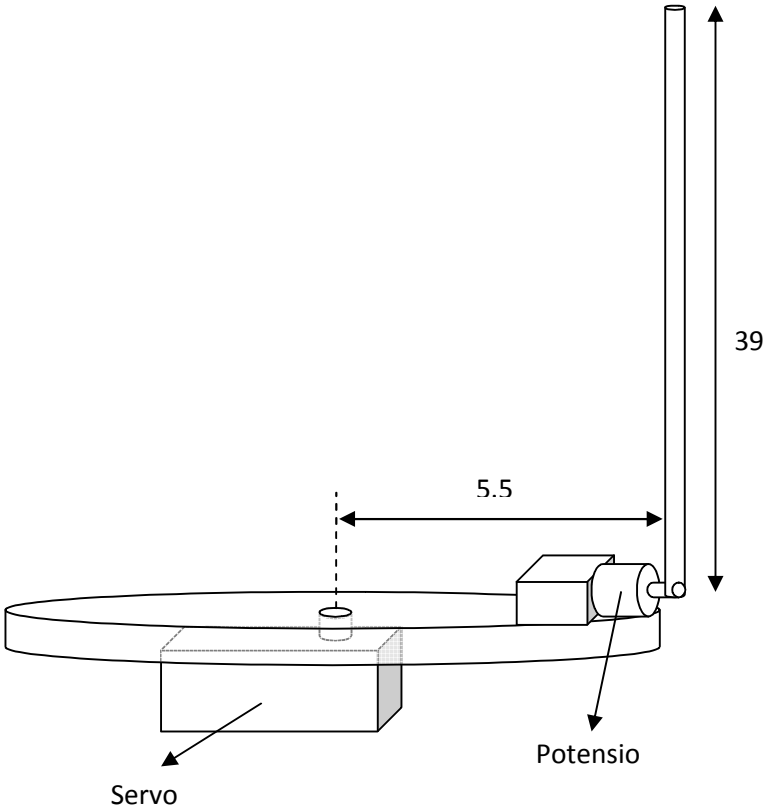
BAB III

PERANCANGAN DAN PEMBUATAN SISTEM

Pada bab ini akan membahas tentang cara perencanaan dan pembuatan perangkat keras maupun perangkat lunak yang meliputi:

3.1 Konstruksi Fisik Pendulum Terbalik

Konstruksi fisik pada pendulum terbalik didesain dengan tujuan memperoleh kestabilan. Sehingga hasil konstruksi fisiknya harus menyusaikan dengan tujuan tersebut. Maka dari itu, dibutuhkan sebuah batang stick sebagai pendulumnya, yang akan bergerak jatuh kekanan maupun kekiri hingga memperoleh titik kestabilannya atau sesuai dengan setpoint yang telah di*setting* atau berada pada posisi tengah.



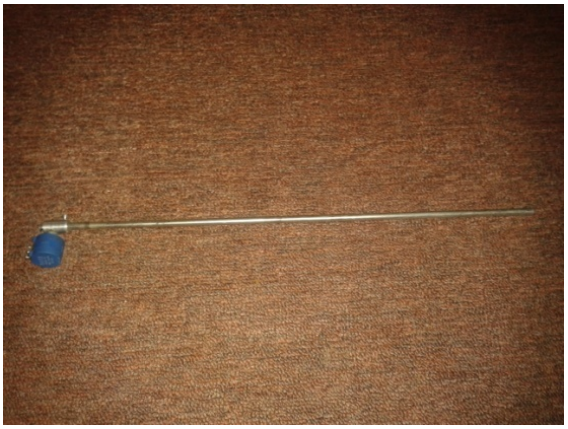
Gambar 3.1 Model rancangan pendulum terbalik



(a)

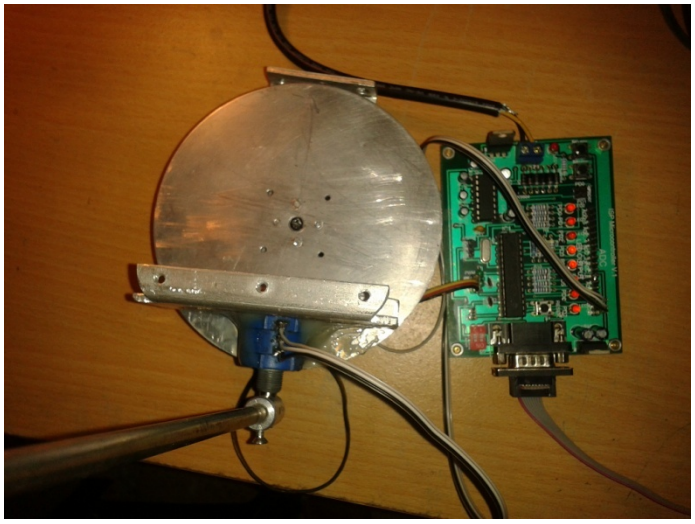


(b)



(c)

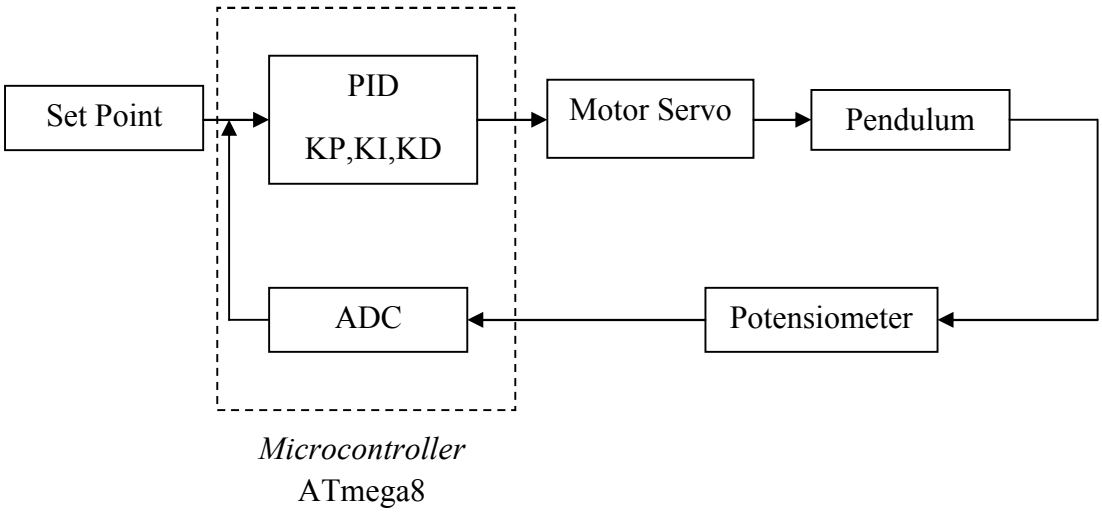
Gambar 3.2 (a) Dudukan motor servo tampak samping, (b) Dudukan motor servo tampak atas, (c) Batang stick



Gambar 3.3 Konstruksi fisik pendulum terbalik

3.2 Rangkaian Elektronika Pendulum Terbalik

Hal pertama yang akan dilakukan dalam perancangan untuk pendulum terbalik adalah membuat sistem blok diagram. Dimana blok-bloknya memiliki fungsi masing-masing. Gambar 3.4 menunjukkan suatu blok diagram sistem.

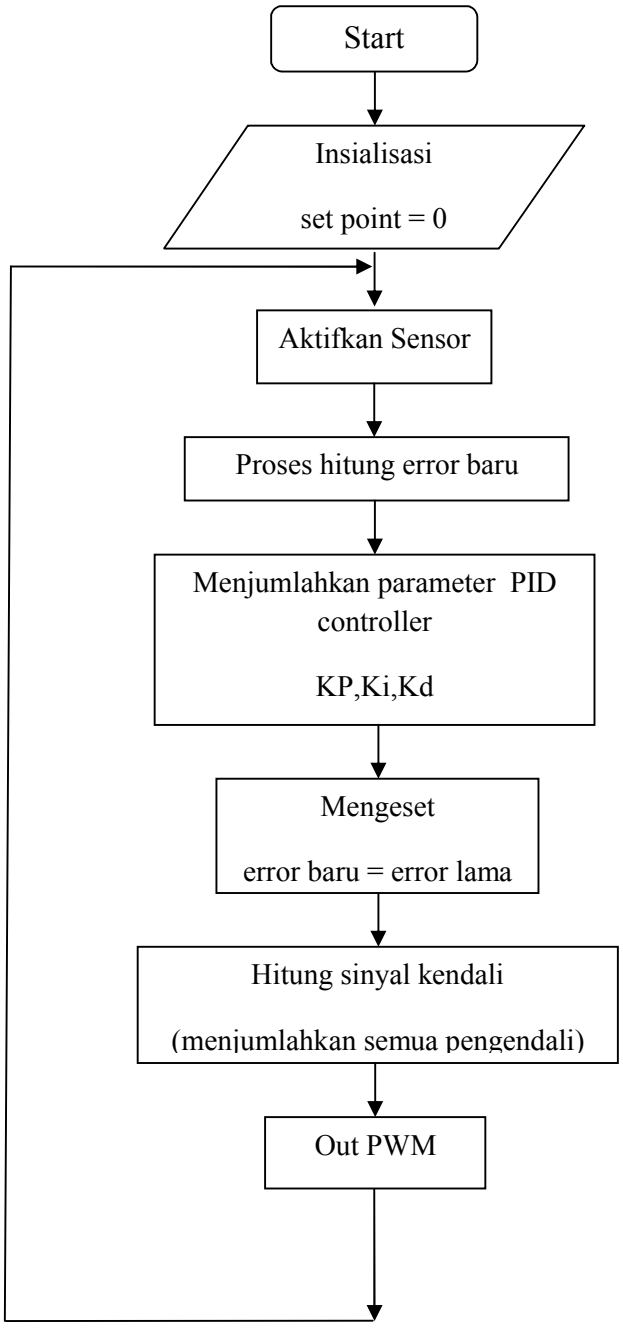


Gambar 3.4 Blok diagram sistem pendulum terbalik

Dari Gambar 3.4 diatas, dijelaskan bahwa inputannya berupa setpoint. Agar pendulum dalam keadaan seimbang setpoint yang diberikan adalah 0. Nilai θ ini kemudian akan dikalikan dengan masukan PID yaitu dengan mengeset nilai Kp, Ki dan Kd secara *trial and error*. Dan mengatur PWM agar dapat menggerakkan motor servo, karena menggunakan motor servo maka dalam tugas akhir ini tidak membutuhkan sebuah driver motor seperti pada penggunaan motor DC maupun motor stepper. Dengan *microcontroller* ATmega8, maka untuk pengaturan PWM pada motor servo dapat diatur langsung ke salah satu *port* yaitu dengan menghubungkan *input data signal* motor servo. Motor servo ini akan

menggerakkan pendulum sesuai data yang telah dieksekusi. Lalu sensor yang digunakan adalah potensiometer yang bertindak dalam mengatur posisi atau letak pendulum. Keluaran yang berupa tegangan dari potensiometer akan dihubungkan ke salah satu *port Analog to Digital Converter* (ADC) yang terletak dalam *microcontroller* ATmega8. Proses ini akan terus berlangsung sampai pendulum mencapai titik kestabilan atau keseimbangannya sesuai dengan sudut yang diinginkan.

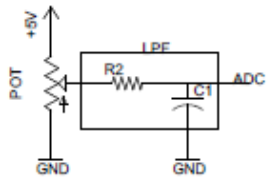
Pada gambar 3.5 menggambarkan *flow chart* dari perancangan perangkat keras untuk sistem realisasi pengendali PID pada pendulum terbalik.



Gambar 3.5 *Flowchart* system

3.3 Rangkaian Potensiometer Sebagai Sensor

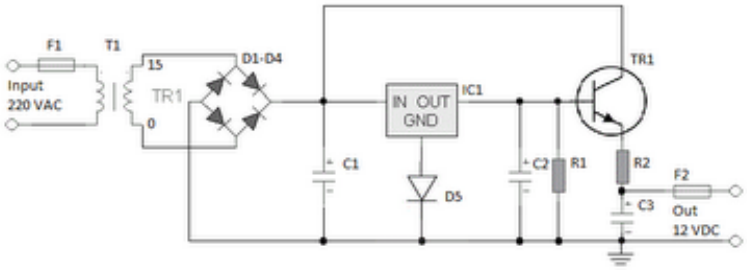
Sensor yang digunakan dalam tugas akhir ini adalah potensiometer yang bertindak sebagai sensor dalam menentukan posisi pendulum. Potensiometer memiliki tiga kaki yaitu, kaki satu dihubungkan ke tegangan 5 Volt, kaki dua dihubungkan ke ground dan kaki tiga yang dihubungkan ke pin ADC dalam mikrokontroller atau bisa dikatakan kaki ini sebagai sinyal keluaran. Karena potensiometer memiliki banyak noise maka untuk meredam noisenya maka dibutuhkan sebuah rangkaian *Low Pass Filter* (LPF) yang terdiri dari sebuah resistor 4K7 Ω yang diseri dengan sebuah kapasitor 0,1 µF. Rangkaian ini kemudian dihubungkan ke kaki sinyal keluaran potensiometer yang akan terhubung ke salah satu pin ADC pada mikrokontroller untuk dibaca tegangannya.



Gambar 3.6 Rangkaian potensiometer ke mikro

3.4 Rangkaian Adaptor

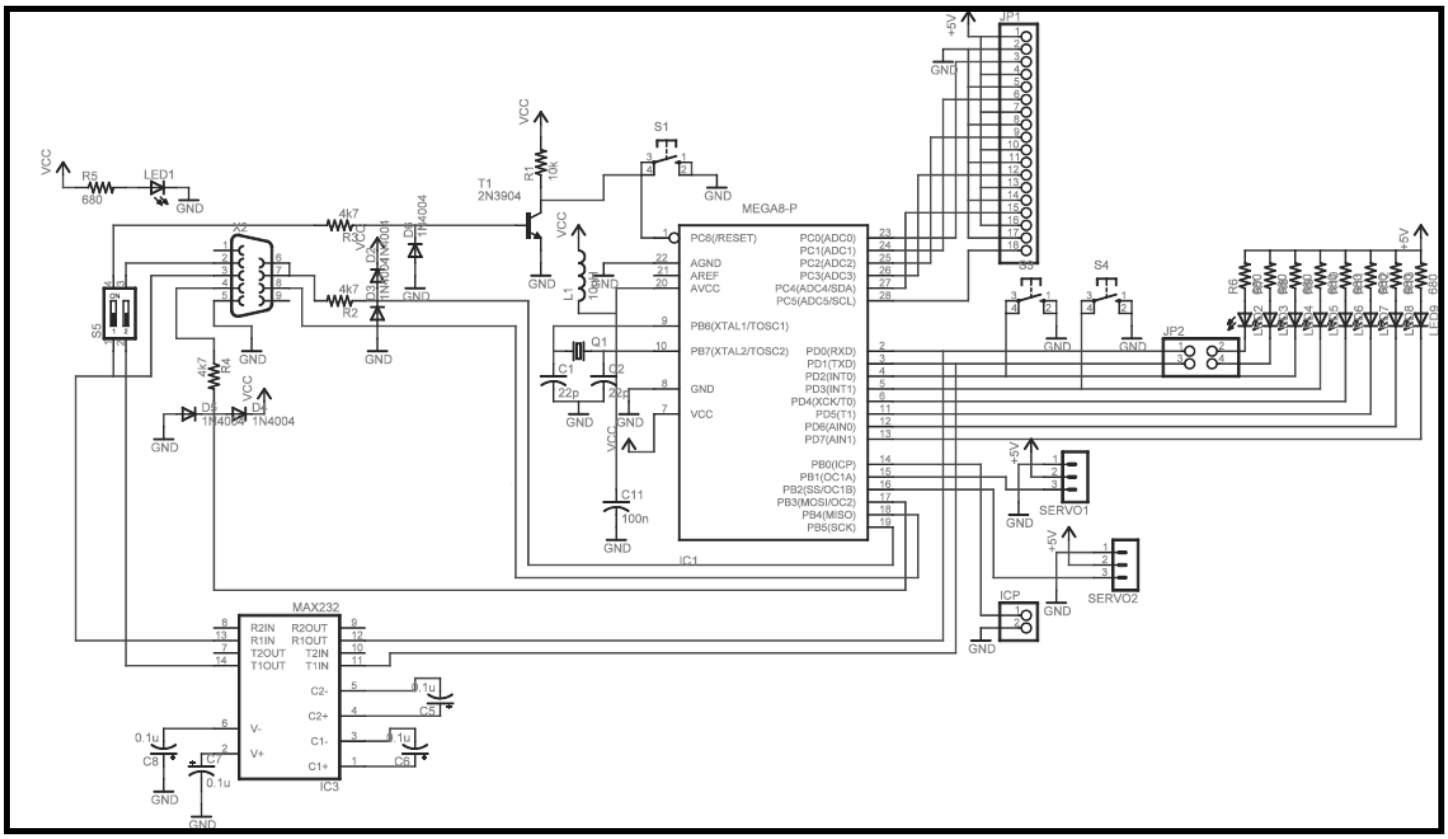
Adaptor ini memiliki keluaran sebesar 12 Volt yang teregulasi dengan baik. Jika komponen ini terpasang sesuai dengan skema rangkaian maka adaptor ini aman pada beban hingga 5A. Jika menginginkan adaptor ini mampu memberikan aliran yang lebih besar, maka dapat menambahkan transistor (2N3055) dan mengganti dioda bridge dengan dioda yang memiliki arus yang lebih tinggi.



Gambar 3.7 Rangkaian adaptor

3.5 Rangkaian Sistem Minimun, ADC Dan Motor Servo

Dalam Gambar 3.8 dibawah ini menunjukkan rangkaian sistem minimum, rangkaian motor servo dan rangkaian ADCnya. Dalam rangkaian sistem minimum disini menggunakan kristal sebesar *Crystal frequency* 7.3728 MHz. Selanjutnya terdapat 6 susunan pin-pin ADC. Setiap ADC terdiri dari 3 pin. Masing-masing pin adalah VCC, GND, dan Input ADC. Ketika membahas fitur ini, maka kita akan menggunakan potensiometer sebagai sensor analog yang tegangannya akan dikonversi ke dalam sinyal digital. Untuk menggunakan PWM disediakan 2 PORT khusus yang bernama SERVO1 dan SERVO2 sebagaimana yang terdapat pada development board. Karena dalam tugas akhir ini hanya menggunakan satu motor servo, maka bisa dihubungkan ke salah satunya saja. Motor servo ini tidak membutuhkan sebuah driver motor sehingga input data PWMnya bisa langsung dihubungkan ke PB1 untuk Servo1 atau PB2 untuk Servo2.



Gambar 3.8 Rangkaian sistem minimum, ADC dan motor servo

3.6 Program Pendulum Terbalik Pada WinAVR

```
/*****************************************
This program was produced by the
CodeWizardAVR V1.24.0 Standard
Automatic Program Generator
© Copyright 1998-2003 HP InfoTech s.r.l.
http://www.hpinfotech.ro
e-mail:office@hpinfotech.ro
Project :
Version :
Date    : 6/6/2011
Author  : Talitha Reyhan
Company :
Comments:
Chip type           : ATmega8
Program type        : Application
```

```
Clock frequency     : 7.372800 MHz
Memory model        : Small
External SRAM size  : 0
Data Stack size     : 256
*****************************************/
#include <mega8.h>
#include <stdio.h>
#include <stdlib.h>

#definetrue     1
#define false   0

#definemaxPWM       1843
#define minPWM      921
#define midPWM      1382

#define ADC_VREF_TYPE 0x40

//Variabel-variabel
float SetPoint;
float Posisi;
int PosADC;
float ErrorBaru;
float ErrorLama;
float Kp;
float Ki
float Kd;
float BagProp;
float BagInteg;
float BagDer;
float FreqSampling;
int SinyalKendali;
bit Sampling;

void SaturasiPWM(void);
void BacaPosisi(void);

//Fungsi Interrupt Timer 0 yang terjadi setiap 20 mS//
// Timer 1 overflow interrupt service routine
interrupt [TIM1_OVF] void timer1_ovf_isr(void)
{
// Place your code here
```

```
  Sampling = true;
}

//Fungsi membaca ADC//
  unsigned int read_adc(unsigned char adc_input){
  ADMUX=adc_input|ADC_VREF_TYPE;
  ADCSRA|=0x40;
  while ((ADCSRA & 0x10)==0);
  ADCSRA|=0x10;
  return ADCW;
}

void main(void)
{
Sampling = false;
FreqSampling = 50;

Kp = 50;
Ki = 0.01;
Kd = 0.1;

SinyalKendali = midPWM;

// Input/Output Ports initialization
// Port B initialization
PORTB=0x00;
DDRB=0x02;

//Inisialisasi Timer 1 untuk menjadi PWM //
TCCR1A=0x82;
TCCR1B=0x1A;
TCNT1H=0x00;
TCNT1L=0x00;
ICR1H =0x48;
ICR1L =0x00;
OCR1A =1382;  //Nilai tengah servo (1.5 mS lebar pulsa)

// Timer(s)/Counter(s) Interrupt(s) initialization
TIMSK=0x04;

//Inisialisasi Komunikasi Serial//
// USART initialization
```

```
UCSRA=0x00;
UCSRB=0x18;
UCSRC=0x86;
UBRRH=0x00;
UBRRL=0x03;

// Analog Comparator initialization
// Analog Comparator: Off
// Analog Comparator Input Capture by Timer/Counter 1: Off
// Analog Comparator Output: Off
ACSR=0x80;
SFIOR=0x00;

//Inisialisasi ADC//
ADMUX=ADC_VREF_TYPE;
ADCSRA=0x86;
SFIOR&=0xEF;
SFIOR|=0x10;

// Global enable interrupts
#asm("sei")

while (1)
      {
      if(Sampling){
      SetPoint = 516;
      BacaPosisi();
      ErrorBaru = SetPoint - Posisi;
      BagProp = Kp*ErrorBaru;
      BagInteg = Ki*(ErrorLama + ErrorBaru)*1/FreqSampling;
      BagDer = Kd*(ErrorBaru - ErrorLama)*FreqSampling;
      ErrorLama = ErrorBaru;
      SinyalKendali = SinyalKendali + (int)BagProp + (int)BagInteg +
      (int)BagDer;
      SaturasiPWM();
      OCR1A = SinyalKendali;
      printf("$%dx10\r\n",PosADC);
      Sampling = false;
       }
      };
}
//Fungsi membaca posisi pendulum
```

```
  void BacaPosisi(void){
  PosADC = read_adc(0);
  Posisi = (float)PosADC;
}

//Fungsi membatasi nilai PWM
  void SaturasiPWM(void){
  if(SinyalKendali > maxPWM) SinyalKendali = maxPWM;
  else if(SinyalKendali < minPWM) SinyalKendali = minPWM;
}
```