UNIVERSITAS INDONESIA

RANCANG BANGUN PROTOTIPE SISTEM AKTUATOR SIRIP ROKET MENGGUNAKAN MOTOR SERVO

SKRIPSI

ARIEL YAGUSANDRI
0906602452

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
DESEMBER 2011

UNIVERSITAS INDONESIA

RANCANG BANGUN PROTOTIPE SISTEM AKTUATOR SIRIP ROKET MENGGUNAKAN MOTOR SERVO

SKRIPSI

Diajukan sebagai salah satu syarat untuk memperoleh gelar Sarjana Teknik

ARIEL YAGUSANDRI
0906602452

FAKULTAS TEKNIK
PROGRAM STUDI TEKNIK ELEKTRO
DEPOK
DESEMBER 2011

HALAMAN PERNYATAAN ORISINALITAS

Skripsi ini adalah hasil karya saya sendiri,
dan semuas sumber yang sikutip maupun dirujuk
telah saya nyatakan dengan benar.

Nama	**: Ariel Yagusandri**
NPM	**: 0906602452**
Tanda Tangan	**: .....**
Tanggal	**: 27 Desember 2011**

HALAMAN PENGESAHAN

Skripsi ini diajukan oleh	:
Nama	: Ariel Yagusandri
NPM	: 0906602452
Program Studi	: Teknik Elektro
Judul Skripsi	: Rancang Bangun Prototipe Sistem Aktuator Sirip Roket Menggunakan Motor Servo

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

Dewan Penguji

Pembimbing : Dr. Abdul Halim M.Eng (.....)

Penguji : Dr. Abdul Muis ST, M.Eng (.....)

Penguji : Ir. Aries Subiantoro M.SEE (.....)

Ditetapkan di : Depok

Tanggal : 20 Januari 2012

KATA PENGANTAR

Segala puji hanya penulis haturkan kepada Tuhan yang telah memberikan karunia-Nya, sehingga penulisan laporan skripsi ini dapat diselesaikan dengan baik. Dalam skripsi ini penulis mencoba memaparkan skripsi yang berjudul "**Rancang Bangun Prototipe Sistem Aktuator Sirip Roket Menggunakan Motor Servo**" yang juga merupakan salah satu syarat untuk memperoleh gelar Sarajana Teknik pada Fakultas Teknik Universitas Indonesia

Adapun penulisan skripsi ini bertujuan untuk pengembangan kemampuan mahasiswa dibidang perancangan dan desain suatu kontrol. Sehingga sangat dibutuhkan bimbingan dari dosen yang menguasai teori dasar untuk perancangan kontrol ini, dimana tanpa bantuan beliau maka perancangan kontrol ini tidak akan dapat diselesaikan dengan baik, dimana dalam skripsi ini sangat dibutuhkan banyak komponen, dan juga teori-teori yang diaplikasikan. Sehingga dalam kesempatan ini, penulis ingin mengucapkan banyak terima kasih yang sebesar-besarnya kepada:

1. Allah Bapa, anak-Nya Yesus Kristus, dan Roh Kudus yang senantiasa selalu memberikan kekuatan, dan hikmat kepada penulis dalam penyelesaian skripsi ini beserta laporannya.
2. Bapak Dr. Abdul Halim, M.Eng selaku pembimbing yang telah banyak memberikan bimbingan dan motivasi dalam pembuatan skripsi ini.
3. Bapak Ir. Gunawan Wibisono, M.Sc, Ph.D selaku pembimbing akademis
4. Orangtua saya yang selalu memberi dukungan dalam proses penyelesaian skripsi ini.
5. Abang dan adik ku tersayang "Bentoni Venson Ruliawan Pardosi dan Mori Vita Victoria Pardosi"
6. Teman-teman Ekstensi Teknik Elektro 2009, atas semangatnya pada waktu mengambil data dan menyelesaikan skripsi ini.

Penulis menyadari bahwa masih banyak kekurangan-kekurangan dalam skripsi ini, baik dalam susunan kalimat, paragraf , halaman, penyusunan proses perancangannya, proses perancangan, dan juga dalam pembuatan barang jadinya.

Oleh karena itu penulis sangat mengharapapkan kritik, ide, saran, dan pandangan-pandangan yang mendukung dari para pembaca, agar untuk kedepanya penulis dapat lebih senpurna lagi dalam perancangan dan pembuatan suatu sistem kontrol.

Depok, Desember 2011

Penulis

HALAMAN PERNYATAAN PERSETUJUAN PUBLIKASI TUGAS AKHIR UNTUK KEPENTINGAN AKADEMIS

Sebagai civitas akademik Universitas Indonesia, saya yang bertanda tangan dibawah ini:

Nama	: Ariel Yagusandri
NPM	: 0906602452
Program Studi	: Teknik Elektro
Departemen	: Teknik Elektro
Fakultas	: Teknik
Jenis karya	: Skripsi

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

RANCANG BANGUN PROTOTIPE SISTEM AKTUATOR SIRIP ROKET MENGGUNAKAN MOTOR SERVO

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

Demikian pernyataan ini saya buat dengan sebenarnya.

Dibuat : Depok

Pada Tanggal : 27 Desember 2011

Yang menyatakan

(Ariel Yagusandri)

ABSTRAK

Nama : Ariel Yagusandri
Program Study: Teknik Elaktro
Judul : Rancang Bangun Prototipe Sistem Aktuator Sirip Roket Menggunakan Motor Servo

Pada saat ini pengembangan roket sudah memasuki tahap roket kendali. Salah satu tema penting pada perkembangan roket adalah pengembangan aktuator sirip roket. Dalam skripsi ini akan dibahas mengenai sistim kontrol aktuator sirip roket kendali. Sirip pada aktuator ini digunakan untuk mengatur arah pergerakan dari roket. Sirip ini di kendalikan dengan menggunakan ATMEGA8535 dengan menggunakan bascomAVR. Mikrokontroler ATMEGA8535 akan menerima data serial dari *autopilot* berupa besaran sudut gerak yang diinginkan dari sirip aktuator, yang mana pergerakan dari sirip ini akan mengubah arah dari roket. Pada kontrol sirip aktuator ini digunakan kontrol PID untuk menghasilkan respon sistem yang baik. Metode chien servo digunakan untuk memperoleh nilai parameter PID. sirip aktuator menggunakan motor servo untuk sebagai penggeraknya. Motor servo yang dipilih harus memiliki ukuran torsi yang sesuai, sehingga dapat menahan seluruh gaya yang terjadi.

Kata kunci: roket, aktuator sirip, mikrokontroler, motor servo, bacomAVR, PID, chien servo

ABSTRACT

Name : Ariel Yagusandri
Study Program: Electrical Engineering
Title : Design of Rocket Fin Actuator System Prototype Using Servo Motor

Nowadays the development rocket has entering guided rocket development phase. One important theme in the development of the rocket is the development of rocket fin actuator. This final project will discuss the control system for fin actuator guided rocket.Fin connected to actuator is used to regulate the movement direction of rocket. The fin is controlled by using ATMEGA8535 programmed by bascomAVR. ATMEGA8535 microcontroller receives serial data from autopilot. The data contains desired angle of fin deflection that is used to change rocket movement. Fin actuator is using PID control in order to get a good system response. The method of chien servo is used to determine the parameter of PID. Fin actuator is using servo motor as its mover. High torque Servo motor is selected, so it can withstand all of the forces that occur in fin.

Keyword: rocket, fin actuator, microcontroller, servo motor, bascomAVR, PID, chien servo

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Pengembangan roket kendali yang dilakukan Indonesia, sedikit demi sedikit telah menunjukkan hasil yang cukup menggembirakan. Hal ini didukung oleh telemetri, sensor navigasi, sensor indra dinamika dan tentu saja system aktuator.

Pada saat ini pengembangan roket sudah memasuki tahap roket kendali. Salah satu tema penting pada perkembangan roket adalah pengembangan sirip aktuator roket. Dalam skripsi ini akan dibahas mengenai sistem kontrol sirip aktuator roket kendali. Pada aktuator ini digunakan sirip untuk mengatur pergerakan arah dari roket. Sirip ini di kendalikan dengan menggunakan ATMEGA8535 dengan menggunakan bahasa basic AVR(bascom AVR). Mikrokontroler ATMEGA8535 akan menerima data serial dari pusat kendali berupa besaran sudut gerak yang diinginkan dari sirip aktuator, yang mana pergerakan dari sirip ini akan mengubah arah dari roket. Sebagai penggerak dari sirip tersebut digunakan motor servo. Motor servo yang dipilih harus memiliki ukuran torsi yang sesuai, sehingga dapat menahan seluruh gaya yang terjadi pada sirip pada saat sirip ini bekerja.

Untuk menghasilkan respon sistem yang baik, maka motor servo yang dikontrol akan menggunakan sistim kontrol *closed-loop* (lingkar tertutup) yaitu kontrol PID. Paramater-parameter kontrol PID akan ditentukan dengan mencari nilai dari parameter PID secara teori, dalam hal ini menggunakan metode chien servo dan kemudian dilakukan pengaturan untuk mendapatkan nilai PID yang paling ideal. Penentuan nilai parameter yang tepat akan menghasilkan respon sitem yang baik.

1.2 Perumusan Masalah

Pada pembuatan kontrol sirip aktuator roket kendali menggunakan mikrokontroler ATMEGA8535 dengan menggunakan bahasa basic AVR(bascom

AVR). Adapun penggerak sirip yang digunakan adalah motor DC servo dengan torsi yang cukup kuat untuk menahan seluruh gaya yang diakibatkan oleh pergerakan roket sendiri. Untuk sudut pergerakan sirip diatur oleh mikrokontroler dengan memperoleh data serial dari pusat kendali roket.

Mengacu pada permasalahan tersebut maka perumusan masalah akan ditekankan pada:

1. Bagaimana membuat program bascom AVR pada mikrokontroler ATMEGA8535 untuk mengendalikan seluruh pergerakan sirip.
2. Bagaimana menentukan nilai parameter kontrol PID(K_P, K_I, K_D) yang sesuai sehingga menghasilkan respon sistem yang diinginkan.
3. Bagaimana menerapkan teori kontrol PID kedalam aplikasi digital pada pemrograman ATMEGA8535.
4. Bagaimana memilih motor servo yang tepat yang dapat digunakan untuk menggerakkan sirip aktuator.

1.3 Tujuan Penelitian

Tujuan penelitian ini adalah:

1. Merencanakan, merancang, dan membuat suatu sistem aktuator sirip untuk mengatur arah pergerakan roket kendali menggunakan motor servo.
2. Mengaplikasikan teknologi kontrol PID pada sirip aktuator roket kendali.

1.4 Manfaat Penelitian

Manfaat penelitian ini adalah:

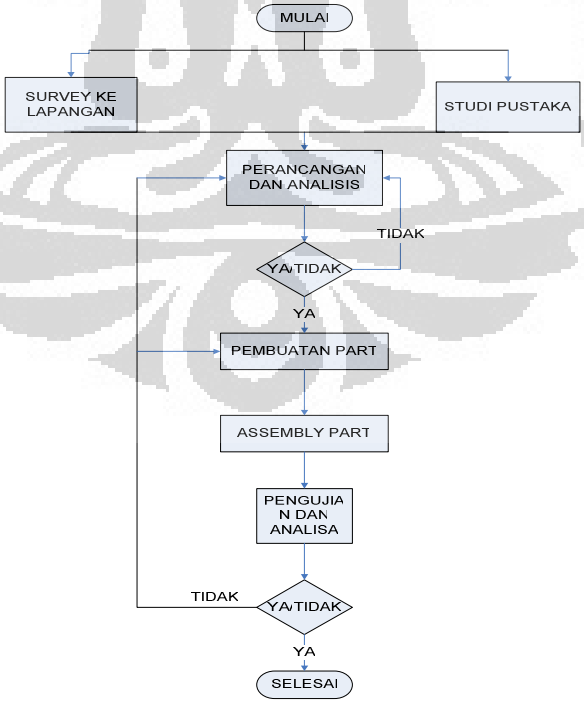
1. Penelitian ini diharapkan dapat menjadi dasar untuk pengembangan penelitian berikutnya seperti menemukan dan membuat sirip aktuator roket kendali beserta dengan system kontrol yang cocok untuk mengendalikan sirip aktuator roket kendali
2. Melalui penelitian ini diharapkan dapat memperoleh pengembangan dalam pembuatan program yang lebih baik untuk kontrol sirip aktuator roket kendali.

1.5 Batasan Masalah

Pembahasan pada penelitian ini difokuskan pada perancangan rangkaian sistem minimum *mikrokontroler*, dan komunikasi serial. Permasalahan dibatasi hanya pada pembuatan perangkat keras dan perangkat lunak yang memberikan data masukan bagi proses pengontrolan sirip aktuator roket kendali. Dengan batasan masalah meliputi:

1. Membuat perangkat lunak yang dapat memberikan data masukan untuk proses pengontrolan sirip aktuator roket kontrol.
2. Mikrokontroler yang digunakan adalah mikrokontroler atmel dari keluarga AVR ATMEGA8535.
3. Untuk komunikasi serial digunakan rangkaian RS 232.
4. Untuk menentukan parameter kontrol PID secara teori menggunakan metode Chien Servo.
5. Untuk penggerak dari sirip aktuator roket kendali digunakan motor servo

1.6 Metodologi Penyelesaian Masalah



Gambar 1.1 Blok Diagram Penyelesaian Masalah

Untuk menyelesaikan permasalahan tersebut, ada beberapa metode yang dilakukan diantaranya:

1. Survey langsung ke lapangan mengenai konstruksi-konstruksi yang ada hubungannya dengan masalah diatas.
2. Studi pustaka, untuk elemen-elemen elektronika yang berhubungan dengan masalah diatas akan dirancang mealui perhitungan yang sesuai dengan disiplin ilmu.
3. Konsultasi dengan dosen pembimbing dan minta pendapat mengenai rancangan kontrol PID yang akan dibuat.

1.7 Sistematika Penulisan

Laporan ini disusun berdasarkan sistematika penulisan yang dibagi dalam lima bab, yang terdiri dari:

BAB I : PENDAHULUAN

Berisi latar belakang masalah, perumusan masalah, tujuan penelitian, manfaat penelitian, batasan masalah serta sistematika penulisan yang digunakan.

BAB II : TINJAUAN PUSTAKA

Merupakan bagian yang menguraikan teori yang mendukung penyelesaian masalah dan teori yang terkait dengan sistem kontrol sirip aktuator roket kendali.

BAB III : PERANCANGAN SISTEM AKTUATOR SIRIP

Bab ini menginformasikan penjelasan tentang metodologi dan tahap- tahap penelitian yang akan dilakukan dari awal hingga akhir dan memuat bahan/alat yang digunakan, serta prosedur penelitian.

BAB IV : PENGUJIAN HASIL DAN ANALISA

Bab ini berisi hasil dari penelitian yang telah dilakukan berdasarkan prosedur yang tertera di Bab III. Dalam bab ini juga terdapat analisis dan pembahasan dari hasil penelitian yang telah diperoleh.

BAB V : PENUTUP

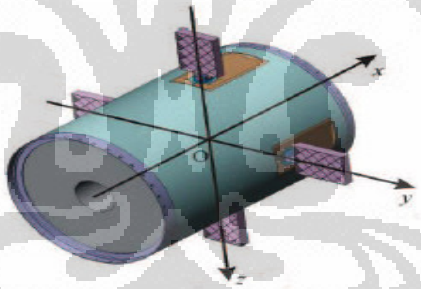
Berisi rangkuman keseluruhan dari inti penelitian yang telah dilakukan, serta mengacu pada hasil yang telah diperoleh. Bab ini berisi kesimpulan dan saran dari hasil penelitian.

BAB II
TINJAUAN PUSTAKA

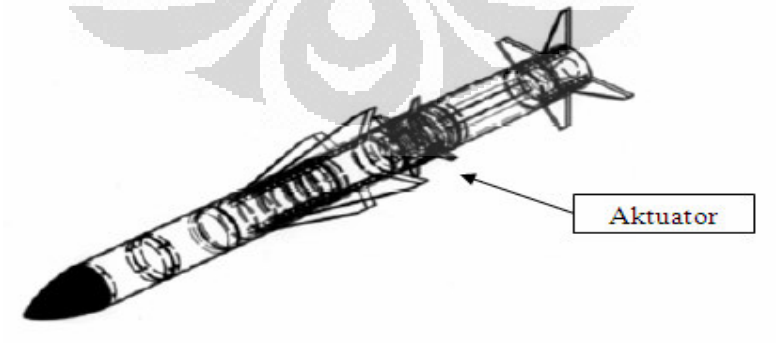
Pada bab ini akan dijelaskan beberapa bagian-bagian yang menjadi dasar dalam pembuatan kendali sirip aktuator roket kendali.

2.1 Pengendalian Sirip Roket

Dalam pergerakannya roket menggunakan sirip untuk mengatur arah geraknya. Sirip roket ini tergabung dalam sebuah bagian pada sebuah roket, dan bagian itu disebut aktuator. Pada aktuator ini memiliki 4 buah sirip yang digunakan untuk mengatur arah terbang dari roket. Aktuator ini bisa berupa sirip dibelakang roket, canard di depan roket ataupun aeileron di sayap roket. Sirip roket ini memiliki peranan yang sangat penting pada akselerasi roket. Perancangan yang kurang bagus akan berpengaruh buruk pada akkselerasi roket. Aktuator roket ini dapat dilihat pada Gambar 2.1 dan posisi letak aktuator pada roket dapat dilihat pada Gambar 2.2.



Gambar 2.1 Aktuator Roket Dengan 4 Sirip



Gambar 2.2 Posisi Aktuator Pada Roket

2.2 Mikrokontroler ATMEGA8535

Mikrokontroler AVR memiliki arsitektur RISC 8 bit, dimana semua instruksi dikemas dalam kode 16-bit (16-bits word) dan sebagian besar instruksi di eksekusi dalam 1 (satu) siklus *clock*, berbeda dengan instruksi MCS51 berteknologi yang membutuhkan 12 siklus *clock*. AVR berteknologi RISC (*Reduced Instruction Set Computing*), sedangakn seri MCS51 berteknologi CISC (*Complex Instruction Set Computing*). Secara umum AVR dapat dikelompokkan menjadi 4 kelas yaitu: keluarga ATtiny, keluarga AT90Sxx, keluarga ATMega, dan AT86RFxx. Pada dasarnya yang membedakan masing-masing kelas adalah memori, peripheral, dan fungsinya.

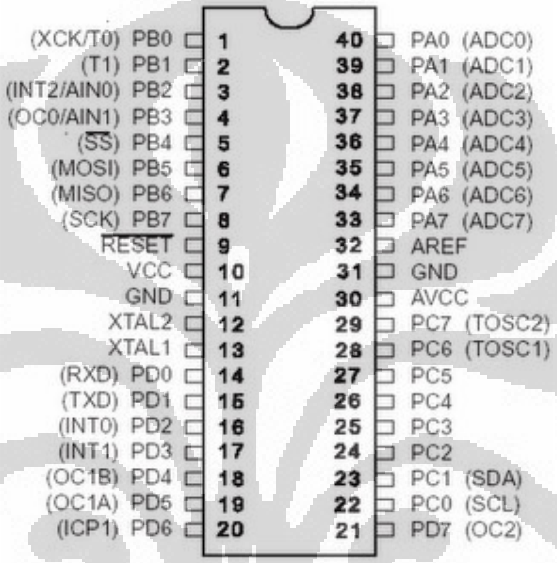
Kapabilitas detail dari ATMega8535 adalah sebagai berikut:

1. Sistem *mikroprosesor* 8 bit berbasis RISC dengan kecepatan meksimal 16 Mhz.
2. Kapabilitas memori flash 8 kb, SRAM sebesar 512 byte, dan EEPROM (*Electrically Erasable Programmable Read Only Memory*) 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.

konfigurasi pin ATMega8535 bisa dilihat pada Gambar 2.3. Dari Gambar 2.4 dapat dijelaskan secara fungsional konfigurasi pin ATMega8535 sebagai berikut:

1. VCC merupakan pin yang berfungsi sebagai pin masukan catu daya.
2. GND merupakan pin ground.
3. Port A (PA0-PA7) merupakan pin I/O dua arah dan pin masukan ADC.
4. Port B (PB0-PB7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu *Timer/Counter*, komparator analog, dan SPI.
5. Port C (PC0-PC7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu TWI, komparator analog, dan Timer Oscilator.

6. Port D (PD0-PD7) merupakan pin I/O dua arah dan pin fungsi khusus, yaitu komparator analog, interupsi eksternal, dan komunikasi serial.
7. RESET merupakan pin yang digunakan untuk me-reset *mikrokontroler*.
8. XTALL1 dan XTALL2 merupakan pin masukan clock eksternal.
9. AVCC merupakan pin masukan tegangan untuk ADC.
10. AREF merupakan pin masukan tegangan referensi ADC.



Gambar 2.3 Pin ATMega8535[7]

2.3 Motor Servo

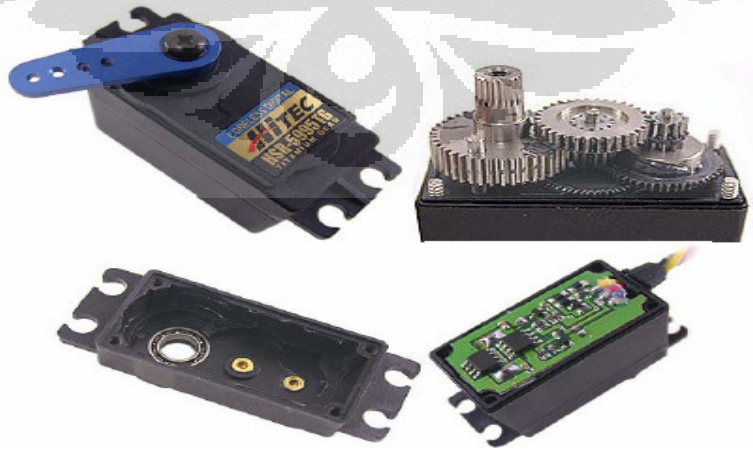
Motor servo adalah sebuah motor dengan sistem closed feedback di mana posisi dari motor akan diinformasikan kembali ke rangkaian kontrol yang ada di dalam motor servo. Motor ini terdiri dari sebuah motor, serangkaian gear, potensiometer dan rangkaian kontrol. Potensiometer berfungsi untuk menentukan batas sudut dari putaran servo. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor. Tampak pada gambar dengan pulsa 1.5 mS pada periode selebar 2 mS maka sudut dari sumbu motor akan berada pada posisi tengah. Semakin lebar pulsa OFF maka akan semakin besar gerakan sumbu ke arah jarum jam dan semakin kecil pulsa OFF maka akan semakin besar gerakan sumbu ke arah yang berlawanan dengan jarum jam.

Motor servo biasanya hanya bergerak mencapai sudut tertentu saja dan tidak kontinyu seperti motor DC maupun motor stepper. Walau demikian, untuk beberapa keperluan tertentu, motor servo dapat dimodifikasi agar bergerak kontinyu. Pada robot, motor ini sering digunakan untuk bagian kaki, lengan atau bagianbagian lain yang mempunyai gerakan terbatas dan membutuhkan torsi cukup besar.

Motor servo adalah motor yang mampu bekerja dua arah (CW dan CCW) dimana arah dan sudut pergerakan rotornya dapat dikendalikan hanya dengan memberikan pengaturan duty cycle sinyal PWM pada bagian pin kontrolnya. Motor Servo tampak pada Gambar 2.4.

Gambar 2.4 Motor Servo

Motor Servo merupakan sebuah motor DC yang memiliki rangkaian control elektronik dan internal gear untuk mengendalikan pergerakan dan sudut angularnya. Sistem Mekanik Motor Servo tampak pada Gambar 2.5.



Gambar 2.5 Sistem Mekanik Motor Servo[3]

Motor servo adalah motor yang berputar lambat, dimana biasanya ditunjukkan oleh rate putarannya yang lambat, namun demikian memiliki torsi yang kuat karena internal gearnya.

Lebih dalam dapat digambarkan bahwa sebuah motor servo memiliki :

- 3 jalur kabel : power, ground, dan control
- Sinyal control mengendalikan posisi
- Operasional dari servo motor dikendalikan oleh sebuah pulsa selebar ± 20 ms, dimana lebar pulsa antara 0.5 ms dan 2 ms menyatakan akhir dari range sudut maksimum.
- Konstruksi didalamnya meliputi internal gear, potensiometer, dan feedback control.

2.3.1 Jenis Motor Servo

1. Motor Servo Standar 180°

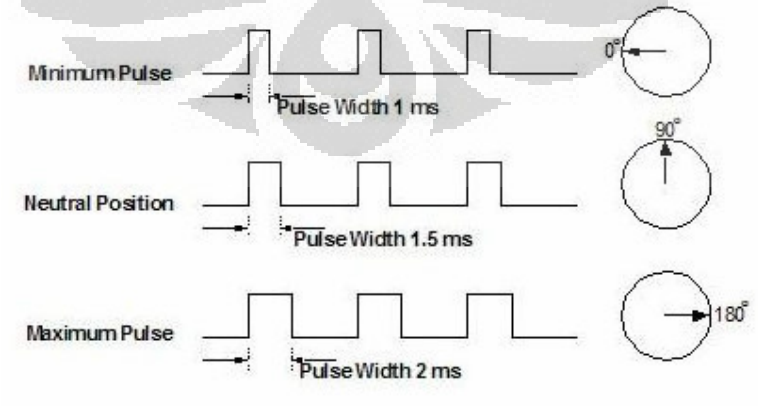
   Motor servo jenis ini hanya mampu bergerak dua arah (CW dan CCW) dengan defleksi masing-masing sudut mencapai 90° sehingga total defleksi sudut dari kanan – tengah – kiri adalah 180°.

2. Motor Servo Continous

   Motor servo ini mampu bergerak dua arah (CW dan CCW) tanpa batasan defleksi sudut putar(dapat berputar secara kontinyu).

2.3.2 Pensinyalan Motor Servo

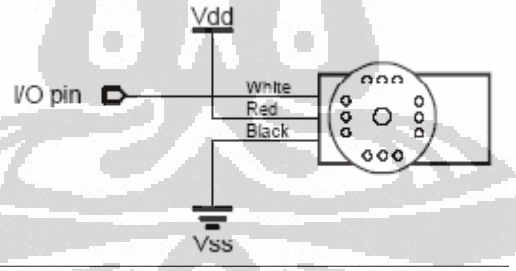
Mode pensinyalan motor servo tampak pada Gambar 2.6.



Gambar 2.6 Pensinyalan Motor Servo[3]

Contoh dimana bila diberikan pulsa dengan besar 1.5ms mencapai gerakan 90 derajat, maka bila kita berikan data kurang dari 1.5 ms maka posisi mendekati 0 derajat dan bila kita berikan data lebih dari 1.5 ms maka posisi mendekati 180 derajat. Beberapa hal yang harus diperhatikan dalam pensinyalan motor servo adalah:

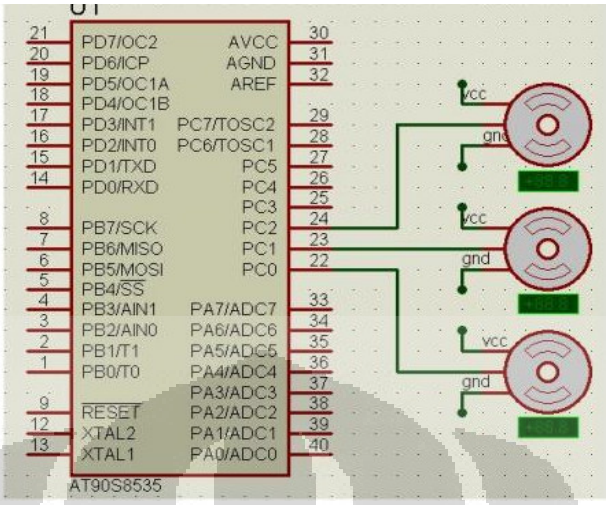
1. Motor Servo akan bekerja secara baik jika pada bagian pin kontrolnya diberikan sinyal PWM dengan frekuensi 50Hz.
2. Dimana pada saat sinyal dengan frekuensi 50Hz tersebut dicapai pada kondisi Ton duty cycle 1.5ms, maka rotor dari motor akan berhenti tepat di tengah-tengah (sudut 0° / netral).
3. Pada saat Ton duty cycle dari sinyal yang diberikan kurang dari 1.5ms, maka rotor akan berputar ke arah kiri dengan membentuk sudut yang besarnya linier terhadap besarnya Ton duty cycle, dan akan bertahan diposisi tersebut.
4. Dan sebaliknya, jika Ton duty cycle dari sinyal yang diberikan lebih dari 1.5ms, maka rotor akan berputar ke arah kanan dengan membentuk sudut yang linier pula terhadap besarnya Ton duty cycle, dan bertahan diposisi tersebut.



Gambar 2.7 Pin-Pin dan Pengkabelan Pada Motor Servo[3]

2.3.3 Rangkaian Driver Motor Servo

Rangkaian berikut adalah rangkaian driver motor servo. Rangkaian tersebut digunakan untuk mengendalikan motor servo.



Gambar 2.8 Rangkaian Driver Motor Servo[3]

2.4 Rotary Sensor

Rotary sensor adalah sensor yang digunakan untuk memeriksa sebuah elemen yang berputar, apakah putaran tersebut telah sesuai dengan yang diinginkan. Prinsip kerja rotary sensor ini memanfaatkan perubahan hambatan dari potensiometer jika posisinya diubah. Secara umum output yang digunakan adalah perubahan nilai tegangan sebagai akibat dari perubahan resistansi dari potensiometer pada rotary sensor. Rotary sensor sendiri memiliki banyak jenis, pada skripsi ini rotary sensor yang digunakan adalah rotation sensor buatan DF Robot. Contoh rotary sensor dapat dilihat pada Gambar 2.9.

Gambar 2.9 Rotary Sensor

2.5 Roda Gigi

Rodagigi digunakan untuk mentransmisikan daya besar dan putaran yang tepat. Rodagigi memiliki gigi di sekelilingnya, sehingga penerusan daya dilakukan oleh gigi-gigi kedua roda yang saling berkait. Rodagigi sering digunakan karena dapat meneruskan putaran dan daya yang lebih bervariasi dan lebih kompak daripada menggunakan alat transmisi yang lainnya. Rodagigi harus

mempunyai perbandingan kecepatan sudut tetap antara dua poros. Di samping itu terdapat pula rodagigi yang perbandingan kecepatan sudutnya dapat bervariasi. Ada pula rodagigi dengan putaran yang terputus-putus.

Roda gigi diklasifikasikan seperti Tabel 2.1, menurut letak poros, arah putaran, dan bentuk jalur gigi.

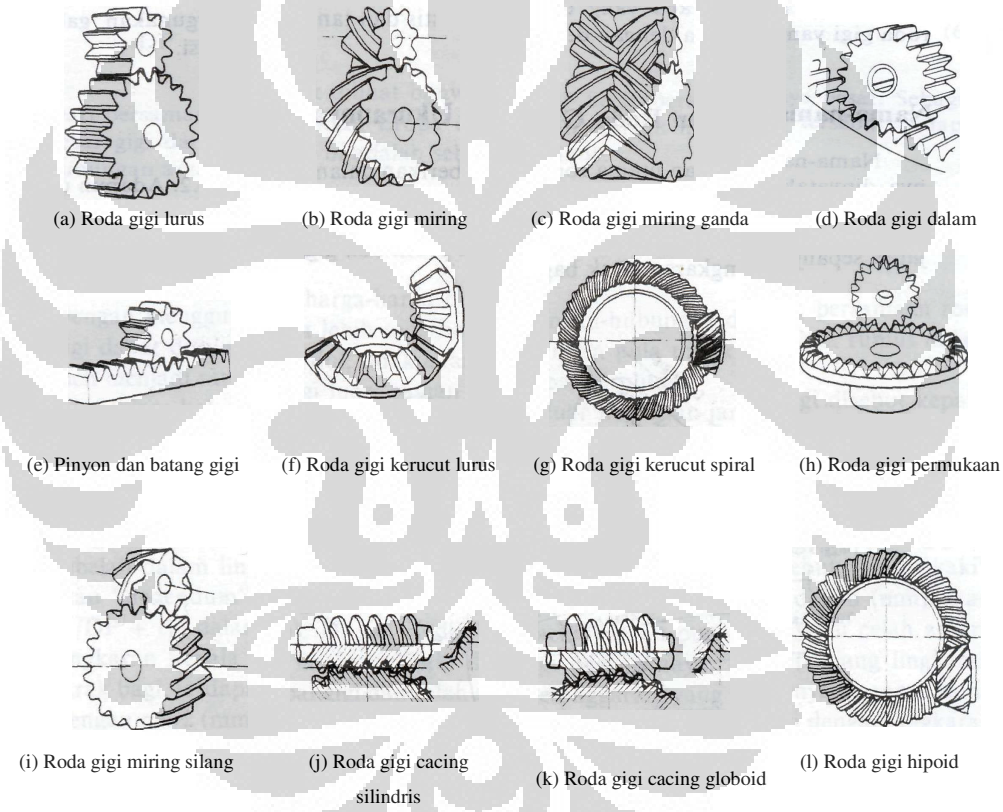
Roda gigi dengan poros sejajar adalah roda gigi dimana giginya berjajar pada dua bidang silinder (disebut ”bidang jarak bagi”); kedua bidang silinder tersebut bersinggungan dan yang satu menggelinding pada bagian yang lain dengan sumbu tetap sejajar.

Tabel 2.1 Klasifikasi roda gigi[2]

Letak Poros	Roda Gigi	Keterangan
Roda gigi dengan poros sejajar	Roda gigi lurus,(a) Roda gigi miring,(b) roda gigi miring ganda,(c)	(Klasifikasi atas dasar bentuk alur gigi)
	Roda gigi luar Roda gigi dalam dan pinyon,(d) Batang gigi dan pinyon,(e)	Arah putaran berlawanan Arah putaran sama Gerakan lurus dan berputar
Roda gigi dengan poros berpotongan	Roda gigi kerucut lurus,(f) Roda gigi kerucut spiral,(g)Roda gigi ZEROL Roda gigi kerucut miring Roda gigi kerucut miring ganda	(klasifikasi atas dasa bentuk jalur gigi)
	Roda gigi permukaan dengan poros berpotongan.(h)	(Roda gigi dengan poros berpotongan berbentuk istimewa)
Roda gigi dengan poros silang	Roda gigi miring silang,(i) Batang gigi miring silang	Kontak titik gerakan lurus dan berputar
	Roda gigi cacing silindris,(j) Roda gigi cacing selubung ganda (globoid),(k) Roda gigi cacing	

	samping,(l)	
	Roda gigi hyperboloid Roda gigi hipoid, Roda gigi permukaan silang	

Dalam teori roda gigi pada umumnya dianut anggapan bahwa roda gigi merupakan benda kaku yang hampir tidak mengalami perubahan bentuk untuk jangka waktu lama. Namun pada transmisi harmonis, dipergunakan gabungan roda gigi yang bekerja dengan deformasi elastis dan tanpa deformasi.



Gambar 2.10 Macam-macam Roda Gigi[2]

2.6 Port Serial Mikrokontroler

AVR ATMega8535 memiliki 4 buah *register* I/O yang berkaitan dengan komunikasi memakai UART, yaitu UART I/O Data Register (UDR), UART Baud Rate Register (UBRR), UART Status Register (USR) dan UART Control Register (UCR).

2.6.1 Register Data I/O UART(UDR)

Proses pengiriman data secara serial dapat dimulai setelah UDR diberi karakter data. Pada sisi penerima, UART memiliki *buffer* sehingga UDR dapat dibaca ketika sebuah data baru sedang digeser masuk.

2.6.2 Register Baud Rate UART(UBRR)

UBRR digunakan untuk menentukan *clock* yang dibangkitkan oleh *baud rate generator*. Nilai *baud rate* ditentukan dengan mengisi UBRR sesuai persamaan 2.1[7].

$$\mathbf{UBRR} = \frac{fclock}{16 * baud} - 1 \qquad (2.1)$$

Misal, diinginkan *baud rate* sebesar 9600 bps. Dengan kristal 11.059.200 Hz maka register UBRR akan bernilai 71.

2.6.3 Register Status UART(USR)

Register USR menyimpan berbagai *flag* status seperti interupsi, *overflow*, dan *framing error*. Susunan bit register USR ditunjukkan dalam Gambar 2.11.

bit	7	6	5	4	3	2	1	0
	RXC	TXC	UDRE	FE	OR	-	-	-

Gambar 2.11 Register status UART[7]

Penjelasan bit-bit register USR adalah sebagai berikut

1. RXC (*Receive Complete*) ; bernilai 1 otomatis setelah UART menerima sebuah karakter secara lengkap,
2. TXC (*Transmit Complete*) ; bernilai 1 jika sebuah karakter telah selesai digeser keluar dari register geser kirim,
3. UDRE (UART *Data Register Empty*) ; bernilai 1 jika UDR kosong,
4. FE (*Framing Error*) ; bernilai 1 jika Stop Bit tidak diterima dengan benar, yaitu jika Stop Bit terbaca 0,

OR (*Overrun*) ; bernilai 1 jika ada karakter yang dipindahkan dari register geser terima ke UDR sebelum karakter yang diterima sebelumnya dibaca.

2.6.4 Register Kontrol UART(UCR)

Register UCR mengendalikan berbagai fungsi penerima dan pengirim, serta interupsinya. Susunan bit *register* UCR ditunjukkan dalam Gambar 2.12.

bit	7	6	5	4	3	2	1	0
	RXCIE	TXCIE	UDRIE	RXEN	TXEN	CHR9	RXB8	TXB8

Gambar 2.12 Register kendali UART[7]

Penjelasan bit-bit register UCR adalah sebagai berikut

1. RXCIE (*Receive Complete Interrupt Enable*) ; jika bernilai 1, UART akan membangkitkan interupsi ketika sebuah karakter selesai diterima,
2. TXCIE (*Transmit Complete Interrupt Enable*) ; bernilai 1 setelah karakter terkirim dan membangkitkan interupsi,
3. UDRIE (UART *Data Register Empty Interrupt Enable*) ; jika bernilai 1, sebuah interupsi terjadi ketika UDR kosong (bit UDRE bernilai 1),
4. RXEN (*Receiver Enable*) ; jika bernilai 1, penerima UART diaktifkan dan pin RxD menjadi pin input yang terhubung ke UART,
5. TXEN (*Transmitter Enable*) ; jika bernilai 1, pengirim UART diaktifkan dan pin TxD menjadi pin output dari pengirim UART,
6. CHR9 (*9-Bit Characters*) ; jika bernilai 1, ukuran karakter yang dikirim menjadi 9-bit, dan bit ke-9 berada pada bit RXB8 dan TXB8,
7. RXB8 (*Receive Data Bit 8*) ; jika CHR9 bernilai 1, bit ini adalah bit ke-9 dari data yang diterima,

8. TXB8 (*Transmit Data Bit 8*) ; jika CHR9 bernilai 1, bit ini adalah bit ke-9 dari data yang dikirim, jadi TXB8 harus diisi sebelum pengiriman.

2.6.5 Inisialisasi USART

Dalam proses inisialisasi maka ada beberapa buah register yang perlu ditentukan nilainya yaitu :

1. UBRR (USART Baud Rate Register).
2. UCSRB (USART Control and Status Register B).
3. UCSRC (USART Control and Status Register C).

UBRR merupakan register 16 bit yang berfungsi untuk melakukan penentuan kecepatan transmisi data yang akan digunakan. UBRR dibagi menjadi dua yaitu UBRRH dan UBRRL seperti pada Gambar 2.13.

URSEL	-	-	-	UBRR[11..8]	UBRRH
UBRR[7..0]					UBRRL

Gambar 2.13 Register UBRR[7]

Bit penyusunnya dapat dijelaskan sebagai berikut :

a. URSEL merupakan bit pemilih antara akses UBRR dan UCSRC. Hal ini disebabkan karena keduanya menempati lokasi yang sama. Untuk akses UBRR maka bit ini bernilai 0.
b. UBRR[11..0] merupakan bit penyimpan konstanta kecepatan komunikasi serial.UBRRH menyimpan 4 bit tertinggi data seting baud rate dan UBRRL menyimpan 8 bit sisanya. Data yang dimasukkan ke UBRRH dan UBRRL dihitung menggunakan rumus sesuai Tabel 2.2. U2X merupakan bit pada register UCSRA.

Tabel 2.2 Rumus penghitungan nilai UBRR untuk berbagai mode operasi[7]

Mode operasi	Rumus nilai UBRR
Asinkron mode kecepatan normal(u2x=0)	$UBRR = \frac{f_{osc}}{16 \times baudrate} - 1$
Asinkron mode kecepatan ganda(u2x=1)	$UBRR = \frac{f_{osc}}{8 \times baudrate} - 1$
sinkron	$UBRR = \frac{f_{osc}}{2 \times baudrate} - 1$

UCSRB merupakan register 8 bit pengatur aktivasi penerima dan pengirim USART. Memiliki komposisi seperti Gambar 2.14.

Gambar 2.14 Register UCSRB[7]

Bit penyusunnya dapat dijelaskan sebagai berikut :

a. RXCIE mengatur aktivasi interupsi penerimaan data serial. Bernilai awal 0 sehingga proses penerimaan data berdasar pada sistem pooling. Jika bernilai 1 maka jika bit RXC pada UCSRA bernilai 1, interupsi penerimaan data serial akan dieksekusi.
b. TXCIE mengatur aktivasi interupsi pengiriman data serial. Bernilai awal 0. Jika bernilai 1 maka jika bit TXC pada UCSRA bernilai 1, interupsi pengiriman data serial akan dieksekusi.
c. UDRIE mengatur aktivasi interupsi yang berhubungan dengan kondisi bit UDRE pada UCSRA. Bernilai awal 0. Jika bernilai 1 maka interupsi akan terjadi jika bit UDRE bernilai 1.
d. RXEN merupakan bit aktivasi penerima serial ATMEGA8535. Bernilai awal 0. Jika bernilai 1 maka penerima data serial diaktifkan.

e. TXEN merupakan bit aktivasi pengirim serial ATMEGA8535. Bernilai awal 0. Jika bernilai 1 maka pengirim data serial diaktifkan.

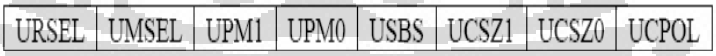
f. UCSZ2 bersama dengan bit UCSZ1 dan UCSZ0 di register UCSRC menentukan ukuran karakter serial yang dikirimkan.

Pada saat awal maka ukuran karakter diset pada 8 bit. Detail nilai bit-bit ini seperti pada Tabel 2.3.

Tabel 2.3 Penentuan ukuran karakter[7]

UCSZ[2..0]	Ukuran karakter dalam bit
000	5
001	6
010	7
011	8
100 – 110	Tidak dipergunakan
111	9

UCSRC merupakan register 8 bit yang dipergunakan untuk mengatur mode dan kecepatan komunikasi serial yang dilakukan. Memiliki komposisi seperti Gambar 2.15.



Gambar 2.15 Register UCSRC[7]

Bit penyusunnya dapat dijelaskan sebagai berikut :

a. URSEL merupakan bit pemilih akses antara UCSRC dan UBRR. Bernilai awal 1 sehingga secara normal akan selalu mengakses register UCSRC.

b. UMSEL merupakan bit pemilih mode komunikasi serial antara sinkron dan asinkron. Bernilai awal 0 sehingga modenya asinkron. Jika bernilai 1 maka modenya sinkron.

c. UPM[1..0] merupakan bit-bit pengatur paritas. Bernilai awal 00 sehingga paritas tidak dipergunakan. Detail nilainya dapat dilihat pada Tabel 2.4.

Tabel 2.4 Penentuan mode paritas[7]

UPM[1..0]	Mode paritas
00	Tidak aktif
01	Tidak digunakan
10	Paritas genap
11	Paritas ganjil

d. USBS merupakan bit pemilih ukuran bit stop. Bernilai awal 0 sehingga jumlah bit stop yaitu 1 bit. Jika bernilai 0 maka jumlah bit stop yaitu 2 bit.

e. UCSZ1 dan UCSZ0 merupakan bit pengatur jumlah karakter serial.

UCPOL merupakan bit pengatur hubungan antara perubahan data keluaran dan data masukan serial dengan clock sinkronisasi. Hanya berlaku untuk mode sinkron. Untuk mode asinkron bit ini diset 0.

2.6.6 Mengirim Data Melalui Port Serial

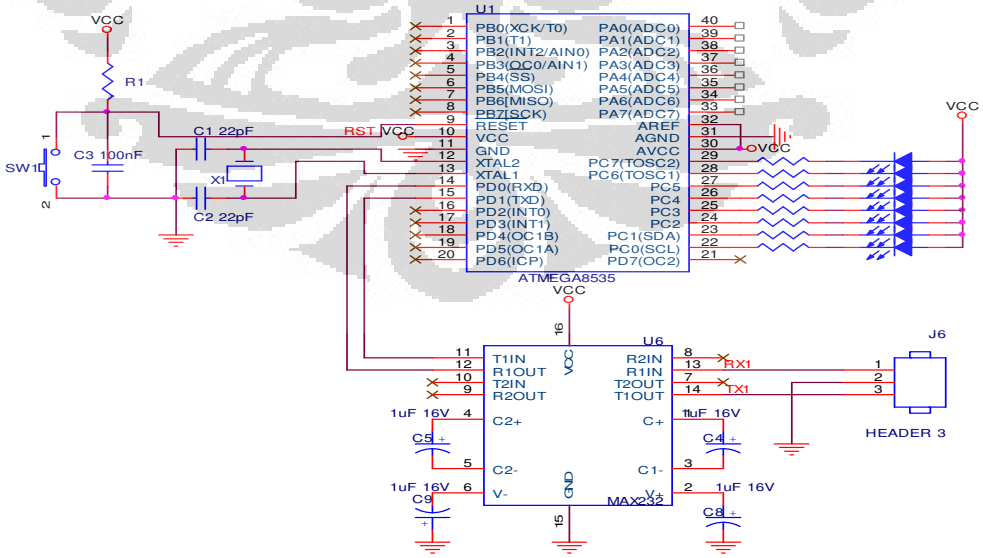
Proses pengiriman data serial dilakukan per byte data dengan menunggu register UDR yang merupakan tempat data serial akan disimpan menjadi kosong sehingga siap ditulis dengan data yang baru. Proses ini menggunakan bit yang ada pada register UCSRA, yaitu bit UDRE (USART Data Register Empty). Bit UDRE merupakan indikator kondisi register UDR. Jika UDRE bernilai 1 maka register UDR telah kosong.

2.6.7 Menerima Data Melalui Port Serial

Proses penerimaan data serial diakukan dengan mengecek nilai bit RXC (USART Receive Complete) pada register UCSRA. RXC akan bernilai satu jika ada data yang siap dibaca di buffer penerima, dan bernilai nol jika tidak ada data pada buffer penerima. Jika penerima USART dinonaktifkan maka bit ini akan selalu bernilai nol.

2.6.8 Rangkaian Serial Mikrokontroler

Rangkaian berikut digunakan untuk interfacing Led dengan port serial. Rangkaian tersebut, sebagai konverter dari serial ke pararel. Berikut adalah rangkaian serial led driver yang akan kita hubungkan pada port serial. Rangkaian Led Driver Serial menggunakan Microcontroller ATMEGA8535 yang dihubungkan ke port serial dengan menggunakan IC RS232 Rangkaian Serial LED Driver ini akan mendeteksi setiap pengiriman data karakter dari port serial computer.



Gambar 2.16 Hasil Pemasangan Komponen Rangkaian Serial Mikrokontroler[3]

2.7 Analog To Digital Converter(ADC)

ADC digunakan untuk mengubah sinyal input yang analog menjadi sinyal digital agar dapat diolah oleh mikrokontroler. Adapun fitur dari ADC ATMEGA8535 adalah sebagai berikut:

1. Resolusi 10 bit.
2. Waktu konversi 65-260 us.
3. 0-vcc *range input* ADC.
4. Memiliki 8 *channel* input(A.0-A.7).
5. Tiga mode pemilihan tegangan referensi.

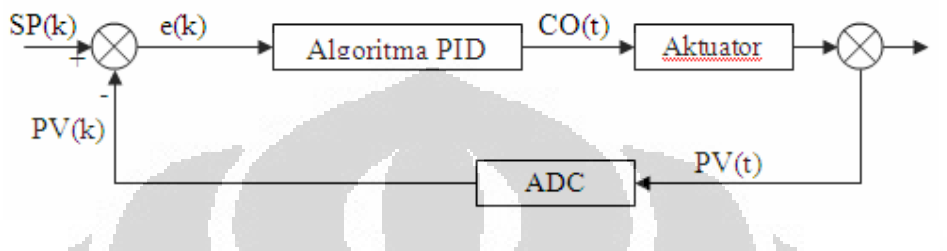
Pada ADC ATMEGA8535 digunakan dua mode ADC, yaitu single conversion dan free running. Pada mode single conversion, pengguna harus mengaktifkan setiap kali ADC akan digunakan. Sedangkan pada mode free running, pengguna cukup sekali saja mengaktifkan, sehingga ADC akan terus mengkonversi tanpa henti. Terdapat beberapa register I/O yang terlibat dalam proses konversi ADC, antara lain: ADMUX (ADC Multiplesxer Selsection Register). Register ADMUX berisi bit-bit yang mengatur pilihan kanal (MUX4:0), bit pengatur penyajian data (ADLAR), dan bit pemilih tegangan referensi (REFS1:0). Gambar 2.20 menunjukkan isi register ADMUX.

2.8 Kontrol PID

Pada pembahasan skripsi ini menggunakan realisasi praktis dari PID digital. Istilah PID digital pada dasarnya mengacu pada jenis perangkat keras dimana sistem kontrol PID tersebut ditanamkan. Berbeda dengan kontrol PID analog yang realisasi praktisnya dijumpai dalam bentuk perangkat keras rangkaian

elektronika, sistem kontrol PID digital implementasinya dapat dijumpai dalam bentuk persamaan matematis yang ditanam pada sistem mikroprosesor.

Dalam bentuk diagram blok, kontrol PID digital dapat diilustrasikan seperti Gambar 2.17.



Gambar 2.17 Diagram blok Modul PID digital[6]

Berbeda dengan kontrol PID analog yang pengolahannya bersifat kontinyu. Didalam sistem mikroprosesor, pengolahan sinyal kontrol oleh modul digital dilakukan hanya pada waktu-waktu diskret. Dalam hal ini, konversi sinyal dari analog ke digital, pengolahan sinyal error, sampai konversi balik digital ke analog dilakukan pada interval atau waktu cuplik(sampling) Tc tertentu. Luas nya penggunaan kontrol PID pada dasarnya dilatarbelakangi beberapa hal diantaranya:

- Kesederhanaan sistem kontrol. Hanya memiliki 3 parameter utama yang perlu diatur(tuning).
- Kontrol PID memiliki sejarah yang panjang. PID telah digunakan jauh sebelum era digital berkembang(1930an).

Kontrol PID dalam banyak kasus telah terbukti menghasilkan unjuk kerja yang relatif memuaskan baik digunakan sebagai sistem regulator maupun sebagai sistem servo.

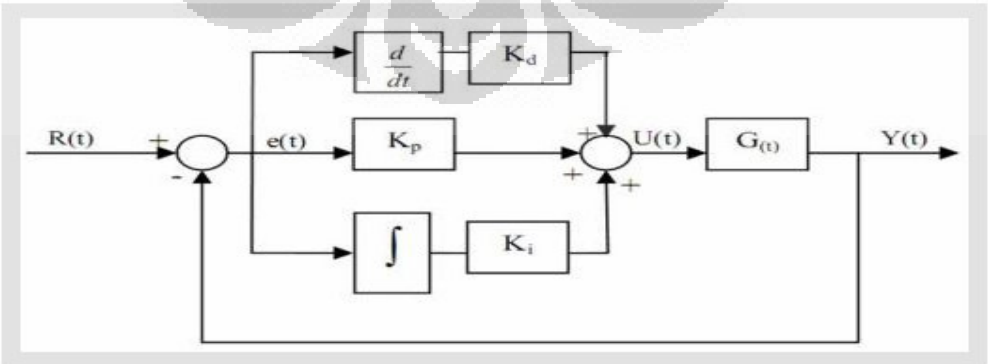
Kontrol PID merupakan gabungan dari kontrol proporsional, kontrol integral, dan kontrol derivatif. Setiap kekurangan dan kelebihan dari masing-masing kontroler P, I dan D dapat saling menutupi dengan menggabungkan

ketiganya secara paralel menjadi kontroler proposional plus integral plus diferensial (kontroller PID). Elemen-elemen kontroller P, I dan D masing-masing secara keseluruhan bertujuan untuk mempercepat reaksi sebuah sistem, menghilangkan *offset* dan menghasilkan perubahan awal yang besar. Pada Tabel 2.5 dapat dilihat karakteristik dari masing-masing kontroler.

Tabel 2.5 Karakteristik Kontroler

Parameter	Proporsional(P)	Integral(I)	Derivatif(D)
Persamaan analog	$u(t) = K_p e(t)$	$u(t) = K_i \int_0^t e(t)$	$u(t) = K_d \frac{de(t)}{dt}$
Fungsi	Memperkuat sinyal kesalahan, sehingga mempercepat keluaran mencapai titik referensi	Menghilanggkan kesalahan tunak, yang dihasilkan oleh kontrol proporsional	Mempercepat respon awal suatu sistem

Gabungan dari kontrol proporsional, integral, dan derivatif adalah kontrol PID. Diagram blok dari kontrol PID dapat dilihat pada Gambar 2.18



Gambar 2.18 Diagram Blok Kontrol PID[6]

Sehingga persamaan untuk kontrol PID adalah[6]:

$$u(t) = K_p e(t) + K_i \int_0^t e(t)dt + K_d \frac{de(t)}{dt} \quad (2.2)$$

$$u(t) = K_p \left(e(t) + \frac{1}{T_i} \int_0^t e(t)dt + T_d \frac{de(t)}{dt} \right) \quad (2.3)$$

Dengan:

u(t) = sinyal output pengendali PID

Kp = konstanta proporsional

Ti = waktu integral

Td = waktu derivatif

Ki = konstanta integral (Kp/Ti)

Kd = konstanta derivatif (Kp x Td)

e(t) = sinyal error = referensi-keluaran plant = set point – nilai sensor

Untuk persamaan PID no. (2.2) merupakan bentuk PID independent dan persamaan PID no.(2.3) merupakan PID bentuk dependent. Istilah tersebut mengacu kepada ketergantungan setiap suku persamaan terhadap nilai Kp. Untuk persamaan no. (2.2), jika dilakukan perubahan nilai kepada konstanta proporsional (Kp) maka tidak akan mempengaruhi konstanta parameter lainnya. Sedangkan untuk persamaan no. (2.3), dengan mengubah nilai Kp makan akan berubah nilai dari parameter-parameter lainnya. Pada pembuatan skripsi ini menggunakan persamaan PID bentuk independent. Jika ingin menggunakan persamaan dependent, maka tinggal memasukkan nilai dari Ki=Kp/Ti dan Kd=Kp x Td.

Pada persamaan (2.2) dan (2.3) merupakan persamaan dalam kawasan waktu continous (analog). Sedangkan agar persamaan tersebut dapat

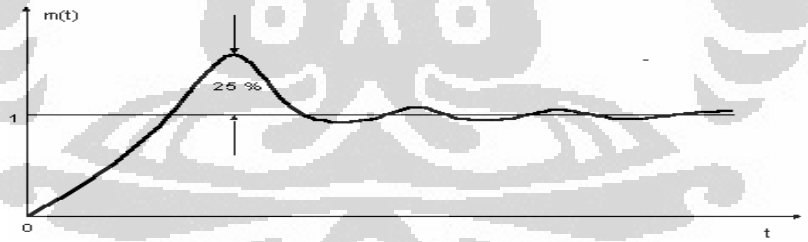
direalisasikan dalam bentuk pemrograman, maka persamaan dalam bentuk waktu continous tersebut harus di diskretisasi terlebih dahulu (kawasan digital).

Aspek yang sangat penting dalam merancang kontrol PID ialah penentuan parameter kontroler PID supaya sistem close loop memenuhi kriteria performansi yang diinginkan, untuk tujuan inilah maka perlu dilakukan tuning PID.

Untuk melakukan tuning kontrol PID dapat dilakukan dengan 2 cara:

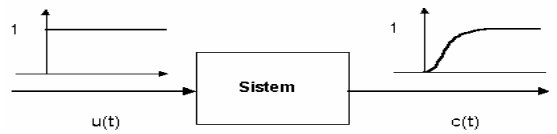
- Untuk menentukan nilai Kp, Ki, Kd dapat dilakukan dengan menggunakan metode trial and error (coba-coba).
- Untuk menentukan nilai Kp, Ki, Kd dapat digunakan dengan menggunakan metode yang sudah ada, pada skripsi ini akan digunakan metode Chien Servo.

Metode Chien Servo ditujukan untuk meghasilkan respon sistem dengan lonjakan maksimum sebesar 25%.

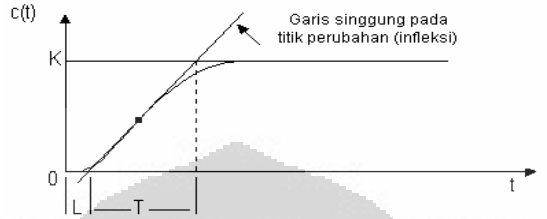


Gambar 2.19 Kurva respons tangga satuan yang memperlihatkan 25 % lonjakan maksimum[8]

Metode kurva reaksi didasarkan terhadap reaksi sistem untaian terbuka. Plant sebagai untaian terbuka dikenai sinyal fungsi tangga satuan (Gambar 2.20). Kalau plant minimal tidak mengandung unsur integrator ataupun pole-pole kompleks, reaksi sistem akan berbentuk S. Gambar 2.21 menunjukkan kurva berbentuk S tersebut.



Gambar 2.20 Respon Tangga Satuan Sistem[8]



Gambar 2.21 Kurva Respon Berbentuk S[8]

Kurva berbentuk-s mempunyai dua konstanta, waktu mati (dead time) L dan waktu tunda T. Dari gambar 2.21 terlihat bahwa kurva reaksi berubah naik, setelah selang waktu L. Sedangkan waktu tunda menggambarkan perubahan kurva setelah mencapai 66% dari keadaan mantapnya. Pada kurva dibuat suatu garis yang bersinggungan dengan garis kurva. Garis singgung itu akan memotong dengan sumbu absis dan garis maksimum. Perpotongan garis singgung dengan sumbu absis merupakan ukuran waktu mati, dan perpotongan dengan garis maksimum merupakan waktu tunda yang diukur dari titik waktu L.

Penalaan parameter PID didasarkan perolehan kedua konstanta itu. Metode Chien Servo menyarankan parameter penyetelan nilai Kp, Ti, dan Td dengan didasarkan pada kedua parameter tersebut. Tabel 2.6 merupakan rumusan penalaan parameter PID berdasarkan cara kurva reaksi.

Tabel 2.6 Penalaran Parameter PID Dengan Metode Kurva Reaksi[6]

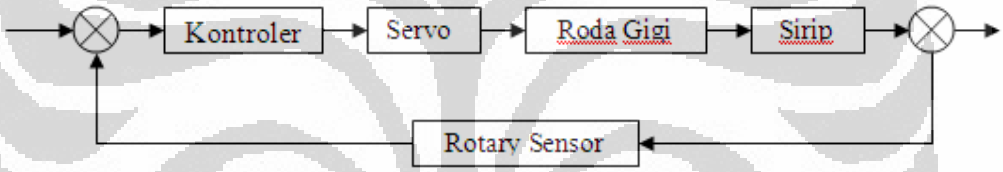
Metoda	K_P	T_I	T_D	K_I	K_D
Chien Servo	$\frac{0.6T}{L}$	T	$0.5L$	$\frac{K_p \times T_C}{T_I}$	$\frac{K_P \times T_D}{T_C}$

BAB III
PERANCANGAN SISTEM AKTUATOR SIRIP

Bab ini menjelaskan perancangan sistem pada actuator sirip mulai dari perancangan mekanik (prototype), elektronik (mikrokontroler), sampai dengan perancangan program.

3.1 Blok Diagram

Dalam perancangan suatu sistem, dibutuhkan suatu blok diagram yang dapat menerangkan sistem secara keseluruhan agar sistem yang dibuat dapat berjalan dengan baik. Berikut blok diagram dalam perancangan kontrol fin aktuator roket kendali.

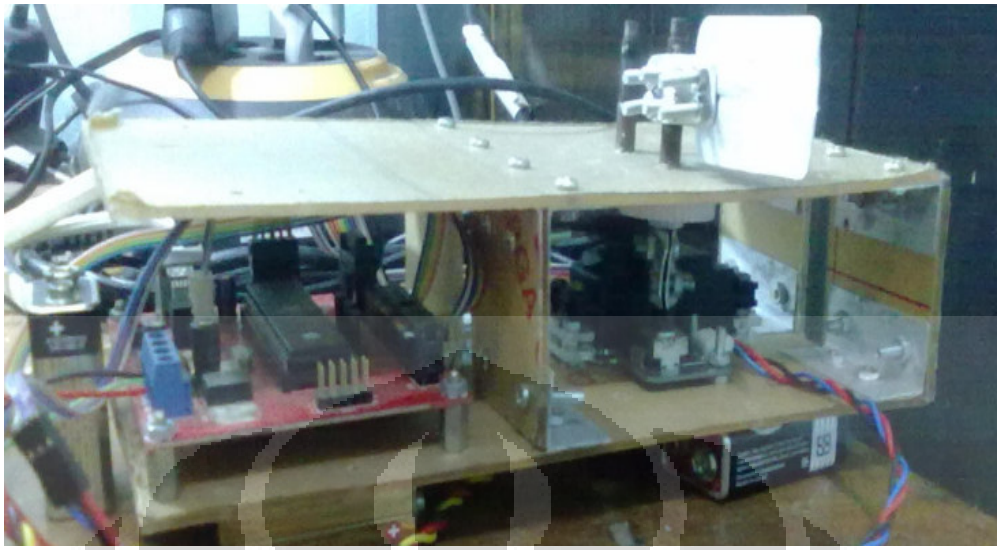


Gambar 3.1 Blok Diagram Sistem

Dari blok diagram diatas dapat dijelaskan bahwa, untuk besarnya derajat pergerakan dari fin aktuator roket kendali diperoleh data serial dari kendali pusat. Kemudian mikrokontroler akan memproses input data serial tersebut yang kemudian akan menghasilkan autput program untuk menggerakkan fin aktuator roket kendali.

3.2 Perancangan Prototype

Sistem mekanik merupakan bagian yang sangat penting pada sirip roket kendali, untuk itu perancangan sistem mekanik harus sangat diperhatikan mulai konfigurasi roda gigi yang digunakan sampai sirip yang akan digunakan. Pada perancangan prototype sirip menggunakan 2 buah roda gigi lurus dengan ukuran diameter yang sama. Pada prototype seperti Gambar 3.2, dapat dilihat sistem kerja mekanis sirip aktuator roket kendali.



Gambar 3.2 Prototype Sirip Aktuator Roket Kendali

Output dari motor servo tidak langsung digunakan sebagai keluaran untuk menngerakkan sirip, tetapi dihubungkan terlebih dahulu dengan roda gigi lain yang terhubung dengan poros dari sirip. Poros sirip ini juga terhubung dengan sensor rotary. Untuk lebih jelas dapat dilihat pada Gambar 3.3.

Gambar 3.3 Hubungan Roda Gigi Pada Servo dengan Roda Gigi Pada Poros Sirip

3.2.1 Kebutuhan Torsi

Mengetahui kebutuhan torsi untuk menggerakkan sirip pada kondisi roket sedang terbang pada kecepatan 0,9 Mach berguna untuk menentukan jenis motor yang akan dipakai serta sistem transmisinya. Kondisi terbang roket kendali seperti dalam Tabel 3.1.

Tabel 3.1 Kondisi Terbang Roket Kendali[11]

Kecepatan dengan menggunakan booster	0,7 Mach
Kecepatan dengan menggunakan sustainer	0,9 Mach
Ketinggian terbang (*Altitude*)	Maks. 2 km dari atas permukaan air laut
Sudut Elevasi	60^o

Untuk memperkirakan besar torsi yang dibutuhkan, dapat digunakan persamaan 3.1 sebagai persamaan empiris untuk menghitung perkiraan gaya pada sirip(*Drag*) dan persamaan 3.2 sebagai persamaan untuk menghitung nilai torsi[11].

$$D = \frac{1}{2}\rho V^2 C_D \qquad (3.1)$$

$$\tau = F \times r \qquad (3.2)$$

Tabel 3.2 Parameter Kerja Roket[11]

ρ = *Density of air*	1,225 kg/m^3
V= *Speed*	306,27 m/s
S= *Reference Area*	0,001107939 m^2
C_D= *Coefficient Drag*	0,265

Apabila menggunakan perhitungan empiris dengan menggunakan rumus, akan didapatkan besar *Drag* = 16,87 N.

$$D = \frac{1}{2}\rho V^2 S C_D$$

$$D = \frac{1}{2.1{,}225} . 306{,}27^{2.0{,}001107939} . 0{,}265$$

$$D = 16{,}87\ N$$

Bila dibuat menjadi kebutuhan torsi untuk *shaft* berdiameter 25 mm maka didapatkan nilai torsi sebesar 0,210856466 Nm.

$$\tau = F\ x\ r$$

$$\tau = 16{,}87\ N\ x\ 0{,}0125\ m$$

$$\tau = 0{,}210856466\ \mathbf{Nm}$$

Tabel 3.3 Konversi Satuan Torsi Hasil Perhitungan Empiris[11]

Torque	0,210856	Nm
	0,021714	Kgfm
	30,15425	oz in
	2,171422	kgfcm

Menurut perhitungan numerik dan empiris torsi yang dibutuhkan sekitar 0,21-0,236 Nm. Berdasarkan perhitungan tersebut, maka dibutuhkan motor yang memiliki torsi diatas dari nilai hasil perhitungan.

3.2.2 Pemilihan Motor Servo

Motor *Servo* yang dipakai adalah motor *servo* digital dengan torsi yang dihasilkan adalah 3,495 N m. Motor jenis ini dipilih karena memiliki keistimewaan. Keistimewaan motor ini adalah mampu dikontrol dan memiliki ketepatan derajat yang baik. Motor servo ini juga memiliki torsi di atas dari perkiraan torsi yang dibutuhkan oleh sirip, sahingga motor servo ini diperkirakan mampu bekerja dengan baik. Dapat dilihata pada Gambar 3.4.

Gambar 3.4 Motor Servo HS-798OTH

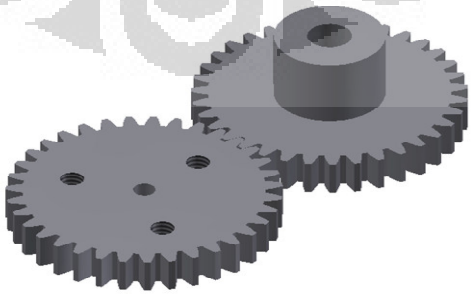
Tabel 3.4 Spesifikasi Motor Servo HS-798OTH

	AT 6,0 Volt	AT 7,4 Volt
Operating Speed	0,21 sec/ 60°	0,17 sec/ 60°
Output Torque	36,0 kg-cm (500 oz-in)	44,0 kg-cm (611 0z-in)
Weight	78,2 g (2,76oz)	
Size	43,8 x 22,4 x 40,0mm (1,72x0,88x1,57in)	

3.2.3 Perancangan Roda Gigi

Pada desain sistem transmisi yang dipakai adalah *spur gear*. *Spur gear* memiliki keistimewaan dapat mentransmisikan daya tanpa adanya slip. Dirancang dengan ratio 1:1 dan memiliki jenis material A322-4340 dengan proses *heat treatment* berupa *nitrited*.

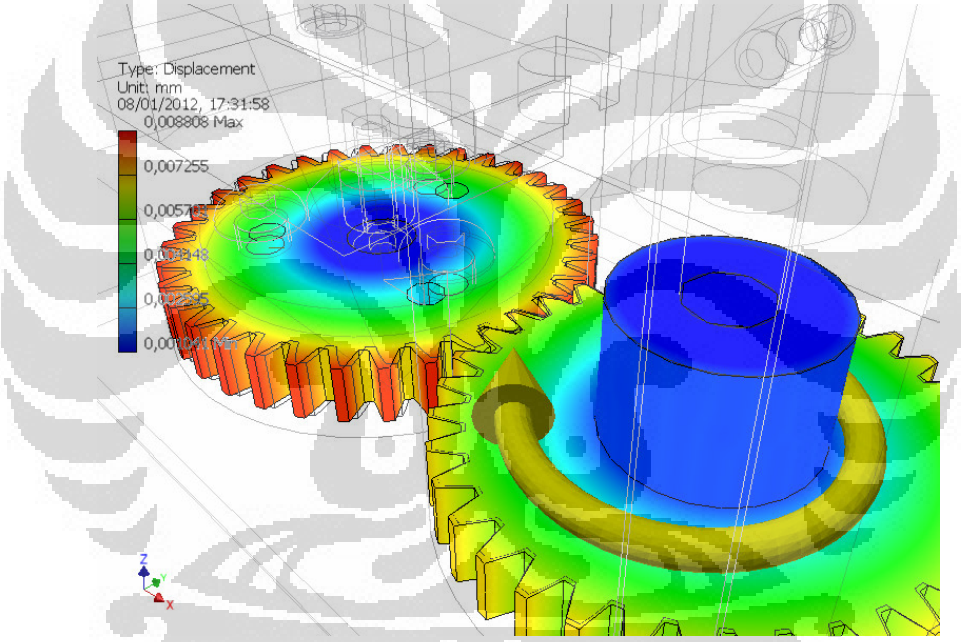
Simulasi *analysis stress* pada roda gigi tersebut dilakukan untuk melihat apakah desain ini *compliance* atau tidak. Simulasi tersebut menggunakan *software* Autodesk Inventor[11].



Gambar 3.5 Spur Gear *Pada Modul Sirip[11]*

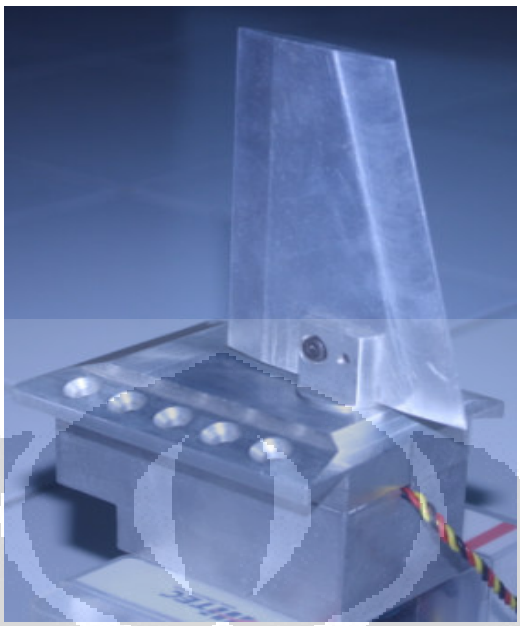
Tabel 3.5 *Result Summary Stress Analysis Spur Gear* Pada Modul Sirip[11]

Name	Minimum	Maximum
Volume	277243 mm^3	
Mass	0,0833134 kg	
Von Mises Stress	0,00148643 MPa	128,945 MPa
1st Principal Stress	-30,066 MPa	136,806 MPa
3rd Principal Stress	-141,034 MPa	30,6126 MPa
Displacement	0,00104134 mm	0,00880813 mm
Safety Factor	1,60533 ul	15 ul



Gambar 3.6 Stress Analysis Type Displacement Spur Gear Modul Aktuator Sirip[11]

Dari Gambar 3.7 dapat dilihat modul aktuator sirip roket kendali dengan menggunakan motor servo.



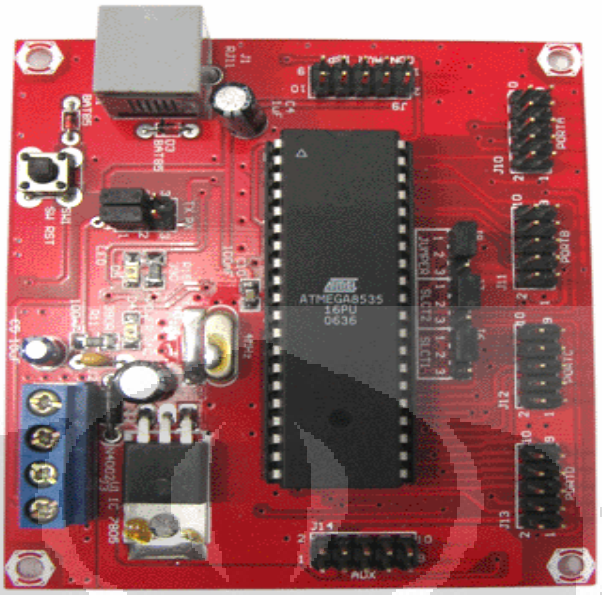
Gambar 3.7 Modul Sirip Aktuator

3.3 Perancangan Elektronik

Perangkat elektronik yang digunakan di sini sebagai komponen yang digunakan untuk mengendalikan sirip roket kendali. Komponen elektronika yang digunakan pada kendali sirip roket ini adalah DT-AVR dan rotary sensor.

3.3.1 DT-AVR

DT-AVR merupakan sebuah modul single chip dengan basis mikrokontroler AVR dan memiliki kemampuan untuk melakukan komunikasi data serial secara UART RS-232 serta pemrograman memory melalui ISP (In-System Programming). Modul ini dapat digunakan untuk komunikasi serial dengan modul lain menggunakan ATMEGA8535 dengan menggunakan kristal 4 MHz. selain itu, untuk mengatur pergerakan dari fin aktuator roket kendali digunakan modul DT-AVR ini. Bentuk modul DT-AVR dapat dilihat pada Gambar 3.8.



Gambar 3.8 DT-AVR ATMEGA8535

3.3.2 Rotary Sensor

Rotary sensor merupakan komponen elektronik yang berfungsi untuk mendeteksi pergerakan dari sirip roket apakah sudah sesuai dengan keinginan. Pada pemasangannya sensor rotary dipasang satu sumbu dengan poros sirip roket, sehingga rotary sensor ini bisa mendeteksi setiap pergerakan dari sirip roket. Gambar 3.9 menunjukkan pemasangan rotary sensor.

Gambar 3.9 Pemasangan Rotary Sensor

3.4 Perancangan Program

Setiap komponen elektronika pada kendali sirip roket dikontrol dengan menggunakan program basic AVR. Beberapa hal yang harus dilakukan pemrograman adalah kendali posisi motor servo, komunikasi serial, ADC, dan kontrol kendali PID.

3.4.1 Kendali Posisi Motor Servo

Perancangan kendali posisi motor servo dengan menggunakan Bascom AVR dilakukan dengan mengatur config servo yang terdapat pada jendela Bascom-AVR IDE. Adapun pengaturan config servo tersebut dapat dilihat pada potongan program dibawah ini:

```
'-----------------------------------------------------
'DEKLARASI KONFIGURASI MOTOR SERVO
'-----------------------------------------------------
Config Servos = 4 , Servo1 = Portc.0 , Servo2 = Portc.1 , Servo3 = Portc.2
, Servo4 = Portc.3 , Reload = 1      'Timer = Timer1
'-----------------------------------------------------
'OUTPUT SERVO
'-----------------------------------------------------
Servo(1) = 70
Servo(2) = 60
Servo(3) = 70
Servo(4) = 80
Waitus 1
```

Dari potongan program diatas sapat dilihat bahwa:

1. Motor servo yang akan digunakan sebagai aktuator adalah sebanyak 4 buah. Perintah yang digunakan untuk mendeklarasikan 4 buah motor servo ini adalah Config Servos = 4
2. Sebagai output keluaran untuk pergerakan motor servo digunakan port c pada sistem minimum DT-AVR ATMEGA8535. Perintah yang digunakan untuk menggunakan port c ini sebagai output untuk pergerakan servo adalah Servo1 = Portc.0 .

3. Empat buah motor servo yang akan bergerak sesuai dengan pulsa yang diberikan. Sebagai contoh: servo(1) akan bergerak menuju sudut 0^0 sesuai dengan pulsa yang diberikan terhadapnya. Dalam hal ini motor servo(1) diberikan pulsa 70.

Setiap besar sudut yang diinginkan memiliki besar pulsa yang berbeda-beda. Untuk itu, pada Tabel 3.6 akan ditunjukkan nilai sudut yang diinginkan dengan lebar pulsa yang harus diberikan.

Tabel 3.6 Besar Sudut dan Lebar Pulsa

Sudut(Derajat)	Lebar Pulsa
-10	65
-9	66
-8	66
-7	67
-6	67
-5	68
-4	68
-3	69
-2	69
-1	70
0	70
1	70
2	71
3	71
4	72
5	72
6	73
7	73
8	74
9	74
10	75

Pada program pergerakan motor servo ini, setiap satu kali pulsa yang diberikan motor akan bergerak sebesar 2^0 dan kelipatannya. Apabila motor servo ingin digerakkan membentuk sudut sebesar 1^0 atau kelipatanya, maka ini tidak bisa dilakukan. Supaya itu dapat dilakukan maka harus dilakukan modifikasi program pada sintax bascomAVR. Sehingga setiap 1 kali pulsa yang diberikan motor servo dapat bergerak sebesar 1^0 atau setiap kelipatanya.

3.4.2 Komunikasi Serial

Dalam perancangan komunikasi serial dengan menggunakan Bascom-AVR cukup sederhana. Salah satu bagian yang paling penting dalam merancang komunikasi serial dengan Bascom-AVR adalah mengatur baudrate. Nilai baudrate pada sistem minimum AVR ATMEGA8535 harus sama dengan dengan niai baudrate pada sistem yang akan dihubungkan secara serial. Potongan program dibawah ini menggambarkan program dasar komunikasi serial.

```
'-----------------------------------------------------
'DEKLARASI CRISTAL 4MHz
'-----------------------------------------------------
$crystal = 4000000
'-----------------------------------------------------
'DEKLARASI HEADER
'-----------------------------------------------------
$regfile = "m8535.dat"
'-----------------------------------------------------
'-----------------------------------------------------
'Deklarasi Baud Rate
'-----------------------------------------------------
$baud = 9600
'-----------------------------------------------------
'-----------------------------------------------------
'Deklarasi Variabel
'-----------------------------------------------------
Dim S As String * 1
```

```
Print "Program Kontrol Fin Aktuator Roket Kendali"
```

Dari program diatas dapat dilihat bahwa sistem minimum mikrokontroler menggunakan cristal 4MHz, dengan baudrate sebesar 9600. Baudrate ini berguna untuk menentukan kecepatan pengiriman data. Pada potongan program tersebut dapat dilihat bahwa sistem minimum mikrokontroler akan mengirim karakter ” Program Kontrol Fin Aktuator Roket Kendali” dengan kecepatan pengiriman data 9600 bps.

3.4.3 ADC

Ada beberapa hal yang harus diperhatikan dalam pengaturan config ADC pada Bascom-AVR. Beberapa yang harus diperhatikan itu adalah pemilihan config ADC, kemudian pemilihan prescaler untuk proses pembacaan ADC. Potongan program dibawah ini menggambarkan proses pembacaan serial mikrokontroler menggunakan Bascom-AVR ATMEGA8535.

```
'-----------------------------------------------------
'KONFIGURASI ADC
'-----------------------------------------------------
Config Adc = Single , Prescaler = Auto
Start Adc
'-----------------------------------------------------
'ambil data sensor dari ADC
'-----------------------------------------------------
L = 0
H = Getadc(0)
Print "Channel " ; L ; " value " ; H
```

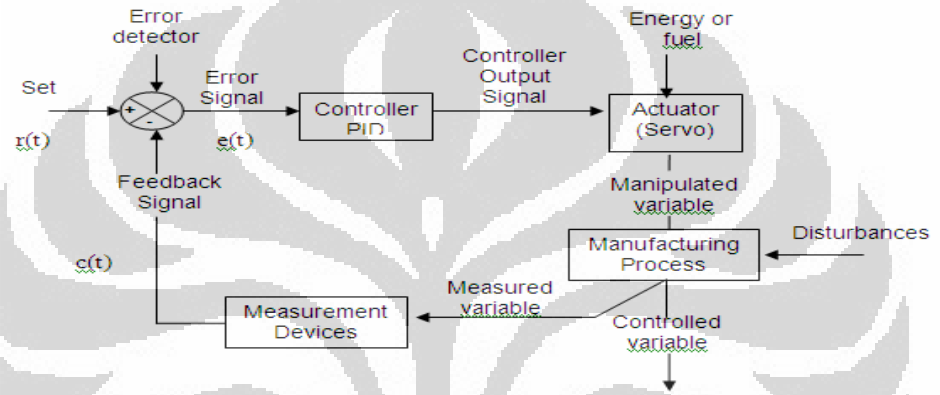
Dari potongan program diatas dapat dilihat bawa pada:

- Pada pemrograman ADC, config yang digunakan adalah single conversion
- Penentuan prescaler dilakukan dengan otomatis.

Data dari sensor, dalam hal ini adalah rotari sensor dibaca pada kanal ADC yang pertama yaitu A.0. ini dapat dilihat dari instruksi ”H = Getadc(0)”.

3.4.4 Kontrol Kendali PID

Sebelum merancang kontrol PID, harus diketahui terlebih dahulu diagram sistem dari aktuator yang akan dikontrol. Gambar 3.10 menunjukkan diagram blok dari sistem.



Gambar 3.10 Diagram Blok Kontroler

Dari Gambar 3.10 dapat dilihat bahwa pada kondisi awal motor servo akan bergerak sesuai dengan nilai *set point* yang diberian, tetapi selanjutnya nilai input dari motor servo diperoleh dari nilai error (e(t)) yang dihasilkan dari karena pergerakan yang dihasilkan motor servo itu sendiri yang diolah terlebih dahulu dengan menggunakan kontrol PID pada mikrokontroler ATMEGA 8535. nilai error (e(t)) ini diperoleh dari hasil pengurangan nilai sensor (c(t)) sebagai nilai feedback dengan nilai *set point* (r(t)), untuk lebih jelas dapat dilihat pada persamaan 3.1. Nilai c(t) diperoleh dari hasil pembacaan sensor(*measurement device*) dalam hal ini digunakan rotary sensor.

$$e(t) = r(t) - c(t) \qquad (3.3)$$

Seperti yang telah dijelaskan sebelumnya bahwa persamaan-persamaan dari kontrol PID masih dalam bentuk variabel waktu. Untuk menggunakannya ke dalam aplikasi, harus dilakukan diskretisasi sehingga persamaan-persamaan tersebut dapat diaplikasikan ke dalam mikrokontroler. Berikut ini adalah diskretisasi menggunakan metode numerik rectangular mundur (*backward rectangular*).

- Kontrol proporsional

  Diskretisasi persamaan proporsional[6]:

$$u(k) = K_p e(k) \tag{3.4}$$

- Kontrol integral

  Diskretisasi persamaan integral [6]:

$$u(k) = K_i \sum_{i=0}^{k} e(i) T_c$$

$$u(k) = K_i T_c \sum_{i=0}^{k} e(i) = K_i T_c [e(0) + e(1) + ... + e(k-1) + e(k)]$$

$$u(k-1) = K_i T_c [e(0) + e(1) + e(2) + ... + e(k-1)]$$

$$u(k) = u(k-1) + K_i T_c e(k) \tag{3.5}$$

  Tc= waktu *sampling*

  Integral adalah suatu operator matematis dalam kawasan kontinyu jika didiskretisasi maka akan menjadi sigma, yang merupakan operator matematis dalam kawasan diskret. Dimana fungsi operator sigma adalah menjumlahkan nilai ke i sampai dengan nilai ke k. Berdasarkan perhitungan diatas vairabel error (e) yang diintegralkan sehingga dalam kawasan diskret menjadi e(0)+e(1)+...+e(k-1)+e(k) atau dengan kata lain

error yang sebelumnya dijumlahkan dengan error-error yang sebelumnya hingga error yang sekarang.

- Kontrol derivatif

  Diskretisasi persamaan derivatif[6]:

$$u(k) = K_d \frac{e(k) - e(k-1)}{T_c} \tag{3.6}$$

  Tc = waktu *sampling* atau waktu cuplik(*Sampling Time*)

  Derivatif (de/dt)adalah suatu operator matematis dalam kawasan kontinyu jika didiskretisasi maka akan menjadi limit, yang merupakan operator matematis dalam kawasan diskret. Dimana fungsi dari operator limit adalah mengurangi nilai ke K dengan nilai K-1. Berdasarkan perhitungan di atas variabel error (e) yang diderivatifkan, atau dengan kata lain error yang sekarang dikurangi error sebelumnya.

  Waktu *sampling* adalah lamanya waktu yang digunakan untuk mencuplik atau men*sampling* nilai dari sensor. Nilai dari sensor ini berguna untuk mendapatkan sinyal error (error(e)=*set point*-nilai sensor). Dimana waktu *sampling* ini sangat berpengaruh pada kesensitifan sistem yang dikontrol.

Idealnya waktu *sampling* adalah 0 detik sehingga tidak ada satupun data yang tidak di *sampling*, namu hal itu tidak akan tercapai karena untuk mencapai jeda *sampling* 0 detik memerlukan frekuensi *sampling* yang dapat dihitung dengan persamaan 3.5[9].

$$f_c = \frac{1}{T_c} \tag{3.7}$$

$$f_c = \frac{1}{0} = \infty$$

Sementara f_c harus bernialai tertentu agar dapat direalisasikan. Teorema *sampling* shannon telah menjabarkan secara matematik bahwa[9]:

$$f_c \geq 2f_s \tag{3.8}$$

Dimana nilaia f_s merupakan frekuensi sinyal, namun apabila hal ini direalisasikan pada umumnya sinyal tidak dapat direkonstruksikan kembali. Sehingga beberapa ahli mengambil inisiatif untuk meningkatkan frekuensi *sampling* menjadi[10]:

$$f_c \geq 10f_s \tag{3.9}$$

Apabila ingin ditinjau pada bidang waktu, maka[10]:

$$T_c \leq 0.1T_r \tag{3.10}$$

Perpanjangan rentang waktu *sampling* mengakibatkan ketidakstabilan dan kehilanagan data. Penurunan rentang waktu *sampling* mengakibatkan perubahan antara nilai-nilai *sampling* akan menurun, dan ini mengakibatkan penurunan resolusi yang pada akhirnya menjadikan informasi ada yang hilang.

Pada program yang sebenarnya, *sampling* dilakukan dengan menggunakan perintah waitus. Dengan menggunakan perintah ini sebagai pengatur waktu *sampling* adalah pasti konstan. Akantetapi akurasi pewaktuan yang tidak akurat dapat terjadi dengan menggunakan perintah ini. Sebagai penambahan, penggunaan *interupt* pada program dapat memperlambat rutin.

Dari persamaan-persamaan yang telah didiskretisasi tadi, maka persamaan-persamaan tersebut sudah dapat diaplikasikan kedalam pemrograman dengan terlebih dahulu membuat *pseudocode* dari kontrol PID. Dibawah ini dapat kita lihat *pseudocode* dari kontrol PID.

Sebelumnya kita harus menentukan terlebih dahulu nilai *set point* dan waktu *sampling*. Serta melakukan pembacaan nilai dari sensor dengan lamanya periode sesuai waktu *sampling*.

set_point=367, Tc=250 μs ;

//jika setpoint bernilai 367 dan lamanya waktu *sampling* 250 μs

Dan untuk mendapatkan nilai dari sensor dapat dilakukan dengan pembacaan ADC, dan untuk men*sampling* nilai ADC selama 250 μs detik dapat menggunakan fitur *Timer* yang sudah ada pada AVR.

nilai_sensor=read_adc(0);

Untuk kontrol proporsional

error=set_point-nilai_sensor;
outP=Kp*error; //nilai Kp ditentukan dengan metode chien servo

Untuk kontrol integral

outI=outI+(Kp*Tc*error)/Ti; // nilai Kp ditentukan dengan metode chien servo

/*untuk menggeser nilai error integral sekarang menjadi nilai error hasil penjumlahan-penjumlahan sebelumnya untuk digunakan pada rekursi berikutnya*/

Untuk kontrol derivatif

outD=Kp*Td*(error-pref_error)/Tc; // nilai Kp ditentukan dengan metode chien servo

prev_error=error;

/*untuk menggeser nilai error derivatif sekarang menjadi sebelumnya untuk digunakan pada rekursi berikutnya*/

Sedangkan untuk kontrol PID merupakan gabungan dari ketiga kontrol diatas, dan menjadi:

outPID=outP+outI+outD;

CATATAN: nilai Kp, Ki dan Kd didapat melalui *tuning* dengan menggunakan metode yang sudah ada atau melalui metode *try and error* (coba-coba).

Dari *pseudocode* program yang telah dibuat, maka dapat dibuat program kontrol PID dengan menggunakan Bascom-AVR. Pada Lampiran 1 dapat dilihat listing program kontrol PID yang digunakan untuk empat buah motor servo.

BAB IV
PENGUJIAN HASIL DAN ANALISA

Pada bab ini akan sibahas menegenai respon sistem dari motor servo. Pada dasarnya motor servo sudah memiliki sistem close loop sendiri pada driver motor servo, sehingga sebenarnya motor servo ini sudah cukup akurat dalam penggunaannya tanpa harus ditambahkan kontrol PID terhadap motor servo. Tetapi kecepatan respon sistem dari motor servo juga sangat diperhatikan dalam pergerakannya. Semakin cepat dan stabil suatu respon sistem tersebut, maka sistem tersebut semakin baik. Pada bagian ini akan diperlihatkan bagaimana pengontrolan motor servo tanpa menggunakan kontrol PID, dengan pengontrolan motor servo dengan menggunakan kontrol PID.

4.1 Respon Sistem Lingkar Terbuka(*Open Loop*)

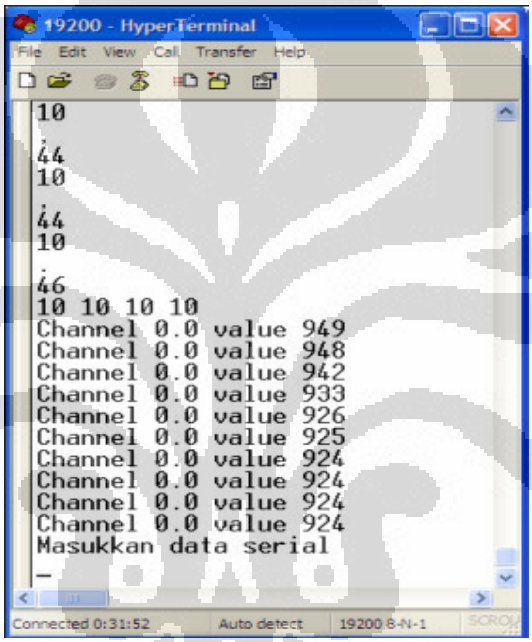
Pertama-tama percobaan dilakukan pada motor servo dengan menggunakan sistem *open loop*(lingkar terbuka) yang berarti percobaan dilakukan tanpa menggunakan kontrol PID. Percobaan ini dilakukan untuk mengetahui respon awal dari motor servo dimana dari respon yang dihasilkan dapat digunakan untuk mencari parameter-parameter PID(K_P, K_I, K_D) yang nantinya akan digunakan pada sistem kontrol *closed loop*(lingkar tertutup). Pada sistem *open loop* ini akan dilakukan percobaan dengan beberapa kondisi, diantaranya:

1. Pergerakan motor servo dengan pergerakan sudut dari -10^0 menuju ke 10^0 tanpa menggunakan beban.
2. Pergerakan motor servo dengan pergerakan sudut dari -10^0 menuju ke 10^0 dengan menggunakan beban.
3. Pergerakan motor servo dengan pergerakan sudut dari 0^0 menuju ke 2^0.

Percobaan dilakukan pada dua kondisi pergerakan sudut berbeda, yaitu: pada pergerakan sudut maksimal (-10^0 menuju 10^0) dan pergerakan sudut minimal(0^0 menuju 2^0). Hal ini dilakukan untuk mengetahui perbandingan tingkat kesalahan(error) pada kedua kondisi sudut yang berbeda tersebut.

4.1.1 Pergerakan -10^0 ke 10^0 Tidak Berbeban

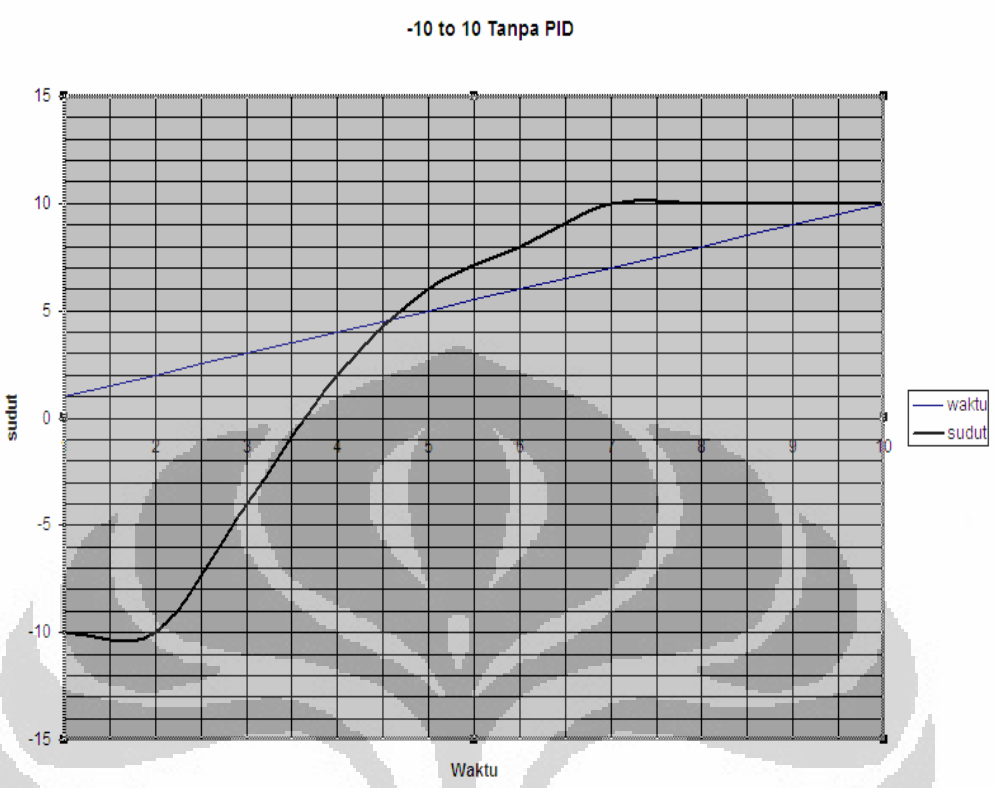
Respon sistem pergerakan motor servo tanpa menggunakan kontrol PID sangat tergantung dengan sumber supply yang digunakan pada sistem minimum kontroler. Suatu sistem akan sangat susah untuk dikendalikan apabila sumber supply untuk sistem itu kurang stabil atau banyak terdapat ripple. Untuk itu diperlukan sumber supply atau baterai yang benar-benar stabil dan sedikit mengandung riple.



Gambar 4.1 Data Pembacaan ADC Tanpa Kontrol PID Pada Servo Tanpa Beban

Tabel 4.1 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Tanpa Beban dan Tanpa Kontrol PID

Waktu	Data ADC	Sudut
1	949	-10
2	948	-10
3	942	-4
4	933	2
5	926	6
6	925	8
7	924	10
8	924	10
9	924	10
10	924	10



Gambar 4.2 Respon Sistem Servo Tanpa Beban dan Tanpa Kontrol PID

Gambar 4.1 menunjukkan data pembacaan ADC yang merupakan data pergerakan dari servo saat bergerak dari sudut -10^0 menuju ke 10^0, yang berasal dari rotary sensor. Data ADC yang diperoleh tersebut merupakan data digital, untuk itu harus terlebih dahulu di ubah ke dalam bentuk data analog. Hasil konversi dari data digital ke data analog dapat dilihat pada Tabel 4.1. Konversi data digital ke data analog dapat diperoleh dari persamaan 4.1.

$$Sudut = \frac{DataADC \times Sudutset}{ADCset} \qquad (4.1)$$

Keterangan:

ADCset : merupakan nilai pembacaan ADC untuk pergerakan n^0

Sudutset : nilai sudut untuk pergerakan n^0

Contoh:

Dari Tabel 4.1 pada contoh sampling yang pertama, nilai data ADC nya adalah 987. Maka nilai outputnya dapat dihitung dengan menggunakan persamaan 4.1.

Pembahasan:

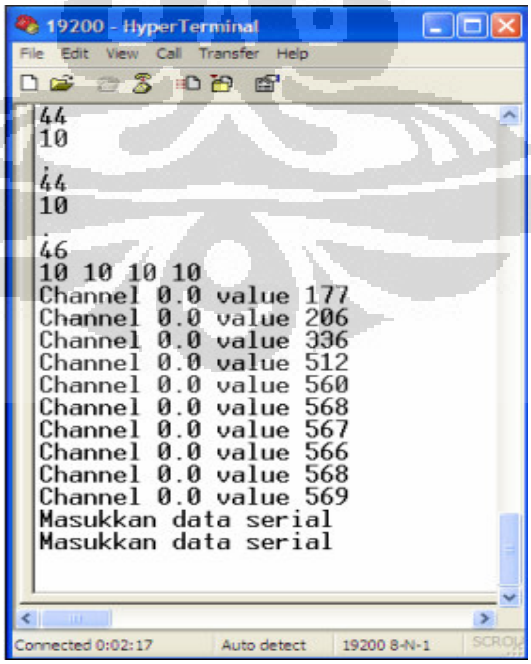
$$Sudut = \frac{987 \times -10}{987}$$

$$Sudut = -10$$

Demikian seterusnya untuk nilai data ADC yang berbeda, sampai ke sampling data yang terakhir. Apabila konversi data digital ke data analog telah selesai dilakukan sampai data yang terakhir, maka dapat digambarkan grafik dari respon sistem motor servo seperti yang diperlihatan pada Gambar 4.2.

4.1.2 Pergerakan -10^0 ke 10^0 Berbeban

Pergerakan motor servo yang berbeban akan memiliki respon sistem yang sama dengan pergerakan motor servo yang tidak berbeban, sepanjang beban yang dialami oleh motor servo masih dalam toleransi beban motor servo yang dapat digerakkan. Respon sistem akan kacau apabila beban yang digunakan melebihi kemampuan dari motor servo, karena motor servo tidak mampu menahan beban yang melebihi kemampuan motor servo itu sendiri. Gambar 4.6 menunjukkan data ADC pergerakan motor servo berbeban tanpa PID dengan sudut pergerakan -10^0 sampai ke 10^0 dengan menggunakan beban berupa air didalam botol sebanyak 600 ml.

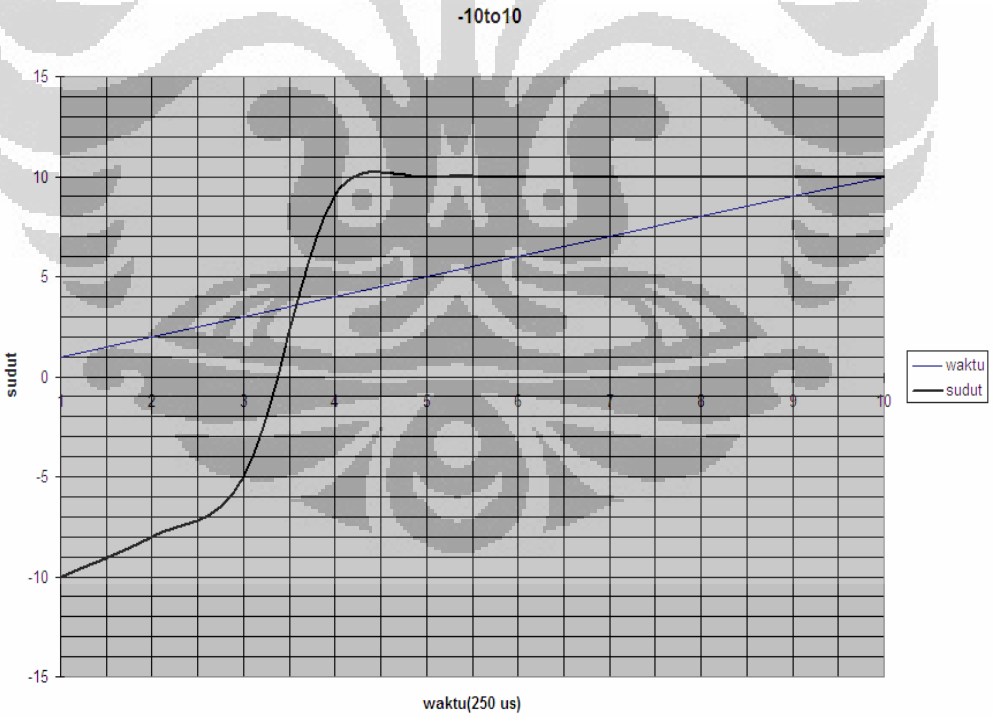


Gambar 4.3 Data Pembacaan ADC Tanpa Kontrol PID Pada Servo Berbeban(air 600 ml)

Data dari pembacaan ADC tersebut, selanjutnya diolah dengan mengubah dari data digital(data ADC) ke dalam data analog (output(v)). Tabel 4.2 merupakan hasil konversi data ADC(digital) ke data analog(V) pada motor servo tanpa beban dengan menggunakan kontrol PID.

Tabel 4.2 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Berbeban(air 600 ml) dan Tanpa Kontrol PID

Waktu	Data ADC	Output(V)
1	177	-10
2	206	-8
3	336	-5
4	512	10
5	560	10
6	568	10
7	567	10
8	566	10
9	568	10
10	569	10



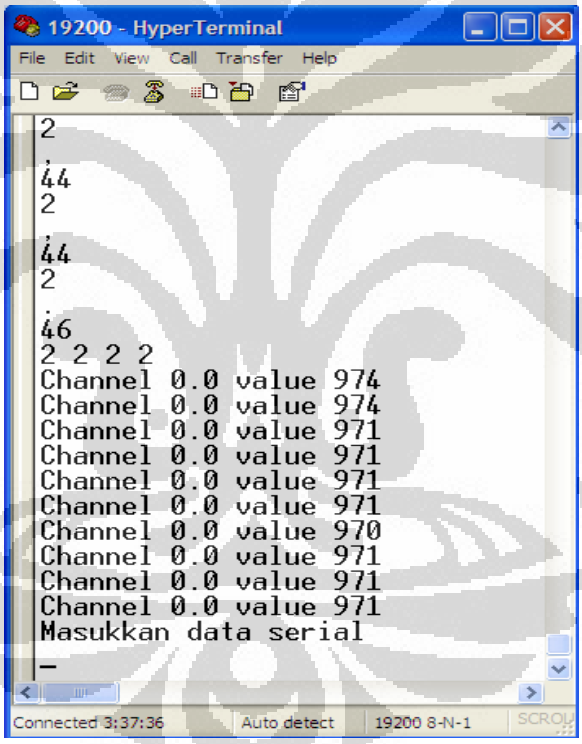
Gambar 4.4 Respon Sistem Servo Berbeban(600 ml air) dan Tanpa Kontrol PID

Dari Gambar 4.4 terlihat bahwa grafik respon sistem motor servo yang berbeban 600 ml air dan tanpa menggunakan kontrol PID, relatif sama dengan

grafik respon sistem motor servo yang tanpa menggunakan beban dan tanpa kontrol PID seperti pada Gambar 4.2. Hal ini terjadi karena torsi motor servo masih memiliki kemampuan untuk menahan beban air 600 ml tersebut.

4.1.3 Pergerakan 0^0 ke 2^0

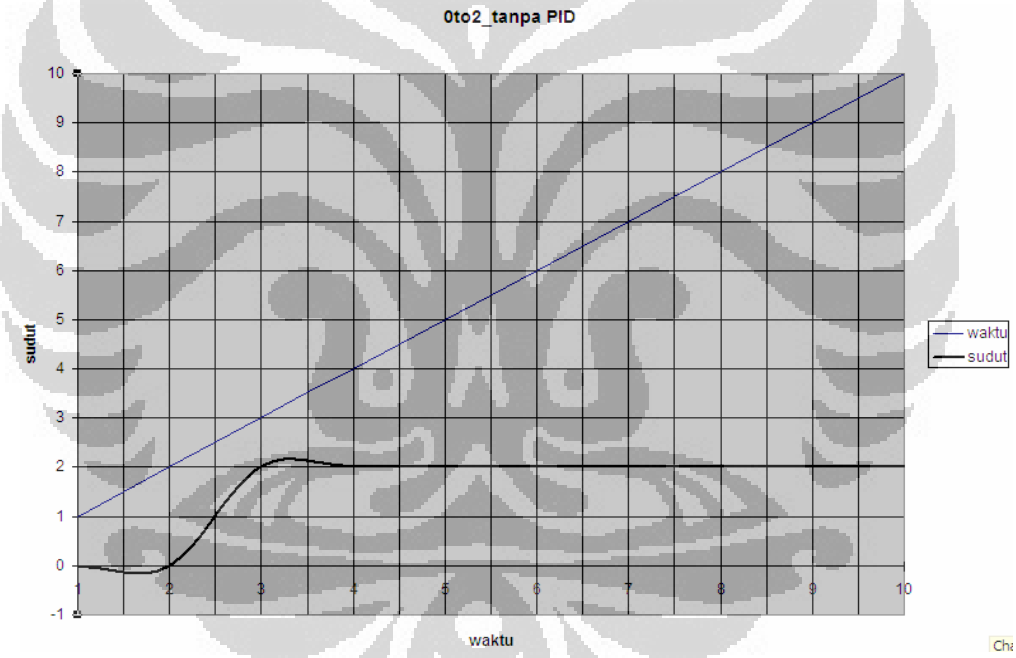
Tingkat error respon motor servo pada saat motor servo bergerak dengan sudut pergerakan yang besar akan berbeda dengan pada saat motor servo bergerak dengan sudut pergerakan yang kecil. Semakin besar sudut pergerakan motor servo, maka respon sistem motor servo tersebut akan semakin lambat respon sistemnya. Sebaliknya, semakin kecil sudut pergerakan motor servo maka respon sistem motor servo akan semakin cepat.



Gambar 4.5 Data Pembacaan ADC Tanpa Kontrol PID Pada Servo Tanpa Beban Dengan Sudut Pergerakan 0^0 menuju ke 2^0

Tabel 4.3 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Tanpa Beban dan Tanpa Kontrol PID Dengan Sudut Pergerakan 0^0 menuju ke 2^0

Waktu	Data ADC	Sudut
1	974	0
2	974	0
3	971	2
4	971	2
5	971	2
6	971	2
7	970	2
8	971	2
9	971	2
10	971	2

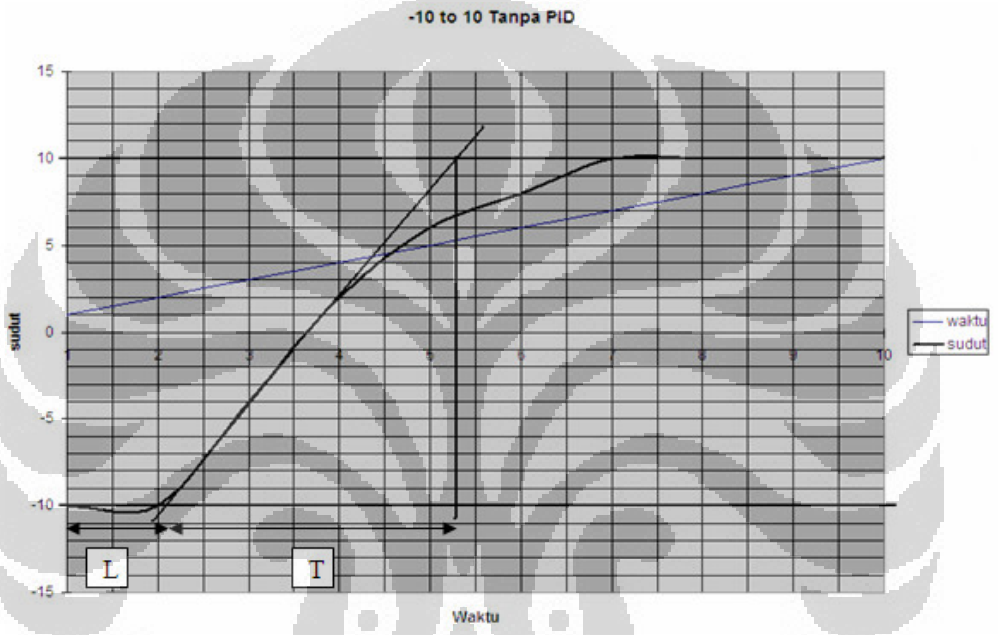


Gambar 4.6 Respon Sistem Servo Tanpa Beban dan Tanpa Kontrol PID Dengan Sudut Pergerakan 0^0 menuju ke 2^0

Dari Gambar 4.11 respon sistem dari pergerakan motor servo dengan sudut pergerakan 0^0 menuju ke 2^0 dapat dilihat bahwa error perubahan tegangan yang terjadi sangat kecil, ini dikarenakan perubahan sudut yang terjadi juga sangat kecil yaitu dari 0^0 menuju ke 2^0. Akan tetapi error tersebut akan coba diminimalkan dengan memberikan kontrol PID pada kontrol motor servo.

4.1.4 Analisa Respon Sistem Lingkar Terbuka(*Open Loop*)

Secara teori, nilai dari setiap parameter PID(K_P, K_I, K_D) dapat di prediksi dengan mengamati grafik respon sistem pada kondisi awal. Dari grafik respon sistem tersebut dapat diketahui nilai-nilai parameter yang digunakan untuk menghitung nilai parameter PID yang diingnkan, seperti nilai keterlambatan transportasi(L) dan konstanta waktu proses(T). Pada skripsi ini digunakan metode Chien Servo untuk menentukan nilai-nilai dari parameter PID tersebut.



Gambar 4.7 Parameter Keterlambatan Transportasi(L) dan Konstanta Waktu Proses(T) Respon Sistem Servo Tanpa Beban dan Tanpa Kontrol PID

Dari Gambar 4.7 dapat diketahui nilai L=284.9 µs dan nilai T=738.2 µs. Dari nilai parameter yang telah diketahui tersebut, maka dapat dihitung perkiraan nilai K_P, K_I, dan K_D dari sistem tersebut. Selain itu dalam menerapkan kontrol PID harus diperhatikan time sampling yang sesuai kontrol PID dapat menghasilkan respon sistem yang baik. Berdasarkan grafik respon sistem pergerakan motor servo pada sistem lingkaran terbuka dengan sudut pergerakan -10^0 ke 10^0 dapat ditentukan nilai time sampling yang sesuai. Nilai time sampling yang disarankan adalah:

$$T_c = 0.1 \times T_R$$

$T_c = 0.1 \times 890 \mu s$

$T_c = 89 \mu s$

- Nilai K_P

$$K = \frac{\Delta PV}{\Delta CO}$$

$$K = \frac{10 - (-10)}{948 - 923}$$

$$K = 0.8$$

$$K_P = \frac{0.6T}{0.8L}$$

$$K_P = \frac{0.6 \times 738.2}{0.8 \times 284.9}$$

$$K_P = 1.9$$

- Nilai K_I

$T_I = T$

$T_I = 738.2$

$$K_I = \frac{K_P \times T_C}{T_I}$$

$$K_I = \frac{1.6 \times 89 \mu s}{738.2 \mu s}$$

$K_I = 0.23$

- Nilai K_D

$T_D = 0.5L$

$T_D = 142.45$

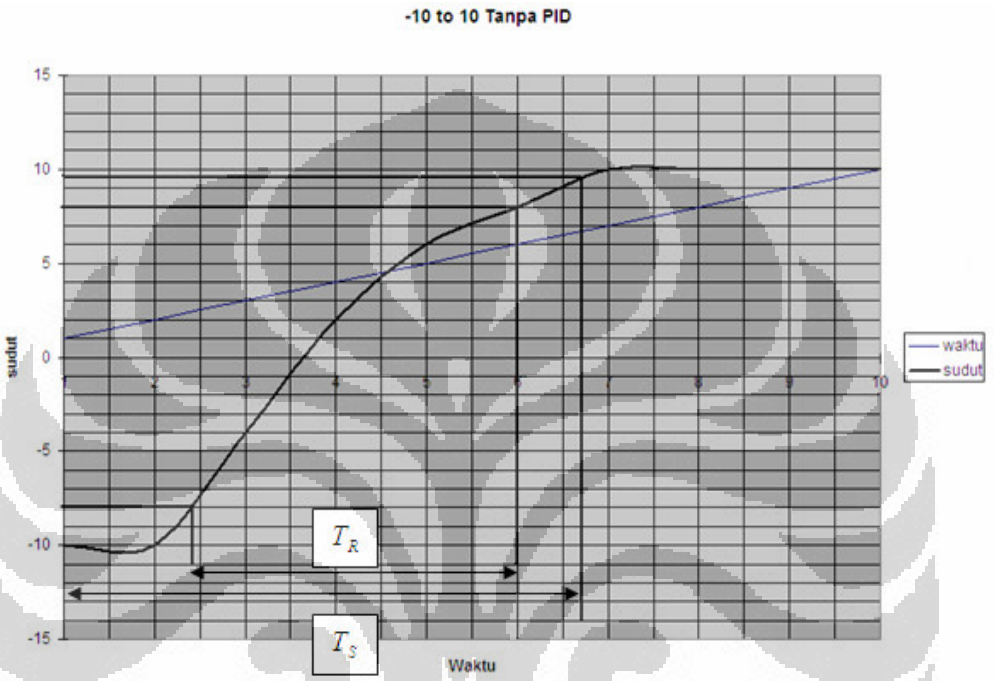
$$K_D = \frac{K_P \times T_D}{T_C}$$

$$K_D = \frac{1.9 \times 142.45 \mu s}{89 \mu s}$$

$K_D = 3$

Selanjutnya nilai parameter PID yang telah diperoleh dengan menggunakan metode chien servo ini dapat digunakan pada pengontrolan dengan PID.

Dari garafik respon sistem pada Gambar 4.2 dapat dilihat time respon yang terjadi pada pergerakan motor servo seperti pada Gambar 4.8.



Gambar 4.8 Time Respon Sistem Motor Servo

Dari Gambar 4.8 dapat dilihat nilai dari waktu naik(T_R) dan waktu perubahan(T_S) dari respon sistem motor servo. Waktu naik(T_R) adalah waktu yang diperlukan untuk bergerak dari 0.1 sampai 0.9 dari nilai akhir, sedangkan waktu perubahan(T_S) adalah waktu yang diperlukan untuk bergerak dari *set point* awal sampai 0.98 dari nilai akhirnya. Selain waktu naik(T_R) dan waktu perubahan(T_S), %Overshoot juga dapat dihitung dengan manggunakan persamaan 4.1.

$$\%Overshoot = \frac{C\max - Cfinal}{Cfinal} \times 100\% \qquad (4.2)$$

Keterangan:

Cmax = Nilai output tertinggi yang dihasilkan dari respon sistem

Cfinal = Nilai ouput yang stabil.

Dari respon sistem pergerakan motor servo pada sistem lingkaran terbuka dengan sudut pergerakan -10^0 ke 10^0, %Overshoot yang dihasilkan adalah:

$$\%Overshoot = \frac{10-10}{10} \times 100\%$$

$$\%Overshoot = 0\%$$

Tabel 4.4 Karakteristik Respon Sistem Open Loop

Gerak Servo	$T_R(\mu s)$	$T_S(\mu s)$	$\%OS$
-10to10(tanpa beban)	890	1421	0
-10to10(beban)	468.8	773.4	0
0 to 2	171.9	390	0

Dari percobaan respon sistem lingkar terbuka(*open loop*) dan dianalisa beberapa hal dari respon sistem yang dihasilkan, dapat diketahui bahwa:

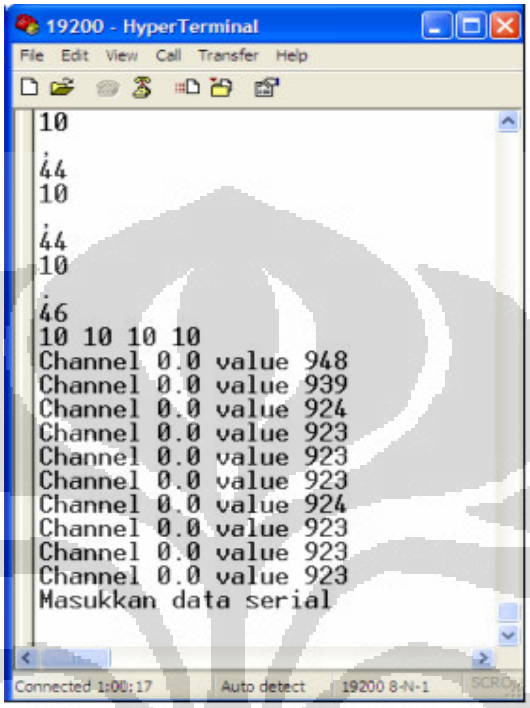
1. Dari respon sistem awal motor servo dapat ditentukan nilai parameter PID yang akan digunakan pada kontrol lingkar tertutup(closed loop).
2. Pada pergerakan servo dari -10^0 ke 10^0 dengan beban atau tanpa beban tidak jauh bebeda selama beban yang diberikan pada motor servo masih dapat diterima oleh motor servo.
3. Semakin kecil pergerakan sudut motor servo(0^0 ke 2^0) akan menghasilkan kemungkinan error yang lebih kecil dibandingkan dengan pergerakan sudut yang lebih besar(-10^0 ke 10^0). Hal ini dapat dilihat dari Tabel 4.1, Tabel 4.2, dan Tabel 4.1.
4. Time sampling yang digunakan untuk kontrol PID adalah sebesar 89 μs .

4.2 Respon Sistem Lingkar Tertutup(*Closed Loop*)

Dari hasil analisa pada respon sitem lingkar terbuka(*open loop*), diperloeh nilai paramter PID(K_P, K_I, K_D) yang akan digunakan pada sitem lingkar tertutup(*closed loop*) ini dan diharapkan dapat memberikan respon sistem yang lebih baik daripada respon sistem tanpa diberikan kontrol PID. Pada percobaan sistem lingkar tertutup(*closed loop*) ini juga dilakukan pada beberapa kondisi yang sama seperti yang dilakukan pada percobaan sistem lingkar terbuka(*open loop*).

4.2.1 Pergerakan -10^0 ke 10^0 Tidak Berbeban Dengan PID

Gambar 4.9 menunjukkan data ADC dari respon motor servo dengan menggunakan kontrol PID pada pergerakan dari -10^0 sampai ke 10^0.

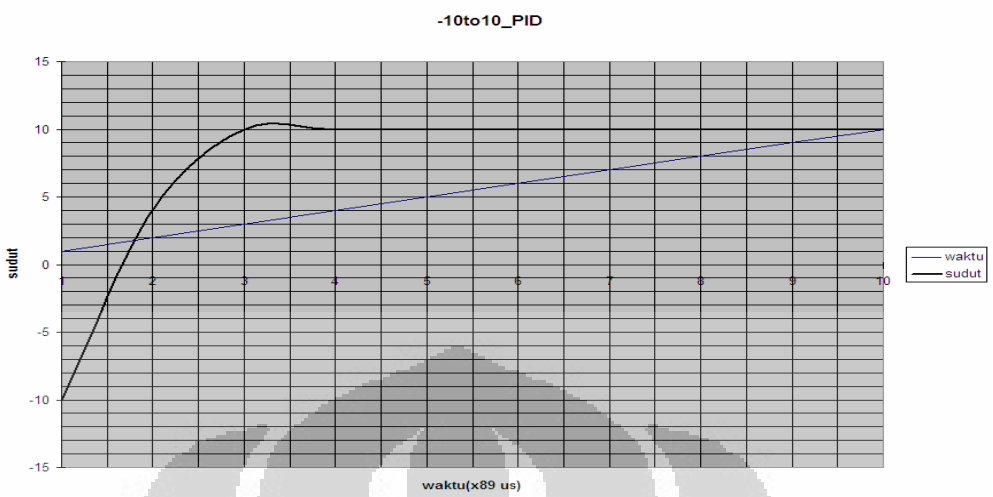


Gambar 4.9 Data Pembacaan ADC Dengan Kontrol PID Pada Servo Tanpa Beban

Data dari pembacaan ADC tersebut, selanjutnya diolah dengan mengubah dari data digital(data ADC) ke dalam data analog (output(v)). Tabel 4.5 merupakan hasil konversi data ADC(digital) ke data analog(V) pada motor servo tanpa beban dengan menggunakan kontrol PID.

Tabel 4.5 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Tanpa Beban dan Dengan Kontrol PID

Waktu	Data ADC	Sudut
1	948	-10
2	939	4
3	924	10
4	923	10
5	923	10
6	923	10
7	924	10
8	923	10
9	923	10
10	923	10

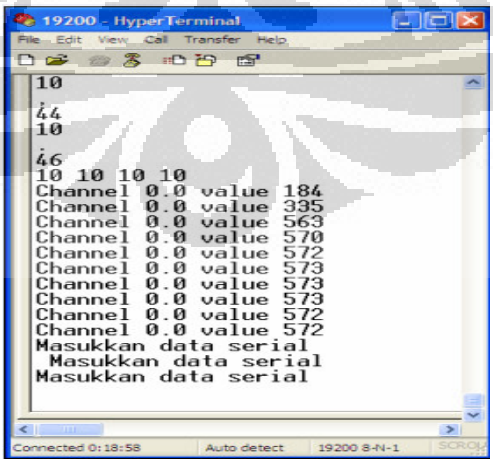


Gambar 4.10 Respon Sistem Servo Tanpa Beban dengan Kontrol PID($K_P = 1.9, K_I = 0.23, K_D = 3$)

Gambar 4.10 menunjukkan grafik respon sistem motor servo tidak berbeban dengan menggunakan kontrol PID dengan parameter $K_P = 1.9, K_I = 0.23, K_D = 3$.

4.2.2 Pergerakan -10^0 ke 10^0 Berbeban Dengan PID

Seperti pada pergerakan motor servo tanpa beban, pada pergerakan motor servo berbeban(600 ml air) juga diberikan kontrol PID dengan parameter nilai K_P=1.6, K_I=1.4, dan K_D=0.65. Maka diperolehlah data ADC dari pergerakan motor servo seperti terlihat pada Gambar 4.11.

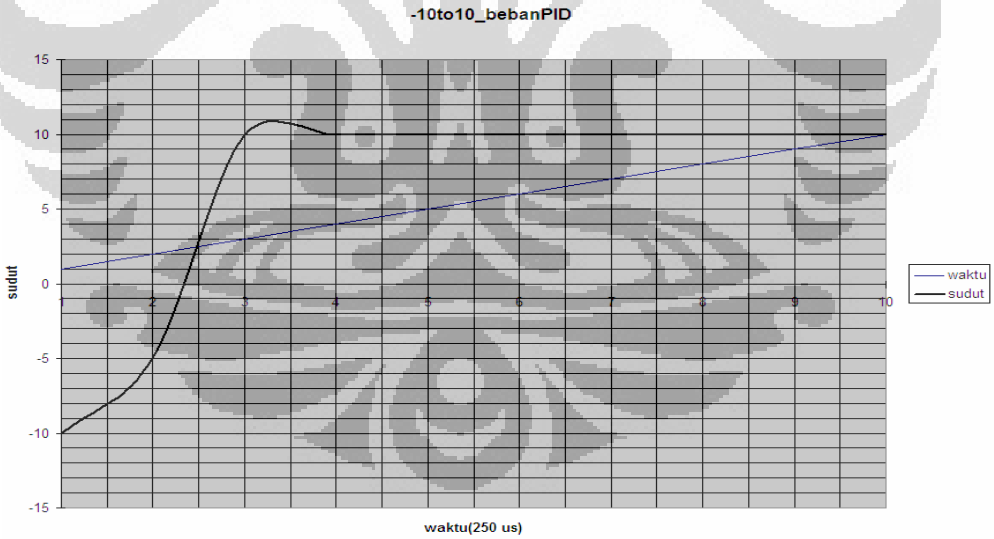


Gambar 4.11 Data Pembacaan ADC Dengan Kontrol PID Pada Servo Berbeban(600 ml air) dengan Nilai Parameter K_P=1.6, K_I=1.4, dan K_D=0.65

Data dari pembacaan ADC tersebut, selanjutnya diolah dengan mengubah dari data digital(data ADC) ke dalam data analog (output(v)). Tabel 4.6 merupakan hasil konversi data ADC(digital) ke data analog(V) pada motor servo berbeban dengan menggunakan kontrol PID, dengan nilai paramete K_P=1.6, K_I=1.4, dan K_D=0.65.

Tabel 4.6 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Berbeban dan Dengan Kontrol PID (K_P=1.6, K_I=1.4, dan K_D=0.65)

Waktu	Data ADC	Output(V)
1	184	-10
2	335	-5
3	563	10
4	570	10
5	572	10
6	573	10
7	573	10
8	573	10
9	572	10
10	572	10

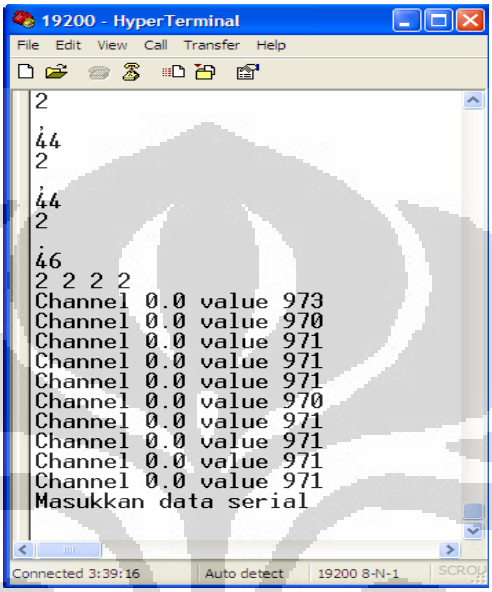


Gambar 4.12 Respon Sistem Servo Berbeban(600 ml air) Dengan Kontrol PID dengan Nilai Parameter K_P=1.6, K_I=1.4, dan K_D=0.65

Dari Gambar 4.12 dapat dilihat juga bahwa grafik respon sistem dari motor servo menggunakan beban 600 ml air dengan menggunakan kontrol PID relatif sama dengan grafik respon sistem pada motor servo tanpa beban dengan

menggunakan kontrol PID dengan Nilai Parameter K_P=1.6, K_I=1.4, dan K_D=0.65.

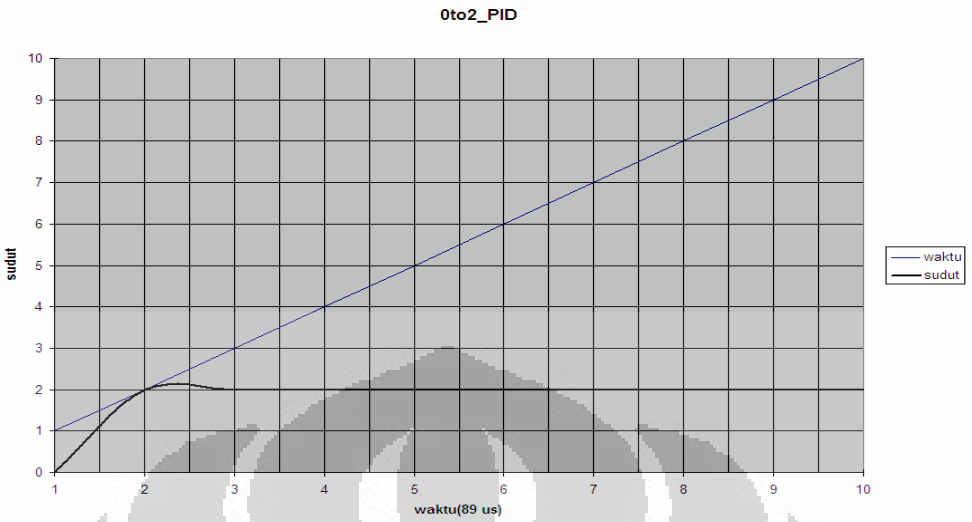
4.2.3 Pergerakan 0^0 ke 2^0 Dengan PID



Gambar 4.13 Data Pembacaan ADC Dengan Kontrol PID dengan Nilai Parameter K_P=1.6, K_I=0.52, dan K_D=0.9 Pada Servo Tanpa Beban Dengan Sudut Pergerakan 0^0 menuju ke 2^0

Tabel 4.7 Hasil Konversi Data ADC(digital) Ke Data Analog(V) Pada Servo Tanpa Beban dan Dengan Kontrol PID dengan Nilai Parameter K_P=1.9, K_I=0.23, dan K_D=3 Dengan Sudut Pergerakan 0^0 menuju ke 2^0

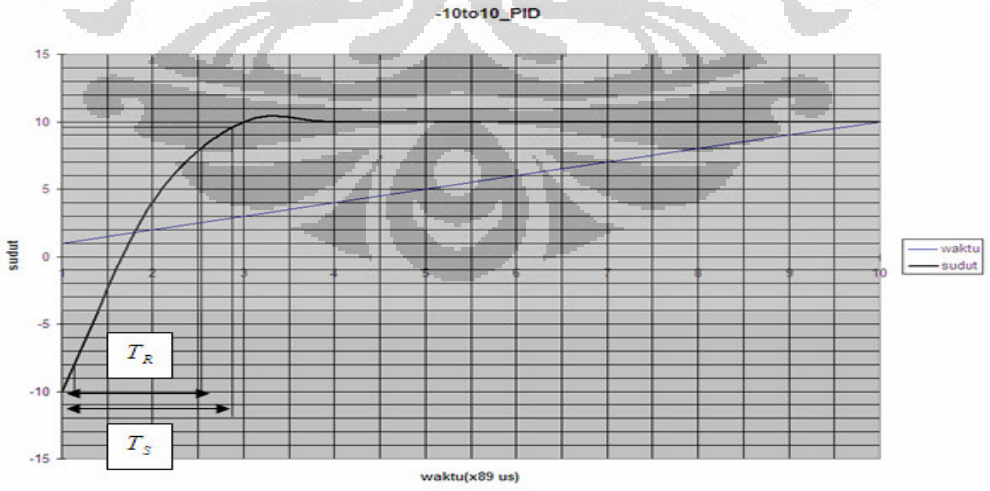
Waktu	Data ADC	Output(V)
1	936	0
2	934	2
3	934	2
4	934	2
5	934	2
6	934	2
7	934	2
8	934	2
9	934	2
10	934	2



Gambar 4.14 Respon Sistem Servo Tanpa Beban dan Dengan Kontrol PID dengan Nilai Parameter K_P=1.9, K_I=0.23, dan K_D=3 Dengan Sudut Pergerakan 0^0 menuju ke 2^0

Dari Gambar 4.14 dapat dilihat grafik respon sistem dari motor servo dengan sudut pergerakan 0^0 menuju ke 2^0 dengan menggunakan kontro PID, menghasilkan grafik respon sistem yang lebih cepat dan lebih stabil bila debandingkan dengan grafik respon sistem motor servo sudut pergerakan 0^0 menuju ke 2^0 tanpa menggunakan kontrol PID seperti pada Gambar 4.6.

4.2.4 Analisa Respon Sistem Lingkar Tertutup(*Closed Loop*)



Gambar 4.15 Time Respon Sistem Motor Servo Dengan PID

Dari garafik respon sistem pada Gambar 4.10 dapat dilihat time respon yang terjadi pada pergerakan motor servo seperti pada Gambar 4.15. Pada respon sistem motor servo lingkar tertutup dengan sudur pergerakan -10^0 sampai ke 10^0 tanpa beban dapat diketahui bahwa sistem tersebut memiliki waktu naik(T_R) sebesar 359 μs dan waktu perubahan(T_s) sebesar 484.4 μs. %Overshoot yang dihasilkan adalah:

$$\%Overshoot = \frac{10-10}{10} \times 100\%$$

$$\%Overshoot = 0\%$$

Tabel 4.8 Karakteristik Respon Sistem Closed Loop

Gerak Servo	$T_R(\mu s)$	$T_S(\mu s)$	$\%OS$
-10to10(tanpa beban)	122.4	163.2	0
-10to10(beban)	343.6	484.4	0
0 to 2	64.9	81.6	0

Setelah dilakukan percobaan respon sistem lingkar tertutup(*closed loop*) ini, dapat diketahui bahwa:

1. Nilai parameter PID yang diperoleh pada percobaan respon sistem lingkar terbuka(*open loop*) adalah benar dapat digunakan pada percobaan respon sistem lingkar tertutup(*closed loop*) dengan menghasilkan respon sistem yang lebih baik.
2. Pada percobaan respon sistem lingkar tertutup(*closed loop*) dengan menggunakan beban, respon sistemnya tidak akan terganggu selama beban yang diberikan masih dapat di tahan oleh motor servo.
3. Pada sistem lingkar tertutup motor servo, waktu naik(T_R) dan waktu perubahan(T_s) lebih kecil dari pada sistem lingkar terbuka. Ini menunjukkan bahwa pada saat sistem diberi kontrol PID, maka respon sistem tersebut akan menjadi lebih baik dari pada saat sistem tanpa diberi kontrol PID.

BAB V
PENUTUP

5.1 Kesimpulan

Berdasarkan hasil penelitian dan pembahasan, diperoleh kesimpulan sebagai berikut:

1. Motor servo dapat digunakan sebagai actuator pada sistem actuator roket kendali.
2. Dengan menggunakan kontrol PID, respon sistem motor servo menjadi lebih baik dibandingkan dengan tanpa menggunakan kontrol PID.
3. Untuk menentukan nilai parameter dari PID(K_P, K_I, K_D) digunakan dengan menggunakan metode chien servo.
4. Program yang digunakan menggunakan Bascom AVR, dengan 1 kali pulsa yang diberikan maka motor servo akan bergerak sebesar 2^0 secara bersamaan untuk 4 buah motor servo.

5.2 Saran

Saran terkait dengan hasil penelitian ini adalah sebagai berikut:

1. Diperlukan motor servo dengan torsi yang lebih besar untuk memantapkan gerak dari fin.
2. Diperlukan sebuah softaware aplikasi untuk membuat program control yang 1 kali pulsa yang diberikan dapat membuat motor servo bergerak sebesar 1^0 secara bersamaan, bukan hanya sebesar 2^0 sehingga pergerakan akan semakin teliti.

DAFTAR REFERENSI

[1] D. Sharon dan J,"Robot dan otomasi industri", PT. Elex Media Komputindo, Jakarta, 2003

[2] Gde Brata Indrawan,"robot pembersih kaca gedung dengan perangkat kamera", PA PENS-ITS, Agustus 2004

[3] Suhata, ST. "pengontrolan peralatan elektronik melalui komputer", Elex Media Komputindo, Jakarta, 2005

[4] Rezia Molfino, " Roboclimber the 3 ton spider", Research paper, automatic industrial (IAI-CSIC) of Madrid, Spain, 2004

[5] Takashi Yagi, Industrial Robot: An International Journal. Volume 29 Number 6 2002 pp. 495-499

[6] Iwan Setiawan,"Kontrol PID Untuk Industri", Elex Media Computindo, Jakarta, 2008

[7] Iswanto, ST. "Design dan Implementasi Sistem Mikrokontroller ATMega8535 dengan Bahas Basic", Gava Media, Jakarta, 2008

[8] Norman S.Nise, "Control Sytem Engineering", John Wiley and Son, United State of America, 2004

[9] Philips Charles L. And Nagle H. Troy. Thirs edition 1995. Digital Control System Analysis and Design. Mexico. Prentice Hall International.

[10] Rudy S. Wahyudi. "Pengaruh Digitalisasi Pada Sistem Kendali", volume 3, Jakarta, 2003.

[11] Anton Royanto Ahmad, "Desain Fin Control Actuator System(FCAS) Pada Roket Kendali rkx200", Final Project, Depok, 2012

Lampiran 1. Listing Program PID

```
'-----------------------------------------------------
'DEKLARASI KONFIGURASI MOTOR SERVO
'-----------------------------------------------------
Config Servos = 4 , Servo1 = Portc.0 , Servo2 = Portc.1 , Servo3 = Portc.2 ,
Servo4 = Portc.3 , Reload = 1       'Timer = Timer1
'-----------------------------------------------------
'D = G * 80
'Portb = F                                          ' kirim triger ke osiloskop
'-----------------------------------------------------
'OUTPUT SERVO
'-----------------------------------------------------
Servo(1) = C
Servo(2) = B
Servo(3) = D
Servo(4) = A
'Waitms 1

'-----------------------------------------------------
'KONFIGURASI ADC
'-----------------------------------------------------
Config Adc = Single , Prescaler = Auto

Start Adc
'-----------------------------------------------------
'ambil data sensor dari ADC
'-----------------------------------------------------
'While M <> 15

L = 0
'Input Tanda
'Print Tanda

H = Getadc(0)
'T = H
Print "Channel " ; L ; " value " ; H
Incr O
'---------------------------
'Proportional servo1
'---------------------------
L = H * A1
M = L / G                                          'M=nilai sensor
Errorp = A1 - M

If Errorp < 0 Then
Errorp = Errorp * -1
End If
```

```
Outp = Kp * Errorp
'A = Outp
'---------------------------
'Integral servo1
'---------------------------
'Errorl = Errorp + Error_sblml
'Q = Errorl * Tc
Q = Ki * Errorp
Outi = Outi + Q
'Error_sblml = Errorl
Outpi = Outp + Outi
'A = Outpi

'---------------------------
'derivatif servo1
'---------------------------
Errord = Errorp - Error_sblmd

If Errord < 0 Then
Errord = Errord * -1
End If

Outd = Kd * Errord
'Outd = R / Tc
Outpid = Outpi + Outd
A = Outpid
Error_sblmd = Errorp


Waitus 89
Wend
O = 0
'Print "A=" ; A

Loop
End
```