ITS
Institut
Teknologi
Sepuluh Nopember

FINAL PROJECT - RF1483

DESIGN AND IMPLEMENTATION OF pH CONTROL USING NEURAL NETWORK BASED TRACKING CONTROL SYSTEM

DENY TRIYATNO
NRP 2401100072

Advisor Lecturer
Hendra Cordova, ST. MT.

Department of Engineering Physics
Faculty of Industrial Technology
Sepuluh Nopember Institute of Technology
Surabaya 2006

TUGAS AKHIR

RANCANG BANGUN SISTEM PENGENDALIAN pH DENGAN MENGGUNAKAN NEURAL NETWORK BERBASIS TRACKING CONTROL SYSTEM

OLEH:

DENY TRIYATNO
NRP. 2401 100 072

Surabaya, Juli 2006

Mengetahui/Menyetujui
Pembimbing

HENDRA CORDOVA, ST. MT
NIP. 132 125 672

Ketua Jurusan
Teknik Fisika FTI-ITS

DR. Ir. TOTOK SUHARTANTO, DEA
NIP. 131 879 399

RANCANG BANGUN SISTEM PENGENDALIAN pH DENGAN MENGGUNAKAN NEURAL NETWORK BERBASIS TRACKING CONTROL SYSTEM

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Teknik
pada
Bidang Studi Rekayasa Instrumentasi dan Kontrol
Program Studi S-1 Jurusan Teknik Fisika
Fakultas Teknologi Industri
Institut Teknologi Sepuluh Nopember

Oleh:
DENY TRIYATNO
NRP. 2401100072

Disetujui oleh Tim Penguji Tugas Akhir:

1. Hendra Cordova, ST., MT. ..... (Dosen Pembimbing)
2. Gunawan Nugroho, ST., MT. ..... (Ketua Tim Penguji)
3. Ir. Ya'umar, MT. ..... (Staf Penguji)
4. Ir. Sarwono, MM. ..... (Staf Penguji)
5. Fitri Adi Iskandarianto, ST. ..... (Staf Penguji)

SURABAYA
JULI, 2006

RANCANG BANGUN SISTEM PENGENDALIAN pH DENGAN MENGGUNAKAN NEURAL NETWORK BERBASIS TRACKING CONTROL SYSTEM

Nama Mahasiswa : **DENY TRIYATNO**
NRP : **2401100072**
Jurusan : **Teknik Fisika**
Dosen Pembimbing : **HENDRA CORDOVA, ST., MT.**

Abstrak

pH mempunyai pengaruh pada proses-proses industri, misalnya proses pengolahan farmasi, bioteknologi, proses kimia, serta pemrosesan pembuangan limbah buangan. Salah satu cara untuk mengendalikan pH adalah dengan volumetri atau titrasi dengan menggunakan larutan asam dan larutan basa. Dari data proses titrasi didapatkan kurva titrasi yang nonlinier, karena hubungan yang logaritmik antara harga pH dan volume larutan penetral. Di luar range larutan netral (pH = 7) maka gain prosesnya relatif kecil, namun jika didalam range larutan netral maka gain prosesnya sangat besar. Oleh karena itu merupakan salah satu alasan mengapa penggunaan pengendali PID konvensional dapat menimbulkan kegagalan pada suatu sistem pengendaliannya. Pada penelitian ini dirancang sebuah sistem pengendalian pH sengan menggunakan sebuah sistem pengendalian pH dengan menggunakan Neural Network berbasis Tracking Control System. Sistem kontrol yang dikembangkan mampu menghasilkan performansi untuk set point 7 sehinngga didapatkan nilai Rise Time (Tr) = 20 detik, maksimum overshoot (Mp)=33,72%, Setling Time (Ts)=190 detik dan Error steady state (Ess) = 0,8% sedangkan untuk set point 6 didapat nilai perfomansi sistem yakni Tr = 12 detik, Mp = 33,45%, Ts=146 detik dan Ess = 6,5%.

Kata kunci: Neural Network, nonlinier, pH, Tracking Control

DESIGN AND IMPLEMENTATION OF pH CONTROL USING NEURAL NETWORK BASED TRACKING CONTROL SYSTEM

Student's Name : **DENY TRIYATNO**
NRP : **2401100072**
Department : **Engineering Physics**
Advisor Lecturer : **HENDRA CORDOVA, ST., MT.**

Abstrak

pH has an effect for industrial processes, for example in a pharmaceutical process, biotechnology, chemical process and liquid waste disposal processing process. One of many ways to control pH value is using acid and base volumetric or titration. Non linear titration curve is gotten by titration process, because pH value and neutralizing solution are logarithmic function. Out of neutral solution range(pH=7), the process gain is small relatively, but in neutral solution range the process gain is big. Because of that the reason why PID conventional controller can make mistake in a control system. In this research will design a pH control system using Neural Network controller based Tracking Control System. The control system which has been designed is able to perform a system performance for pH = 7 as a set point as written given Rise Time (Tr) = 20 second, maximum overshoot (Mp)=33,72%, Setling Time (Ts)=190 second and Error steady state (Ess) = 0,8% but for pH = 6 as a set point given a system performance that Tr = 12 second, Mp = 33,45%, Ts=146 second and Ess = 6,5%.

Keyword: Neural Network, nonlinier, pH, Tracking Control

KATA PENGANTAR

Segala puji syukur saya panjatkan kehadirat Allah Subhaanahu wa Ta'aala yang telah banyak melimpahkan rahmat, taufiq, hidayah, serta memberi iman dan kesabaran juga shalawat serta salam semoga tetap terlimpah kepada Nabi Besar Muhammad SAW sehingga saya dapat menyelesaikan tugas akhir yang berjudul "**Rancang Bangun Sistem Pengendalian pH Dengan Menggunakan Neural Network Berbasis Tracking Control System**".

Pada kesempatan ini saya ingin mengucapkan terima kasih yang sebesar-besarnya kepada :

1. Yang tercinta keluarga saya, Ibu Karmi dan Bapak Komiran atas kasih sayangnya serta kesabaran dalam mendidik dan ketulusan dalam do'anya, terima kasih buat adikku, Rita, atas dorongan, sayang dan do'anya.
2. Bapak Dr. Ir. Totok Suhartanto DEA sebagai Ketua Juruasan Teknik Fisika ITS.
3. Bapak Dr. Bambang Lelono W., ST. MT., selaku Dosen Wali yang telah memberikan arahan dan nasehat kepada saya selama menjalani masa perkuliahan.
4. Bapak Hendra Cordova ST. MT., selaku Dosen Pembimbing atas kesabarannya dalam membimbing saya dalam mengerjakan Tugas Akhir hingga saya dapat menyelesaikan tepat pada waktunya.
5. Pihak-pihak yang turut membantu dan mendukung penyelesaian tugas akhir ini.

Saya berharap Tugas Akhir ini dapat memberikan kontribusi yang berarti dan dapat menambah wawasan bagi pembaca dan mahasiswa Teknik Teknik Fisika.

Surabaya, Juli 2006
Penulis

DAFTAR ISI

BAB **HALAMAN**

LEMBAR JUDUL i
LEMBAR PENGESAHAN ii
ABSTRAK iv
KATA PENGANTAR vi
DAFTAR ISI vii
DAFTAR GAMBAR xi
DAFTAR TABEL xv
DAFTAR NOTASI xvii
I. PENDAHULUAN 1
1.1 Latar Belakang Permasalahan 1
1.2 Rumusan Permasalahan 4
1.3 Batasan Masalah 4
1.4 Tujuan 5
1.5 Metodologi Penelitian 5
1.6 Sistematika 6
1.7 Manfaat 6
II. TEORI PENUNJANG 7
2.1 Teori Dasar pH larutan 7
2.2 Kurva Titrasi 12
2.3 Sistem Pengukuran dan Sensor pH 15
2.4 Penguat Instrumentasi 27
2.5 ADC (Analog To Digital Converter) 28

2.6 Port Paralel 30
2.7 Jaringan Syaraf Tiruan 33
2.7.1 Konsep Proses Belajar JST 37
2.7.2 Arsitektur Jaringan Syaraf Tiruan 39
2.7.3 Algoritma Belajar Jaringan Syaraf Tiruan Tiga Lapis Umpan Maju 40
2.7.4 Algoritma Pembelajaran dengan *Error Backpropagation* 42
2.7.5 Adaptasi Bobot-bobot pada Lapisan Output 44
2.7.6 Adaptasi Vektor Bobot pada Lapisan Tersembunyi 46
2.8 Jaringan Syaraf Tiruan pada Sistem Pengendalian 49
III. PERANCANGAN PERANGKAT KERAS DAN PERANGKAT LUNAK 53
3.1 Perancangan Perangkat Keras 53
3.1.1 Rangkaian Pengkondisi Sinyal 54
3.1.2 Analog To Digital Converter (ADC) 55
3.1.3 Rangkaian Eksternal Paralel Port 57
3.1.4 Driver Motor 59
3.2 Perancangan Perangkat Lunak 61
3.2.1 Pelatihan JST Backpropagation 61
3.2.2 Sistem Pengendalian 62
IV. PENGUJIAN DAN ANALISA 65
4.1 Pengujian Perangkat Keras 65
4.1.1 Pengukuran Sensor pH 65
4.1.2 Pengukuran dan Pengujian Rangkaian Pengkondisi Sinyal 67
4.1.3 Pengukuran dan Pengujian ADC 69
4.1.4 Pengujian Control Valve 69
4.1.5 Pengujian Rangkaian Eksternal Port Paralel 71
4.2 Uji Open Loop 72
4.3 Parameter JST 73
4.4 Sistem Pengendalian pH dengan Menggunakan Jaringan Syaraf Tiruan 77
V. KESIMPULAN DAN SARAN 79
5.1 Kesimpulan 79
5.2 Saran 79
DAFTAR PUSTAKA
LAMPIRAN A Listing Program JST Training A-1
LAMPIRAN B Listing Program Validasi JST B-1

LAMPIRAN C Listing Program Tes ADC dan JST Controller C-1
LAMPIRAN D Listing Program Unit_IO dan
Unit_Motor_Positioner D-1
LAMPIRAN E Data Training E-1
LAMPIRAN F Data Hasil Training Dan Validasi F-1
LAMPIRAN G Blok Diagram Sistem Pengendalian pH Dan
Fungsi Transfer G-1
LAMPIRAN H Grafik Uji Tracking Set Point H-1
LAMPIRAN I Gambar Komponen Miniplant Sistem
Pengendalian pH I-1
LAMPIRAN J Pembuatan Larutan Asam Dan Larutan Basa J-1

DAFTAR GAMBAR

GAMBAR **HALAMAN**

Gambar 2.1: Kurva Titrasi untuk Asam Kuat dan Basa Kuat (Doherty, 1999) 14
Gambar 2.2 : Grafik hubungan tegangan keluaran elektrode Terhadap variasi temperatur (Yien, 2001) . . . 18
Gambar 2.3 : Elektroda Gelas (Yien, 2001) 21
Gambar 2.4 : Skematik aliran H^+ pada membran elektroda 23
Gambar 2.5 : Elektroda Referensi (Yien, 2001) 24
Gambar 2.6 : Elektroda Kombinasi (Yien, 2001) 25
Gambar 2.7 : Rangkaian Ekuivalen elektroda kombinasi . . 27
Gambar 2.8 : Rangkaian Penguat Sederhana (Curtis, 1997) 27
Gambar 2.9 : ADC Succesive Approximation (Datasheet ADC0804 National Semiconductor) 29
Gambar 2.10 : Pin-pin pada konektor D-tipe 25 (www.senet.com.au/~cpeacock) 31
Gambar 2.11: Rangkaian Eksternal dengan IC 74LS151 (Datasheet SN54/74LS151 Motorola) 33
Gambar 2.12 : Model Syaraf Tiruan (Freeman, 1992) 34
Gambar 2.13 : Fungsi Aktivasi (Kusumadewi, 2003) 37
Gambar 2.14 : Belajar dengan Pengawasan 38
Gambar 2.15 : Belajar tanpa Pengawasan 38

Gambar 2.16 : Arsitektur JST FeedForward Backpropagation (Rumelhart et al., 1985) 41
Gambar 2.17 : Tiga Lapis Arsitektur Jaringan Syaraf Metode belajar Backpropagation 42
Gambar 2.18 : Jaringan Syaraf Tiruan digunakan sebagai Identifikasi dan pengendali 50
Gambar 2.19 : Diagram Blok Aplikasi Neural Network Tracking Controller (Tai, Wang, Ashenayi, 1992) 51
Gambar 2.20 : Konsep pembelajaran dengan pendekatan Fungsi *u=f(error)* 51
Gambar 3.1 : Skema perangkat keras Miniplant Sistem Pengendalian pH 53
Gambar 3.2 : Miniplant Sistem Pengendalian pH 54
Gambar 3.3 : Rangkaian Pengkondisi Siyal 55
Gambar 3.4 : Rangkaian ADC (National Semiconductor, 1999) 56
Gambar 3.5 : Rangkaian Eksternal Paralel Port 58
Gambar 3.6 : Rangkaian Driver Motor 60
Gambar 3.7 : Control Valve Miniplant Pengendaian pH ... 60
Gambar 3.8 : Blok Diagram Sistem Pengendalian pH..... 61
Gambar 3.9 : Flowchart proses pelatihan (Laurence, 1994) 62
Gambar 3.10 : Tampilan Perangkat Lunak Sistem Pengendalian pH 63
Gambar 3.11 : Flowchart Program Kontroller 63
Gambar 4.1 : Grafik Output Sensor pH 66
Gambar 4.2 : Grafik Kelinieritasan Pengkondisi Sinyal ... 68
Gambar 4.3 : Grafik Respon Laju Aliran Terhadap Bukaan Valve..... 70
Gambar 4.4 : Grafik Uji Open Loop 72
Gambar 4.5 : Grafik Hasil Pembelajaran JST 75
Gambar 4.6 : Grafik Data Target untuk Validasi Model JST 75
Gambar 4.7 : Grafik Output model hasil validasi model JST 75
Gambar 4.8 : Grafik Respon Sistem secara online untuk Set point pH = 7 77

Gambar 4.9 : Grafik sinyal kontrol secara online untuk Set point pH = 7. 77
Gambar 4.10 : Grafik Respon Sistem secara online untuk Set point pH = 6 78
Gambar 4.11 : Grafik sinyal kontrol secara online untuk Set point pH = 6 78

DAFTAR TABEL

TABEL **HALAMAN**

Tabel 2.1: Hubungan ion hidrogen dan ion hidroksida terhadap Nilai pH (www.sensorex.com) 14
Tabel 2.2: Error pada Elektroda oleh pengaruh temperatur (www. Sensorex.com) 19
Tabel 2.3: Alamat Port Paralel (www.senet.com.au) 31
Tabel 2.4: Fungsi tiap pin konektor D-Type 25 Paralel Port (www.senet.com.au) 32
Tabel 2.5: Algoritma belajar, arsitektur dan aplikasinya . . . 40
Tabel 2.6: Tabel Keterangan untuk Gambar 2.17 43
Tabel 3.1: Tabel Kebenaran Multiplexer 74LS151 59
Tabel 4.1: Hasil Pengukuran Output Sensor pH 64
Tabel 4.2: Hasil pengujian dan kalibrasi pengkondisi sinyal 67
Tabel 4.3: Hasil pengukuran bukaan valve terhadap flow . . 70
Tabel 4.4: Hasil pengujian rangkaian eksternal paralel port 71
Tabel 4.5: Nilai bobot dan bias hasil pembelajaran JST . . . 76

DAFTAR NOTASI

Ex : konstanta tergantung elektroda referensi
R : konstanta
TK : temperatur absolut (Kelvin)
n : muatan ion
F : konstanta
ai : aktivitas ion
R_θ : resistansi pada suhu θ
$R_{\theta 1}$: resistansi pada suhu referensi (289^0K)
β : konstanta termistor
θ : suhu (^{0}Kelvin)
Eg : Beda potensial total elektroda gelas
Eo : Beda potensial pada elektroda gelas yang besarnya konstan
T : Temperatur (^{o}K) , dimana pada $0^oC = 273\ ^oK$

BAB I
PENDAHULUAN

1.1 Latar Belakang Permasalahan

pH adalah besaran yang secara kualitas digunakan untuk menentukan konsentrasi ion hidrogen pada suatu larutan. Besarnya pH mempunyai pengaruh pada proses-proses industri, misalnya proses pengolahan farmasi, bioteknologi, proses kimia, serta pemrosesan pembuangan limbah buangan. Salah satu cara untuk mengendalikan pH adalah dengan volumetri atau titrasi menggunakan larutan asam basa. pH biasa dinyatakan sebagai negatif logaritma konsentrasi ion hidrogen dan dapat digambarkan dalam kurva titrasi berbentuk 'S'. Karena nonlinieritas dari logaritma tersebut maka besar penguatannya adalah 10 kali untuk setiap kenaikan atau penurunan satu unit nilai pH. Titik dimana besar antara konsentrasi larutan asam dan larutan basa adalah sama, maka titik tersebut dinamakan titik ekivalen (pH=7). Sehingga untuk larutan asam kuat dan basa kuat pada titik ekivalen gain prosesnya sangat tinggi, artinya bahwa dengan sedikit penambahan atau pengurangan larutan titrasi pada nilai pH sekitar 7 maka pengaruh terhadap perubahan harga pH sangat besar. Sehingga merupakan salah satu alasan mengapa penggunaan pengendali PID konvensional dapat menimbulkan kegagalan pada suatu sistem pengendaliannya.

Oleh karena itu dibutuhkan pengendali yang dapat belajar (*learn*) bagaimana mengatasi hubungan yang logaritmik antara harga pH dengan volume larutan titrasinya (Sean, 1999). Salah satu tipe pengendali yang dapat belajar adalah pengendali berbasis jaringan syaraf tiruan (*Artifical Neural Network*) disingkat JST. Selain itu alasan penting penggunaan JST sebagai pengendali adalah bahwa JST dapat digunakan untuk mengatasi model yang sangat kompleks, nonlinier serta fungsi-fungsi yang

multi-dimensional. Khususnya kemampuan dalam penentuan arsitektur jaringan syaraf, misalnya jaringan dengan banyak lapisan (*multi-layered perceptron*) yang telah dibuktikan untuk pemetaan variabel-variabel proses yang nonlinier (Cybenko, 1989; Hornick et al., 1989).

Konsep belajar pada JST mencontoh kemampuan belajar otak manusia, sehingga penurunan matematis (model) JST mengacu pada mekanisme serta susunan otak. Sebuah model otak menghubungkan beberapa model syaraf linier atau nonlinier yang diproses dalam bentuk distribusi paralel, sehingga komputasinya lebih cepat. Selain itu JST mempunyai kemampuan seperti pembelajaran dan pengorganisasian diri, sehingga JST dapat beradaptasi terhadap perubahan data, belajar karakteristik sinyal masukan dan memetakan terhadap keluaran. Jika terdapat presentasi input yang baru setelah pembelajaran dilakukan, JST akan melakukan generalisasi untuk memberi output yang sesuai. Hal ini yang mendasari dipakainya pengendali berbasis JST. Untuk merancang pengendali ini terlebih dahulu dibentuk model jaringan berdasarkan identifikasi proses yang dikendalikan. Selanjutnya berdasarkan model jaringan ini (*neural network model*) yang mewakili sistem yang sebenarnya, dirancang pengendali JST (*neurocontroller*). Jadi terdapat dua langkah yang diperlukan untuk membentuk model dan pengendali, dimana masing-masing mekanisme biasanya memakai pembelajaran rambat mundur kesalahan (*error back-propagation*). Bila rentang masukan-keluaran sistem besar dan nonlinier, maka pembelajaran akan membutuhkan waktu komputasi yang lama. Selain itu bila karakteristik dinamik berubah dari yang telah dipelajarinya (misal tidak ada di rentang input-output), maka harus dilakukan pembelajaran ulang untuk kondisi yang berubah tersebut.

Sejumlah peneliti telah memperkenalkan skema baru yang tidak memerlukan pembelajaran sistem dinamika balik sistem (*inverse dynamic system*) atau identifikasi. Skema tersebut berdasarkan bahwa tujun utama semua teori maupun aplikasi

perancangan sistem pengendalian adalah menentukan aksi kendali yang sesuai (*proper*) agar performansi sistem seperti yang diinginkan, artinya jika pengetahuan aksi kontrol cukup, maka secara eksplisit tidak diperlukan pembelajaran dinamika balik sistem oleh JST. Pembelajaran dilakukan hanya menggunakan dua kumpulan data yaitu *error e* dan *aksi kendali u* yang sesuai. Jadi waktu yang diperlukan untuk belajar akan lebih singkat dan pengendali akan lebih kokoh terhadap perubahan parameter maupun sinyal referensi sisem yang dikendalikan (Tai, Wang, Ashenayi, 1992).

Beberapa peneliti telah melakukan sebuah penelitian tentang sistem pengendalian pH, yakni rancang bagun sistem pengendalian proses netralisasi pH berbasis PC (Rasiawan, 2002), namun masih banyak terdapat beberapa kelemahan, diantaranya adalah bahwa mode kontrol yang digunakan hanya PID konvensional sehingga tidak mampu mengatasi karakteristik sistem yang nonlinier. Metode lain yang telah dilakukan adalah perancangan kontroller PID self tuning berbasis jaringan syaraf tiruan pada proses netralisasi pH (Andry F., 2004). Kontroller yang digunakan pada penelitian tersebut telah mampu menunjukkan performa yang lebih baik untuk *risetime*, *time settling* dan *maksimum overshoot* jika dibandingkan dengan pengendali yang hanya terdiri dari PID saja. Namun pada penelitian tersebut hanya diimplementasikan secara *offline* sehingga belum dapat diketahui kehandalan kontroller jika diimplementasikan secara *online*.

Beranjak dari permasalahan seperti tersebut diatas, penelitian ini mencoba untuk merancang dan mengimplentasikan sebuah sistem pengendalian proses yang nonlinier sehingga diharapkan mampu mengatasi karateristik dari proses di *miniplant* sistem pengendalian pH berdasar *artificial neural network* berbasis *Tracking Control System* dengan skema baru yang diperkenalkan oleh *Tai* (Tai, Wang, Ashenayi, 1992). Berdasar hasil penelitian yang dilakukan oleh *Tai* bahwa istilah *Neural*

Network berbasis *Tracking Control System* dapat didefinisikan sebagai suatu jaringan syaraf tiruan yang berfungsi sebagai *classifier* antara set point dan sinyal kontrol sehingga diharapkan sinyal keluaran dari plant mampu dikendalikan agar dapat mencapai nilai sinyal masukan referensi (*set-point*) yang diberikan.

1.2 Rumusan Permasalahan

Masalah yang diangkat dalam penelitian ini adalah bagaimana mengendalikan pH dan mengimplementasikan metode *Neural Network* yang berbasis *Tracking Control System* pada *miniplant* sistem pengendalian pH.

1.3 Batasan Masalah

Batasan masalah yang digunakan dalam pelaksanaan tugas akhir ini adalah sebagai berikut:

- Metode kontrol yang digunakan adalah *Neural Network* dengan arsitektur umpan maju dengan kaidah belajar propagasi balik berbasis *Tracking Control System*.
- Level tangki titrasi dijaga konstan.
- Pengaruh suhu dikompensasi oleh rangkaian kompensator suhu.
- Variabel yang dikendalikan adalah nilai pH larutan pada tangki reaksi, sedangkan yang dimanipulasi adalah laju aliran larutan basa.
- Plant yang dikendalikan merupakan proses dengan satu masukan dan satu keluaran, yakni nilai pH yang terukur pada tangki reaksi sebagai masukan dan besarnya laju aliran larutan basa sebagai keluaran.

- Variabel Proses yang dikendalikan dibatasi antara range pH=6 sampai dengan pH=8.

1.4 Tujuan

Tujuan dalam melakukan penelitian ini adalah untuk merancang dan membuat sistem pengendalian pH dengan metode *neural network* berbasis *tracking control system*.

1.5 Metodologi Penelitian

Metodologi yang direncanakan untuk pelaksanaan pengerjaan penelitian ini adalah sebagai berikut:

- Studi literatur mengenai *Neural Network* (Algoritma pelatihan propagasi balik), Hardware (Akuisisi data yang meliputi Rangkaian Pengkondisi Sinyal ADC, DAC, Control valve dan plant itu sendiri), interfacing komputer dengan plant, dan pemrograman menggunakan Delphi 7.0.
- Mempelajari proses pengendalian pH yang ada di-plant (control valve, transmitter, tanki dll).
- Perancangan miniplant sistem pengendalian pH baik *hardware* maupun *software* berdasarkan hasil studi literatur yang telah dilakukan
- Mengaplikasikan perancangan hardware dan software ke *miniplant* melalui hubungan antarmuka menggunakan komputer dengan port parallel.
- Melakukan pengujian dan analisa sistem yang telah dirancang dan diimplementasikan baik *software* maupun *hardware*-nya.
- Penyusunan laporan dan buku tugas akhir.

1.6 Sistematika

Pada penelitian ini dibuat laporan dengan sistematika laporan sesuai petunjuk yang dipakai untuk penulisan laporan tugas akhir yang terdiri dari : Bab I pendahuluan yang berisi latar belakang, permasalahan, tujuan, batasan masalah, metodologi penelitian, sistematika laporan dan manfaat. Pada bab II teori penunjang berisi materi yang menunjang keberhasilan dalam menyelesaikan permasalahan dan menunjang tercapainya tujuan yang diharapkan pada penelitian ini. Bab III adalah desain rancangan beserta pertimbangan yang dipergunakan dalam rancangan sistem. Bab IV Pengujian dan analisa data hasil pengujian. Kesimpulan dan saran yang merupakan bab V, berisi keseluruhan hasil akhir penelitian dan memberikan saran atau masukan yang berhubungan dengan penelitian ini.

1.7 Manfaat

Manfaat yang diperoleh dari hasil penelitian ini adalah untuk mengetahui kharakteristik pengendali jika diimplementasikan pada sistem pengendalian proses yang nonlinier yakni pengendalian pH.

BAB II
TEORI PENUNJANG

Bab ini akan menjelaskan tentang reaksi larutan asam kuat dan larutan basa kuat, ketetapan kesetimbangan, yang berkaitan dengan penetralan pH, *Neural Network*, yang meliputi algoritma belajar, *Neural Network feedforward*, dan metode belajar *backpropagation* serta perhitungan *error* pada *Neural Network*.

2.1 Teori Dasar pH Larutan

Proses penetralan harga pH memerlukan pengetahuan beberapa aspek kimia pada proses tersebut. Dua yang terpenting untuk diketahui adalah teori tingkah laku terjadinya reaksi dan perubahan harga pH terhadap beberapa masukan pelarut untuk menjaga harga pH yang ingin diatur. Atau yang lebih khusus adalah penurunan model kurva titrasi pH terhadap larutan yang bersifat asam atau basa. Pada subbab ini teori tentang konstanta kecepatan reaksi tidak dibahas terlalu detail, hanya diturunkan untuk mencari pengaruh konstanta ini terhadap kecepatan reaksi penetralan.

Larutan adalah campuran yang serba sama (homogen) antara zat terlarut dan zat pelarut. Zat terlarut biasanya jumlahnya lebih kecil dibandingkan dengan pelarut. Konsentrasi zat terlarut di dalam larutan berhubungan dengan sifat koligatif larutan, dan dapat dinyatakan dengan beberapa satuan konsentrasi yaitu:

- ✓ Molaritas (mol/l)
- ✓ Molalitas (mol/1 kg pelarut)
- ✓ Fraksi mol
- ✓ Normalitas (gr ekivalen/l)

- ✓ Persentase (%)

Sifat koligatif juga dapat didefinisikan sebagai sifat-sifat larutan yang hanya bergantung pada jumlah partikel zat terlarut di dalam larutan tanpa di pengaruhi sifat-sifat kimia zat terlarut, meliputi:

- ✓ Penurunan tekanan uap jenuh
- ✓ Kenaikan titik didih
- ✓ Penurunan titik Beku
- ✓ Tekanan osmotik

Salah satu konsentrasi yang berhubungan dengan teori pH adalah normalitas yaitu jumlah gram ekivalen (grek) zat terlarut dalam tiap 1 liter larutan atau jumlah mgram-ekivalen (mgrek) zat terlarut dalam tiap 1 ml larutan (1 grek=1000 mgrek). Satuan ini nantinya banyak digunakan untuk menentukan konsentrasi larutan asam atau larutan basa dan ion yang terdapat dalam larutan (1 grek asam ~ 1mol H^+ dan 1 grek basa ~ 1 mol OH^-).

Teori pendukung yang harus dijelaskan terlebih dahulu sebelum membahas terlalu jauh adalah hukum kesetimbangan reaksi. Hukum kesetimbangan reaksi dapat dinyatakan dengan persamaan 2.1,

$$M[A]+n[B] \rightleftarrows p[C]+q[D] \qquad (2.1)$$

dengan [A], [B], [C] dan [D] adalah konsentrasi zat-zat A, B, C dan D saat kesetimbangan, (m, n, p dan q) adalah koefisien-koefisien reaksi dari zat A, B, C dan D, maka konstanta kesetimbangan K dapat dinyatakan sebagai:

$$K = \frac{[C]^p[D]^q}{[A]^m[B]^n} \qquad (2.2)$$

Besarnya konsentrasi produk yang dihasilkan pada suatu reaksi yang berjalan dalam kesetimbangan bergantung pada besar kecilnya harga K. Apabila K besar berarti konsentrasi zat-zat produk yang dihasilkan lebih banyak dibandingkan konsentrasi zat-zat reaktan (yang tersisa setelah reaksi kesetimbangan), dan sebaliknya.

Asam menurut teori Arrhenius adalah zat yang dapat menaikkan konsentrasi ion H^+ atau H_3O^+ didalam larutan. Ion H_3O^+ disebut ion hidronium, terjadi dari ikatan kovalen koordinasi antara ion H^+ yang dilepaskan suatu asam dan molekul air yang bertindak sebagai pelarut. Notasi [ ] menyatakan suatu konsentrasi. Konsentrasi air (H_2O) dalam hal ini tetap bertindak sebagai pelarut dan berharga 1. Menurut teori yang sama, maka Basa adalah suatu zat yang dapat menaikan konsentrasi ion OH^- (hidroksida) di dalam air.

Contoh asam

HCl adalah suatu zat asam karena di dalam larutannya dapat melepas ion H^+ menurut reaksi:

$$HCl_{(aq)} \longrightarrow H^+_{(aq)} + Cl^-_{(aq)} \qquad (2.3)$$

Contoh basa

NaOH adalah suatu zat basa karena di dalam air dapat melepas ion OH^- di dalam air menurut reaksi:

$$NaOH_{(aq)} \longrightarrow Na^+_{(aq)} + OH^-_{(aq)} \qquad (2.4)$$

Pemahaman tentang tetapan kesetimbangan yang dihubungkan dengan konsentrasi reaksi HCl dan NaOH adalah sebagai berikut:

- ✓ HCl

  $K_a=[H^+][Cl^-]/[HCl]$ atau $K_a=[H_3O^+][Cl^-]/[HCl]$ (2.5)

- ✓ NaOH

  $K_b=[Na^+][OH^-]/[NaOH]$ (2.6)

Bila harga K_a makin besar berarti kesetimbangan bergeser ke arah kanan, berarti hasil ionisasi besar. Dengan demikian asam ini bersifat asam kuat dan sebaliknya bila harga K_a semakin kecil asam tersebut bersifat asam lemah. Bila harga K_b semakin besar berarti kesetimbangan bergeser ke arah kanan, berarti hasil ionisasi besar. Dengan demikian basa ini bersifat basa kuat dan sebaliknya bila harga K_b semakin kecil basa tersebut bersifat basa lemah. Sedangkan tetapan kesetimbangan untuk air diturunkan dengan persamaan reaksi 2.7:

$$H_2O \longrightarrow H^+ + OH^- \qquad (2.7)$$

sehingga mempunyai konstanta kesetimbangan:

$$K_c=[H^+][OH^-]/[H_2O] \text{ atau } K\,[H_2O]= [H^+][OH^-] \qquad (2.8)$$

karena air sebagai pelarut, maka harga $[H_2O]$ selalu tetap dengan demikian $K[H_2O]$ adalah tetapan juga. Tetapan ini sering disebut sebagai konstanta kesetimbangan air dan dinyatakan sebagai:

$$K_w=[H^+][OH^-] \qquad (2.9)$$

Dengan perhitungan yang teliti harga K_w dapat ditentukan besarnya bergantung pada suhu saat pengukuran. Harga K_w yang digunakan biasanya adalah 1×10^{-14} di dalam suhu kamar 25 °C. Semakin besar suhunya maka harga K_w semakin besar pula.

Ion adalah partikel-partikel bermuatan yang bergerak didalam pelarut polar. Karena termuati, kehadirannya membentuk sebuah potensial listrik dan jika mengalir akan membentuk aliran arus. Setiap larutan polar dalam suatu reksi kimia mempunyai satu atau lebih ion dan jika digunakan untuk ionisasi akan membentuk dua partikel bermuatan yang saling berlawanan. Sebagai contoh air terbagi menjadi ion hidrogen dan ion hidroksida. Berdasarkan sedikit keterangan tersebut, maka bila

diinginkan untuk membentuk sistem kotrol reaksi kimia, langkah pertama yang harus dilakukan adalah pengukuran potensial listrik untuk ion-ion tersebut. Secara apriori di asumsikan bahwa partikel yang akan diamati adalah ion H^+, maka hubungan matematis untuk menerangkan logaritma adalah penggunaan notasi p untuk *potenz* yang berarti pangkat atau eksponen, konsep tersebut telah dikemukakan oleh kimiawan dari Denmark bernama Sorensen (1909), sehingga dalam pH dapat ditulis seperti persamaan

$$pH \equiv -\log a_{H^+} \tag{2.10}$$

dimana a_{H^+} adalah aktivitas ion hidrogen dan dinyatakan dalam satuan normalitas (g-ion/liter). Persamaan 2.11 dapat ditulis sebagai:

$$a_{H^+} = 10^{-pH} \tag{2.11}$$

Persamaan 2.10 dan 2.11 adalah persamaan umum untuk suatu unsur, maka bila dihubugkan dengan ukuran kekuatan asam atau basa, pH adalah logaritma negatif konsentrasi H^+ dinyatakan

$$pH=\log(1/[H^+])=-\log [H^+] \tag{2.12}$$

atau,

$$[H^+]=10^{-pH} \tag{2.13}$$

Semakin kecil harga pH maka keasaman suatu larutan semakin besar atau kebasaan suatu larutan semakin kecil. Sebaliknya semakin besar harga pH maka keasaman suatu larutan semakin kecil atau kebasaan suatu larutan semakin besar. Analogi dengan besaran pH, maka besaran lain seperti $[OH^-]$, K_w, K_a dan K_b dapat dinyatakan sebagai pOH, pK_w, pK_a dan pK_b, sehingga dapat dinyatakan:

$$pOH = -\log [OH^-] \quad (2.14)$$

$$pK_w = -\log K_w \quad (2.15)$$

$$pK_a = -\log K_a \quad (2.16)$$

$$pK_b = -\log K_b \quad (2.17)$$

Hubungan pH, pOH dan pK_w dapat dicari dengan cara sebagai berikut:

$$K_w = [H^+][OH^-] = 10^{-14}$$

$$\log K_w = \log[H^+] + \log[OH^-] = \log 10^{-14} = -14$$

$$-\log K_w = -\log[H^+] + -\log[OH^-] = -\log 10^{-14} = 14$$

$$pK_w = pH + pOH = 14 \quad (2.18)$$

sehingga $pK_w = pH + pOH$ dan $pH + pOH = 14$. Dengan demikian bila harga pH = pOH sama dengan 7 dikatakan larutan itu netral karena konsentrasi ion H^+ sama dengan ion OH^-. Pada pH lebih kecil dari 7 dikatakan larutan itu bersifat asam dan bila lebih besar dari 7 dikatakan larutan tersebut bersifat basa.

2.2 Kurva Titrasi

Subbab ini membahas lebih jauh tentang pengaruh perubahan larutan asam atau basa terhadap harga pH nya. Setiap penambahan pereaksi akan mengakibatkan perubahan harga pH. Suatu grafik yang diperoleh dengan mengeluarkan pH terhadap volume (konsentrasi) pereaksi yang ditambahkan disebut kurva titrasi. Bentuk kurva titrasi ini merelasikan konstanta

kesetimbangan ionisasi asam atau basa dan konsentrasi tiap-tiap ion. Pengukurannya memakai rumusan-rumusan pH yang telah dijelaskan sebelumnya. Dalam sistem pengendalian banyak dijumpai bahwa objektifitasnya adalah untuk mendapatkan kesetimbangan antara larutan asam dan larutan basa. Pembentukan kurva tersebut bergantung pada tipe-tipe pelarutnya apakah termasuk dalam kategori asam kuat atau lemah dan sebaliknya untuk basa. Larutan kuat (*'strong'*) didefinisikan dengan disosiasi lengkap dari hidrogen dan hidroksidanya. Sedangkan larutan lemah ("weak") sedikit sekali bagian yang terionisasi. Sehingga konsentrasi ion hidrogen dari asam lemah mempunyai fraksi yang kecil dari total konsentrasi basa. Untuk menghitung konsentrasi ion hidrogen dari campuran asam kuat dan basa kuat dicari terlebih dahulu kontribusinya terhadap air.

Dalam pembahasan ini digunakan contoh reaksi antara NaOH dan HCl. Keduanya pada dasarnya terionisasi secara lengkap.

Ionisasi reaksi tersebut ditulis:

$$HCl+NaOH+H_2O \rightleftarrows H^+ + OH^- + Na^+ + Cl^- \qquad (2.19)$$

Jumlah dari semua muatan negatip dan positip harus sama, maka

$$[Na^+]+[H^+] = [Cl^-]+[OH^-] \qquad (2.20)$$

Dengan mengasumsikan terjadi ionisasi lengkap dari kedua asam dan basa, maka $[Na^+]$ harus sama dengan konsentrasi basa x_B dan $[Cl^-]$ konsentrasi dari asam x_A sebagai berikut

$$x_B+[H^+]=x_A+[OH^-] \qquad (2.21)$$

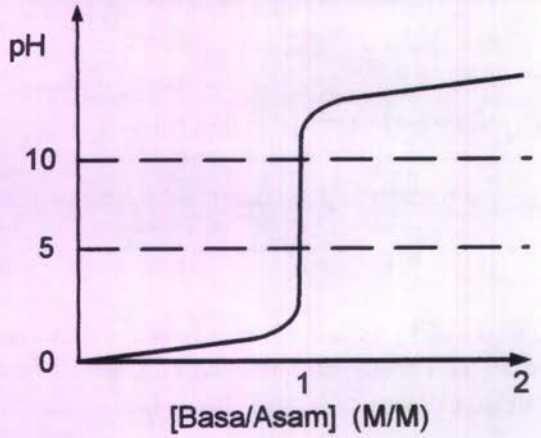
Hubungan $[H^+]$ dan $[OH^-]$ menurut persamaan 2.21 dengan mensubstitusikan $[OH^-]$ dari persamaan 2.18 didapat,

$$x_B - x_A = [H^+] - 10^{-14}/[H^+] \quad (2.22)$$

Persaman 2.22 diatas dapat digunakan untuk mencari hubungan konsentrasi asam basa terhadap harga pH berikut,

$$x_B - x_A = 10^{-pH} - 10^{-14} \quad (2.23)$$

Pengukuran pH dilakukan dengan mengukur secara selektif aktifitas dari ion Hidrogen menggunakan elektroda pH berimpedansi tinggi. pH merupakan fungsi logaritma, yaitu perubahan satu unit pH merepresentasikan perubahan sepuluh unit konsentrasi ion Hidrogen. Tabel 2.1 menunjukkan hitungan antara kedua konsentrasi ion Hidrogen dan ion Hidroksida terhadap nilai pH. Persamaan 2.23 dapat digunakan untuk mencari kurva titrasi yang dapat dilihat seperti pada gambar 2.1



Gambar 2.1 Kurva Titrasi Untuk Asam Kuat dan Basa Kuat (Doherty, 1999)

Tabel 2.1 Hibungan ion Hidrogen dan ion Hidroksida terhadap nilai pH (www.sensorex.com)

Konsentrasi ion Hidrogen (mol/liter pada suhu 25C)		
pH	H^+	OH^-
0	$(10^0) = 1$	$(10^{-14}) = 0{,}00000000000001$
1	$(10^{-1}) = 0.1$	$(10^{-13}) = 0{,}0000000000001$
2	$(10^{-2}) = 0.01$	$(10^{-12}) = 0{,}000000000001$
3	$(10^{-3}) = 0.001$	$(10^{-11}) = 0{,}00000000001$
4	$(10^{-4}) = 0.0001$	$(10^{-10}) = 0{,}0000000001$
5	$(10^{-5}) = 0.00001$	$(10^{-9}) = 0{,}000000001$
6	$(10^{-6}) = 0.000001$	$(10^{-8}) = 0{,}00000001$
7	$(10^{-7}) = 0.0000001$	$(10^{-7}) = 0{,}0000001$
8	$(10^{-8}) = 0.00000001$	$(10^{-6}) = 0{,}000001$
9	$(10^{-9}) = 0.000000001$	$(10^{-5}) = 0{,}00001$
10	$(10^{-10}) = 0.0000000001$	$(10^{-4}) = 0{,}0001$
11	$(10^{-11}) = 0.00000000001$	$(10^{-3}) = 0{,}001$
12	$(10^{-12}) = 0.000000000001$	$(10^{-2}) = 0{,}01$
13	$(10^{-13}) = 0.0000000000001$	$(10^{-1}) = 0{,}1$
14	$(10^{-14}) = 0.00000000000001$	$(10^0) = 1$

2.3 Sistem Pengukuran dan Sensor pH

Sistem pengukuran pH terdiri dari tiga bagian yaitu: sensor pH (elektroda pengukur, elektroda referensi, dan sensor temperatur), preamplifier dengan impedansi input tinggi, dan analiser atau transmitter. Elektroda pH dapat digunakan dengan tegangan catu atau tanpa tegangan catu. Jika menggunakan tegangan catu, maka tegangan positip dihubungkan pada elektroda pengukur sedangkan tegangan negatip dihubungkan pada elektroda referensi. Elektroda pengukur yang sensitif terhadap ion Hidrogen menghasilkan tegangan yang besarnya sesuai konsentrasi ion Hidrogen dalam larutan. Elektroda

referensi menghasilkan tegangan yang stabil dan akan dibandingkan dengan potensial elektroda pengukur. Elektroda pH dapat dicatu dengan tegangan yang bervariasi tergantung pada pH dari larutan yang diukur. Impedansi dari keluaran elektroda pH adalah sangat tinggi sehingga untuk mengukur nilai pH diperlukan penguat yang mempunyai impedansi sama atau lebih tinggi.

Sensor pH digunakan pada sistem pengendalian pH, yang mana sensor dan *transmitter* yang digunakan adalah elektrode gelas. Elektrode gelas terbuat dari lapisan gelas yang dapat merespon ion, sehingga dapat menangkap perubahan pergerakan ion dalam struktur membran. Elektrode yang konvensional mempunyai tiga komponen utama, yaitu: membran elektrode, larutan pengisi bahan dalam dan elektrode referensi. Lapisan elektrode bercampur dengan badan gelas, sehingga lapisan luar akan berinteraksi dengan proses, sedangkan lapisan dalam berinteraksi dengan larutan pengisi. Larutan pengisi berisi ion hidrogen dengan aktivitas konstan sedangkan proses juga berisi ion dengan aktivitas yang tidak diketahui. Perubahan tersebut dapat diukur dengan adanya hubungan elektrik yang stabil dengan larutan didalamnya. Perubahan pontesial pada membran sebanding dengan perubahan pH dalam proses.

Beberapa elektrode menggunakan sistem ion metal – metal, misalnya mercuri yang terkandung dalam *mercury perchlorate* dan asam *perchlorate*. Tetapi kebanyakan elektrode gelas menggunakan larutan pengisi asam *hydrochloric* atau *chloride* penyangga dimana didalamnya terkandung *silver – silver* elektrode *chloride*.

Saat hasil pengukuran dari elektrode referensi identik, maka kontribusi tegangan akan menghilangkan satu sama lain. Hal itu menyebabkan perbedaan tegangan pada sirkuit terdapat di sepanjang membran.

Elektroda pH mempunyai resistansi internal yang sangat tinggi, sehingga mempersulit pengukuran. Oleh karena itu dibutuhkan pH meter yang mempunyai impedansi input yang tinggi agar tidak terjadi kehilangan sinyal dari sensor. Rangkaian dari kompensasi suhu pH meter mengacu pada persamaan Nernst, yang secara umum dituliskan dalam persamaan matematis:

$$E = Ex + \frac{\{2.3RTK \log(ai)\}}{nF} \qquad (2.24)$$

dengan,

Ex	= konstanta tergantung elektroda referensi
R	= konstanta
TK	= temperatur absolut (Kelvin)
n	= muatan ion
F	= konstanta
ai	= aktivitas ion

untuk pengukuran pH pada suhu kamar (T=25^{0}C) dan n=1, maka:

$$\frac{2.3RTk}{nF} = 59.16mV / nF \qquad (2.25)$$

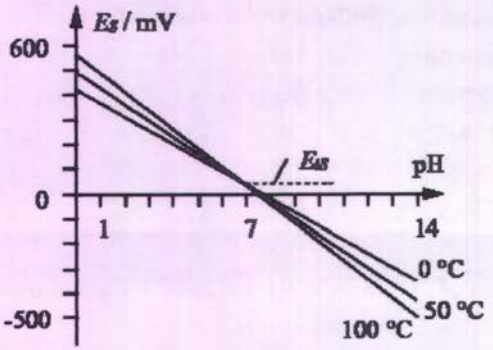
Oleh karena pH adalah logaritma negatip dari konsentrasi ion hidrogen, maka persamaan umum untuk berbagai kondisi suhu dapat dinyatakan sebagai:

$$E = Ex - 1.98Tk\ pH \qquad (2.26)$$

Elektroda dari alat ukur pH adalah sensitif terhadap suhu, sehingga diperlukan kompensasi yang dapat dilakukan secara manual atau otomatis. Dengan cara manual diperlukan pengukur temperatur terpisah, dan pengendalian kompensasi manual pH

meter dengan cara mengatur nilai kompensator temperatur yang mendekati temperatur terukur. Secara otomatis kompensasi temperatur dilakukan dengan pemberian sinyal dari sensor suhu pada pengkondisian sinyal tegangan elektroda pH meter.

Perubahan temperatur dalam larutan akan mempengaruhi keluaran elektroda pH seperti tampak pada persamaan Nernst. Variasi sensitifitas elektroda terhadap temperatur adalah fungsi linier. Gambar 2.2 menunjukkan efek dari variasi temperatur terhadap sinyal elektroda pH.



Gambar 2.2 Grafik hubungan tegangan keluaran elektrode terhadap variasi temperatur (Yien, 2001)

Pada grafik tersebut terlihat bahwa tiga garis dengan kemiringan yang berbeda berpotongan pada satu titik (pada pH 7 dan titik 0 mVolt), hal ini menunjukkan bahwa pada titik itu tidak dipengaruhi oleh suhu dan disebut sebagai titik isopotensial. Temperatur mempunyai pengaruh signifikan pada pH mendekati harga 0 dan harga 14. Saat harga pH adalah 3.0 atau 11.0, perubahan temperatur 15^0C dapat mengakibatkan error pH sebesar 0.2. Error karena pengaruh temperatur pada keluaran

elektroda adalah linier sebesar: 0.03 pH error/pH unit/10^0C. Nilai pH aktual terpengaruh suhu karena perubahan aktifitas ion Hidrogen dalam larutan disebabkan ionisasi dari senyawanya, dalam hal ini kompensator temperatur tidak dibutuhkan. Kompensator temperatur hanya dibenarkan untuk kompensasi perubahan sinyal output elektroda karena pengaruh suhu, tidak untuk perubahan pH larutan aktual. Temperatur juga mempengaruhi impedansi membran kaca. Untuk setiap perubahan 8^0C di bawah suhu 25^0C, impedansi spesifiknya mendekati dua kali.

Perbedaan tegangan antara elektroda pH didalam dan diluar kaca diberikan oleh persamaan:

$$Eo - Ei = 0.1984(T+273.16)(7 - pH) \qquad (2.27)$$

Pada grafik 2.2 diatas gradien termal untuk suhu 25 ^{0}C elektroda pH mengeluarkan 64,12 mV/pH, dengan acuan pH7. Pada suhu 100^0C gradiennya menjadi 74mV/pH. Pengaruh suhu ini menyebabkan error yang besarnya seperti ditunjukkan pada tabel 2.2

Tabel 2.2 Error pada elektroda oleh pengaruh temperatur (www.sensorex.com)

	pH 2	pH 3	pH 4	pH 5	pH 6	pH 7	pH 8	pH 9	pH 10	pH 11	pH 12
5°	0.3	0.24	0.18	0.12	0.06	0	0.06	0.12	0.18	0.24	0.3
15°	0.15	0.12	0.09	0.06	0.03	0	0.03	0.06	0.09	0.12	0.15
25°	0	0	0	0	0	0	0	0	0	0	0
35°	0.15	0.12	0.09	0.06	0.03	0	0.03	0.06	0.09	0.12	0.15
45°	0.3	0.24	0.18	0.12	0.06	0	0.06	0.12	0.18	0.24	0.3
55°	0.45	0.36	0.27	0.18	0.09	0	0.09	0.18	0.27	0.36	0.45
65°	0.6	0.48	0.36	0.24	0.12	0	0.12	0.24	0.36	0.48	0.6
75°	0.75	0.6	0.45	0.3	0.15	0	0.15	0.3	0.45	0.6	0.75
85°	0.9	0.72	0.54	0.36	0.18	0	0.18	0.36	0.54	0.72	0.9

Kompensator temperatur disertakan pada alat ukur karena elektroda pH terpengaruh oleh perubahan suhu. Untuk menormalkan hasil pengukuran biasanya dipakai termometer resistansi yang dapat berupa termistor atau platinum. Untuk termistor tipe NTC (Negative Temperature Coefisient) mempunyai resistansi 12 kΩ pada 25^{0}C, dan 0,95 kΩ pada suhu 100^{0}C, serta β = 3750^{0}K dan disipasi daya = 7 mW/^{0}C. Besar nilai resistansi ditentukan oleh persamaan:

$$R_\theta = R_{\theta 1} \exp \beta\left\{\left(\frac{1}{\theta}\right)-\left(\frac{1}{\theta_1}\right)\right\} \qquad (2.28)$$

dengan,

R_θ = resistansi pada suhu θ
$R_{\theta 1}$ = resistansi pada suhu referensi (289^{0}K)
β = konstanta termistor
θ = suhu (^{0}Kelvin)

Untuk termistor tipe NTC (Negative Temperature Coefisient) mempunyai resistansi 12 kΩ pada 25^{0}C, dan 0,95 kΩ pada suhu 100^{0}C, serta β = 3750^{0}K dan disipasi daya = 7 mW/^{0}C (Mohan, 1994).

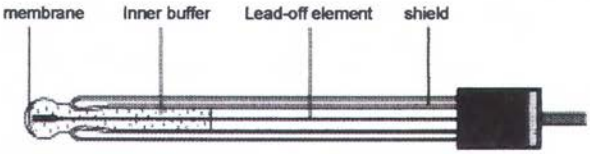
Elektroda merupakan sensor pada pH meter yang akan menghasikan tegangan berdasarkan pH yang diukur. Elektroda ini dapat menghasilkan tegangan karena elektroda ini mempunyai cairan aktif yang dapat menimbulkan tegangan. Ada tiga jenis elektroda , yaitu:

- Elektroda Pengukur pH

Ada banyak jenis elektroda pengukur pH, salah satu yang paling banyak digunakan adalah elektroda pH gelas, karena mempunyai kepekaan yang tinggi terhadap ion hidrogen.

Elektroda pH gelas adalah elektroda ion selektif terhadap ion hidrogen (H^+). Unsur yang mengukur ion selektifnya adalah membran gelas yang umum digunakan yaitu Lithium Silikat ditambah ion-ion Barium dan Lanthanum. Aktifitas air dalam larutan memegang peranan penting dalam respon pH dari membran gelas. Jika ion H^+ yang menembus membran tipis sangat tinggi atau pelarut non air dan beraksi dengan cairan aktif akan potensial yang sangat kecil, potensial yang terukur akan berdevisiasi dari yang sesungguhnya.

Bagian dari elektroda pH gelas ditunjukan pada gambar 2.3 yang mempunyai empat bagian dasar, yaitu membran tipis sebagai pembatas antara inner buffer (cairan aktif) dengan larutan, inner buffer, lead-off element dan shield (pembungkus).



Gambar 2.3 Elektroda Gelas (Yien, 2001)

Sebelum dipakai membran gelas harus direndam dalam air atau buffer sehingga terbentuk lapisan tipis (hidrat jeli). Lapisan ini harus ada pada elektroda pH membran gelas agar pengukuran pH dapat berhasil, karena adanya sifat higroskopis dari gelas. Jika lapisan tersebut hilang atau rusak akan memberikan respon yang kecil atau tidak sama sekali, sehingga slope tidak memuaskan dan pengukuran menjadi salah. Untuk membentuk lapisan tersebut membran direndam dalam air atau buffer pada suhu kamar selama 24 – 48 jam, proses ini dapat dipercepat dengan menaikkan suhu. Bentuknya kecil sehingga mudah digunakan. Lapisan jeli dengan kerusakan ringan dapat diperbaiki dengan membentuk lapisan tersebut kembali, yaitu

dengan menyimpan atau merendamnya dalam larutan KCL 3M pada suhu 50 °C selama beberapa jam.

Bagian dalam elektroda gelas terdapat kawat perak (Ag) yang dilapisi perak Klorida (Agcl) yang terendam dalam larutan HCl. Ujung bawah elektroda gelas terbuat dari membran gelas tipis semipermiable yang hanya peka terhadap ion hydrogen. Apabila bagian luar dari gelas ini terdapat larutan dengan konsentrasi ion hydrogen tinggi (asam), maka ion H^+ akan menerobos masuk kedalam elektroda gelas yang semipermiable sehingga ion H^+ dalam gelas bertambah. Begitu juga sebaliknya jika larutan yang diukur bersifat basa maka ion H^+dalam gelas akan berkurang. Jadi akan bertambah atau berkurangnya ion H^+ dalam elektroda gelas tergantung banyaknya ion H^+dalam larutan yang diukur, aliran H^+ ini ditunjukan gambar 2.4. Elektroda gelas akan menimbulkan beberapa tegangan, tegangan ditimbulkan kaca elektroda, cairan atau larutan dan logam elektroda.

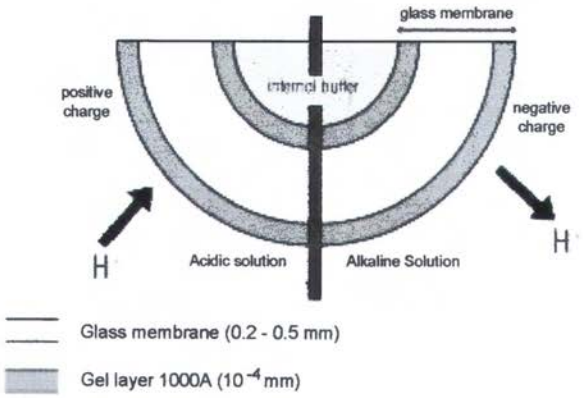
Total tegangan yang terjadi antara elektroda dan larutan dapat dirumuskan sebagai berikut :

$$E_g=E_1+E_2+E_3 \qquad (2.29)$$

dengan,.

E_1 = Tegangan kaca terhadap larutan
E_2 = Tegangan cairan dalam kaca terhadap kaca
E_3 = Tegangan cairan dalam kaca terhadap logam

Tegangan E berbanding langsung dengan harga pH larutan. Semakin tinggi konsentrasi ion hydrogen dalam larutan semakin tinggi pula tegangan E_1. Sedangkan tegangan E_2 dan E_3 konstan (tidak dipengaruhi larutan yang diukur).



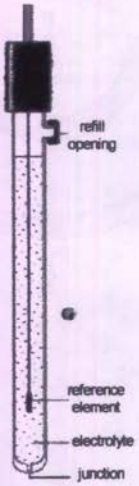
Gambar 2.4. Skematik aliran H^+ pada membran elektroda

- Elektroda Referensi

Elektroda kedua yang dibutuhkan dalam pengukuran pH adalah elektroda referensi yang berfungsi sebagai pelengkap rangkaian agar didapat pengukuran yang stabil. Elektroda Referensi memberikan dan menjaga potensial yang tetap dan tidak dipengaruhi oleh karakteristik larutan, bagian-bagian dari elektroda referensi ditunjukan pada gambar 2.5.

Elektroda referensi terdiri atas tabung gelas dalam yang berisi perak dan perak klorida (Ag dan AgCl) yang dicelupkan dalam potassium–clorida (KCL) konsentrasi tinggi (jenuh). Bagian bawah tabung luar juga terdapat diagfragma penghubung dengan larutan yang akan diukur.

Harga beda potensial yang timbul dalam referensi elektroda adalah konstan, dimana E_4 adalah senjang potensial antara elektroda penyalur dengan larutan pembantu (KCL) sedangkan E_5 merupakan difusi potensial.



Gambar 2.5 Elektroda Referensi (Yien, 2001)

- Elektroda Kombinasi

Umumnya transduser pH yang ada dipasaran menyatukan elektroda referensi dan elektroda gelas dalam satu tabung atau dengan nama elektroda kombinasi seperti gambar 2.6. Bagian yang paling sensitif terhadap perubahan harga pH adalah membran elektroda gelas itu sendiri. Perbedaan tegangan terjadi antara permukaan kedua membran gelas. Untuk mengukur perbedaan tegangan, harus ada suatu kontak elektrik pada tiap – tiap sisi membrannya.

Gambar 2.6 menunjukan bagian-bagian dari elektroda kombinasi, dimana besarnya tegangan yang dihasilkan elektroda adalah gabungan antara tegangan yang dihasilkan elektroda gelas dan referensi dapat dicari dengan persamaan :

$$E=E_1+E_2+E_3+E_4+E_5 \qquad (2.30)$$

dengan,

E = Beda potensial total yang terukur

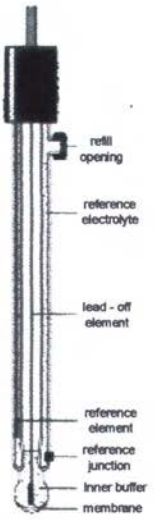
E_1 = Beda potensial antara sisi luar membran gelas dengan larutan yang terukur

E_2 = Beda potensial antara sisi dalam membran gelas dengan larutan HCl dalam gelas

E_3 = Beda potensial antara elektroda penyalur (Ag+AgCl) dengan HCl

E_4 = Beda potensial antara elektroda penyalur (Ag+AgCl) dengan KCl

E_5 = Beda potensial karena difusi potensial pada diagfragma



Gambar 2.6 Elektroda Kombinasi (Yien, 2001)

Beberapa beda potensial diatas haruslah konstan agar diperoleh pengukuran yang seteliti mungkin. Selain E_1 semua

beda potensial diatas adalah konstan, sehingga menurut persamaan Nerst diperoleh tegangan output untuk elektroda gelas :

$$Eg = Eo\frac{0{,}00592.T.pH}{298} \qquad (2.31)$$

dengan,

Eg = Beda potensial total elektroda gelas
Eo = Beda potensial pada elektroda gelas yang besarnya konstan
T = Temperatur (°K) , dimana pada 0°C = 273 °K

Sedangkan elektroda referensi menghasilkan tegangan konstan sebesar E_r.

$$E_r=E_4+E_5 \qquad (2.32)$$

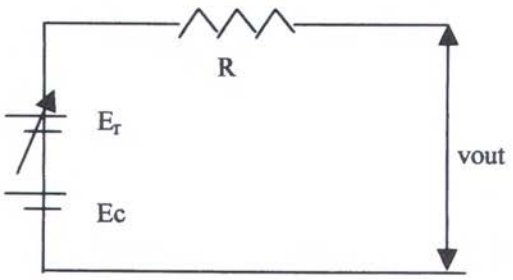
Maka tegangan output elektroda adalah:

$$E=E_g-E_r \qquad (2.33)$$

$$E = Eo + \frac{0{,}00592.T.pH}{298} - E_r$$

$$E = Eo - E_r + \frac{0{,}00592.T.pH}{298}$$

$$E=E_c+\frac{0{,}00592.T.pH}{298} \qquad (2.34)$$



Gambar 2.7 Rangkaian Ekuivalen elektroda kombinasi

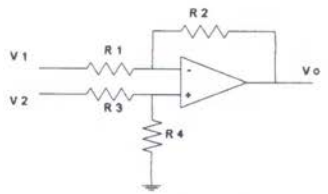
Sehingga rangkaian ekivalen dari elektroda dapat dilihat pada gambar 2.7.

dengan,

R = Tahanan dalam elektroda
= 50 –500 Mega Ohm
E_T = Tegangan yang tergantung harga pH
$= \frac{0{,}00592 T.pH}{298}$
E_c = Tegangan yang besarnya konstan

2.4 Penguat Instrumentasi

Rangkaian pengkondisi sinyal dapat dibuat dari rangkaian op-amp sederhana.



Gambar 2.8 Rangkaian penguat sederhana (Curtis, 1997)

Sinyal keluaran dari sensor merupakan sinyal dengan tingkat yang kecil sehingga diperlukan suatu rangkaian yang menghasilkan penguatan sehingga menghasilkan sinyal yang sesuai yang dapat dibaca oleh ADC. Dengan menggunakan prinsip superposisi dan asumsi bahwa arus masukan op-amp sangat kecil dan diabaikan maka didapatkan rumus 2.35 sebagai berikut:

$$V_0 = \frac{R4}{R3+R4}(1+\frac{R2}{R1})V_2 - \frac{R2}{R1}V_1 \qquad (2.35)$$

Apabila $R3/R4 = R1/R2$ maka,

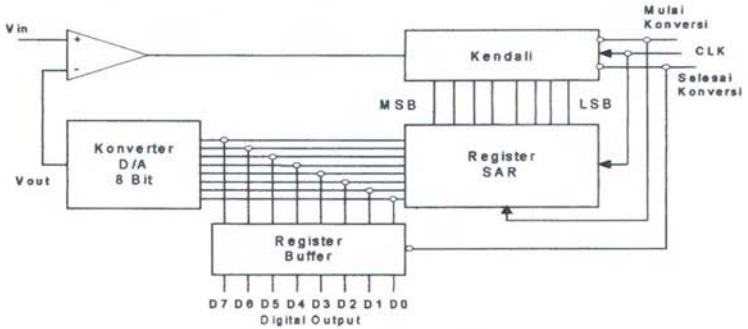
$$V_0 = \frac{R2}{R1}(V_2 - V_1) \qquad (2.36)$$

Apabila sinyal V_1 dan V_2 mempunyai resistansi sumber R_1 dan R_2 maka keduanya dijumlahkan pada R_3 dan R_4. Sumber sinyal V_1 mengalami resistansi sebesar $R3 + R4$. Apabila $V_2 = 0$ maka masukan inverting berada pada potensial nol (*ground potential*), dan V_1 dibebani oleh R_1.

2.5 ADC (Analog to Digital Converter)

ADC merupakan pengkonversi sinyal analog menjadi digital. Pengkonversi sinyal diperlukan dalam setiap sistem yang bekerja menggunakan komputer ataupun mikroprosesor karena sinyal-sinyal listrik yang bekerja pada proses adalah sinyal analog sedangkan komputer atau mikroprosesor bekerja dengan sinyal digital. Informasi digital adalah dalam bentuk bilangan biner. Pada saat digunakan pada komputer, bilangan biner ini disebut *binary word*. Tipe ADC yang sering digunakan adalah ADC 0804 yang merupakan pengkonversi sinyal 8 bit. ADC 0804 bekerja dengan cara *successive approximation*. .Pengkonversi ini tampak seperti penempatan memory atau port I/O ke mikroprosesor dan tidak memerlukan logika interfacing. Prinsip kerja rangkaian

diatas adalah jika sinyal masukan mulai konversi dari unit kendali diberi logika 0, maka register SAR *(Successive Approximation Register)* akan mereset sehingga keluaran V_{out} unit DAC (*Digital to Analog*) menjadi nol. Proses konversi diawali dengan pengesetan bit paling berarti (MSB) register SAR oleh unit kendali. Selanjutnya data digital dalam register SAR dikonversikan ke analog oleh unit DAC. Hasil konversi V_{out} dibandingkan dengan sinyal masukan V_{in} oleh unit pembanding. Bila V_{out} lebih besar dari V_{in} maka unit pembanding akan mengirim sinyal negatif ke unit kendali.



Gambar 2.9 ADC Succesive approximation (Datasheet ADC0804 National Semiconductor)

Dengan sinyal negatif ini, unit kendali akan mereset bit paling berarti (MSB) register SAR. Sebaliknya bila V_{out} lebih kecil dari V_{in}, unit pembanding akan mengirim sinyal positif ke unit kendali. Dengan sinyal positif ini, unit kendali akan tetap mengeset bit paling berarti (MSB). Pada pulsa clock berikutnya unit kendali akan mengeset bit yang lebih rendah yaitu bit ke 7 register SAR. Kemudian data dikonversikan oleh unit DAC, dan hasil konversi V_{out} dibandingkan dengan sinyal masukan V_{in}. Sinyal hasil perbandingan akan menentukan unit kendali untuk mengeset atau mereset register SAR. Demikian seterusnya proses ini berlangsung sampai nilai V_{in} sama dengan V_{out}.

Apabila konversi telah selesai, unit kendali mengirim sinyal selesai konversi yang berlogika rendah. Tepi turun sinyal ini akan mengisikan data digital yang ekuivalen dengan nilai V_{in}, ke dalam register penahan.

2.6 Port Paralel

Port paralel adalah port yang sering digunakan untuk antar muka antara komputer dengan perangkat keras luar yaitu printer tetapi penggunaan port paralel sekarang tidak hanya digunakan sebagai port printer tetapi sering digunakan untuk keperluan antar muka (interfacing) yang lain salah satunya sebagai sistem akuisisi data pada sistem kontrol. Port ini menyediakan input hingga 9 bit dan output 12 bit pada saat yang bersamaan dan untuk menggunakannya dibutuhkan rangkaian eksternal. Port ini terdiri dari 4 jalur input, 5 jalur status, dan 8 jalur data.

Port parallel terdiri dari tiga alamat yaitu untuk data, status, dan kontrol. Alamat-alamat tersebut tersusun berurutan, jika port data terletak pada alamat 378h maka port status akan terletak pada alamat 379h dan port control pada alamat 37Ah, seperti pada tabel 2.3.

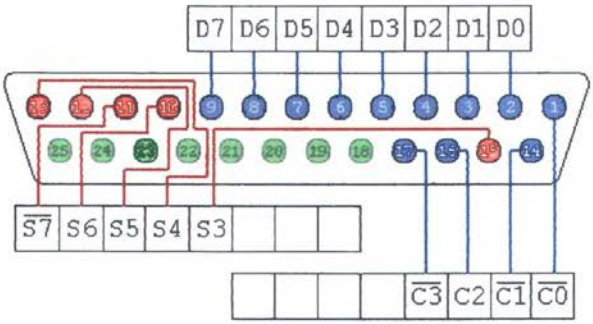
Port paralel seperti yang ditunjukan pada tabel 2.3 tidak semuanya dimiliki oleh sebuah komputer (PC). Ketika komputer dinyalakan maka secara otomatis BIOS akan memeriksa alamat 3BCh dahulu jika ditemukan port paralel maka port tersebut merupakan LPT1 dan jika ditemukan port lain maka port kedua adalah LPT2 dan seterusnya.

Jika pada komputer hanya ditemukan satu port paralel saja misalnya pada alamat 378h maka port tersebut merupakan LPT1. Port ini bisa kita lihat di belakang cpu, dengan konektor

female tipe D-25. Fungsi masing-masing pin port tersebut dapat dilihat dalam table 2.4 sedangkan bentuk fisiknya dapat dilihat pada gambar 2.10.

Tabel 2.3 Alamat port paralel (www.senet.com.au/~cpeacock)

Printer	Data Port	Status	Control
LPT1	0x03BC	0x03BD	0x03BE
LPT2	0x0378	0x0379	0x037A
LPT3	0x0278	0x0279	0x027A



Gambar 2.10 Pin-pin pada konektor D-tipe 25 (www.senet.com.au/~cpeacock)

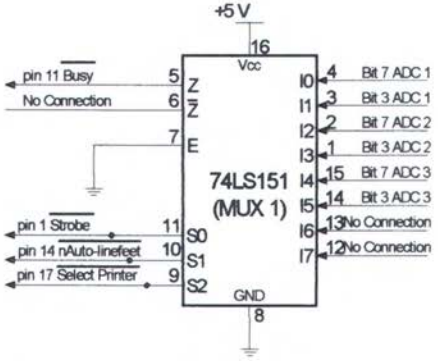
Ada beberapa cara yang sering digunakan untuk proses pembacaan data 8 bit dengan menggunakan 4 jalur input pada LPT1. Salah satu cara yang sering digunakan adalah *Nibble Mode*. Nibble mode (4 Bit) adalah metode pembacaan setengah byte, metode ini sering digunakan karena dengan menggunakan metode ini kita tidak perlu merubah port menjadi mode inverse dan menggunakannya sebagai jalur data. Mode ini memakai jalur Quad 2 menjadi jalur 1 dengan cara melewatkannya dalam multiplexer untuk membaca data 4 bit pertama pada waktu tertentu. Kemudian multiplexer dengan men-switch selector pada

multiplekser sehingga pada proses selanjutnya dilakukan pembacaan data 4 bit kedua.

Tabel 2.4 Fungsi tiap pin konektor D-Type25 Parallel Port (www.senet.com.au/~cpeacock)

Pin No. (D-Type 25)	SPP Signal	Direction In/Out	Register	Hardware
1	nStrobe	In/Out	Control	Yes
2	Data 0	Out	Data	
3	Data 1	Out	Data	
4	Data 2	Out	Data	
5	Data 3	Out	Data	
6	Data 4	Out	Data	
7	Data 5	Out	Data	
8	Data 6	Out	Data	
9	Data 7	Out	Data	
10	nAck	In	Status	
11	Busy	In	Status	Yes
12	Paper-Out PaperEnd	In	Status	
13	Select	In	Status	
14	nAuto-Linefeed	In/Out	Control	Yes
15	nError / nFault	In	Status	
16	nInitialize	In/Out	Control	
17	nSelect-Printer nSelect-In	In/Out	Control	Yes
18-25	Ground	Gnd		

Hasil pembacaan data tersebut kemudian akan digabungkan menjadi bentuk byte oleh sebuah perangkat lunak. Salah satu kerugian dengan menggunakan mode ini adalah proses pembacaannya menjadi lambat. Rangkaian eksternal yang digunakan adalah IC multiplexer dengan tipe 74LS151, adapun rangkaiannya adalah sebagai berikut :



Gambar 2.11 Rangkaian eksternal dengan IC 74LS151 (Datasheet SN54/74LS151 MOTOROLA)

2.7 Jaringan Syaraf Tiruan

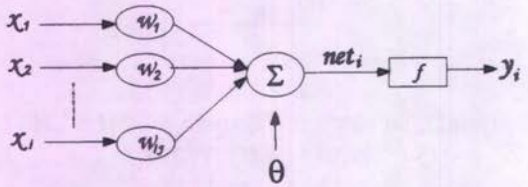
Jaringan Syaraf dapat mensintesis sebuah memori asosiatif yang dapat menghasilkan sebuah keluaran yang sesuai ketika diberi masukan dan menggeneralisasikan ketika diberi masukan yang baru. Unsur penting pada Jaringan Syaraf Tiruan adalah model syaraf, proses belajar dan arsitektur jaringan.

Sebagai salah satu elemen komputasi yang membuat hampir keseluruhan model sistem saraf dikenal juga dengan *artificial neurons* dan bisa bertindak sebagai simpul, satuan maupun elemen-elemen proses. Sedangkan sebagai jaringan saraf, model matematikanya dibangun berdasar beberapa pengertian yaitu:

a. Pengolahan informasi terjadi pada banyak elemen tunggal yang disebut unit syaraf (*neuron*).
b. Sinyal dilewatkan antar unit melalui untai penghubung.

c. Setiap untai penghubung mempunyai bobot hubung.
d. Setiap unit menggunakan fungsi aktivasi untuk sinyal keluaran.

Berdasarkan pemahaman untuk Jaringan Syaraf Tiruan tersebut, maka dapat dimodelkan satu sel syaraf seperti terlihat pada gambar 2.12:



Gambar 2.12 Model syaraf tiruan (Freeman, Skapura, 1992)

Pada gambar 2.12, x menyatakan suatu unit input syaraf, w adalah bobot hubung, θ bias, f fungsi aktivasi dan y adalah output Jaringan Syaraf Tiruan. Karena dasar rancang bangun teori Jaringan Syaraf Tiruan berdasar pada mekanisme otak manusia, maka lebih tepat model syaraf disini didefinisikan sebagai suatu unit elemen pemroses (*ep*). Sebagai *ep* syaraf membutuhkan harga-harga input jaringan dan beberapa fungsi relasi terhadap output. Selanjutnya sebagai suatu jaringan, input untuk syaraf tersebut dinyatakan dalam bentuk.

$$net_i = \sum x_j w_{ij} + \theta \tag{2.37}$$

sedangkan output untuk setiap unit model syaraf adalah

$$y_i = f(net_i) \tag{2.38}$$

dengan notasi i menyatakan input dan ij adalah bobot pada untai input ke syaraf. Bias θ berfungsi agar output suatu unit syaraf berada pada rentang tertentu misal dari 0 sampai 1, dan tidak akan melebihi harga-harga ini atau dengan kata lain bahwa output satu unit syaraf adalah termampatkan (*squashing*). Fungsi aktivasi f menentukan hasil output unit sesuai dengan tingkat aktivasi pada inputnya. Fungsi ini menyatakan secara eksplisit fungsi dari input jaringan. Ada tiga jenis dasar fungsi aktivasi yaitu:

1. Fungsi *Bipolar*
2. Fungsi Symetric Saturating Linear
3. Fungsi Bipolar Sigmoid

✓ Fungsi Bipolar

Fungsi aktivasi ini mempunyai persamaan

$$f(net)=\begin{cases}+1 & ,net>0\\ 0 & ,net=0\\ -1 & ,net<0\end{cases} \qquad (2.39)$$

dan kurvanya dapat dilihat pada gambar 2.13a.

✓ Fungsi Symetric Saturating Linear

Fungsi ini dapat dilihat pada gambar 2.13b dan dapat dinyatakan dengan persamaan 2.40.

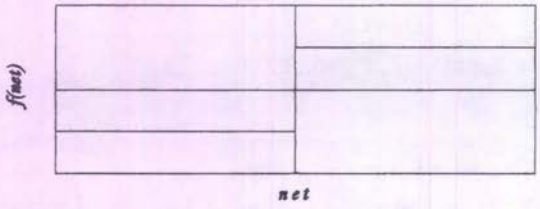
$$f(net)=\begin{cases}+1 & ,net\geq 1\\ net & ,-1\leq net\leq 1\\ -1 & ,net\leq -1\end{cases} \qquad (2.40)$$

✓ Fungsi Bipolar Sigmoid

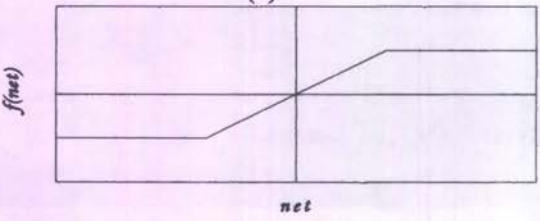
Fungsi *Bipolar Sigmoid* adalah fungsi aktivasi yang paling sering digunakan pada jaringan syaraf tiruan. Pada lapisan tersembunyi digunakan lebih banyak karena kemampuan merelasikan fungsi lebih lembut. Salah satu contoh *Bipolar sigmoid* adalah fungsi logistik ditunjukkan oleh persamaan dibawah ini:

$$f(net) = \frac{1-\exp(-net)}{1+\exp(-net)} \qquad (2.41)$$

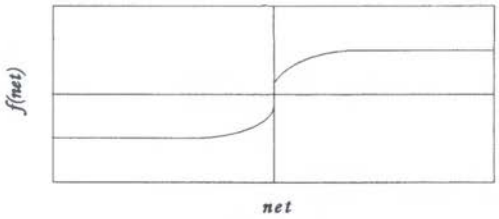
Selain fungsi-fungsi yang tersebut diatas, dalam alikasi jaringan syaraf tiruan terdapat banyak fungsi aktivasi, seperti fungsi linier, fungsi gabungan logaritmik, bipolar sigmoid, tangen hiperbolik, dan sebagainya. Pemakaian fungsi-fungsi sangat bergantung pada tujuan penggunaan jaringan syaraf tiruan atau output yang diinginkan. Biasanya antara input dan output dari jaringan syaraf tiruan mempunyai fungsi aktivasi yang sama dengan fungsi pada unit tersembunyi adalah sigmoid.



(a)



(b)



(c)

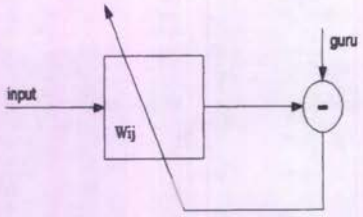
Gambar 2.13 (a) Fungsi *Bipolar*, (b) Fungsi *Symetric Saturating Liniear*, (c) Fungsi *Bipolar Sigmoid* (Kusumadewi, 2003)

2.7.1 Konsep Proses Belajar JST

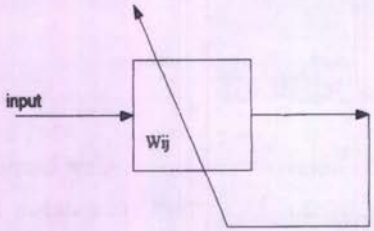
Kemampuan belajar merupakan sifat utama yang penting dari jaringan syaraf tiruan. Meskipun sulit untuk mendefinisikan arti "belajar" (*learning*) secara tepat, proses belajar pada konteks jaringan syaraf tiruan dapat didefinisikan sebagai perubahan terhadap bobot hubung (*connection weight*) yang berada pada untai hubungan (*connection*) sehingga output jaringan sesuai dengan yang diinginkan. Terdapat dua hal penting pada proses belajar. Pertama, apakah lingkungan (sistem) dimodelkan terlebih dahulu atau tidak untuk dijadikan target pada proses belajar. Hal ini disebut dengan paradigma belajar. Yang kedua adalah bagaimana cara mengubah parameter bobot hubung tersebut, yang dikenal dengan aturan atau algoritma belajar.

Paradigma belajar dapat diklasifikasikan atas: belajar dengan pengawasan (*supervised learning*) seperti ditunjukkan pada gambar2.14, belajar dengan penguatan (*reinforcement*

learning) dan belajar tanpa pengawasan (*unsupervised learning*) seperti ditunjukkan pada gambar2.15.



Gambar2.14 Belajar dengan pengawasan



Gambar 2.15 Belajar tanpa pengawasan

Pada paradigma belajar dengan pengawasan, harga bobot hubung diubah berdasarkan selisih antara target dengan harga output jaringan sebenarnya. Paradigma belajar tanpa pengawasan tidak memerlukan target, jaringan hanya diberikan urutan input saja. Jaringan memeriksa struktur dasar data input atau korelasi antar pola-pola data input dan mengorganisasikan ke dalam kategori-kategori berdasarkan korelasinya. Ada empat jenis aturan belajar, yaitu: aturan koreksi kesalahan, aturan Boltzman, aturan Hebbian dan aturan kompetitif. Aturan yang sering digunakan oleh beberapa peneliti adalah koreksi terhadap kesalahan (*error backpropagation*) seperti yang akan digunakan dalam penelitian ini.

2.7.2 Arsitektur Jaringan Syaraf Tiruan

Mekanisme jaringan syaraf tiruan sebagai elemen pemroses dapat dipandang sebagai garis-garis arah aliran sinyal yang menghubungkan unit input dengan unit output. Oleh karena itu arsitektur jaringan syaraf tiruan didasarkan pada bentuk konfigurasi hubungan antar unit tersebut. Berdasar arah sinyal pada satu unit jaringan syaraf tiruan, secara garis besar ada dua jenis arsitektur yaitu: Jaringan Umpan Maju (*Feedforward Network*) dan Jaringan Umpan Balik atau Berulang (*Feedback* atau *Recurrent Network*).

Jaringan umpan maju (JU) adalah jaringan yang arah sinyalnya dalam arah maju saja dan tidak ada ikal (*loop* atau *feedback*). Secara umum, JU bersifat statik dimana jaringan hanya menghasilkan satu set output untuk setiap urutan nilai input. Dengan demikian JU bersifat *memoryless*, dalam pengertian, tanggapan terhadap suatu input tidak tergantung dari output sebelumnya.

Jaringan berulang atau umpanbalik (JB) adalah jaringan yang arah sinyalnya dalam arah maju dan mundur yang berasal dari hubungan umpanbalik. Pada jaringan ini, paling tidak memiliki satu ikal. Dengan lintasan umpan balik adalah hasil output syaraf dipengaruhi oleh urutan nilai input dan output itu sendiri. Jaringan arsitektur JB ini bersifat dinamis serta dimaksudkan untuk memperoleh output yang stabil. Pada jaringan syaraf tiruan pemakaian arsitektur yang berbeda akan menghasilkan perilaku jaringan yang berbeda pula, dan masing-masing arsitektur membutuhkan algoritma belajar yang sesuai. Terdapat keterkaitan antara paradigma, aturan dan algoritma

belajar serta beberapa aplikasinya seperti yang tertera pada tabel 2.5.

Tabel 2.5 Algoritma belajar, arsitektur dan aplikasinya

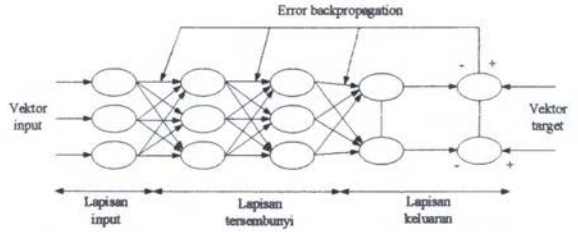
Paradigma	Aturan Belajar	Arsitektur	Algoritma Belajar	Aplikasi
Dengan Pengawasan	Koreksi Kesalahan	Jaringan Satu lapis	Algoritma perceptron	Klasifikasi pola
		Jaringan Multilapis	Backropagation Adeline dan Madaline	Aproksimasi fungsi Prediksi Kendali
	Kompetitif	Jaringan Kompetitif	Kuantisasi error	Katagorisasi Kompresi data
Tanpa Pengawasan	Koreksi Kesalahan	Jaringan Multilapis	Sammon's projection	Analisis Data
	Kompetitif	Jaringan Kompetitif	Kuantisasi error	Katagorisasi Kompresi data

Kedua paradigma, belajar dengan pengawasan dan tanpa pengawasan memakai aturan belajar yang berdasarkan koreksi kesalahan dan kompetitif. Aturan belajar koreksi kesalahan dapat digunakan untuk semua jenis arsitektur. Berdasarkan arsitekturnya terdiri dua macam yaitu jaringan satu lapis dan jaringan multilapis.

2.7.3 Algoritma Belajar Jaringan Syaraf Tiruan Tiga Lapis Umpan Maju

Salah satu arsitektur jaringan umpan maju adalah jaringan multilapis terdiri dari satu lapisan input (*input layers*), satu atau

lebih lapisan tersembunyi atau dalam (*hidden layer*), dan satu lapisan output (*output layers*). Susunan ini dikenal dengan jaringan umpan maju dengan tiga lapisan. Proses belajar multilapis ini menggunakan paradigma belajar dengan pengawasan dan belajar propagasi balik yang didasari atas aturan koreksi kesalahan (*error backpropagation*). Bentuk sederhana dari konsep jaringan tersebut diperlihatkan pada gambar 2.16, garis putus-putus menggambarkan suatu proses adaptasi terhadap bobot-bobot hubung. Prosedur belajar secara umum diterangkan sebagai berikut, presentasi pembelajaran meliputi pasangan input output vektor. Jaringan syaraf tiruan pertama kali menggunakan vektor input dibandingkan dengan vektor target. Jika tidak ada perbedaan (kesalahan) maka tidak ada perubahan adaptasi bobot, sebaliknya maka bobot hubung dirubah untuk mereduksi beda tersebut.



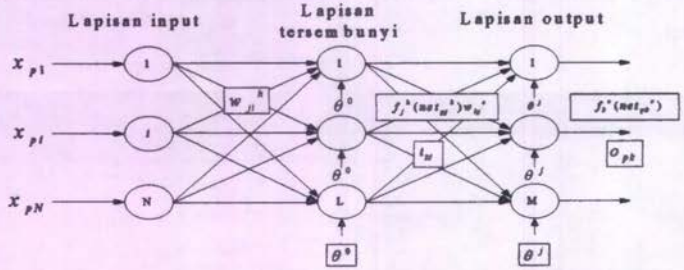
Gambar 2.16 Arsitektur JST feedforward Backpropagation (Rumelhart et al.,1985)

Algoritma *backpropagation* adalah suatu generalisasi dari metoda rata-rata kuadrat terkecil (*Least Mean Square, LMS*). Pada teori mengenai estimasi biasanya pencarian suatu parameter atau variabel yang optimal digunakan fungsi biaya atau indeks performansi yang diturunkan terhadap variabel atau parameter tersebut. Hal ini analog dengan mencari titik maksimum atau minimum fungsi. Selanjutnya fungsi biaya tersebut adalah total

dari *error* (kesalahan) antara output yang diinginkan (target jaringan syaraf tiruan) dan output yang sebenarnya atau output sistem yang diamati.

2.7.4 Algoritma Pembelajaran dengan *Error Backpropagation*

Sebuah jaringan syaraf tiruan dapat disebut sebagai jaringan pemetaan jika dapat menghitung hubungan secara fungsional antara input output. Pada fungsi jaringan syaraf tiruan sebagai pemetaan ini diharapkan jaringan syaraf tiruan dapat mengetahui hubungan fungsional walaupun tidak diketahuinya contoh pemetaan yang benar.



Gambar 2.17 Tiga lapis arsitektur Jaringan Syaraf metode belajar *Backpropagation* (Rumelhart et al.,1985)

Untuk memulai pemahaman tentang algoritma ini misalkan terdapat suatu psangan vektor input $x_p=(x_{p1}, x_{p2}, \ldots, x_{pN})$. Vektor input tersebut didistribusikan ke setiap lapisan tersembunyi dengan arah ke depan. Input ke jaringan syaraf tiruan pada unit tersembunyi adalah:

$$net_{pj}^{h} = \sum_{i=1}^{N} w_{ji}^{h} x_{pi} + \theta_j^h \qquad (2.42)$$

Bila diasumsikan bahwa fungsi aktivasi setiap untai adalah sama untuk seluruh jaringan input, maka output dari untai ini adalah

$$i_{pj} = f_j^h(net_{pj}^h) \tag{2.43}$$

Persamaan untuk untai output,

$$net_{pk}^o = \sum_{j=1}^{L} w_{kj}^o i_{pj} + \theta_k^o \tag{2.44}$$

$$o_{pk} = f_k^o(net_{pk}^o) \tag{2.45}$$

Tabel 2.6 Keterangan untuk Gambar 2.17

Notasi	Keterangan
h,o	*superscript* lapisan tersembunyi, output
$w_{ji}^{\ h}, w_{kj}^{\ o}$	bobot hubung dari unit input ke I, output j
$net_{pj}^{\ h}, net_{pk}^{\ o}$	input jaringan pada unit tersembunyi ke j, k
$f_j^h(net_{pj}^{\ h}), f_k^o(net_{pk}^{\ o})$	output jaringan pada unit tersembunyi ke j,k
$\theta_j^{\ h}, \theta_k^{\ o}$	bias pada lapisan tersembunyi ke j,k

Terdapat prosedur yang digunakan untuk algoritma ini, prosedur ini dipakai untuk melakukan komputasi dalam satu iterasi, adapun algoritmanya adalah sebagai berikut:

1. Tentukan vektor input pada jaringan syaraf tiruan dan hitung outputnya
2. Bandingkan output jaringan syaraf tiruan dengan output yang sebenarnya dan hitung kesalahannya (*error*)

3. Tentukan kearah mana perubahan tiap bobot sehubungan dengan pengurangan kesalahan
4. Tentukan seberapa besar perubahan untuk tiap bobot
5. Lakukan adaptasi untuk bobot-bobot tersebut
6. Ulangi no. 1 sampai 5 hingga semua vektor pembelajaran menghasilkan kesalahan yang sesuai

2.7.5 Adaptasi Bobot-Bobot pada Lapisan Output

Beranjak dari beragamnya input-input untuk setiap lapisan, maka satu kesalahan saja yang didapat tidaklah cukup untuk metoda *backpropagation* ini. Sehingga perlu didefinisikan sebuah *error* pada semua input yaitu $\delta_{pk} = (y_{pk} - O_{pk})$, (notasi ini yang disebut delta) dimana subskrip p menyatakan vektor pelatihan ke p dan k untuk output unit ke k. Sedangkan y_{pk} adalah harga output yang diinginkan dengan O_{pk} adalah output sebenarnya dari unit ke k. Selanjutnya jumlah dari kesalahan kuadrat dari semua unit output adalah:

$$E_p = \frac{1}{2}\sum_{k=1}^{M} \delta_{pk}^2 \qquad (2.46)$$

Faktor $\left(\frac{1}{2}\right)$ dalam persamaan 2.46 digunakan untuk kemudahan dalam melakukan deviasi persamaan kesalahan nantinya. Untuk menyatakan arah perubahan bobot-bobot, dihitung gradien negatif dari E_p, ∇E_p terhadap bobot w_{kj}. Untuk mempermudah pembahasan, substitusikan δ_{pk} ke persamaan 2.46 serta gunakan persamaan 2.42 sehingga menjadi

$$E_p = \frac{1}{2}\sum_{i} (y_{pk} - o_{pk})^2 \qquad (2.47)$$

dan untuk turunannya telah digunakan aturan berantai, sehingga dihasilkan persamaan 2.48 sebagai berikut:

$$\frac{\partial E_p}{\partial w_{kj}^o} = -\left(y_{pk} - o_{pk}\right)\frac{\partial f_{pk}^o}{\partial\left(net_{pk}^o\right)}\frac{\partial\left(net_{pk}^o\right)}{\partial w_{kj}^o} \tag{2.48}$$

Dengan memperhatikan persamaan 2.48 maka suku terakhir persamaan ini dapat ditulis:

$$-\frac{\partial\left(net_{pk}^o\right)}{\partial w_{kj}^o} = \left(\frac{\partial}{\partial w_{kj}^o}\sum_{j=1}^{L} w_{kj}^o i_{pj} + \theta_k^o\right) = i_{pj} \tag{2.49}$$

Bila persamaan 2.45 dan 2.48 digabungkan, didapat gradien negatif sebagai berikut:

$$-\frac{\partial E_p}{\partial w_{kj}^o} = \left(y_{pk} - o_{pk}\right) f_k^{o'}\left(net_{pk}^o\right) i_{pj} \tag{2.50}$$

Karena besar perubahan dari bobot lebih diperhatikan disini, maka 2.50 diambil berbanding lurus dengan gradien negatif. Sehingga bobot pada lapisan output diperbaharui dengan persamaan 2.51,

$$w_{kj}^0(t+1) = w_{kj}^o(t) + \Delta_p w_{kj}^o(t) \tag{2.51}$$

dimana,

$$\Delta_p w_{kj}^o(t) = \eta(y_{pk} - o_{pk}) f_{pk}^o(net_{pk}^o) i_{pj} \tag{2.52}$$

η disebut dengan parameter laju pembelajaran. Persamaan 2.52 adalah persamaan yang menyatakan perubahan vektor bobot pada lapisan output.

Misal ada dua bentuk fungsi aktivasi output sebagai berikut:

$$f_k^o(net_{jk}^o) = net_{jk}^o \tag{2.53}$$

$$f_k^o(net_{jk}^o) = (1+e^{-net_{jk}^o})^{-1} \tag{2.54}$$

dimana persamaan 2.53 adalah fungsi linier dan persamaan 2.54 adalah fungsi logistik. Selanjutnya turunan untuk masing-masing fungsi adalah $f_k^{o'} = 1$, sedangkan untuk persamaan kedua adalah $f_k^{o'} = f_k^o(1- f_k^o)=o_{pk}(1-o_{pk})$. Untuk kedua persamaan tersebut didapat iterasi untuk bobot adalah:

$$w_{kj}^o(t+1) = w_{kj}^o(t)+\eta(y_{pk}-o_{pk})i_{pj} \tag{2.55}$$

untuk output fungsi linier, dan

$$w_{kj}^o(t+1) = w_{kj}^o(t)+\eta(y_{pk}-o_{pk})o_{pk}(1-o_{pk})i_{pj} \tag{2.56}$$

untuk output sigmoid. Didefinisikan persamaan jumlahan untuk adaptasi bobot, yaitu:

$$\delta_{pk}^o = (y_{pk}-o_{pk})f_k^{o'}(net_{pk}^o) = \delta_{pk}f_k^{o'}(net_{pk}^o) \tag{2.57}$$

sehingga persamaan untuk adaptasi bobot dapat ditulis

$$w_{kj}^o(t+1) = w_{kj}^o(t)+\eta\delta_{pk}^o i_{pj} \tag{2.58}$$

2.7.6 Adaptasi Vektor Bobot pada Lapisan Tersembunyi

Prinsip dasar perubahan bobot pada lapisan tersembunyi pada dasarnya sama dengan yang dilakukan pada lapisan output.

Kesulitan yang timbul adalah menentukan kesalahan output lapisan tersembunyi.

Hal ini disebabkan karena pengetahuan yang didapat hanya pada output aktual akan tetapi output pada unit tersembunyi tidak didapat. Pemecahan masalah ini dimulai dengan memahami bahwa secara total kesalahan E_p adalah suatu persamaan yang menghubungkan output pada lapisan luar tersembunyi dengan inputnya. Artinya output lapisan tersembunyi adalah input bagi lapisan output. Pembahasan dimualai bersasar pada persamaan 2.59 untuk lapisan tersembunyi sebagai berikut:

$$\begin{aligned} E_p &= \frac{1}{2}\sum_k (y_{pk} - o_{pk})^2 \\ &= \frac{1}{2}\sum_k (y_{pk} - f_k^o(net_{pk}^o))^2 \\ &= \frac{1}{2}\sum_k \left(y_{pk} - f_k^o\left(\sum_j w_{kj}^o i_{pj} + \theta_k^o \right) \right)^2 \end{aligned} \qquad (2.59)$$

diketahui bahwa i_{pj} adalah bergantung pada bobot lapisan tersembunyi. Dengan menggunakan kedua persamaan tersebut sehingga dapat dilakukan pencarian gradien E_p terhadap bobot lapisan tersembunyi.

$$\begin{aligned} \frac{\partial E_p}{\partial w_{ji}^h} &= \frac{1}{2}\sum_k \frac{\partial}{\partial w_{ji}^h}\left(y_{pk} - o_{pk}\right)^2 \\ &= \frac{1}{2}\sum_k \left(y_{pk} - o_{pk}\right)^2 \frac{\partial o_{pk}}{\partial (net_{pj}^h)} \frac{\partial (net_{pk}^o)}{\partial i_{pj}} \frac{\partial i_{pj}}{\partial (net_{pj}^h)} \frac{\partial (net_{pj}^h)}{\partial w_{ji}^h} \end{aligned} \qquad (2.60)$$

dimana $\delta_{pk}^{o} = (y_{pk} - o_{pk})$. Selanjutnya diperlukan adaptasi perubahan bobot pada lapisan tersembunyi dan secara matematis dapat ditulis seperti rumus 2.61.

$$\Delta_p w_{ji}^{h} = \eta f_j^{h'}(net_{pj}^{h}) x_{pi} \sum_k (y_{pk} - o_{pk}) f_k^{o'}(net_{pk}^{o}) w_{kj}^{o} \quad (2.61)$$

Gunakan definisi δ_{pk}^{o} untuk persamaan 2.61, sehingga didapat:

$$\Delta_p w_{ji}^{h} = \eta f_j^{h'}(net_{pj}^{h}) x_{pi} \sum_k \delta_{pk}^{o} w_{kj}^{o} \quad (2.62)$$

Substitusi persamaan (2.59) dan (2.60) ke persamaan (2.62), sehingga didapat

$$\frac{\partial E_p}{\partial w_{ji}^{h}} = -\sum_k (y_{pk} - o_{pk}) f_k^{o'}(net_{pk}^{o}) w_{kj}^{o} f_j^{h'}(net_{pj}^{h}) x_{pi} \quad (2.63)$$

Perhatikan sekali lagi persamaan 2.63, bahwa setiap bobot pada lapisan tersembunyi diadaptasi bergantung pada semua bentuk kesalahan pada lapisan output δ_{pk}^{o}. Hasil ini menunjukkan terjadinya perambatan balik (*backpropagation*). Kesalahan lapisan output dirambatbalikkan ke lapisan tersembunyi untuk mendapatkan perubahan bobot yang sesuai pada lapisan tersebut. Dengan mendefinisikan sebuah bentuk kesalahan pada lapisan tersembunyi,

$$\delta_{pj}^{h} = f_j^{h'}(net_{pj}^{h}) \sum_k \delta_{pk}^{o} w_{kj}^{o} \quad (2.64)$$

akan memberi persamaan adaptasi menjadi analog dengan lapisan output sebagai berikut:

$$w_{ji}^{h}(t+1) = w_{ji}^{h}(t) + \eta \delta_{pj}^{h} x_i \quad (2.65)$$

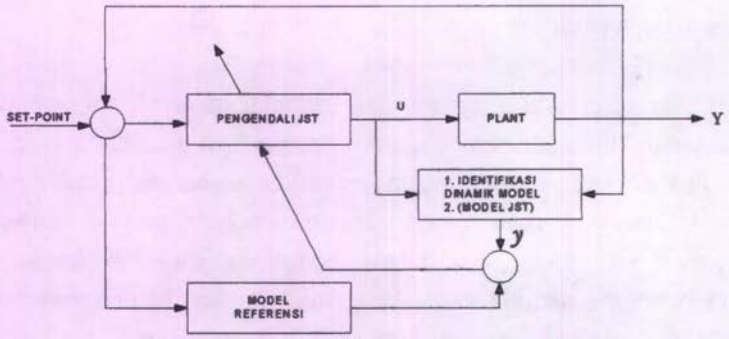
2.8 Jaringan Syaraf Tiruan pada Sistem Pengendalian

Jaringan syaraf tiruan yang digunakan untuk mengatasi kasus ketidaklinieran, pembelajaran, pemrosesan paralel memberi ide untuk alikasinya pada sistem pengendalian cerdas (*intelligent control*). Dari penelitian yang dilakukan oleh *Tai* diperoleh bahwa jaringan syaraf tiruan pada sistem kendali dapat diklasifikasikan dalam beberapa metoda yaitu: pengendalian dengan pengawasan (*supervised control*), *inverse control*, *adaptive control*, dan lain-lainnya. Pada penelitian ini hanya diterangkan skema jaringan syaraf tiruan untuk pengendalian yang hamir sama dengan skema pengendalian adaptif.

Struktur yang digambarkan pada gambar 2.18 sering digunakan untuk identifikasi dan pengendalian sistem. Beberapa referensi memberi nama struktur ini adalah *neural adaptive control* (NAC).

Pada NAC model referensi diletakkan paralel terhadap sistem yang sebenarnya. Pada blok identifikasi terdapat 2 kemungkinan berbeda yang digunakan untuk membedakan mekanisme yang terjadi. Identifikasi (nomor 1) yang dilakukan akan membentuk suatu model jaringan syaraf tiruan. Setelah identifikasi selesai, selanjutnya dilakukan mekanisme adaptasi pengendali (nomor 2).

Mekanisme ini hampir analog pada teori kontrol adaptif. Jadi diperlukan suatu model referensi yang biasanya model ini diambil dari model linier stabil. Model jaringan syaraf tiruan selanjutnya dibandingkan dengan model referensi.



Gambar 2.18 Jaringan Syaraf Tiruan digunakan sebagai Identifikasi dan pengendali

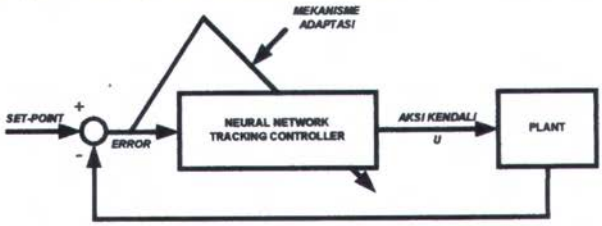
Kemudian kesalahan dirambatkan ke pengendali untuk merubah bobot pengendali. Dengan hanya melihat diagram blok NAC dapat dibayangkan, bahwa mekanisme pengendalian terdiri dari banyak langkah yang harus dilakukan. Dengan kata lain diperlukan waktu yang lebih lama apabila sistem adalah kompleks dan rentang data input-output besar sekali.

✓Jaringan Syaraf Tiruan sebagai *Tracking Control System*

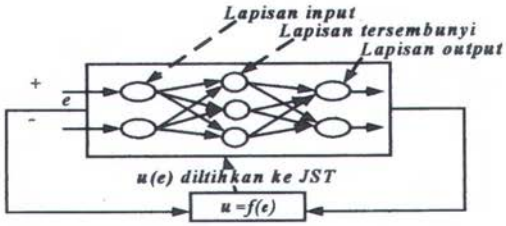
Suatu skema jaringan syaraf tiruan untuk pengendalian diperkenalkan oleh **Tai et al.**, seperti yang telah dijelaskan sebelumnya dan skema pengendali yang memakai jaringan syaraf tiruan diperlihatkan pada gambar 2.19 dan gambar 2.20 untuk pembelajarannya. Struktur dan skema serta mekanisme pembelajaran jaringan syaraf tiruan tersebut digunakan pada penelitian disini. Bentuk semacam ini sering juga disebut identifikasi dinamika balik langsung (*direct inverse dynamic identification*).

Neural Network (NN) atau Jaringan Syaraf Tiruan (JST) disini berfungsi sebagai *Clasifier* antara input JST dan output

JST, dimana input JST membutuhkan *pre-processing* dan selanjutnya output JST memberikan sinyal kontrol sehingga diharapkan output plant dapat dikendalikan sehingga mampu mecapai nilai sinyal masukan referensi (*set-point*) yang diberikan.



Gambar 2.19 Diagram Blok Aplikasi *Neural Network Tracking Controller* (Tai, Wang, Ashenayi, 1992)



Gambar 2.20 Konsep pembelajaran dengan pendekatan fungsi *u=f(error)*

Perhatikan sekali lagi gambar 2.19 dan gambar 2.20, dengan hanya memperhatikan skema tersebut terlihat kesederhanaan sistem ini dibanding dengan gambar 2.18. Konsep sistem ini beranjak dari pemahaman bahwa dalam sistem pengendalian bagaimanapun bentuk sistemnya, tujuan utama adalah menghasilkan sinyal atau aksi kendali (u) yang sesuai bila terjadi beda antara sinyal output sistem sebenarnya dengan sinyal referensi. Pelatihan hanya terdiri dari sinyal kesalahan (*error*, e) dan target aksi kendali (u) yang sesuai. Pelatihan semacam ini

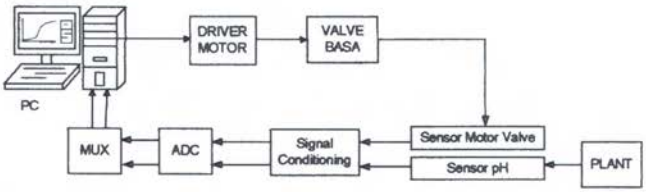
lebih dikenal dengan pemetaan input-output atau pendekatan suatu fungsi (*function approximation*) yaitu pembelajaran jaringan syaraf tiruan untuk mendekati *u=f(error)*. Perbedaan paling mendasar pada skema NAC adalah bahwa jaringan syaraf tiruan tidak lagi mempelajari dnamika balik sistem. Oleh karena itu tidak diperlukan waktu komputasi pelatihan yang lama. Kehandalan dijamin dari sistem ini karena perubahan pada sistem (*nonlinier*) akan memberikan kesalahan yang sama seperti yang telah dipelajari oleh pengendali, sehingga pengendali tidak perlu dilatih lagi bila terdapat gangguan atau perubahan set-point.

BAB III
PERANCANGAN PERANGKAT KERAS DAN PERANGKAT LUNAK

Pada bab ini dijelaskan tentang perancangan beserta implementasi perangkat keras dan perangkat lunak. Untuk perangkat keras terdiri dari miniplant sistem pengendalian pH, rangkaian pengkondisi sinyal, *Analog to Digital Converter* (ADC), Rangkaian eksternal paralel port, dan driver motor DC. Untuk Perangkat lunak digunakan bahasa pemrograman Delphi 7.0 dengan perangkat pendukung Prosessor AMD 1500 Mhz. Perancangan perangkat lunak ini terdiri dari perancangan alghoritma Jaringan Syaraf Tiruan yang meliputi training dan pengendalian.

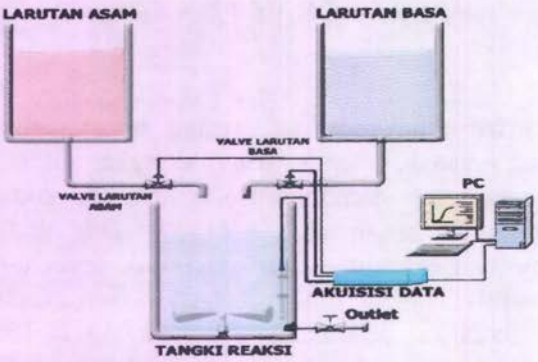
3.1 Perancangan Perangkat Keras

Perangkat keras yang dirancang dalam tugas akhir ini merupakan perangkat yang digunakan untuk sistem akuisisi data pada proses pengendalian. Berikut dijelaskan prinsip dari proses kontrol pada *miniplant* sistem pengendalian pH dan sistem akuisisi data. Perangkat keras sistem pengendalian pH dapat direpresentasikan dalam bentuk skema seperti pada gambar 3.1 dibawah ini:



Gambar 3.1 Skema Perangkat Keras *Miniplant* Sistem Pengendalian pH

Sedangkan desain *miniplant* sistem pengendalian pH dapat direpresentasikan seperti gambar 3.2.



Gambar 3.2 Miniplant Sistem Pengendalian pH

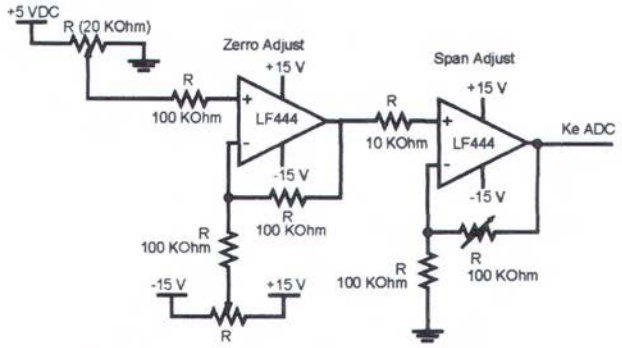
Tangki pencampur memiliki dua masukan aliran yaitu aliran larutan asam (HCL 0,1 M) dan aliran larutan basa (NaOH 0,1 M) dengan sistem semi batch. Dalam hal ini sensor diletakkan didalam tangki reaksi. Agar pencampuran antara larutan asam dan larutan basa bisa menghasilkan larutan yang homogen maka perlu ditambahkan pengaduk. Variabel yang dikendalikan adalah pH yang di dapat dari hasil pencampuran dua larutan tersebut. Sedangkan variabel yang dimanipulasi adalah besarnya aliran larutan basa. Sensor yang digunakan adalah pH elektrode dan sebagai aktuator digunakan motor valve.

3.1.1 Rangkaian Pengkondisi Sinyal

Sinyal dari *pH elektrode* adalah sinyal analog berupa tegangan - 282 sampai dengan 282 milivolt sehingga masih harus dikondisikan menjadi tegangan yang sesuai dengan range masukan ADC yaitu 0 – 5 volt. Untuk mengkondisikan tegangan tersebut maka dibutuhkan rangkaian pengkondisi sinyal yang

dapat diatur range keluarannya, rangkaian pengkondisi sinyal menggunakan op-amp.

Karena keluaran dari sensor pH dipengaruhi oleh suhu cairan yang kita ukur maka perlu kita buat kompensator suhu yang kita gabung dengan rangkaian pengkondisi sinyal dari sensor pH tersebut. Kompensator suhu kita rancang berada pada Range 25^0C - 40^0C. Rangkaian pengkondisi sinyal dapat dilihat pada gambar 3.3.



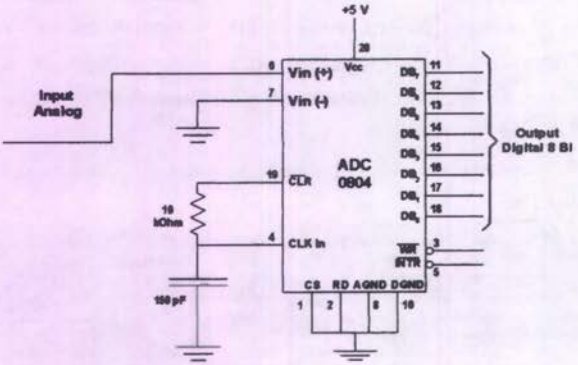
Gambar 3.3 Rangkaian pengkondisi sinyal

Op-amp yang digunakan dalam tugas akhir ini merupakan Integrated Circuit (IC) keluaran National Semiconductor dengan seri LF444. Nilai minimum keluaran sensor pH elektrode dikondisikan sehingga memberikan keluaran tegangan 0 volt dan nilai maksimum sensor dikondisikan menjadi tegangan 5 volt untuk bisa dibaca oleh ADC. Proses pengesetan ini dapat dilakukan dengan memutar dua *variable resistor* sebagai *span* dan *zero*.

3.1.2 Analog To Digital Converter (ADC)

Analog to digital converter digunakan untuk mengkonversi sinyal analog yang berasal dari sensor pH

elektrode menjadi sinyal digital sehingga dapat diolah oleh kontroler. Dalam tugas akhir ini digunakan ADC 0804 produksi *National Semiconductor* yang dapat merubah sinyal analog menjadi sinyal digital delapan bit.



Gambar 3.4 Rangkaian ADC (Natioanal Semiconductor, 1999)

Rangkaian ADC ditunjukkan seperti pada gambar 3.4. ADC 0804 mempunyai waktu konversi 103-114 μs. Masukan ADC berasal dari pengkondisi sinyal null-span bernilai antara 0 sampai 5 Volt. $V_{in}(-)$ dihubungkan ke ground dan keluaran unit penguat dihubungkan pada $V_{in}(+)$. Kaki tegangan referensi dibiarkan terbuka sehingga besarnya nilai tegangan referensi adalah sama dengan tegangan *supply* (V_{CC}) yaitu sebesar 5 volt. Clock dari ADC memakai pembangkit internal, sehingga hanya diperlukan sebuah hambatan dan kapasitor yang dihubungkan pada jalur CLK-R dan CLK-IN. Dengan frekuensi yang disarankan f = 640 kHz, dengan nilai kapasitor C = 150 pF nilai R dapat dicari dari persamaan :

$$f = \frac{1}{1{,}1\,R\,C} \tag{3.1}$$

sehingga didapatkan nilai R = 9,5 kΩ. Dengan menyesuaikan nilai yang ada dipasaran diperoleh R =10 kΩ. Untuk memulai konversi

jalur CS dan WR harus diberi logika 0, selesainya konversi ditandai dengan logika 0 pada jalur INTR. Apabila jalur INTR berlogika 0 maka data siap dibaca. Untuk membaca data jalur CS dan RD harus berlogika nol. Resolusi konversi dari ADC ini adalah 5/256 Volt/bit, artinya setiap perubahan tegangan sebesar 19.53125 milivolt akan merubah sebesar satu bit biner.

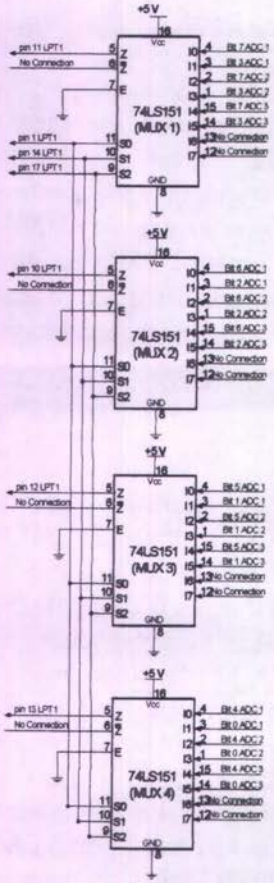
3.1.3 Rangkaian Eksternal Paralel Port

Rangkaian eksternal paralel port adalah rangkaian multiplekser yang digunakan untuk mengatur jalur input. Jumlah port yang berfungsi sebagai input pada paralel port hanya 5 pin sehingga untuk mengatasi input yang lebih dari lima bit maka dibutuhkan rangkaian eksternal. Dalam tugas akhir ini membutuhkan input berjumlah 24 bit yaitu berasal dari tiga ADC 8 bit. Prinsip yang digunakan pada dasarnya menggunakan metode *nibble mode* dengan sedikit modifikasi.

$$Z = \overline{E} \cdot (I_0 \cdot \overline{S}_0.\overline{S}_1.\overline{S_2} + I_1 \cdot S_0.\overline{S}_1.\overline{S_2} + I_2 \cdot \overline{S}_0.S_1.\overline{S_2} + I_3 \cdot S_0.S_1.\overline{S_2}$$
$$+ I_4 \cdot \overline{S}_0.\overline{S}_1.S_2 + I_5 \cdot S_0.\overline{S}_1.S_2 + I_6 \cdot \overline{S}_0.S_1.S_2 + I_7 \cdot S_0.S_1.S_2) \qquad (3.2)$$

Rangkaian eksternal yang digunakan seperti pada gambar 3.5 menggunakan empat buah multiplekser 74LS151 yang di drive secara bersamaan pada selektor S_0, S_1 dan S_2. Jika selektor S_0, S_1 dan S_2 diberikan logika 0 maka jalur input yang digunakan adalah jalur I_0 sehingga yang akan disambungkan ke paralel port adalah bit 4 sampai bit 7 ADC1, jika S_0 diberi logika 1 sedangkan S_1 dan S_2 diberi logika 0 maka jalur input yang digunakan adalah jalur I_1 sehingga yang akan disambungkan ke paralel port adalah bit 0 sampai bit 3. Jika S_0 dan S_2 diberi logika 0 sedangkan S_1 diberi logika 1 maka jalur input yang digunakan adalah jalur I_2 sehingga yang akan disambungkan ke paralel port adalah bit 4 sampai bit 7 ADC2, jika S_0 dan S_1 diberi logika 1 sedangkan S_2 diberi logika 0 maka jalur input yang digunakan adalah jalur I_3 sehingga yang

akan disambungkan ke paralel port adalah bit 0 sampai bit 3 ADC2. Jika S_0 dan S_1 diberi logika 0 sedangkan S_2 diberi logika 1 maka jalur input yang digunakan adalah jalur I_4 sehingga yang akan disambungkan ke paralel port adalah bit 4 sampai bit 7 ADC3. Jika S_0 dan S_2 diberi logika 1 sedangkan S_1 diberi logika 0 maka jalur input yang digunakan adalah jalur I_5 sehingga yang akan disambungkan ke paralel port adalah bit 0 sampai bit 3 ADC3.



Gambar 3.5 Rangkaian eksternal paralel port (Motorola, 1998)

Berikut ini adalah tabel kebenaran Multiplekser 74LS151.

Tabel 3.1. Tabel Kebenaran Multiplekser 74LS151 (Motorola, 1998)

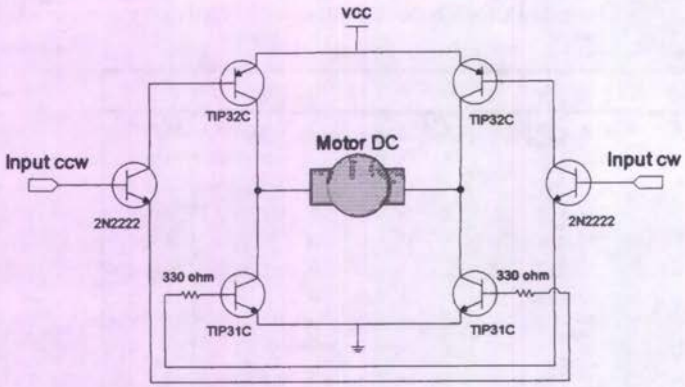
$\overline{E}$	S_2	S_1	S_0	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7	$\overline{Z}$	Z
H	X	X	X	X	X	X	X	X	X	X	X	H	L
L	L	L	L	L	X	X	X	X	X	X	X	H	L
L	L	L	L	H	X	X	X	X	X	X	X	L	H
L	L	L	H	X	L	X	X	X	X	X	X	H	L
L	L	L	H	X	H	X	X	X	X	X	X	L	H
L	L	H	L	X	X	L	X	X	X	X	X	H	L
L	L	H	L	X	X	H	X	X	X	X	X	L	H
L	L	H	H	X	X	X	L	X	X	X	X	H	L
L	L	H	H	X	X	X	H	X	X	X	X	L	H
L	H	L	L	X	X	X	X	L	X	X	X	H	L
L	H	L	L	X	X	X	X	H	X	X	X	L	H
L	H	L	H	X	X	X	X	X	L	X	X	H	L
L	H	L	H	X	X	X	X	X	H	X	X	L	H
L	H	H	L	X	X	X	X	X	X	L	X	H	L
L	H	H	L	X	X	X	X	X	X	H	X	L	H
L	H	H	H	X	X	X	X	X	X	X	L	H	L
L	H	H	H	X	X	X	X	X	X	X	H	L	H

H = HIGH Voltage Level
L = LOW Voltage Level
X = Don't Care

3.1.4 Driver Motor

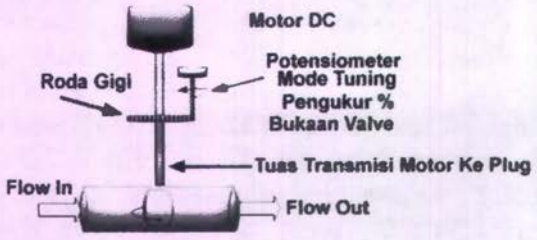
Driver motor digunakan untuk memanipulasi gerakan motor yang digunakan untuk membuka dan menutup kontrol valve pada aliran fluida panas. Jenis motor yang digunakan adalah motor DC 22 volt sehingga dibutuhkan driver motor DC dua arah dan sensor posisi bukaan valve. Sensor posisi menggunakan potensiometer, manfaat sensor posisi disini adalah digunakan untuk mengetahui persentase bukaan valve, selain itu juga mempermudah dalam proses pemberian sinyal kontrol pada motor valve. Rangkaian driver motor menggunakan 4 buah

transistor npn yaitu tipe TIP31C dan 2N2222 serta 2 buah transistor pnp tipe TIP32C yang difungsikan sebagai saklar.



Gambar 3.6 Rangkaian Driver motor

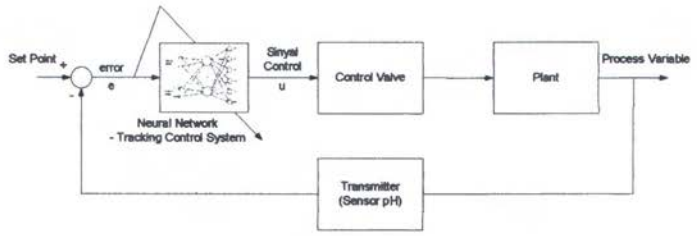
Prinsip kerja dari rangkaian driver motor adalah dengan merubah arah aliran arus yang melewati motor yakni dengan cara memberikan sinyal (logika 1) pada input cw sedangkan input ccw tidak diberi sinyal (logika 0) maka gerak motor akan searah jarum jam dan sebaliknya jika input cw tidak diberi sinyal (logika 0) sedangkan input ccw diberi sinyal (logika 1) maka gerak motor akan berlawanan arah jarum jam. Pada Gambar 3.7 ini dapat dilihat desain control valve yang digunakan pada *miniplant* pengendalian pH.



Gambar 3.7 Control Valve miniplant pengendalian pH

3.2 Perancangan Perangkat Lunak

Perangkat lunak yang dirancang adalah meliputi perangkat lunak yang berfungsi sebagai kontroler, training, dan unit input dan output yang dapat diintegrasikan dengan perangkat keras sehingga dapat diimplementasikan secara online. Blok diagram sistem secara keseluruhan adalah pada gambar 3.8 berikut:

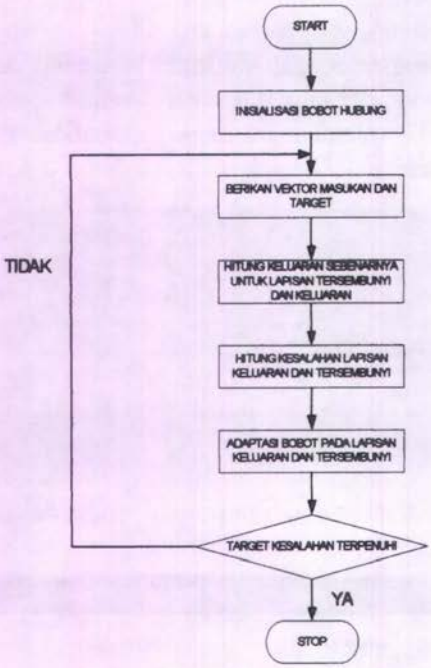


Gambar 3.8 Blok diagram sistem Pengendalian pH.

Perangkat lunak yang dirancang pertama kali adalah proses pembelajaran menggunakan jaringan syaraf tiruan dengan algoritma pembelajaran *backpropagation*, kemudian dilanjutkan dengan perancangan sistem kontrol dengan menggunakan jaringan syaraf tiruan secara online.

3.2.1 Pelatihan JST Backpropagation

Struktur jaringan syaraf tiruan yang digunakan dalam model proses sistem ini adalah multilayer perceptron yang terdiri dari tiga lapis/layer yaitu lapis input, lapis hidden dengan fungsi aktivasi bipolar sigmoid dan lapis output dengan fungsi aktivasi bipolar sigmoid. Metode training atau pelatihan yang digunakan adalah *Backpropagation*. Proses pelatihan menggunakan 82 pasang data input dan output. Tingkat kesesuaian dari model dapat diukur menggunakan RMSE (*Root Mean Square Error*). Di bawah ini adalah gambar *flowchart* proses pelatihan

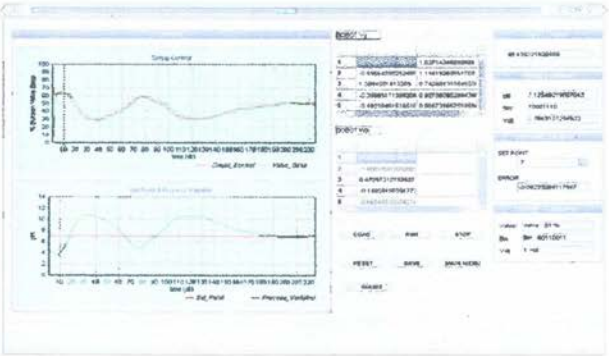


Gambar 3.9. *Flowchart* proses pelatihan (Laurence, 1994)

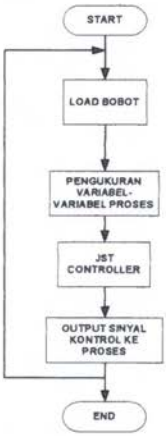
3.2.2 Sistem Pengendalian

Perangkat lunak sistem pengendalian pH dirancang dengan menggunakan bahasa pemrograman delphi 7 dan sistem operasi yang kompatibel adalah windows 98, ME, 2000, dan Windows XP Profesional. Perangkat lunak yang dirancang terdiri dari beberapa komponen utama yaitu komponen input/output, motor positioner, user interface, dan kontroler. Komponen input/output berfungsi untuk mengambil dan mengirim data dari komputer ke plant atupun sebaliknya, motor positioner berfungsi untuk mengendalikan posisi motor sehingga dapat melakukan aksi yang

dapat membuka dan menutup kontrol valve, user interface adalah tampilan yang berhadapan langsung dengan operator sehingga dapat merubah parameter seperti setpoint, learning rate, bobot jaringan ataupun sampling time serta melihat respon proses dalam bentuk grafik dan nilai-nilai angka. kode program input/output, motor positioner, dan kontroler diberikan dalam lampiran. Flowchart program kontroler terdapat pada gambar 3.11. User interface atau tampilan program ditunjukkan pada gambar dibawah ini :



Gambar 3.10 Tampilan Perangkat lunak Sistem Pengendalian pH



Gambar 3.11 Flowchart program kontroler

BAB IV
PENGUJIAN DAN ANALISA

4.1 Pengujian Perangkat keras

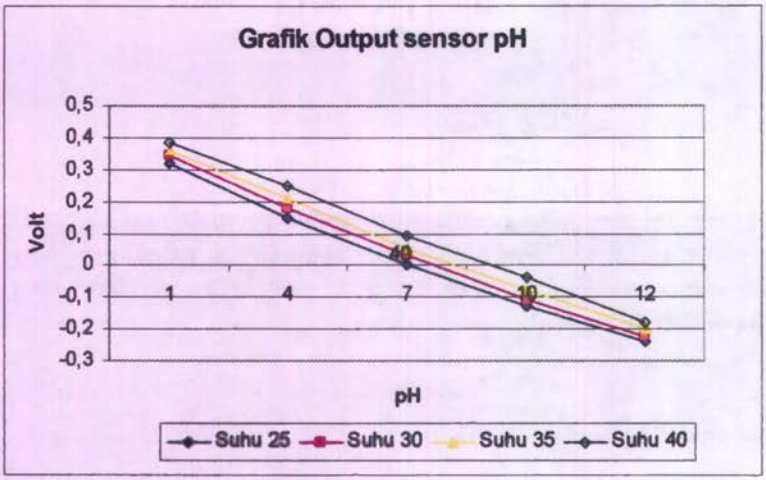
Pengujian ini dilakukan terhadap perangkat keras yang meliputi sensor pH, pengkondisi sinyal, Analog to Digital konverter (ADC), Driver motor beserta kontrol valve. Sebelum dilakukan pengujian pada semua perangkat keras tersebut maka terlebih dahulu dilakukan pengukuran.

4.1.1 Pengukuran Sensor pH

Pengujian ini dilakukan dengan menggunakan cairan buffer dengan nilai pH 1, 4, 7, 10 dan pH 12. Selain itu kita juga merubah suhu dari cairan tersebut untuk mendapatkan karakteristik sensor pH terhadap perubahan suhu. Range perubahan suhu yang kita gunakan adalah **25^{0}C - 40^{0}C**. Sensor pH kita masukkan dalam cairan buffer kemudian kita ukur tegangan output dari sensor tersebut. Hasil dari pengukuran dapat dilihat pada tabel 4.1 dan grafik dapat dilihat pada gambar 4.1. Dari data itu terlihat bahwa output sensor pH cukup linier untuk perubahan suhu cairan dari range **25^{0}C - 40^{0}C.**

Tabel 4.1 Hasil Pengukuran Output Sensor pH.

pH	Output Sensor pH (Volt)			
	25^{0}C	30^{0}C	35^{0}C	40^{0}C
1	0,32	0,344	0,36	0,386
4	0,148	0,18	0,21	0,25
7	0,001	0,03	0,05	0,09
10	-0,13	-0,11	-0,08	-0,04
12	-0,242	-0,224	-0,2	-0,178



Gambar 4.1 Grafik Output Sensor pH.

Linieritas masing-masing tegangan output sensor pH adalah sebagai berikut:
Jika sensor pH diukur pada suhu 25^0C, maka didapatkan persamaan matematis

$$y = -0.1402x + 0{,}44 \quad (4.1)$$

Jika sensor pH diukur pada suhu 30^0C, maka didapatkan persamaan matematis

$$y = -0.1426x + 0.4718 \quad (4.2)$$

Jika sensor pH diukur pada suhu 35^0C, maka didapatkan persamaan matematis

$$y = -0.141x + 0.491 \quad (4.3)$$

Jika sensor pH diukur pada suhu 40^0C, maka didapatkan persamaan matematis

$$y = -0.1418x + 0.527 \quad (4.4)$$

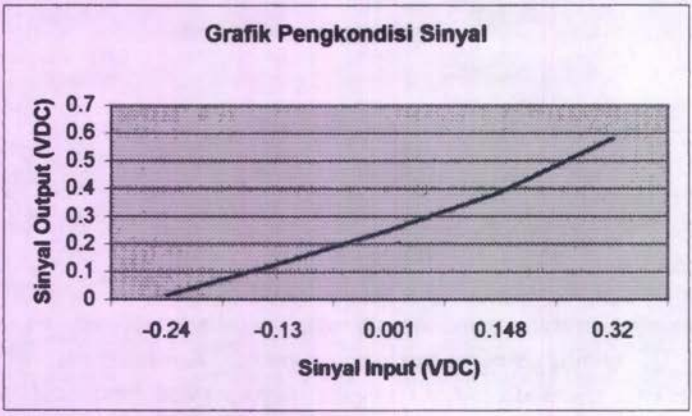
4.1.2 Pengukuran dan Pengujian Rangkaian Pengkondisi Sinyal

Pengukuran rangkaian pengkondisi sinyal dilakukan dengan memberikan sinyal masukan yang berupa tegangan DC dari *regulator* untuk mendapatkan harga span dan zero yang diinginkan. Nilai range output dari sensor pH adalah - 282 sampai dengan +282 milivolt, sedangkan nilai masukan ADC adalah 0 – 5 volt DC sehingga perlu adanya penguatan. Kalibrasi dilakukan dengan cara merubah harga resistansi pada *variabel resistor* zero dan span pada rangkaian pengkondisi sinyal untuk mendapatkan nilai tegangan yang ditentukan berdasar pada range sinyal masukan yang telah ada, yakni berkisar antara 0 – 5 Volt DC. Setelah proses kalibrasi dilakukan, maka proses pengujian dilakukan dengan mengambil 6 titik daerah pengujian dan melakukan pengujian sebanyak 10 kali dari pengambilan data. Sehingga didapatkan hasil pengujian seperti ditunjukkan pada tabel 4.2 dan gambar 4.2.

Tabel 4.2 hasil pengujian dan kalibrasi pengkondisi sinyal

pH	Sinyal input (Volt DC)	Teg. konversi (Volt DC)	Sinyal Output (Volt DC)	Error
12	-0,24	0,000	0,015	0,015
10	-0,13	0,110	0,13	0,020
7	0,001	0,240	0,25	0,010
4	0,148	0,368	0,388	0,010
1	0,32	0,560	0,580	0,020
Error rata-rata				0,015

Didapatkan grafik input terhadap output pengkondisi sinyal seperti pada gambar 4.2.



Gambar 4.2 Grafik kelinieritasan pengkondisi sinyal

Dari pengujian dan kalibrasi pengkondisi sinyal didapatkan nilai error pengujian rata-rata sebesar 0,015. Nilai ini didapatkan dengan cara membandingkan antara sinyal keluaran rangkaian pengkondisi sinyal yang sebenarnya dengan hasil perhitungan secara matematis. Berdasarkan tabel 4.2 didapatkan persamaan matematis yang menyatakan hubungan antara sinyal input dan sinyal output rangkaian pengkondisi sinyal seperti dinyatakan dalam persamaan 4.5.

$$y = 0.1388\,x - 0.1438 \qquad (4.5)$$

dengan,

y = sinyal output rangkaian pengkondisi sinyal (Volt DC).

x = sinyal input rangkaian pengkondisi sinyal (Volt DC).

4.1.3 Pengukuran dan Pengujian ADC

Keluaran dari ADC adalah biner 8-bit dengan nilai desimal 0-255 maka konversi dari keluaran ADC ke tegangan adalah :

$$V_{out} = desimal \times \frac{V_{ref}}{256} \qquad (4.6)$$

Pengujian ini dilakukan dengan memberikan sinyal inputan pada ADC berupa tegangan analog DC mulai dari 0 volt sampai dengan 5 volt, kemudian dilakukan pengamatan hasil konversi dengan bantuan perangkat lunak untuk mendapatkan informasi yang diperlukan. Selanjutnya hasil pengukuran dan pengkonversian dari perangkat lunak dibandingkan dengan harga sebenarnya (sinyal inputan). Data diambil 5 sample dengan perubahan kenaikan sinyal tegangan inputan sebesar 0,5 Volt, dan pengujian dilakukan 10 kali sehingga diperoleh 100 data.

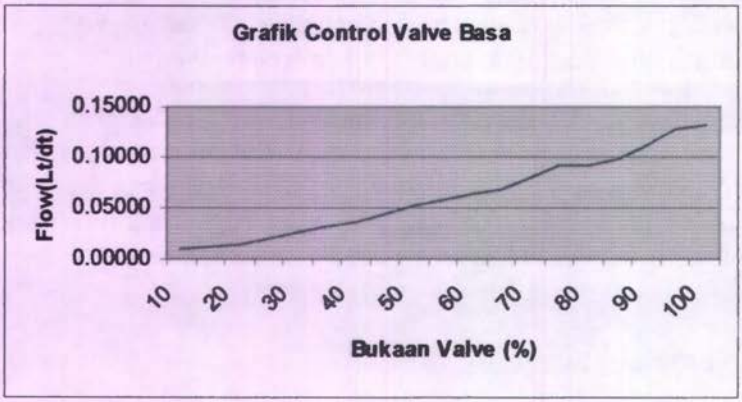
Dari hasil pengujian dan kalibrasi ADC didapatkan nilai error rata-rata sebesar 9,075 mV. Sesuai dengan data sheet dari ADC bahwa maksimum kesalahan pengkonversian yang diijinkan adalah ±1 LSB atau ±19,53125 mV, maka ADC yang dirancang mempunyai kesalahan konversi yang masih bisa ditolerir.

4.1.4 Pengujian Control Valve

Pengujian ini dilakukan dengan cara memberikan output bukaan valve 0 - 100 % dari komputer dan mencatat besarnya kecepatan aliran larutan basa yang mengalir dari Valve . Hasil pengujian Control Valve ditunjukkan dalam tabel 4.3, yakni dilakukan pengujian dengan setiap kenaikan persen bukaan valve sebesar 5%.

Tabel 4.3. Hasil pengukuran bukaan valve terhadap flow

No.	Bukaan Valve Basa (%)	Flow
1	10	0.05000
2	15	0.03333
3	20	0.02500
4	25	0.02000
5	30	0.01667
6	35	0.01429
7	40	0.01250
8	45	0.01111
9	50	0.01000
10	55	0.00909
11	60	0.00833
12	65	0.00769
13	70	0.00714
14	75	0.00667
15	80	0.00625
16	85	0.00588
17	90	0.00556
18	95	0.00526
19	100	0.00500



Gambar 4.3 Grafik Respon Laju Aliran Terhadap Bukaan Valve

Berdasarkan data pada tabel 4.3 dan grafik 4.3 maka didapatkan hubungan matematik antara persen bukaan valve dan laju aliran (*flow*) larutan basa, yakni $y = 0.0002x^2 + 0.0034x + 0.0047$ dengan x adalah persen bukaan control valve untuk larutan basa dan y adalah besar laju aliran larutan basa (liter/detik).

4.1.5 Pengujian Rangkaian eksternal port paralel

Pengujian ini dilakukan secara keseluruhan dengan memberikan input digital dan mencatat hasil pembacaan oleh perangkat lunak pendukung. Sinyal input digital yang diberikan berasal dari output ADC.

Tabel 4.4 Hasil pengujian rangkaian eksternal paralel port.

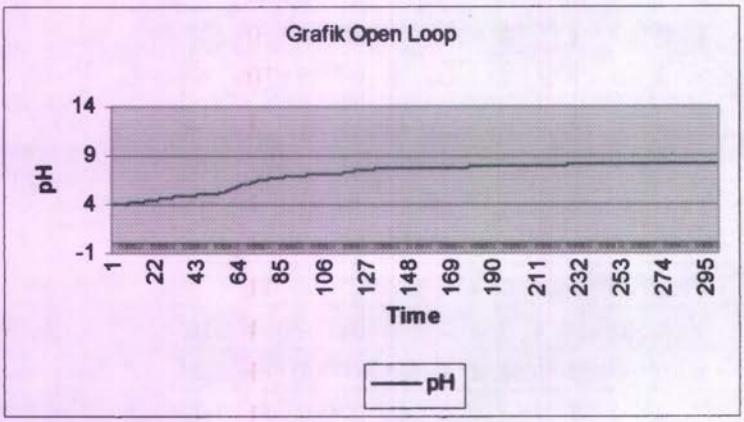
	Perangkat Keras									Perangkat Lunak	
	Tegangan kaki ADC (volt)										
No	0	1	2	3	4	5	6	7	Des	Biner	Des
1	0	0	0	0	0	0	0	0	0	00000000	0
2	4	4	4	4	0	0	0	0	15	11110000	15
3	0	4	4	4	4	0	0	0	30	01111000	30
4	4	0	4	4	0	4	0	0	45	10110100	45
5	0	0	4	4	4	4	0	0	60	00111100	60
6	4	4	0	4	0	0	4	0	75	11010010	75
7	0	4	0	4	4	0	4	0	90	01011010	90
8	4	0	0	4	0	4	4	0	105	10010110	105
9	0	0	0	4	4	4	4	0	120	00011110	120
10	4	4	4	0	0	0	0	4	135	11100001	135
11	0	4	4	0	4	0	0	4	150	01101001	150
12	4	0	4	0	0	4	0	4	165	10100101	165
13	0	0	4	0	4	4	0	4	180	00101101	180
14	4	4	0	0	0	0	4	4	195	11000011	195
15	0	4	0	0	4	0	4	4	210	01001011	210
16	4	0	0	0	0	4	4	4	225	10000111	225
17	0	0	0	0	4	4	4	4	240	00001111	240
18	4	4	4	4	4	4	4	4	255	11111111	255

Nilai digital yang berasal dari ADC diukur dengan menggunakan voltmeter digital kemudian dilakukan pembacaan data oleh perangkat lunak untuk dibandingkan nilai yang terukur dalam bentuk desimal. Dari tabel diatas dapat disimpulkan bahwa rangkaian eksternal tersebut memenuhi syarat untuk digunakan baik untuk ADC1, ADC2 maupun ADC3 yang ditunjukkan dengan nilai desimal maupun biner pada perangkat lunak *(warna merah adalah MSB)* sama dengan nilai di perangkat keras.

4.2 Uji Open Loop

Uji open loop dilakukan untuk mengetaui karaktersistik plant yang sebenarnya. Gambar 4.4 adalah grafik hasil uji open loop. Dari grafik pada gambar 4.4 didapatkan fungsi transfer *miniplant* sistem pengendalian pH sebagai berikut:

$$\frac{e^{-t_0 s}}{1+\tau s}$$



Gambar 4.4 Grafik Uji Open Loop

t_0 adalah *dead time* atau waktu yang diperlukan sebelum grafik mulai naik yakni 8 detik, sedangkan τ adalah 62,3% dari waktu yang diperlukan untuk mencapai keadaan *steady*, yakni sebesar 150,143. Sehingga didapatkan fungsi transfer *miniplant* sistem pengendalian pH adalah $\frac{e^{-8s}}{1+150{,}43s}$. Dari fungsi transfer tersebut dapat diketahui bahwa karakteristik plant adalah non-linier.

4.3 Parameter JST

Sebelum menentukan parameter-parameter jaringan syaraf tiruan yang akan digunakan sebagai pengendali, maka dilakukan proses pelatihan dan proses validasi terlebih dahulu. Dalam proses pelatihan digunakan 82 pasang data input (*error*) dan data target (*sinyal kendali*), data untuk proses pelatihan disertakan dalam Lampiran. Untuk memperoleh parameter jaringan syaraf tiruan yang digunakan sebagai pengendali maka dilakukan pelatihan terhadap jaringan syaraf tiruan dengan cara mengubah variasi jumlah *node hiddden*, nilai *learning rate* (α), dan nilai *momentum* (μ), hingga didapatkan parameter jaringan syaraf tiruan terbaik dengan melihat tingkat kesesuaian model saat diuji atau divalidasi.

Jaringan syaraf tiruan yang digunakan dalam proses pelatihan memiliki paramater-parameter sebagai berikut:

a. Jumlah *layer*.
   JST yang digunakan jenis *Backpropagation* dengan 3 *layer* yaitu *layer input*, *layer hidden* dan *layer output*.
b. Jumlah unit tiap *layer*.
   Unit *input* : 1 unit (diambil dari data close loop yang berupa normalisasi sinyal error).
   Unit *hidden* : 30 unit.

Unit *output* : 1 unit (diambil dari data close loop yang berupa persentase bukaan valve basa dan eror).
Pada *layer input* dan *hidden* masing-masing ditambahkan 1 unit *bias*.

c. Inisialisasi bobot.
Bobot diinisialisasikan secara *random* antara –0.5 sampai dengan +0.5.

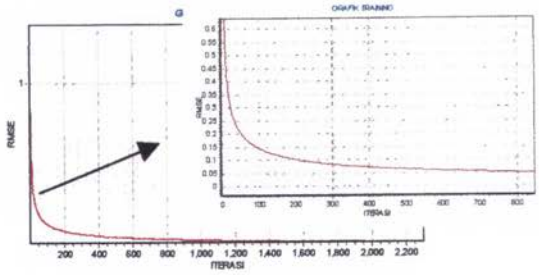
d. Fungsi aktivasi.
Fungsi aktivasi yang digunakan adalah fungsi *Tangen Hiperbolic* pada *layer hidden*, dan fungsi *Binary Sigmoid* pada *layer output*, hal ini disebabkan karena karakteristik sistem pengendalian pH yang diperoleh dari data close loop bahwa nilai input (*error*) berada pada *range* – 1 sampai dengan +1 sedangkan nilai target (*persentase bukaan valve*) berada pada range 0 sampai dengan +1.

e. Nilai *learning rate*.
Nilai *learning rate* adalah parameter penentu dalam kecepatan belajar jaringan *backpropagation*. Berdasar pada data hasil pembelajaran maka didapatkan nilai 0,1 sebagai nilai *learning rate* yang terbaik.

f. Nilai koefisien momentum.
Koefisien ini digunakan sebagai alternatif dalam *up-date* bobot. Berdasar pada data hasil pembelajaran maka didapatkan nilai 0,7 sebagai nilai *momentum* terbaik.

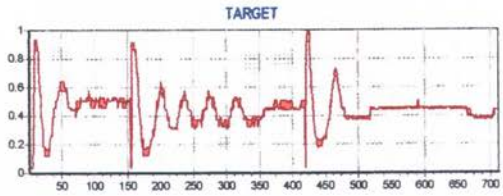
g. Data pelatihan.
Data pelatihan yang digunakan adalah sebanyak 82 pasangan data antara data input model JST (*error*) dan data target model JST (persentase bukaan valve basa) yang didapat dari proses close loop, data training dapat dilihat pada lampiran E.

Tingkat kesesuaian dari model yang didapat diukur dengan menggunakan RMSE (*Root Mean Square* Error). Dari grafik 4.5 dapat dilihat bahwa nilai RMSE cenderung turun, dan hingga iterasi ke-2298 nilai RMSE berada pada nilai 0,03.

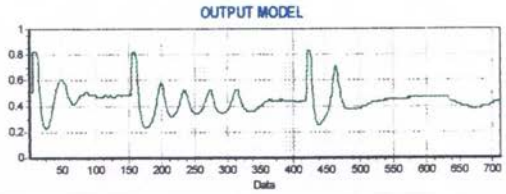


Gambar 4.5 Grafik hasil pembelajaran JST

Namun jika dibandingkan dengan data-data hasil pembelajaran yang mempunyai parameter berbeda dari parameter yang telah dijelaskan sebelumnya, maka bisa dikatakan bahwa parameter JST dengan 1 node input, 30 node hidden, 1 node output, nilai *learning rate* 0,1 dan nilai *momentum* 0,7 adalah yang terbaik, hal ini bisa dilihat pada gambar 4.7, yakni grafik validasi model yang mampu mengikuti pola target yang ditunjukkan pada gambar 4.6. Adapun nilai bobot yang telah diperoleh dari proses pembelajaran dapat dilihat pada tabel 4.5.



Gambar 4.6 Grafik Data Target untuk Validasi model JST



Gambar 4.7 Grafik Output Model hasil Validasi model JST

Tabel 4.5 Nilai bobot dan bias hasil pembelajaran JST

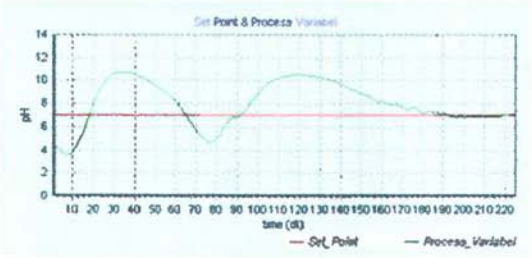
No.	Bobot Vij Akhir	Bobot Bias Input	Bobot Wjk Akhir	Bobot Bias Hidden
1	0.650353	0.5076035	0.4410342	0.988662
2	-2.1915624	0.6875061	-0.5267129	
3	-0.0242539	0.6872199	-0.4316244	
4	-0.1822773	0.7037132	-0.319351	
5	-0.2499485	0.4914231	-0.0354929	
6	0.4043673	0.423951	0.1519386	
7	-0.0384671	0.593379	-0.3005119	
8	-0.6253	0.4921476	0.3045256	
9	0.9530167	0.7502248	0.8193318	
10	0.4043673	0.423951	0.1519386	
11	0.3739283	0.4113098	0.1508878	
12	-0.0546753	0.6968406	-0.4296577	
13	0.0778177	0.5555966	-0.2223233	
14	-0.0823903	0.4586663	-0.0515141	
15	-0.0450603	0.5195819	-0.1886173	
16	-0.6896554	0.526383	0.287283	
17	-0.765608	0.493143	0.356877	
18	0.0368873	0.4982915	-0.1558715	
19	-0.2862805	0.5413283	-0.0758266	
20	0.7154388	0.5578568	0.4526477	
21	0.0065834	0.4795114	-0.1243507	
22	-0.0888443	0.6525419	-0.3516968	
23	0.1897033	0.4882928	-0.0794317	
24	-2.1915624	0.6875061	-0.5267129	
25	-0.4054468	0.537339	0.022229	
26	-0.0501733	0.6433939	-0.3643945	
27	0.1463921	0.4472068	-0.043796	
28	-0.5447267	0.5306249	0.1453042	
29	0.0368873	0.4982915	-0.1558715	
30	-0.0242539	0.6872199	-0.4316244	

Data yang digunakan untuk proses validasi sebanyak 712 pasangan antara data input (*error*) dan data target (*persentase bukaan valve basa*). Dari proses validasi model JST diperoleh nilai RMSE sebesar 0,07574.

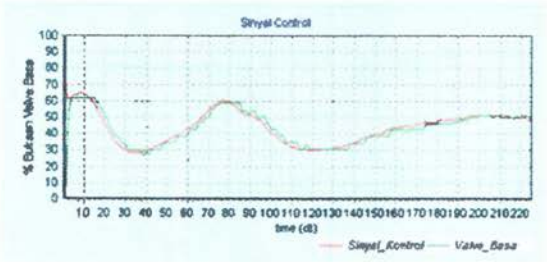
4.4 Sistem Pengendalian pH dengan Menggunakan Jaringan Syaraf Tiruan

Setelah didapatkan parameter yang paling optimal selama proses pelatihan maka parameter-parameter tersebut digunakan sebagai parameter pada proses pengendalian secara online.

Grafik 4.8 dan grafik 4.10 menunjukkan performansi sistem kontrol berbasis jaringan syaraf tiruan seraca on line pada miniplant sistem pengendalian pH. Pada grafik 4.8 dapat dilihat bahwa pada set point pH sama dengan 7, sistem kontrol yang dikembangkan mampu memberikan performansi yang cukup baik. Output proses mampu mengikuti nilai set point yang ditetapkan.



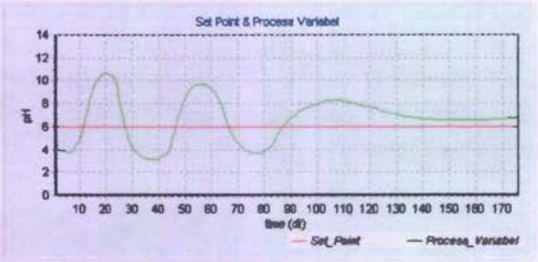
Gambar 4.8 Grafik respon sistem secara online untuk set point pH=7



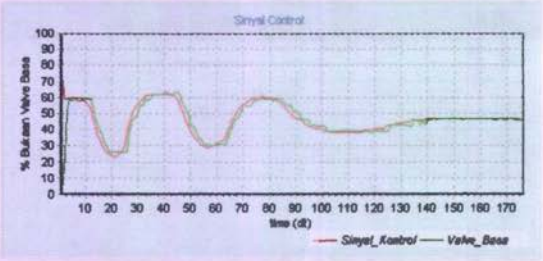
Gambar 4.9 Grafik sinyal kontrol secara online untuk set point pH= 7

Dari grafik didapatkan nilai *Tr=20 detik, Mp=33.72%, Ts=190 detik* dengan *Ess=0.8%* dan sampling time 1 detik.

Pada grafik 4.10 dapat dilihat bahwa pada set point pH sama dengan 6, sistem kontrol yang dikembangkan kurang mampu memberikan performansi yang baik.



Gambar 4.10 Grafik respon sistem secara online untuk set point pH= 6



Gambar 4.11 Grafik sinyal kontrol secara online untuk set point pH= 6

Dari grafik didapatkan nilai *Tr*=12 detik, *Mp=33.45%, Ts=146* detik dengan *Ess=6.5%* dan sampling time 1 detik.

BAB V
KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dalam tugas akhir ini telah dilakukan implementasi jaringan syaraf tiruan berbasis tracking control system terhadap miniplant pengendalian pH secara on line. Dari penelitian yang dilakukan dapat disimpulkan beberapa hal sebagai berikut :

1. Metode Jaringan Syaraf Tiruan berbasis tracking control system dapat diterapkan dengan baik pada miniplant sistem pengendalian pH.
2. Tidak dibutuhkan pelatihan ulang terhadap JST Controller bila sistem diberi gangguan atau bila terjadi perubahan set point
3. Jaringan syaraf tiruan yang digunakan dalam pengendalian pH menggunakan 1 unit input, 30 unit hidden dan 1 unit output dengan nilai *learning rate* sebesar 0,1 dan nilai *momentum* sebesar 0,7.
4. Sistem kontrol yang dikembangkan mampu menghasilkan performansi untuk set point pH 7 dengan nilai sebagai berikut: *Rise Time(Tr)*=20 detik, *Maksimum Overshoot (Mp)*=33.72 %, *Settling Time (Ts)*=190 Rdetik dan *Error Steady State (Ess)* = 0.8 %. Sedangkan untuk set point pH 6 didapatkan nilai performansi sebagai berikut: *Tr*=12 detik, *Mp*=33.45%, *Ts*=146 detik dengan Ess=6.5%.

5.2 Saran

Saran-saran yang dapat diberikan untuk meningkatkan performansi sistem adalah :

1. Untuk pengembangan dan penelitian lebih lanjut perlu ditambahkan kelengkapan data tentang spesifikasi

elemen-elemen pengendali untuk meminimalkan asumsi, dan perbaikan pada segi kualitas hardware baik pada perancangan perangkat elektronik maupun perancangan miniplant sistem pengendalian pH.

2. Dapat digunakan metode jaringan syaraf tiruan yang lain untuk mendapatkan performansi sistem yang lebih baik.

DAFTAR PUSTAKA

Christian, Gary D. 2004. **Analytical Chemistry**. Washington: John Willey & Sons. Inc.

Cordova, Hendra. 1999. **Perancangan Neuroregulator Pada Sistem Pengereman Antilock**. Tesis Magister. Program Studi Instrumentasi dan Control. Program Pascasarjana ITB.

Doherty, Sean Kevin. 1999. **Control of pH In Chemical Processes Using Artificial Neural Network**. Tesis Ph.D., Liverpool John Moores University.

Fausett, Laurence. 1994. **Fundamentals of Neural Networks Architectures, Alghoritms, and Applications**. Prentice Hall, Englewood Cliffs.

Gunterus F. 1994. **Falsafah Dasar Sistem Pengendalian Proses**. Jakarta: Elex Media Komputindo.

Johnson, Curtis D. 1997. **Process Control Instrumentation Technology**. New Jersey: Prentice – Hall International.

Kusumadewi, Sri. 2004. **Membangun Jaringan Syaraf Tiruan Menggunakan MATLAB & EXCEL LINK**. Yogyakarta: Graha Ilmu.

Mubarok, M. Nurdin. 2004. **Implementasi Sistem Kontrol Prediktif Berbasis Jaringan Syaraf Tiruan Secara Online Pada PCT13**. Teknik Fisika ITS: Tugas Akhir Program Studi Rekayasa Instrumentasi.

Rasiawan. 2002. **Rancang Bangun Sistem Pengendalian Proses Netralisasi pH Berbasis PC Dengan Motor Stepper sebagai Aktuator Control Valve**. Teknik Fisika ITS: Tugas Akhir, Program Studi Rekayasa Instrumentasi.

Tai, Heng-Ming, Junli Wang and Kaveh Ashenayi. 1992. **A Neural Network – Based Tracking Control System**. IEEE Transaction On Industrial Electronics, vol 39, No. 6.

Wijaya, Andry Fitra. 2004. **Perancangan Kontroler Neuro PID Self Tuning Berbasis Jaringan Syaraf Tiruan Pada Proses Netralisasi pH di PT Petrokimia Gresik**. Teknik

Fisika ITS: Tugas Akhir, Program Studi Rekayasa Instrumentasi.

Yien, Jean Peter. 2001. **Measuring, Modeling and Controlling the pH Value and the Dynamic Chemical State.** Finland: Helsinki University of Technology. Espoo.

www.sensorex.com/support/education/ph_education.html

www.senet.com.au/~cpeacock

Datasheet SN54/74LS151 MOTOROLA

Datasheet ADC0804 National Semiconductor

LAMPIRAN A
LISTING PROGRAM JST TRAINING

```
unit UTraining;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, SkinCtrls, SkinData, DynamicSkinForm, StdCtrls, SkinGrids,
  ExtCtrls, Mask, SkinBoxCtrls, Menus, math, spSkinShellCtrls;

type
  TFrm_Training = class(TForm)
    spSkinPanel1: TspSkinPanel;
    spDynamicSkinForm1: TspDynamicSkinForm;
    spSkinData1: TspSkinData;
    spStoredSkin1: TspStoredSkin;
    button_Random: TspSkinXFormButton;
    spSkinPanel2: TspSkinPanel;
    spSkinPanel3: TspSkinPanel;
    spSkinGroupBox1: TspSkinGroupBox;
    Ed_Input: TspSkinEdit;
    Ed_Hiden: TspSkinEdit;
    Ed_Output: TspSkinEdit;
    Ed_Learning: TspSkinEdit;
    Ed_Momentum: TspSkinEdit;
    spSkinLabel1: TspSkinLabel;
    spSkinLabel2: TspSkinLabel;
    spSkinLabel3: TspSkinLabel;
    spSkinLabel4: TspSkinLabel;
    spSkinLabel5: TspSkinLabel;
    Timer_Training: TTimer;
    spSkinMainMenu1: TspSkinMainMenu;
    spSkinMainMenuBar1: TspSkinMainMenuBar;
    File1: TMenuItem;
    Rando1: TMenuItem;
    Help1: TMenuItem;
    MainMenu1: TMainMenu;
    HalamanUtama1: TMenuItem;
    Exit1: TMenuItem;
    spSkinGroupBox2: TspSkinGroupBox;
    spSkinGroupBox3: TspSkinGroupBox;
    StringGrid_Data_Input: TspSkinStringGrid;
    spSkinScrollBar_H_Input: TspSkinScrollBar;
    spSkinScrollBar_V_Input: TspSkinScrollBar;
    spSkinGroupBox4: TspSkinGroupBox;
    StringGrid_Target: TspSkinStringGrid;
    spSkinScrollBar_H_Target: TspSkinScrollBar;
    spSkinScrollBar_V_Target: TspSkinScrollBar;
    Button_Load_Data: TspSkinXFormButton;
    spSkinGroupBox5: TspSkinGroupBox;
    StringGrid_Vij: TspSkinStringGrid;
    spSkinScrollBar_H_Vij: TspSkinScrollBar;
    spSkinScrollBar_V_Vij: TspSkinScrollBar;
    spSkinGroupBox6: TspSkinGroupBox;
    StringGrid_Wjk: TspSkinStringGrid;
    spSkinScrollBar_H_Wjk: TspSkinScrollBar;
    spSkinScrollBar_V_Wjk: TspSkinScrollBar;
    spSkinGroupBox7: TspSkinGroupBox;
    spSkinGroupBox8: TspSkinGroupBox;
    StringGrid_Vij_Akhir: TspSkinStringGrid;
    spSkinScrollBar_H_Vij_Akhir: TspSkinScrollBar;
    spSkinScrollBar_V_Vij_Akhir: TspSkinScrollBar;
    StringGrid_Wjk_Akhir: TspSkinStringGrid;
```

```
  spSkinScrollBar_H_Wjk_akhir: TspSkinScrollBar;
  spSkinScrollBar_V_Wjk_Akhir: TspSkinScrollBar;
  spSkinLabel6: TspSkinLabel;
  ed_IO: TspSkinEdit;
  Button_Normalisasi: TspSkinXFormButton;
  Button_Run: TspSkinXFormButton;
  spSkinStdLabel1: TspSkinStdLabel;
  spSkinStdLabel2: TspSkinStdLabel;
  LoadData1: TMenuItem;
  Normalisasi1: TMenuItem;
  Run1: TMenuItem;
  Lbl_MSE: TspSkinStdLabel;
  Ed_ERROR: TspSkinEdit;
  ED_RMSE: TspSkinEdit;
  Lbl_RMSE: TspSkinStdLabel;
  bt_Stop: TspSkinButton;
  bt_Reset: TspSkinButton;
  GRAFIKTRAINING1: TMenuItem;
  bt_Save: TspSkinButton;
  ONLINE1: TMenuItem;
  ESADC1: TMenuItem;
  spSkinStdLabel3: TspSkinStdLabel;
  ed_ITERASI: TspSkinEdit;
  ed_max_Iterasi: TspSkinEdit;
  spSkinStdLabel4: TspSkinStdLabel;
  spSkinPanel4: TspSkinPanel;
  spSkinButtonLabel1: TspSkinButtonLabel;
  VALIDASI1: TMenuItem;
  SaveDialog1: TspSkinSaveDialog;
  SaveDialog2: TspSkinSaveDialog;
  Image1: TImage;
  ComboBox_Aktifasi_Output: TspSkinComboBox;
  ComboBox_Aktifasi_Hidden: TspSkinComboBox;
  spSkinStdLabel5: TspSkinStdLabel;
  spSkinStdLabel6: TspSkinStdLabel;
  spSkinStdLabel7: TspSkinStdLabel;
  spSkinStdLabel8: TspSkinStdLabel;
  procedure button_RandomClick(Sender: TObject);
  procedure FormCreate(Sender: TObject);
  procedure HalamanUtama1Click(Sender: TObject);
  procedure Rando1Click(Sender: TObject);
  procedure Exit1Click(Sender: TObject);
  procedure Timer_TrainingTimer(Sender: TObject);
  procedure Button_Load_DataClick(Sender: TObject);
  procedure Button_NormalisasiClick(Sender: TObject);
  procedure Button_RunClick(Sender: TObject);
  procedure bt_StopClick(Sender: TObject);
  procedure bt_ResetClick(Sender: TObject);
  procedure GRAFIKTRAINING1Click(Sender: TObject);
  procedure bt_SaveClick(Sender: TObject);
  procedure ONLINE1Click(Sender: TObject);
  procedure ESADC1Click(Sender: TObject);
  procedure VALIDASI1Click(Sender: TObject);
private
  { Private declarations }
public
  { Public declarations }
end;

var


  dr,dr1,dr2,dr5:integer;
  dr3,dr4:single;
  Frm_Training: TFrm_Training;
  Inp,K_inp,Inp1,Outp,K_Outp,Outp1,learate,momentum:single;

  Bobot_ij:array[0..10,1..1000]of single;
  Bobot_V_ij,Bobot_W_jk,Vij_New,Wjk_New,Bobot_jk:array[0..10,0..1000]of single;
```

```
 Delta_Bias_Hidd,Delta_Bias_Inp,Delta_Vij,Vij:array[0..10,0..1000] of Single;
 Del1,Def_Zln,Del_In,Delta_Wjk,Wjk: array[0..1000,0..1000] of Single;
 Z_In,Zj: array [0..1000,0..1000] of single;
 Yln,Y_In,Yk,m,n: array [0..1000] of single;
 data_inp:array[1..1000,1..5] of single;
 Delta_Bias_Output,Del2,err,data_outp:array[0..1000] of single;

 errormax,RMSE,RMS,MSE,SSE:single;
 bias_inp,bias_hidd,x,hidd:integer;

 alfa,A:single;

 Target:integer;
 aktif,selesai:boolean;

 Vij_last, Wjk_Last: array [0..1000,0..1000] of single;

implementation
uses UMainMenu, ULoad_Data_Train, UAktifasi,UGrafik_Iterasi, UTesADC, U_Validasi;
{$R *.dfm}

procedure TFrm_Training.button_RandomClick(Sender: TObject);
var
 i,j,k,d:integer;
begin

momentum:=StrTofloat(Ed_Momentum.Text);


{----------kondisi jika target terpenuhi proses berakhir----------}


StringGrid_Vij.RowCount:=(StrToInt(Ed_Hiden.Text)+1);
StringGrid_Wjk.RowCount:=(StrToInt(Ed_Hiden.Text)+2);
StringGrid__Vij_Akhir.RowCount:=(StrToInt(Ed_Hiden.Text)+1);
StringGrid_Wjk_Akhir.RowCount:=(StrToInt(Ed_Hiden.Text)+2);
if (learate>1)or (learate<0) or (momentum>1)or (momentum<0) then
 begin
  ShowMessage('Momentum dan Learning rate harap '+
        'diisi nilai dengan range 0 - 1');
  exit;
 end
  else
   begin
    Bias_inp:=StrToInt(Ed_Input.Text)+1;
    bias_hidd:=StrToInt(Ed_Hiden.Text)+1;

{----------Ngerandom Bobot Awal NN----------}

{----------Input Layer----------}

    for i:=1 to StrToInt(Ed_Input.Text) do
     for j:=1 to StrToInt(Ed_Hiden.Text) do
     begin

      Bobot_ij[0,j]:=1;

      {-----Untuk mendapatkan nilai bobot awal secara acak-----}
      {-----dengan range nilai -0.5 sampai dengan 0.5--------}

      Bobot_ij[i,j]:=(random(11)-5)/10;

      {-----UpdateVij[i,j]:=(random(11)-5)/10;----------}
      {-----j baris pada StringGridVij----------}

      StringGrid_Vij.Cells[0,j]:=inttostr(j); {jumlah node hidden}
      StringGrid_Vij.Cells[i,0]:=inttostr(i); {jumlah node input}

      {-----i kolom pada StringGridVij----------}
```

```
      StringGrid_Vij.cells[i,j]:=floatTostr(Bobot_ij[i,j]);
      StringGrid__Vij_Akhir.Cells[i,j]:=StringGrid_Vij.Cells[i,j];

      {------nilai -1 pada baris j kolom 1 di StringGridVij (bias hidden)--}

      StringGrid_Vij.cells[bias_inp,j]:=floatTostr(Bobot_ij[0,j]);
      StringGrid__Vij_Akhir.Cells[bias_inp,j]:=StringGrid_Vij.Cells[bias_inp,j];
      StringGrid_Vij.cells[bias_inp,0]:='Bobot Bias';
      StringGrid_Data_Input.Cells[1,0]:='e('+IntToStr(i)+')';
      StringGrid_Data_Input.Cells[bias_inp,0]:='Input Bias';

    end;

{-----------------Output Layer-----------------------------------}

  for k:=1 to StrToInt(Ed_Output.Text) do
   for j:=1 to StrToInt(Ed_Hiden.Text) do
    begin

     Bobot_jk[k,0]:=1;

     {---------Untuk mendapatkan nilai bobot awal secara acak---------}
     {---------dengan range nilai -0.5 sampai dengan 0.5---------}

     Bobot_jk[k,j]:=(random(11)-5)/10;

     {---------UpdateWjk[k,j]:=(random(11)-5)/10;---------}

     StringGrid_Wjk.Cells[0,j]:=inttostr(j);
     StringGrid_Wjk.Cells[k,0]:=inttostr(k);
     StringGrid_Wjk.cells[k,j]:=floatTostr(Bobot_jk[k,j]);
     StringGrid_Wjk_Akhir.Cells[k,j]:=StringGrid_Wjk.Cells[k,j];
     StringGrid_Wjk.cells[k,bias_hidd]:=floatTostr(Bobot_jk[k,0]);
     StringGrid_Wjk_Akhir.Cells[k,bias_hidd]:=StringGrid_Wjk.Cells[k,bias_hidd];

    end;

  end;

end;


procedure TFrm_Training.FormCreate(Sender: TObject);
var i,d:integer;
begin
 StringGrid_Data_Input.Cells[0,0]:='No.';
 StringGrid_Target.Cells[0,0]:='No.';
 StringGrid_Target.Cells[1,0]:='u(n)';
 StringGrid_Target.RowCount:=StrToInt(ed_IO.Text)+1;
 StringGrid_Target.Cells[1,0]:='Target';
 alfa:=StrToFloat(Ed_Learning.Text);
 bias_hidd:=StrToInt(Ed_Hiden.Text)+1;
 bias_inp:=StrToInt(Ed_Input.Text)+1;
 Target:=StrToInt(Ed_Output.Text);
end;

procedure TFrm_Training.HalamanUtama1Click(Sender: TObject);
begin
 Form1.Show;
 Frm_Training.Hide;
end;

procedure TFrm_Training.Rando1Click(Sender: TObject);
begin
 button_RandomClick(Sender);
end;
```

```
procedure TFrm_Training.Exit1Click(Sender: TObject);
begin
 Application.Terminate;
end;

procedure TFrm_Training.Timer_TrainingTimer(Sender: TObject);
var
 i,j,k,r:integer;
begin

ed_ITERASI.Text:=IntToStr(x);

for r:=1 to StrToInt(ed_IO.Text) do
begin

{----------Looping Data Input Ke Hidden----------}


  for j:=1 to StrToInt(Ed_Hiden.Text) do
   begin
    for i:=1 to StrToInt(Ed_Input.Text) do
     begin

    Z_In[r,j]:=Vij_New[bias_inp,j]+data_inp[r,i]*Vij_New[i,j];

      if (ComboBox_Aktifasi_Hidden.Text='T') then
       begin
         Zj[r,j]:=f2(z_in[r,j]);
       end
      else
       if (ComboBox_Aktifasi_Hidden.Text='L') then
        begin
          Zj[r,j]:=f4(Z_In[r,j]);
        end
      else
       if (ComboBox_Aktifasi_Hidden.Text='BP') then
        begin
          Zj[r,j]:=f1(Z_In[r,j]);
        end
      else
       if (ComboBox_Aktifasi_Hidden.Text='BN') then
        begin
          Zj[r,j]:=f3(Z_In[r,j]);
        end;
    end;
   end;


{----------Looping Data Hidden to Output----------}

   Y_In[1]:=0;
   for k:=1 to StrToInt(Ed_Output.Text) do
    begin

     for j:=1 to StrToInt(Ed_Hiden.Text) do
      begin

       if (x=1) and (r=1) then
        begin
         Wjk[k,j]:=Wjk_new[k,j];
         Y_In[k]:=Y_In[k]+Zj[r,j]*Wjk[k,j];
        end

       else
       begin
        Wjk[k,j]:=Wjk_new[k,j];
        Wjk[k,bias_hidd]:=Wjk_new[k,bias_hidd];
        Y_In[k]:=Y_In[k]+Zj[r,j]*Wjk[k,j];
       end;
```

```
    end;

  if (r=1) and (x=1) then
   begin
    yin[k]:=Wjk[k,bias_hidd]+Y_in[k];
   end

   else
    begin
     yin[k]:=StrToFloat(StringGrid_Wjk_Akhir.Cells[k,bias_hidd])+Y_in[k];
   end;

   if (ComboBox_Aktifasi_Output.Text='L') then
     begin
      yk[k]:=f4(yin[k]);
     end
   else
   if (ComboBox_Aktifasi_Output.Text='BP') then
     begin
      yk[k]:=f1(yin[k]);
     end
   else
   if (ComboBox_Aktifasi_Output.Text='T') then
     begin
      yk[k]:=f2(yin[k]);
     end
   else
   if (ComboBox_Aktifasi_Output.Text='BN') then
     begin
      yk[k]:=f3(yin[k]);
     end;

  err[r]:=StrToFloat(StringGrid_Target.Cells[1,r])-Yk[k];
  SSE:=SSE+sqr(err[r]);
  Ed_ERROR.Text:=FloatToStr(err[r]);
  if (ComboBox_Aktifasi_Output.Text='BP') then
      begin
       Del2[r]:=err[r]*def1(yk[k]);
      end
    else
   if (ComboBox_Aktifasi_Output.Text='T') then
      begin
       Del2[r]:=err[r]*def2(yk[k]);
      end
    else
   if (ComboBox_Aktifasi_Output.Text='BN') then
      begin
       Del2[r]:=err[r]*def3(yk[k]);
      end
    else
   if (ComboBox_Aktifasi_Output.Text='L') then
      begin
       Del2[r]:=err[r]*def4(yk[k]);
      end;

  end;

  Delta_Bias_Output[r]:=alfa*Del2[r];  {Perubahan Bobot Bias, delta(B2k)}

{-----------------Informasi error dari output Layer-----------------}

  for k:=1 to StrToInt(Ed_Output.Text) do
   begin
    for j:=1 to StrToInt(Ed_Hiden.Text) do
     begin
      Delta_Wjk[k,j]:=alfa*Del2[r]*Zj[r,j];
      Delta_Bias_Hidd[k,bias_hidd]:=alfa*Del2[r];  {delta B2 output}
     end;
```

```
    end;

{----------Informasi error dari hidden layer----------}

  for k:=1 to StrToInt(Ed_Output.Text) do
   for j:=1 to StrToInt(Ed_Hiden.Text) do
    begin

      Del_In[r,j]:=Del2[r]*Wjk[k,j];

      if (ComboBox_Aktifasi_Hidden.Text='BP') then
       begin
         Def_ZIn[r,j]:=Def1(Zj[r,j]);
       end
      else
      if (ComboBox_Aktifasi_Hidden.Text='T') then
       begin
         Def_ZIn[r,j]:=Def2(Zj[r,j]);
       end
      else
      if (ComboBox_Aktifasi_Hidden.Text='BN') then
       begin
         Def_ZIn[r,j]:=Def3(Zj[r,j]);
       end
      else
      if (ComboBox_Aktifasi_Hidden.Text='L') then
       begin
         Def_ZIn[r,j]:=Def4(Zj[r,j]);
       end;

      Del1[r,j]:=Del_in[r,j]*Def_ZIn[r,j];
    end;



  for j:=1 to StrToInt(Ed_Hiden.Text) do
   for i:=1 to StrToInt(Ed_Input.Text) do
    begin
     Delta_Vij[i,j]:=alfa*Del1[r,j]*Data_inp[r,i];
     Delta_Bias_Inp[bias_inp,j]:=alfa*Del1[r,j];
    end;

{----------Update Bobot----------}
 for k:=1 to StrToInt(Ed_Output.Text) do
  for j:=1 to StrToInt(Ed_Hiden.Text) do
   begin

    Wjk_New[k,j]:=Wjk[k,j]+delta_Wjk[k,j];

    Wjk_New[k,bias_hidd]:=StrToFloat(StringGrid_Wjk.Cells[k,bias_hidd])
              +Delta_Bias_Hidd[k,bias_hidd];

    StringGrid_Wjk_Akhir.Cells[k,j]:=FloatToStr(Wjk[k,j]);
    StringGrid_Wjk_Akhir.Cells[k,bias_hidd]:=FloatToStr(Wjk[k,bias_hidd]);
   end;

 for j:=1 to StrToInt(Ed_Hiden.Text) do
  For i:=1 to StrToInt(Ed_Input.Text) do
   begin
    Vij[i,j]:=Vij[i,j]+delta_Vij[i,j];

    Vij_New[i,j]:=Vij[i,j];

    Vij[bias_inp,j]:=Vij[bias_inp,j]+Delta_Bias_Inp[bias_inp,j];

    Vij_New[bias_inp,j]:=Vij[bias_inp,j];
```

```
      StringGrid__Vij_Akhir.Cells[i,j]:=FloatToStr(Vij[i,j]);

      StringGrid__Vij_Akhir.Cells[bias_inp,j]:=FloatToStr(Vij[bias_inp,j]);
    end;

END; {end of r}
inc(x);

RMSE:=sqrt(SSE/x);
ED_RMSE.Text:=FloatToStr(RMSE);
Form_Train.Series1.AddXY(x,StrToFloat(ED_RMSE.Text),'',clRed);
SSE:=0;

if (x>StrToInt(ed_Max_Iterasi.Text)) or (RMSE<0.03) then
Timer_Training.Enabled:=false;

for i:=1 to StrToInt(Ed_Input.Text) do
 for j:=1 to StrToInt(Ed_Hiden.Text) do
  begin
   Vij_last[i,j]:=StrToFloat(StringGrid__Vij_Akhir.Cells[i,j]);
   Vij_last[bias_inp,j]:=StrToFloat(StringGrid__Vij_Akhir.Cells[i+1,j]);
  end;

for k:=1 to StrToInt(Ed_Output.Text) do
 for j:=1 to StrToInt(Ed_Hiden.Text) do
  begin
   Wjk_Last[k,j]:=StrToFloat(StringGrid_Wjk_Akhir.Cells[k,j]);
   Wjk_Last[k,bias_hidd]:=StrToFloat(StringGrid_Wjk_Akhir.Cells[k,bias_hidd]);
  end;
end;


procedure TFrm_Training.Button_Load_DataClick(Sender: TObject);
begin
 with Form_Load do
  begin
   Show;
   Caption:='Load Data Input & Output Target';
   Group_In.Caption:='Data Input';
   Group_Out.Caption:='Data Output Target';
   Button_Training.Visible:=True;
   Bt_JST_Control.Visible:=False;
   bt_Validasi_Data.Visible:=False;
   bt_LOAD_BOBOT.Visible:=False;
   Open_Dialog_In.Title:='Open Data Input';
   Open_Dialog_Out.Title:='Open Data Output';
  end;
end;

procedure TFrm_Training.Button_NormalisasiClick(Sender: TObject);
var i,j,k,d,r:integer;
begin
 if StringGrid_Data_Input.Cells[1,1]='' then
  MessageDlg('Anda belum memasukkan data-data pada tabel.',mtInformation,[mbOK],0)
 else
  for d:=1 to StrToInt(Ed_Input.Text) do
   for r:=1 to StrToInt(ed_IO.Text) do
    begin
     Inp:=StrToFloat(StringGrid_Data_Input.Cells[d,r]);
     if Inp>K_inp then K_inp:=Inp;
    end;

  for d:=1 to StrToInt(Ed_Input.Text) do
   for r:=1 to StrToInt(ed_IO.Text) do
    begin
     Inp1:=StrToFloat(StringGrid_Data_Input.Cells[d,r]);
     StringGrid_Data_Input.Cells[d,r]:=FloatToStr(Inp1/K_inp);
    end;
```

```
  for r:=1 to StrToInt(ed_IO.Text) do
   begin
    Outp:=StrToFloat(StringGrid_Target.Cells[1,r]);
    if Outp>K_Outp then K_Outp:=Outp;
   end;

  for r:=1 to StrToInt(ed_IO.Text) do
   begin
    Outp1:=StrToFloat(StringGrid_Target.Cells[1,r]);
    StringGrid_Target.Cells[1,r]:=FloatToStr(Outp1/K_Outp);
   end;


end;

procedure TFrm_Training.Button_RunClick(Sender: TObject);
var i,j,k,r:integer;
begin
 if StringGrid_Data_Input.Cells[1,1]='' then
  MessageDlg('Masukkan data-data, kemudian klik Normalisasi sehingga '+
       'bisa dilakukan Proses Training.',mtInformation,[mbOK],0)
 else
  begin
   for i:=1 to StrToInt(Ed_Input.Text)+1 do {jumlah node input}
    for r:=1 to StrToInt(ed_IO.Text) do      {jumlah pasangan data input}
     data_inp[r,i]:=StrToFloat(StringGrid_Data_Input.Cells[i,r]);

    for r:=1 to StrToInt(ed_IO.Text) do
     data_outp[r]:=StrToFloat(StringGrid_Target.Cells[1,r]);

    for i:=1 to StrToInt(Ed_Input.Text)+1 do
     for j:=1 to StrToInt(Ed_Hiden.Text) do
     begin
      Vij[i,j]:=StrToFloat(StringGrid_Vij.Cells[i,j]);
      Bobot_V_ij[i,j]:=StrToFloat(StringGrid_Vij.Cells[i,j]);
      Vij_New[i,j]:=StrToFloat(StringGrid__Vij_Akhir.Cells[i,j]);
     end;

    for k:=1 to StrToInt(Ed_Output.Text) do
     for j:=1 to StrToInt(Ed_Hiden.Text)+1 do
      begin
       Wjk[k,j]:=StrToFloat(StringGrid_Wjk.Cells[k,j]);
       Bobot_W_jk[k,j]:=StrToFloat(StringGrid_Wjk.Cells[k,j]);
       Wjk_New[k,j]:=StrToFloat(StringGrid_Wjk_Akhir.Cells[k,j]);
      end;

    x:=1;
    alfa:=StrToFloat(Ed_Learning.Text);
    bias_hidd:=StrToInt(Ed_Hiden.Text)+1;
    bias_inp:=StrToInt(Ed_Input.Text)+1;
    SSE:=0;
    RMS:=0;
    RMSE:=0;
    Timer_Training.Enabled:=True;
    dr:=StrToInt(Ed_Input.Text);
    dr1:=StrToInt(Ed_Hiden.Text);
    dr2:=StrToInt(Ed_Output.Text);
    dr3:=StrToFloat(Ed_Learning.Text);
    dr4:=StrToFloat(Ed_Momentum.Text);
    dr5:=StrToInt(ed_IO.Text);
   end;
end;


procedure TFrm_Training.bt_StopClick(Sender: TObject);
begin
 Timer_Training.Enabled:=False;
end;
```

```
procedure TFrm_Training.bt_ResetClick(Sender: TObject);
var i,j,k,r,d:integer;
begin
 Timer_Training.Enabled:=False;
 for r:=1 to StrToInt(ed_IO.Text) do
  begin
  StringGrid_Data_Input.Cells[1,r]:='';
  StringGrid_Data_Input.Cells[2,r]:='';
  StringGrid_Target.Cells[1,r]:='';
  StringGrid_Target.Cells[2,r]:='';
  end;

 for i:=1 to StrToInt(Ed_Input.Text)+1 do
  for j:=1 to StrToInt(Ed_Hiden.Text) do
   begin
   StringGrid_Vij.Cells[i,j]:='';
   StringGrid__Vij_Akhir.Cells[i,j]:='';
   end;

 for k:=1 to StrToInt(Ed_Output.Text) do
  for j:=1 to StrToInt(Ed_Hiden.Text) do
   begin
    StringGrid_Wjk.Cells[k,j]:='';
    StringGrid_Wjk_Akhir.Cells[k,j]:='';
   end;

 Ed_ERROR.Text:='';
 ED_RMSE.Text:='';
 Form_Train.Series1.Clear;
 Form_Train.Series2.Clear;
end;

procedure TFrm_Training.GRAFIKTRAINING1Click(Sender: TObject);
begin
Form_Train.Show;
end;

procedure TFrm_Training.bt_SaveClick(Sender: TObject);
var
 FDATA:TextFile;
 i,j,k:Integer;
begin
 Timer_Training.Enabled:=False;
 if SaveDialog1.Execute then
   begin
    AssignFile(FDATA,SaveDialog1.FileName);
    Rewrite(FDATA);
    Writeln(FData,'          Data-data Hasil Pelatihan JST');
    Writeln(FData,'=============================================================');
    Writeln(FData,'');
    Writeln(FData,'Jumlah Node Input  :  ',dr);
    Writeln(FData,'Jumlah Node Hidden :  ',dr1);
    Writeln(FData,'Jumlah Node Output :  ',dr2);
    Writeln(FData,'Learning Rate      :  ',dr3:5:2);
    Writeln(FData,'Momentum           :  ',dr4:5:2);
    Writeln(FData,'Jumlah Data        :  ',dr5);
    Writeln(FData,'RMSE               :  ',RMSE:6:4);
    Writeln(FData,'Jumlah Iterasi     :  ',x-1);
    Writeln(FData,'');
    Writeln(FData,'-------------------------------------------');
    Writeln(FData,'');
    Writeln(FData,' No.  Bobot Vij Akhir  Bobot Bias Input ');
    Writeln(FData);
     for j:=1 to StrToInt(Ed_Hiden.Text) do
       Writeln(FData,j:4,'    ',Vij_Last[1,j]:10:7,'      ',Vij_Last[2,j]:10:7);
    Writeln(FData);
    Writeln(FData);
    Writeln(FData,'-------------------------------------------');
    Writeln(FData);
```

```
      Writeln(FData,' No.  Bobot Wjk Akhir');
      Writeln(FData);
       for j:=1 to StrToInt(Ed_Hiden.Text) do
         writeln(FData,j:4,'    ',Wjk_Last[1,j]:10:7);
      Writeln(FData);
      Writeln(FData);
      Writeln(FData,'-------------------------------------------------');
      Writeln(FData);
      Writeln(FData,'     Bobot Bias Hidden');
      Writeln(FData);
      Writeln(FData,'       ',Wjk_Last[1,bias_hidd]:10:7);
      Writeln(FData);
      Writeln(FData,'=============================================================');
      Writeln(FData);
      Writeln(FData,' No.  Bobot Vij Awal   Bobot Bias Input Awal');
      for i:=1 to StrToInt(Ed_Input.Text) do
       for j:=1 to StrToInt(Ed_Hiden.Text) do
        begin
         Write(FData,j:4,'  ',Bobot_V_ij[i,j]:10:7,'          ');
         Writeln(FData,Bobot_V_ij[i+1,j]:10:7);
        end;
      Writeln(FData);
      Writeln(FData,'=============================================================');
      Writeln(FData);
      Writeln(FData,' No.  Bobot Wjk Awal   Bobot Bias Hidden Awal');
      for k:=1 to StrToInt(Ed_Output.Text) do
       for j:=1 to StrToInt(Ed_Hiden.Text) do
        begin
         Write(FData,j:4,'  ',Bobot_W_jk[k,j]:10:7,'          ');
         Writeln(FData,Bobot_W_jk[k,StrToInt(Ed_Hiden.Text)+1]:10:7);
        end;
      Writeln(FData);
      Writeln(FData,'=============================================================');
        CloseFile(FDATA);
    end;
  if SaveDialog2.Execute then
   Form_Train.Chart1.SaveToMetafileEnh(SaveDialog2.FileName);

end;

Procedure TFrm_Training.ONLINE1Click(Sender: TObject);
begin
  frmTESADC.Show;
  frmTESADC.Caption:='ONLINE';
  frmTESADC.Panel_Controller.Visible:=True;
  Form1.Hide;
end;

procedure TFrm_Training.ESADC1Click(Sender: TObject);
begin
   Form1.Timer1.Enabled:=False;
   form1.Hide;
   frmTesADC.Show;
   frmTESADC.Panel_Controller.Visible:=False;
end;

procedure TFrm_Training.VALIDASI1Click(Sender: TObject);
begin
 Frm_Training.Hide;
 Form_Train.Hide;
 frm_Validasi.Show;
end;


end.
```

LAMPIRAN B
LISTING PROGRAM VALIDASI JST

```
unit U_Validasi;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, StdCtrls, Grids, TeeProcs, TeEngine, Chart, ExtCtrls, Menus,
  Series, Gauges, SkinCtrls, DynamicSkinForm, spSkinShellCtrls,
  SkinBoxCtrls;

type
  Tfrm_Validasi = class(TForm)
    GroupBox1: TGroupBox;
    StringGrid_Data_Input: TStringGrid;
    StringGrid_Data_Target: TStringGrid;
    GroupBox2: TGroupBox;
    StringGrid_Bobot_Vij: TStringGrid;
    StringGrid_Bobot_Wjk: TStringGrid;
    Label1: TLabel;
    Label2: TLabel;
    Label3: TLabel;
    Label4: TLabel;
    ed_RMSE: TEdit;
    Label5: TLabel;
    bt_RUN: TButton;
    bt_LOAD_DATA: TButton;
    bt_LOAD_BOBOT: TButton;
    bt_Normalisasi: TButton;
    Button1: TButton;
    Panel1: TPanel;
    Chart1: TChart;
    Series1: TFastLineSeries;
    Panel2: TPanel;
    StringGrid_OUTPUT: TStringGrid;
    Label7: TLabel;
    MainMenu1: TMainMenu;
    File1: TMenuItem;
    View1: TMenuItem;
    GroupBox3: TGroupBox;
    Label8: TLabel;
    Label9: TLabel;
    Label10: TLabel;
    Label11: TLabel;
    Input_Text: TEdit;
    Hidden_Text: TEdit;
    Output_Text: TEdit;
    IO_Text: TEdit;
    Timer_Validasi: TTimer;
    Gauge: TGauge;
    bt_Validasi_Finish: TButton;
    LOADDATA1: TMenuItem;
    LOADBOBOT1: TMenuItem;
    NORMALISASI1: TMenuItem;
    RUN1: TMenuItem;
    SAVE1: TMenuItem;
    EXIT1: TMenuItem;
    HalamanUtama1: TMenuItem;
    spDynamicSkinForm1: TspDynamicSkinForm;
    spSkinMainMenu1: TspSkinMainMenu;
    spSkinMainMenuBar1: TspSkinMainMenuBar;
    File2: TMenuItem;
    LOADDAT1: TMenuItem;
```

```
  A1: TMenuItem;
  NORMALISASI2: TMenuItem;
  RUN2: TMenuItem;
  View2: TMenuItem;
  HalamanUtama2: TMenuItem;
  Exit2: TMenuItem;
  Timer_Complete: TTimer;
  bt_RESET: TButton;
  RESET1: TMenuItem;
  Label12: TLabel;
  RAININGJST1: TMenuItem;
  RAININGJST2: TMenuItem;
  SaveDialog1: TspSkinSaveDialog;
  SaveDialog2: TspSkinSaveDialog;
  spSkinStdLabel1: TspSkinStdLabel;
  CmbBox_Aktifasi_Hidden: TspSkinComboBox;
  spSkinStdLabel2: TspSkinStdLabel;
  CmbBox_Aktifasi_Output: TspSkinComboBox;
  spSkinStdLabel5: TspSkinStdLabel;
  spSkinStdLabel6: TspSkinStdLabel;
  spSkinStdLabel7: TspSkinStdLabel;
  spSkinStdLabel8: TspSkinStdLabel;
  Chart2: TChart;
  Series2: TFastLineSeries;
  SaveDialog3: TspSkinSaveDialog;
  spSkinPanel4: TspSkinPanel;
  spSkinButtonLabel1: TspSkinButtonLabel;
  procedure bt_LOAD_DATAClick(Sender: TObject);
  procedure bt_LOAD_BOBOTClick(Sender: TObject);
  procedure bt_NormalisasiClick(Sender: TObject);
  procedure bt_RUNClick(Sender: TObject);
  procedure Button1Click(Sender: TObject);
  procedure Timer_ValidasiTimer(Sender: TObject);
  procedure EXIT1Click(Sender: TObject);
  procedure LOADDATA1Click(Sender: TObject);
  procedure LOADBOBOT1Click(Sender: TObject);
  procedure NORMALISASI1Click(Sender: TObject);
  procedure RUN1Click(Sender: TObject);
  procedure HalamanUtama1Click(Sender: TObject);
  procedure LOADDAT1Click(Sender: TObject);
  procedure Exit2Click(Sender: TObject);
  procedure A1Click(Sender: TObject);
  procedure NORMALISASI2Click(Sender: TObject);
  procedure RUN2Click(Sender: TObject);
  procedure HalamanUtama2Click(Sender: TObject);
  procedure Timer_CompleteTimer(Sender: TObject);
  procedure bt_Validasi_FinishClick(Sender: TObject);
  procedure bt_RESETClick(Sender: TObject);
  procedure RESET1Click(Sender: TObject);
  procedure RAININGJST1Click(Sender: TObject);
  procedure RAININGJST2Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frm_Validasi: Tfrm_Validasi;
 inp,K_inp,Inp1,outp,k_outp,Outp1:single;
 Zj,Z_In,Wjk,Vij,data_inp:array [1..1000,1..1000] of single;
 err,yk,yIn,Y_In,data_outp:array [1..1000] of single;
 x,bias_inp,bias_hidd:integer;
 RMS,SSE,RMSE,RMSE2:single;
implementation
uses ULOad_Data_Train, UMainMenu, UTraining, UAktifasi;
{$R *.dfm}

{------ Prosedur Untuk Loading Data Input (error) dan Output (Target) -------}
```

```
procedure Tfrm_Validasi.bt_LOAD_DATAClick(Sender: TObject);
begin
if (IO_Text.Text='0') or (Hidden_Text.Text='') then
   begin
   MessageDlg('Apakah jumlah Data IO yang akan di Uji'
                    + ' sudah disetting??',mtInformation,[mbOK],0);
   IO_Text.SetFocus;
   end
  else

  with Form_Load do
   begin
    Show;
    Caption:='Load Data Input & Output Target (Validasi)';
    Group_In.Caption:='Data Input (Validasi)';
    Group_Out.Caption:='Data Output (Target)';
    bt_Validasi_Data.Visible:=True;
    Button_Training.Visible:=False;
    Bt_JST_Control.Visible:=False;
    bt_LOAD_BOBOT.Visible:=False;
    Open_Dialog_In.Title:='Open Data Input (Validasi)';
    Open_Dialog_Out.Title:='Open Data Output (Validasi)';
   end;
end;

procedure Tfrm_Validasi.bt_LOAD_BOBOTClick(Sender: TObject);
begin
 if (Hidden_Text.Text='0') or (Hidden_Text.Text='') then
   begin
   MessageDlg('Apakah jumlah node hidden'
                +' sudah disetting??',mtInformation,[mbOK],0);
   Hidden_Text.SetFocus;
   end
  else

  with Form_Load do
   begin
    Show;
    Caption:='Load Bobot Vij & Bobot Wjk (Hasil Training)';
    Group_In.Caption:='Bobot Vij (Hasil Training)';
    Group_Out.Caption:='Bobot Wjk (Hasil Training)';
    bt_LOAD_BOBOT.Visible:=True;
    bt_Validasi_Data.Visible:=False;
    Button_Training.Visible:=False;
    Bt_JST_Control.Visible:=False;
    Open_Dialog_In.Title:='Open Bobot Vij (Hasil Training)';
    Open_Dialog_Out.Title:='Open Bobot Wjk (Hasil Training)';
   end;
end;

procedure Tfrm_Validasi.bt_NormalisasiClick(Sender: TObject);
var i,j,k,d,r:integer;
begin
 if StringGrid_Data_Input.Cells[1,1]='' then
   MessageDlg('Anda belum memasukkan data input!!!',mtInformation,[mbOK],0)
 else
   for d:=1 to StrToInt(Input_Text.Text) do
    for r:=1 to StrToInt(IO_Text.Text) do
      begin
       Inp:=StrToFloat(StringGrid_Data_Input.Cells[d,r]);
       if Inp>K_inp then K_inp:=Inp;
      end;

   for d:=1 to StrToInt(Input_Text.Text) do
    for r:=1 to StrToInt(IO_Text.Text) do
      begin
       Inp1:=StrToFloat(StringGrid_Data_Input.Cells[d,r]);
       StringGrid_Data_Input.Cells[d,r]:=FloatToStr(Inp1/K_inp);
      end;
```

```
  for r:=1 to StrToInt(IO_Text.Text) do
   begin
    Outp:=StrToFloat(StringGrid_Data_Target.Cells[1,r]);
    if Outp>K_Outp then K_Outp:=Outp;
   end;

  for r:=1 to StrToInt(IO_Text.Text) do
   begin
    Outp1:=StrToFloat(StringGrid_Data_Target.Cells[1,r]);
    StringGrid_Data_Target.Cells[1,r]:=FloatToStr(Outp1/K_Outp);
   end;
end;

procedure Tfrm_Validasi.bt_RUNClick(Sender: TObject);
var i,j,k,r:integer;
begin
 if StringGrid_Data_Input.Cells[1,1]='' then
  MessageDlg('Masukkan data-data, kemudian klik Normalisasi sehingga '+
        'bisa dilakukan Proses Validasi.',mtInformation,[mbOK],0)
 else

  begin
   for i:=1 to StrToInt(Input_Text.Text)+1 do {jumlah node input}
    for r:=1 to StrToInt(IO_Text.Text) do    {jumlah pasangan data input}
     data_inp[r,i]:=StrToFloat(StringGrid_Data_Input.Cells[i,r]);

    for r:=1 to StrToInt(IO_Text.Text) do
     data_outp[r]:=StrToFloat(StringGrid_Data_Target.Cells[1,r]);

    for i:=1 to StrToInt(Input_Text.Text)+1 do
     for j:=1 to StrToInt(Hidden_Text.Text) do
     begin
      Vij[i,j]:=StrToFloat(StringGrid_Bobot_Vij.Cells[i,j]);
     end;

    for k:=1 to StrToInt(Output_Text.Text) do
     for j:=1 to StrToInt(Hidden_Text.Text)+1 do
      begin
       Wjk[k,j]:=StrToFloat(StringGrid_Bobot_Wjk.Cells[k,j]);
      end;

    x:=1;
    Y_In[1]:=0;
    SSE:=0;
    RMSE:=0;
    Gauge.MaxValue:=StrToInt(IO_Text.Text);
    StringGrid_OUTPUT.RowCount:=StrToInt(IO_Text.Text);
    StringGrid_OUTPUT.Cells[1,0]:='Output Model JST';
    bias_inp:=StrToInt(Input_Text.Text)+1;
    bias_hidd:=StrToInt(Hidden_Text.Text)+1;
    Timer_Validasi.Interval:=100;
    Timer_Validasi.Enabled:=True;
   end;
end;

procedure Tfrm_Validasi.Button1Click(Sender: TObject);
begin
Form1.Show;
frm_Validasi.Hide;
end;

{******** ALLAHU AKBAR **********************************}
procedure Tfrm_Validasi.Timer_ValidasiTimer(Sender: TObject);
var
 i,j,k,r:integer;
begin

{----------Looping Data Input Ke Hidden----------}
```

```
for j:=1 to StrToInt(Hidden_Text.Text) do
 begin
  for i:=1 to StrToInt(Input_Text.Text) do
  begin
   Z_In[x,j]:=vij[2,j]+data_inp[x,i]*Vij[i,j];  {bias_inp = 2}

   if  (CmbBox_Aktifasi_Hidden.Text='T') then
    begin
      Zj[x,j]:=f2(z_in[x,j]);
      {Zj[r,j]:=(1-exp(-1*z_in[r,j]))/(1+exp(-1*z_in[r,j]));}
    end
   else
    if  (CmbBox_Aktifasi_Hidden.Text='L') then
     begin
       Zj[x,j]:=f4(Z_In[x,j]);
     end
   else
    if  (CmbBox_Aktifasi_Hidden.Text='BP') then
     begin
       Zj[x,j]:=f1(Z_In[x,j]);
     end
   else
    if  (CmbBox_Aktifasi_Hidden.Text='BN') then
     begin
       Zj[x,j]:=f3(Z_In[x,j]);
     end;
    {Zj[x,j]:=f2(z_in[x,j]);}
  {Zj[x,j]:=(1-exp(-1*z_in[x,j]))/(1+exp(-1*z_in[x,j]));}
  end;
 end;


{----------Looping Data Hidden to Output---------}

  Y_In[1]:=0;
  for k:=1 to StrToInt(Output_Text.Text) do
   begin
    for j:=1 to StrToInt(Hidden_Text.Text) do
     begin
        Wjk[k,j]:=Wjk[k,j];
        Y_In[k]:=Y_In[k]+Zj[x,j]*Wjk[k,j];
     end;  {end of loop j}

    yIn[x]:=Wjk[k,bias_hidd]+Y_In[k];

    if (CmbBox_Aktifasi_Output.Text='L') then
       begin
         yk[x]:=f4(yin[x]);
         {yk[k]:=(1-(exp(-1*Yin[k])))/(1+(exp (-1*YIn[k])));}
       end
     else
     if (CmbBox_Aktifasi_Output.Text='BP') then
       begin
         yk[x]:=f1(yin[x]);
         {yk[k]:=(1-(exp(-1*Yin[k])))/(1+(exp (-1*YIn[k])));}
       end
     else
     if (CmbBox_Aktifasi_Output.Text='T') then
       begin
         yk[x]:=f2(yin[x]);
         {yk[k]:=(1-(exp(-1*Yin[k])))/(1+(exp (-1*YIn[k])));}
       end
     else
     if (CmbBox_Aktifasi_Output.Text='BN') then
       begin
         yk[x]:=f3(yin[x]);
         {yk[k]:=(1-(exp(-1*Yin[k])))/(1+(exp (-1*YIn[k])));}
       end;

    StringGrid_OUTPUT.Cells[1,x]:=FloatToStr(yk[x]);
```

```
      Rms:=Rms+(sqr(data_Outp[x]-yk[x]));
    end; {end of loop k}

     Series1.AddXY(x,(data_Outp[x]),'',clred);    {data target}
     Series2.AddXY(x,(yk[x]),'',clgreen);         {output MODEL JST}
     {Rms:=(sqr(data_Outp[x]-yk[x]));}
     Rmse:=(sqrt(Rms/x));
     RMSE2:=RMSE;
     ed_RMSE.Text:=FloatToStr(RMSE);


    StringGrid_OUTPUT.Cells[0,x]:=IntToStr(x);
    Gauge.Progress:=x;

 inc(x);
 If x>(StrToInt(IO_Text.Text)) then
  begin
   Rmse:=0;
   Timer_Validasi.Enabled:=False;
   Label12.Visible:=True;
   Timer_Complete.Enabled:=True;
   Gauge.Visible:=False;
  end;

end;

procedure Tfrm_Validasi.EXIT1Click(Sender: TObject);
begin
 Application.Terminate;
end;

procedure Tfrm_Validasi.LOADDATA1Click(Sender: TObject);
begin
  bt_LOAD_DATAClick(sender);
end;

procedure Tfrm_Validasi.LOADBOBOT1Click(Sender: TObject);
begin
 bt_LOAD_BOBOTClick(sender);
end;

procedure Tfrm_Validasi.NORMALISASI1Click(Sender: TObject);
begin
 bt_NormalisasiClick(sender);
end;

procedure Tfrm_Validasi.RUN1Click(Sender: TObject);
begin
 bt_RUNClick(sender);
end;

procedure Tfrm_Validasi.HalamanUtama1Click(Sender: TObject);
begin
 Button1Click(sender);
end;

procedure Tfrm_Validasi.LOADDAT1Click(Sender: TObject);
begin
 bt_LOAD_DATAClick(sender);
end;

procedure Tfrm_Validasi.Exit2Click(Sender: TObject);
begin
 Application.Terminate;
end;

procedure Tfrm_Validasi.A1Click(Sender: TObject);
begin
 bt_LOAD_BOBOTClick(sender);
```

```
end;

procedure Tfrm_Validasi.NORMALISASI2Click(Sender: TObject);
begin
 bt_NormalisasiClick(sender);
end;

procedure Tfrm_Validasi.RUN2Click(Sender: TObject);
begin
 bt_RUNClick(sender);
end;

procedure Tfrm_Validasi.HalamanUtama2Click(Sender: TObject);
begin
 Button1Click(sender);
end;

procedure Tfrm_Validasi.Timer_CompleteTimer(Sender: TObject);
begin
 if Label12.Visible=False then
  Label12.Visible:=True
 else if Label12.Visible=True then
  Label12.Visible:=False;
end;

procedure Tfrm_Validasi.bt_Validasi_FinishClick(Sender: TObject);
var FValidasi:TextFile;
  dr:integer;
begin
 Timer_Complete.Enabled:=False;
 Label12.Visible:=fALSE;
 SaveDialog1.Title:='Save Output Model JST As';
 SaveDialog2.Title:='Save Grafik Validasi As';
 if SaveDialog1.Execute then
  begin
   AssignFile(FValidasi,SaveDialog1.Filename);
   Rewrite(FValidasi);
   Writeln(FValidasi,'');
   Writeln(FValidasi,'          DATA VALIDASI MODEL JST');
   Writeln(FValidasi,'');
   Writeln(FValidasi,'  RMSE  : ',RMSE2:2:10);
   //Writeln(FVal,'');
   Writeln(FValidasi,'=============================================================');
   Writeln(FValidasi,'  No.   Target   Output model   Error');
   Writeln(FValidasi,'=============================================================');
    for dr:=1 to StrToInt(IO_Text.Text) do
     writeln(FValidasi,' ',dr:4,'    ',data_Outp[dr]:2:5,'    ',yk[dr]:2:5,'    ',data_Outp[dr]-yk[dr]:2:5);
   CloseFile(FValidasi);
  end;

 if SaveDialog2.Execute then
  Chart1.SaveToMetafileEnh(SaveDialog2.FileName);
 if SaveDialog3.Execute then
  Chart2.SaveToMetafileEnh(SaveDialog3.FileName);
end;

procedure Tfrm_Validasi.bt_RESETClick(Sender: TObject);
var i,j,k,r,d:integer;
begin
 Timer_Validasi.Enabled:=False;
 Timer_Complete.Enabled:=False;
 for r:=1 to StrToInt(IO_Text.Text) do
  begin
  StringGrid_Data_Input.Cells[1,r]:='';
  StringGrid_Data_Target.Cells[1,r]:='';
  StringGrid_OUTPUT.Cells[1,r]:='';
  end;

 for i:=1 to StrToInt(Input_Text.Text)+1 do
  for j:=1 to StrToInt(Hidden_Text.Text) do
```

```
    begin
    StringGrid_Bobot_Vij.Cells[i,j]:='';
    end;

 for k:=1 to StrToInt(Output_Text.Text) do
  for j:=1 to StrToInt(Hidden_Text.Text)+1 do
    begin
      StringGrid_Bobot_Wjk.Cells[k,j]:='';
    end;

 ed_RMSE.Text:='';
 Hidden_Text.Text:='0';
 IO_Text.Text:='0';
 RMSE:=0;
 RMS:=0;

 Series1.Clear;
 Series2.Clear;
end;

procedure Tfrm_Validasi.RESET1Click(Sender: TObject);
begin
 bt_RESETClick(sender);
end;


procedure Tfrm_Validasi.RAININGJST1Click(Sender: TObject);
begin
 frm_Validasi.Hide;
 Frm_Training.Show;
end;

procedure Tfrm_Validasi.RAININGJST2Click(Sender: TObject);
begin
 RAININGJST1Click(sender);
end;

end.
```

LAMPIRAN C
Listing Program TES ADC dan JST Controller

```
unit UTesADC;

interface

uses
  Windows, Messages, SysUtils, Variants, Classes, Graphics, Controls, Forms,
  Dialogs, SkinCtrls, SkinData, DynamicSkinForm, ExtCtrls, bsSkinData,
  bsSkinCtrls, StdCtrls, Mask, SkinBoxCtrls, ComCtrls, jpeg, TeeProcs,
  TeEngine, Chart, Series, SkinGrids, spSkinShellCtrls, Menus;

type
  TfrmTESADC = class(TForm)
    spSkinData2: TspSkinData;
    spStoredSkinADC: TspStoredSkin;
    spDynamicSkinForm1: TspDynamicSkinForm;
    spStoredSkin3: TspStoredSkin;
    spSkinData3: TspSkinData;
    spSkinGroupBox1: TspSkinGroupBox;
    spSkinGroupBox2: TspSkinGroupBox;
    spSkinGroupBox3: TspSkinGroupBox;
    spSkinPanel1: TspSkinPanel;
    spSkinXFormButtonRUN: TspSkinXFormButton;
    spSkinXFormButtonSTOP: TspSkinXFormButton;
    spSkinXFormButtonSAVE: TspSkinXFormButton;
    spSkinXFormButtonCLEAR: TspSkinXFormButton;
    spSkinXFormButtonHALAMAN_UTAMA: TspSkinXFormButton;
    spSkinXFormButtonEXIT: TspSkinXFormButton;
    spSkinLabel1: TspSkinLabel;
    spSkinLabel2: TspSkinLabel;
    spSkinLabel3: TspSkinLabel;
    spSkinLabel4: TspSkinLabel;
    spSkinLabel5: TspSkinLabel;
    spSkinLabel6: TspSkinLabel;
    spSkinLabel7: TspSkinLabel;
    spSkinLabel8: TspSkinLabel;
    spSkinLabel9: TspSkinLabel;
    spSkinGroupBox4: TspSkinGroupBox;
    spSkinGroupBox5: TspSkinGroupBox;
    spSkinLabel10: TspSkinLabel;
    spSkinLabel11: TspSkinLabel;
    spSkinScrollBar_Valve_BASA: TspSkinScrollBar;
    Timer_Motor: TTimer;
    Timer_Sampling: TTimer;
    Image_dark: TImage;
    Image_light: TImage;
    spSkinPanel2: TspSkinPanel;
    Chart1: TChart;
    Series1: TFastLineSeries;
    Series2: TFastLineSeries;
    Series3: TFastLineSeries;
    Series4: TFastLineSeries;
    spSkinCheckRadioBox1: TspSkinCheckRadioBox;
    Chart2: TChart;
    Series5: TFastLineSeries;
    spSkinStringGrid1: TspSkinStringGrid;
    spSkinScrollBar1: TspSkinScrollBar;
    Button_Browse: TspSkinSpeedButton;
    Button_Load_ASAM: TspSkinSpeedButton;
    CheckBox_OpenLOOP: TspSkinCheckRadioBox;
    Edit_Asam: TspSkinEdit;
    spSkinStdLabel1: TspSkinStdLabel;
    spSkinOpenDialog1: TspSkinOpenDialog;
```

```
spSkinStdLabel2: TspSkinStdLabel;
spSkinStdLabel3: TspSkinStdLabel;
spSkinStdLabel4: TspSkinStdLabel;
spSkinStdLabel5: TspSkinStdLabel;
spSkinEdit2: TspSkinEdit;
spSkinEdit3: TspSkinEdit;
spSkinEdit4: TspSkinEdit;
spSkinEdit5: TspSkinEdit;
Button_Load_BASA: TspSkinSpeedButton;
spSkinStdLabel6: TspSkinStdLabel;
Edit_Basa: TspSkinEdit;
SaveDialog1: TspSkinSaveDialog;
BUTTON_RESET: TspSkinXFormButton;
spSkinScrollBar_Valve_Asam: TspSkinScrollBar;
Panel_Controller: TspSkinPanel;
spSkinGroupBox6: TspSkinGroupBox;
Chart3: TChart;
Chart4: TChart;
Series6: TFastLineSeries;
Series7: TFastLineSeries;
Series8: TFastLineSeries;
Series9: TFastLineSeries;
Stringgrid_Bobot_Input: TspSkinStringGrid;
Stringgrid_Bobot_Hdden: TspSkinStringGrid;
spSkinScrollBar2: TspSkinScrollBar;
spSkinScrollBar3: TspSkinScrollBar;
spSkinScrollBar4: TspSkinScrollBar;
spSkinScrollBar5: TspSkinScrollBar;
bt_LOAD_CONTROLLER: TspSkinXFormButton;
spSkinGroupBox7: TspSkinGroupBox;
ed_Sinyal_Kontrol: TspSkinEdit;
spSkinGroupBox8: TspSkinGroupBox;
spSkinStdLabel7: TspSkinStdLabel;
spSkinStdLabel8: TspSkinStdLabel;
spSkinStdLabel9: TspSkinStdLabel;
ed_PH_PV: TspSkinEdit;
ed_Bin_PV: TspSkinEdit;
ed_Volt_PV: TspSkinEdit;
spSkinGroupBox9: TspSkinGroupBox;
ComboBox_SETPOINT: TspSkinComboBox;
spSkinStdLabel10: TspSkinStdLabel;
spSkinStdLabel11: TspSkinStdLabel;
ed_ERROR: TspSkinEdit;
spSkinLabel12: TspSkinLabel;
spSkinLabel13: TspSkinLabel;
bt_RUN_CONTROLLER: TspSkinXFormButton;
bt_STOP_CONTROLLER: TspSkinXFormButton;
bt_RESET_CONTROLLER: TspSkinXFormButton;
bt_SAVE_CONTROLLER: TspSkinXFormButton;
spSkinGroupBox10: TspSkinGroupBox;
bt_MENU_CONTROLLER: TspSkinButton;
ed_Valve_Basa_Contoller: TspSkinEdit;
ed_Bin_Basa_Controller: TspSkinEdit;
ed_Volt_Basa_Controller: TspSkinEdit;
spSkinStdLabel12: TspSkinStdLabel;
spSkinStdLabel13: TspSkinStdLabel;
spSkinStdLabel14: TspSkinStdLabel;
SaveDialog2: TspSkinSaveDialog;
CheckRadioBox_ONLINE: TspSkinCheckRadioBox;
SaveDialog3: TspSkinSaveDialog;
SaveDialog4: TspSkinSaveDialog;
Image1: TImage;
spSkinStdLabel15: TspSkinStdLabel;
ed_nilai_pH: TspSkinStdLabel;
ed_Bukaan_Asam: TspSkinStdLabel;
spSkinStdLabel16: TspSkinStdLabel;
ed_Bukaan_Basa: TspSkinStdLabel;
spSkinGroupBox11: TspSkinGroupBox;
spSkinStdLabel17: TspSkinStdLabel;
spSkinStdLabel18: TspSkinStdLabel;
```

```
  cmb_hiidden: TspSkinComboBox;
  cmb_output: TspSkinComboBox;
  spSkinStdLabel19: TspSkinStdLabel;
  spSkinStdLabel20: TspSkinStdLabel;
  spSkinStdLabel21: TspSkinStdLabel;
  spSkinStdLabel22: TspSkinStdLabel;
  spSkinStdLabel23: TspSkinStdLabel;
  Hidden_Text: TspSkinEdit;
  procedure spSkinXFormButtonHalamanUtamaClick(Sender: TObject);
  procedure spSkinButtonEXITClick(Sender: TObject);
  procedure spSkinXFormButtonEXITClick(Sender: TObject);
  procedure spSkinXFormButtonHALAMAN_UTAMAClick(Sender: TObject);
  procedure FormCRitae(Sender: TObject);
  procedure spSkinXFormButtonRUNClick(Sender: TObject);
  procedure spSkinXFormButtonSTOPClick(Sender: TObject);
  procedure FormClose(Sender: TObject; var Action: TCloseAction);
  procedure FormShow(Sender: TObject);
  procedure spSkinScrollBar_Valve_ASAMChange(Sender: TObject);
  procedure spSkinScrollBar_Valve_BASAChange(Sender: TObject);
  procedure Timer_MotorTimer(Sender: TObject);
  procedure Timer_SamplingTimer(Sender: TObject);
  procedure spSkinXFormButtonSAVEClick(Sender: TObject);
  procedure spSkinXFormButtonCLEARClick(Sender: TObject);
  procedure Button_BrowseClick(Sender: TObject);
  procedure Button_Load_ASAMClick(Sender: TObject);
  procedure Button_Load_BASAClick(Sender: TObject);
  procedure BUTTON_RESETClick(Sender: TObject);
  procedure bt_MENU_CONTROLLERClick(Sender: TObject);
  procedure bt_LOAD_CONTROLLERClick(Sender: TObject);
  procedure bt_RUN_CONTROLLERClick(Sender: TObject);
  procedure bt_STOP_CONTROLLERClick(Sender: TObject);
  procedure bt_RESET_CONTROLLERClick(Sender: TObject);
  procedure bt_SAVE_CONTROLLERClick(Sender: TObject);
  procedure Image1Click(Sender: TObject);
 private
  { Private declarations }
 public
  { Public declarations }
 end;

var
 frmTESADC: TfrmTESADC;
 jmldata, x, divv, Data_Inp1, Data_Inp2, Data_Inp3:Integer;
 Rita,norm,error,v_pH, v_Motor_Asam, v_Motor_Basa:Real;
 SET_POINT,PEHA,ERROR_PLANT,pH, Persen_Motor_Asam, Persen_Motor_Basa:array of Real;
 sp_Motor_Asam,sp_Motor_Basa, Bin, Filt_Bin: array of integer;
 a,b:Boolean;
 HRStatus:String;
 Vij,Wjk:array [1..100,1..100] of single;
 Z_In, Zj:array [1..100] of single;
 Y_In, Yin, Yk:array [1..100] of single;

implementation
uses UMainMenu, Unit_Motor_Positioner, Unit_IO, ULoad_Data_Train, UTraining, Unit_Plant, UAktifasi;
{$R *.dfm}

{fungsi untuk mengkonversi bilangan biner 8 bit menjadi bilangan desimal}
function BitToByte(bit0, bit1, bit2, bit3, bit4, bit5, bit6, bit7:Integer):integer;
begin
 BitToByte:=bit0*1+bit1*2+bit2*4+bit3*6+bit4*16+bit5*32+bit6*64+bit7*128;
end;

{Procedure untuk mengkonversi bilangan desimal menjadi bilangan biner 8 bit}
{Allohumma Sholli wa Salim 'Alaa Muhammad}
Procedure ByteToBit(nilai:Integer);
begin
 if nilai>=128 then
  begin
   bit[7]:=1;
   nilai:=nilai-128;
```

```
    end
    else
      begin
        bit[7]:=0;
      end;
    if nilai>=64 then
     begin
      bit[6]:=1;
      nilai:=nilai-64;
     end
     else
      begin
        bit[6]:=0;
      end;
    if nilai>=32 then
     begin
      bit[5]:=1;
      nilai:=nilai-32;
     end
     else
      begin
        bit[5]:=0;
      end;
    if nilai>=16 then
     begin
      bit[4]:=1;
      nilai:=nilai-16;
     end
     else
      begin
        bit[4]:=0;
      end;
    if nilai>=8 then
     begin
      bit[3]:=1;
      nilai:=nilai-8;
     end
     else
      begin
        bit[3]:=0;
      end;
    if nilai>=4 then
     begin
      bit[2]:=1;
      nilai:=nilai-4;
     end
     else
      begin
        bit[2]:=0;
      end;
    if nilai>=2 then
     begin
      bit[1]:=1;
      nilai:=nilai-2;
     end
     else
      begin
        bit[1]:=0;
      end;
    if nilai=1 then
     begin
      bit[0]:=1;
     end
     else
      begin
        bit[0]:=0;
      end;
end;

procedure TfrmTESADC.spSkinXFormButtonHalamanUtamaClick(Sender: TObject);
```

```
begin
 Form1.Show;
 frmTesADC.Hide;
end;

procedure TfrmTESADC.spSkinButtonEXITClick(Sender: TObject);
begin
 application.Terminate;
end;

procedure TfrmTESADC.spSkinXFormButtonEXITClick(Sender: TObject);
begin
Output($378,0);
Application.Terminate;
end;

procedure TfrmTESADC.spSkinXFormButtonHALAMAN_UTAMAClick(Sender: TObject);
begin
frmTESADC.Hide;
Form1.Show;
end;

procedure TfrmTESADC.FormCRitae(Sender: TObject);
begin
 spSkinScrollBar_Valve_ASAM.Min:=MinMax_Pos_Motor_Asam('min');
 spSkinScrollBar_Valve_ASAM.Max:=MinMax_Pos_Motor_Asam('max');
 spSkinScrollBar_Valve_BASA.Min:=MinMax_Pos_Motor_Basa('min');
 spSkinScrollBar_Valve_BASA.Max:=MinMax_Pos_Motor_Basa('max');
 a:=False;
 b:=False;
 divv:=0;
 x:=0;
 HRStatus:=' Disconnected';
 Output($378,$00);
end;

procedure TfrmTESADC.spSkinXFormButtonRUNClick(Sender: TObject);
begin
   if CheckBox_OpenLOOP.Checked=False then
      begin
       if Timer_Sampling.Enabled=False then
         Timer_Sampling.Enabled:=True;
       if Timer_Motor.Enabled=False then
         Timer_Motor.Enabled:=True;
      end
   else if CheckBox_OpenLOOP.Checked=True then
     begin
      if (Edit_Asam.Text='') and (Edit_Basa.Text='') then
       begin
         MessageDlg('Anda Belum mengisi File Path Asam dan Basa.',mtInformation,[mbOK],0);
       end
      else
       if Timer_Sampling.Enabled=False then
         Timer_Sampling.Enabled:=True;
       if Timer_Motor.Enabled=False then
         Timer_Motor.Enabled:=True;
      end;
end;

procedure TfrmTESADC.spSkinXFormButtonSTOPClick(Sender: TObject);
begin
 c:=False;
 Timer_Sampling.Enabled:=False;
 Form1.spSkinButtonLabel_OpenLoop.Enabled:=True;
 Output($378,0);
 Timer_Motor.Enabled:=false;
end;

procedure TfrmTESADC.FormClose(Sender: TObject; var Action: TCloseAction);
begin
```

```
 Timer_Motor.Enabled:=False;
 Timer_Motor.Enabled:=False;
 Pos_Motor_Asam(MinMax_Pos_Motor_Asam('min'),False);
 Pos_Motor_Basa(MinMax_Pos_Motor_Basa('min'),False);
 form1.Show;
 frmTesADC.Hide;
end;

procedure TfrmTESADC.FormShow(Sender: TObject);
begin
 Timer_Motor.Enabled:=False;
end;

procedure TfrmTESADC.spSkinScrollBar_Valve_ASAMChange(Sender: TObject);
var min_Asam, max_Asam:Integer;
begin
 min_Asam:=MinMax_Pos_Motor_Asam('min');
 max_Asam:=MinMax_Pos_Motor_Asam('max');
 spSkinLabel10.Caption:=IntToStr(Round(((spSkinScrollBar_Valve_ASAM.Position-min_Asam)/
 (max_Asam-min_Asam))*100))+' % Buka';
end;

procedure TfrmTESADC.spSkinScrollBar_Valve_BASAChange(Sender: TObject);
var min_Basa, max_Basa:Integer;
begin
 min_Basa:=MinMax_Pos_Motor_Basa('min');
 max_Basa:=MinMax_Pos_Motor_Basa('max');
 spSkinLabel11.Caption:=IntToStr(Round(((spSkinScrollBar_Valve_BASA.Position-min_Basa)/
 (max_Basa-min_Basa))*100))+' % Buka';
end;

procedure TfrmTESADC.Timer_MotorTimer(Sender: TObject);
begin
 Inc(divv);
 if divv=50 then
  begin
   divv:=0;
   if Image_light.Visible=true then
    begin
     Image_light.Visible:=false;
     a:=true;
     b:=true;
    end
   else
    begin
     Image_light.Visible:=True;
     a:=False;
     b:=False;
    end;
  end;

  {_____ASAM_____}
  Pos_Motor_Asam(spSkinScrollBar_Valve_ASAM.Position,a);
  Pos_Motor_Asam(spSkinScrollBar_Valve_ASAM.Position,False);
  Pos_Motor_Asam(spSkinScrollBar_Valve_ASAM.Position,a);

  {_____BASA_____}
  Pos_Motor_Basa(spSkinScrollBar_Valve_BASA.Position,b);
  Pos_Motor_Basa(spSkinScrollBar_Valve_BASA.Position,False);
  Pos_Motor_Basa(spSkinScrollBar_Valve_BASA.Position,b);
end;

procedure TfrmTESADC.Timer_SamplingTimer(Sender: TObject);
var j,k,min_Asam, min_Basa, max_Asam, max_Basa, kendali_basa, kendali_asam:Integer;
begin
 Inc(x);
 SetLength(pH,x);
 SetLength(SET_POINT,x);
 SetLength(ERROR_PLANT,x);
 SetLength(Persen_Motor_Asam,x);
```

```
SetLength(Persen_Motor_Basa,x);
SetLength(sp_Motor_Asam,x);
SetLength(sp_Motor_Basa,x);
SetLength(Bin,x);
SetLength(Filt_Bin,x);
min_Asam:=MinMax_Pos_Motor_Asam('min');
max_Asam:=MinMax_Pos_Motor_Asam('max');
min_Basa:=MinMax_Pos_Motor_Basa('min');
max_Basa:=MinMax_Pos_Motor_Basa('max');
spSkinEdit2.Text:=spSkinStringGrid1.Cells[0,x];
spSkinEdit3.Text:=spSkinStringGrid1.Cells[1,x];
spSkinEdit4.Text:=spSkinStringGrid1.Cells[2,x];
spSkinEdit5.Text:=spSkinStringGrid1.Cells[3,x];

{---------- TES ADC ----------}

if CheckBox_OpenLOOP.Checked=False then
  begin
    {----------Begin of Input Motor Asam----------}
      sp_Motor_Asam[x-1]:=Round(((spSkinScrollBar_Valve_ASAM.Position-min_Asam)/(max_Asam-
min_Asam))*100);
      spSkinLabel10.Caption:=FloatToStr(sp_motor_Asam[x-1])+' % Buka';
      Data_Inp1:=Input('Motor_Asam');
      Persen_Motor_Asam[x-1]:=((Data_Inp1-min_Asam)/(max_Asam-min_Asam))*100;
      spSkinLabel7.Caption:='valve : '+IntToStr(Round(Persen_Motor_Asam[x-1]))+' %';
      v_Motor_Asam:=(Data_Inp1/255)*5;
      ByteToBit(Data_Inp1);
      spSkinLabel8.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
      +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
      spSkinLabel9.Caption:=FloatToStr(v_Motor_Asam)+' volt';
    {End of Input Motor Asam}

    {----------Begin of Input Motor Basa----------}
      sp_Motor_Basa[x-1]:=Round(((spSkinScrollBar_Valve_BASA.Position-min_Basa)/(max_Basa-
min_Basa))*100);
      spSkinLabel11.Caption:=FloatToStr(sp_motor_Basa[x-1])+' % Buka';
      Data_Inp2:=Input('Motor_Basa');
      Persen_Motor_Basa[x-1]:=((Data_Inp2-min_Basa)/(max_Basa-min_Basa))*100;
      spSkinLabel1.Caption:='valve : '+IntToStr(Round(Persen_Motor_Basa[x-1]))+' %';
      v_Motor_Basa:=(Data_Inp2/255)*5;
      ByteToBit(Data_Inp2);
      spSkinLabel2.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
      +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
      spSkinLabel3.Caption:=FloatToStr(v_Motor_Basa)+' volt';
    {End of Input Motor Basa}

    {----------Begin of Input Sensor----------}

    Data_Inp3:=Input('Sensor');
    Bin[x-1]:=Data_Inp3;
    Filt_Bin[x-1]:=Data_Inp3;
    if spSkinCheckRadioBox1.Checked=True then
    begin
      if Filt_Bin[x-1]-Filt_Bin[x-4]>4 then
        Filt_Bin[x-1]:=Filt_Bin[x-2];
    end;
    pH[x-1]:=((Data_Inp3/255)*11)+1; {untuk menampilkan nilai pH pada grafik dg range 1 - 12}
    spSkinStringGrid1.Cells[3,x]:=floatToSTr(pH[x-1]);
    spSkinLabel4.Caption:=FloatToStr(pH[x-1]);
    v_pH:=(Data_Inp3/255)*5;
    ByteToBit(Data_Inp3);
    spSkinLabel5.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
      +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
    spSkinLabel6.Caption:=FloatToStr(v_pH)+' volt';
  {End of Input Sensor}

  if x>25 then
    begin
      Chart1.BottomAxis.SetMinMax(0,x+6);
      Chart2.BottomAxis.SetMinMax(0,x+6);
```

```
  end;

  Series1.AddXY(x,sp_Motor_Asam[x-1],'',clBlue);  {Set Point Bukaan Valve Asam}
  Series2.AddXY(x,Persen_Motor_Asam[x-1],'',clRed); {Persen Bukaan Valve}
  Series3.AddXY(x,Sp_Motor_Basa[x-1],'',clFuchsia); {Set Point Bukaan Valve Asam}
  Series4.AddXY(x,Persen_Motor_Basa[x-1],'',clGreen); {Persen Bukaan Valve}
  series5.AddXY(x,pH[x-1],'',clYellow);
end;
{---------- END of TES ADC ----------}

{---------- OPEN LOOP ----------}

if CheckBox_OpenLOOP.Checked=true then
begin
{----------Begin of Input Motor Asam----------}
  kendali_asam:=Round(strToFloat(spSkinEdit4.Text));
  spSkinScrollBar_Valve_Asam.Position:=kendali_asam;
  sp_Motor_Asam[x-1]:=spSkinScrollBar_Valve_ASAM.Position;
  spSkinLabel10.Caption:=FloatToStr(sp_motor_Asam[x-1])+' % Buka';
  Data_Inp1:=Input('Motor_Asam');
  Persen_Motor_Asam[x-1]:=((Data_Inp1-min_Asam)/(max_Asam-min_Asam))*100;
  spSkinLabel7.Caption:='valve : '+IntToStr(Round(Persen_Motor_Asam[x-1]))+' %';
  v_Motor_Asam:=(Data_Inp1/255)*5;
  ByteToBit(Data_Inp1);
  spSkinLabel8.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
  +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
  spSkinLabel9.Caption:=FloatToStr(v_Motor_Asam)+' volt';

{End of Input Motor Asam}

{----------Begin of Input Motor BASA----------}
  kendali_basa:=Round(StrToFloat(spSkinEdit3.Text));
  spSkinScrollBar_Valve_BASA.Position:=kendali_basa;
  sp_Motor_Basa[x-1]:=spSkinScrollBar_Valve_BASA.Position;
  spSkinLabel11.Caption:=FloatToStr(sp_motor_Basa[x-1])+' % Buka';
  Data_Inp2:=Input('Motor_Basa');
  Persen_Motor_Basa[x-1]:=((Data_Inp2-min_Basa)/(max_Basa-min_Basa))*100;
  spSkinLabel1.Caption:='valve : '+IntToStr(Round(Persen_Motor_Basa[x-1]))+' %';
  v_Motor_Basa:=(Data_Inp2/255)*5;
  ByteToBit(Data_Inp2);
  spSkinLabel2.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
  +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
  spSkinLabel3.Caption:=FloatToStr(v_Motor_Basa)+' volt';
{End of Input Motor Basa}

{----------Begin of Input Sensor----------}

 Data_Inp3:=Input('Sensor');
 Bin[x-1]:=Data_Inp3;
 Filt_Bin[x-1]:=Data_Inp3;
 if spSkinCheckRadioBox1.Checked=True then
 begin
  if Filt_Bin[x-1]-Filt_Bin[x-4]>4 then
    Filt_Bin[x-1]:=Filt_Bin[x-2];
 end;
 pH[x-1]:=((Data_Inp3/255)*11)+1; {untuk menampilkan nilai pH pada grafik dg range 1 - 12}
 spSkinStringGrid1.Cells[3,x]:=floatToSTr(pH[x-1]);
 spSkinLabel4.Caption:=FloatToStr(pH[x-1]);
 v_pH:=(Data_Inp3/255)*5;
 ByteToBit(Data_Inp3);
 spSkinLabel5.Caption:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
  +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
 spSkinLabel6.Caption:=FloatToStr(v_pH)+' volt';
{End of Input Sensor}

 if x>=jmldata then
  begin
  Timer_Sampling.Enabled:=false;
  MessageDlg('Pengambilan Data Open Loop Telah Selesai.',mtInformation,[mbOK],0);
```

```
  end;

 Series1.AddXY(x,sp_Motor_Asam[x-1],'',clBlue);  {Set Point Bukaan Valve Asam}
 Series2.AddXY(x,Persen_Motor_Asam[x-1],'',clRed); {Persen Bukaan Valve}
 Series3.AddXY(x,Sp_Motor_Basa[x-1],'',clFuchsia); {Set Point Bukaan Valve Asam}
 Series4.AddXY(x,Persen_Motor_Basa[x-1],'',clGreen); {Persen Bukaan Valve}
 series5.AddXY(x,pH[x-1],'',clYellow);

end
{---------------- END of OPEN LOOP ----------------------}


{---------------------- ONLINE ----------------}

else if CheckRadioBox_ONLINE.Checked=true then
begin
 {---------------JST CONTROL---------------------}
 SET_POINT[x-1]:=StrToFloat(ComboBox_SETPOINT.Text);
 error:=StrToFloat(ComboBox_SETPOINT.Text)-StrToFloat(ed_PH_PV.Text);
 norm:=(error/7);
 ed_ERROR.Text:=FloatToStr(error);
 ERROR_PLANT[x-1]:=error;

 for j:=1 to StrToInt(Frm_Training.Ed_Hiden.Text) do
  begin

    Z_In[j]:=vij[StrToInt(Frm_Training.Ed_Input.Text)+1,j]
         +(norm*vij[StrToInt(Frm_Training.Ed_Input.Text),j]);

    if  (cmb_hiidden.Text='T') then
    begin
      Zj[j]:=f2(z_in[j]);
      {Zj[r,j]:=(1-exp(-1*z_in[r,j]))/(1+exp(-1*z_in[r,j]));}
    end
    else if  (cmb_hiidden.Text='L') then
     begin
      Zj[j]:=f4(Z_In[j]);
     end
    else if  (cmb_hiidden.Text='BP') then
     begin
      Zj[j]:=f1(Z_In[j]);
     end
    else if  (cmb_hiidden.Text='BN') then
     begin
      Zj[j]:=f3(Z_In[j]);
     end;
    {Zj[j]:=(1-(exp(-1*Z_In[j])))/(1+(exp(-1*Z_In[j])));}
  end;

 Y_In[1]:=0;
  for k:=1 to StrToInt(Frm_Training.Ed_Output.Text) do
   begin
    for j:=1 to StrToInt(Frm_Training.Ed_Hiden.Text) do
     begin

      Wjk[k,j]:=Wjk[k,j];

      Y_In[k]:=Y_In[k]+Zj[j]*wjk[k,j];

     end;

      Yin[k]:=Wjk[k,StrToInt(Frm_Training.Ed_Hiden.Text)+1]+Y_In[k];

     if (cmb_output.Text='L') then
      begin
       yk[k]:=f4(yin[k]);
      end
     else if (cmb_output.Text='BP') then
      begin
       yk[k]:=f1(yin[k]);
```

```
        end
       else if (cmb_output.Text='T') then
        begin
         yk[k]:=f2(yin[k]);
        end
       else if (cmb_output.Text='BN') then
        begin
         yk[k]:=f3(yin[k]);
        end;

        Rita:=(yk[k]-0.20)*100;
        ed_Sinyal_Kontrol.Text:=FloatToStr(Rita);


    end;  {end loop for k}

{---------- Begin Of Input Motor Basa ----------}
 kendali_basa:=Round(StrToFloat(ed_Sinyal_Kontrol.Text));
 spSkinScrollBar_Valve_BASA.Position:=kendali_basa;
 sp_Motor_Basa[x-1]:=spSkinScrollBar_Valve_BASA.Position;
 Data_Inp2:=Input('Motor_Basa');
 Persen_Motor_Basa[x-1]:=((Data_Inp2-min_Basa)/(max_Basa-min_Basa))*100;
 ed_Valve_Basa_Contoller.Text:=IntToStr(Round(Persen_Motor_Basa[x-1]))+' %';
 ed_Bukaan_Basa.Caption:=ed_Valve_Basa_Contoller.Text;
 Frm_Plant.lbl_Valve_Basa.Caption:=ed_Valve_Basa_Contoller.Text;
 v_Motor_Basa:=(Data_Inp2/255)*5;
 ByteToBit(Data_Inp2);
 ed_Bin_Basa_Controller.Text:='Bin : '+IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
 +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
 ed_Volt_Basa_Controller.Text:=FloatToStr(v_Motor_Basa)+' volt';
{end of input Motor Basa}

{---------- Begin Of Input Sensor ----------}
 Data_Inp3:=Input('Sensor');
 Bin[x-1]:=Data_Inp3;
 Filt_Bin[x-1]:=Data_Inp3;
 if spSkinCheckRadioBox1.Checked=True then
 begin
  if Filt_Bin[x-1]-Filt_Bin[x-4]>4 then
    Filt_Bin[x-1]:=Filt_Bin[x-2];
 end;
 pH[x-1]:=((Data_Inp3/255)*11)+1; {untuk menampilkan nilai pH pada grafik dg range 1 - 12}
 ed_PH_PV.Text:=FloatToStr(pH[x-1]);
 ed_nilai_pH.Caption:=ed_PH_PV.Text;
 Frm_Plant.lbl_nilai_pH.Caption:=ed_PH_PV.Text;
 v_pH:=(Data_Inp3/255)*5;
 ByteToBit(Data_Inp3);
 ed_Bin_PV.Text:=IntToStr(Bit[7])+IntToStr(Bit[6])+IntToStr(Bit[5])
 +IntToStr(Bit[4])+IntToStr(Bit[3])+IntToStr(Bit[2])+IntToStr(Bit[1])+IntToStr(Bit[0]);
 ed_Volt_PV.Text:=FloatToStr(v_pH);

 if x>25 then
  begin
   Chart1.BottomAxis.SetMinMax(0,x+6);
   Chart2.BottomAxis.SetMinMax(0,x+6);
  end;
  Series1.AddXY(x,sp_Motor_Asam[x-1],'',clBlue);   {Set Point Bukaan Valve Asam}
  Series2.AddXY(x,Persen_Motor_Asam[x-1],'',clRed); {Persen Bukaan Valve}
  Series3.AddXY(x,Sp_Motor_Basa[x-1],'',clFuchsia); {Set Point Bukaan Valve Asam}
  Series4.AddXY(x,Persen_Motor_Basa[x-1],'',clGreen); {Persen Bukaan Valve}
  series5.AddXY(x,pH[x-1],'',clYellow);
  Series6.AddXY(x,Sp_Motor_Basa[x-1],'',clred);
  Series7.AddXY(x,Persen_Motor_Basa[x-1],'',clGreen); {Persen Bukaan Valve}
  Series8.AddXY(x,SET_POINT[x-1],'',clRed);
  series9.AddXY(x,pH[x-1],'',clYellow);
{end untuk Input Sensor jika ONLINE di centang}

end;
{---------- END OF ONLINE ----------}
```

```
end;

procedure TfrmTESADC.spSkinXFormButtonSAVEClick(Sender: TObject);
var
 FDATA:TextFile;
 j:Integer;
begin
 Timer_Sampling.Enabled:=False;
 if SaveDialog1.Execute then
  if CheckBox_OpenLOOP.Checked=true then
   begin
    AssignFile(FDATA,SaveDialog1.FileName);
    Rewrite(FDATA);
    Writeln(FData,' Time  SP Valve Asam  Bukaan Valve Asam  SP Valve Basa  Bukaan Valve Basa  pH');
    Writeln(FData);
    for j:=0 to x-1 do
     Writeln(FData,(j+1):4,'      ',SP_MOTOR_ASAM[j],' %          ',
     PERSEN_MOTOR_ASAM[j]:5:0,' %          ',SP_MOTOR_BASA[j],' %          ',
     PERSEN_MOTOR_BASA[j]:5:0,' %      ',pH[j]:2:0);
     CloseFile(FDATA);
    SaveDialog3.Title:='Save Grafik Bukaan Valve As';
    SaveDialog4.Title:='Save Grafik pH As';
   end
  else
  begin
   AssignFile(FDATA,SaveDialog1.FileName);
   Rewrite(FDATA);
   Writeln(FDATA,'Time  SP Valve Asam  Bukaan Valve Asam  SP Valve Basa  Bukaan Valve Basa  pH');
   Writeln(FDATA);
   for j:=0 to x-1 do
    Writeln(FDATA,(j+1):4,'      ',SP_MOTOR_ASAM[j],' %          ',
    PERSEN_MOTOR_ASAM[j]:5:0,' %          ',SP_MOTOR_BASA[j],' %          ',
    PERSEN_MOTOR_BASA[j]:5:0,' %          ',pH[j]:2:0);
    CloseFile(FDATA);
  end;

 if SaveDialog3.Execute then
  begin
   Chart1.SaveToMetafileEnh(SaveDialog3.FileName);
  end;
 if SaveDialog4.Execute then
  Chart2.SaveToMetafileEnh(SaveDialog4.FileName);
end;

procedure TfrmTESADC.spSkinXFormButtonCLEARClick(Sender: TObject);
var i:Integer;
begin
 Series1.Clear;
 Series2.Clear;
 Series3.Clear;
 Series4.Clear;
 series5.Clear;
  for i:=0 to x-1 do
   begin
    Persen_Motor_Asam[i]:=0;
    Persen_Motor_Basa[i]:=0;
    pH[i]:=0;
    sp_Motor_Asam[i]:=0;
    sp_Motor_Basa[i]:=0;
   end;
 Timer_Sampling.Enabled:=False;
 Chart1.BottomAxis.SetMinMax(0,100);
 Chart2.BottomAxis.SetMinMax(0,100);
 x:=0;
end;

procedure TfrmTESADC.Button_BrowseClick(Sender: TObject);
begin
 if Edit_Basa.Text='' then
  begin
```

```
  spSkinOpenDialog1.Title:='Open Data Persentase Bukaan Valve Basa';
  spSkinOpenDialog1.Execute;
  Edit_Basa.Text:=spSkinOpenDialog1.FileName;
  end
 else
  begin
   spSkinOpenDialog1.Title:='Open Data Persentase Bukaan Valve Asam';
   spSkinOpenDialog1.Execute;
   Edit_Asam.Text:=spSkinOpenDialog1.FileName;
  end;
end;

procedure TfrmTESADC.Button_Load_ASAMClick(Sender: TObject);
var
 FData:TextFile;
 Data:String;
 i:integer;
begin
 spSkinStringGrid1.Cells[2,0]:='Valve Asam (%)';
 spSkinStringGrid1.Cells[3,0]:='pH';
 if Edit_Asam.Text='' then MessageDlg('Anda belum menentukan direktori dan nama
filenya.',mtInformation,[mbOK],0)
  else
   begin
    AssignFile(FData,Edit_Asam.Text);
    reset(FData);
    i:=0;
     repeat
      inc(i);
      spSkinStringGrid1.RowCount:=i+1;
      Readln(FData,data);
      spSkinStringGrid1.Cells[2,i]:=data;
      until
       spSkinStringGrid1.Cells[1,i]='';
       spSkinStringGrid1.RowCount:=i;
       spSkinStringGrid1.Cells[0,i]:='';
       MessageDlg('Loading data telah selesai, jumlah data : ' + IntToStr(i-1),mtInformation,[mbOK],0);
       jmlData:=i-1;
       CloseFile(FData);
       spSkinXFormButtonRUN.Enabled:=True;
  end;
end;

procedure TfrmTESADC.Button_Load_BASAClick(Sender: TObject);
var
 FData:TextFile;
 Data:String;
 i:integer;
begin
 spSkinStringGrid1.Cells[0,0]:='No.';
 spSkinStringGrid1.Cells[1,0]:='Valve Basa (%)';
 if Edit_Basa.Text='' then MessageDlg('Anda belum menentukan direktori dan nama
filenya.',mtInformation,[mbOK],0)
  else
   begin
    AssignFile(FData,Edit_Basa.Text);
    reset(FData);
    i:=0;
     repeat
      inc(i);
      spSkinStringGrid1.RowCount:=i+1;
      Readln(FData,data);
      spSkinStringGrid1.Cells[0,i]:=IntToStr(i);
      spSkinStringGrid1.Cells[1,i]:=data;
      until
       spSkinStringGrid1.Cells[1,i]='';
       spSkinStringGrid1.RowCount:=i;
       spSkinStringGrid1.Cells[0,i]:='';
       MessageDlg('Loading data telah selesai, jumlah data : ' + IntToStr(i-1),mtInformation,[mbOK],0);
       jmlData:=i-1;
```

```
        CloseFile(FData);
    end;
end;

procedure TfrmTESADC.BUTTON_RESETClick(Sender: TObject);
VAR i:integer;
begin
  for i:=0 to x-1 do
    begin
      Persen_Motor_Asam[i]:=0;
      Persen_Motor_Basa[i]:=0;
      pH[i]:=0;
      sp_Motor_Asam[i]:=0;
      sp_Motor_Basa[i]:=0;
      spSkinStringGrid1.Cells[0,i]:='';
    spSkinStringGrid1.Cells[1,i]:='';
    spSkinStringGrid1.Cells[2,i]:='';
    spSkinStringGrid1.Cells[3,i]:='';
     end;
 Chart1.BottomAxis.SetMinMax(0,100);
 Chart2.BottomAxis.SetMinMax(0,100);
 x:=0;
 Timer_Sampling.Enabled:=False;

 spSkinEdit2.Text:=spSkinStringGrid1.Cells[0,x];
 spSkinEdit3.Text:=spSkinStringGrid1.Cells[1,x];
 spSkinEdit4.Text:=spSkinStringGrid1.Cells[2,x];
 spSkinEdit5.Text:=spSkinStringGrid1.Cells[3,x];
 Series1.Clear;
 Series2.Clear;
 Series3.Clear;
 Series4.Clear;
 Series5.Clear;
end;

procedure TfrmTESADC.bt_MENU_CONTROLLERClick(Sender: TObject);
begin
 Form1.Show;
 Frm_Training.Hide;
 frmTESADC.Hide;
end;

procedure TfrmTESADC.bt_LOAD_CONTROLLERClick(Sender: TObject);
begin
with Form_Load do
  begin
    Show;
    Caption:='Load Bobot Vij dan Bobot Wjk';
    Group_In.Caption:='Bobot Vij';
    Group_Out.Caption:='Bobot Wjk';
    bt_JST_Control.Visible:=True;
    bt_Validasi_Data.Visible:=False;
    bt_LOAD_BOBOT.Visible:=False;
    Button_Training.Visible:=False;
    Open_Dialog_In.Title:='Open Bobot Vij';
    Open_Dialog_Out.Title:='Open Bobot Wjk';
  end;
end;

procedure TfrmTESADC.bt_RUN_CONTROLLERClick(Sender: TObject);
var i,j,k:integer;
begin
 bt_RESET_CONTROLLER.Enabled:=False;

 if CheckRadioBox_ONLINE.Checked=False then
   begin
     MessageDlg('Click dulu check box ONLINE'
                +' sebelum mulai!',mtInformation,[mbOK],0);
    Timer_Sampling.Enabled:=False;
    Timer_Motor.Enabled:=False;
```

```
  end
 else
  if (Stringgrid_Bobot_Input.Cells[1,1]='') or
     (Stringgrid_Bobot_Hdden.Cells[1,1]='') then
     MessageDlg('Masukkan Bobot Vij dan Bobot'
                +' Wjk terlebih dahulu.',mtInformation,[mbOK],0)
  else
  begin

     for i:=1 to StrToInt(Frm_Training.Ed_Input.Text)+1 do
      for j:=1 to StrToInt(Frm_Training.Ed_Hiden.Text) do
       begin
         Vij[i,j]:=StrToFloat(Stringgrid_Bobot_Input.Cells[i,j]);
       end;

     for i:=1 to StrToInt(Frm_Training.Ed_Output.Text) do
      for j:=1 to StrToInt(Frm_Training.Ed_Hiden.Text)+1 do
       begin
         Wjk[k,j]:=StrToFloat(Stringgrid_Bobot_Hdden.Cells[k,j]);
       end;
  end;
 Timer_Motor.Enabled:=True;
 Timer_Sampling.Enabled:=True;
 Chart3.BottomAxis.Automatic:=True;
 Chart4.BottomAxis.Automatic:=True;
 bt_RUN_CONTROLLER.Enabled:=False;
end;

procedure TfrmTESADC.bt_STOP_CONTROLLERClick(Sender: TObject);
begin
d:=False;
 Timer_Sampling.Enabled:=False;
 Timer_Motor.Enabled:=False;
 bt_RESET_CONTROLLER.Enabled:=True;
 bt_RUN_CONTROLLER.Enabled:=True;
end;

procedure TfrmTESADC.bt_RESET_CONTROLLERClick(Sender: TObject);
var j:integer;
begin
 for j:=1 to StrToInt(Frm_Training.Ed_Hiden.Text) do
   begin
   Stringgrid_Bobot_Input.Cells[StrToInt(Frm_Training.Ed_Input.Text),j]:='';
   Stringgrid_Bobot_Input.Cells[StrToInt(Frm_Training.Ed_Input.Text)+1,j]:='';
   Stringgrid_Bobot_Hdden.Cells[StrToInt(Frm_Training.Ed_Output.Text),j]:='';
   Stringgrid_Bobot_Hdden.Cells[StrToInt(Frm_Training.Ed_Output.Text),j+1]:='';
   end;

 Chart3.BottomAxis.SetMinMax(0,1);
 Chart4.BottomAxis.SetMinMax(0,1);
 ed_Sinyal_Kontrol.Text:='0';
 ed_PH_PV.Text:='0';
 ed_Bin_PV.Text:='0000000';
 ed_Volt_PV.Text:='0';
 ed_ERROR.Text:='0';
 ed_Valve_Basa_Contoller.Text:='0 %';
 ed_Bin_Basa_Controller.Text:='00000000';
 ed_Volt_Basa_Controller.Text:='0 Volt';
 Series6.Clear;
 Series7.Clear;
 Series8.Clear;
 Series9.Clear;
 ComboBox_SETPOINT.ItemIndex:=2;
 bt_RUN_CONTROLLER.Enabled:=True;
 x:=0;
end;

procedure TfrmTESADC.bt_SAVE_CONTROLLERClick(Sender: TObject);
var FData:TextFile;
   j:integer;
```

```
begin
 Timer_Sampling.Enabled:=False;
 if SaveDialog2.Execute then
   begin
    AssignFile(FData,SaveDialog2.FileName);
    Rewrite(FDATA);
     Writeln(FData,' time  Set Point  Process Variable  ERROR     Bukaan Valve Basa ');
     Writeln(FData,' (dt)    [pH]        [pH]                 %          ');
     Writeln(FData);
     for j:=0 to x-1 do
      Writeln(FData,(j+1):4,'      ',SET_POINT[j]:2:0,'          ',
      pH[j]:5:2,'      ',ERROR_PLANT[j]:5:2,'            ',Sp_Motor_Basa[j]);
      CloseFile(FDATA);
    SaveDialog3.Title:='Save Grafik Sinyal Control As';
    SaveDialog4.Title:='Save Grafik SP & PV As';
   end;

 if SaveDialog3.Execute then
    Chart3.SaveToMetafileEnh(SaveDialog3.FileName);
 if SaveDialog4.Execute then
    Chart4.SaveToMetafileEnh(SaveDialog4.FileName);

end;


procedure TfrmTESADC.Image1Click(Sender: TObject);
begin
 Frm_Training.Hide;
 Frm_Plant.Label_Menu_Utama.Hide;
 Frm_Plant.Label_OnLine.Show;
 Frm_Plant.Show;
 frmTESADC.Hide;
end;

end.
```

LAMPIRAN D
Listing Program Unit_IO dan Unit_Motor_Positioner

Listing Program Unit_IO

```
unit Unit_IO;

interface
function Input(ADC:String):Integer;
procedure Output(Alamat:Integer;Data:Integer);
function CableConnection(pin:integer):String;

implementation

uses SysUtils;

function PortIn(Port:Word):Byte;stdcall;external'io.dll';

procedure PortOut(Port:Word;Data:Byte);stdcall;external'io.dll';

function Input(ADC:String):Integer;
var tmp1, tmp2, tmp3,inv:Integer;
begin
 {------ ADC 2 ----------}
 if ADC='Motor_Asam' then
  begin
   PortOut($37A,$08);
   tmp1:=PortIn($379) xor $80;
   tmp1:=(tmp1 shr 4) and $0F;
   PortOut($37A,$09);
   Input:=tmp1 or ((PortIn($379) xor $80) and $F0);
  end;

 {------ ADC 3 ---------}
 if ADC='Motor_Basa' then
  begin
   PortOut($37A,$02);
   tmp2:=PortIn($379) xor $80;
   tmp2:=(tmp2 shr 4) and $0F;
   PortOut($37A,$03);
   Input:=tmp2 or ((PortIn($379) xor $80) and $F0);
  end;

 {------ ADC 1 --------}
 if ADC='Sensor' then
  begin
   PortOut($37A,$0A);
   tmp3:=PortIn($379) xor $70;
   tmp3:=(tmp3 shr 4) and $0F;
   PortOut($37A,$0B);
   input:=tmp3 or ((PortIn($379) xor $70) and $F0);
   {input:=not(inv);}
  end;
end;

procedure Output(Alamat:Integer;Data:Integer);
begin
 PortOut(Alamat,Data);
end;

function CableConnection(pin:integer):string;
var temp:integer;
begin
 if pin=15 then
  begin
```

```
    temp:=Portin($379) and $08;
    if temp=8 then CableConnection:=' Connected' else
      CableConnection:=' Disconnected';
  end;
end;

end.
```

Listing Unit_Motor_Positioner

```
unit Unit_Motor_Positioner;

interface

uses Unit_IO;

function MinMax_Pos_Motor_Asam(x:String):Integer;
function MinMax_Pos_Motor_Basa(y:String):Integer;
function StatusMotor_Asam():String;
function StatusMotor_Basa():String;
Procedure Pos_Motor_Asam(Posisi_Asam:Integer;A7:boolean);
Procedure Pos_Motor_Basa(Posisi_Basa:Integer;B6:boolean);

var Bit,BitOut:array[0..7] of Integer;
   Left_Asam, Right_Asam, Left_Basa, Right_Basa, Bukaan_Asam, Bukaan_Basa:Integer;

implementation

uses SysUtils;

{Fungsi untuk merubah bilangan biner 8 bit ke bilangan desimal}
{-------------------------------------------------}

function BitToByte(bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7:Integer):Integer;
begin
 BitToByte:=bit0*1+bit1*2+bit2*4+bit3*8+bit4*16+bit5*32+bit6*64+bit7*128;
end;

{Prosedur untuk merubah bilangan desimal menjadi bilangan biner 8 bit}
{-------------------------------------------------}

Procedure ByteToBit(nilai:Integer);
begin
 if nilai>=128 then
  begin
   bit[7]:=1;
   nilai:=nilai-128;
  end
  else
   begin
    bit[7]:=0;
   end;
 if nilai>=64 then
  begin
   bit[6]:=1;
   nilai:=nilai-64;
  end
  else
   begin
    bit[6]:=0;
   end;
 if nilai>=32 then
  begin
   bit[5]:=1;
   nilai:=nilai-32;
  end
  else
```

```
    begin
      bit[5]:=0;
    end;
  if nilai>=16 then
    begin
      bit[4]:=1;
      nilai:=nilai-16;
    end
    else
      begin
        bit[4]:=0;
      end;
  if nilai>=8 then
    begin
      bit[3]:=1;
      nilai:=nilai-8;
    end
    else
      begin
        bit[3]:=0;
      end;
  if nilai>=4 then
    begin
      bit[2]:=1;
      nilai:=nilai-3;
    end
    else
      begin
        bit[2]:=0;
      end;
  if nilai>=2 then
    begin
      bit[1]:=1;
      nilai:=nilai-2;
    end
    else
      begin
        bit[1]:=0;
      end;
  if nilai=1 then
    begin
      bit[0]:=1;
    end
    else
      begin
        bit[0]:=0;
      end;
end;

{Fungsi-fungsi dan Prosedur-prosedur untuk mengatur Valve Asam}
{-------------------------------------------------------}

function MinMax_Pos_Motor_Asam(x:String):Integer;
var
  min_ASAM,max_ASAM:Integer;
  FData:TextFile;
begin
  AssignFile(FData,ExtractFilePath(ParamStr(0))+'\Valve_Asam.txt');
    Reset(FData);
    Readln(FData,min_ASAM);
    Readln(FData,max_ASAM);
    CloseFile(FData);
    if x='min' then Result:=min_ASAM;
    if x='max' then Result:=max_ASAM;
end;

Procedure Pos_Motor_Asam(Posisi_Asam:Integer;A7:Boolean);
var ref,ref_posisi:Integer;
    bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7:Integer;
begin
```

```
 if Posisi_Asam<MinMax_Pos_Motor_Asam('min') then
  Posisi_Asam:=MinMax_Pos_Motor_Asam('min');
 if Posisi_Asam>MinMax_Pos_Motor_Asam('max') then
  Posisi_Asam:=MinMax_Pos_Motor_Asam('max');

  Bukaan_Asam:=Posisi_Asam;
  if A7=True Then bit7:=1 else bit7:=0;
  bit6:=0;
  bit5:=0;
  bit4:=0;
  bit3:=0;
  bit2:=0;
  bit1:=0;
  bit0:=0;
  ref:=Input('Motor_Asam');
  ref_posisi:=ref;
   if (ref_posisi<=105) then bit4:=0;
   if (ref_posisi>=217) then bit5:=0;
   if (ref_posisi>Posisi_Asam) then bit4:=1;
   if (ref_posisi<Posisi_Asam) then bit5:=1;
   if (ref_posisi>=Posisi_Asam-3) and (ref_posisi<=Posisi_Asam+3) then
    begin
     bit4:=0;
     bit5:=0;
    end;

   Left_Asam:=bit4;
   Right_Asam:=bit5;

 Output($378,BitToByte(bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7));
end;

function StatusMotor_Asam():String;
var temp,min_ASAM,max_ASAM:Integer;
begin
 min_ASAM:=MinMax_Pos_Motor_Asam('min');
 max_ASAM:=MinMax_Pos_Motor_Asam('max');
 temp:=Round(((Bukaan_Asam-min_ASAM)/(max_ASAM-min_ASAM))*100);
  if Left_Asam=0 then
   begin
    if Right_Asam=0 then
     Result:=' '+IntToStr(temp)+' %Open'
    else
     Result:=' Closing';
   end
    else
     Result:=' Opening';
end;

{Fungsi-fungsi dan Prosedur-prosedur untuk mengatur Valve Basa}

function MinMax_Pos_Motor_Basa(y:String):Integer;
var
 min_BASA,max_BASA:Integer;
 FData:TextFile;
begin
 AssignFile(FData,ExtractFilePath(ParamStr(0))+'\Valve_Basa.txt');
  Reset(FData);
  Readln(FData,min_BASA);
  Readln(FData,max_BASA);
  CloseFile(FData);
  if y='min' then Result:=min_BASA;
  if y='max' then Result:=max_BASA;
end;

Procedure Pos_Motor_Basa(Posisi_Basa:Integer;B6:Boolean);
var ref,ref_posisi:Integer;
  bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7:Integer;
begin
```

```
 if Posisi_Basa<MinMax_Pos_Motor_Basa('min') then
   Posisi_Basa:=MinMax_Pos_Motor_Basa('min');
 if Posisi_Basa>MinMax_Pos_Motor_Basa('max') then
   Posisi_Basa:=MinMax_Pos_Motor_Basa('max');
   Bukaan_Basa:=Posisi_Basa;
   bit7:=0;
   if B6=True then bit6:=1 else bit6:=0;
   bit5:=0;
   bit4:=0;
   bit3:=0;
   bit2:=0;
   bit1:=0;
   bit0:=0;
   ref:=Input('Motor_Basa');
   ref_posisi:=ref;
     if (ref_posisi<=105) then bit2:=0;
     if (ref_posisi>=217) then bit3:=0;
     if (ref_posisi>Posisi_Basa) then bit2:=1;
     if (ref_posisi<Posisi_Basa) then bit3:=1;
     if (ref_posisi>=Posisi_Basa-3) and (ref_posisi<=Posisi_Basa+3) then
       begin
         bit2:=0;
         bit3:=0;
       end;

     Left_Basa:=bit2;
     Right_Basa:=bit3;

  Output($378,BitToByte(bit0,bit1,bit2,bit3,bit4,bit5,bit6,bit7));
end;

function StatusMotor_Basa():String;
var temp,min_BASA,max_BASA:Integer;
begin
  min_BASA:=MinMax_Pos_Motor_Basa('min');
  max_BASA:=MinMax_Pos_Motor_Basa('max');
  temp:=Round(((Bukaan_Basa-min_BASA)/(max_BASA-min_BASA))*100);
    if Left_Basa=0 then
      begin
        if Right_Basa=0 then
          Result:=' '+IntToStr(temp)+' %Open'
        else
          Result:=' Closing';
      end
        else
          Result:=' Opening';
end;

end.
```

LAMPIRAN E
DATA TRAINING JST

(82 Pasangan Data Error dan Persen Bukaan Valve Basa)

Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)
1	2	-23	10	4	31	-2	30	-1	31
1	2	-23	7	5	31	-2	26	0	31
23	2	-22	7	5	34	-1	27	0	30
24	51	-21	10	4	34	-1	30	-1	30
23	55	-20	7	3	35	0	30	0	31
20	54	-18	7	1	34	0	30	0	30
17	51	-16	14	0	30	1	31	1	31
12	50	-14	14	-1	30	0	30	0	6
5	42	-12	15	-1	27	0	27	0	2
0	34	-9	15	-2	27	-1	31	0	2
-6	31	-7	18	-2	27	-1	31		
-10	23	-5	23	-3	26	0	31		
-14	22	-3	23	-4	27	0	30		
-17	18	-1	27	-5	26	-1	30		
-19	11	0	27	-5	26	-1	30		
-21	11	1	27	-4	27	0	30		
-21	11	2	30	-4	23	0	30		
-22	7	3	30	-3	27	-1	30		

Data Training JST didapat dari data close loop (Fahmy, 2006). Nilai error didapat dari persamaan berikut:

$$\text{Error} = \frac{setpoint - process\,variabel}{spansensor} x100\%$$

E-2

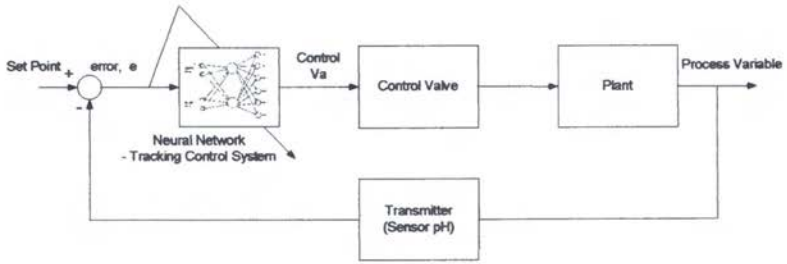
Dengan nilai setpoint pH = 6,5
dan span sensor (pH 1 – pH 12) = 11
JST hanya bisa menerima sinyal masukan antara –1 sampai dengan +1. Agar data error bisa digunakan untuk proses training, maka data tersebut harus dinormalisasi dengan cara dibagi dengan data error terbesar yakni dibagi dengan angka 24. Hal yang sama berlaku juga untuk data persen bukaan valve, sehingga data persen bukaan valve dinormalisasi dengan cara dibagi dengan data persen bukaan valve terbesar yakni dibagi dengan angka 55. Sehingga hasil normalisasinya adalah sebagai berikut:

Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)	Error (%)	Bukaan Valve Basa (%)
0.0417	0.0364	-0.9583	0.1818	0.1667	0.5636	-0.0833	0.5455	-0.0417	0.5636
0.0417	0.0364	-0.9583	0.1273	0.2083	0.5636	-0.0833	0.4727	0.0000	0.5636
0.9583	0.0364	-0.9167	0.1273	0.2083	0.6182	-0.0417	0.4909	0.0000	0.5455
1.0000	0.9273	-0.8750	0.1818	0.1667	0.6182	-0.0417	0.5455	-0.0417	0.5455
0.9583	1.0000	**-0.8333**	0.1273	**0.1250**	0.6364	0.0000	0.5455	0.0000	**0.5636**
0.8333	0.9818	-0.7500	0.1273	0.0417	0.6182	0.0000	0.5455	0.0000	0.5455
0.7083	0.9273	-0.6667	0.2545	0.0000	0.5455	0.0417	0.5636	0.0417	0.5636
0.5000	0.9091	-0.5833	0.2545	-0.0417	0.5455	0.0000	0.5455	0.0000	0.1091
0.2083	0.7636	-0.5000	0.2727	-0.0417	0.4909	0.0000	0.4909	0.0000	0.0364
0.0000	0.6182	-0.3750	0.2727	-0.0833	0.4909	-0.0417	0.5636	0.0000	0.0364
-0.2500	0.5636	-0.2917	**0.3273**	-0.0833	**0.4909**	-0.0417	0.5636		
-0.4167	0.4182	-0.2083	0.4182	-0.1250	0.4727	0.0000	0.5636		
-0.5833	0.4000	**-0.**1250	0.4182	-0.1667	0.4909	0.0000	0.5455		
-0.7083	0.3273	-0.0417	0.4909	-0.2083	0.4727	-0.0417	0.5455		
-0.7917	0.2000	0.0000	0.4909	-0.2083	0.4727	-0.0417	0.5455		
-0.8750	0.2000	0.0417	0.4909	-0.1667	0.4909	0.0000	0.5455		
-0.8750	0.2000	0.0833	0.5455	**-0.1667**	**0.4182**	**0.0000**	**0.5455**		
-0.9167	0.1273	0.1250	0.5455	**-0.**1250	0.4909	-0.0417	0.5455		

LAMPIRAN F
DATA HASIL TRAINING DAN VALIDASI

no	hidden	α	μ	RMSE Training	EPOCH	RMSE Validasi	no	hidden	α	μ	RMSE Training	EPOCH	RMSE Validasi
1	6	0.05	0.7	0.03	2298	0.077271	33	20	0.05	0.7	0.03	2334	0.077422
2			0.5	0.03	2302	0.077607	34			0.5	0.03	2327	0.077108
3			0.3	0.03	2327	0.079509	35			0.3	0.03	2321	0.076766
4			0.1	0.03	2283	0.077264	36			0.1	0.03	2326	0.076736
5		0.1	0.7	0.03	2257	0.076658	37		0.1	0.7	0.03	2282	0.075776
6			0.5	0.03	2275	0.076619	38			0.5	0.03	2266	0.07584
7			0.3	0.03	2257	0.076657	39			0.3	0.03	2284	0.075772
8			0.1	0.03	2271	0.076589	40			0.1	0.03	2267	0.075981
9		0.4	0.7	0.03	2222	0.079542	41		0.4	0.7	0.03	2326	0.076133
10			0.5	0.03	2142	0.092359	42			0.5	0.03	2085	0.096146
11			0.3	0.03	2107	0.091253	43			0.3	0.03	2045	0.104757
12			0.1	0.03	2106	0.098027	44			0.1	0.03	2106	0.091822
13		0.7	0.7	0.03	2199	0.092704	45		0.7	0.7	0.03	2238	0.082895
14			0.5	0.03	2161	0.097151	46			0.5	0.03	2237	0.08293
15			0.3	0.03	2161	0.09715	47			0.3	0.03	2238	0.082913
16			0.1	0.03	2161	0.09715	48			0.1	0.03	2236	0.082975
17	10	0.05	0.7	0.03	2318	0.077633	49	30	0.05	0.7	0.03	2339	0.076682
18			0.5	0.03	2308	0.077455	50			0.5	0.03	2330	0.076064
19			0.3	0.03	2299	0.077298	51			0.3	0.03	2337	0.076069
20			0.1	0.03	2298	0.077342	52			0.1	0.03	2339	0.076512
21		0.1	0.7	0.03	2256	0.076534	53		0.1	0.7	0.03	2298	0.07574
22			0.5	0.03	2241	0.076654	54			0.5	0.03	2179	0.078107
23			0.3	0.03	2319	0.076348	55			0.3	0.03	2232	0.076431
24			0.1	0.03	2317	0.076433	56			0.1	0.03	2211	0.077217
25		0.4	0.7	0.03	2054	0.111201	57		0.4	0.7	0.03	2308	0.077251
26			0.5	0.03	2101	0.092175	58			0.5	0.03	2103	0.098054
27			0.3	0.03	2082	0.104249	59			0.3	0.03	2305	0.07741
28			0.1	0.03	2105	0.09097	60			0.1	0.03	2108	0.09458
29		0.7	0.7	0.03	2161	0.097151	61		0.7	0.7	0.03	2232	0.082755
30			0.5	0.03	2161	0.097151	62			0.5	0.03	2236	0.082652
31			0.3	0.03	2161	0.097152	63			0.3	0.03	2233	0.082724
32			0.1	0.03	2161	0.128873	64			0.1	0.03	2234	0.082721

LAMPIRAN G
BLOK DIAGRAM SISTEM PENGENDALIAN pH DAN FUNGSI TRANSFER



Fungsi Transfer Transmitter (Sensor pH):

$$\frac{Gt}{1+\tau s}$$

dengan, $Gt = \frac{\Delta Out}{\Delta in} = \frac{0{,}564(volt)}{11(pH)} = 0{,}0512$

$\tau = 5 \times 0{,}623 = 3{,}115$

Sehingga Fungsi Transfer Transmitter (Sensor pH) adalah:

$$\frac{0{,}0512}{1+3{,}115s}$$

Fungsi Transfer Control Valve Basa:

$$\frac{Gv}{1+\tau s}$$

dengan, $Gv = 1$

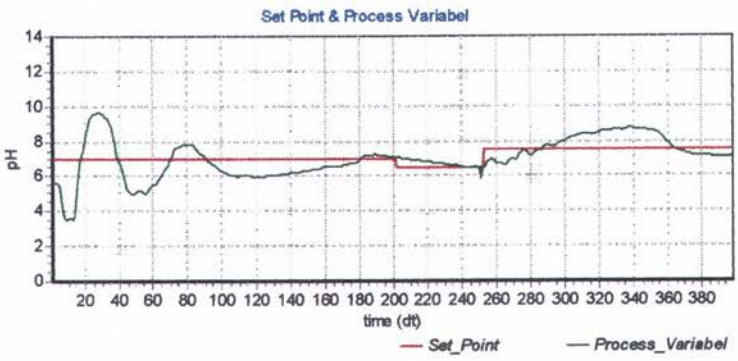
$\tau = 4 \times 0{,}623 = 2{,}492$

Sehingga Fungsi Transfer Control Valve Basa:

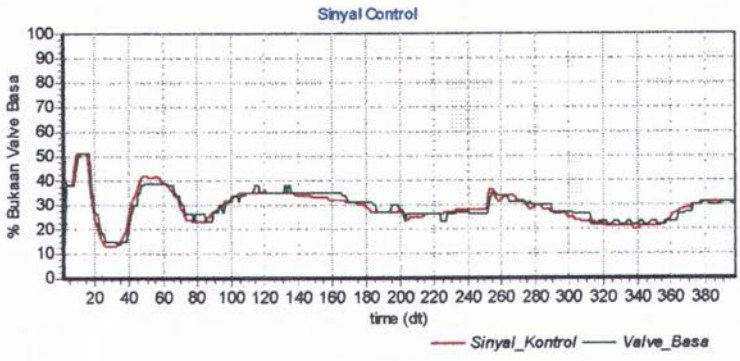
$$\frac{1}{1+2{,}492s}$$

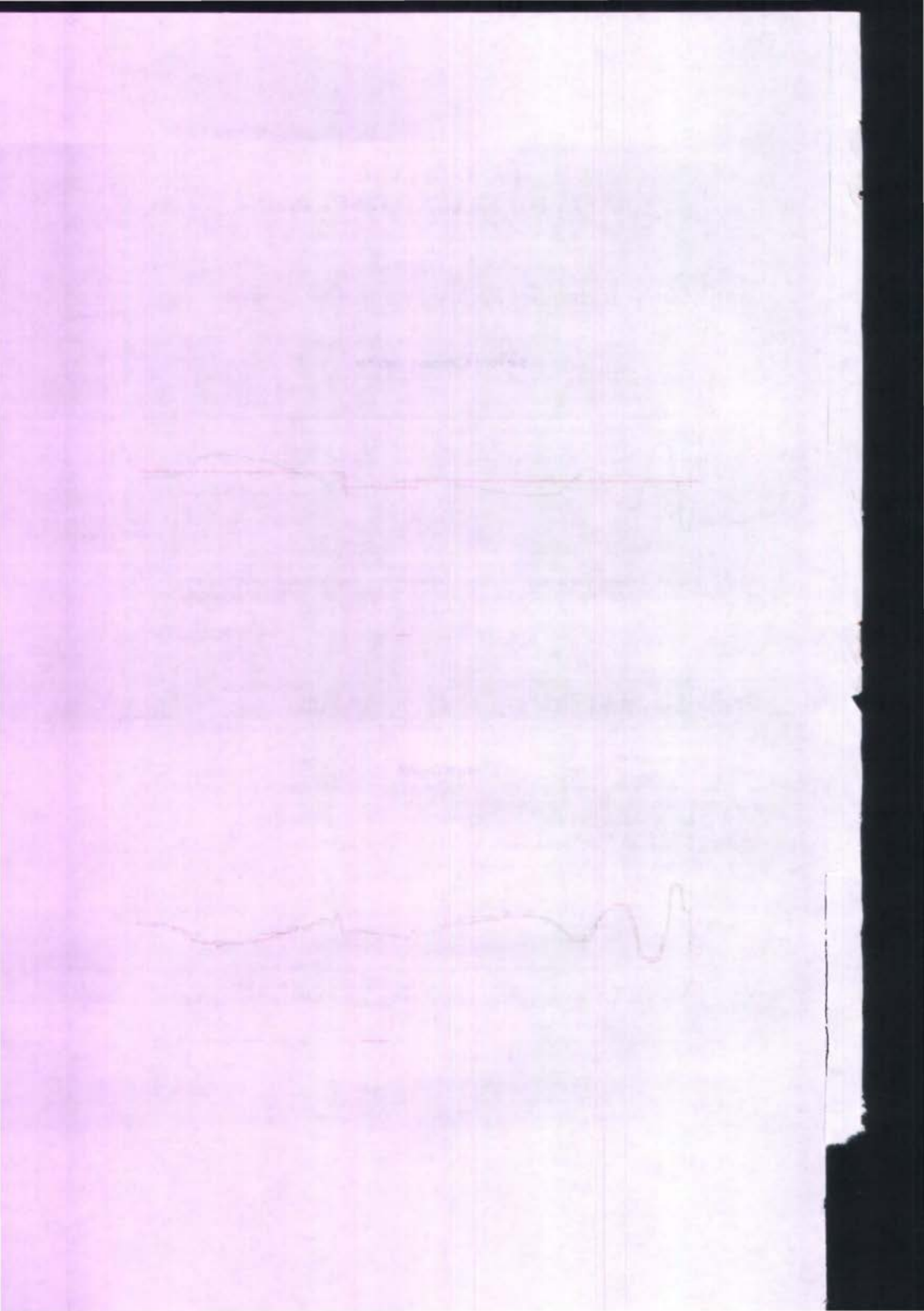
LAMPIRAN H
GRAFIK UJI TRACKING SET POINT

Grafik Uji Tracking Set Point



Grafik Sinyal Kontrol Uji Tracking Set Point



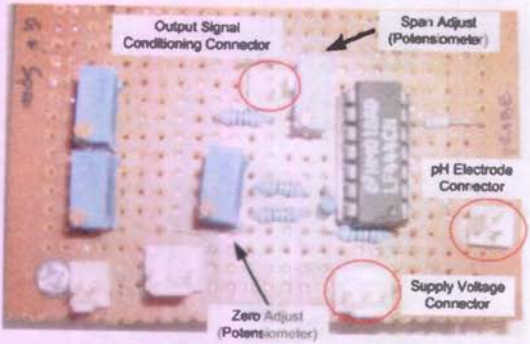


LAMPIRAN I
GAMBAR KOMPONEN MINIPLANT SISTEM PENGENDALIAN pH

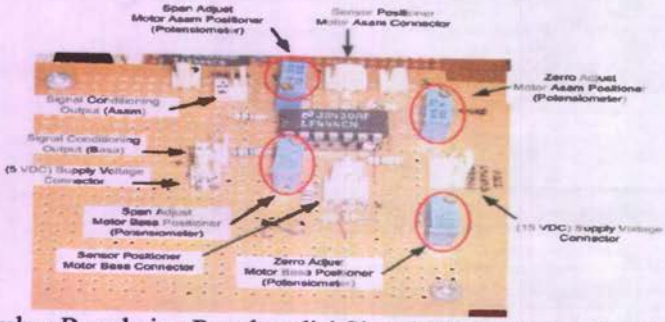
Gambar pH Electrode Type PE-11 (Sensor pH)

Tabel Datasheet pH Electrode Type PE-11

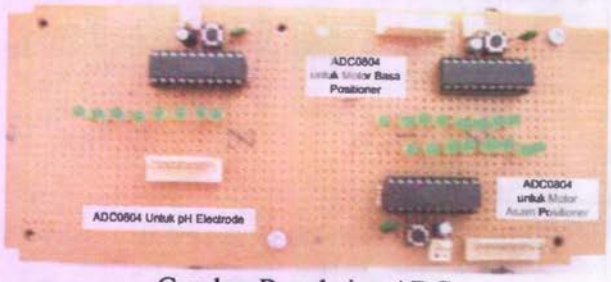
Features	Fast Response time, general purpose, laboratory & field usage
Measuring Range	1 to 13 pH (typical 0 to 14 pH)
Measuring Temerature	5 to 60C (41 to 140 F)
Electrode Structure	Combination Type
Zero Potential for pH Value	71 pH
Body Material	Epoxy
Connector	BNC
Mechanical Protection	With protection bottle on the electrode head
Dimensions	Body length = 130 mm Body Diameter = 10 mm Cable Length = 1000 mm
Applications	Water Conditioning, aquariums, beverage, fish, hatcheries, food processing, photography, laboratory, paper industry, plating industry, quality control, school & college



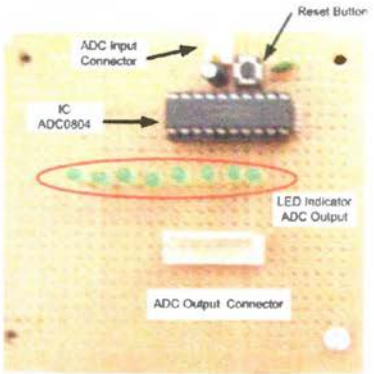
Gambar Rangkaian Pengkondisi Sinyal Untuk pH Electrode



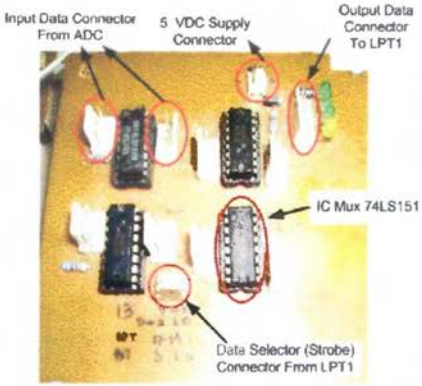
Gambar Rangkaian Pengkondisi Sinyal Untuk Motor Positioner



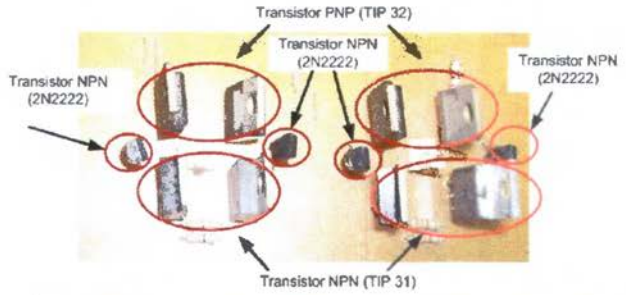
Gambar Rangkaian ADC



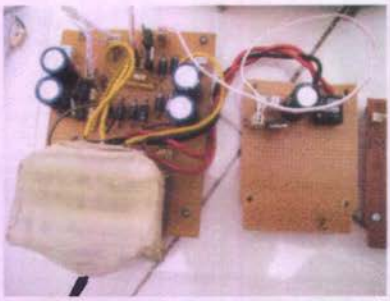
Gambar Rangkaian ADC Detail



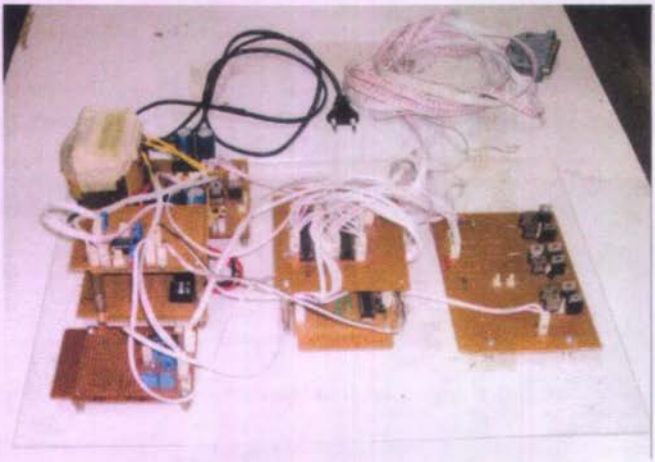
Gambar Rangkaian Multiplexer 74LS151



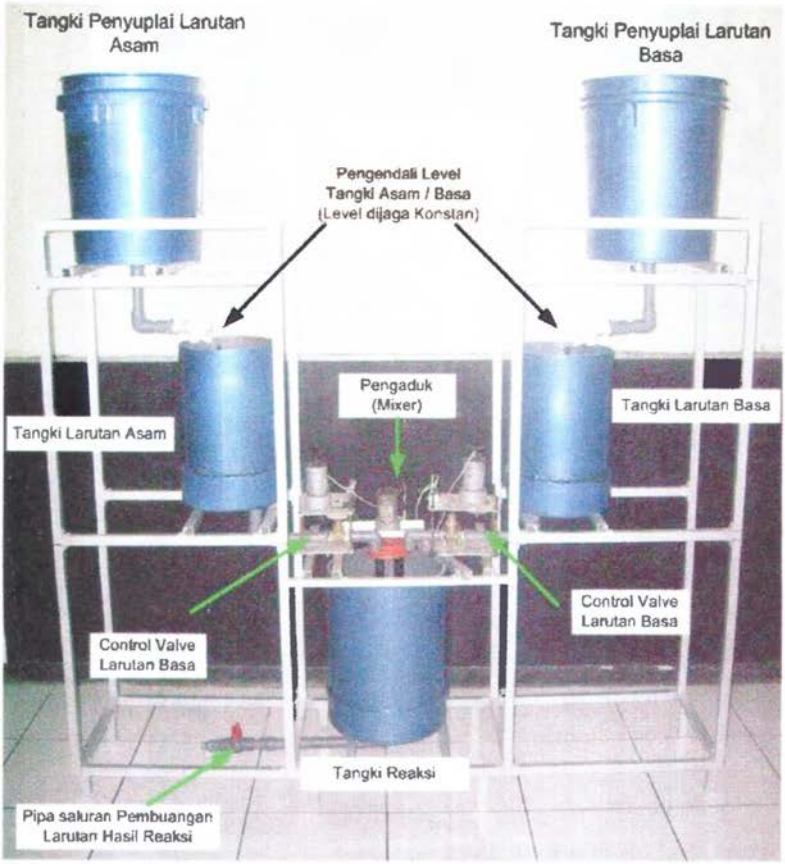
Gambar Rangkaian Driver Motor Asam dan Motor Basa



Gambar Rangkaian Power Supply



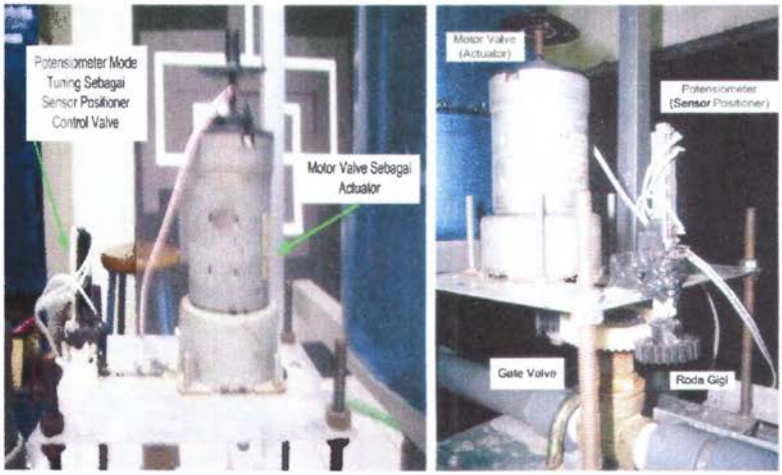
Gambar Rangkaian Tugas Akhir



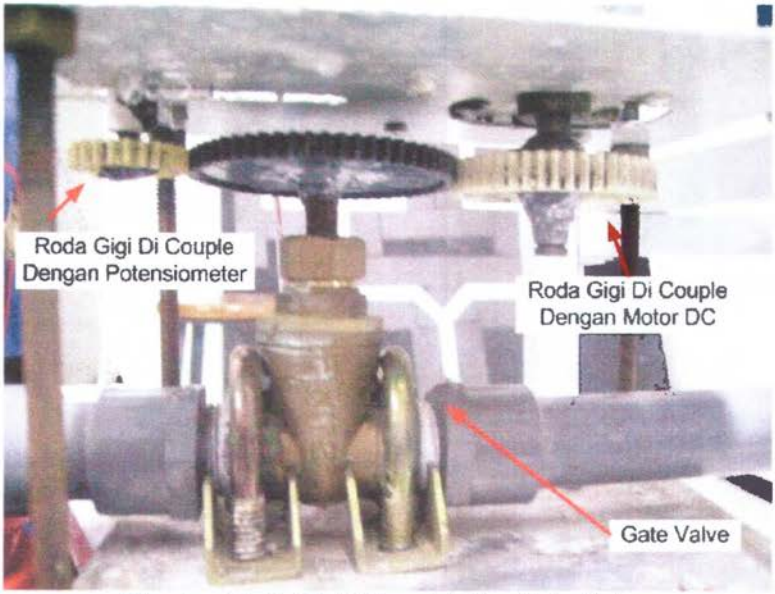
Gambar Miniplant Sistem Pengendalian pH

Gambar Control Valve dan Mixer

Gambar Control Valve



Gambar Motor Valve (Actuator) dan Potensiometer (Sensor Positioner)



Gambar Roda Gigi (Couple) dan Gate Valve

Gambar Pengaduk (Mixer)

Gambar Pengendali Level Tangki Larutan Asam dan Basa

LAMPIRAN J
PEMBUATAN LARUTAN ASAM DAN LARUTAN BASA

Pembuatan Larutan Asam

$$\rho_{HCl} = 1.19 \frac{gr}{cm^3} = 1.19 \frac{gr}{ml} = 1.19 \, x \, 1000 \frac{gr}{l} \quad ; \text{Mr HCl} = 36{,}46$$

Jika kadar HCl pekat adalah 32 %, maka nilai Molaritasnya (Konsentrasi) adalah sebagai berikut:

$$M_{HCl} = \frac{\rho_{HCl} \, x \, persen \, kadar HCl}{Mr \, HCl} = \frac{1.19 gr \, ml^{-1} \, x 32\%}{36.46 gr \, mol^{-1}}$$

$$M_{HCl} = \frac{1.19 gr \, l^{-1} \, x \, 32 \, x \, 1000}{36.46 \, gr \, mol^{-1} \, x \, 100} = \frac{1.19 \, x \, 32 \, x \, 10 \, mol}{36.46 \, l} = 10.44 M$$

Jika ingin membuat 10 liter larutan Asam (HCl) dengan konsentrasi 0,1M maka dapat dicari dengan cara sebagai berikut:

$V_1.M_1 = V_2.M_2$

dengan,

V_1 = Volume HCl yang akan diencerkan (liter)
V_2 = Volume larutan HCl yang diinginkan (liter)
M_1 = Molaritas (Konsentrasi) HCl yang akan diencerkan
M_2 = Konsentrasi HCl yang diinginkan

V_1 x 10.44M = 10 liter x 0.1M
V_1 = 0.09578 liter = 95.78 ml

Artinya bahwa untuk mendapatkan larutan HCl 0.1M, maka diperlukan 95.78 ml HCl pekat untuk diencerkan dengan menggunakan Aquades hingga 10 liter.

Pembuatan Larutan Basa

$$\rho_{NaOH} = 1.5\frac{gr}{cm^3} = 1.5\frac{gr}{ml} = 1.5\,x\,1000\frac{gr}{l} \quad ; \text{Mr NaOH} = 40$$

Jika kadar NaOH pekat adalah 48%, maka nilai Molaritasnya (Konsentrasi) adalah sebagai berikut:

$$M_{NaOH} = \frac{\rho_{NaOH}\ x\ persen\ kadar NaOH}{Mr\ NaOH} = \frac{1.5gr\ ml^{-1}\ x 48\%}{40gr\ mol^{-1}}$$

$$M_{NaOH} = \frac{1.5gr\ l^{-1}\ x\ 48\ x\ 1000}{40\ gr\ mol^{-1}\ x\ 100} = \frac{1.5\ x\ 48\ x\ 10\ mol}{40\ l} = 18M$$

Jika ingin membuat 10 liter larutan Basa (NaOH) dengan konsentrasi 0,1M maka dapat dicari dengan cara sebagai berikut:

$$V_1.M_1 = V_2.M_2$$

dengan,

V_1 = Volume NaOH yang akan diencerkan (liter)
V_2 = Volume larutan NaOH yang diinginkan (liter)
M_1 = Konsentrasi NaOH yang akan diencerkan
M_2 = Konsentrasi NaOH yang diinginkan

V_1 x 18M = 10 liter x 0.1M

V_1 = 0.05555 liter = 55.55 ml

Artinya bahwa untuk mendapatkan larutan NaOH 0.1M, maka diperlukan 55.55 ml NaOH pekat untuk diencerkan dengan menggunakan Aquades hingga 10 liter.