PEMODELAN SISTEM PLANT SUHU DENGAN METODE IDENTIFIKASI RECURSIVE LEAST SQUARE

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan

Memperoleh gelar Sarjana Teknik

Disusun oleh :

TEGUH BUDI WIBOWO

NIM. 0510630100

DEPARTEMEN PENDIDIKAN NASIONAL

UNIVERSITAS BRAWIJAYA

FAKULTAS TEKNIK

MALANG

2009

PEMODELAN SISTEM PLANT SUHU DENGAN METODE IDENTIFIKASI RECURSIVE LEAST SQUARE

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
Memperoleh gelar Sarjana Teknik

Disusun oleh :

TEGUH BUDI WIBOWO

NIM. 0510630100

Telah Diperiksa dan Disetujui oleh:

DOSEN PEMBIMBING

M. Aziz Muslim, ST., MT., Ph.D	**Goegoes Dwi Nusantoro, ST., MT**
19741203 200012 1 001	**19711013 200604 1 001**

PEMODELAN SISTEM PLANT SUHU DENGAN METODE IDENTIFIKASI RECURSIVE LEAST SQUARE

Disusun oleh:

TEGUH BUDI WIBOWO
NIM. 0510630100 – 63

Skripsi ini telah diuji dan dinyatakan lulus pada
Tanggal 23 Desember 2009

DOSEN PENGUJI

Ir. Purwanto, MT
NIP. 19540424 198601 1 001

Fitriana Suhartati, ST., MT.
NIP. 19741017 199802 2 001

Ir. Erni Yudaningtyas, MT
NIP. 19650913 199002 2 001

Mengetahui,
Ketua Jurusan Teknik Elektro

Rudy Yuwono, ST., MSc.
NIP. 19710615 199802 1003

Kata Pengantar

Puji syukur kepada Tuhan YME yang telah memberikan segala nikmat dan rahmat-Nya sehingga penyusunan skripsi ini dapat diselesaikan. Karena hanya dengan pertolongan-Nya semata penulis mampu melewati segala kendala yang ada selama penyusunan skripsi ini.

Skripsi berjudul "Pemodelan Sistem Plant Suhu dengan Metode Identifikasi Recursive Least Square" ini disusun sebagai salah satu syarat untuk mendapatkan gelar Sarjana Teknik di Jurusan Teknik Elektro, Fakultas Teknik, Universitas Brawijaya. Terselesaikannya skripsi ini tentunya tidak lepas juga dari bantuan berbagai pihak. Oleh sebab itu, dengan segala kerendahan hati penulis menyampaikan terima kasih kepada:

1. Bapak Rudy Yuwono, ST. MSc. selaku Ketua Jurusan Teknik Elektro dan Bapak M. Aziz Muslim, ST., MT., Ph.D selaku Sekretaris Jurusan Teknik Elektro.
2. Bapak Ir. Purwanto, MT selaku Ketua Kelompok Dosen Keahlian konsentrasi Sistem Kontrol Jurusan Teknik Elektro.
3. Bapak M. Aziz Muslim, ST., MT., Ph.D dan Goegoes Dwi Nusantoro, ST., MT selaku dosen pembimbing atas bantuan dan motivasi serta bimbingannya selama ini.
4. Ibu Fitriana Suhartati, ST., MT. selaku Kepala Laboratorium Sistem Kontrol Teknik Elektro.
5. Kedua orang tua dan saudara-saudara saya, yang selalu mendukung dan mendoakan keberhasilan saya.
6. Semua pihak yang telah membantu terselesaikannya skripsi ini.

Penulis menyadari bahwa skripsi ini masih banyak kesalahan. Oleh karena itu kritik dan saran yang membangun sangat diharapkan.

Malang, Nopember 2009

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

PEMODELAN SISTEM PLANT SUHU DENGAN METODE IDENTIFIKASI RECURSIVE LEAST SQUARE

ABSTRAK

Salah satu cara untuk memodelkan suatu sistem fisik dapat dilakukan dengan identifikasi. Cara ini dilakukan berdasarkan pengolahan data yang diperoleh melalui percobaan. Pada skripsi ini dilakukan sebuah proses identifikasi dengan menggunakan metode *Recursive Least Square* (RLS) dengan struktur model ARX orde 4. Proses identifikasi ini juga membutuhkan sinyal uji sebagai masukan sistem fisik dan yang dipilih adalah sinyal *Pseudo Random Binary Sequance* (PRBS). Proses pembangkitan sinyal uji dan pengambilan data *input-output* sistem fisik dilaksanakan menggunakan mikrokontroler atmega 8535 dan sebagai *interface* pada komputer digunakan sebuah *software* yang dibuat dengan Borland delphi 7. Data yang dikirim oleh mikrokontroler akan diterima oleh *software* identifikasi dan langsung diolah, sehingga proses identifikasi berjalan secara *real-time*. Pada *software* ini ditampilkan setiap perubahan parameter dan hasil identifikasi dalam bentuk fungsi alih diskrit. Dari pengujian yang telah dilakukan didapatkan hasil y(k)=0.72367y(k-1)+0.29387y(k-2)+0.058514y(k-3)-0.17061y(k-4)+0.036505u(k-1)+0.024752u(k-2)+0.023322u(k-3)+0.007752u(k-4). Setelah melalui proses validasi diperoleh hasil Akaike's FPE= 0.0004697, FIT untuk sinyal uji PRBS= 91.3989, FIT untuk sinyal uji kotak= 79.922, FIT untuk sinyal uji step= 91.532, dan *whiteness test* adalah R(0)= 0.00046034, RN(0)=1, RN(1)= -0.065968, RN(2)= -0.080171, RN(3)= 0.04237, RN(4)= 0.10554.

Kata kunci: *Identifikasi, RLS, ARX, PRBS, plant suhu, real-time*

BAB I
Pendahuluan

1.1 Latar Belakang

Membangun sebuah model dari sebuah sistem fisik dapat dilakukan dengan 2 cara, yaitu melalui pendekatan analisis dan eksperimen. Untuk metode dengan pendekatan analisis, sistem yang nyata diwakili oleh sebuah gabungan elemen-elemen pembentuk yang dianggap ideal. Cara untuk memperoleh model sistem tersebut adalah dengan menggunakan persamaan-persamaan dari hukum fisika (seperti: hukum Kirchhoff dan hukum Newton) dan komponen-komponen yang terpasang di dalam sistem fisik (seperti: resistor dan kapasitor). (William L. Brogan, 2000). Kelemahan dari metode ini terletak pada pengidealan komponen pembentuk yang tentunya akan mempengaruhi ketepatan model yang akan diperoleh, selain itu kompleksitas persamaan matematis dari hukum fisika yang harus diselesaikan, semakin rumit dan besar sistem maka tentu akan semakin banyak melibatkan persamaan matematis. Selain kelemahan-kelemahan tersebut biasanya ketidaktersediaan informasi mengenai komponen-komponen yang ada di dalam sistem tersebut juga akan mempersulit dalam melakukan pemodelan. Oleh karena kelemahan-kelemahan tersebut, digunakanlah metode yang kedua yaitu dengan metode dengan pendekatan eksperimen.

Metode dengan pendekatan eksperimen (metode identifikasi) melakukan pembentukan model matematis dari sebuah sistem fisik berdasarkan data observasi dari sistem yang dimodelkan, maksudnya metode ini bekerja dengan mencatat setiap data masukan dan keluaran dari suatu sistem fisik. Kemudian pasangan-pasangan data tersebut dikalkulasi dengan suatu algoritma dari metode identifikasi sehingga akan diperoleh parameter filter yang dapat mewakili sistem fisik yang sebenarnya. Algoritma yang digunakan untuk proses ini ada bermacam-macam, salah satunya adalah algoritma *Recursive Least Square* (RLS). Kelebihan dari metode ini adalah sistem ini menganggap sistem fisik yang akan dimodelkan sebagai sebuah *black box*, sehingga apapun jenis komponen yang ada di dalam sistem fisik dan apapun jenis bahannya tidak dipermasalahkan dan tidak perlu

diperhatikan. Selain itu, algoritma RLS ini dapat diaplikasikan secara *real time* (Bobál, 2005)

Metode identifikasi ini juga memiliki kendala tersendiri dalam pengaplikasiannya yaitu kendala untuk dapat diintegrasikan dengan sebuah perangkat komputer dan berjalan dengan sistem *real time*, diperlukan perangkat pendukung seperti penghubung antara sistem komputer yang digital dengan sistem yang diidentifikasi yang biasanya analog dan juga perangkat untuk identifikasi ini tidak tersedia di laboratorium sistem kontrol Universitas Brawijaya. Oleh karena itu dalam skripsi ini akan dikembangkan sebuah sistem identifikasi secara *real time*.

1.2 Rumusan Masalah

Berdasarkan permasalahan yang terdapat pada latar belakang pendahuluan dibuat rumusan masalah sebagai berikut :

1. Bagaimana merancang alat untuk mengaplikasikan metode identifikasi secara *online*?
2. Bagaimana cara merancang sebuah *software* yang dapat menjembatani *hardware* dari luar dengan komputer sehingga dapat mencatat, mengolah dan menampilkan data masukan-keluaran plant dan hasil identifikasi?
3. Bagaimana cara memodelkan sistem yang belum diketahui persamaan matematisnya dengan menggunakan metode identifikasi dengan RLS?
4. Apakah dengan metode RLS diperoleh model yang dapat mewakili sistem yang diidentifikasi?

1.3 Batasan Masalah

Dalam perancangan ini permasalahan dibatasi oleh hal-hal sebagai berikut :

1. Metode pemodelan yang digunakan adalah identifikasi dengan *Recursive Least Square*.
2. Struktur model yang digunakan adalah ARX.
3. Orde model tertinggi adalah orde 4.
4. Plant yang diidentifikasi adalah prototipe pengaturan suhu ruangan (73412) yang terdapat pada laboratorium sistem kontrol Universitas Brawijaya Malang.

5. Sistem Plant yang akan diidentifikasi dikondisikan kecepatan kipasnya pada 1 skala dan pintu pengatur aliran keluar udara pada 1 skala.
6. Tegangan masukan untuk plant adalah 15 Volt.
7. Pada skripsi ini digunakan *software* code vision avr 1.24.8d, borland delphi 7 dan matlab 7.04.365.
8. Mikrokontroler yang digunakan adalah jenis Atmega8535.
9. Pengondisi sinyal yang digunakan adalah optocoupler jenis 4n33 dan transistor tipe TIP41C.
10. Tidak membahas detail mengenai rangkaian elektrik.

1.4 Tujuan

Tujuan yang ingin dicapai dari skripsi ini adalah tercipta suatu sistem alat pemodelan yang dapat mengidentifikasi sistem secara *real time* dan mudah dalam penggunaannya sehingga dapat digunakan lebih lanjut dalam perancangan sistem kontrol.

1.5 Manfaat

Perancangan sistem *hardware* dan *software* untuk pemodelan dengan metode identifikasi *recursive least square* ini diharapkan dapat membantu dalam mengidentifikasi plant yang belum diketahui persamaan matematisnya dengan lebih mudah dan cepat sehingga dapat membantu dalam penyelesaian masalah kontrol.

1.6 Sistematika Pembahasan

BAB I Pendahuluan

Bab ini berisi tentang uraian latar belakang, tujuan, batasan masalah, rumusan masalah, manfaat serta sistematika penulisan.

BAB II Tinjauan Pustaka

Tinjauan Pustaka berisi dasar teori penunjang penelitian. Pustaka yang diambil adalah pustaka yang relevan dan sesuai serta mendukung penelitian, seperti buku-buku ilmu pengaturan, identifikasi, dan lain-lain. Selain dari buku pustaka juga akan diambil dari jurnal, internet, dan sumber pengetahuan yang lain.

BAB III Metodologi Penelitian

Bab ini berisi tentang metode yang digunakan dalam pengerjaan alat seperti perancangan dan pembuatan rangkaian *interface*, pengujian alat, pengambilan data dan analisis data yang digunakan dalam skripsi ini.

BAB IV Perancangan dan Pembuatan Modul Identifikasi Sistem

Bab ini berisi tentang perancangan keseluruhan sistem identifikasi metode *recursive least square*. Setelah itu, bagaimana menerapkannya ke dalam mikrokontroler dan *software* di komputer, sehingga sistem dapat bekerja dengan baik.

BAB V Pengujian dan Analisis Sistem

Bab ini berisi tentang hasil pengujian sistem yang sudah dibuat, serta analisis hasil yang diperoleh.

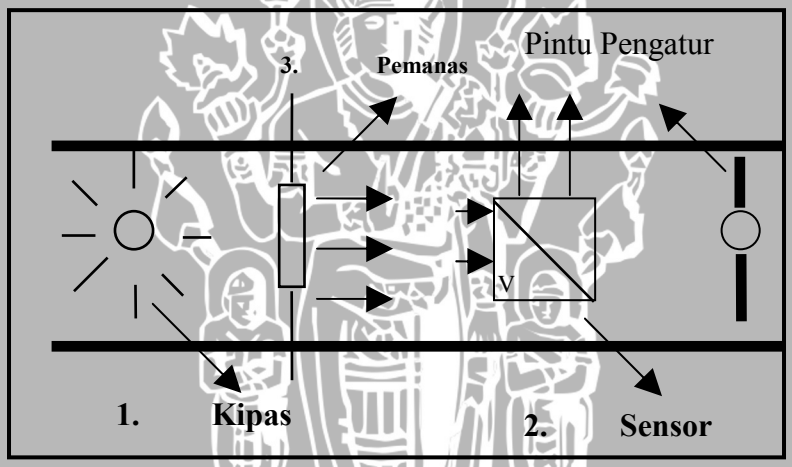
BAB VI Kesimpulan dan Saran

Dalam bab ini, maka semua hal yang sudah dikerjakan pada bab sebelumnya, dianalisis, dan diambil kesimpulan. Serta rekomendasi dan saran untuk pengembangan alat.

BAB II
Dasar Teori

2.1. Sistem Plant Suhu (73412)

Komponen yang mempengaruhi perubahan suhu untuk modul *plant* suhu terdiri dari kipas angin dan pemanas (lampu 24 Volt) yang diletakkan pada sebuah lorong sempit. Di sebelah pemanas terdapat sensor temperatur (PTC) yang bertujuan untuk mengukur udara panas yang mengalir pada lorong sempit. Kipas angin (ventilator angin) yang diletakkan pada lorong muka bertujuan untuk menyedot udara dari luar. Kipas angin ini dioperasikan dengan kecepatan konstan. Pada lorong ujung akhir dipasang sebuah penyekat yang dapat diatur posisi kemiringannya. Dengan mengatur sudut kemiringan penyekat tersebut aliran udara panas yang keluar dapat diperbesar maupun diperkecil.

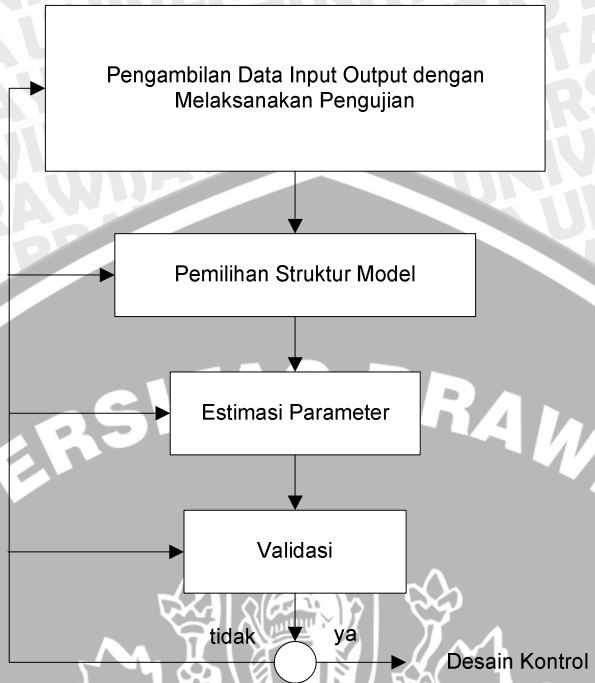


Gambar 2.1. Komponen Plant Suhu
Sumber: modul praktikum sistem kontrol

Tabung ini dilengkapi dengan tombol potensiometer untuk mempercepat putaran ventilator-motor dan tombol jendela untuk mengatur bukaan campuran udara sekitar dengan udara di dalam tabung. Gambar 2.1 menunjukkan skema diagram komponen yang diatur modul pengaturan temperatur udara. (Modul praktikum sistem kontrol)

2.2 Dasar Identifikasi Sistem

Secara umum proses identifikasi ditunjukan oleh Gambar 2.2.



Gambar 2.2. Struktur dari metode identifikasi secara recursive
Sumber: Landau-Zito, 2006

Untuk melaksanakan proses identifikasi sistem seperti yang telah ditunjukkan gambar di atas diperlukan langkah-langkah sebagai berikut:

1. Pengambilan data input-output
2. Menentukan struktur model
3. Estimasi parameter
4. Validasi model

2.2.1. Pengambilan Data Input-Output

Langkah awal dalam melaksakan identifikasi sistem adalah pengambilan data *input-output*. Pengujian ini tentu memerlukan sinyal uji tertentu yang akan diberikan kepada sistem fisik yang akan diidentifikasi. Agar diperoleh model yang tepat maka dalam pemilihan sinyal uji ini tidak boleh sembarangan. Syarat pemilihannya adalah suatu sinyal uji harus memiliki cakupan frekuensi yang lebar dan standard yang digunakan adalah sinyal *Pseudo Random Binary Sequences* (PRBS). (Landau, 2006)

Pseudo Random Binary Sequence (PRBS) adalah sinyal kotak yang termodulasi pada lebarnya dan berlangsung secara sekuensial. Sinyal ini biasanya dibangkitkan menggunakan *Linear Feedback Shift Register* (LFSR). Pada LFSR memiliki 2 parameter dasar yang menentukan sifat sekuensial yang dihasilkan, yaitu: panjang dari *shift register* dan susunan umpanbalik. PRBS memiliki variasi panjang sekuensialnya, tergantung dari panjangnya *shift register* seperti ditunjukkan Tabel 2.1

Tabel 2.1 tabel variasi panjang sekuensial PRBS

Panjang Register (N)	Panjang Sekuensial $L=2^N-1$	Posisi Tap Umpan Balik
2	3	1 dan 2
3	7	1 dan 3
4	15	3 dan 4
5	31	3 dan 5
6	63	5 dan 6
7	127	4 dan 7
8	255	2, 3, 4, dan 8
9	511	5 dan 9
10	1023	7 dan 10

Sumber: Landau, 2006

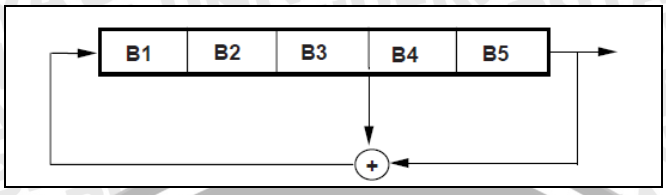
Panjang dari *shift register* menentukan periode maksimum yang dapat dihasilkan dari sekuensial PRBS yan tidak berulang dan dapat dinyatakan dengan persamaan:

$$L_{PRBS}=2^n-1 \quad \text{(2.1)}$$

Dimana n adalah panjang dari register LFSR (jumlah bit). Panjang maksimum dari PRBS disebut M-*sequence*.

Panjang maksimum yang dapat dihasilkan PRBS untuk panjang tertentu dari *shift register* dapat dicapai dengan mempersiapkan konfigurasi dari *feedback*. Pada dasarnya ada 2 kemungkinan untuk realisasi LFSR, yaitu: Fibonacci (*many to one*) dan Galois (*one to many*). Keduanya dapat didasarkan pada gerbang XOR atau XNOR menggunakan bermacam-macam angka dan kombinasi dari *feedback*-

taps-keluaran dari *shift register* khusus. Dengan mengubah konfigurasi umpanbalik (jumlah tap dan posisinya) memungkinkan untuk menemukan M-*sequence* yang berbeda untuk panjang tertentu dari *shift register*.



Gambar 2.3. Register Geser 5 Bit dengan Umpan Balik
Sumber: Landau, 2006

Prinsip pengujian proses dengan sinyal PRBS adalah membuat perubahan input kecil secara acak untuk membangkitkan gangguan (*perturbation*) yang kontinyu pada variabel *output*. Salah satu keuntungan penggunaan dari pendekatan ini adalah amplitudo perubahan input yang dibutuhkan dapat lebih kecil jika dibandingkan dengan perubahan *step* pada *step testing*. Selain itu, proses pengujian dapat dilakukan tanpa harus menunggu proses dalam keadaan tunak (*steadystate*). Jika pengujian dengan sinyal PRBS dilakukan, sinyal input secara teoritis disebut *white* (tidak berkorelasi) dan akan menghasilkan parameter model estimasi yang lebih baik. Frekuensi dari PRBS dapat dipilih untuk putaran (*flips*) cepat (*fast*) atau lambat (*slow*). Pemilihan frekuensi ini dapat menentukan jenis informasi terbaik yang akan didapat, misalnya untuk *fast* akan memberikan informasi yang akurat mengenai *deadtime*, *slow* akan memberikan informasi *steadystate gain* yang tepat sedangkan *medium* memberikan informasi *time constant* lebih baik. Dalam merancang PRBS perlu diperhatikan beberapa hal:

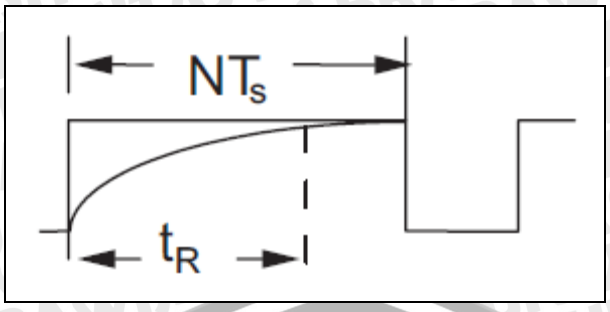
1. Menentukan Ukuran PRBS

Untuk mengidentifikasi secara tepat penguatan *steady state* dari model dinamis plant, durasi paling tidak, 1 dari pulsa harus lebih besar dari *rise time* t_R dari plant (termasuk *time delay*). Durasi maksimum (T_{im}) dari pulsa adalah $N.T_s$, kondisi tersebut menyebabkan:

$$T_{im}=NT_s > t_R \quad \text{..... (2.2)}$$

Dengan: T_{im} = Durasi maksimum
N = Panjang register
T_s = Periode sampling

Ditunjukkan oleh Gambar 2.4:



Gambar 2.4. Lebar Pulsa Sinyal PRBS
Sumber: Landau, 2006

Agar seluruh jangkauan spektrum frekuensi yang dihasilkan oleh sinyal PRBS dapat terpakai, maka lamanya percobaan minimal kurang atau sama dengan panjang sekuensialnya. Pada kebanyakan kasus, durasi dari percobaan (L) yang dipilih adalah sama dengan panjangnya sekuensial, jika tidak maka harus dipastikan bahwa

$$2^{N-1}T_s<L \text{ ; L=durasi percobaan} \quad \text{(2.3)}$$

Jika proses mengalami *noise* yang tinggi dan pergerakan *input* terlalu kecil sehingga berada dalam daerah *bandwith noise*, maka periode pengujian harus lebih lama untuk mendapatkan kualitas model yang lebih baik

2. Memilih Magnitude dari PRBS

Magnitude yang dihasilkan oleh PRBS mungkin akan sangat kecil, namun dia harus lebih besar daripada amplitudo dari gangguan (*noise*). Jika perbandingan magnitudo sinyal dengan *noise* terlalu kecil maka penting untuk menambah waktu pengujian agar diperoleh estimasi parameter yang lebih baik. (Landau, 2006)

Dengan memilih amplituda yang cukup besar dapat dilihat efek dari pergerakan data, tetapi juga jangan terlalu besar sehingga dapat menyebabkan proses *upset* (di atas *setpoint*) dan operator harus mengkompensasi variabel yang lain.

2.2.2. Menentukkan Struktur Model

Secara umum struktur dalam identifikasi tampak pada persamaan 2.4, yaitu:

$$A(q)y(k) = \frac{B(q)}{F(q)}u(k-nk) + \frac{C(q)}{D(q)}e(k) \quad \text{(2.4)}$$

Dengan: $A(q)=1+a_1q^{-1}+a_2q^{-2}...+a_{na}q^{-na}$

$B(q)=b_1q^{-1}+b_2q^{-2}\ldots+b_{nb}q^{-nb}$

$C(q)=c_1q^{-1}+c_2q^{-2}\ldots+c_{nc}q^{-nc}$

$D(q)=1+d_1q^{-1}+d_2q^{-2}\ldots+d_{nd}q^{-nd}$

$F(q)=1+f_1q^{-1}+f_2q^{-2}\ldots+f_{nd}q^{-nf}$

$y(k)$= keluaran

$u(k)$= masukan

$e(k)$= gangguan

Struktur model yang didapatkan tergantung pada adanya polynomial A, B, C, D, dan F, seperti yang ditunjukkan pada Tabel 2.2

Tabel 2.2. Macam-macam Struktur Model

Struktur	Model
B	AR
C	MA
A C	ARMA
A B	ARX
A B C	ARMAX
A B D	ARARX
B F	Ouput-Error (OE)
B F C D	Box-Jenkins (BJ)

Sumber: Brown dan Hwang, 1992

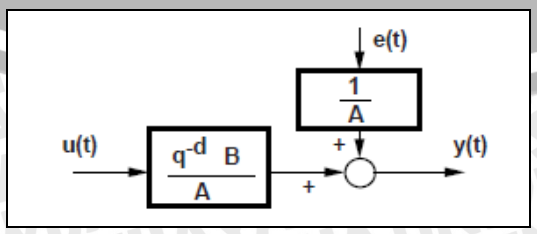
Dalam skripsi ini struktur model yang digunakan adalah *Auto Regresive with Exogenous input* (ARX), yang dapat dituliskan pada persamaan 2.5:

$$A(q)y(k) = B(q)u(k-nk) + e(k) \quad \ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots\ldots (2.5)$$

Dengan: $A(q)=1+a_1q^{-1}+a_2q^{-2}\ldots+a_{na}q^{-na}$

$B(q)=b_1q^{-1}+b_2q^{-2}\ldots+b_{nb}q^{-nb}$

Atau dalam bentuk diagram seperti pada Gambar 2.5

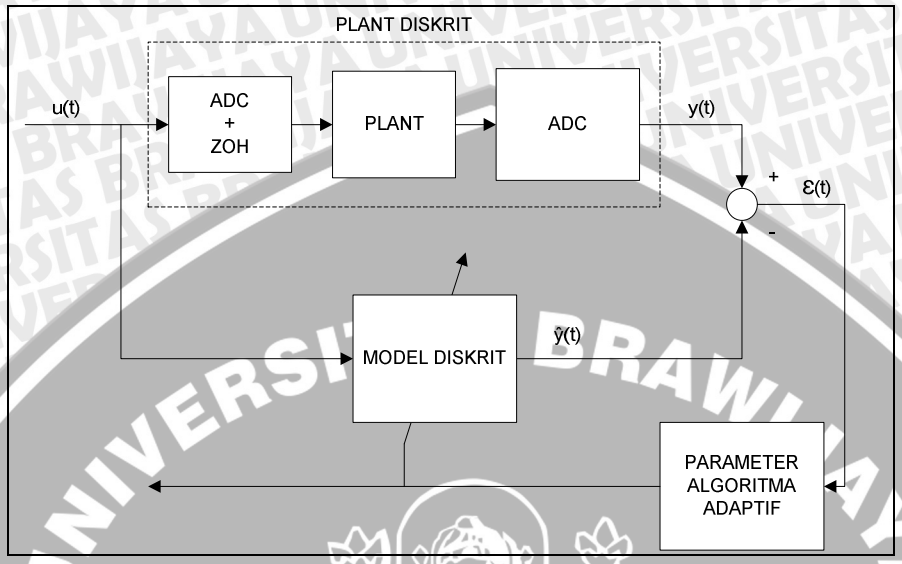


Gambar 2.5. Struktur model ARX

Sumber: Landau, 2006

2.2.3. Estimasi Parameter (Landau, 2006)

Proses estimasi parameter adaptif ditunjukkan oleh Gambar 2.6



Gambar 2.6. Blok Diagram Estimasi Parameter
Sumber: Landau, 2006

Untuk mengestimasi parameter model, sebuah kriteria estimasi harus dinyatakan terlebih dahulu. Kriteria yang akan digunakan di sini adalah *least-Squared error*, yang dinyatakan dalam persamaan:

$$V_t(\theta, Z^t) = \sum_{i=1}^{t} \varepsilon(i)^2 \quad \text{.....(2.6)}$$

dimana:

$$\varepsilon(t) = y(i) - \hat{y}(t) \quad \text{.....(2.7)}$$

$$\hat{y} = \theta^T \varphi(i) \quad \text{.....(2.8)}$$

$$\varepsilon(t) = y(i) - \theta^T \varphi(i) \quad \text{.....(2.9)}$$

$$\varphi(t) = [-y(i-1) - y(i-2) \dots - y(i-n_a) \quad u(i-1) \dots u(i-n_b)]^T \quad \text{...(2.10)}$$

dengan: Θ = parameter estimasi

$\mathcal{E}$ = error estimasi

$y(t)$ = output sebenarnya

$\hat{y}(t)$ = output estimasi

φ = vektor regresi

Untuk meminimalkan $e(t)$ maka berlaku

$$V_t(\theta, Z^t) = \sum_{i=1}^{t} [y(i) - \varphi^T(i)\theta]^2 \quad \text{.....(2.11)}$$

Sehingga

$$\hat{\theta}_t^{LS} = \arg\min V_t(\theta, Z^t) = [\sum_{i=1}^{t} \varphi(i-1)\varphi^T(i-1)]^{-1} \sum_{i=1}^{t} \varphi(i-1)y(i-1) \quad (2.12)$$

Dimana

$$F(t)^{-1} = \sum_{i=1}^{t} \varphi(i-1)\varphi^T(i-1) \quad (2.13)$$

Persamaan 2.12 masih merupakan persamaan *least square* yang belum *recursive*, untuk membuat persamaan tersebut menjadi *recursive* maka diberikan:

$$\hat{\theta}(t+1) = F(t+1) \sum_{i=1}^{t+1} \varphi(i-1)y(i) \quad (2.14)$$

$$F(t+1)^{-1} = \sum_{i=1}^{t+1} \varphi(i-1)\varphi^T(i-1) = F(t)^{-1} + \varphi(t)\varphi^T(t) \quad (2.15)$$

$$\hat{\theta}(t+1) = \hat{\theta}(t) + \Delta\hat{\theta}(t+1) \quad (2.16)$$

Dari persamaan 2.15 (dengan menambah $\varphi(t)\varphi^T(t)\hat{\theta}(t)$) sehingga diperoleh

$$\sum_{i=1}^{t+1} y(i)\varphi(i-1) = \sum_{i=1}^{t} y(i)\varphi(i-1) + y(t+1)\varphi(t) \pm \varphi(t)\varphi^T(t)\hat{\theta}(t) \quad (2.17)$$

Berdasarkan persamaan 2.12, 2.14, dan 2.15, persamaan 2.17 dapat ditulis kembali dalam persamaan 2.18

$$\sum_{i=1}^{t+1} \varphi(i-1)y(i) = F(t+1)^{-1}\hat{\theta}(t+1)$$

$$\sum_{i=1}^{t} \varphi(i-1)y(i) = F(t)^{-1}\hat{\theta}(t) + \varphi(t)\varphi^T(t)\hat{\theta}(t) + \varphi(t)[y(t+1) - \hat{\theta}(t)^T\varphi(t)] \quad (2.18)$$

$$F(t+1)^{-1}\hat{\theta}(t+1) = F(t+1)^{-1}\hat{\theta}(t) + \varphi(t)\varepsilon(t+1) \quad (2.19)$$

$$\hat{\theta}(t+1) = \hat{\theta}(t) + F(t+1)\varphi(t)\varepsilon(t+1) \quad (2.20)$$

Untuk membuat formula *recursive* bagi *F(t)* digunakan persamaan recursive $F(t)^{-1}$ pada persamaan 2.15. Pada persamaan tersebut berlaku matrix inversion lemma.

Lemma: misalkan F adalah matrix dengan dimensi (nxn) dan φ adalah vektor dari dimensi n, maka[1]

$$(F^{-1} + \varphi\varphi^T)^{-1} = F - \frac{F\varphi\varphi^T F}{1+\varphi^T F\varphi} \quad (2.21)$$

Dari persamaan 2.20 dan 2.22 maka

$$F(t+1) = F(t) - \frac{F(t)\varphi(t)\varphi(t)^T F(t)}{1+\varphi(t)^T F(t)\varphi(t)} \quad (2.23)$$

F adalah *adaptation gain* (konstanta atau berubah terhadap waktu), φ adalah vektor regresi, dan ε adalah error prediksi (selisih antara keluaran yang sebenarnya dengan keluaran model).

[1]Salah satu dapat menyederhanakan perkalian kedua bentuk dengan $F^{-1} + \varphi\varphi^T$ untuk memverifikasi bentuk invers

2.2.4. Validasi Model

Validasi model digunakan untuk membedakan model yang benar terhadap model yang kurang benar. Validasi model dapat dilakukan dengan cara uji *whitness*, *Akaike's FPE*, dan uji keakurasian.

2.2.4.1. Whiteness Test (Landau, 2006)

Langkah-langkah dalam melaksanakan *whiteness test* adalah:

1. Membuat data *input output* dari model yang telah didapatkan dengan menggunakan sinyal masukan yang sama ketika proses identifikasi.
2. Menghitung error prediksi, yaitu dengan cara mengurangkan antara nilai dari hasil pengukuran dengan nilai dari model .
3. Melaksanakan *whiteness* (*Uncorellation*) *test* pada nilai error prediksi (dikenal juga dengan nama *residual test*).

Whiteness test ini dilaksanakan dengan menghitung RN(0) dan RN(i) dari error prediksi, dimana kedua nilai tersebut diperoleh dari:

$$R(0) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon^2(t) \quad \text{.....(2.24)}$$

$$RN(0) = \frac{R(0)}{R(0)} = 1 \quad \text{.....(2.25)}$$

$$R(i) = \frac{1}{N}\sum_{t=1}^{N} \varepsilon(t)\varepsilon(t-i);\ i = 1,2,3,\ldots,i_{max} \quad \text{.....(2.26)}$$

$$RN(i) = \frac{R(i)}{R(0)};\ i = 1,2,3,\ldots,i_{max} \quad \text{.....(2.27)}$$

Secara teori hasil yang diharapkan adalah error prediksi memiliki sifat *white* yaitu RN(0)=1 dan RN(i)=0, namun hasil tersebut tidaklah mungkin didapatkan pada percobaan yang sebenarnya oleh karena itulah muncul kriteria validasi

$$RN(0) = 1 \quad \text{.....(2.28)}$$

$$|RN(i)| \leq \frac{2.17}{\sqrt{N}};\ i \geq 1 \quad \text{.....(2.29)}$$

Atau secara lengkap ditunjukkan pada Tabel 2.3

Tabel 2.3. Kriteria Validasi *Whitness Test*

Tingkat Signifikan	kriteria	N=128	N=256	N=512
3%	$2.17\sqrt{N}$	0.192	0.136	0.0096
5%	$1.96\sqrt{N}$	0.173	0.122	0.087
7%	$1.808\sqrt{N}$	0.16	0.113	0.08

Sumber: Landau, 2006

Namun demi penyederhanaan, oleh Landau kriteria tersebut tidak dipergunakan secara praktek dan memberikan kriteria secara umum, yaitu:

$$|RN(t)| \leq 0.15 \quad \text{.....(2.30)}$$

2.2.4.2. Akaike's Final Prediction Error (Akaike's FPE) (Ljung, 1999)

Diketahui persamaan ekspektasi dari $\bar{V}_N(\hat{\theta}_N)$ adalah

$$\bar{J}_p(M) = E\bar{V}_N(\hat{\theta}_N) \quad \text{.....(2.31)}$$

Dengan

$$\bar{J}_p(M) = E\bar{V}_N(\hat{\theta}_N) \approx V_N(\theta, Z^N) + \frac{2\lambda_0}{N} tr[\bar{V}''(\theta_0)[\bar{V}''(\theta_0)]^{-1}]$$

$$\bar{J}_p(M) = V_N(\theta, Z^N) + \lambda_0 \frac{2d_M}{N} \quad \text{.....(2.32)}$$

$$\hat{\lambda}_N = \frac{V_N(\hat{\theta}_N, Z^N)}{1-(d_M/N)} \quad \text{.....(2.33)}$$

Jika d_M merupakan banyaknya parameter hasil estimasi berdasarkan N data yang digunakan dalam validasi suatu model linier (persamaan 2.6):

$$V_N(\theta, Z^N) = \frac{1}{N}\sum_{t=1}^{n} \varepsilon(t)^2$$

Sehingga didapatkan kriteria FPE:

$$\bar{J}_p(M) \approx \frac{1+(d_M/N)}{1-(d_M/N)} V_N(\theta, Z^N)$$

$$\bar{J}_p(M) \approx \frac{1+(d_M/N)}{1-(d_M/N)} \frac{1}{N} \sum_{t=1}^{n} \varepsilon(t)^2 \quad \text{.....(2.34)}$$

Kriteria ini dikemukakan oleh Akaike pada tahun 1969 sebagai error prediksi akhir. Pada kriteria tersebut menunjukkan bagaimana memodifikasi persamaan

loss function untuk mendapatkan estimasi yang *reasonable* dari informasi estimasi saja.

2.2.4.3. Uji Keakurasian (Ljung, 1999)

Keakurasian model diuji dengan cara membandingkan respon model dengan respon sistem yang sebenarnya terhadap sinyal masukan tertentu seperti step, kotak, dan PRBS. Angka keakurasian ini dinyatakan dalam persentase, semakin besar nilainya (maksimal 100%) berarti keluaran model sudah mendekati keluaran sistem yang sesungguhnya. Nilai *Fitness* ini dapat dihitung dengan persamaan:

$$FIT = [1 - NORM(Y - Y_{model})/NORM(Y - MEAN(Y))] * 100 \quad \text{.....(2.26)}$$

Dengan: FIT= Nilai keakurasian (0-100%)

Y= keluaran sistem yang sebenarnya

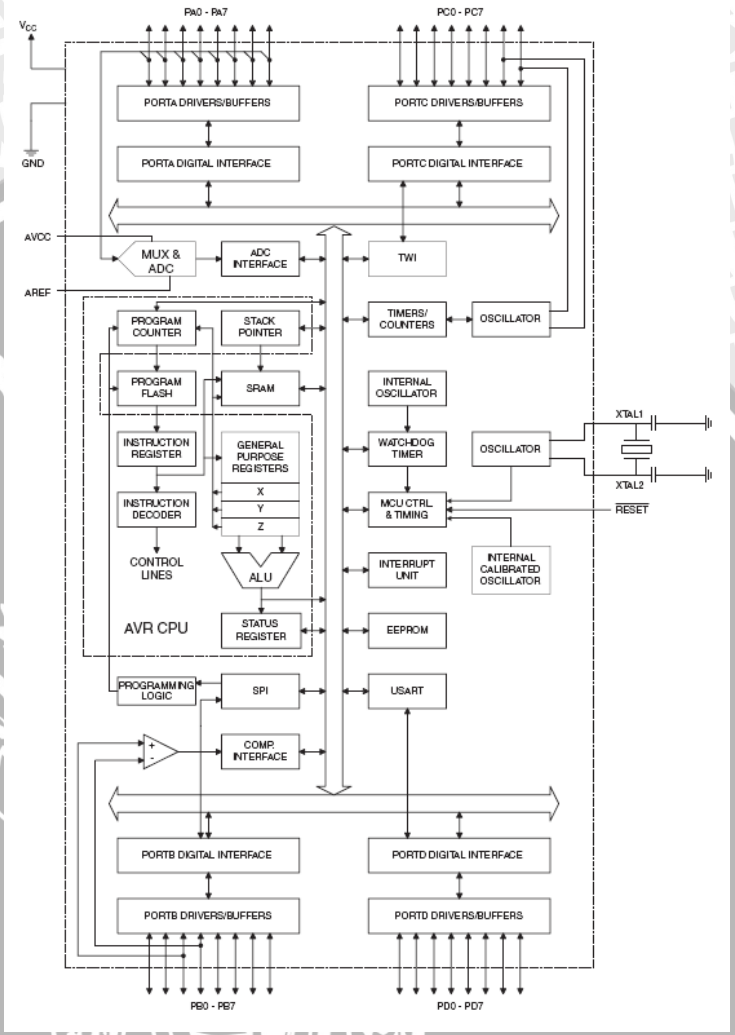
Y_{model}= keluaran model

2.2. Mikrokontroler ATmega 8535 (Atmel, 2007)

Mikrokontroler Atmega 8535 merupakan mikrokontroler CMOS 8 bit performa tinggi produksi Atmel dengan teknologi RISC yang terintegrasi dalam *Single Chip*. Mikrokontroler ini terdiri atas CPU, ADC, timer, paralel dan serial I/O, *flash* PEROM *(Programmable and Erasable Read Only Memory)*, RAM *(Random Acess Memory*), EEPROM *(Electrical Erasable Programmable Read Only Memory)*, dan on chip clock.

2.2.1. Arsitektur ATmega8535

Gambar 2.7 menunjukkan Blok Diagram mikrokontroler ATmega 8535.



Gambar 2.7. Blok Diagram Mikrokontroler ATmega 8535
Sumber: datasheet 8535, 2003

Dari gambar tersebut dapat dilihat bahwa ATmega8535 memiliki bagian sebagai berikut:

1. Saluran I/O sebanyak 32 buah, yaitu Port A, Port B, Port C, dan Port D.
2. ADC 10 bit sebanyak 8 saluran.
3. Tiga buah Timer/Counter dengan kemampuan perbandingan.
4. CPU yang terdiri atas 32 buah register.
5. Wacthdog Timer dengan osilator internal.
6. SRAM sebesar 512 byte.
7. Memori Flash sebesar 8 kb dengan kemampuan Read While Write.

8. Unit interupsi internal dan eksternal.
9. Port antarmuka SPI.
10. EEPROM sebesar 512 byte yang dapat diprogram saat operasi.
11. Antarmuka komparator analog.
12. Port USART untuk komunikasi serial.

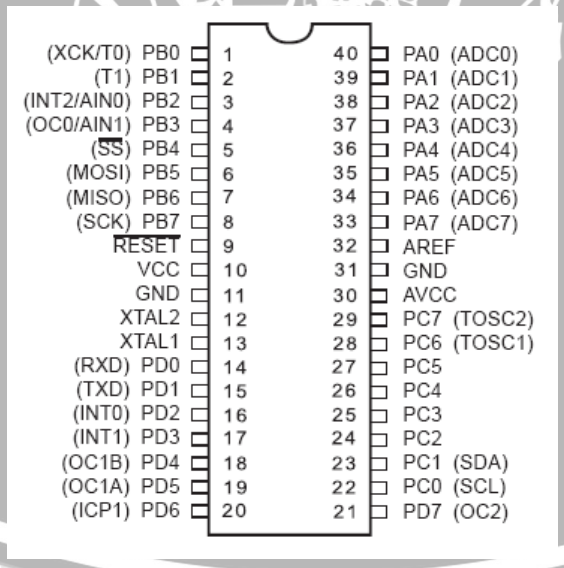
2.2.2. Fitur ATmega8535

Kapabilitas detail dari Atmega8535 adalah sebagai berikut:

1. Sistem mikroprosesor 8 bit berbasis RISC dengan kecepatan maksimal 16 Mhz.
2. Kapabilitas memori flash 8 KB, SRAM sebesar 512 byte, dan EEPROM sebesar 512 byte.
3. ADC internal dengan fidelitas 10 bit sebanyak 8 channel.
4. Portal komunikasi serial USART dengan kecepatan maksimal 2.5 Mbps.
5. Enam pilihan mode sleep menghemat penggunaan daya listrik.

2.2.3. Konfigurasi PIN ATmega8535

Konfigurasi PIN ATmega8535 ini bisa dilihat dalam Gambar 2.8.



Gambar 2.8. Konfigurasi PIN ATmega8535

Sumber: datasheet 8535, 2003

Penjelasan masing-masing pin:

1. VCC Power supply
2. GND Ground

3. AREF — Analog input referensi untuk ADC
4. AVCC — Power supply untuk ADC
5. RESET — Merupakan pin yang digunakan untuk me-reset mikrokontroler.
6. XTAL1 — Input untuk inverting oscillator amplifier dan input bagi clock internal.
7. XTAL2 — Output inverting oscillator amplifier.
8. PORT A — Port A merupakan Port I/O 8 bit dua arah dengan pull-up internal sekaligus sebagai input analog untuk ADC.
9. PORT B — Port B merupakan Port I/O dua arah . Fungsi tambahan dari Port B seperti yang terlihat dalam Tabel 2.4

Tabel 2.4. Fungsi tambahan dari port B

Port Pin	Fungsi tambahan
PB7	SCK (Bus serial clok SPI)
PB6	MISO (Bus Master Input/Slave Output SPI)
PB5	MOSI (Bus Master Output/Slave Input SPI)
PB4	SS (Pemilih input slave SPI)
PB3	OCO (output compare match pada timer/counter 0), AN1 (Input inferting analog komparator)
PB2	AN2 (Input noninferting analog komparator), INT0
PB1	T1 (Input counter pada timer/counter 1)
PB0	T0 (Input counter pada timer/counter 0), XCK (Input/output clok eksternal dari USART)

Sumber: datasheet 8535, 2003

10. PORT C — Port C merupakan Port I/O dua arah dan pin fungsi khusus yaitu TWI, komparator analog, dan Timer Oscillator.
11. PORT D — Port D merupakan Port I/O dua arah . Fungsi tambahan dari Port D seperti yang terlihat dalam Tabel 2.5

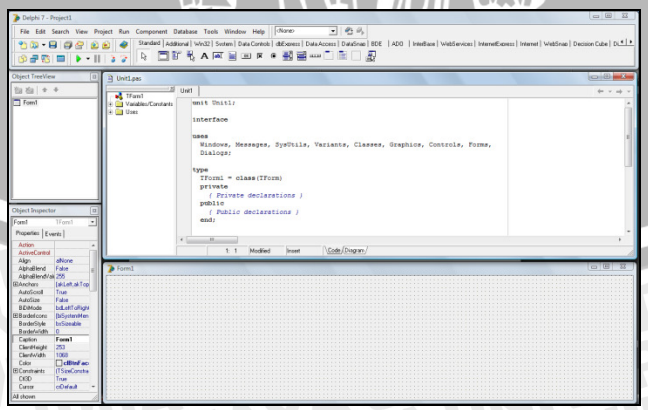
Tabel 2.5. Fungsi tambahan dari port D

Port pin	Fungsi tambahan
PD7	OC2 (Output Compare pada timer/counter 2)
PD6	ICP1
PD5	OC1A (Output compare A pada timer/counter 1)
PD4	OC1B (Output Compare B pada timer1)
PD3	INT1 (Interupt eksternal 1)
PD2	INT0 (Interupt eksternal 0)
PD1	TXD (output pin pada USART)
PD0	RDX (Input pin pada USART)

Sumber: datasheet 8535, 2003

2.3. Borland Delphi 7

Delphi merupakan salah satu piranti pengembangan aplikasi bebasis windows yang dikeluarkan oleh Borland International. Bahasa Pemrograman Delphi awalnya dari bahasa pemrograman Pascal setingkat Visual C, Visual Basic dan sejenisnya. Delphi dikemas sedemikian sehingga pemberian perintah untuk membuat obyek dapat dilakukan secara visual. Pemrogram tinggal memilih obyek apa yang ingin dimasukkan ke dalam form, lalu tingkah laku obyek tersebut saat menerima event/aksi tringgal dibuat programnya. Berikut adalah tampilan editor Delphi 7



Gambar 2.7. Tampilan Editor Delphi 7

Sumber: Perancangan

BAB III
Metodologi Penelitian

Untuk merealisasikan alat yang telah dirancang, langkah-langkah yang dilakukan adalah sebagai berikut:

3.1. Studi Literatur

Perencanaan sistem dilakukan sebagai langkah awal sebelum terbentuknya suatu sistem beserta rangkaian elektronik pendukungnya, hal ini dimaksudkan agar sistem pengindentifikasi dapat berjalan sesuai dengan yang telah direncanakan.

Perancangan sistem yang dilakukan meliputi:

1. Perancangan sistem identifikasi dengan algoritma *recursive least square*.
2. Penentuan spesifikasi sistem yang akan dibuat, meliputi:
    a. Penentuan deskripsi kerja sistem secara keseluruhan.
    b. Mikrokontroler dan rangkaian elektronik pendukung.
3. Penentuan rangkaian yang digunakan.
4. Perancangan program mikrokontroler.
5. Perancangan program Delphi 7 pada komputer.

3.2 Perancangan dan Pembuatan Modul Identifikasi Sistem

1. Perancangan Sistem Identifikasi
    a. Pengambian Data Input Output
    b. Menentukan Struktur Model
    c. Estimasi Parameter
    d. Validasi Model
2. Integrasi sistem keseluruhan
    a. Spesifikasi Alat
    b. Perancangan Perangkat Lunak

3.2. Perancangan dan Pembuatan Alat

Sebelum melakukan proses identifikasi diperlukan beberapa alat bantu. Berikut beberapa alat yang dibutuhkan:

a. Minimum sistem mikrokontroler Atmega8535 sebagai pembangkit sinyal PRBS
b. Pengkondisi sinyal sebagai jembatan dari mikrokontroler menuju plant dan juga sebaliknya

3.4. Perancangan dan Pembuatan Perangkat lunak

Software yang akan dibuat adalah pemrograman untuk pembangkit sinyal PRBS pada mikrokontroler, akusisi dan pengolahan data (proses identifikasi) pada delphi.

3.5 Pembuatan Keseluruhan Sistem

Setelah melakukan perancangan sistem identifikasi, maka selanjutnya akan dilakukan penggabungan semua sub sistem yang telah dirancang.

3.6. Pengujian Sistem dan Analisis

Pengujian dilakukan untuk menganalisis alat yang dibuat telah memberikan hasil sesuai dengan yang direncanakan atau tidak. Pengujian dilakukan secara per blok terlebih dahulu dan kemudian secara keseluruhan sistem.

3.7. Pengambilan Kesimpulan dan Saran

Tahap berikutnya adalah pengambilan kesimpulan dari peralatan yang dibuat. Pengambilan kesimpulan ini didasarkan pada kesesuaian antara perancangan dengan hasil pengujian. Tahap terakhir adalah saran yang dimaksudkan untuk memperbaiki kesalahan–kesalahan yang terjadi serta menyempurnakan penulisan.

BAB IV
Perancangan dan Pembuatan Modul Identifikasi Sistem

4.1. Perancangan Modul Identifikasi Sistem

Identifikasi sistem merupakan suatu cara menentukan model dari sistem dinamis dengan melaksanakan percobaan. Langkah-langkah percobaan untuk proses identifikasi meliputi:

5. Pengambilan data input-output
6. Menentukan struktur model
7. Estimasi parameter
8. Validasi model

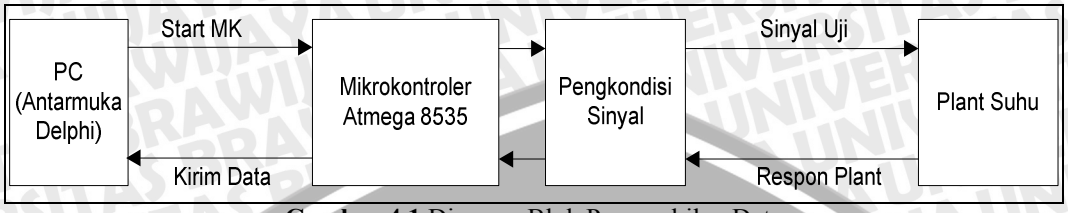
Pada skripsi ini akan dibangun sebuah sistem identifikasi yang akan dapat mengidentifikasi sebuah sistem plant secara *real-time* dengan kata lain langkah-langkah percobaan identifikasi dari no. 1 hingga 3 diharapkan dapat dijalankan secara *real-time*. Namun untuk membangun sistem tersebut harus dilakukan pembangunan sistem secara bertahap, yaitu:

1. Membuat sistem pengambilan data *input-output* secara otomatis menggunakan komputer dan merekamnya
2. Pembuatan perangkat lunak sebagai antarmuka plant dengan komputer
3. Pembuatan perangkat lunak algoritma estimasi

Pada skripsi ini digunakan aplikasi Delphi sebagai antarmukanya. Semua bagian tersebut nantinya akan bergabung menjadi kesatuan sistem identifikasi secara *real-time*. Selain semua bagian tersebut, diperlukan juga sebuah pembanding agar dapat diketahui apakah sistem identifikasi yang dibuat nantinya sudah benar atau belum, oleh karena itu digunakan juga aplikasi matlab untuk pembanding hasil identifikasi secara *real-time*. Nantinya hasil *real-time* yang didapat dari sistem Delphi akan dibandingkan dengan hasil perhitungan secara *off-line* yang didapat dengan menggunakan matlab.

4.1.1. Pengambilan Data Input-Output

Pengambilan data *input/output* sistem dilaksanakan dengan mengondisikan sistem dengan rangkaian *loop* terbuka, seperti ditunjukkan Gambar 4.1



Gambar 4.1 Diagram Blok Pengambilan Data
Sumber: Perancangan

Pada skripsi ini sinyal uji yang digunakan adalah jenis PRBS (*Pseudo Random Binary Sequance*) dengan jumlah bit 7. Sinyal uji ini akan dibangkitkan oleh mikrokontroler. Karena digunakan panjang register 7 bit maka sesuai dengan Tabel 2.1, panjang sekuensial yang akan dihasilkan adalah 127 bit untuk 1 kali proses generasi sinyal uji PRBS.

Untuk mengidentifikasi secara tepat penguatan *steady state* dari model dinamis plant, paling tidak ada 1 pulsa dengan durasi pulsa harus lebih besar dari *rise time* t_R dari plant (termasuk *time delay*), oleh karena itu diperlukan data uji plant dengan *input step* dan diperoleh grafik respon seperti ditunjukkan oleh Gambar 4.2

Gambar 4.2 Respon Plant terhadap Input Step
Sumber: Perancangan

Nilai akhir respon yang sebenarnya adalah 6.6275 volt dan pada grafik di atas telah mengalami proses normalisasi. Sesuai dengan pengertian dari *rise time* yaitu waktu yang dibutuhkan respon agar bertambah dari 10% menjadi 90% dari nilai akhir maka dari grafik dapat dilihat bahwa *rise time* sistem adalah 139 detik atau 2.32 menit. Maka, minimal 1 pulsa dari PRBS harus lebih lama dari 2.32 menit dan pada skripsi ini dipilih 3.5 menit. Periode sampling pengambilan data ADC mikrokontroler ditentukan berdasarkan Tabel 4.1.

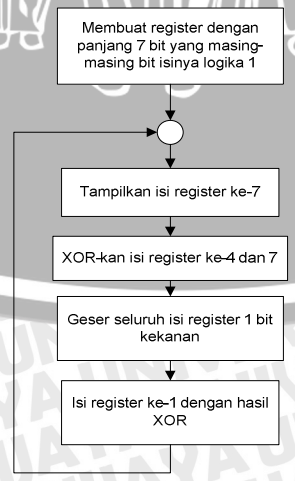
Tabel 4.1. Periode Sampling Berdasarkan Jenis Plant

Jenis Plant	Periode Sampling (s)
Tingkat Aliran	1-3
Level	5-10
Tekanan	1-5
Suhu	10-180
Distilasi	10-180
Mekanisme servo	0.001-0.05
Katalis reactor	10-45
Proses semen	20-45
Pengering	20-45

Sumber: Landau, 2006

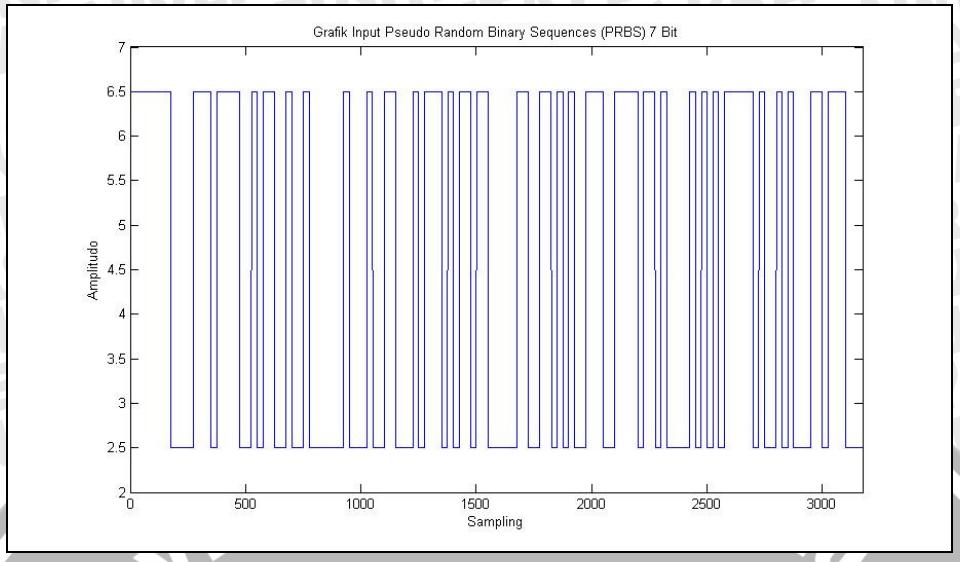
Berdasarkan tabel tersebut untuk prototipe pengaturan suhu ruangan (plant suhu) menggunakan periode sampling 10 detik.

Proses pembentukan sinyal PRBS ditunjukkan oleh Gambar 4.3



Gambar 4.3. Diagram Proses pembentukan sinyal PRBS
Sumber: Perancangan

Hasilnya ditunjukkan oleh Gambar 4.4:

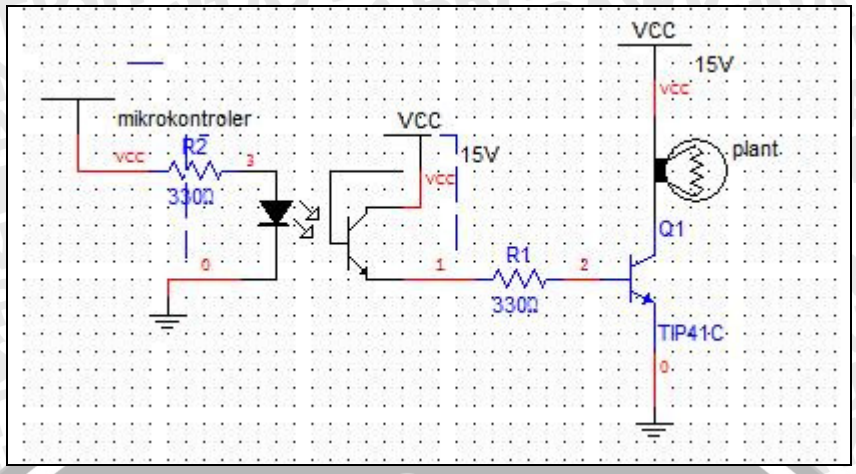


Gambar 4.4. Sinyal Pseudo Random Binary Sequences
Sumber: Perancangan

Pada skripsi ini pembangkit sinyal uji dan penerima data respon plant adalah mikrokontroler ATmega 8535. Karena mikrokontroler tersebut hanya memiliki tegangan keluaran 5 volt dan tegangan maksimum ADC sebesar 5 volt, maka diperlukan pengondisi sinyal pada bagian keluaran mikrokontroler yang menuju plant dan juga pada sebelum masukan ADC.

4.1.1.1. Pengondisi Sinyal Input

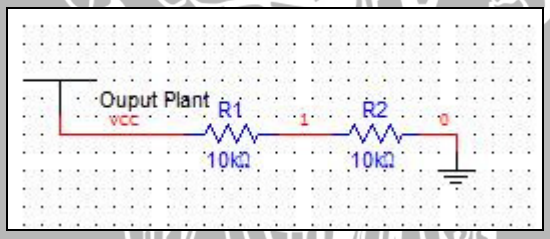
Rangkaian pengondisi sinyal pada bagian input ini diperlukan karena sinyal uji yang dihasilkan oleh mikrokontroler masih berupa sinyal digital dengan nilai tegangan maksimum 5V dan minimum 0V, sedangkan plant yang akan diuji memerlukan tegangan masukan sebesar 15V, sehingga digunakan optocopler dan transistor sebagai pengondisinya. Rangkaian pengondisi tersebut ditunjukkan oleh Gambar 4.5.



Gambar 4.5. Rangkaian Pengondisi pada Keluaran Mikrokontroler
Sumber: Perancangan

4.1.1.2. Pengondisi Sinyal Sebelum Masukan Mikrokontroler

Rangkaian pengondisi sinyal pada bagian sebelum input ADC mikrokontroler ini diperlukan karena respon plant yang diterima oleh mikrokontroler maksimal 15V dan minimal 0V, sedangkan ADC mikrokontroler hanya dapat menerima tegangan masukan sebesar 5V, sehingga digunakan rangkaian pembagi tegangan sebagai pengondisinya. Rangkaian pengondisi tersebut ditunjukkan oleh Gambar 4.6.



Gambar 4.6. Rangkaian Pengondisi Sinyal Sebelum Masukan Mikrokontroler
Sumber: Perancangan

4.1.2. Menentukan Struktur Model

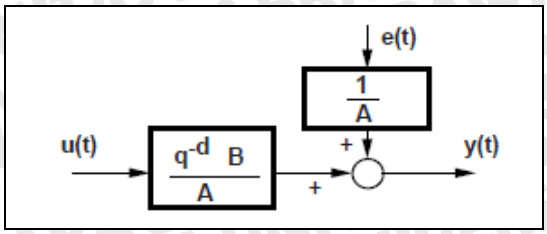
Dalam skripsi ini struktur model yang digunakan adalah ARX (*Auto Regresive with Exogenous input*), yang dapat dinyatakan sebagai(persamaan 2.5):

$$A(q)y(k) = B(q)u(k - nk) + e(k)$$

Dengan: $A(q)=1+a_1q^{-1}+a_2q^{-2}\ldots+a_{na}q^{-na}$

$B(q)=b_1q^{-1}+b_2q^{-2}\ldots+b_{nb}q^{-nb}$

Atau dalam bentuk diagram seperti pada Gambar 4.7



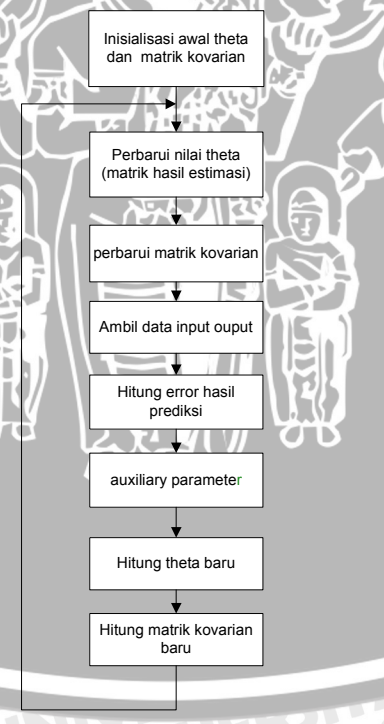
Gambar 4.7. Struktur model ARX

Sumber: Landau, 2006

Model ARX ini mengandung 2 macam parameter yang akan diestimasi, yaitu A dan B dengan Masing-masing parameter memiliki orde dari 1 hingga ke-n. Untuk menentukan orde yang cocok untuk plant ini maka digunakan cara dengan membandingkan pada orde berapa (1 hingga 4) yang memiliki *loss function* terkecil.

4.1.3 Estimasi Parameter

Untuk memudahkan dalam pembuatan program estimasi maka dibuat diagram sesuai dengan persamaan estimasi yang telah ditentukan sebelumnya



Gambar 4.8. Diagram Estimasi Recursive Least Square

Sumber: Perancangan

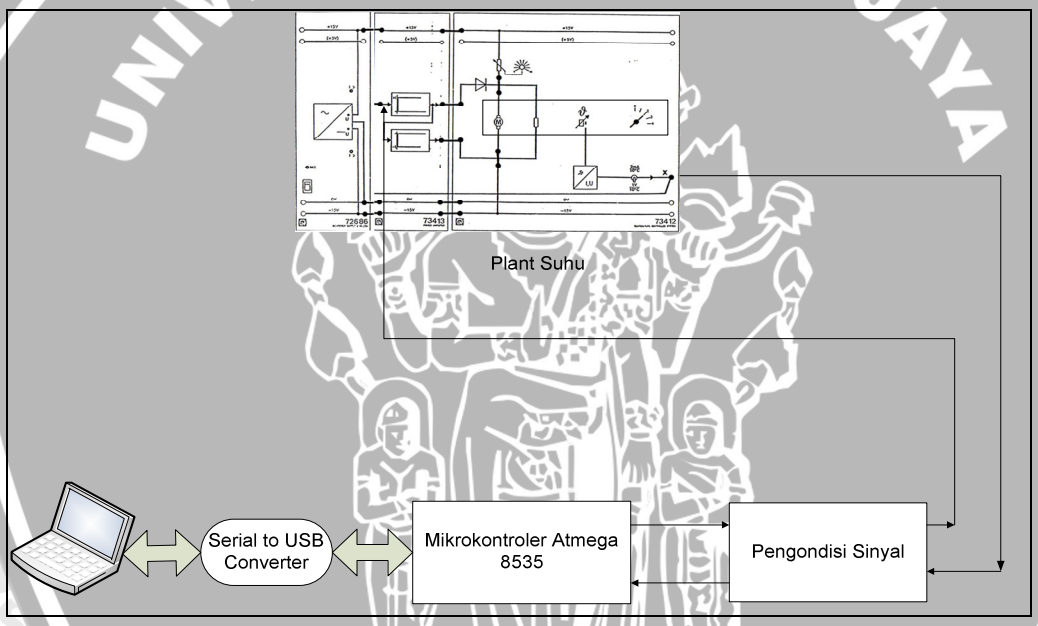
Seluruh proses tersebut akan diprogramkan kedalam Delphi.

4.1.4. Validasi Model

Proses validasi hasil identifikasi yang akan digunakan adalah *whiteness test*, uji dengan *akaike's FPE* (*Final Prediction Error*), dan uji keakurasian dengan menguji model yang diperoleh dengan memberi masukan fungsi *step*.

4.2. Integrasi Sistem Keseluruhan

Perencanaan ini dilakukan secara bertahap untuk memudahkan dalam menganalisa setiap bagian sistem maupun keseluruhan sistem. Setelah menggabungkan seluruh sub sistem yang telah dirancang sebelumnya maka akan diperoleh keseluruhan diagram blok perancangan sistem identifikasi secara *on-line* yang dapat dilihat pada Gambar 4.9.



Gambar 4.9. Blok Diagram Sistem
Sumber: Perancangan

Keterangan blok diagram:

1. Komputer sebagai penampil, penyimpan, dan pengolah data masukan dan keluaran sistem.
2. *Serial to USB converter* sebagai konverter komunikasi serial ke USB.
3. Mikrokontroler Atmega 8535 sebagai pengatur lalu lintas data dan penghasil sinyal uji.
4. Pengondisi sinyal yang akan digunakan adalah optocoupler yang berfungsi mengkonversi dari sinyal digital ke analog.

5. Plant suhu sebagai objek yang dimodelkan.
6. Model Plant berupa mikrokontroler yang berisi model dari plant.

Prinsip kerja dari proses identifikasi ini adalah sebagai berikut:

1. Mikrokontroler menunggu perintah dari komputer untuk memulai proses eksekusi program.
2. Bila perintah untuk mengeksekusi program telah diterima oleh mikrokontroler maka mikrokontroler akan menghasilkan sinyal uji sebagai masukan untuk plant.
3. Sebelum masuk ke plant, sinyal dari mikrokontroler yang terdiri dari dua keadaan logika 1 (5 V) dan logika 0 (0 V) dikonversi menjadi sinyal analog oleh pengondisi sinyal.
4. Pengondisi sinyal mengubah logika 1 menjadi tegangan 15 V dan logika 0 menjadi 0 V.
5. Keluaran dari pengondisi sinyal menjadi masukan bagi plant suhu.
6. Plant suhu akan memberikan respon.
7. Respon atau keluaran dari plant suhu akan diterima oleh mikrokontroler.
8. Sebelum masuk ke ADC mikrokontroler, sinyal analog dari plant melewati pengondisi sinyal lagi. Pengondisi sinyal ini berupa rangkaian pembagi tegangan.
9. Sinyal yang masuk ke pengondisi sinyal tegangannya akan dibagi 2 atau dilemahkan menjadi 0.5 dari nilai sebenarnya.
10. Keluaran dari pengondisi sinyal masuk ke ADC mikrokontroler.
11. Data sinyal uji dan respon plant pada 1 siklus proses dikirim oleh mikrokontroler ke komputer.
12. Oleh komputer (program antarmuka) data tersebut akan diterima. Selain disimpan juga langsung diolah dengan program estimator *recursive least square*.
13. Proses dari awal akan terus berulang sehingga menjadikan sistem ini bekerja secara *real-time.*

4.2.1. Spesifikasi Alat

Spesifikasi alat yang akan dibuat adalah sebagai berikut:

1. Sebuah komputer
2. Minimum sistem mikrokontroler atmega 8535
3. Catu daya yang dibutuhkan: 5 V untuk mikrokontroler dan 15 V untuk sistem plant.
4. Tegangan keluaran dari mikrokontroler sebesar 0V-5V
5. Sebagai pengondisi sinyal dari mikrokontroler menuju plant suhu digunakan optocoupler tipe 4n33 dan transistor tipe TIP41C
6. Tegangan keluaran transistor 0V-15V
7. Tegangan keluaran plant 0 V- 10V
8. Sebagai pengondisi sinyal dari plant suhu menuju mikrokontroler digunakan rangkain pembagi tegangan dengan 2 buah resistor dengan nilai resistensi yang sama.
9. Tegangan yang diterima mikrokontroler 0V-5V
10. Komunikasi dari mikrokontroler ke PC secara serial dengan standard RS232.

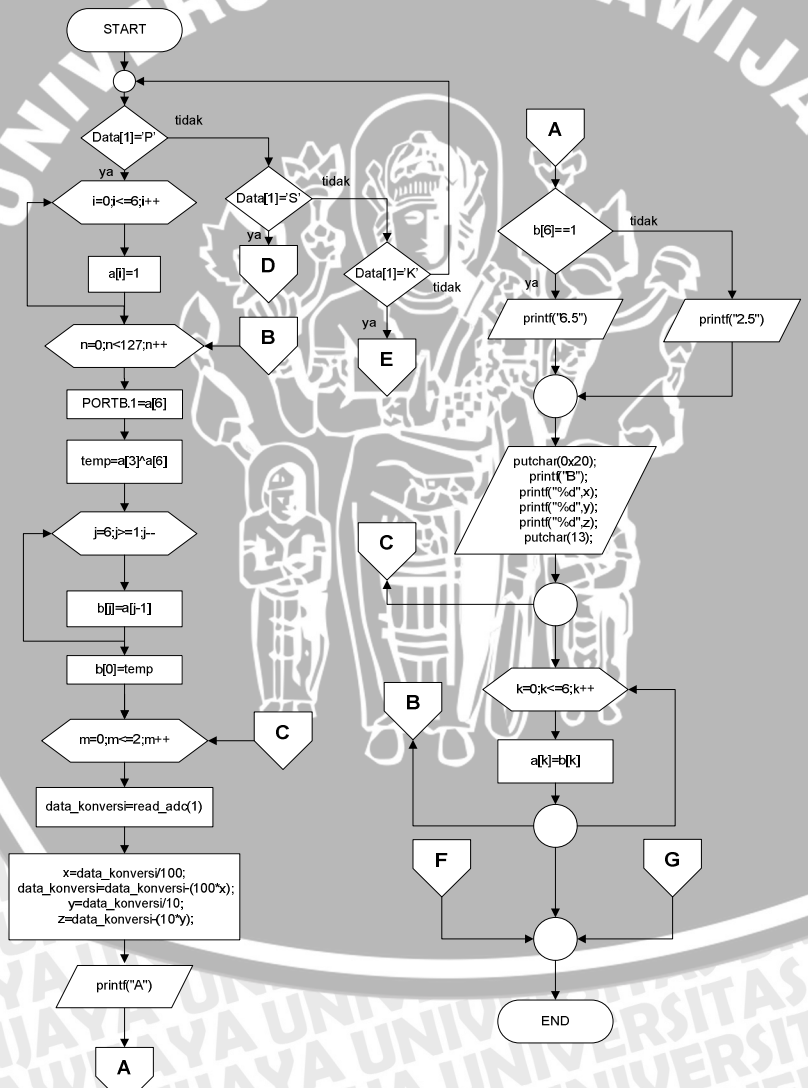
4.2.2. Perencanaan Perangkat Lunak

Pada bagian ini ada 2 bagian perangkat lunak, yaitu perangkat lunak pada mikrokontroler dan komputer. Perangkat lunak pada mikrokontroler berfungsi sebagai pembangkit sinyal uji PRBS, mengkonversi data respon plant dari data analog ke digital, dan mengirimkan data *input-output* ke komputer. Perangkat lunak pada komputer dibuat dengan program delphi berfungsi sebagai penerima data dari mikrokontroler, perekam data (disimpan dalam format *.txt) dan estimator parameter.

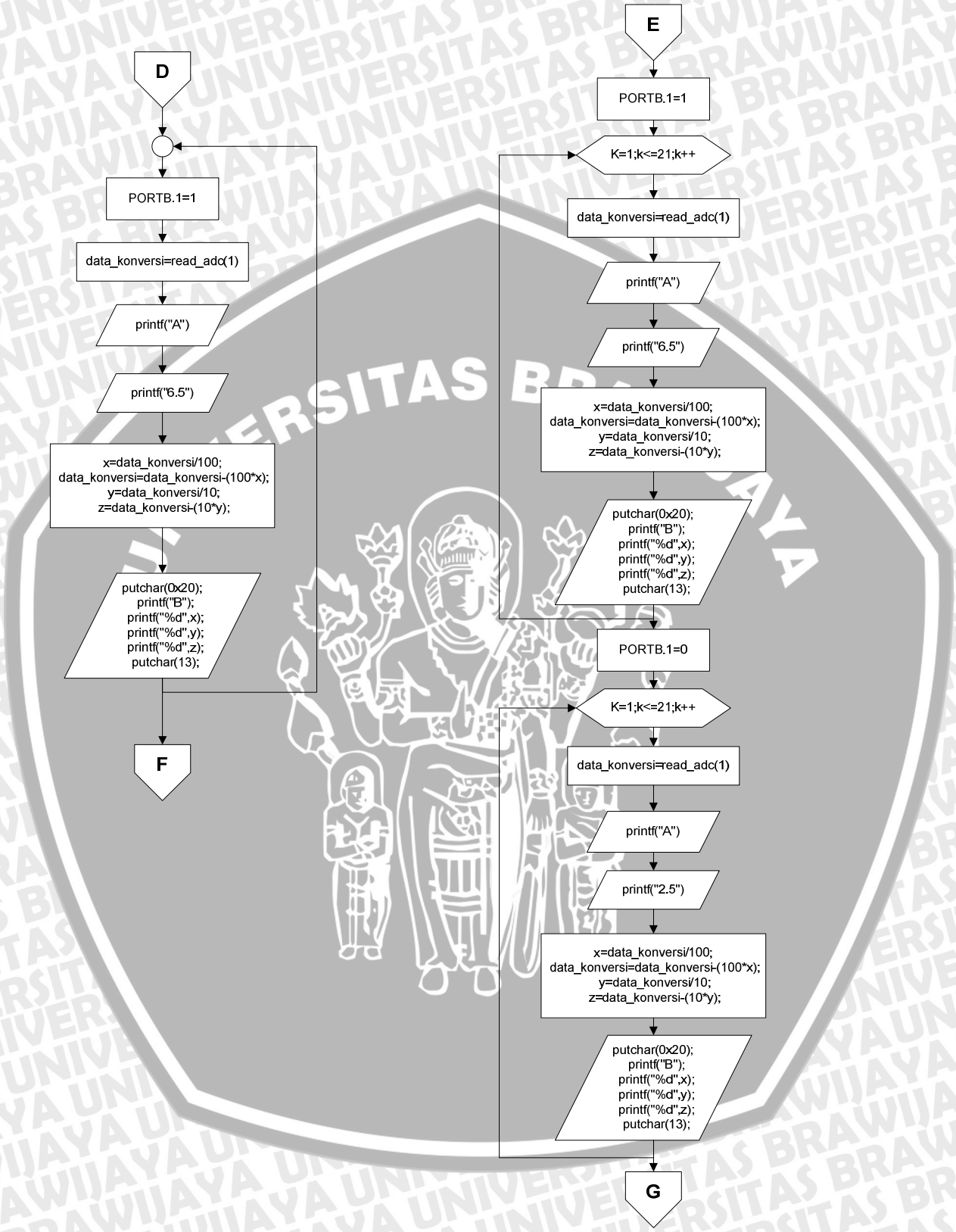
4.2.2.1. Perangkat Lunak pada Mikrokontroler

Perangkat lunak pada mikrokontroler bertugas membangkitkan sinyal uji PRBS, mengambil data respon plant, dan mengirimkan kekomputer. Agar dapat melaksanakan tugas-tugas tersebut maka perlu diatur waktu kerja dari mikrokontroler (MK). Pertama yang dilakukan MK adalah menyiapkan register sebanyak 7 bit untuk proses pembangkitan sinyal PRBS. Setelah 7 bit register siap

maka MK akan memulai membangkitkan sinyal PRBS, namun keseluruhan sekuensial sinyal PRBS tidak langsung terbentuk sekali proses selesai melainkan secara bertahap tiap bit. Setelah isi logika bit ke-7 dari register dikeluarkan disalahsatu pin MK selanjutnya MK akan melakukan proses PRBS selanjutnya (telah ditampilkan pada ambar 4.3.), namun di antara proses itu MK diberi tugas mengambil data respon plant dari ADC dan mengirimkan kekomputer (proses ambil data dan kirim data dilakukan beberapa kali sesuai *delay* waktu yang diinginkan). Setelah rutin ambil data dan kirim data selesai MK akan melanjutkan proses PRBS. Proses ini akan berulang terus. *Flow chart* nya ditunjukkan oleh Gambar 4.10.



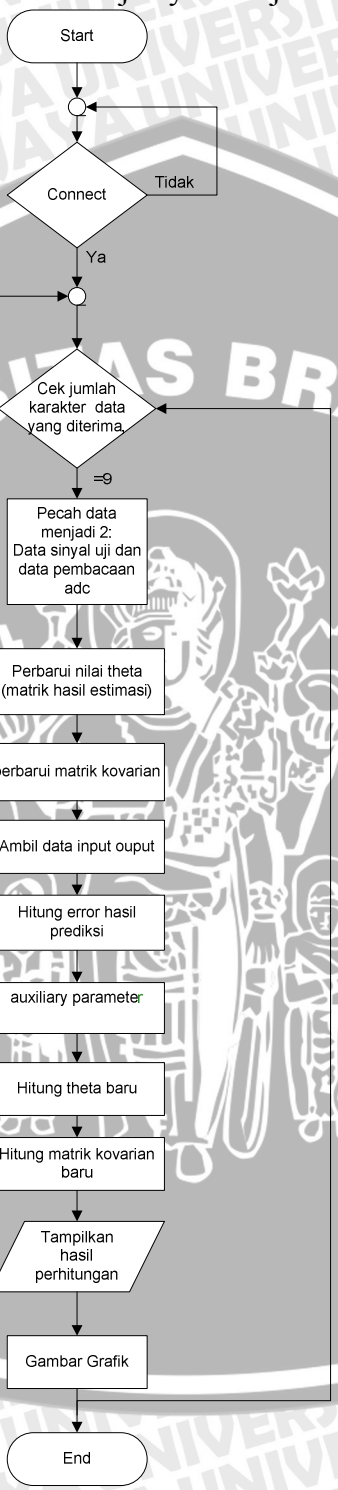
Gambar 4.10. Diagram alir perangkat lunak pembangkit sinyal uji (PRBS,step,kotak) dan pengirim data input output
Sumber: Perancangan



Gambar 4.10. (Lanjutan) Diagram alir perangkat lunak pembangkit sinyal uji (PRBS,step,kotak) dan pengirim data input output
Sumber: Perancangan

4.2.2.2. Perangkat Lunak pada Komputer

Perangkat lunak pada komputer bertugas menerima data dari MK dan sebagai estimator parameter. Proses kerja nya ditunjukkan oleh gambar 4.11.



Gambar 4.11. Diagram alir perangkat Lunak pada Komputer
Sumber: Perancangan

BAB V
Pengujian dan Analisa Data

Pengujian alat ini bertujuan untuk menentukan apakah alat yang telah dibuat berfungsi dengan baik dan sesuai dengan perencanaan. Pengujian ini meliputi pengujian setiap blok maupun pengujian secara keseluruhan. Pengujian setiap blok ini dilakukan untuk mempermudah analisis apabila alat tidak bekerja sesuai dengan perencanaan.

5.1. Pengujian Perangkat Keras

5.1.1. Pengujian I/O Minimum Sistem Mikrokontroler

a. Tujuan

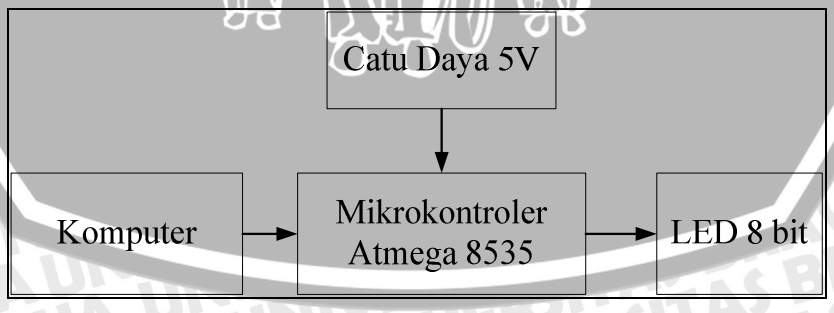
Mengetahui kondisi awal dari sistem mikrokontroler agar sesuai dengan yang diharapkan.

b. Peralatan

1. Komputer dengan kabel penghubungnya.
2. Minimum sistem mikrokontroler atmega 8535.
3. Lampu LED mewakili keluaran 8 bit.
4. Catu daya 5 Volt.

c. Langkah Pengujian

1. Merangkai peralatan seperti dalam Gambar 5.1. Keluaran terhubung dengan LED yang mewakili keluaran 8 bit.



Gambar 5. 6. Blok Diagram Pengujian Mikrokontroler

2. Mengisi mikrokontroler dengan program sederhana yaitu dengan mengeluarkan data biner dari 0FH sampai dengan F0H pada Port A, kemudian diisikan pada mikrokontroler ATmega 8535.
3. Mengaktifkan catu daya.
4. Mencatat data keluaran yang diwakili oleh lampu led 8 bit ke dalam bentuk biner.

d. Hasil Pengujian dan Analisis

Tabel 5. 4. Hasil Pengujian Sistem Mikrokontroler

Kondisi	Keluaran pada led display							
	Bit 0	Bit 1	Bit 2	Bit 3	Bit 4	Bit 5	Bit 6	Bit 7
I	1	1	1	1	0	0	0	0
II	0	0	0	0	1	1	1	1

Dari Tabel 5.1. terlihat bahwa Port A memberikan logika 0FH dan F0H secara bergantian sesuai dengan isi program. Dengan demikian rangkaian minimum sistem mikrokontroler Atmega 8535 dapat bekerja dengan baik.

5.1.2. Pengujian Input ADC Mikrokontroler

a. Tujuan

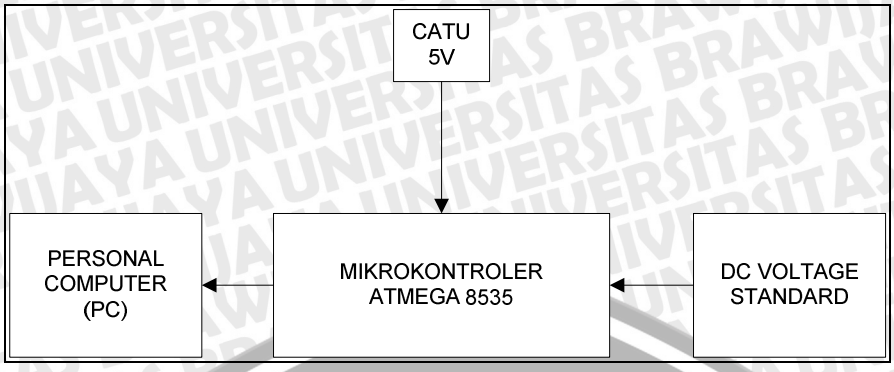
Mengetahui besarnya penyimpangan nilai dari pembacaan ADC mikrokontroler

b. Peralatan

1. Komputer dengan kabel penghubungnya
2. Mikrokontroler Atmega 8535
3. DC Voltage Standard
4. Catu daya 5V
5. Kabel serial

c. Langkah pengujian

1. Merangkai peralatan seperti dalam Gambar 5.2.
2. Mengaktifkan catu daya.
3. Menuliskan program untuk mengaktifkan mode ADC mikrokontroler dan agar dapat mengirim data ke PC.
4. Mencatat data pembacaan ADC.

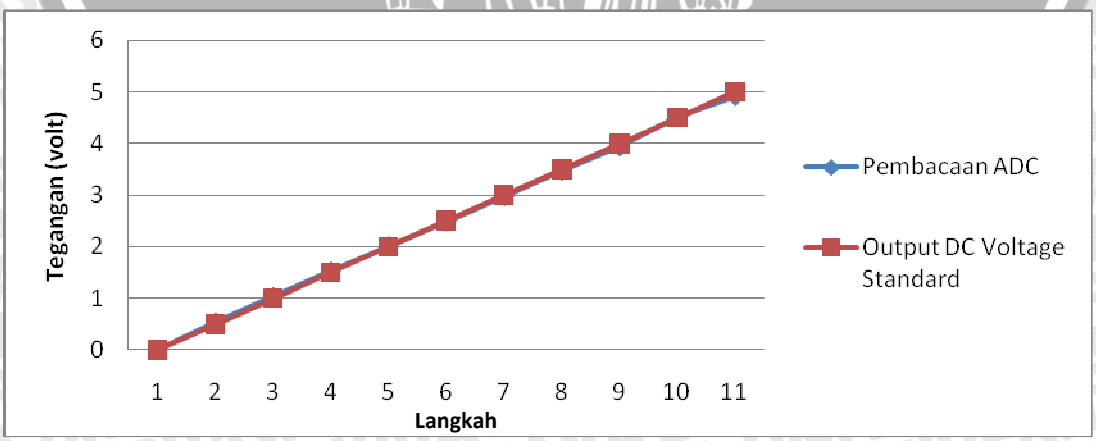


Gambar 5. 7. Blok Diagram Pengujian ADC

d. Hasil pengujian dan analisa

Tabel 5. 5. Hasil Pengujian ADC

No.	Output DC voltage standard (V)	Pembacaan ADC MK (V)	Error	Persentase Error
1	0.5	0.549	0.049	9.8
2	1	1.0392	0.0392	3.92
3	1.5	1.5294	0.0294	1.96
4	2	2.0196	0.0196	0.98
5	2.5	2.4901	0.0099	0.396
6	3	2.9803	0.0197	0.66
7	3.5	3.4709	0.0291	0.83
8	4	3.9411	0.0589	1.47
9	4.5	4.5098	0.0098	0.22
10	5	4.9215	0.0785	1.57
Rata-rata persentase error pembacaan ADC				1.98



Gambar 5. 8. Grafik pembacaan ADC MK terhadap Output DC Voltage Standard

Berdasarkan hasil pengujian Tabel 5.2. diketahui bahwa pada pengujian ADC, penyimpangan rata-rata nya sebesar 1,98%.

5.1.3. Pengujian Komunikasi Serial

a. Tujuan

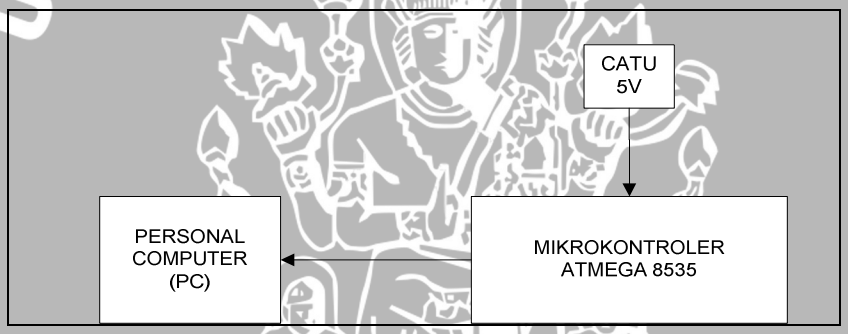
Mengetahui kinerja dari konverter RS-232 - USB

b. Peralatan

1. Mikrokontroler Atmega 8535
2. Kabel DB-9
3. Konverter RS-232 - USB
4. Komputer

c. Langkah pengujan

1. Merangkai peralatan seperti Gambar 5.4.



Gambar 5.4. Blok Diagram Pengujian Komunikasi Serial

2. Mengisi mikrokontroler dengan program sederhana untuk mengirim beberapa karakter ke komputer melalui komunikasi serial.
3. Menghubungkan ujung kabel DB-9 dengan konverter RS-232 – USB dan kemudian dihubungkan ke komputer melalui USB
4. Mengaktifkan mikrokontroler dan program hyper terminal pada komputer.
5. Mengamati perubahan tampilan pada hyper terminal untuk setiap perubahan karakter yang dikirimkan.

d. Hasil pengujian dan analisa

Tabel 5.3. Hasil Pengujian Komunikasi Serial

Karakter yang dikirim	Karakter yang diterima
A	A
B	B
1	1
2	2

Berdasarkan dari hasil pengujian Tabel 5.3. diketahui bahwa karakter yang diterima oleh komputer sama dengan karakter yang dikirim oleh mikrokontroler.

5.1.4. Pengujian Pengondisi Sinyal

a. Tujuan

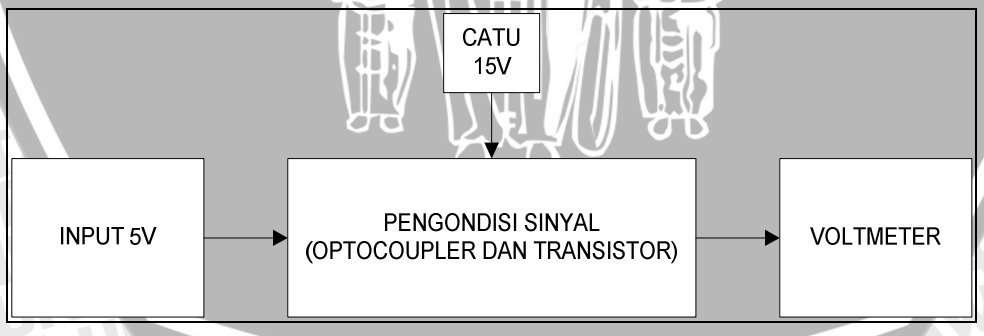
Mengetahui kondisi awal dari pengondisi apabila diberi masukan tertentu.

b. Peralatan

1. Rangkaian pengondisi sinyal terdiri dari optocoupler dan transistor
2. Catu daya 5V dan 15V

c. Langkah pengujian

1. Merangkai peralatan seperti Gambar 5.5.
2. Memberikan input sebesar 5V kepada pengondisi sinyal
3. Mencatat keluaran pengondisi sinyal.



Gambar 5. 5. Blok diagram pengujian pegondisi sinyal

d. Hasil pengujian dan analisa

Tabel 5.4. Hasil Pengujian Pengondisi Sinyal

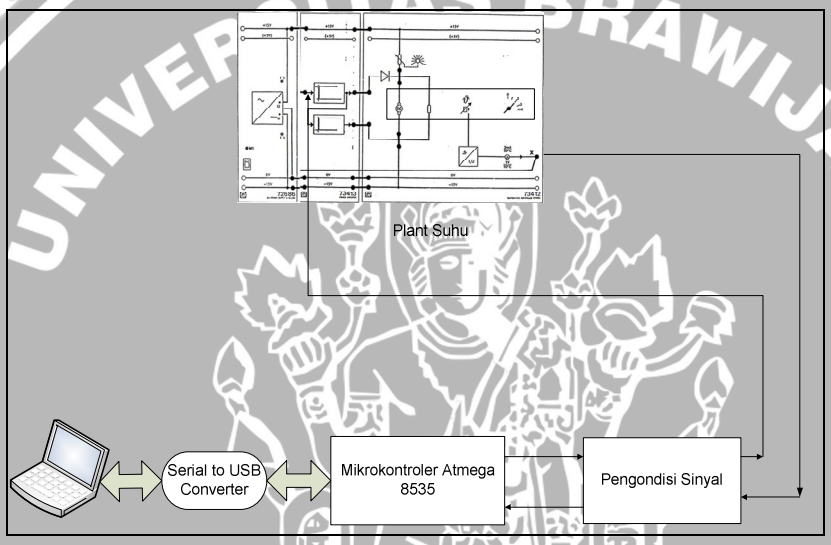
Input	Output
0 V	0 V
5 V	15 V

Berdasarkan hasil pengujian pada Tabel 5.4. diketahui bahwa rangkaian pengondisi sinyal dapat bekerja sesuai harapan, yaitu aktif (output=15 V) ketika input= 5V dan tidak aktif (Output= 0V) ketika input= 0V.

5.2. Pengujian Keseluruhan Sistem Identifikasi

a. Tujuan

Pengujian ini bertujuan untuk mengetahui kinerja perangkat keras dengan perangkat lunak setelah diintegrasikan bersama-sama. Cara pengujiannya yaitu dengan merangkai blok seperti ditunjukkan oleh Gambar 5.6.



Gambar 5. 6. Blok Diagram Pengujian Sistem Secara Keseluruhan

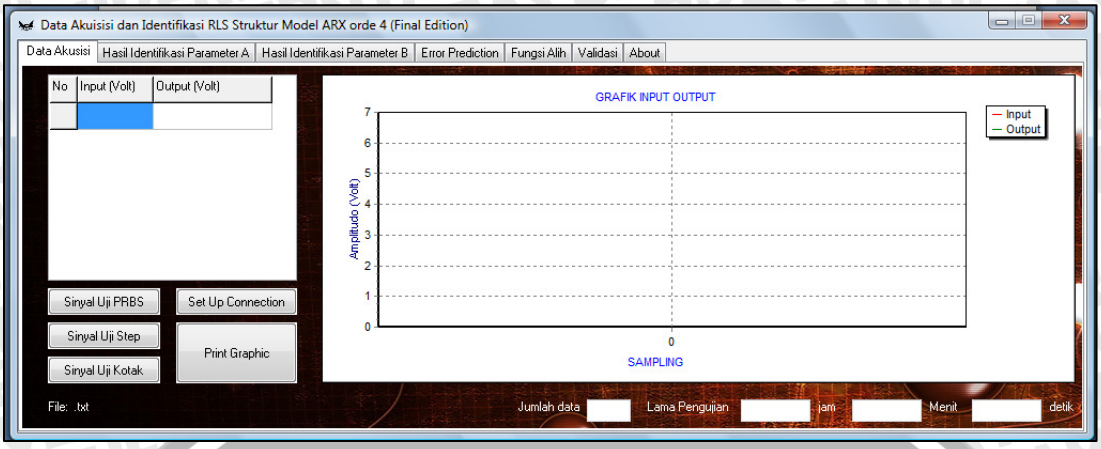
b. Peralatan

1. Minimum sistem atmega 8535
2. Pengondisi sinyal
3. Serial to USB konverter
4. Plant suhu
5. Komputer

c. Hasil Pengujian dan Analisa

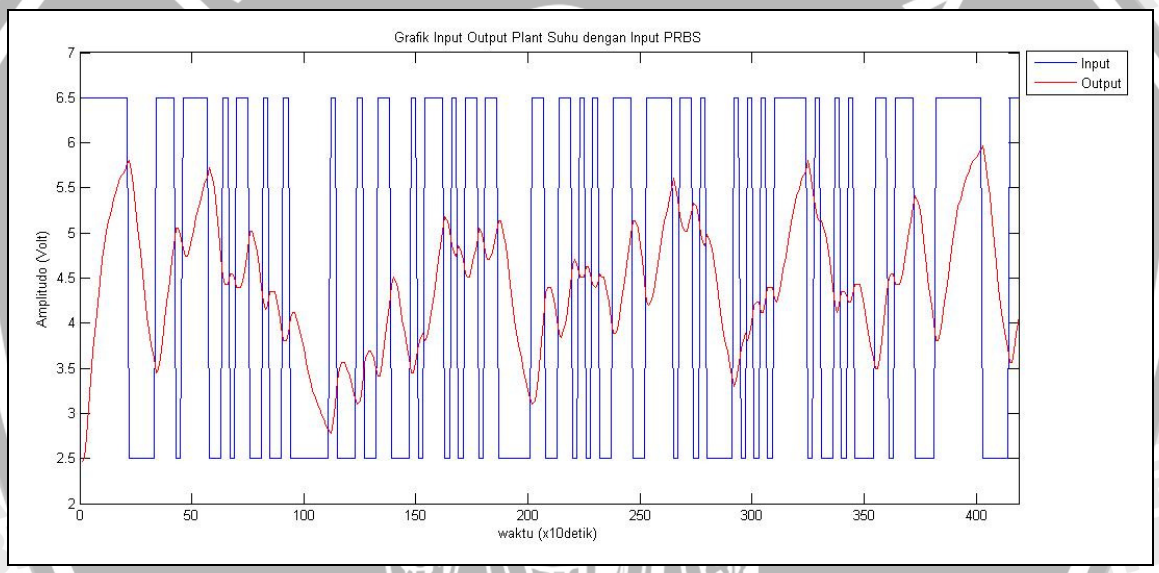
5.2.1. Pengambilan Data Input Output

Proses pengambilan data dimulai dengan menekan tombol salah satu tombol sesuai dengan sinyal uji yang akan dipilih pada jendela perangkat lunak, seperti terlihat pada gambar 5.7.



Gambar 5.7. Tampilan Perangkat Lunak Pengambil Data

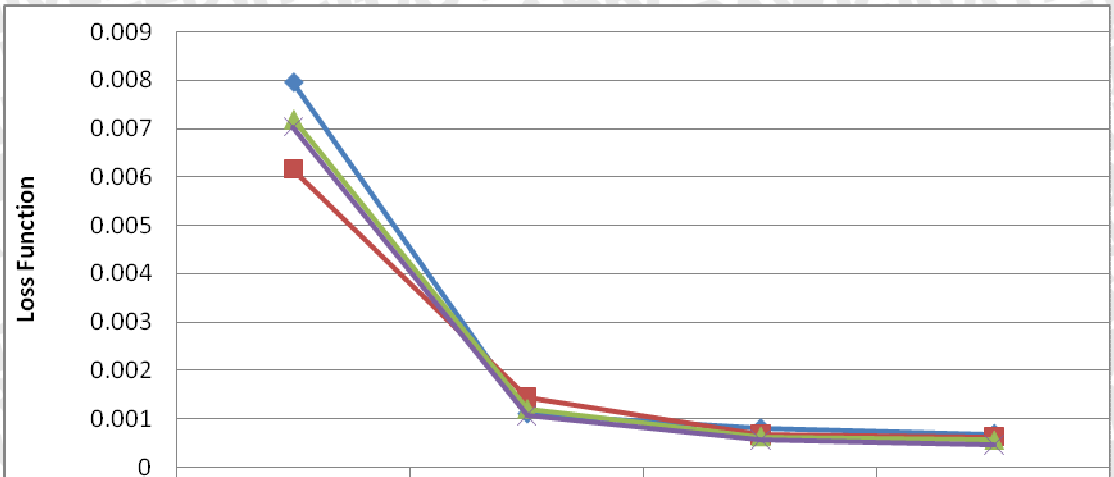
Grafik sinyal uji PRBS dan respon plant suhu ditampilkan pada Gambar 5.8.



Gambar 5.8. Grafik Input Output Plant Suhu dengan sinyal Uji PRBS

5.2.2. Pemilihan Orde Sistem

Pemilihan orde sistem ditentukan dengan melihat pada orde berapa nilai *loss function* dari *error prediction* diperoleh nilai terkecil. hasil perbandingan antara orde model dengan *loss function* ditunjukkan oleh Gambar 5.9.



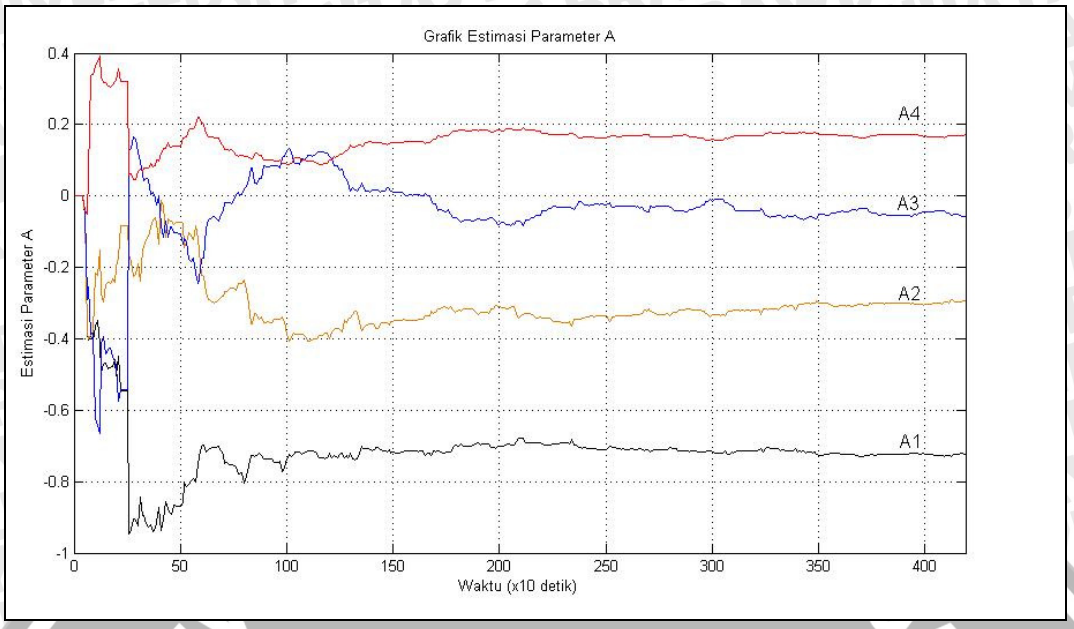
Orde	1	2	3	4
data 1	0.0079569	0.00109472	0.000795727	0.00068052
data 2	0.00616276	0.00143616	0.00067183	0.000630968
data 3	0.00720418	0.00120343	0.000631772	0.000554703
data 4	0.00703931	0.00106874	0.000561238	0.000460896

Gambar 5.9. Grafik Hubungan Orde dengan *Loss Function*

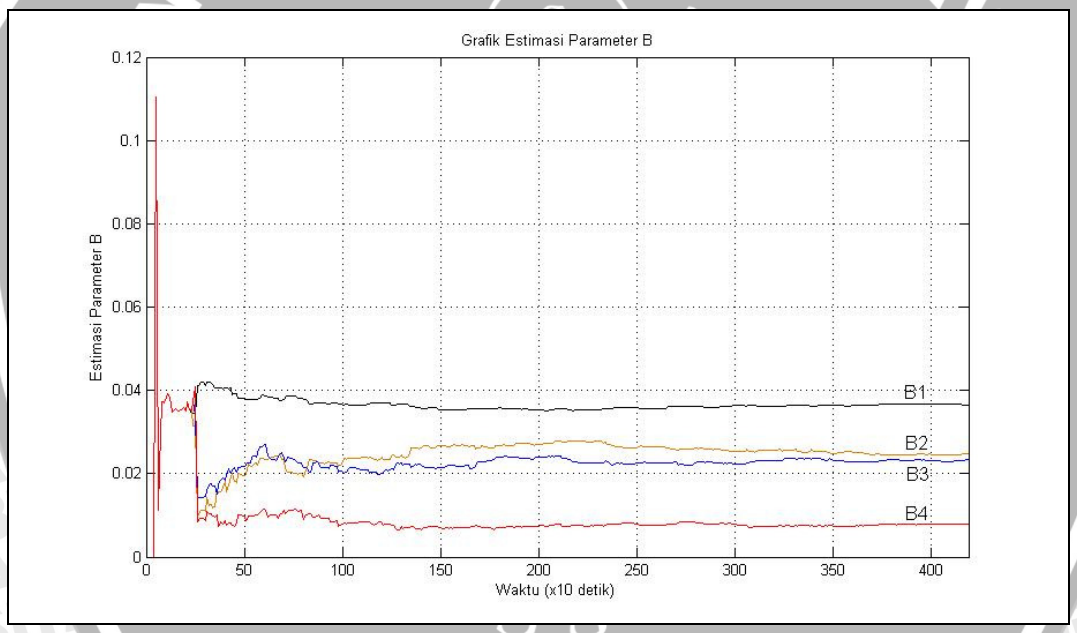
Dari Gambar 5.9. dapat dilihat bahwa dari 4 data yang ada menunjukkan kecenderungan yang sama yaitu semakin tinggi orde model maka *loss function* yang diperoleh juga semakin kecil, oleh karena itu orde 4 dipilih sebagai orde dari model ARX.

5.2.3. Proses Estimasi Parameter

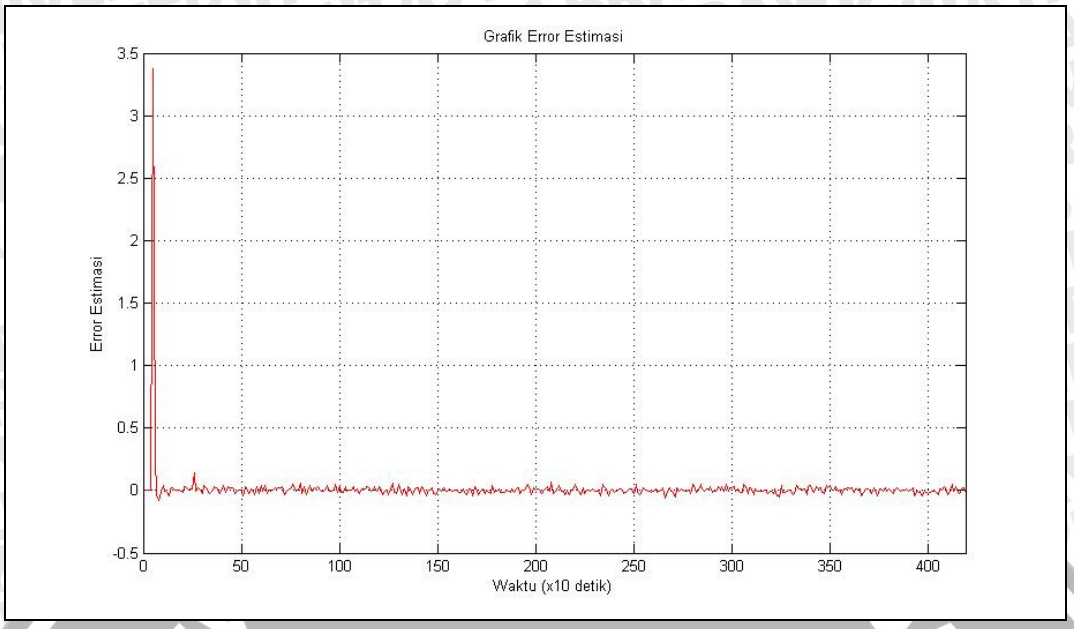
Proses estimasi dimulai secara otomatis setelah data ke-5 diterima oleh perangkat lunak identifikasi dan persamaan yang digunakan untuk proses estimasi ini berasal dari persamaan 2.16 hingga 2.23. Setelah dilakukan proses estimasi maka didapatkan grafik estimasi seperti ditunjukkan oleh gambar 5.10, 5.11, 5.12.



Gambar5.10. Estimasi Parameter A (denominator)



Gambar 5.11. Estimasi Parameter B (numerator)



Gambar 5.12. Error Estimasi

Proses estimasi parameter A terlihat dari Gambar 5.10 dan hasil estimasi relatif tidak terlalu besar perubahannya pada kisaran data ke-150 hingga 200. Gambar 5.11 menunjukkan proses estimasi parameter B dan proses ini lebih cepat mencapai daerah dimana parameter estimasi sudah tidak mengalami perubahan besar yaitu pada kisaran data ke-100. Sedangkan gambar yang terakhir yaitu Gambar 5.12 menunjukkan perubahan error estimasi. Grafik ini diperoleh dari pengurangan nilai *output* pengukuran dengan nilai *output* model dan error estimasi sudah relatif kecil ketika data kurang dari 50.

Hasil akhir dari estimasi ditunjukkan oleh parameter A1 hingga B4 dan hasilnya disajikan sbb:

A1= -0.72367	A2= -0.29387	A3= -0.058514	A4= 0.17061
B1= 0.036505	B2= 0.024752	B3= 0.023322	B4= 0.007752

5.2.4. Validasi

Ada 3 macam validasi yang dilakukan yaitu *whiteness test*, Akaike's FPE, dan uji keakurasian.

5.2.4.1. Whiteness Test

Syarat minimum lolos *whiteness test* ditentukan dari:

$$|RN(0)| = 1 \text{ dan } |RN(i)| \leq 0.15; i > 0$$

Diperoleh hasil sbb:

R (0) = 0.00046034

RN (0) = 1

RN (1) = -0.065968

RN (2) = -0.080171

RN (3) = 0.04237

RN (4) = 0.10554

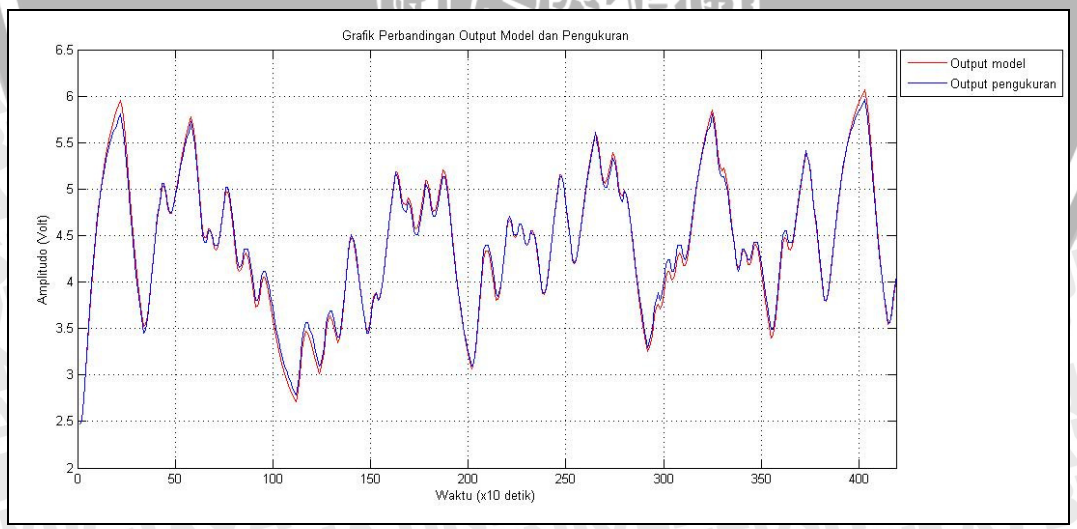
Dari hasil *whiteness test* tersebut menunjukkan model telah lolos *whiteness test* dan berarti model yang diperoleh telah mewakili data hasil pengukuran plant.

5.2.4.2. Akaike's FPE (Final Prediction Error)

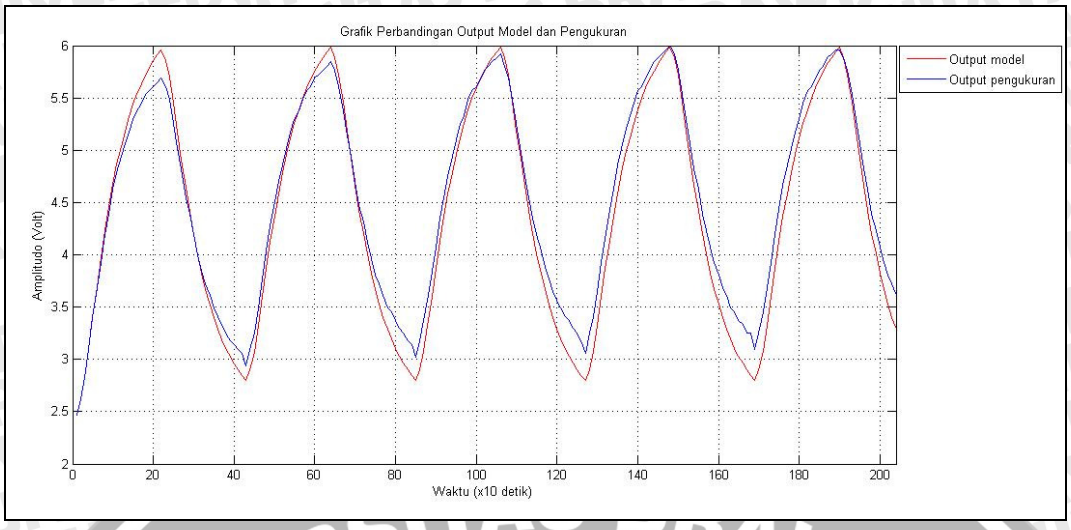
Dari persamaan 2.34 diperoleh hasil FPE= 0.0004697. Hasil yang diperoleh menunjukkan kualitas model yang diperoleh, semakin kecil nilai FPE maka semakin baik kualitas model yang diperoleh.

5.2.4.3. Uji Keakurasian

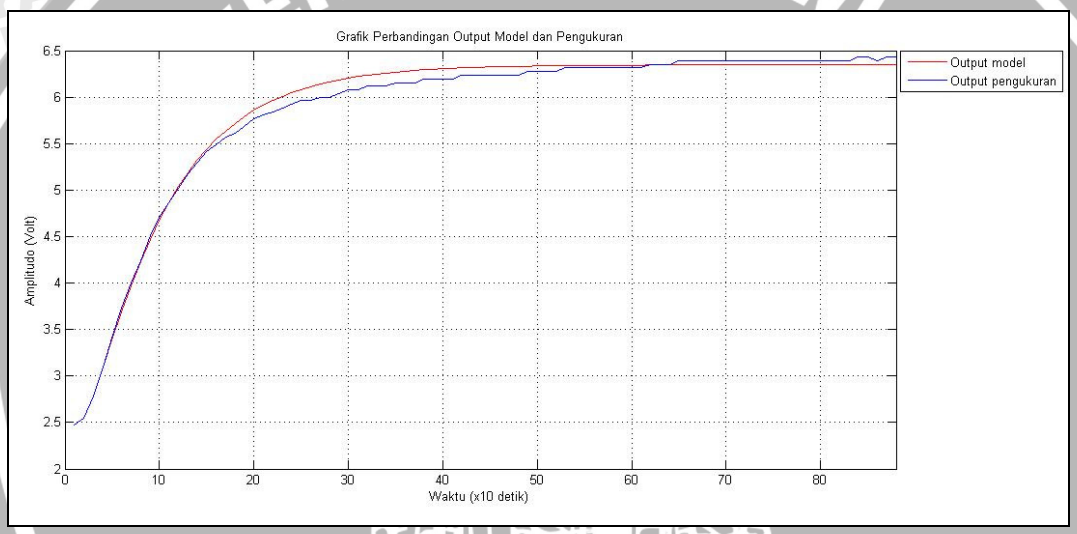
Uji keakurasian dilakukan dengan cara membandingkan antara keluaran model dengan plant yang sebenarnya yang diberi sinyal masukan PRBS, kotak, dan step. Gambar 5.13 – 5.15 menunjukkan perbandingan keduanya.



Gambar 5.13. Grafik Perbandingan Antara Output Plant dan Model dengan Input PRBS



Gambar 5.14 Grafik Perbandingan Antara Output Plant dan Model Input Kotak



Gambar 5.15 Grafik Perbandingan Antara Output Plant dan Model Input Step

Dari persamaan 2.26 dapat dihitung persentase *fit* antara data *output* pengukuran dan data *output* dari model seperti ditunjukkan oleh Tabel 5.5

Tabel 5.5. Perbandingan fit dengan sinyal uji berbeda

Sinyal Uji	*FIT* (%)
PRBS	91.3989
Kotak	79.922
Step	91.532

Dari Tabel 5.5. terlihat bahwa setiap pengujian menghasilkan nilai *fit* yang mendekati 100 dan itu berarti *output* model mendekati *output* pengukuran.

BAB VI
PENUTUP

6.1. Kesimpulan

Berdasarkan pengujian yang telah dilakukan baik pengujian perblok rangkaian maupun pengujian sistem secara keseluruhan diperoleh kesimpulan sebagai berikut:

1. Seluruh sistem identifikasi baik *hardware* maupun *software* sudah dapat berfungsi sesuai yang diharapkan yaitu data dari plant dapat diterima oleh komputer dan kemudian proses identifikasi dapat berlangsung.
2. Untuk sistem plant suhu ini orde model yang terbaik adalah orde 4 karena nilai *loss function* pada orde ini paling kecil dibandingkan dengan orde 1 hingga 3.
3. Hasil parameter yang diperoleh adalah A1= -0.72367, A2= -0.29387, A3= -0.058514, A4= 0.17061, B1= 0.036505, B2= 0.024752, B3= 0.023322, B4= 0.007752 .
4. Hasil *whiteness test* yang diperoleh adalah R(0)= 0.00046034, RN(0)= 1, RN(1)= -0.065968, RN(2)= -0.080171, RN(3)= 0.04237, RN(4)= 0.10554.
5. Hasil validasi Akaike's FPE= 0.0004697.
6. Hasil Pengujian keakurasian dengan sinyal uji PRBS, kotak, step berturut-turut adalah 91.3989%, 79.922%, 91.532%.

6.2. Saran

Meskipun alat ini sudah dapat bekerja sesuai dengan spesifikasi yang diinginkan, namun ada beberapa hal yang dapat dikembangkan dari alat ini di kemudian hari, antara lain:

1. Disarankan untuk pengembangan lebih lanjut untuk validasi secara *online*.
2. Disarankan agar hasil estimasi digunakan untuk metode kontrol yang *real time* juga seperti *self tuning regulator* karena kondisi plant yang dinamis.

3. Untuk membandingkan pengaruh hasil identifikasi terhadap struktur model dan metode identifikasi disarankan untuk mecoba struktur model lain seperti AR, ARMA, OE, dll dan metode identifikasi yang lain seperti ELS, RML, OEEPM, dll.

DAFTAR PUSTAKA

ATMEL. 2007. **ATMEGA8535/ATMEGA8535L, 8-bit AVR Microcontroller with 8 Kbytes in System Programable Flash**.

Bobál V, J. Böhm, J. Fessl dan J. Macháček. 2005. **Digital Self-tuning Controllers Algorithms, Implementation and Applications.** Germany: Springer-Verlag London Limited.

Brogan, William. 2000. **Model**. In :**The Electrical Engineering Handbook**. Richard dorf. Boca Raton: CRC Press LLC.

Modul Praktikum Sistem Kontrol. **Sistem Pengaturan Suhu dengan Kontroler PID**. Laboratorium Sistem Kontrol Teknik Elektro Universitas Brawijaya. Malang.

Landau, Ioan dan Gianluca Zito. 2006. **Digital Control Systems Design, Identification and Implementation**. Germany: Springer-Verlag London Limited.

Ljung, Lennart. 1999. **System Identification Theory For The User second edition**. New Jersey: Prentice Hall.

LAMPIRAN

A

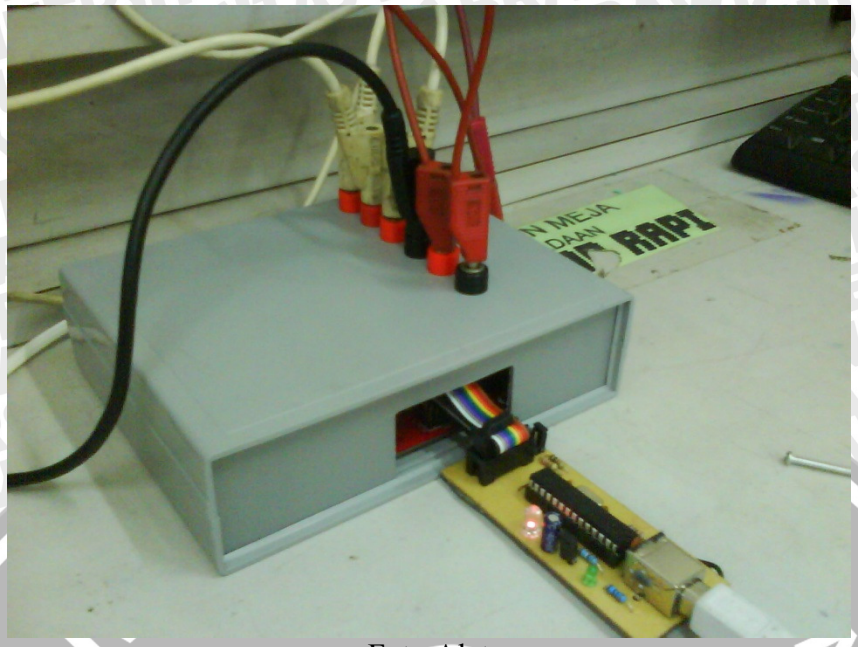


Foto Alat

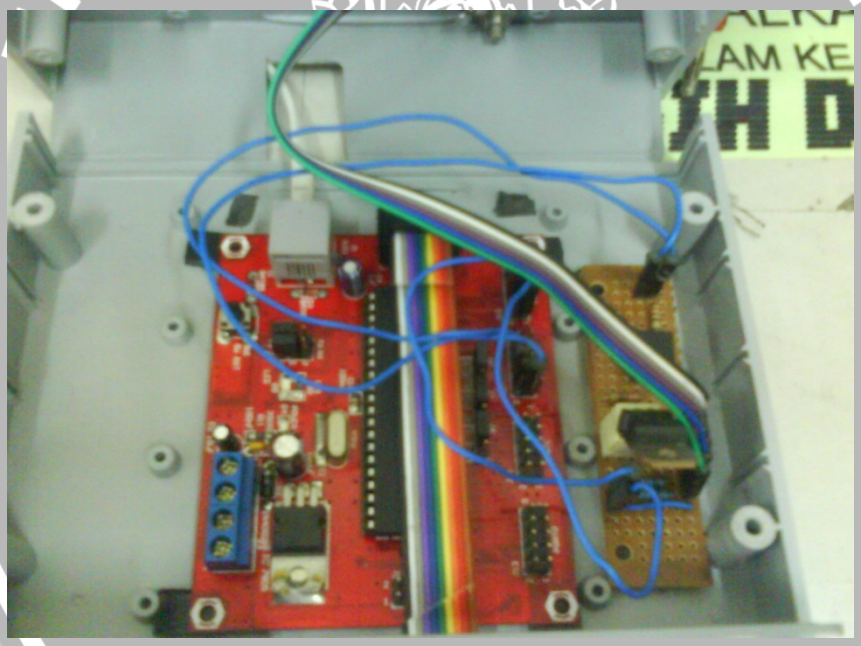


Foto bagian dalam alat

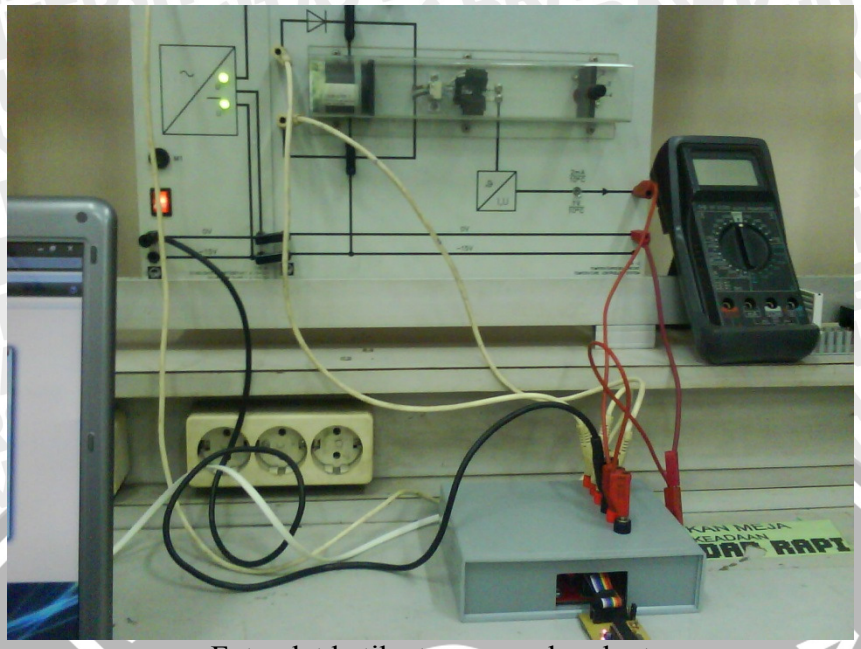


Foto alat ketika terpasang ke plant

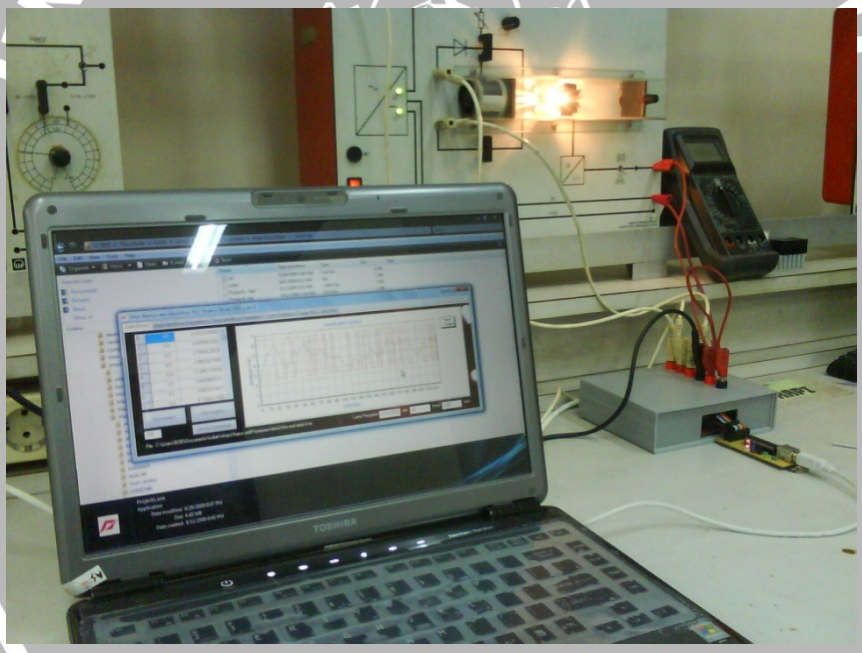


Foto keseluruhan sistem terpasang

LAMPIRAN

B

```
/*****************************************
*******
This program was produced by the
CodeWizardAVR V1.24.8d Professional
Automatic Program Generator
© Copyright 1998-2006 Pavel Haiduc, HP
InfoTech s.r.l.
http://www.hpinfotech.com

Project : PRBS generator + ADC + kirim
data
Version : 2.0 final
Date    : 5/15/2009
Author  : bob
Company : bob
Comments: 7 bit 3.5 menit


Chip type           : ATmega8535
Program type        : Application
Clock frequency     : 4.000000 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 128
*****************************************
******/

#include <mega8535.h>
#include <delay.h>
#include <stdio.h>
#define ADC_VREF_TYPE 0x20

unsigned char  a[7],b[7],data[9];
int i,j,m,n,k,temp;
char data_konversi;
int x,y,z;

void create_reg();
void xor();
void geser_bit();
void after_reg();
void create_new_reg();
void init_USART();
void kirim_data();
void init_ADC();
void prepare();

#define RXB8 1
#define TXB8 0
#define UPE 2
#define OVR 3
#define FE 4
#define UDRE 5
#define RXC 7

#define FRAMING_ERROR (1<<FE)
#define PARITY_ERROR (1<<UPE)
#define DATA_OVERRUN (1<<OVR)
#define DATA_REGISTER_EMPTY
(1<<UDRE)
#define RX_COMPLETE (1<<RXC)

// USART Receiver buffer
#define RX_BUFFER_SIZE 8
char rx_buffer[RX_BUFFER_SIZE];

#if RX_BUFFER_SIZE<256
unsigned char
rx_wr_index,rx_rd_index,rx_counter;
#else
unsigned int
rx_wr_index,rx_rd_index,rx_counter;
#endif

// This flag is set on USART Receiver buffer
overflow
bit rx_buffer_overflow;

// USART Receiver interrupt service routine
interrupt [USART_RXC] void
usart_rx_isr(void)
{
char status;
status=UCSRA;
data[1]=UDR;
if ((status & (FRAMING_ERROR |
PARITY_ERROR |
DATA_OVERRUN))==0)
  {
  rx_buffer[rx_wr_index]=data[1];
  if (++rx_wr_index ==
RX_BUFFER_SIZE) rx_wr_index=0;
  if (++rx_counter == RX_BUFFER_SIZE)
     {
     rx_counter=0;
     rx_buffer_overflow=1;
     };
  };
while(data==0x0f){};
}

#ifndef _DEBUG_TERMINAL_IO_
// Get a character from the USART Receiver
buffer
#define _ALTERNATE_GETCHAR_
#pragma used+
char getchar(void)
{
char data;
while (rx_counter==0);
data=rx_buffer[rx_rd_index];
if (++rx_rd_index == RX_BUFFER_SIZE)
rx_rd_index=0;
#asm("cli")
--rx_counter;
#asm("sei")
```

```
return data;
}
#pragma used-
#endif

// setting USART
void init_USART()
{
UCSRA=0x00;
UCSRB=0x98;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x19;
}

// buat register awal sebanyak 7
void create_reg()
{
 for (i=0;i<=6;i++)
  {
          a[i]=1;
  }
}

// xor kan tab 4 dan 7
void xor()
{
 temp=a[3]^a[6];
}

// geser bit register lama
void geser_bit()
{
 for (j=6;j>=1;j--)
 {
          b[j]=a[j-1];
 }
}

// isi bit ke 1 dengan hasil xor
void after_reg()
{
 b[0]=temp;
}

// isi register lama dengan register baru
void create_new_reg()
{
 for(k=0;k<=6;k++)
  {
          a[k]=b[k];
  }
}
// initialization ADC
void init_ADC()
{
ACSR=0x80;
SFIOR=0x00;
// ADC initialization
// ADC Clock frequency: 1000.000 kHz
// ADC Voltage Reference: AVCC pin
// ADC High Speed Mode: Off
// ADC Auto Trigger Source: None
// Only the 8 most significant bits of
// the AD conversion result are used
ADMUX=ADC_VREF_TYPE;
ADCSRA=0x82;
SFIOR&=0xEF;
}

unsigned char read_adc(unsigned char
adc_input)
{
ADMUX=adc_input|ADC_VREF_TYPE;
// Start the AD conversion
ADCSRA|=0x40;
// Wait for the AD conversion to complete
while ((ADCSRA & 0x10)==0);
ADCSRA|=0x10;
return ADCH;
}

// konversi data
void prepare()
{
 x=data_konversi/100;
 data_konversi=data_konversi-(100*x);
 y=data_konversi/10;
 z=data_konversi-(10*y);
}
// kirim data
void kirim_data()
{
 for(m=0;m<=2;m++)
 {
  data_konversi=read_adc(1);
  prepare();
  printf("A");
  if(a[6]==1)
   { printf("6.5");}
  else
   { printf("2.5");}
  putchar(0x20);
  printf("B");
  printf("%d",x);
  printf("%d",y);
  printf("%d",z);
  putchar(13);
  delay_ms(10000);
 }
}

void main()
{
```

```
PORTB=0x00;
DDRB=0xFF;
PORTA=0xFF;
DDRA=0x00;
init_USART();
init_ADC();
#asm("sei")

while(1)
{
// sinyal uji PRBS
while (data[1]=='P')
  {
    create_reg();
    for (n=0;n<127;n++) // sekali percobaan
sebanyak 127 bit PRBS
      {
            PORTB.1=a[6];
            xor();
            geser_bit();
            after_reg();
            kirim_data();
            create_new_reg();
      }
  }
// sinyal uji step
while (data[1]=='S')
  {
   PORTB.1=1;
   data_konversi=read_adc(1);
   printf("A");
   printf("6.5");
   x=data_konversi/100;
   data_konversi=data_konversi-(100*x);
   y=data_konversi/10;
   z=data_konversi-(10*y);
   putchar(0x20);
   printf("B");
   printf("%d",x);
   printf("%d",y);
   printf("%d",z);
   putchar(13);
   delay_ms(10000);
  }
// sinyal uji kotak
while (data[1]=='K')
  {
  PORTB.1=1;
  for(k=1;k<=21;k++)
   {
   data_konversi=read_adc(1);
   printf("A");
   printf("6.5");
   x=data_konversi/100;
   data_konversi=data_konversi-(100*x);
   y=data_konversi/10;
   z=data_konversi-(10*y);
   putchar(0x20);
   printf("B");
   printf("%d",x);
   printf("%d",y);
   printf("%d",z);
   putchar(13);
   delay_ms(10000);
   }
  PORTB.1=0;
  for(n=1;n<=21;n++)
   {
   data_konversi=read_adc(1);
   printf("A");
   printf("2.5");
   x=data_konversi/100;
   data_konversi=data_konversi-(100*x);
   y=data_konversi/10;
   z=data_konversi-(10*y);
   putchar(0x20);
   printf("B");
   printf("%d",x);
   printf("%d",y);
   printf("%d",z);
   putchar(13);
   delay_ms(10000);
   }
  }
 };
}
```

LAMPIRAN C

```
unit Unit1;
interface
uses
  Windows, Messages, SysUtils, Variants,
Classes, Graphics, Controls, Forms,
  Dialogs, CPort, StdCtrls, XPMan, jpeg,
ExtCtrls, DB, DBTables, Grids,
  DBGrids, TeEngine, Series, TeeProcs,
Chart, ExtDlgs, ComCtrls, Menus;

type
  TForm1 = class(TForm)
    ComPort1: TComPort;
    Button1: TButton;
    XPManifest1: TXPManifest;
    SaveDialog1: TSaveDialog;
    Label1: TLabel;
    Label2: TLabel;
    PrintDialog1: TPrintDialog;
    Button2: TButton;
    StringGrid1: TStringGrid;
    OpenDialog1: TOpenDialog;
    Memo1: TMemo;
    Label3: TLabel;
    Edit2: TEdit;
    Label6: TLabel;
    Button5: TButton;
    Label14: TLabel;
    Label15: TLabel;
    Label16: TLabel;
    Label18: TLabel;
    Edit4: TEdit;
    Edit3: TEdit;
    Memo2: TMemo;
    Memo3: TMemo;
    Edit1: TEdit;
    Edit5: TEdit;
    Edit6: TEdit;
    Edit7: TEdit;
    Edit8: TEdit;
    Edit9: TEdit;
    Edit10: TEdit;
    Edit11: TEdit;
    Edit12: TEdit;
    Memo4: TMemo;
    Memo5: TMemo;
    Memo6: TMemo;
    Memo7: TMemo;
    Memo8: TMemo;
    Memo9: TMemo;
    Memo10: TMemo;
    Memo11: TMemo;
    Edit13: TEdit;
    Edit14: TEdit;
    Edit15: TEdit;
    Edit16: TEdit;
    Edit17: TEdit;
    Edit18: TEdit;
    Edit19: TEdit;
    Edit20: TEdit;
    Edit21: TEdit;
    Edit22: TEdit;
    Edit23: TEdit;
    Edit24: TEdit;
    Edit25: TEdit;
    Edit26: TEdit;
    Edit27: TEdit;
    Edit28: TEdit;
    Edit29: TEdit;
    Edit30: TEdit;
    Edit31: TEdit;
    Edit32: TEdit;
    Edit33: TEdit;
    Edit34: TEdit;
    Memo12: TMemo;
    Memo13: TMemo;
    Memo14: TMemo;
    Memo15: TMemo;
    Memo16: TMemo;
    Memo17: TMemo;
    Memo18: TMemo;
    Memo19: TMemo;
    Memo20: TMemo;
    Memo21: TMemo;
    Memo22: TMemo;
    Memo23: TMemo;
    Memo24: TMemo;
    Memo25: TMemo;
    Memo26: TMemo;
    Memo27: TMemo;
    Memo28: TMemo;
    Memo29: TMemo;
    Memo30: TMemo;
    Memo31: TMemo;
    Memo32: TMemo;
    Memo33: TMemo;
    Memo34: TMemo;
    Memo35: TMemo;
    Memo36: TMemo;
    Memo37: TMemo;
    Memo38: TMemo;
    Memo39: TMemo;
    Memo40: TMemo;
    Memo41: TMemo;
    Memo42: TMemo;
    Memo43: TMemo;
    Memo44: TMemo;
    Memo45: TMemo;
    Memo46: TMemo;
    Memo47: TMemo;
    Memo48: TMemo;
    Memo49: TMemo;
    Memo50: TMemo;
    Memo51: TMemo;
    Memo52: TMemo;
    Memo53: TMemo;
```

Memo54: TMemo;
Memo55: TMemo;
Memo56: TMemo;
Memo57: TMemo;
Memo58: TMemo;
Memo59: TMemo;
Memo60: TMemo;
Memo61: TMemo;
Memo62: TMemo;
Memo63: TMemo;
Memo64: TMemo;
Memo65: TMemo;
Memo66: TMemo;
Memo67: TMemo;
Memo68: TMemo;
Memo69: TMemo;
Memo70: TMemo;
Memo71: TMemo;
Memo72: TMemo;
Memo73: TMemo;
Memo74: TMemo;
Memo75: TMemo;
Edit35: TEdit;
Edit36: TEdit;
Edit37: TEdit;
Edit38: TEdit;
Edit39: TEdit;
Edit40: TEdit;
Edit41: TEdit;
Edit42: TEdit;
Edit43: TEdit;
Edit44: TEdit;
Edit45: TEdit;
Edit46: TEdit;
Edit47: TEdit;
Edit48: TEdit;
Edit49: TEdit;
Edit50: TEdit;
Edit51: TEdit;
Edit52: TEdit;
Edit53: TEdit;
Edit54: TEdit;
Edit55: TEdit;
Edit56: TEdit;
Edit57: TEdit;
Edit58: TEdit;
Edit59: TEdit;
Edit60: TEdit;
Edit61: TEdit;
Edit62: TEdit;
Edit63: TEdit;
Edit64: TEdit;
Edit65: TEdit;
Edit66: TEdit;
Edit67: TEdit;
Edit68: TEdit;
Edit69: TEdit;
Edit70: TEdit;
Edit71: TEdit;
Edit72: TEdit;
Edit73: TEdit;
Edit74: TEdit;
Edit75: TEdit;
Edit76: TEdit;
Edit77: TEdit;
Edit78: TEdit;
Edit79: TEdit;
Edit80: TEdit;
Edit81: TEdit;
Edit82: TEdit;
Edit83: TEdit;
Edit84: TEdit;
image1: TImage;
PageControl1: TPageControl;
TabSheet1: TTabSheet;
TabSheet2: TTabSheet;
TabSheet3: TTabSheet;
TabSheet4: TTabSheet;
TabSheet5: TTabSheet;
Chart1: TChart;
Series1: TLineSeries;
Series2: TLineSeries;
Chart2: TChart;
Series3: TLineSeries;
Series4: TLineSeries;
Series5: TLineSeries;
Series6: TLineSeries;
Chart3: TChart;
Series7: TLineSeries;
Series8: TLineSeries;
Series9: TLineSeries;
Series10: TLineSeries;
Edit85: TEdit;
Edit86: TEdit;
Edit87: TEdit;
Edit88: TEdit;
Image2: TImage;
Label4: TLabel;
Label5: TLabel;
Label7: TLabel;
Label8: TLabel;
Image3: TImage;
Edit89: TEdit;
Edit90: TEdit;
Edit91: TEdit;
Edit92: TEdit;
Image4: TImage;
Label9: TLabel;
Label10: TLabel;
Label11: TLabel;
Label12: TLabel;
Chart4: TChart;
Series11: TLineSeries;
Edit93: TEdit;
Image5: TImage;
Label13: TLabel;

TabSheet6: TTabSheet;
Image6: TImage;
Label17: TLabel;
Edit94: TEdit;
Edit95: TEdit;
Edit96: TEdit;
Edit97: TEdit;
Edit98: TEdit;
Edit99: TEdit;
Edit100: TEdit;
Edit101: TEdit;
Image7: TImage;
Label19: TLabel;
Edit102: TEdit;
Label20: TLabel;
Label21: TLabel;
Edit103: TEdit;
Edit104: TEdit;
Label22: TLabel;
Label23: TLabel;
SaveDialog2: TSaveDialog;
SaveDialog3: TSaveDialog;
SaveDialog4: TSaveDialog;
Button3: TButton;
Button4: TButton;
Button6: TButton;
Memo76: TMemo;
Memo77: TMemo;
Memo78: TMemo;
TabSheet7: TTabSheet;
Image8: TImage;
Button7: TButton;
Button8: TButton;
Chart5: TChart;
Series12: TLineSeries;
Series13: TLineSeries;
Edit105: TEdit;
Edit106: TEdit;
Label24: TLabel;
Label25: TLabel;
Button9: TButton;
Label26: TLabel;
Edit107: TEdit;
Button10: TButton;
Series14: TLineSeries;
Memo79: TMemo;
Memo80: TMemo;
Memo81: TMemo;
Memo82: TMemo;
Edit108: TEdit;
Label27: TLabel;
Button11: TButton;
Button12: TButton;
Label28: TLabel;
Memo83: TMemo;
Button13: TButton;
Button14: TButton;
Label29: TLabel;
Label30: TLabel;
Label31: TLabel;
Label32: TLabel;
Label33: TLabel;
Label34: TLabel;
Edit109: TEdit;
Edit110: TEdit;
Edit111: TEdit;
Edit112: TEdit;
Edit113: TEdit;
Memo84: TMemo;
procedure ComPort1RxChar(Sender: TObject; Count: Integer);
procedure Button1Click(Sender: TObject);
procedure FormCreate(Sender: TObject);
procedure Button2Click(Sender: TObject);
procedure Button5Click(Sender: TObject);
procedure Button3Click(Sender: TObject);
procedure Button6Click(Sender: TObject);
procedure Button4Click(Sender: TObject);
procedure Button8Click(Sender: TObject);
procedure Button7Click(Sender: TObject);
procedure Button9Click(Sender: TObject);
procedure Button10Click(Sender: TObject);
procedure Button11Click(Sender: TObject);
procedure Button12Click(Sender: TObject);
procedure Button13Click(Sender: TObject);
procedure Button14Click(Sender: TObject);
private
{ Private declarations }
public
{ Public declarations }
end;

var
Form1: TForm1;
Buffer: WideString;
Connect: boolean;
F:TextFile;

ep_kuad,out_model,waktu_det,waktu_mnt,
waktu_jam,sinyal_uji, data_adc,
data_analog,data_real,u,y,ep,eps,a1,a2,a3,a4
,b1,b2,b3,b4,u_lama1,u_lama2,u_lama3,u_l

```
ama4,y_lama1,y_lama2,y_lama3,y_lama4:
extended;
  ii,i,j,l: integer;
  nmfile,data,r:string;
  theta,d,temp_eps:array[1..8]of extended;
  temp_theta:array[1..8]of extended;
  temp_cd:array[1..8]of extended;
  c,temp_cdd,sem_c:array[1..64] of
extended;

error,error_kuad,jum_err_kuad,jum_y,loss,f
pe,op1,op2,op3,op4,ip1,ip2,ip3,ip4,yp:
extended;
p1,p2,p3,p4,p5,p6,p7,p8,norm_e,om,om2,op
p1,opp2,opp3,opp4,error_fit,error_fit_kuad,j
um_err_kuad_fit:extended;

error1,error2,error3,error4,jum_e1,jum_e2,j
um_e3,jum_e4: extended;
implementation

{$R *.dfm}

procedure
TForm1.ComPort1RxChar(Sender: TObject;
Count: Integer);
begin
   comport1.ReadStr(data,count);
   Buffer := Buffer + data;
  if (length (Buffer)=10) then begin
    r:= copy(Buffer, pos('A',Buffer)+1, 3);
    sinyal_uji:=StrToFloat(r);
    r:= copy(Buffer, pos('B',Buffer)+1, 3);
    data_adc:=StrToFloat(r);
    data_analog:=((5/255)*data_adc);
    data_real:=data_analog*2;
    AssignFile(F,nmfile);
    Rewrite(F);
    Write(F,Memo1.Text);
    Closefile(F);

Memo1.Lines.Append("+FloatToStr(sinyal_
uji)+' '+FloatToStr(data_real));

Memo2.Lines.Append("+FloatToStr(sinyal_
uji));

Memo3.Lines.Append("+FloatToStr(data_re
al));
    if (l>3) then
    begin
    edit3.Text:=FloatToStr(sinyal_uji);
    edit4.Text:=FloatToStr(data_real);
    // (u-1) ... (u-4)
    edit5.Text:=memo2.Lines[j+3];  // u-1
    edit6.Text:=memo2.Lines[j+2];  // u-2
    edit7.Text:=memo2.Lines[j+1];  // u-3
    edit8.Text:=memo2.Lines[j];    // u-4
    // (y-1) ... (y-4)
    edit9.Text:=memo3.Lines[j+3];  // y-1
    edit10.Text:=memo3.Lines[j+2]; // y-2
    edit11.Text:=memo3.Lines[j+1]; // y-3
    edit12.Text:=memo3.Lines[j];   // y-4
    // basic RLS
    u:=sinyal_uji;
    y:=data_real;

    // (u-1) ... (u-4)
    u_lama1:=strtofloat(edit5.text);
    u_lama2:=strtofloat(edit6.text);
    u_lama3:=strtofloat(edit7.text);
    u_lama4:=strtofloat(edit8.text);

    // (y-1) ... (y-4)
    y_lama1:=strtofloat(edit9.text);
    y_lama2:=strtofloat(edit10.text);
    y_lama3:=strtofloat(edit11.text);
    y_lama4:=strtofloat(edit12.text);
    // theta
    edit13.Text:=memo4.Lines[j]; //theta1
    edit14.Text:=memo5.Lines[j]; //theta2
    edit15.Text:=memo6.Lines[j]; //theta3
    edit16.Text:=memo7.Lines[j]; //theta4
    edit17.Text:=memo8.Lines[j];
    edit18.Text:=memo9.Lines[j];
    edit19.Text:=memo10.Lines[j];
    edit20.Text:=memo11.Lines[j];
    // c1 ... c64
    edit21.Text:=memo12.Lines[j];
    edit22.Text:=memo13.Lines[j];
    edit23.Text:=memo14.Lines[j];
    edit24.Text:=memo15.Lines[j];
    edit25.Text:=memo16.Lines[j];
    edit26.Text:=memo17.Lines[j];
    edit27.Text:=memo18.Lines[j];
    edit28.Text:=memo19.Lines[j];
    edit29.Text:=memo20.Lines[j];
    edit30.Text:=memo21.Lines[j];
    edit31.Text:=memo22.Lines[j];
    edit32.Text:=memo23.Lines[j];
    edit33.Text:=memo24.Lines[j];
    edit34.Text:=memo25.Lines[j];
    edit35.Text:=memo26.Lines[j];
    edit36.Text:=memo27.Lines[j];
    edit37.Text:=memo28.Lines[j];
    edit38.Text:=memo29.Lines[j];
    edit39.Text:=memo30.Lines[j];
    edit40.Text:=memo31.Lines[j];
    edit41.Text:=memo32.Lines[j];
    edit42.Text:=memo33.Lines[j];
    edit43.Text:=memo34.Lines[j];
    edit44.Text:=memo35.Lines[j];
    edit45.Text:=memo36.Lines[j];
    edit46.Text:=memo37.Lines[j];
    edit47.Text:=memo38.Lines[j];
    edit48.Text:=memo39.Lines[j];
```

```
edit49.Text:=memo40.Lines[j];
edit50.Text:=memo41.Lines[j];
edit51.Text:=memo42.Lines[j];
edit52.Text:=memo43.Lines[j];
edit53.Text:=memo44.Lines[j];
edit54.Text:=memo45.Lines[j];
edit55.Text:=memo46.Lines[j];
edit56.Text:=memo47.Lines[j];
edit57.Text:=memo48.Lines[j];
edit58.Text:=memo49.Lines[j];
edit59.Text:=memo50.Lines[j];
edit60.Text:=memo51.Lines[j];
edit61.Text:=memo52.Lines[j];
edit62.Text:=memo53.Lines[j];
edit63.Text:=memo54.Lines[j];
edit64.Text:=memo55.Lines[j];
edit65.Text:=memo56.Lines[j];
edit66.Text:=memo57.Lines[j];
edit67.Text:=memo58.Lines[j];
edit68.Text:=memo59.Lines[j];
edit69.Text:=memo60.Lines[j];
edit70.Text:=memo61.Lines[j];
edit71.Text:=memo62.Lines[j];
edit72.Text:=memo63.Lines[j];
edit73.Text:=memo64.Lines[j];
edit74.Text:=memo65.Lines[j];
edit75.Text:=memo66.Lines[j];
edit76.Text:=memo67.Lines[j];
edit77.Text:=memo68.Lines[j];
edit78.Text:=memo69.Lines[j];
edit79.Text:=memo70.Lines[j];
edit80.Text:=memo71.Lines[j];
edit81.Text:=memo72.Lines[j];
edit82.Text:=memo73.Lines[j];
edit83.Text:=memo74.Lines[j];
edit84.Text:=memo75.Lines[j];

theta[1]:=strtofloat(edit13.Text);
theta[2]:=strtofloat(edit14.Text);
theta[3]:=strtofloat(edit15.Text);
theta[4]:=strtofloat(edit16.Text);
theta[5]:=strtofloat(edit17.Text);
theta[6]:=strtofloat(edit18.Text);
theta[7]:=strtofloat(edit19.Text);
theta[8]:=strtofloat(edit20.Text);

c[1]:=strtofloat(edit21.Text);
c[2]:=strtofloat(edit22.Text);
c[3]:=strtofloat(edit23.Text);
c[4]:=strtofloat(edit24.Text);
c[5]:=strtofloat(edit25.Text);
c[6]:=strtofloat(edit26.Text);
c[7]:=strtofloat(edit27.Text);
c[8]:=strtofloat(edit28.Text);
c[9]:=strtofloat(edit29.Text);
c[10]:=strtofloat(edit30.Text);
c[11]:=strtofloat(edit31.Text);
c[12]:=strtofloat(edit32.Text);
c[13]:=strtofloat(edit33.Text);
c[14]:=strtofloat(edit34.Text);
c[15]:=strtofloat(edit35.Text);
c[16]:=strtofloat(edit36.Text);
c[17]:=strtofloat(edit37.Text);
c[18]:=strtofloat(edit38.Text);
c[19]:=strtofloat(edit39.Text);
c[20]:=strtofloat(edit40.Text);
c[21]:=strtofloat(edit41.Text);
c[22]:=strtofloat(edit42.Text);
c[23]:=strtofloat(edit43.Text);
c[24]:=strtofloat(edit44.Text);
c[25]:=strtofloat(edit45.Text);
c[26]:=strtofloat(edit46.Text);
c[27]:=strtofloat(edit47.Text);
c[28]:=strtofloat(edit48.Text);
c[29]:=strtofloat(edit49.Text);
c[30]:=strtofloat(edit50.Text);
c[31]:=strtofloat(edit51.Text);
c[32]:=strtofloat(edit52.Text);
c[33]:=strtofloat(edit53.Text);
c[34]:=strtofloat(edit54.Text);
c[35]:=strtofloat(edit55.Text);
c[36]:=strtofloat(edit56.Text);
c[37]:=strtofloat(edit57.Text);
c[38]:=strtofloat(edit58.Text);
c[39]:=strtofloat(edit59.Text);
c[40]:=strtofloat(edit60.Text);
c[41]:=strtofloat(edit61.Text);
c[42]:=strtofloat(edit62.Text);
c[43]:=strtofloat(edit63.Text);
c[44]:=strtofloat(edit64.Text);
c[45]:=strtofloat(edit65.Text);
c[46]:=strtofloat(edit66.Text);
c[47]:=strtofloat(edit67.Text);
c[48]:=strtofloat(edit68.Text);
c[49]:=strtofloat(edit69.Text);
c[50]:=strtofloat(edit70.Text);
c[51]:=strtofloat(edit71.Text);
c[52]:=strtofloat(edit72.Text);
c[53]:=strtofloat(edit73.Text);
c[54]:=strtofloat(edit74.Text);
c[55]:=strtofloat(edit75.Text);
c[56]:=strtofloat(edit76.Text);
c[57]:=strtofloat(edit77.Text);
c[58]:=strtofloat(edit78.Text);
c[59]:=strtofloat(edit79.Text);
c[60]:=strtofloat(edit80.Text);
c[61]:=strtofloat(edit81.Text);
c[62]:=strtofloat(edit82.Text);
c[63]:=strtofloat(edit83.Text);
c[64]:=strtofloat(edit84.Text);
d[1]:=-(y_lama1);
d[2]:=-(y_lama2);
d[3]:=-(y_lama3);
d[4]:=-(y_lama4);
d[5]:=u_lama1;
d[6]:=u_lama2;
```

```
    d[7]:=u_lama3;
    d[8]:=u_lama4;
out_model:=(theta[1]*d[1])+(theta[2]*d[2])
+(theta[3]*d[3])+(theta[4]*d[4])+(theta[5]*d
[5])+(theta[6]*d[6])+(theta[7]*d[7])+(theta[
8]*d[8]);
    ep:=y-out_model;
    ep_kuad:=ep*ep;
temp_eps[1]:=(d[1]*c[1])+(d[2]*c[9])+(d[3]
*c[17])+(d[4]*c[25])+(d[5]*c[33])+(d[6]*c[
41])+(d[7]*c[49])+(d[8]*c[57]);
temp_eps[2]:=(d[1]*c[2])+(d[2]*c[10])+(d[3
]*c[18])+(d[4]*c[26])+(d[5]*c[34])+(d[6]*c
[42])+(d[7]*c[50])+(d[8]*c[58]);
temp_eps[3]:=(d[1]*c[3])+(d[2]*c[11])+(d[3
]*c[19])+(d[4]*c[27])+(d[5]*c[35])+(d[6]*c
[43])+(d[7]*c[51])+(d[8]*c[59]);
temp_eps[4]:=(d[1]*c[4])+(d[2]*c[12])+(d[3
]*c[20])+(d[4]*c[28])+(d[5]*c[36])+(d[6]*c
[44])+(d[7]*c[52])+(d[8]*c[60]);
temp_eps[5]:=(d[1]*c[5])+(d[2]*c[13])+(d[3
]*c[21])+(d[4]*c[29])+(d[5]*c[37])+(d[6]*c
[45])+(d[7]*c[53])+(d[8]*c[61]);
temp_eps[6]:=(d[1]*c[6])+(d[2]*c[14])+(d[3
]*c[22])+(d[4]*c[30])+(d[5]*c[38])+(d[6]*c
[46])+(d[7]*c[54])+(d[8]*c[62]);
temp_eps[7]:=(d[1]*c[7])+(d[2]*c[15])+(d[3
]*c[23])+(d[4]*c[31])+(d[5]*c[39])+(d[6]*c
[47])+(d[7]*c[55])+(d[8]*c[63]);
temp_eps[8]:=(d[1]*c[8])+(d[2]*c[16])+(d[3
]*c[24])+(d[4]*c[32])+(d[5]*c[40])+(d[6]*c
[48])+(d[7]*c[56])+(d[8]*c[64]);
eps:=(temp_eps[1]*d[1])+(temp_eps[2]*d[2
])+(temp_eps[3]*d[3])+(temp_eps[4]*d[4])+
(temp_eps[5]*d[5])+(temp_eps[6]*d[6])+(te
mp_eps[7]*d[7])+(temp_eps[8]*d[8]);
temp_theta[1]:=(ep*((c[1]*d[1])+(c[2]*d[2])
+(c[3]*d[3])+(c[4]*d[4])+(c[5]*d[5])+(c[6]*
d[6])+(c[7]*d[7])+(c[8]*d[8])))/(1+eps);
temp_theta[2]:=(ep*((c[9]*d[1])+(c[10]*d[2
])+(c[11]*d[3])+(c[12]*d[4])+(c[13]*d[5])+
(c[14]*d[6])+(c[15]*d[7])+(c[16]*d[8])))/(1
+eps);
temp_theta[3]:=(ep*((c[17]*d[1])+(c[18]*d[
2])+(c[19]*d[3])+(c[20]*d[4])+(c[21]*d[5])
+(c[22]*d[6])+(c[23]*d[7])+(c[24]*d[8])))/(
1+eps);
temp_theta[4]:=(ep*((c[25]*d[1])+(c[26]*d[
2])+(c[27]*d[3])+(c[28]*d[4])+(c[29]*d[5])
+(c[30]*d[6])+(c[31]*d[7])+(c[32]*d[8])))/(
1+eps);
temp_theta[5]:=(ep*((c[33]*d[1])+(c[34]*d[
2])+(c[35]*d[3])+(c[36]*d[4])+(c[37]*d[5])
+(c[38]*d[6])+(c[39]*d[7])+(c[40]*d[8])))/(
1+eps);
temp_theta[6]:=(ep*((c[41]*d[1])+(c[42]*d[
2])+(c[43]*d[3])+(c[44]*d[4])+(c[45]*d[5])
+(c[46]*d[6])+(c[47]*d[7])+(c[48]*d[8])))/(
1+eps);
temp_theta[7]:=(ep*((c[49]*d[1])+(c[50]*d[
2])+(c[51]*d[3])+(c[52]*d[4])+(c[53]*d[5])
+(c[54]*d[6])+(c[55]*d[7])+(c[56]*d[8])))/(
1+eps);

temp_theta[8]:=(ep*((c[57]*d[1])+(c[58]*d[
2])+(c[59]*d[3])+(c[60]*d[4])+(c[61]*d[5])
+(c[62]*d[6])+(c[63]*d[7])+(c[64]*d[8])))/(
1+eps);
    theta[1]:=theta[1]+temp_theta[1];
    theta[2]:=theta[2]+temp_theta[2];
    theta[3]:=theta[3]+temp_theta[3];
    theta[4]:=theta[4]+temp_theta[4];
    theta[5]:=theta[5]+temp_theta[5];
    theta[6]:=theta[6]+temp_theta[6];
    theta[7]:=theta[7]+temp_theta[7];
    theta[8]:=theta[8]+temp_theta[8];
    a1:=theta[1];
    a2:=theta[2];
    a3:=theta[3];
    a4:=theta[4];
    b1:=theta[5];
    b2:=theta[6];
    b3:=theta[7];
    b4:=theta[8];

Memo4.Lines.Append("+FloatToStr(theta[1]
));

Memo5.Lines.Append("+FloatToStr(theta[2]
));

Memo6.Lines.Append("+FloatToStr(theta[3]
));

Memo7.Lines.Append("+FloatToStr(theta[4]
));

Memo8.Lines.Append("+FloatToStr(theta[5]
));

Memo9.Lines.Append("+FloatToStr(theta[6]
));

Memo10.Lines.Append("+FloatToStr(theta[
7]));

Memo11.Lines.Append("+FloatToStr(theta[
8]));
    edit85.Text:=floattostr(a1);
    edit86.Text:=floattostr(a2);
    edit87.Text:=floattostr(a3);
    edit88.Text:=floattostr(a4);
    edit89.Text:=floattostr(b1);
    edit90.Text:=floattostr(b2);
    edit91.Text:=floattostr(b3);
    edit92.Text:=floattostr(b4);
```

```
edit94.Text:=edit85.Text;
edit95.Text:=edit86.Text;
edit96.Text:=edit87.Text;
edit97.Text:=edit88.Text;
edit98.Text:=edit89.Text;
edit99.Text:=edit90.Text;
edit100.Text:=edit91.Text;
edit101.Text:=edit92.Text;

temp_cd[1]:=(c[1]*d[1])+(c[2]*d[2])+(c[3]*
d[3])+(c[4]*d[4])+(c[5]*d[5])+(c[6]*d[6])+(
c[7]*d[7])+(c[8]*d[8]);
temp_cd[2]:=(c[9]*d[1])+(c[10]*d[2])+(c[1
1]*d[3])+(c[12]*d[4])+(c[13]*d[5])+(c[14]*
d[6])+(c[15]*d[7])+(c[16]*d[8]);
temp_cd[3]:=(c[17]*d[1])+(c[18]*d[2])+(c[
19]*d[3])+(c[20]*d[4])+(c[21]*d[5])+(c[22]
*d[6])+(c[23]*d[7])+(c[24]*d[8]);
temp_cd[4]:=(c[25]*d[1])+(c[26]*d[2])+(c[
27]*d[3])+(c[28]*d[4])+(c[29]*d[5])+(c[30]
*d[6])+(c[31]*d[7])+(c[32]*d[8]);
temp_cd[5]:=(c[33]*d[1])+(c[34]*d[2])+(c[
35]*d[3])+(c[36]*d[4])+(c[37]*d[5])+(c[38]
*d[6])+(c[39]*d[7])+(c[40]*d[8]);

temp_cd[6]:=(c[41]*d[1])+(c[42]*d[2])+(c[
43]*d[3])+(c[44]*d[4])+(c[45]*d[5])+(c[46]
*d[6])+(c[47]*d[7])+(c[48]*d[8]);
temp_cd[7]:=(c[49]*d[1])+(c[50]*d[2])+(c[
51]*d[3])+(c[52]*d[4])+(c[53]*d[5])+(c[54]
*d[6])+(c[55]*d[7])+(c[56]*d[8]);
temp_cd[8]:=(c[57]*d[1])+(c[58]*d[2])+(c[
59]*d[3])+(c[60]*d[4])+(c[61]*d[5])+(c[62]
*d[6])+(c[63]*d[7])+(c[64]*d[8]);
  temp_cdd[1]:=temp_cd[1]*d[1];
  temp_cdd[2]:=temp_cd[1]*d[2];
  temp_cdd[3]:=temp_cd[1]*d[3];
  temp_cdd[4]:=temp_cd[1]*d[4];
  temp_cdd[5]:=temp_cd[1]*d[5];
  temp_cdd[6]:=temp_cd[1]*d[6];
  temp_cdd[7]:=temp_cd[1]*d[7];
  temp_cdd[8]:=temp_cd[1]*d[8];
  temp_cdd[9]:=temp_cd[2]*d[1];
  temp_cdd[10]:=temp_cd[2]*d[2];
  temp_cdd[11]:=temp_cd[2]*d[3];
  temp_cdd[12]:=temp_cd[2]*d[4];
  temp_cdd[13]:=temp_cd[2]*d[5];
  temp_cdd[14]:=temp_cd[2]*d[6];
  temp_cdd[15]:=temp_cd[2]*d[7];
  temp_cdd[16]:=temp_cd[2]*d[8];
  temp_cdd[17]:=temp_cd[3]*d[1];
  temp_cdd[18]:=temp_cd[3]*d[2];
  temp_cdd[19]:=temp_cd[3]*d[3];
  temp_cdd[20]:=temp_cd[3]*d[4];
  temp_cdd[21]:=temp_cd[3]*d[5];
  temp_cdd[22]:=temp_cd[3]*d[6];
  temp_cdd[23]:=temp_cd[3]*d[7];
  temp_cdd[24]:=temp_cd[3]*d[8];
  temp_cdd[25]:=temp_cd[4]*d[1];
  temp_cdd[26]:=temp_cd[4]*d[2];
  temp_cdd[27]:=temp_cd[4]*d[3];
  temp_cdd[28]:=temp_cd[4]*d[4];
  temp_cdd[29]:=temp_cd[4]*d[5];
  temp_cdd[30]:=temp_cd[4]*d[6];
  temp_cdd[31]:=temp_cd[4]*d[7];
  temp_cdd[32]:=temp_cd[4]*d[8];
  temp_cdd[33]:=temp_cd[5]*d[1];
  temp_cdd[34]:=temp_cd[5]*d[2];
  temp_cdd[35]:=temp_cd[5]*d[3];
  temp_cdd[36]:=temp_cd[5]*d[4];
  temp_cdd[37]:=temp_cd[5]*d[5];
  temp_cdd[38]:=temp_cd[5]*d[6];
  temp_cdd[39]:=temp_cd[5]*d[7];
  temp_cdd[40]:=temp_cd[5]*d[8];
  temp_cdd[41]:=temp_cd[6]*d[1];
  temp_cdd[42]:=temp_cd[6]*d[2];
  temp_cdd[43]:=temp_cd[6]*d[3];
  temp_cdd[44]:=temp_cd[6]*d[4];
  temp_cdd[45]:=temp_cd[6]*d[5];
  temp_cdd[46]:=temp_cd[6]*d[6];
  temp_cdd[47]:=temp_cd[6]*d[7];
  temp_cdd[48]:=temp_cd[6]*d[8];
  temp_cdd[49]:=temp_cd[7]*d[1];
  temp_cdd[50]:=temp_cd[7]*d[2];
  temp_cdd[51]:=temp_cd[7]*d[3];
  temp_cdd[52]:=temp_cd[7]*d[4];
  temp_cdd[53]:=temp_cd[7]*d[5];
  temp_cdd[54]:=temp_cd[7]*d[6];
  temp_cdd[55]:=temp_cd[7]*d[7];
  temp_cdd[56]:=temp_cd[7]*d[8];
  temp_cdd[57]:=temp_cd[8]*d[1];
  temp_cdd[58]:=temp_cd[8]*d[2];
  temp_cdd[59]:=temp_cd[8]*d[3];
  temp_cdd[60]:=temp_cd[8]*d[4];
  temp_cdd[61]:=temp_cd[8]*d[5];
  temp_cdd[62]:=temp_cd[8]*d[6];
  temp_cdd[63]:=temp_cd[8]*d[7];
  temp_cdd[64]:=temp_cd[8]*d[8];
sem_c[1]:=((temp_cdd[1]*c[1])+(temp_cdd[
2]*c[9])+(temp_cdd[3]*c[17])+(temp_cdd[4
]*c[25])+(temp_cdd[5]*c[33])+(temp_cdd[6
]*c[41])+(temp_cdd[7]*c[49])+(temp_cdd[8
]*c[57]))/(1+eps);
sem_c[2]:=((temp_cdd[1]*c[2])+(temp_cdd[
2]*c[10])+(temp_cdd[3]*c[18])+(temp_cdd[
4]*c[26])+(temp_cdd[5]*c[34])+(temp_cdd[
6]*c[42])+(temp_cdd[7]*c[50])+(temp_cdd[
8]*c[58]))/(1+eps);
sem_c[3]:=((temp_cdd[1]*c[3])+(temp_cdd[
2]*c[11])+(temp_cdd[3]*c[19])+(temp_cdd[
4]*c[27])+(temp_cdd[5]*c[35])+(temp_cdd[
6]*c[43])+(temp_cdd[7]*c[51])+(temp_cdd[
8]*c[59]))/(1+eps);
sem_c[4]:=((temp_cdd[1]*c[4])+(temp_cdd[
2]*c[12])+(temp_cdd[3]*c[20])+(temp_cdd[
4]*c[28])+(temp_cdd[5]*c[36])+(temp_cdd[
```

```
6]*c[44])+(temp_cdd[7]*c[52])+(temp_cdd[
8]*c[60]))/(1+eps);
sem_c[5]:=((temp_cdd[1]*c[5])+(temp_cdd[
2]*c[13])+(temp_cdd[3]*c[21])+(temp_cdd[
4]*c[29])+(temp_cdd[5]*c[37])+(temp_cdd[
6]*c[45])+(temp_cdd[7]*c[53])+(temp_cdd[
8]*c[61]))/(1+eps);
sem_c[6]:=((temp_cdd[1]*c[6])+(temp_cdd[
2]*c[14])+(temp_cdd[3]*c[22])+(temp_cdd[
4]*c[30])+(temp_cdd[5]*c[38])+(temp_cdd[
6]*c[46])+(temp_cdd[7]*c[54])+(temp_cdd[
8]*c[62]))/(1+eps);
sem_c[7]:=((temp_cdd[1]*c[7])+(temp_cdd[
2]*c[15])+(temp_cdd[3]*c[23])+(temp_cdd[
4]*c[31])+(temp_cdd[5]*c[39])+(temp_cdd[
6]*c[47])+(temp_cdd[7]*c[55])+(temp_cdd[
8]*c[63]))/(1+eps);
sem_c[8]:=((temp_cdd[1]*c[8])+(temp_cdd[
2]*c[16])+(temp_cdd[3]*c[24])+(temp_cdd[
4]*c[32])+(temp_cdd[5]*c[40])+(temp_cdd[
6]*c[48])+(temp_cdd[7]*c[56])+(temp_cdd[
8]*c[64]))/(1+eps);
sem_c[9]:=((temp_cdd[9]*c[1])+(temp_cdd[
10]*c[9])+(temp_cdd[11]*c[17])+(temp_cd
d[12]*c[25])+(temp_cdd[13]*c[33])+(temp_
cdd[14]*c[41])+(temp_cdd[15]*c[49])+(tem
p_cdd[16]*c[57]))/(1+eps);
sem_c[10]:=((temp_cdd[9]*c[2])+(temp_cd
d[10]*c[10])+(temp_cdd[11]*c[18])+(temp_
cdd[12]*c[26])+(temp_cdd[13]*c[34])+(tem
p_cdd[14]*c[42])+(temp_cdd[15]*c[50])+(t
emp_cdd[16]*c[58]))/(1+eps);
sem_c[11]:=((temp_cdd[9]*c[3])+(temp_cd
d[10]*c[11])+(temp_cdd[11]*c[19])+(temp_
cdd[12]*c[27])+(temp_cdd[13]*c[35])+(tem
p_cdd[14]*c[43])+(temp_cdd[15]*c[51])+(t
emp_cdd[16]*c[59]))/(1+eps);
sem_c[12]:=((temp_cdd[9]*c[4])+(temp_cd
d[10]*c[12])+(temp_cdd[11]*c[20])+(temp_
cdd[12]*c[28])+(temp_cdd[13]*c[36])+(tem
p_cdd[14]*c[44])+(temp_cdd[15]*c[52])+(t
emp_cdd[16]*c[60]))/(1+eps);
sem_c[13]:=((temp_cdd[9]*c[5])+(temp_cd
d[10]*c[13])+(temp_cdd[11]*c[21])+(temp_
cdd[12]*c[29])+(temp_cdd[13]*c[37])+(tem
p_cdd[14]*c[45])+(temp_cdd[15]*c[53])+(t
emp_cdd[16]*c[61]))/(1+eps);
sem_c[14]:=((temp_cdd[9]*c[6])+(temp_cd
d[10]*c[14])+(temp_cdd[11]*c[22])+(temp_
cdd[12]*c[30])+(temp_cdd[13]*c[38])+(tem
p_cdd[14]*c[46])+(temp_cdd[15]*c[54])+(t
emp_cdd[16]*c[62]))/(1+eps);
sem_c[15]:=((temp_cdd[9]*c[7])+(temp_cd
d[10]*c[15])+(temp_cdd[11]*c[23])+(temp_
cdd[12]*c[31])+(temp_cdd[13]*c[39])+(tem
p_cdd[14]*c[47])+(temp_cdd[15]*c[55])+(t
emp_cdd[16]*c[63]))/(1+eps);
sem_c[16]:=((temp_cdd[9]*c[8])+(temp_cd
d[10]*c[16])+(temp_cdd[11]*c[24])+(temp_
cdd[12]*c[32])+(temp_cdd[13]*c[40])+(tem
p_cdd[14]*c[48])+(temp_cdd[15]*c[56])+(t
emp_cdd[16]*c[64]))/(1+eps);
sem_c[17]:=((temp_cdd[17]*c[1])+(temp_c
dd[18]*c[9])+(temp_cdd[19]*c[17])+(temp_
cdd[20]*c[25])+(temp_cdd[21]*c[33])+(tem
p_cdd[22]*c[41])+(temp_cdd[23]*c[49])+(t
emp_cdd[24]*c[57]))/(1+eps);

sem_c[18]:=((temp_cdd[17]*c[2])+(temp_c
dd[18]*c[10])+(temp_cdd[19]*c[18])+(temp
_cdd[20]*c[26])+(temp_cdd[21]*c[34])+(te
mp_cdd[22]*c[42])+(temp_cdd[23]*c[50])+
(temp_cdd[24]*c[58]))/(1+eps);
sem_c[19]:=((temp_cdd[17]*c[3])+(temp_c
dd[18]*c[11])+(temp_cdd[19]*c[19])+(temp
_cdd[20]*c[27])+(temp_cdd[21]*c[35])+(te
mp_cdd[22]*c[43])+(temp_cdd[23]*c[51])+
(temp_cdd[24]*c[59]))/(1+eps);
sem_c[20]:=((temp_cdd[17]*c[4])+(temp_c
dd[18]*c[12])+(temp_cdd[19]*c[20])+(temp
_cdd[20]*c[28])+(temp_cdd[21]*c[36])+(te
mp_cdd[22]*c[44])+(temp_cdd[23]*c[52])+
(temp_cdd[24]*c[60]))/(1+eps);
sem_c[21]:=((temp_cdd[17]*c[5])+(temp_c
dd[18]*c[13])+(temp_cdd[19]*c[21])+(temp
_cdd[20]*c[29])+(temp_cdd[21]*c[37])+(te
mp_cdd[22]*c[45])+(temp_cdd[23]*c[53])+
(temp_cdd[24]*c[61]))/(1+eps);
sem_c[22]:=((temp_cdd[17]*c[6])+(temp_c
dd[18]*c[14])+(temp_cdd[19]*c[22])+(temp
_cdd[20]*c[30])+(temp_cdd[21]*c[38])+(te
mp_cdd[22]*c[46])+(temp_cdd[23]*c[54])+
(temp_cdd[24]*c[62]))/(1+eps);
sem_c[23]:=((temp_cdd[17]*c[7])+(temp_c
dd[18]*c[15])+(temp_cdd[19]*c[23])+(temp
_cdd[20]*c[31])+(temp_cdd[21]*c[39])+(te
mp_cdd[22]*c[47])+(temp_cdd[23]*c[55])+
(temp_cdd[24]*c[63]))/(1+eps);
sem_c[24]:=((temp_cdd[17]*c[8])+(temp_c
dd[18]*c[16])+(temp_cdd[19]*c[24])+(temp
_cdd[20]*c[32])+(temp_cdd[21]*c[40])+(te
mp_cdd[22]*c[48])+(temp_cdd[23]*c[56])+
(temp_cdd[24]*c[64]))/(1+eps);
sem_c[25]:=((temp_cdd[25]*c[1])+(temp_c
dd[26]*c[9])+(temp_cdd[27]*c[17])+(temp_
cdd[28]*c[25])+(temp_cdd[29]*c[33])+(tem
p_cdd[30]*c[41])+(temp_cdd[31]*c[49])+(t
emp_cdd[32]*c[57]))/(1+eps);
sem_c[26]:=((temp_cdd[25]*c[2])+(temp_c
dd[26]*c[10])+(temp_cdd[27]*c[18])+(temp
_cdd[28]*c[26])+(temp_cdd[29]*c[34])+(te
mp_cdd[30]*c[42])+(temp_cdd[31]*c[50])+
(temp_cdd[32]*c[58]))/(1+eps);
sem_c[27]:=((temp_cdd[25]*c[3])+(temp_c
dd[26]*c[11])+(temp_cdd[27]*c[19])+(temp
_cdd[28]*c[27])+(temp_cdd[29]*c[35])+(te
```

mp_cdd[30]*c[43])+(temp_cdd[31]*c[51])+(temp_cdd[32]*c[59]))/(1+eps);
sem_c[28]:=((temp_cdd[25]*c[4])+(temp_cdd[26]*c[12])+(temp_cdd[27]*c[20])+(temp_cdd[28]*c[28])+(temp_cdd[29]*c[36])+(temp_cdd[30]*c[44])+(temp_cdd[31]*c[52])+(temp_cdd[32]*c[60]))/(1+eps);
sem_c[29]:=((temp_cdd[25]*c[5])+(temp_cdd[26]*c[13])+(temp_cdd[27]*c[21])+(temp_cdd[28]*c[29])+(temp_cdd[29]*c[37])+(temp_cdd[30]*c[45])+(temp_cdd[31]*c[53])+(temp_cdd[32]*c[61]))/(1+eps);
sem_c[30]:=((temp_cdd[25]*c[6])+(temp_cdd[26]*c[14])+(temp_cdd[27]*c[22])+(temp_cdd[28]*c[30])+(temp_cdd[29]*c[38])+(temp_cdd[30]*c[46])+(temp_cdd[31]*c[54])+(temp_cdd[32]*c[62]))/(1+eps);
sem_c[31]:=((temp_cdd[25]*c[7])+(temp_cdd[26]*c[15])+(temp_cdd[27]*c[23])+(temp_cdd[28]*c[31])+(temp_cdd[29]*c[39])+(temp_cdd[30]*c[47])+(temp_cdd[31]*c[55])+(temp_cdd[32]*c[63]))/(1+eps);
sem_c[32]:=((temp_cdd[25]*c[8])+(temp_cdd[26]*c[16])+(temp_cdd[27]*c[24])+(temp_cdd[28]*c[32])+(temp_cdd[29]*c[40])+(temp_cdd[30]*c[48])+(temp_cdd[31]*c[56])+(temp_cdd[32]*c[64]))/(1+eps);
sem_c[33]:=((temp_cdd[33]*c[1])+(temp_cdd[34]*c[9])+(temp_cdd[35]*c[17])+(temp_cdd[36]*c[25])+(temp_cdd[37]*c[33])+(temp_cdd[38]*c[41])+(temp_cdd[39]*c[49])+(temp_cdd[40]*c[57]))/(1+eps);
sem_c[34]:=((temp_cdd[33]*c[2])+(temp_cdd[34]*c[10])+(temp_cdd[35]*c[18])+(temp_cdd[36]*c[26])+(temp_cdd[37]*c[34])+(temp_cdd[38]*c[42])+(temp_cdd[39]*c[50])+(temp_cdd[40]*c[58]))/(1+eps);
sem_c[35]:=((temp_cdd[33]*c[3])+(temp_cdd[34]*c[11])+(temp_cdd[35]*c[19])+(temp_cdd[36]*c[27])+(temp_cdd[37]*c[35])+(temp_cdd[38]*c[43])+(temp_cdd[39]*c[51])+(temp_cdd[40]*c[59]))/(1+eps);
sem_c[36]:=((temp_cdd[33]*c[4])+(temp_cdd[34]*c[12])+(temp_cdd[35]*c[20])+(temp_cdd[36]*c[28])+(temp_cdd[37]*c[36])+(temp_cdd[38]*c[44])+(temp_cdd[39]*c[52])+(temp_cdd[40]*c[60]))/(1+eps);
sem_c[37]:=((temp_cdd[33]*c[5])+(temp_cdd[34]*c[13])+(temp_cdd[35]*c[21])+(temp_cdd[36]*c[29])+(temp_cdd[37]*c[37])+(temp_cdd[38]*c[45])+(temp_cdd[39]*c[53])+(temp_cdd[40]*c[61]))/(1+eps);
sem_c[38]:=((temp_cdd[33]*c[6])+(temp_cdd[34]*c[14])+(temp_cdd[35]*c[22])+(temp_cdd[36]*c[30])+(temp_cdd[37]*c[38])+(temp_cdd[38]*c[46])+(temp_cdd[39]*c[54])+(temp_cdd[40]*c[62]))/(1+eps);
sem_c[39]:=((temp_cdd[33]*c[7])+(temp_cdd[34]*c[15])+(temp_cdd[35]*c[23])+(temp_cdd[36]*c[31])+(temp_cdd[37]*c[39])+(temp_cdd[38]*c[47])+(temp_cdd[39]*c[55])+(temp_cdd[40]*c[63]))/(1+eps);
sem_c[40]:=((temp_cdd[33]*c[8])+(temp_cdd[34]*c[16])+(temp_cdd[35]*c[24])+(temp_cdd[36]*c[32])+(temp_cdd[37]*c[40])+(temp_cdd[38]*c[48])+(temp_cdd[39]*c[56])+(temp_cdd[40]*c[64]))/(1+eps);
sem_c[41]:=((temp_cdd[41]*c[1])+(temp_cdd[42]*c[9])+(temp_cdd[43]*c[17])+(temp_cdd[44]*c[25])+(temp_cdd[45]*c[33])+(temp_cdd[46]*c[41])+(temp_cdd[47]*c[49])+(temp_cdd[48]*c[57]))/(1+eps);
sem_c[42]:=((temp_cdd[41]*c[2])+(temp_cdd[42]*c[10])+(temp_cdd[43]*c[18])+(temp_cdd[44]*c[26])+(temp_cdd[45]*c[34])+(temp_cdd[46]*c[42])+(temp_cdd[47]*c[50])+(temp_cdd[48]*c[58]))/(1+eps);

sem_c[43]:=((temp_cdd[41]*c[3])+(temp_cdd[42]*c[11])+(temp_cdd[43]*c[19])+(temp_cdd[44]*c[27])+(temp_cdd[45]*c[35])+(temp_cdd[46]*c[43])+(temp_cdd[47]*c[51])+(temp_cdd[48]*c[59]))/(1+eps);
sem_c[44]:=((temp_cdd[41]*c[4])+(temp_cdd[42]*c[12])+(temp_cdd[43]*c[20])+(temp_cdd[44]*c[28])+(temp_cdd[45]*c[36])+(temp_cdd[46]*c[44])+(temp_cdd[47]*c[52])+(temp_cdd[48]*c[60]))/(1+eps);
sem_c[45]:=((temp_cdd[41]*c[5])+(temp_cdd[42]*c[13])+(temp_cdd[43]*c[21])+(temp_cdd[44]*c[29])+(temp_cdd[45]*c[37])+(temp_cdd[46]*c[45])+(temp_cdd[47]*c[53])+(temp_cdd[48]*c[61]))/(1+eps);
sem_c[46]:=((temp_cdd[41]*c[6])+(temp_cdd[42]*c[14])+(temp_cdd[43]*c[22])+(temp_cdd[44]*c[30])+(temp_cdd[45]*c[38])+(temp_cdd[46]*c[46])+(temp_cdd[47]*c[54])+(temp_cdd[48]*c[62]))/(1+eps);
sem_c[47]:=((temp_cdd[41]*c[7])+(temp_cdd[42]*c[15])+(temp_cdd[43]*c[23])+(temp_cdd[44]*c[31])+(temp_cdd[45]*c[39])+(temp_cdd[46]*c[47])+(temp_cdd[47]*c[55])+(temp_cdd[48]*c[63]))/(1+eps);
sem_c[48]:=((temp_cdd[41]*c[8])+(temp_cdd[42]*c[16])+(temp_cdd[43]*c[24])+(temp_cdd[44]*c[32])+(temp_cdd[45]*c[40])+(temp_cdd[46]*c[48])+(temp_cdd[47]*c[56])+(temp_cdd[48]*c[64]))/(1+eps);
sem_c[49]:=((temp_cdd[49]*c[1])+(temp_cdd[50]*c[9])+(temp_cdd[51]*c[17])+(temp_cdd[52]*c[25])+(temp_cdd[53]*c[33])+(temp_cdd[54]*c[41])+(temp_cdd[55]*c[49])+(temp_cdd[56]*c[57]))/(1+eps);
sem_c[50]:=((temp_cdd[49]*c[2])+(temp_cdd[50]*c[10])+(temp_cdd[51]*c[18])+(temp_cdd[52]*c[26])+(temp_cdd[53]*c[34])+(te

```
mp_cdd[54]*c[42])+(temp_cdd[55]*c[50])+
(temp_cdd[56]*c[58]))/(1+eps);
sem_c[51]:=((temp_cdd[49]*c[3])+(temp_c
dd[50]*c[11])+(temp_cdd[51]*c[19])+(temp
_cdd[52]*c[27])+(temp_cdd[53]*c[35])+(te
mp_cdd[54]*c[43])+(temp_cdd[55]*c[51])+
(temp_cdd[56]*c[59]))/(1+eps);
sem_c[52]:=((temp_cdd[49]*c[4])+(temp_c
dd[50]*c[12])+(temp_cdd[51]*c[20])+(temp
_cdd[52]*c[28])+(temp_cdd[53]*c[36])+(te
mp_cdd[54]*c[44])+(temp_cdd[55]*c[52])+
(temp_cdd[56]*c[60]))/(1+eps);
sem_c[53]:=((temp_cdd[49]*c[5])+(temp_c
dd[50]*c[13])+(temp_cdd[51]*c[21])+(temp
_cdd[52]*c[29])+(temp_cdd[53]*c[37])+(te
mp_cdd[54]*c[45])+(temp_cdd[55]*c[53])+
(temp_cdd[56]*c[61]))/(1+eps);
sem_c[54]:=((temp_cdd[49]*c[6])+(temp_c
dd[50]*c[14])+(temp_cdd[51]*c[22])+(temp
_cdd[52]*c[30])+(temp_cdd[53]*c[38])+(te
mp_cdd[54]*c[46])+(temp_cdd[55]*c[54])+
(temp_cdd[56]*c[62]))/(1+eps);
sem_c[55]:=((temp_cdd[49]*c[7])+(temp_c
dd[50]*c[15])+(temp_cdd[51]*c[23])+(temp
_cdd[52]*c[31])+(temp_cdd[53]*c[39])+(te
mp_cdd[54]*c[47])+(temp_cdd[55]*c[55])+
(temp_cdd[56]*c[63]))/(1+eps);
sem_c[56]:=((temp_cdd[49]*c[8])+(temp_c
dd[50]*c[16])+(temp_cdd[51]*c[24])+(temp
_cdd[52]*c[32])+(temp_cdd[53]*c[40])+(te
mp_cdd[54]*c[48])+(temp_cdd[55]*c[56])+
(temp_cdd[56]*c[64]))/(1+eps);
sem_c[57]:=((temp_cdd[57]*c[1])+(temp_c
dd[58]*c[9])+(temp_cdd[59]*c[17])+(temp_
cdd[60]*c[25])+(temp_cdd[61]*c[33])+(tem
p_cdd[62]*c[41])+(temp_cdd[63]*c[49])+(t
emp_cdd[64]*c[57]))/(1+eps);
sem_c[58]:=((temp_cdd[57]*c[2])+(temp_c
dd[58]*c[10])+(temp_cdd[59]*c[18])+(temp
_cdd[60]*c[26])+(temp_cdd[61]*c[34])+(te
mp_cdd[62]*c[42])+(temp_cdd[63]*c[50])+
(temp_cdd[64]*c[58]))/(1+eps);
sem_c[59]:=((temp_cdd[57]*c[3])+(temp_c
dd[58]*c[11])+(temp_cdd[59]*c[19])+(temp
_cdd[60]*c[27])+(temp_cdd[61]*c[35])+(te
mp_cdd[62]*c[43])+(temp_cdd[63]*c[51])+
(temp_cdd[64]*c[59]))/(1+eps);
sem_c[60]:=((temp_cdd[57]*c[4])+(temp_c
dd[58]*c[12])+(temp_cdd[59]*c[20])+(temp
_cdd[60]*c[28])+(temp_cdd[61]*c[36])+(te
mp_cdd[62]*c[44])+(temp_cdd[63]*c[52])+
(temp_cdd[64]*c[60]))/(1+eps);
sem_c[61]:=((temp_cdd[57]*c[5])+(temp_c
dd[58]*c[13])+(temp_cdd[59]*c[21])+(temp
_cdd[60]*c[29])+(temp_cdd[61]*c[37])+(te
mp_cdd[62]*c[45])+(temp_cdd[63]*c[53])+
(temp_cdd[64]*c[61]))/(1+eps);
sem_c[62]:=((temp_cdd[57]*c[6])+(temp_c
dd[58]*c[14])+(temp_cdd[59]*c[22])+(temp
_cdd[60]*c[30])+(temp_cdd[61]*c[38])+(te
mp_cdd[62]*c[46])+(temp_cdd[63]*c[54])+
(temp_cdd[64]*c[62]))/(1+eps);
sem_c[63]:=((temp_cdd[57]*c[7])+(temp_c
dd[58]*c[15])+(temp_cdd[59]*c[23])+(temp
_cdd[60]*c[31])+(temp_cdd[61]*c[39])+(te
mp_cdd[62]*c[47])+(temp_cdd[63]*c[55])+
(temp_cdd[64]*c[63]))/(1+eps);
sem_c[64]:=((temp_cdd[57]*c[8])+(temp_c
dd[58]*c[16])+(temp_cdd[59]*c[24])+(temp
_cdd[60]*c[32])+(temp_cdd[61]*c[40])+(te
mp_cdd[62]*c[48])+(temp_cdd[63]*c[56])+
(temp_cdd[64]*c[64]))/(1+eps);
    c[1]:=c[1]-sem_c[1];
    c[2]:=c[2]-sem_c[2];
    c[3]:=c[3]-sem_c[3];
    c[4]:=c[4]-sem_c[4];
    c[5]:=c[5]-sem_c[5];
    c[6]:=c[6]-sem_c[6];
    c[7]:=c[7]-sem_c[7];
    c[8]:=c[8]-sem_c[8];
    c[9]:=c[9]-sem_c[9];
    c[10]:=c[10]-sem_c[10];
    c[11]:=c[11]-sem_c[11];
    c[12]:=c[12]-sem_c[12];
    c[13]:=c[13]-sem_c[13];
    c[14]:=c[14]-sem_c[14];
    c[15]:=c[15]-sem_c[15];
    c[16]:=c[16]-sem_c[16];
    c[17]:=c[17]-sem_c[17];
    c[18]:=c[18]-sem_c[18];
    c[19]:=c[19]-sem_c[19];
    c[20]:=c[20]-sem_c[20];
    c[21]:=c[21]-sem_c[21];
    c[22]:=c[22]-sem_c[22];
    c[23]:=c[23]-sem_c[23];
    c[24]:=c[24]-sem_c[24];
    c[25]:=c[25]-sem_c[25];
    c[26]:=c[26]-sem_c[26];
    c[27]:=c[27]-sem_c[27];
    c[28]:=c[28]-sem_c[28];
    c[29]:=c[29]-sem_c[29];
    c[30]:=c[30]-sem_c[30];
    c[31]:=c[31]-sem_c[31];
    c[32]:=c[32]-sem_c[32];
    c[33]:=c[33]-sem_c[33];
    c[34]:=c[34]-sem_c[34];
    c[35]:=c[35]-sem_c[35];
    c[36]:=c[36]-sem_c[36];
    c[37]:=c[37]-sem_c[37];
    c[38]:=c[38]-sem_c[38];
    c[39]:=c[39]-sem_c[39];
    c[40]:=c[40]-sem_c[40];
    c[41]:=c[41]-sem_c[41];
    c[42]:=c[42]-sem_c[42];
    c[43]:=c[43]-sem_c[43];
    c[44]:=c[44]-sem_c[44];
```

```
c[45]:=c[45]-sem_c[45];
c[46]:=c[46]-sem_c[46];
c[47]:=c[47]-sem_c[47];
c[48]:=c[48]-sem_c[48];
c[49]:=c[49]-sem_c[49];
c[50]:=c[50]-sem_c[50];
c[51]:=c[51]-sem_c[51];
c[52]:=c[52]-sem_c[52];
c[53]:=c[53]-sem_c[53];
c[54]:=c[54]-sem_c[54];
c[55]:=c[55]-sem_c[55];
c[56]:=c[56]-sem_c[56];
c[57]:=c[57]-sem_c[57];
c[58]:=c[58]-sem_c[58];
c[59]:=c[59]-sem_c[59];
c[60]:=c[60]-sem_c[60];
c[61]:=c[61]-sem_c[61];
c[62]:=c[62]-sem_c[62];
c[63]:=c[63]-sem_c[63];
c[64]:=c[64]-sem_c[64];

Memo12.Lines.Append('+FloatToStr(c[1]));

Memo13.Lines.Append('+FloatToStr(c[2]));

Memo14.Lines.Append('+FloatToStr(c[3]));

Memo15.Lines.Append('+FloatToStr(c[4]));

Memo16.Lines.Append('+FloatToStr(c[5]));

Memo17.Lines.Append('+FloatToStr(c[6]));

Memo18.Lines.Append('+FloatToStr(c[7]));

Memo19.Lines.Append('+FloatToStr(c[8]));

Memo20.Lines.Append('+FloatToStr(c[9]));

Memo21.Lines.Append('+FloatToStr(c[10])
);

Memo22.Lines.Append('+FloatToStr(c[11])
);

Memo23.Lines.Append('+FloatToStr(c[12])
);

Memo24.Lines.Append('+FloatToStr(c[13])
);

Memo25.Lines.Append('+FloatToStr(c[14])
);

Memo26.Lines.Append('+FloatToStr(c[15])
);

Memo27.Lines.Append('+FloatToStr(c[16])
);

Memo28.Lines.Append('+FloatToStr(c[17])
);

Memo29.Lines.Append('+FloatToStr(c[18])
);

Memo30.Lines.Append('+FloatToStr(c[19])
);

Memo31.Lines.Append('+FloatToStr(c[20])
);

Memo32.Lines.Append('+FloatToStr(c[21])
);

Memo33.Lines.Append('+FloatToStr(c[22])
);

Memo34.Lines.Append('+FloatToStr(c[23])
);

Memo35.Lines.Append('+FloatToStr(c[24])
);

Memo36.Lines.Append('+FloatToStr(c[25])
);

Memo37.Lines.Append('+FloatToStr(c[26])
);

Memo38.Lines.Append('+FloatToStr(c[27])
);

Memo39.Lines.Append('+FloatToStr(c[28])
);

Memo40.Lines.Append('+FloatToStr(c[29])
);

Memo41.Lines.Append('+FloatToStr(c[30])
);

Memo42.Lines.Append('+FloatToStr(c[31])
);

Memo43.Lines.Append('+FloatToStr(c[32])
);

Memo44.Lines.Append('+FloatToStr(c[33])
);

Memo45.Lines.Append('+FloatToStr(c[34])
);
```

```
Memo46.Lines.Append('+FloatToStr(c[35])
);

Memo47.Lines.Append('+FloatToStr(c[36])
);

Memo48.Lines.Append('+FloatToStr(c[37])
);

Memo49.Lines.Append('+FloatToStr(c[38])
);

Memo50.Lines.Append('+FloatToStr(c[39])
);

Memo51.Lines.Append('+FloatToStr(c[40])
);

Memo52.Lines.Append('+FloatToStr(c[41])
);

Memo53.Lines.Append('+FloatToStr(c[42])
);

Memo54.Lines.Append('+FloatToStr(c[43])
);

Memo55.Lines.Append('+FloatToStr(c[44])
);

Memo56.Lines.Append('+FloatToStr(c[45])
);

Memo57.Lines.Append('+FloatToStr(c[46])
);

Memo58.Lines.Append('+FloatToStr(c[47])
);

Memo59.Lines.Append('+FloatToStr(c[48])
);

Memo60.Lines.Append('+FloatToStr(c[49])
);

Memo61.Lines.Append('+FloatToStr(c[50])
);

Memo62.Lines.Append('+FloatToStr(c[51])
);

Memo63.Lines.Append('+FloatToStr(c[52])
);

Memo64.Lines.Append('+FloatToStr(c[53])
);

Memo65.Lines.Append('+FloatToStr(c[54])
);

Memo66.Lines.Append('+FloatToStr(c[55])
);

Memo67.Lines.Append('+FloatToStr(c[56])
);

Memo68.Lines.Append('+FloatToStr(c[57])
);

Memo69.Lines.Append('+FloatToStr(c[58])
);

Memo70.Lines.Append('+FloatToStr(c[59])
);

Memo71.Lines.Append('+FloatToStr(c[60])
);

Memo72.Lines.Append('+FloatToStr(c[61])
);

Memo73.Lines.Append('+FloatToStr(c[62])
);

Memo74.Lines.Append('+FloatToStr(c[63])
);

Memo75.Lines.Append('+FloatToStr(c[64])
);
    edit93.Text:=floattostr(ep);

Memo77.Lines.Append('+FloatToStr(ep));

Memo76.Lines.Append('+FloatToStr(a1)+'
'+FloatToStr(a2)+' '+FloatToStr(a3)+'
'+FloatToStr(a4));

Memo78.Lines.Append('+FloatToStr(b1)+'
'+FloatToStr(b2)+' '+FloatToStr(b3)+'
'+FloatToStr(b4));
    With Series3 do
      Begin
      Add(  +a1) ;
      end;
    With Series4 do
      Begin
      Add(  +a2) ;
      end;
    With Series5 do
      Begin
      Add(  +a3) ;
      end;
    With Series6 do
      Begin
```

```
        Add(  +a4) ;
        end;
       With Series7 do
        Begin
        Add(  +b1) ;
        end;
       With Series8 do
        Begin
        Add(  +b2) ;
        end;
       With Series9 do
        Begin
        Add(  +b3) ;
        end;
       With Series10 do
        Begin
        Add(  +b4) ;
        end;
       With Series11 do
        Begin
        Add(  +ep) ;
        end;
      j:=j+1;
      end;
      edit2.Text:=inttostr(l);
      stringgrid1.RowCount:=l+1;
      stringgrid1.Cells[0,l]:=inttostr(l);
      stringgrid1.Cells[1,l]:='
'+FloatToStr(sinyal_uji);
      stringgrid1.Cells[2,l]:='
'+FloatToStr(data_real);
       With Series1 do
        Begin
        Add(  +sinyal_uji) ;
        end;
       With Series2 do
        Begin
        Add(  +data_real) ;
        end;

      Buffer:= '';
      waktu_det:=10*l;
      Edit102.Text:=floatToStr(waktu_det);
      waktu_mnt:=waktu_det/60;
      Edit103.Text:=floatToStr(waktu_mnt);
      waktu_jam:=waktu_mnt/60;
      Edit104.Text:=floatToStr(waktu_jam);
      l:=l+1;
      end;
end;
procedure TForm1.Button1Click(Sender:
TObject);
begin
if (connect=false) then begin
   comport1.Open;
   comport1.WriteStr('P');
   connect:=true;
   Button1.Caption:= 'Disconnect';
  end
  else begin
   comport1.Close;
   connect:=false;
   Button1.Caption:= 'Sinyal Uji PRBS';
  end

end;
procedure TForm1.FormCreate(Sender:
TObject);
begin
SaveDialog1.Execute;
  nmfile:=SaveDialog1.FileName+'.txt';

label1.Caption:=SaveDialog1.FileName+'.tx
t';
  stringgrid1.ColWidths[0]:=25;
  stringgrid1.ColWidths[1]:=70;
  stringgrid1.ColWidths[2]:=110;
  stringgrid1.Cells[0,0]:='No';
  stringgrid1.Cells[1,0]:='Input (Volt)';
  stringgrid1.Cells[2,0]:='Output (Volt)';
  form1.Show;
end;

procedure TForm1.Button2Click(Sender:
TObject);
begin
  PrintDialog1.Execute;
  chart1.PrintLandscape;
end;
procedure TForm1.Button5Click(Sender:
TObject);
begin
  comport1.ShowSetupDialog;
end;

procedure TForm1.Button3Click(Sender:
TObject);
begin
  SaveDialog2.Execute;
nmfile:=SaveDialog2.FileName+'.txt';
  AssignFile(F,nmfile);
  Rewrite(F);
  Write(F,Memo76.Text);
  Closefile(F);
end;
procedure TForm1.Button6Click(Sender:
TObject);
begin
  SaveDialog4.Execute;
  nmfile:=SaveDialog4.FileName+'.txt';
  AssignFile(F,nmfile);
  Rewrite(F);
  Write(F,Memo77.Text);
  Closefile(F);
end;
```

```
procedure TForm1.Button4Click(Sender:
TObject);
begin
 SaveDialog3.Execute;
 nmfile:=SaveDialog3.FileName+'.txt';
 AssignFile(F,nmfile);
 Rewrite(F);
 Write(F,Memo78.Text);
 Closefile(F);
end;
procedure TForm1.Button8Click(Sender:
TObject);
var
 aa,a,x,nul,cc: integer;
 yg,inp,gbo:extended;
 aaa:string;
begin
 x:=strtoint(edit108.text);
 nul:=0;
 for cc:=0 to 3 do
 begin
  memo81.Lines.Append('+floattostr(nul));
 end;
 for aa:=0 to 3 do
 begin
  aaa:=memo80.lines[aa];
  memo83.Lines.Append('+aaa);
 end;
 for a:=4 to x-2 do
 begin
  p1:=strtofloat(edit85.text);
  p2:=strtofloat(edit86.text);
  p3:=strtofloat(edit87.text);
  p4:=strtofloat(edit88.text);
  p5:=strtofloat(edit89.text);
  p6:=strtofloat(edit90.text);
  p7:=strtofloat(edit91.text);
  p8:=strtofloat(edit92.text);
  op1:=strtofloat(memo80.lines[a-1]);
  op2:=strtofloat(memo80.lines[a-2]);
  op3:=strtofloat(memo80.lines[a-3]);
  op4:=strtofloat(memo80.lines[a-4]);
  opp1:=strtofloat(memo83.lines[a-1]);
  opp2:=strtofloat(memo83.lines[a-2]);
  opp3:=strtofloat(memo83.lines[a-3]);
  opp4:=strtofloat(memo83.lines[a-4]);
  ip1:=strtofloat(memo79.lines[a-1]);
  ip2:=strtofloat(memo79.lines[a-2]);
  ip3:=strtofloat(memo79.lines[a-3]);
  ip4:=strtofloat(memo79.lines[a-4]);
  yp:=strtofloat(memo80.Lines[a]);
  yg:=strtofloat(memo80.lines[a-4]);
  inp:=strtofloat(memo79.lines[a-4]);
  om:=-(p1*op1)-(p2*op2)-(p3*op3)-
(p4*op4)+(p5*ip1)+(p6*ip2)+(p7*ip3)+(p8
*ip4);
  om2:=-(p1*opp1)-(p2*opp2)-(p3*opp3)-
(p4*opp4)+(p5*ip1)+(p6*ip2)+(p7*ip3)+(p
8*ip4);
  memo81.lines.append('+FloatToStr(om));

memo83.lines.append('+FloatToStr(om2));
  gbo:=strtofloat(memo83.lines[a-4]);
  With Series12 do
    Begin
    Add(  +yg) ;
    end;
  With Series13 do
    Begin
    Add(  +gbo) ;
    end;
  With Series14 do
    Begin
    Add(  +inp ) ;
    end;
  error:=yp-om;
  error_fit:=yp-om2;
  error_fit_kuad:=error_fit*error_fit;
  error_kuad:=error*error;

jum_err_kuad:=jum_err_kuad+error_kuad;

jum_err_kuad_fit:=jum_err_kuad_fit+error_
fit_kuad;
  jum_y:=jum_y+yp;

memo84.lines.append('+FloatToStr(error));
  error1:=error*strtofloat(memo84.lines[a-
1]);
  error2:=error*strtofloat(memo84.lines[a-
2]);
  error3:=error*strtofloat(memo84.lines[a-
3]);
  error4:=error*strtofloat(memo84.lines[a-
4]);
  jum_e1:=jum_e1+error1;
  jum_e2:=jum_e2+error2;
  jum_e3:=jum_e3+error3;
  jum_e4:=jum_e4+error4;
 end;
end;
procedure TForm1.Button7Click(Sender:
TObject);
var
 ww:extended;
begin
 ww:=strtofloat(edit108.text);
 loss:=jum_err_kuad/ww;
 fpe:=loss*((1+8/ww)/(1-8/ww));
 Edit105.Text:=floatToStr(loss);
 Edit106.Text:=floatToStr(fpe);
end;
procedure TForm1.Button9Click(Sender:
TObject);
```

```
var

z,norm_e,y_peng,sel_kuad,norm_sel,fit,rata:
extended;
 sel,sel_ak: extended;
 q,v,ab,nol: integer;
begin
 nol:=0;
 for ab:=0 to 3 do
 begin
  memo82.Lines.Append("+inttostr(nol));
 end;
 v:=strtoint(edit108.text);
 z:=strtofloat(edit108.text);
 norm_e:=sqrt(jum_err_kuad_fit);
 rata:=jum_y/z;
 for q:=0 to v-2 do
 begin
  y_peng:=strtofloat(memo83.Lines[q]);
  sel:=y_peng-rata;
  sel_kuad:=sel*sel;
  sel_ak:=sel_ak+sel_kuad;
  //Application.Terminate;

memo82.Lines.Append("+FloatToStr(sel_ak
));
 end;
 norm_sel:=sqrt(sel_ak);
 fit:=(1-((norm_e)/(norm_sel)))*100;
 edit107.Text:=floattostr(fit);
end;
procedure TForm1.Button10Click(Sender:
TObject);
 var
  myFile: TextFile;
  namafile: string;
  suji,dalog:extended;
 begin
  opendialog1.Execute;
  // Try to open the Test.txt file for writing
to
  namafile:=opendialog1.FileName;
  AssignFile(myFile, namafile);
  // Reopen the file for reading
  Reset(myFile);
   ii:=1;
  // Display the file contents
 while not Eof(myFile) do
 begin
  //  ReadLn(myFile, text);
    ReadLn(myFile, suji, dalog);

Memo79.Lines.Append("+FloatToStr(suji));

Memo80.Lines.Append("+FloatToStr(dalog)
);
    ii:=ii+1;
    end;
  // Close the file for the last time
  CloseFile(myFile);
  edit108.Text:=inttostr(ii);
end;
{procedure TForm1.Button11Click(Sender:
TObject);
begin
chart5.series[0].clear;
chart5.series[1].clear;
chart5.series[2].clear;
memo82.clear;
memo81.clear;
memo79.clear;
memo80.Clear;
end;}
procedure TForm1.Button11Click(Sender:
TObject);
begin
if (connect=false) then begin
   comport1.Open;
   comport1.WriteStr('S');
   connect:=true;
   Button11.Caption:= 'Disconnect';
  end
  else begin
   comport1.Close;
   connect:=false;
   Button11.Caption:= 'Sinyal Uji Step';
  end
end;
procedure TForm1.Button12Click(Sender:
TObject);
begin
if (connect=false) then begin
   comport1.Open;
   comport1.WriteStr('K');
   connect:=true;
   Button12.Caption:= 'Disconnect';
  end
  else begin
   comport1.Close;
   connect:=false;
   Button12.Caption:= 'Sinyal uji Kotak';
  end
end;
procedure TForm1.Button13Click(Sender:
TObject);
begin
  SaveDialog4.Execute;
  nmfile:=SaveDialog4.FileName+'.txt';
  AssignFile(F,nmfile);
  Rewrite(F);
  Write(F,Memo83.Text);
  Closefile(F);
end;
procedure TForm1.Button14Click(Sender:
TObject);
var
```

```
  r1,r2,r3,r4,rn1,rn2,rn3,rn4: extended;
begin
  r1:=jum_e1/strtofloat(edit108.text);
  r2:=jum_e2/strtofloat(edit108.text);
  r3:=jum_e3/strtofloat(edit108.text);
  r4:=jum_e4/strtofloat(edit108.text);
  rn1:=r1/loss;
  rn2:=r2/loss;
  rn3:=r3/loss;
  rn4:=r4/loss;
  edit109.Text:=floattostr(1);
  edit110.Text:=floattostr(rn1);
  edit111.Text:=floattostr(rn2);
  edit112.Text:=floattostr(rn3);
  edit113.Text:=floattostr(rn4);
end; end.
```

LAMPIRAN D

```
% Basic recursive least squares method
% N - number of identification steps
% c - covariance matrix
% d - regression vector
% theta - vector of the parameter estimates
% ep(k)- prediction error
% eps - auxiliary parameter
% y(k) - process output
% u(k) - controller output
% input variables
u=u1;
% output variables
y=y1;
N=381;
theta= [0 0 0 0 0 0 0 0]'; % initial vector of
parameter estimates
c = 1000*eye(8); % initial covariance matrix
for k=4:1:N
d = [-y(k-1) -y(k-2) -y(k-3) -y(k-4) u(k-1)
u(k-2) u(k-3) u(k-4)]'; % new data vector
f(k)=theta'*d;
ep(k)= y(k)-f(k);
e(k)=ep(k)*ep(k);
epk(k)=ep(k)*ep(k-1);
eps = d'*c*d;
theta = theta + (c*d*ep(k)) / (1+eps); %
theta update
c = c-(c*d*d'*c)/(1+eps); % new covariance
matrix
a1(k)=theta(1);
a2(k)=theta(2);
a3(k)=theta(3);
a4(k)=theta(4)
b1(k)=theta(5);
b2(k)=theta(6);
b3(k)=theta(7);
b4(k)=theta(8);
FIT (k)=[1-norm(ep(k))/norm(y(k)-
mean(y))]*100;
end
figure;
plot(a1,'k');
hold on;
plot(a2,'k');
hold on;
plot(a3,'k');
hold on;
plot(a4,'k');
xlabel('time steps')
ylabel('theta')
figure;
plot(b1,'b');
hold on;
plot(b2,'b');
hold on
plot(b3,'b');
hold on;
plot(b4,'b');
xlabel('time steps')
ylabel('theta')
figure;
plot(ep,'r');
figure;
plot(u);
figure;
plot(y);
figure;
plot(f)
hold on
plot(y)


% whiteness test dengan perlakuan output
pengukuran sebagai masukan model
% input variables
u=u1;
% output variables
y=y1;
N=419;
parameter=theta;
for k=5:1:N
data = [-y(k-1) -y(k-2) -y(k-3) -y(k-4) u(k-1)
u(k-2) u(k-3) u(k-4)]'; % new data vector
out_model(k)=parameter'*data;
error(k)= y(k)-out_model(k);
e_kuad(k)=error(k)^2;
end
rata2e=mean(error);
for k=5:1:N;
  centered_error(k)=rata2e-error(k);
end
for k=5:1:N;
 error_kuad(k)=centered_error(k)^2;
 e1(k)=centered_error(k)*centered_error(k-
1);
 e2(k)=centered_error(k)*centered_error(k-
2);
  e3(k)=centered_error(k)*centered_error(k-
3);
  e4(k)=centered_error(k)*centered_error(k-
4);
end
FPE=(1+4/419)/(1-4/419)*mean(e_kuad)
r0=mean(error_kuad);
r1=mean(e1);
r2=mean(e2);
r3=mean(e3);
r4=mean(e4);
rn1=r1/r0
rn2=r2/r0
rn3=r3/r0
rn4=r4/r0
kriteria=[rn1 rn2 rn3 rn4];
```