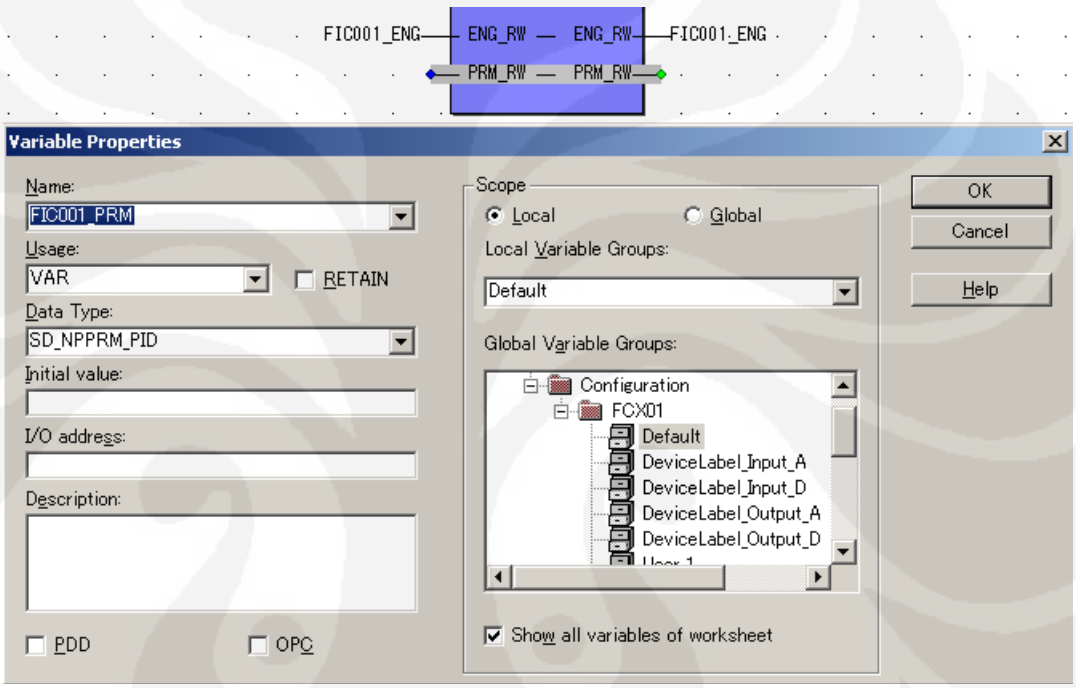


Gambar 3.42. *Variable* akses parameter

8. Pendefinisian variable untuk pembacaan dan penulisan akses parameter.

(1) Klik dua kali lingkaran hijau [PRM_RW] pada [FIC001], kemudian ditampilkan dialog *variable properties*.

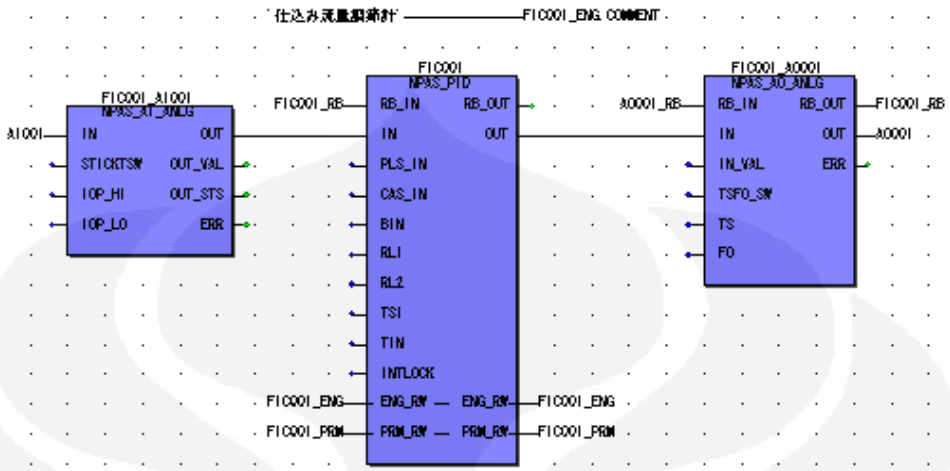
(2) Pastikan *scope* adalah *local*, masukkan [FIC001_PRM] dalam [Name], dan klik [OK], tipe data secara otomatis [SD_NPRM-PID]



Gambar 3.43. Pembuatan *variable* untuk akses parameter

9. Pemeriksaan dan penyimpanan loop kendali yang dibuat.

Sejauh ini telah dibuat loop kendali, hal terakhir yang perlu dilakukan untuk memeriksa loop kendali yang telah dibuat (gambar 3.44).



Gambar 3.44. Hasil akhir dari fungsi blok loop kendali

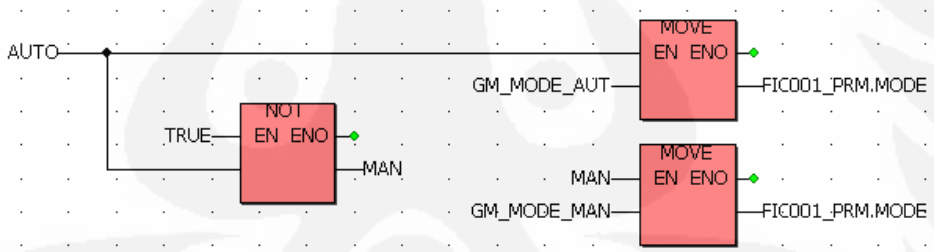
Simpan program yang telah dibuat

(1) Pilih [File]-[Save] dari menu.

10. Pembuatan mode PID

(1) Buka “SINGLE_LOOP” code worksheet, klik [OK].

(2) Seperti instruksi sebelumnya buat aplikasi seperti gambar 3.45.



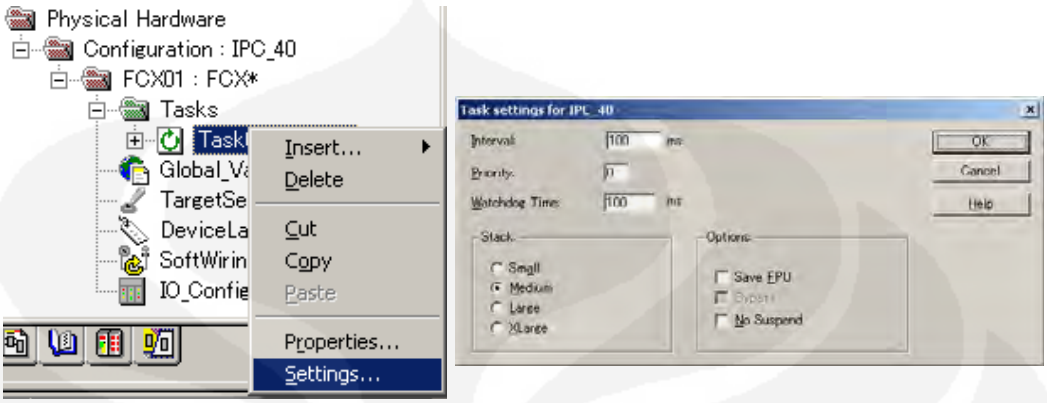
Gambar 3.45. Fungsi blok untuk mode pengendali (Auto atau Manual)

11. Pengujian task

Setelah dibuat aplikasi kendali, untuk mengeksekusi program “SINGLE_LOOP” diperlukan penugasan pada *task*.

(1) Pilih [Task0] pada *project tree*, klik kanan dan pilih [Properties] dari menu, *task0* adalah *cycle task*, setelah pengujian klik [Cencel] untuk menutup dialog.

(2) Pilih [Task0] pada *project tree*, klik kanan, pilih [Setting] dari menu, kemudian ditampilkan dialog [Task setting for IPC_40] (gambar 3.46).

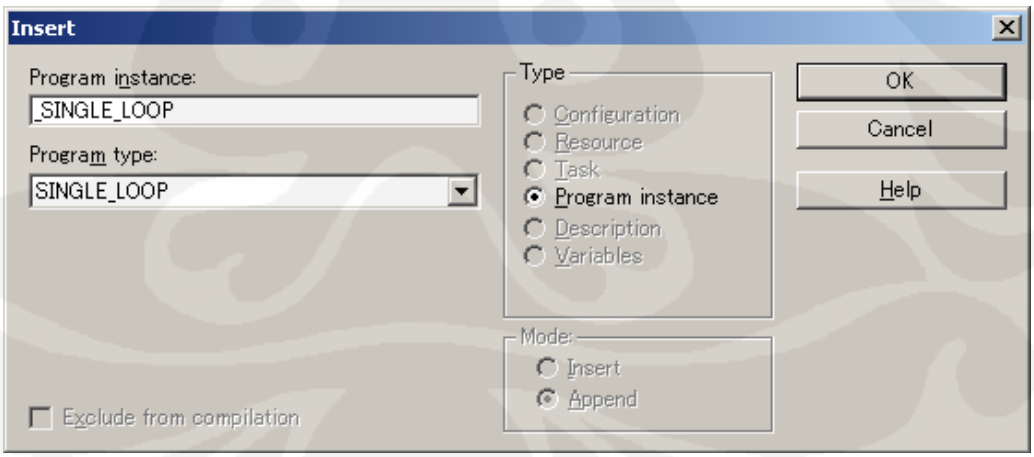


Gambar 3.46. Penugasan program

12. Penugasan program

(1) Pilih [Task0] pada *project tree*, klik kanan, dan pilih [Insert]-[Program instance] dari *popup* menu, kemudian ditampikan dialog *insert* (gambar 3.47).

(2) Masukkan "SINGLE_LOOP" dalam [Program instance], dan pilih [SINGLE_LOOP] untuk [Program type], klik [OK]



Gambar 3.47. Pemberian nama *task*

13. Penghapusan tugas pada task

Jika program "Main" telah terdaftar dalam *template* dihapus, *task* untuk program "Main" menjadi tidah diperlukan.

(1) Pilih [Main] *program instance*, klik kanan, pilih [Delete] dari *menu* (gambar 3.48).

(2) Klik [OK] ketikan dialog konfirmasi *display* untuk delete.

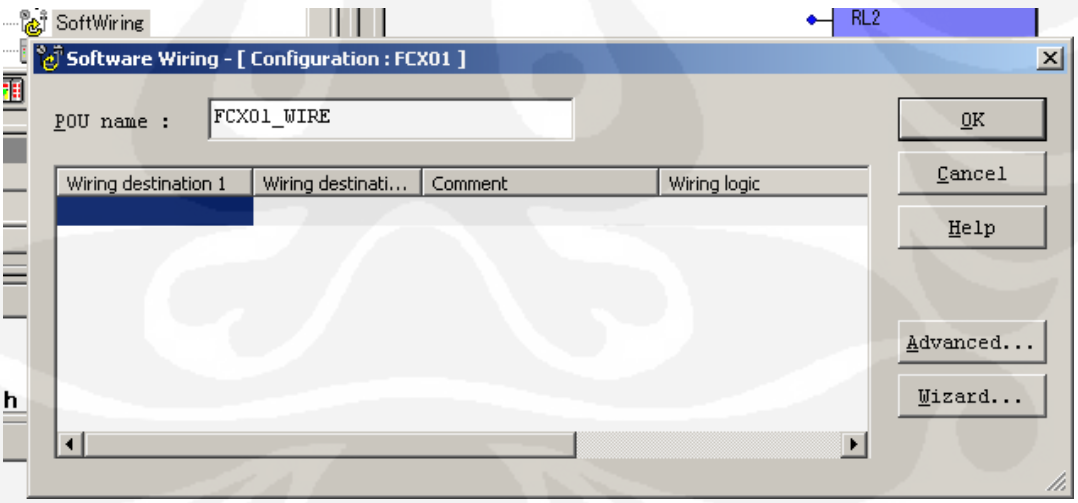
Gambar 3.48. Penghapusan task yang tidak perlu

14. Software wiring

Disini akan dideskripsikan bagaimana menggunakan fungsi *software wiring* untuk melakukan *debug* tanpa menggunakan sinyal input *external*. Fungsi *software wiring* bisa digunakan untuk menyambung input dan output logika kendali oleh perangkat lunak.

(1) Klik dua kali "SoftWiring" pada *project tree* untuk membuka *Software Wiring* dialog.

(2) Klik [Wizard].



Gambar 3.49. Kotak dialog *software wiring*

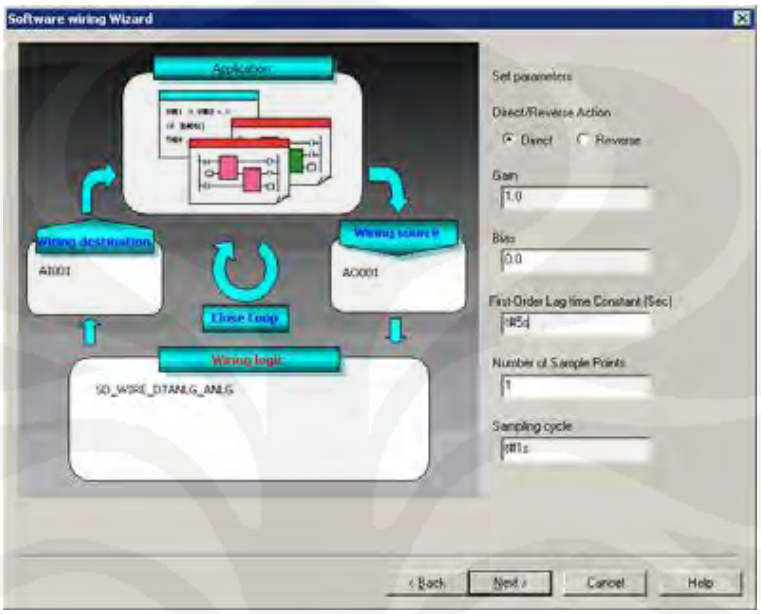
(3) Pilih [AI001] untuk [Wiring destination 1], klik [Next].

(4) Pilih [Close Loop] untuk [Input Form], klik [Next].

(5) Pilih [AO001] untuk [Wiring source 1], klik [Next].

(6) Pilih [SD_WIRE_DTANLG_ANLG] untuk [Wiring Logic], klik [Next].

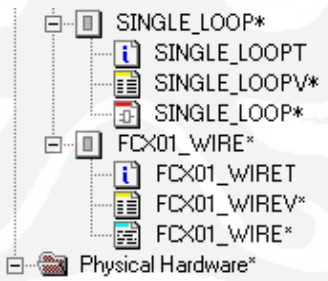
(7) Masukkan "t#5s" (5second) dalam [First-Order Lag time Constant], tinggalkan nilai default, klik [Next] (gambar 3.50).



Gambar 3.50. Kotak dialog *software wiring wizard*

(8) Klik [Finish], keluar *wizard.*

(9) Klik [OK] untuk menutup *Software wiring* dialog, [FCX01_WIRE] ditambahkan dibawah [Logic POU] dalam *project tree* (gambar 3.51).



Gambar 3.51. *Software wiring* dalam *project tree*

15. Penugasan Software wiring ke Task.

Seperti program, *software wiring* tidak bisa dieksekusi kecuali ditugaskan ke *task*.

(1) Pilih [Task0] pada *project tree*, klik kanan, pilih [Insert]-[Program instance] dari *popup menu*, kemudian ditampilkan dialog *insert*.

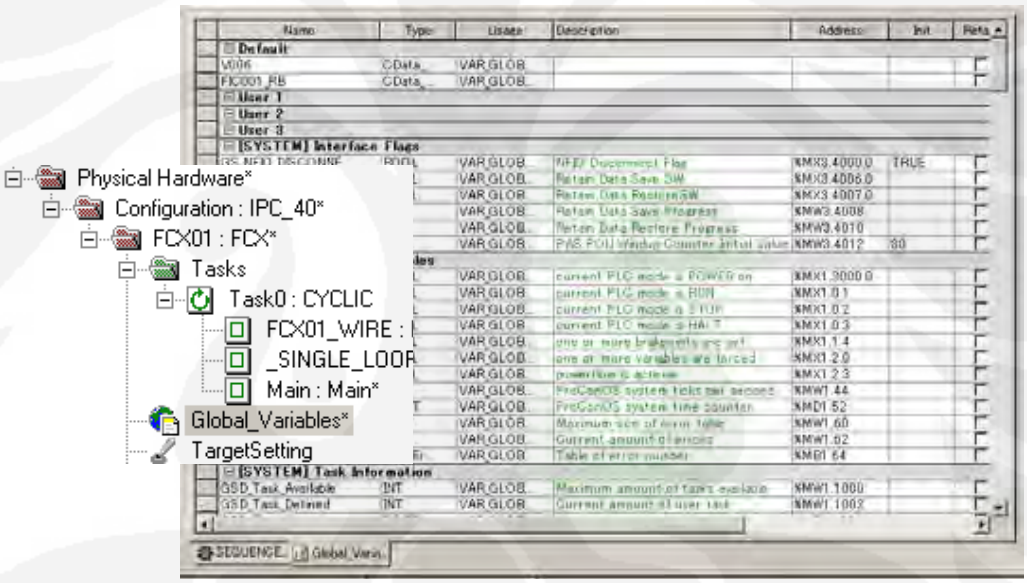
(2) Masukkan “FCX01_WIRE” dalam [Program Instance], dan pilih [FCX01_WIRE] untuk [Program type], klik [OK].

16. Ketidak sambungan I/O

Untuk menggunakan *software waring* dalam proses I/O, pertama I/O harus tidak tersambung. Ini bisa dilakukan dengan merubah nilai *switch* untuk memutuskan I/O, yang disediakan oleh *global variable*.

(1) Klik dua kali [Global _Variable] pada *project tree* untuk membuka *variable worksheet*.

(2) Ubah nilai *inisial* [GS_NFIO_DISCONNF], yang merupakan sakelar pemutus I/O menjadi TRUE (gambar 3.52).

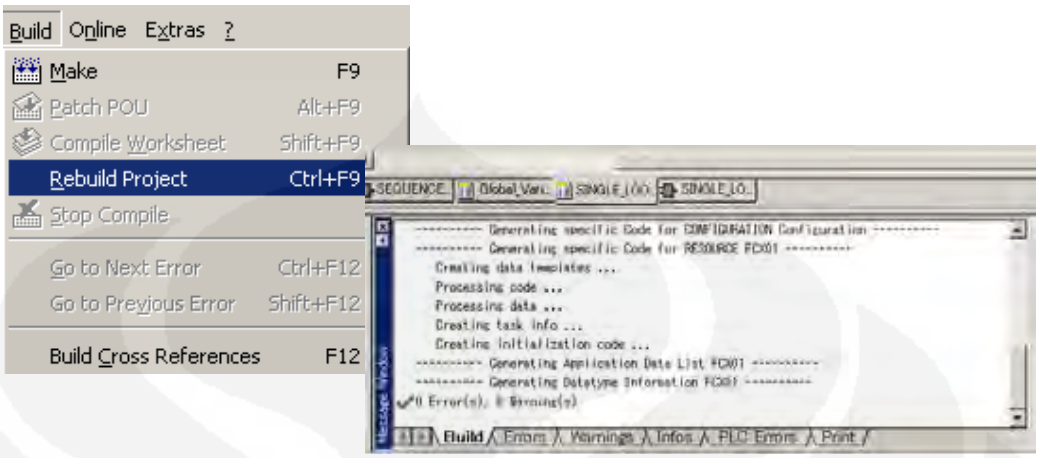


Gambar 3.52. Mengubah nilai GS_NFIO_DISCONNF

17. Kompilasi Project

(1) Pilih [Build]-[Rebuild Project] dari *menu bar*, dialog pesan menunjukkan kemajuan dari kompilasi yang dilakukan.

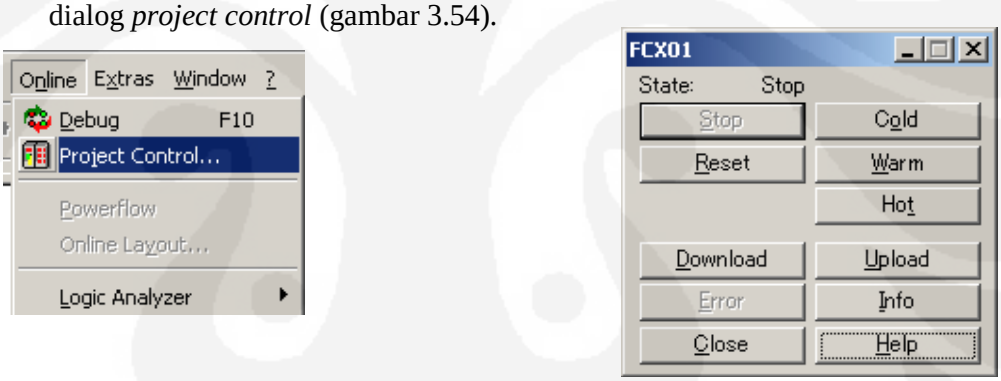
(2) Ketika proses kompilasi selesai, menghasilkan tampilan di jendela pesan, jika tampilan menunjukkan *error* 0 itu berarti proses kompilasi berhasil.



Gambar 3.53. Proses kompilasi

18. Download Project

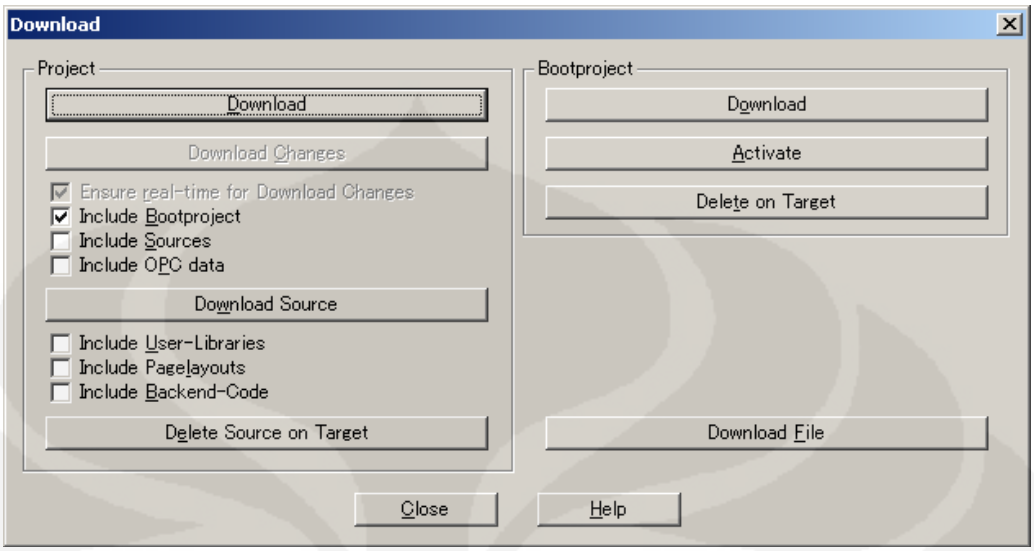
(1) Pilih [Online]-[Project Control] dari *menu bar*, kemudian ditampilkan dialog *project control* (gambar 3.54).



Gambar 3.54. Dialog project control

(2) Klik [Download], download dialog ditampilkan (gambar 3.55).

(3) Klik [Download] tombol dibawah [Project].



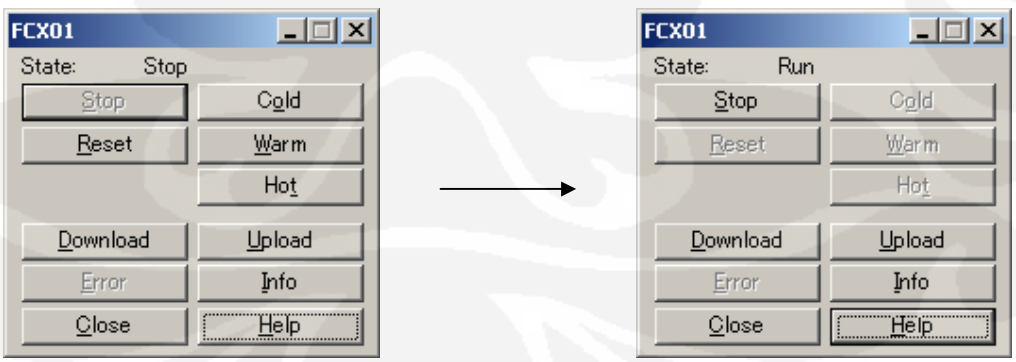
Gambar 3.55. Kotak dialog *download*

(4) Jika *project* lain telah di*download* sebelumnya, akan ditampilkan dialog untuk memastikan apakah *overwrite project* sebelumnya, pilih [Yes].

(5) Kemajuan *download* ditunjukkan pada status *bar*, tunggu sampai proses *download* lengkap (ketika 100% terdownload).

19. Runing Project.

(1) Klik tombol [Cold] pada Project Control dialog, display state on dialog berubah "Run" (gambar 3.56).



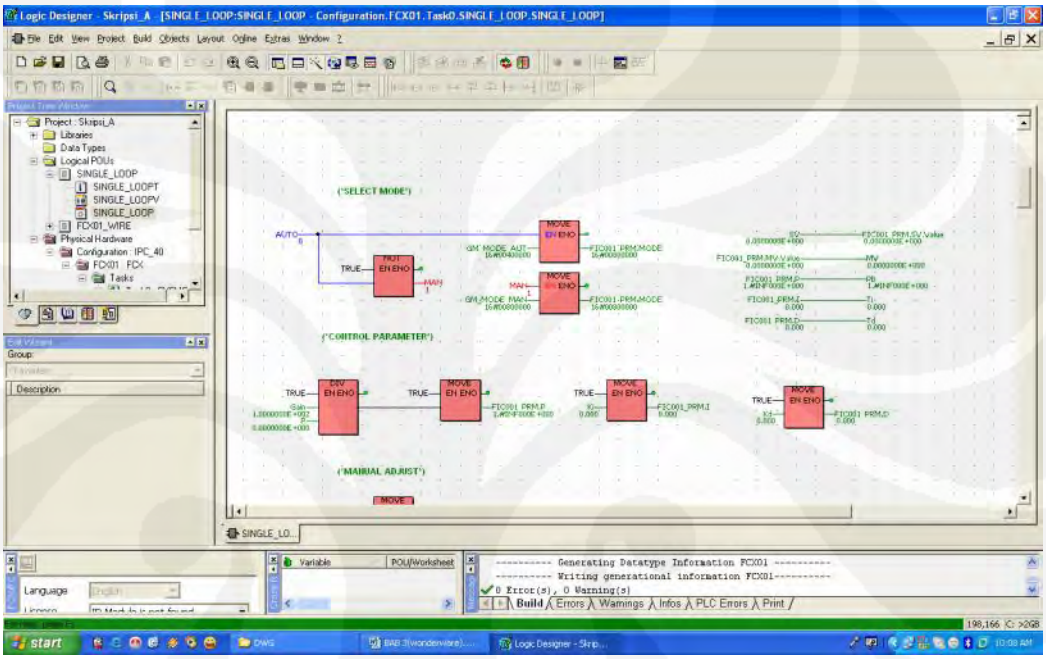
Gambar 3.56. Mengubah status PLC

20. Periksa operasi

Tahap berikutnya pengujian apakah aplikasi kendali berjalan pada FCN/FCJ sudah benar. Pertama *switch logic designer* ke mode *debug* dengan prosedur sebagai berikut:

(1) Buka "SINGLE_LOOP" *code worksheet*.

(2) Klik [Debug on/off*] icon pada toolbar*.



Gambar 3.57. Mode debug PLC

Dalam mode *debug*, pada *logic designer* akan ditampilkan warna hijau, *variable* tipe BOOL akan ditampilkan dengan warna biru jika FALSE tetapi ditampilkan warna merah jika TRUE. Beberapa tipe lain dari variable ditampilkan dengan warna hijau dengan nilai ditampilkan dibawah *variable* seperti pada gambar 3.57.

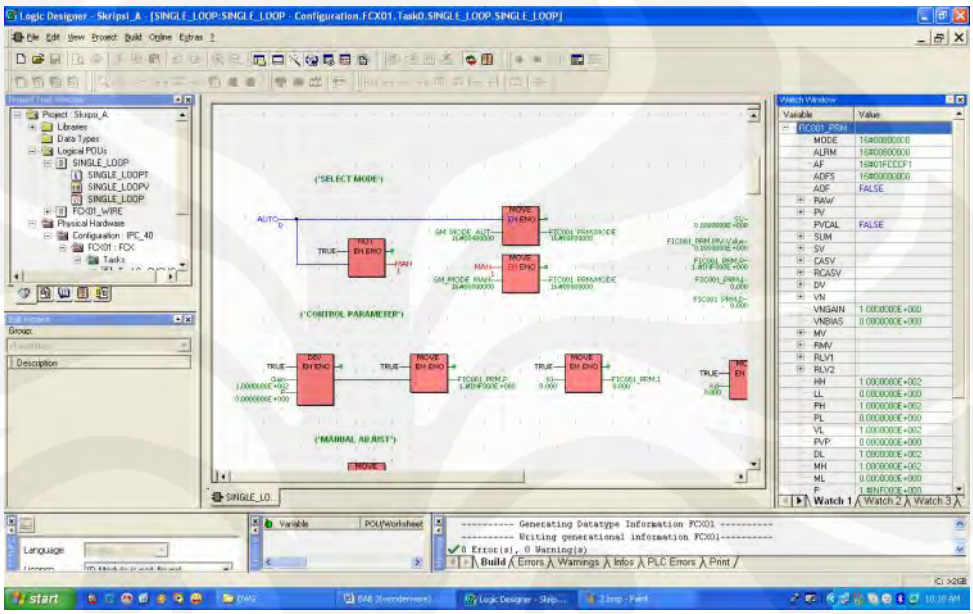
21. Menampilkan akses parameter.

Selanjutnya akan ditampilkan akses parameter pengendali "FIC001" untuk menguji nilai PV, SV dan MV.

23.1. Menampilkan Watch Window

Watch window digunakan untuk menampilkan akses parameter dengan prosedur sebagai berikut:

(1) Pilih [View]-[Watch Window] dari menu untuk menampilkan *watch window* (gambar 3.58).



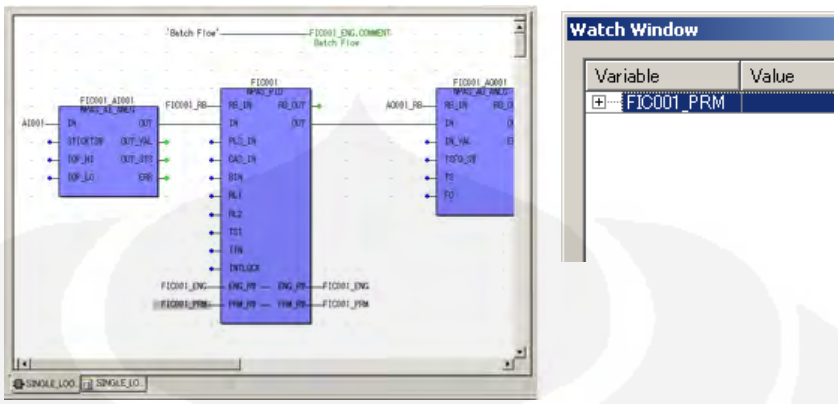
Gambar 3.58. Menampilkan *watch window*

23.2. Penambahan variable untuk baca dan tulis akses parameter pada watch window.

Untuk menampilkan nilai akses parameter pada *watch window*, pertama diperlukan penugasan *variable* untuk baca dan tulis akses parameter pada *watch window* dengan prosedur sebagai berikut:

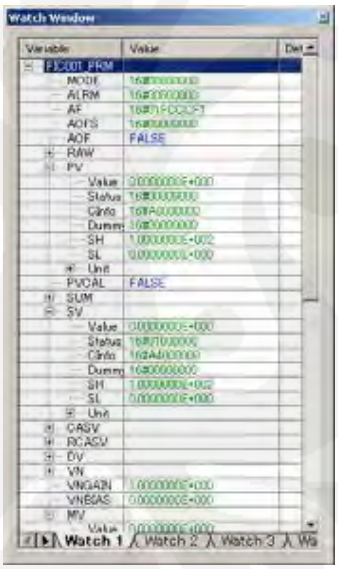
(1) Tampilkan “SINGLE_LOOP” *code worksheet* (gambar 3.59).

(2) Pilih *variable* untuk baca dan tulis akses parameter (“FIC001_PRM”) pada “FIC001” pada *worksheet*, klik kanan dan pilih [Add to Watch Window] dari *popup menu*.



Gambar 3.59. Menambahkan akses parameter pengendali ke *watch window*

(3) Selanjutnya perpanjang [FIC001_PRM] untuk menampilkan nilai PV, SV dan MV (gambar 3.60).



Gambar 3.60. Akses parameter dalam *watch window*

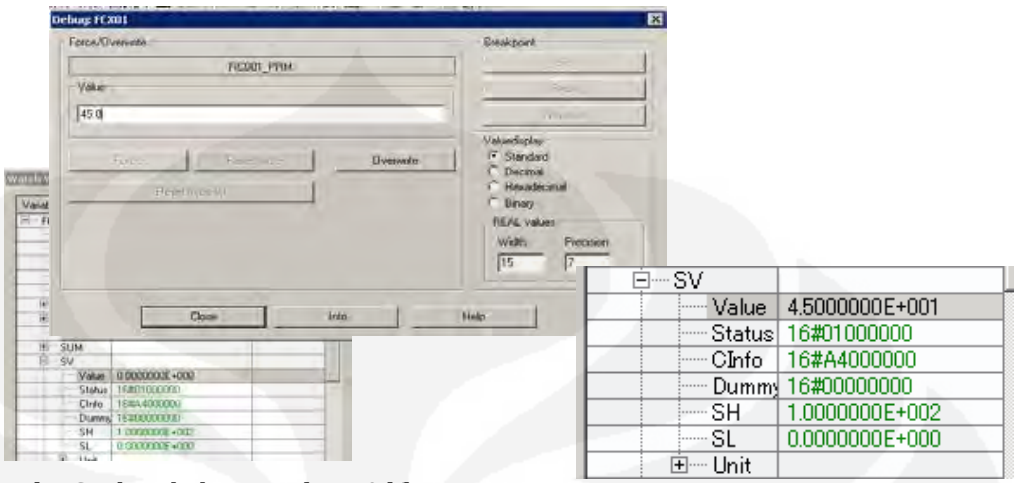
22. Menguji operasi loop kendali

Periksa apakah pengendali “FIC001” beroperasi secara normal oleh perubahan nilai akses parameter loop kendali.

24.1. Mengubah nilai SV

(1) Klik dua kali [FIC001_PRM]-[SV]-[Value] dalam *watch window*, jendela “Debug: FCX01” ditampilkan (gambar 3.61).

(2) Masukkan “45.0 untuk [Value] dan klik tombol [Overwrite].



Nilai SV berubah menjadi 45.0 l/h

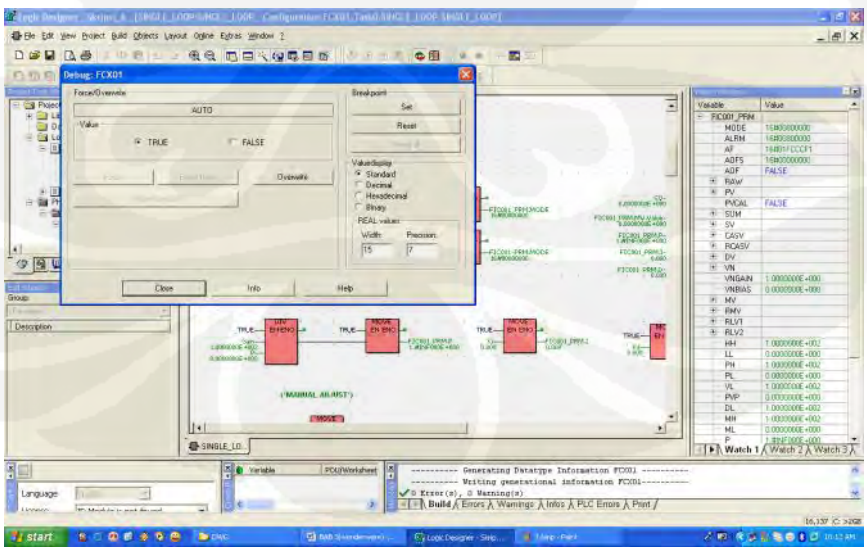
Gambar 3.61. Mengubah nilai *set value*

24.2. Ubah mode pengendali

Sejauh ini nilai SV pengendali “FIC001”telah diubah tetapi nilai lain tidak berubah karena mode block dengan nilai “MAN” (16#00800000), dan harus di *switch* ke mode “AUT” untuk mengecek apakah pengendali beroperasi secara normal.

(1) Klik dua kali [AUTO] item dalam *watch window*, jendela “Debug: FCX01” ditampilkan (gambar 3.62).

(2) Masukkan “TRUE” untuk [Value] dan klik tombol [Overwrite].



Gambar 3.62. Mengubah mode pengendali ke *auto*

Mode blok pengendali “FIC001” telah diubah ke “AUT”, pastikan nilai MV berubah dan nilai PV itu mengikuti *software wiring*.

BAB 4
ANALISA SISTEM KENDALI

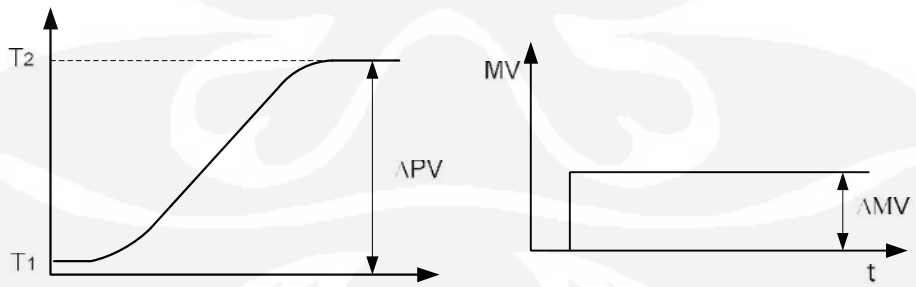
1.11. TUNING PARAMETER PENGENDALI PID.

Untuk mendapatkan aksi kendali sesuai dengan kriteria desain harus melakukan set parameter pengendali PID. Penentuan parameter pengendali PID Kp, Ti, Td dapat diperoleh menggunakan teknik desain sistem kendali klasik. Pada tahun 1940-an, ketika peralatan-peralatan baru saja dikembangkan, *Ziegler* dan *Nichols* (1942) menemukan dua metode empiris untuk mendapatkan parameter-parameter pengendali. Ada dua metode yaitu metode reaksi proses dan metode Osilasi.

4.2. METODA REAKSI PROSES

Ini berdasarkan pada anggapan bahwa respon step pada loop terbuka pada beberapa sistem kendali proses memiliki bentuk S, disebut kurva reaksi proses, seperti gambar 4.1. Kurva reaksi proses terdiri dari bagian tunda waktu td (disebut juga *transportation lag*) dan bagian reaksi sistem orde pertama dengan lengkung tangen maksimal R seperti terlihat pada gambar 4.1.

Dengan cara melakukan perubahan langsung pada output pengendali (MV) dan akan mengakibatkan perubahan pada PV, untuk lebih jelasnya bisa dilihat pada gambar 4.1 berikut ini:



Gambar 4.1.Kurva perubahan PV akibat pengubahan MV

Dengan melakukan perubahan pada MV dapat derekam respon dari proses sehingga dapat ditentukan parameter proses yakni K, τ dan td. Model fungsi alih yang di cari adalah sebagai berikut:

$$G(s) = \frac{K}{\tau s + 1} e^{-tds} \quad (28)$$

Dimana:

K : Gain statis proses

τ : Konstanta waktu proses

td : Tunda waktu (*Dead time* proses)

$$K = \frac{\Delta PV}{\Delta MV} \quad (29)$$

Dari percobaan didapatkan K sebagai berikut:

$$K = \frac{40}{40} = 1 \quad (30)$$

Setelah didapatkan parameter diatas selanjutnya bisa ditentukan parameter pengendali dengan tabel 4.1 berikut:

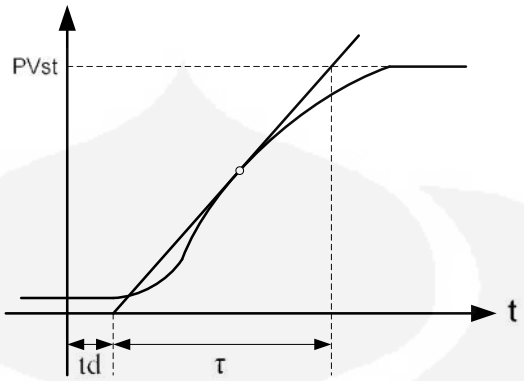
Tabel 4.1. *Ziegler-Nichols* PID parameter menggunakan metode reaksi proses

Tipe Kontroller	P	PI	PID
Kp	$\frac{1}{K}(\frac{td}{\tau})^{-1}$	$\frac{0.9}{K}(\frac{td}{\tau})^{-1}$	$\frac{1.2}{K}(\frac{td}{\tau})^{-1}$
Ti	-	3.33td	2td
Td	-	-	0.5td

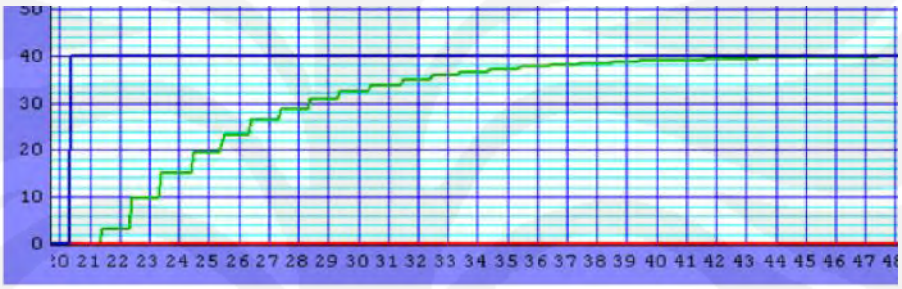
Untuk membaca hasil dari respon PV yang terekam seperti terlihat pada gambar 4.2 ada beberapa metode antara lain sebagai berikut.

4.2.1. Metode Tangen

- Tarik garis singgung yang paling baik
- Td dibaca dari titik potong garis singgung tersebut dengan garis awal
- τ dibaca dari titik titik potong garis singgung dengan garis PV *steady state* (PVst)



(a)



(b)

Gambar 4.2. (a) kurva garis singgung terhadap PV metode tangen,
(b) Kurva karakteristik plant metode tangen.

Dari percobaan sesuai dengan gambar 4.2 (b) didapatkan:

td = 1 detik

τ = 6.4 detik

$$G(s) = \frac{K}{\tau s + 1} e^{-tds} = \frac{1}{6.4s + 1} e^{-s} = \frac{e^{-s}}{6.4s + 1} \qquad (31)$$

a. Jika menggunakan pengendali P

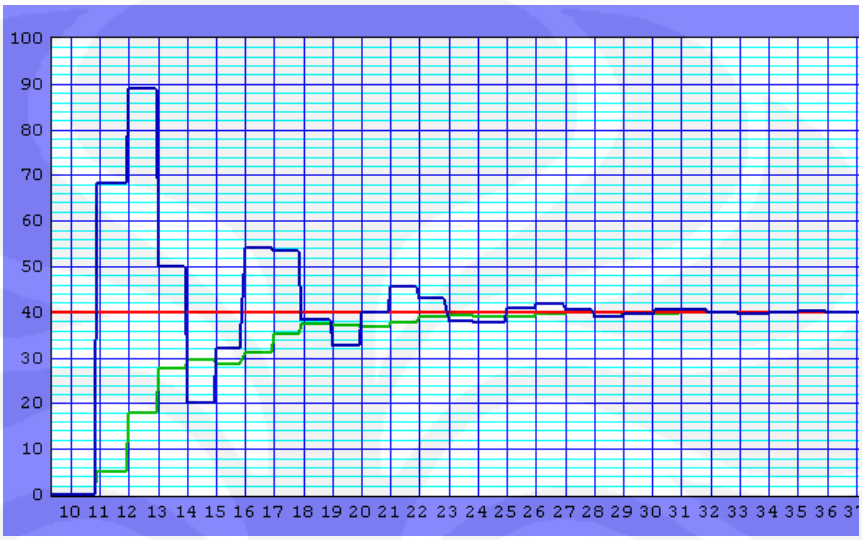
$$Kp = \frac{1}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1}{1}\left(\frac{1}{6.4}\right)^{-1} = 6.4 \qquad (32)$$

b. Jika menggunakan pengendali PI

$$Kp = \frac{0.9}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{0.9}{1}\left(\frac{1}{6.4}\right)^{-1} = 5.76 \qquad (33)$$

$$Ti = 3.33 \times td = 3.33 \times 1 = 3.33 \qquad (34)$$

Gambar 4.3 berikut ini adalah kurva respon sistem dengan pengendali PI metode tangen.



Gambar 4.3. Respon sistem pengendali PI metode tangen.

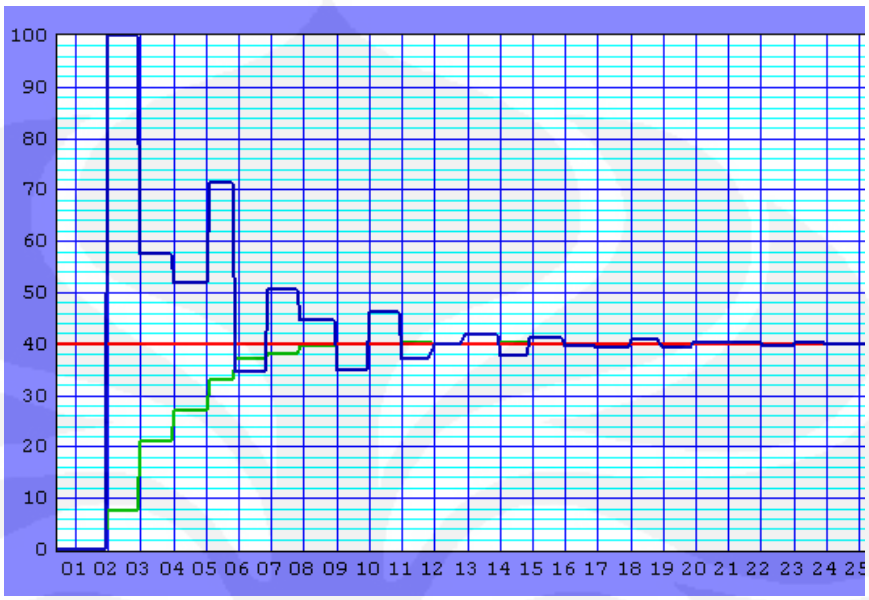
c. Jika menggunakan pengendali PID

$$Kp = \frac{1.2}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1.2}{1}\left(\frac{1}{6.4}\right)^{-1} = 7.68 \qquad (35)$$

$$Ti = 2 \times td = 2 \times 1 = 2 \qquad (36)$$

$$Td = 0.5 \times td = 0.5 \times 1 = 0.5 \qquad (37)$$

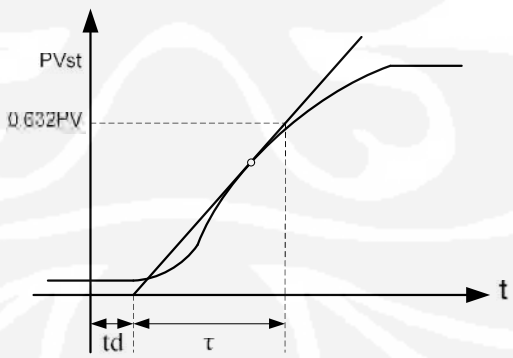
Gambar 4.4 berikut ini adalah kurva respon sistem dengan pengendali PID metode tangen.



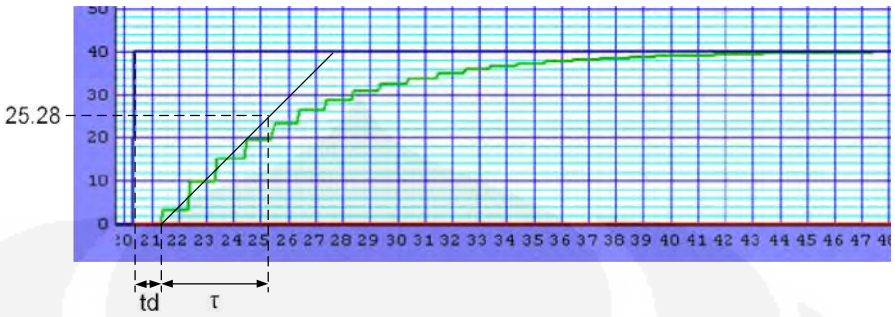
Gambar 4.4. Respon sistem pengendali PID metode tangen

4.2.2. Metode perbaikan tangen (*Modified tangent Method*)

- Untuk mendapatkan td digunakan garis singgung seperti metode tangen
- Nilai τ diambil dari titik yang mencapai 63.2% dari harga PV *steady state*



(a)



(b)

Gambar 4.5. (a) Kurva garis singgung terhadap PV metode perbaikan tangen (b) Kurva karakteristik plant metode perbaikan tangen

Dari percobaan sesuai dengan gambar 4.5 (b) didapatkan:

td = 1 detik

τ = 4 detik

$$G(s) = \frac{K}{\tau s + 1} e^{-tds} = \frac{1}{4s + 1} e^{-s} = \frac{e^{-s}}{4s + 1} \qquad (38)$$

a. Jika menggunakan pengendali P

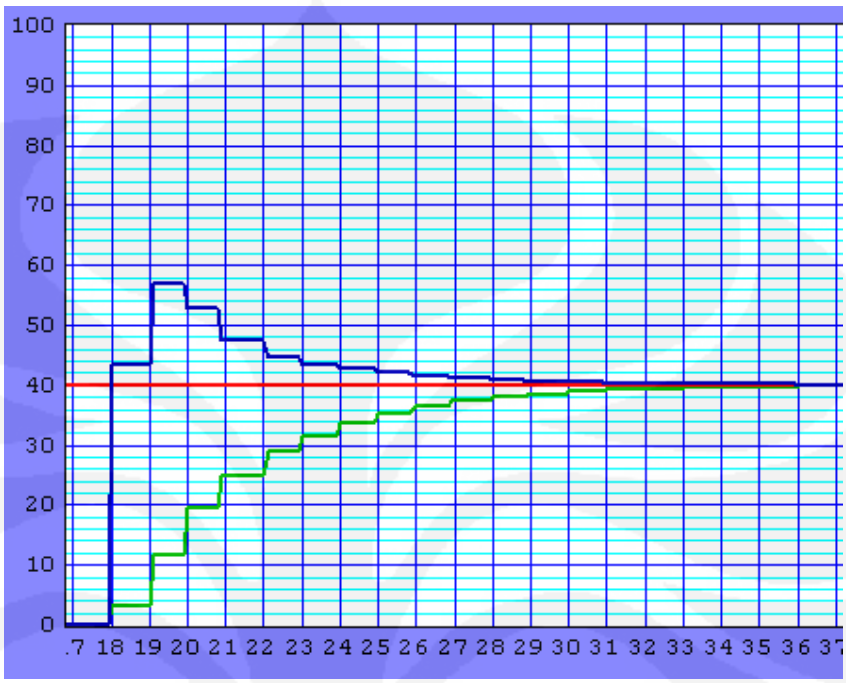
$$Kp = \frac{1}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1}{1}\left(\frac{1}{4}\right)^{-1} = 4 \qquad (39)$$

b. Jika menggunakan pengendali PI

$$Kp = \frac{0.9}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{0.9}{1}\left(\frac{1}{4}\right)^{-1} = 3.6 \qquad (40)$$

$$Ti = 3.33 \times td = 3.33 \times 1 = 3.33 \qquad (41)$$

Gambar 4.6 berikut ini adalah kurva respon sistem dengan pengendali PI metode perbaikan tangen.



Gambar 4.6. Respon sistem pengendali PI metode perbaikan tangen

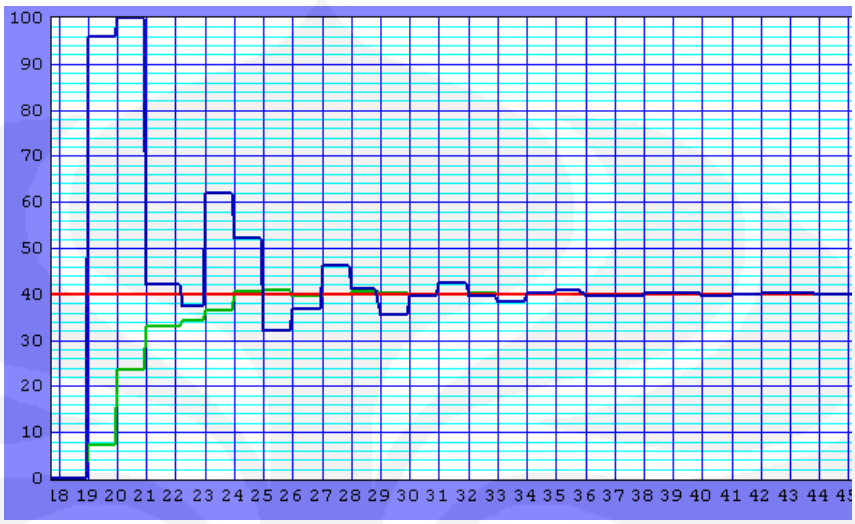
c. Jika menggunakan pengendali PID

$$Kp = \frac{1.2}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1.2}{1}\left(\frac{1}{4}\right)^{-1} = 4.8 \qquad (42)$$

$$Ti = 2 \times td = 2 \times 1 = 2 \qquad (43)$$

$$Td = 0.5 \times td = 0.5 \times 1 = 0.5 \qquad (44)$$

Gambar 4.7 berikut ini adalah kurva respon sistem dengan pengendali PID metode perbaikan tangen.



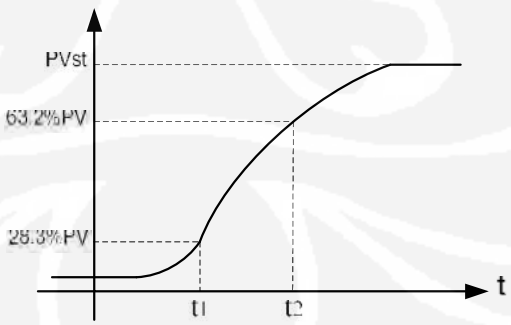
Gambar 4.7. Respon sistem pengendali PID metode perbaikan tangen

4.2.3. Metode Smith

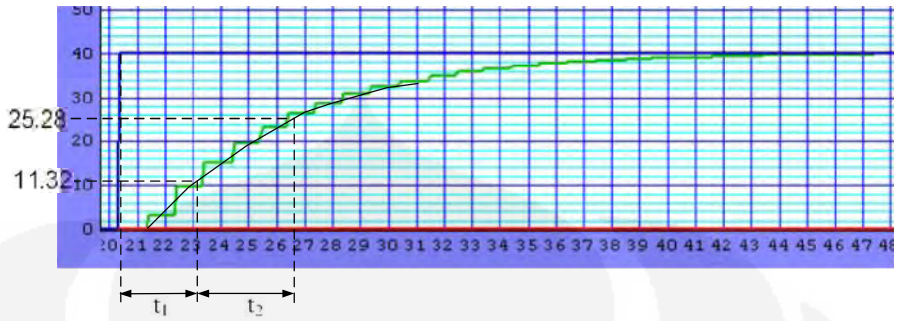
- Catat nilai t_1 dan t_2
- t_1 dibaca dari titik 28.3% PVst
- t_2 dibaca dari titik 63.2% PVst
- Tentukan td dan τ sebagai berikut

$$td = t_2 - \tau \qquad (45)$$

$$\tau = \frac{3}{2}(t_2 - t_1) \qquad (46)$$



(a)



(b)

Gambar 4.8. Kurva garis singgung terhadap PV metode Smith (b) Kurva karakteristik plant metode Smith

Dari percobaan sesuai dengan gambar 4.8 (b) didapatkan:

$t_1 = 2.8$ detik, $t_2 = 3.6$ detik (47)

$$\tau = \frac{3}{2}(t_2 - t_1) = \frac{3}{2}(3.6 - 2.8) = 0.8 \text{ detik} \quad (48)$$

$$td = t_2 - \tau = 3.6 - 0.8 = 2.8 \text{ detik} \quad (49)$$

$$G(s) = \frac{K}{\tau s + 1} e^{-tds} = \frac{1}{0.8s + 1} e^{-2.8s} = \frac{e^{-2.8s}}{0.8s + 1} \quad (50)$$

a. Jika menggunakan pengendali P

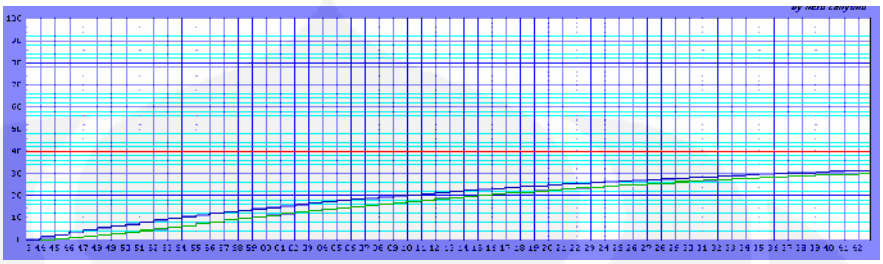
$$Kp = \frac{1}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1}{1}\left(\frac{2.8}{0.8}\right)^{-1} = 0.3 \quad (51)$$

b. Jika menggunakan pengendali PI

$$Kp = \frac{0.9}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{0.9}{1}\left(\frac{2.8}{0.8}\right)^{-1} = 0.27 \quad (52)$$

$$Ti = 3.33 \times td = 3.33 \times 2.8 = 9.3 \quad (53)$$

Gambar 4.9 berikut ini adalah kurva respon sistem dengan pengendali PI metode Smith.



Gambar 4.9. Respon sistem pengendali PI metode Smith

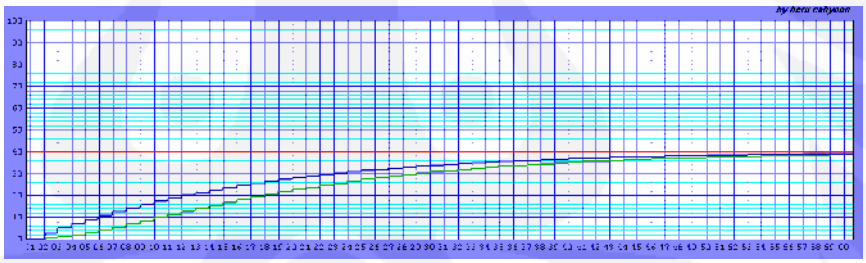
c. Jika menggunakan pengendali PID

$$Kp = \frac{1.2}{K}\left(\frac{td}{\tau}\right)^{-1} = \frac{1.2}{1}\left(\frac{2.8}{0.8}\right)^{-1} = 0.36 \qquad (54)$$

$$Ti = 2 \times td = 2 \times 2.8 = 5.6 \qquad (55)$$

$$Td = 0.5 \times td = 0.5 \times 2.8 = 1.4 \qquad (56)$$

Gambar 4.10 berikut ini adalah kurva respon sistem dengan pengendali PID metode Smith.



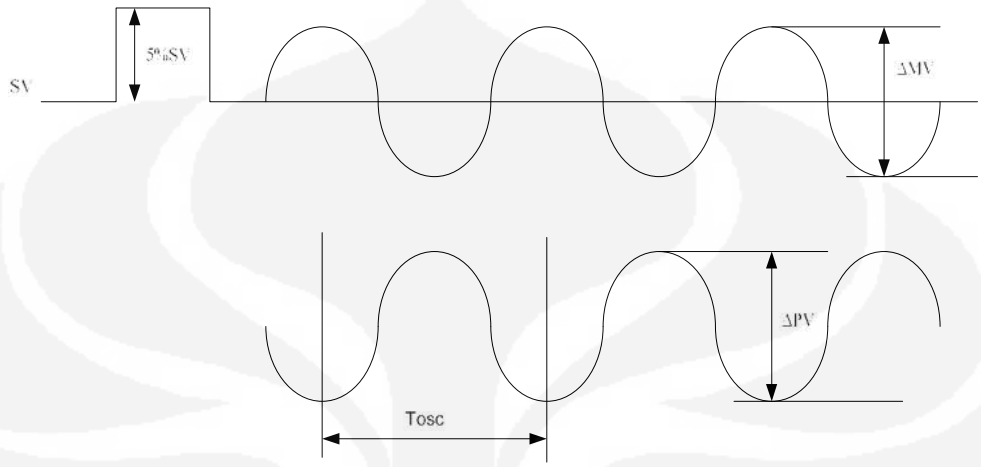
Gambar 4.10. Respon sistem pengendali PID metode Smith

Yang perlu dicatat adalah bahwa metode reaksi proses tidak dapat digunakan jika respon step loop terbuka memiliki *overshoot*, atau berisi *integrator*(s) saja.

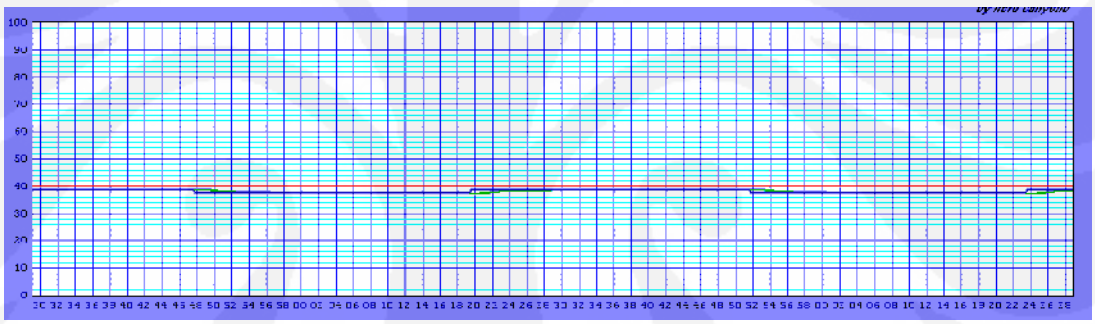
4.3. METODE OSILASI (CONTINUOUS CYCLING)

Metode osilasi adalah teknik untuk loop tertutup, menggunakan kendali proportional saja, gain pengendali Kp dinaikkan (Ti dibuat tak hingga dan Td dibuat 0) sampai didapatkan respon sistem output osilasi pada amplitude konstan,

seperti sistem orde 2 dengan tidak ada *damping ratio*. Kondisi ini merujuk sebagai stabilitas marginal.



(a)



(b)

Gambar 4.11. (a) Respon sistem saat tuning, (b) Respon sistem pada SCADA saat tuning

$$Grc = \frac{\Delta PV}{\Delta MV} \qquad (57)$$

Tabel 4.2. *Ziegler Nichols* PID parameter menggunakan metode osilasi.

Tipe Kontroller	P	PI	PID
Kp	0.5Grc	0.45Grc	0.6Grc
Ti	~	$\frac{Tosc}{1.2}$	$\frac{Tosc}{2}$
Td	0	0	$\frac{Tosc}{8}$

Dari pembahasan diatas dilakukan tuning parameter pengendali PID dengan tahap-tahap sebagai berikut:

1. Tentukan parameter Kp dari nilai yang terkecil sampai nilai yang mendapatkan respon pengendali osilasi dan masukkan nilai Ti tak hingga serta Td 0. Dari hasil percobaan di dapatkan nilai Kp adalah 1.05 dan respon sistem osilasi
2. Setelah didapatkan nilai Kp dengan output osilasi, gunakan tabel 4.2 untuk menentukan parameter pengendali.

   $\Delta MV = 18,\ \Delta PV = 18,\ Tosc = 6.4\,\text{s}$

   $$Grc = \frac{\Delta PV}{\Delta MV} = \frac{18}{18} = 1 \qquad (58)$$

   Jadi dengan menggunakan tabel 4.2 didapatkan parameter pengendali sebagai berikut :

   a. Jika menggunakan pengendali P

      $$Kp = 0.5 \times Grc = 0.5 \times 1 = 0.5 \qquad (59)$$

      $Ti = \infty$

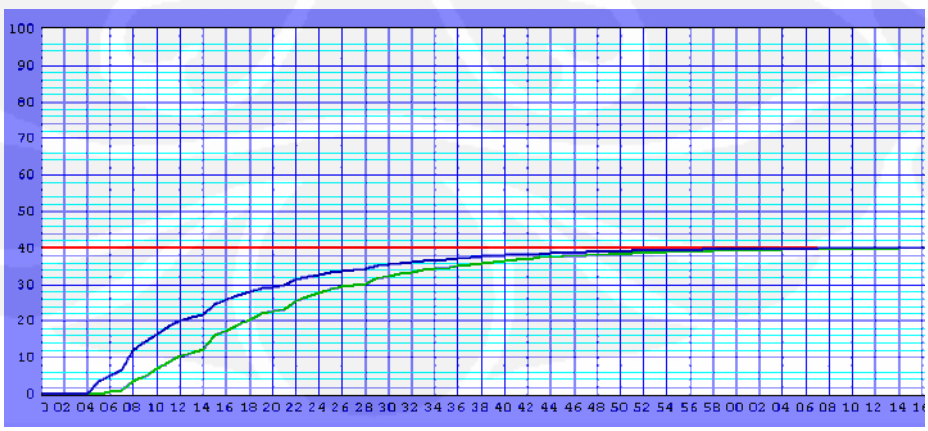
      $Td = 0$

   b. Jika menggunakan pengendali PI

      $$Kp = 0.45 \times Grc = 0.45 \times 1 = 0.45 \qquad (60)$$

      $$Ti = \frac{Tos}{1.2} = \frac{6.4}{1.2} = 5.3 \qquad (61)$$

      Gambar 4.12 berikut ini adalah kurva respon sistem dengan pengendali PI metode osilasi.



Gambar 4.12. Respon sistem pengendali PI metode Osilasi

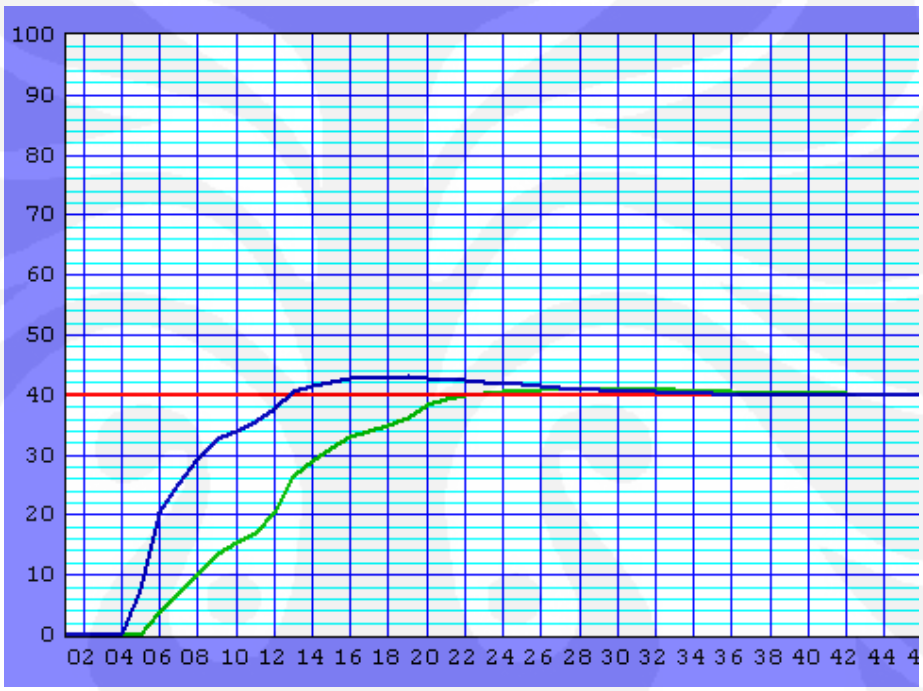
c. Jika menggunakan pengendali PID

$$Kp = 0.6 \times Grc = 0.6 \times 1 = 0.6 \quad (62)$$

$$Ti = \frac{Tos}{2} = \frac{6.4}{2} = 3.2 \quad (63)$$

$$Td = \frac{Tos}{8} = \frac{6.4}{8} = 0.8 \quad (64)$$

Gambar 4.13 berikut ini adalah kurva respon sistem dengan pengendali PID metode osilasi.



Gambar 4.13. Respon sistem pengendali PID metode Osilasi

BAB 5
KESIMPULAN

Dari hasil analisa dan percobaan dalam skripsi ini dapat di simpulkan sebagai berikut :

1. Metode reaksi proses *tuning* parameter pengendali PID oleh Ziegler Nichols dihasilkan nilai parameter pengendali yang menunjukkan *overshoot* yang sangat tinggi,
2. Metode osilasi *tuning* parameter pengendali PID oleh Ziegler Nichols sangat efektif untuk aplikasi dilapangan karena dihasilkan nilai parameter pengendali yang menunjukkan overshoot yang tidak terlalu tinggi.
3. Metode tuning parameter pengendali PID oleh Ziegler Nichols masih diperlukan *fine tuning* untuk mendapatkan nilai parameter pengendali sesuai dengan criteria desain.
4. HMI bisa dikonfigurasi dengan mudah untuk merepresentasikan sistem pengawasan dan kendali aliran dengan cara membuat grafik dan text pada perangkat lunak HMI dan menyambungkannya dengan variabel yang telah dibuat pada PLC untuk ditampilkan pada HMI.
5. Dengan Menggunakan perangkat lunak HMI semua parameter dapat diubah setiap saat tanpa harus mengubah program dalam PLC.

DAFTAR REFERENSI

1. Astrom Karl Johan, ***Control System Design Lecture notes for ME 155A***, University of California Sanra Barbara, Sweden.

2. Bischoff H, D, Terzi E. V. (1997). ***Process Control Engineering,*** Festo Didactic CmbH & Co.

3. Burns Roland S, ***Advanced Control Engineering***, University of Plymouth, United Kingdom.

4. Dunn William C., ***Fundamental of Industrial Instrumentation and Process Contro****l*, McGraw-Hill.

5. Jack Hugh. (2005). ***Automatic Manufacturing System with PLC***, Ed I, Dinastindo, Yogyakarta.

6. Keith Stouffer, Joe Falco, Karen Kent, ***Guide to Supervisory Control and Data Acquisition (SCADA) and Industrial Control System Security***, National Institute of Standard and Technologi, Amerika.

7. McMillan Gregory K. (2001). ***Good Tuning : A Pocket Guide***, ISA-The Instrumentation, System, and Automation Society.

8. Ogata Katsuhito, (1997), ***Modern Control Engineering 3rd***, Prentice Hall, Amerika.

9. Philips Charles L, Harbor Royce D. (1998). ***Sistem Kontrol : Dasar-dasar***, PT. Prenhalindo, Jakarta.

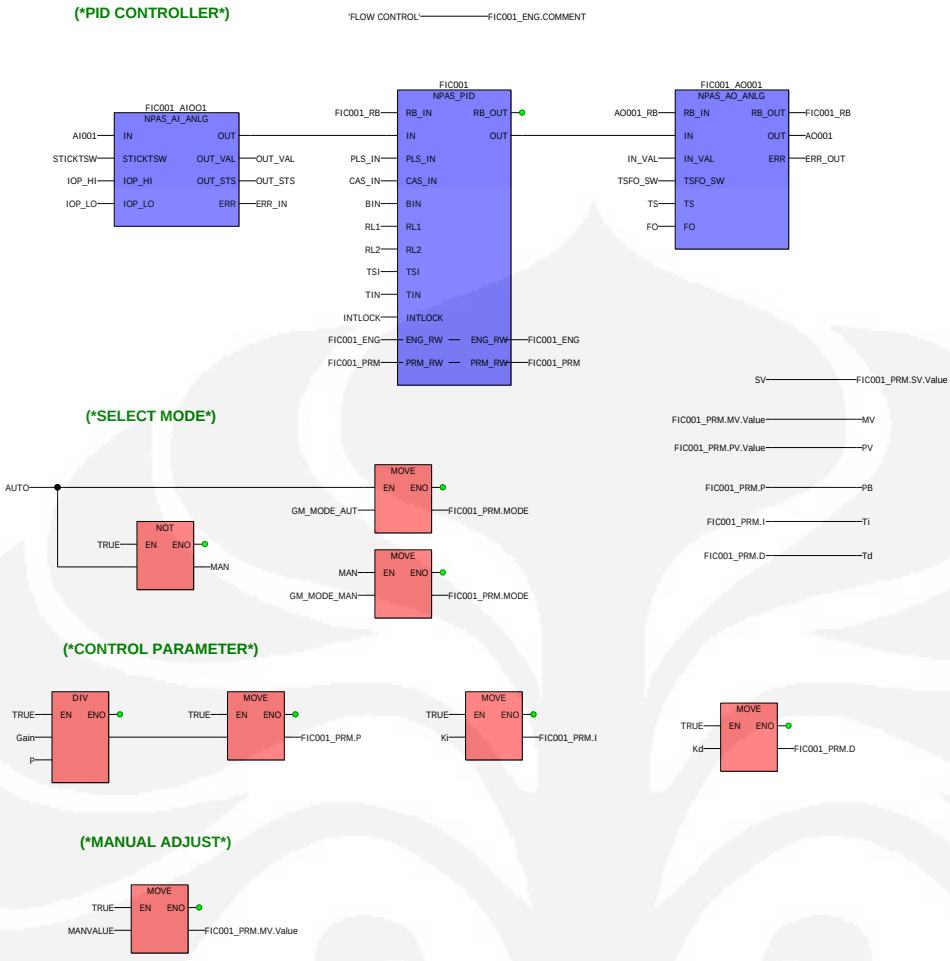
10. ISA-5.3-1983, ***Graphics Symbols for Distributed Control / Shared Display Instrumentation, Logic and Computer System.***

11. William Balton, (2004), ***Programable Logic Controller (PLC)***, Erlangga, Jakarta.

12. YOKOGAWA, ***Manual book VDS dan STARDOM***, YOKOGAWA.

LAMPIRAN 1

PROGRAM FUNGSI BLOK PENGENDALI PID


(*PID CONTROLLER*)
'FLOW CONTROL'
FIC001_ENG.COMMENT
FIC001_AIO01
NPAS_AI_ANLG
AI001
STICKTSW
IOP_HI
IOP_LO
IN
STICKTSW
IOP_HI
IOP_LO
OUT
OUT_VAL
OUT_STS
ERR
OUT_VAL
OUT_STS
ERR_IN
FIC001
NPAS_PID
FIC001_RB
PLS_IN
CAS_IN
BIN
RL1
RL2
TSI
TIN
INTLOCK
FIC001_ENG
FIC001_PRM
RB_IN
IN
PLS_IN
CAS_IN
BIN
RL1
RL2
TSI
TIN
INTLOCK
ENG_RW
PRM_RW
RB_OUT
OUT
ENG_RW
PRM_RW
FIC001_ENG
FIC001_PRM
FIC001_AO001
NPAS_AO_ANLG
AO001_RB
IN_VAL
TSFO_SW
TS
FO
RB_IN
IN
IN_VAL
TSFO_SW
TS
FO
RB_OUT
OUT
ERR
FIC001_RB
AO001
ERR_OUT
SV
FIC001_PRM.SV.Value
FIC001_PRM.MV.Value
MV
FIC001_PRM.PV.Value
PV
FIC001_PRM.P
PB
FIC001_PRM.I
Ti
FIC001_PRM.D
Td
(*SELECT MODE*)
AUTO
TRUE
MAN
GM_MODE_AUT
GM_MODE_MAN
NOT
MOVE
EN
ENO
FIC001_PRM.MODE
(*CONTROL PARAMETER*)
DIV
TRUE
Gain
P
MOVE
FIC001_PRM.P
Ki
FIC001_PRM.I
Kd
FIC001_PRM.D
(*MANUAL ADJUST*)
MANVALUE
FIC001_PRM.MV.Value

LAMPIRAN 2

PROGRAM *SOFTWARE WIRING* SEBAGAI PENGGANTI PLANT

```
(*  *)
SD_WIRE_DTANLG_ANLG_1 (
    OUT1            := AI001,
    IN1             := AO001,
    DIR_F           := DIR_F_1,
    GAIN            := GAIN_1,
    BIAS            := BIAS_1,
    LAGTM           := LAGTM_1,
    SMPLNUM         := SMPLNUM_1,
    SMPLTP          := SMPLTP_1,
    MAN_F           := MAN_F_1,
    MAN_VAL_SRC     := MAN_VAL_SRC_1,
    MAN_STS_SRC     := MAN_STS_SRC_1,
    MAN_VAL_DST     := MAN_VAL_DST_1,
    MAN_STS_DST     := MAN_STS_DST_1,
    RB_OUT1         := AO001_RB
);
AI001           := SD_WIRE_DTANLG_ANLG_1.OUT1;
AO001_RB        := SD_WIRE_DTANLG_ANLG_1.RB_OUT1;
```

Name	Type	Usage	Description	Address	Init	Retain	PDD	OPC
Default								
SD_WIRE_DTANLG_ANLG_1	SD_WIRE_DTANLG_ANLG	VAR						
DIR_F_1	BOOL	VAR			TRUE			
GAIN_1	REAL	VAR			1.0			
BIAS_1	REAL	VAR			0.0			
LAGTM_1	TIME	VAR			t#5s			
SMPLNUM_1	INT	VAR			1			
SMPLTP_1	TIME	VAR			t#1s			
MAN_F_1	BOOL	VAR			FALSE			
MAN_VAL_SRC_1	REAL	VAR						
MAN_STS_SRC_1	WORD	VAR						
MAN_VAL_DST_1	REAL	VAR						
MAN_STS_DST_1	WORD	VAR						
AI001	DTag_I_Anlg	VAR_EXTERNAL						
AO001	DTag_O_Anlg	VAR_EXTERNAL						
AO001_RB	DTag_O_Anlg	VAR_EXTERNAL						