DEVELOPMENT OF A SELF-BALANCED ROBOT & ITS CONTROLLER

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology
in
Mechanical Engineering

BY

SOUMIT KUMAR BISWAL

DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008
2009

DEVELOPMENT OF A SELF-BALANCED ROBOT & ITS CONTROLLER

A THESIS SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS FOR THE DEGREE OF

Bachelor of Technology
in
Mechanical Engineering

BY

SOUMIT KUMAR BISWAL

Under the guidance of

Prof. Dayal R. Parhi

DEPARTMENT OF MECHANICAL ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008

2009

**NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA**

CERTIFICATE

This is to certify that the thesis entitled "**Development of self-balanced robot & its controller**" submitted by Sri Soumit Kumar Biswal in partial fulfillment of the requirements for the award of Bachelor of technology Degree in Mechanical Engineering at the National Institute of Technology, Rourkela (Deemed University) is an authentic work carried out by him under my supervision and guidance.

To the best of my knowledge, the matter embodied in the thesis has not been submitted to any other University / Institute for the award of any Degree or Diploma.

Date:
Rourkela

Prof. Dayal R. Parhi

Department of Mechanical Engineering
National Institute of Technology
Rourkela-769008

ACKNOWLEDGEMENT

I would like to express my deep sense of gratitude and respect to my supervisor Prof. Dayal R. Parhi, for his excellent guidance, suggestions and support. I consider myself extremely lucky to be able to work under the guidance of such a dynamic personality.

I would like to render heartiest thanks to my friends who's ever helping nature and suggestion has helped us to complete this present work

DATE:12-05-2009

SOUMIT KUMAR BISWAL
ROLL NO 10503025, B.TECH
DEPARTMENT OF MECHANICAL
ENGINEERING
NATIONAL INSTITUTE OF TECHNOLOGY
ROURKELA-769008

TABLE OF CONTENTS

NOMENCLATURES

The following are the variables being used in modeling the balancing robot.

Ax	Accelerometer reading along x-direction
Vg	Rate gyro reading
θ_{raw}	Raw angle obtained from rate gyro
$\theta_{gravity}$	Angle obtained from accelerometer
$\theta_{stabilized}$	Final stabilized angle
ω	Angular velocity of robot
$\acute{\omega}$	Angular acceleration of robot
M	Mass of robot
$m_{reaction\ wheel}$	Mass of reaction wheel
m_{motor}	Mass of DC motor
$L_{c.g}$	Height of CG of robot
L	Height of reaction wheel
g	Acceleration due to gravity=9.81ms^{-2}
τ	Restoring torque applied by motor
I	Current
e_g	Rate gyro error
K_1-K_4	Constants

ABSTRACT

Two wheeled balancing robots are based on inverted pendulum configuration which rely upon dynamic balancing systems for balancing and maneuvering. These robot bases provide exceptional robustness and capability due to their smaller size and power requirements. Outcome of research in this field had led to the birth of robots such as Segway, Murata boy etc. Such robots find their applications in surveillance & transportation purpose. This project is based on development of a self balanced two-wheeled robot which has a configuration similar to a bicycle. In particular, the focus is on the electro-mechanical mechanisms & control algorithms required to enable the robot to perceive and act in real time for a dynamically changing world. While these techniques are applicable to many robot applications, the construction of sensors, filters and actuator system is a learning experience.

CHAPTER 1:

INTRODUCTION

Two wheeled balancing robot is a classic engineering problem based on inverted pendulum and is much like trying to balance a broom on the tip of your finger. This challenging robotics, electronics, and controls problem is the basis of my study for this project.

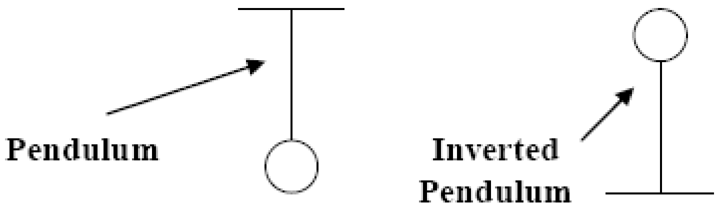


Figure 2: Inverted and non-inverted pendulum

Balancing Process

The word balance means the inverted pendulum is in equilibrium state, which its position is like standing upright 90 degrees. However, the system itself is not balance, which means it keeps falling off, away from the vertical axis. Therefore, a gyro chip is needed to provide the angle position of the inverted pendulum or robot base and input into the microcontroller, which the program in itself is a balancing algorithm. The microcontroller will then provide a type of feedback signal through PWM control to the H-bridge circuit to turn the motor clockwise or anticlockwise, thus balancing the robot.

The code is written in C software and compiled for the Atmel ATMega16 microcontroller, which is interfaced with the sensors and motors. The main goal of the microcontroller is to fuse the wheel encoder, gyroscope and accelerometer sensors to estimate the attitude of the platform and then to use this information to drive the reaction wheel in the direction to maintain an upright and balanced position.

The basic idea for a two-wheeled dynamically balancing robot is pretty simple: move the actuator in a direction to counter the direction of fall. In practice this requires two feedback sensors: a tilt or angle sensor to measure the tilt of the robot with respect to gravity, an accelerometer to calibrate the gyro thus minimizing the drift. Two terms are used to balance the robot. These are 1) the tilt angle and 2) its first derivative, the angle velocity. These two measurements are summed and fed back to the actuator which produces the counter torque required to balance the robot. The robot can be classified into the following parts:

- Inertial sensors
- Logical processing unit
- Actuator

CHAPTER 2:

LITERATURE REVIEW

This section provides an insight and literature review to the current technology available to construct a two-wheel self balancing robot. It also highlights various methods used by researches on this topic.

Balancing robots

The concept of balancing robot is based on the inverted pendulum model. This model has been widely used by researches around the world in controlling a system not only in designing wheeled robot but other types of robot as well such as legged robots. Researches at the Industrial Electronics Laboratory at the Swiss Federal Institute of Technology have built a prototype two wheel robot in which the control is based on a Digital Signal Processor. A linear state space controller using information from a gyroscope and motor encoder sensors is being implemented to make this system stabilize. (Grasser et al.2002).

Another two wheeled robot called 'SEGWAY HT' is available commercially (Dean Kamen ,2001) . It is invented by Dean Kamen who has design more than 150 systems which includes climate control systems and helicopter design. An extra feature this robot has is that it is able to balance while a user is standing on top of and navigate the terrain with it. However, this uses five gyroscopes and a few other tilt sensors to keep it balanced.

Next is the small scale robot, Nbot which is similar to JOE is built by David. P Anderson. (Anderson, David.P) This robot uses a commercially available inertial sensor and position information from motor encoder to balance the system. This robot has won the NASA cool robot of the week in the year 2003.

Steven Hassenplug used a more innovative approach to construct a balancing robot (Steve Hassenplug, 2002). The chassis of the body is constructed by using the LEGO Mindstorms robotics kit. The balancing method of controlling the system is unique with two Electro-Optical Proximity Detector sensors is used to provide the tilt angle information for the controller. This omits the conventional use of gyroscope that has been used by previous robot researchers.

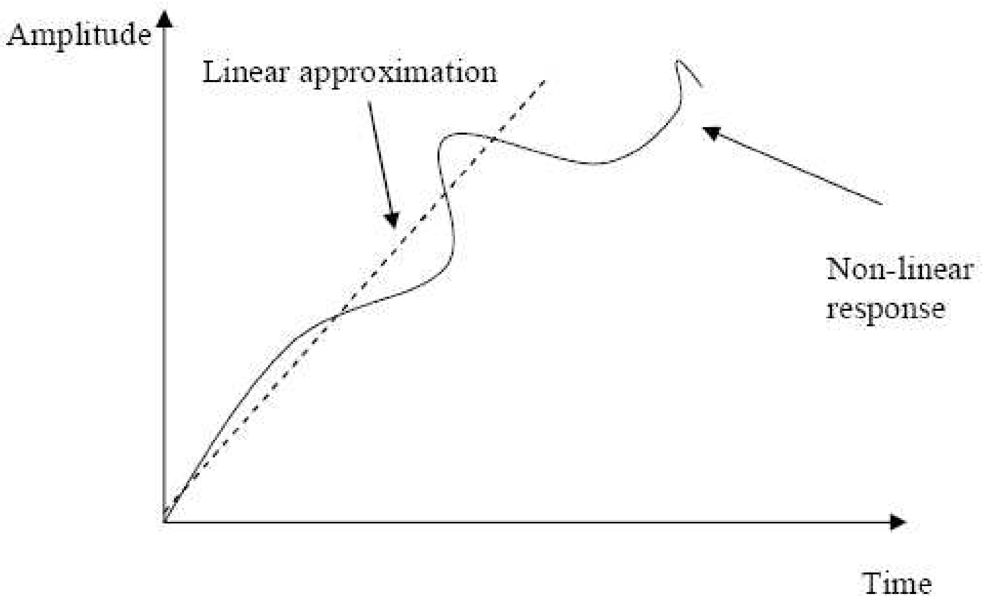
Louis Brennan, an Irish-Australian inventor, was one of the first to patent a gyroscopic stabilizing vehicle. In 1903, Brennan patented a gyroscopically balanced monorail system that he designed for military use; he successfully demonstrated the apparatus in 1909. By mounting one or more gyrostats (a modified gyroscope) along the body, the monorail balanced itself when its equilibrium was disturbed. Brennan feared that the gyrostats would fail in use, causing total system failure; thus, he prevented the monorail from being mass-produced.

More recently, a group from Columbia University manufactured a modernized version of Brennan's monorail. Unfortunately, the group was unable to create a working model. The electronic component of the model continuously overheated during operation, causing the motor to burn out. The electronic segment was improperly modeled, which led to the mechanism's inability to perform.

Control System

Over the years there are only two types of control being used by researchers in controlling a system. The types of control is categorized as linear and non-linear control. In some instances, the linear control is sufficient to control a system. One of the most widely used is the Proportional Derivative Integral controller or better known as the PID controller (Rick Bickle, 2003). The others are linear quadratic controller (LQR), fuzzy logic controller, pole placement controller etc. It is generally accepted that linear control is more popular than non-linear control. There are two reasons for this. In all cases, the modeling of a system requires a lot of parameters to be considered and applied. Therefore, the system is complex. However, some of the parameters values needed to model the system are small. That is why most researchers would prefer to model their applications in a linear approximation, which is simpler and in some instances effective.

However, in most cases the linear control theory is not suitable for real life implementation, which mostly exhibit non-linear response. For better performancesome non-linear approximation can be applied. (J.J. D'Azzo, pg 11)
Figure below shows how a non-linear response can be approximated to a linear response.



Data Acquisition

In a paper ‘Attitude Estimation Using Low Cost Accelerometer and gyroscope’ presented by Young Soo Suh, it shows the two different sensors which is the accelerometer and gyroscope that exhibits poor results when use separately to determine the attitude which is referred as the pitch angle or roll angle. The factor that contributes to the deviation of the desired result of the gyroscope is due to the drift term. Since the drift increases with time error in output data will also increase.

One of the disadvantages of using accelerometer individually is that the device is sensitive to vibration since vibration contains lot of acceleration components. One solution that Young suggested is that a low pass filter is required to limit the high frequency.

However, the gyroscope can combine with accelerometer to determine the pitch or roll angle with much better result with the use of Kalman filter.

Kalman Filter

The purpose of this filter is to solve problems of statistical nature (Kalman, 1960). Kalman filter is applied based on several mathematical equations that provides computational solution by using the least squares method. In other words, in simple explanation the process is done by averaging a sequence of values. This filter is powerful as it can estimate the past, present and future states. This filter is actually applied in state space equation. Since the program to be used is in assembly this method is not to be used. The averaging method is used instead.

CHAPTER 3:

INERTIAL SENSOR UNIT:

The inertial sensors used here are:

- GWS Piezo Rate Gyro
- Freescale 1-axis accelerometer

A gyroscope, made from a spinning wheel, is the classical method for achieving a vertical reference. Unfortunately they are large and clumsy, which is not suitable for Gyro's small design. Thanks to advances in micro-electro-mechanical systems (MEMS), the gyroscope has been reduced to an incredibly small package. By measuring this induced vibration you can tell which way it is rotating and how fast. This is known as a piezoelectric rate gyroscope and Gyro uses one to help achieve a vertical reference.

Unfortunately, these gyroscopes are not perfect. They tend to report a small rate of rotation, even at rest. The gyroscope also develops a slow creeping tilt error due to integration. Since the sensor reports an angular velocity, the integrated value should result in a position. This is not a good estimate of position though because it is only relative to when the software actually started integrating.

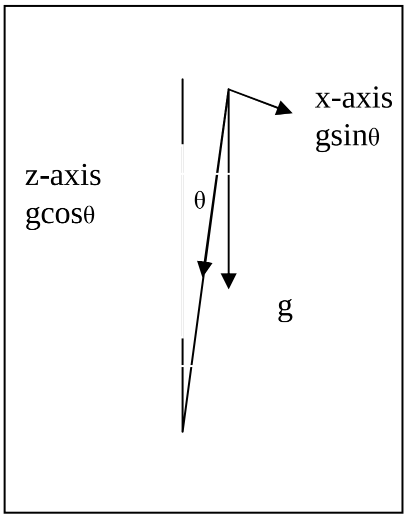
Gyro's software will use the quick reacting rate gyroscope only for a short-term reading. Software will combine this measurement with that of an accelerometer to deduce a better estimate of absolute position. The accelerometer does give a physical reference because it is able to measure the static gravitational force which allows Gyro to make accurate measurements even at rest

A GWS rate gyro and accelerometer (Freescale) were used to make the inertial measurements. The rate gyro has a single analog output for the rotation in the y axis. The gyro is mounted to match the rotation of the robot. The rate gyro measures angular velocity and outputs a voltage V_g

$V_g= \omega + f(T) + e_g$

Where f(T) represents the effect of temperature and e_g represents error, which is not known. As the rate gyro is sensitive to temperature. Since e_g is not known it cannot be subtracted from the signal and so the remaining value of angular velocity will be known as which is not guaranteed to be the same as the actual angular velocity ω.

On the other hand, the accelerometer is free from drift and errors due to integration. The angle of tilt can be calculated by measuring the acceleration along x-direction.



A_x=g sin θ

For small value of θ

A_x =g θ

θ =K_1*(A_x) ; K_1 is a constant

This approximation is only accurate for small values of θ; considering our tilt to be small, we go ahead with this approximation as inverse trigonometric functions will be time consuming for a 8-bit microcontroller.

This is the case considering the static acceleration. In case of dynamic acceleration, the equations are modified as;

$A_x = g \sin\theta - \partial v_x / \partial t$

Considering small angles;

$\theta = K_1 * (A_x + r\acute{\omega})$

For calculating the integrals and derivatives, the PID algorithm is used

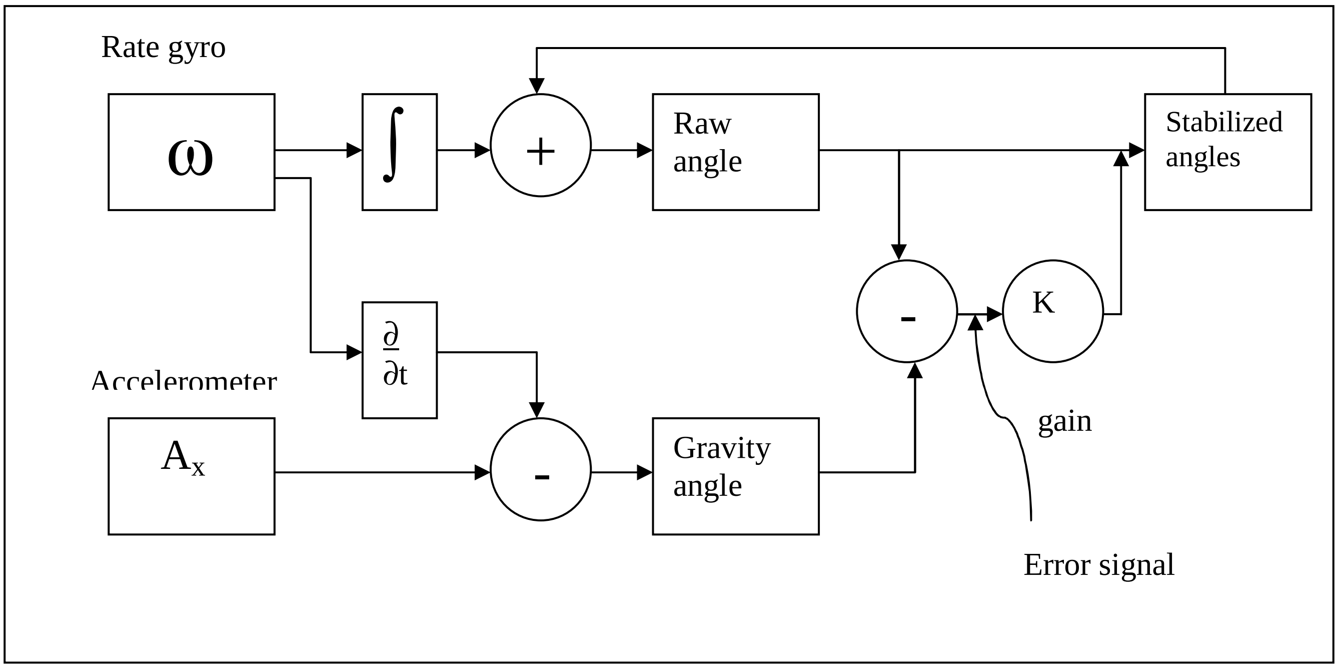
Calculation of θ_t by integration;

$\theta_t = \theta_{t-1} + \omega * \delta t$

Calculation of $\acute{\omega}_t$ by differentiation;

$\acute{\omega}_t = (\omega_t - \omega_{t-1}) / \delta t$

Schematic of the sensor module.



The outputs of the sensor modules are displayed on the results page & complete specifications are given in appendices.

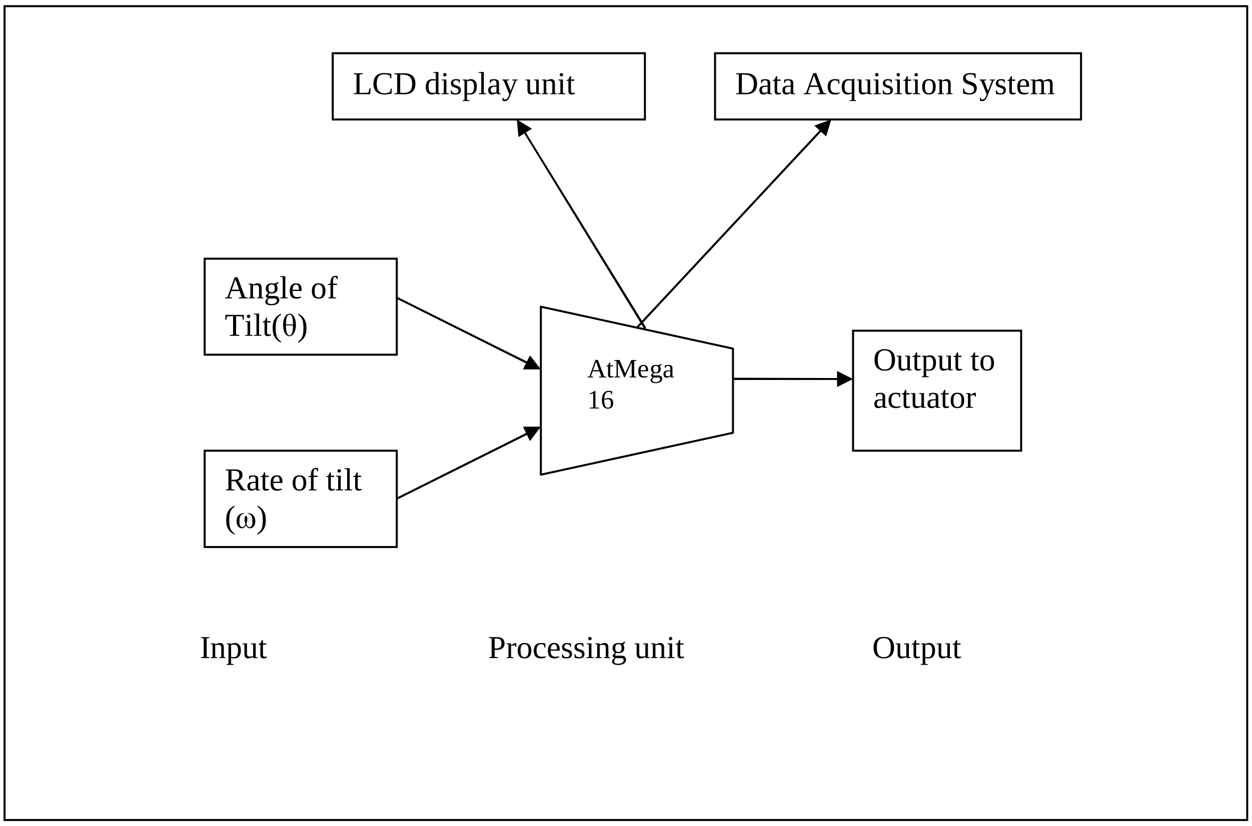
CHAPTER 4:

LOGICAL PROCESSING UNIT:

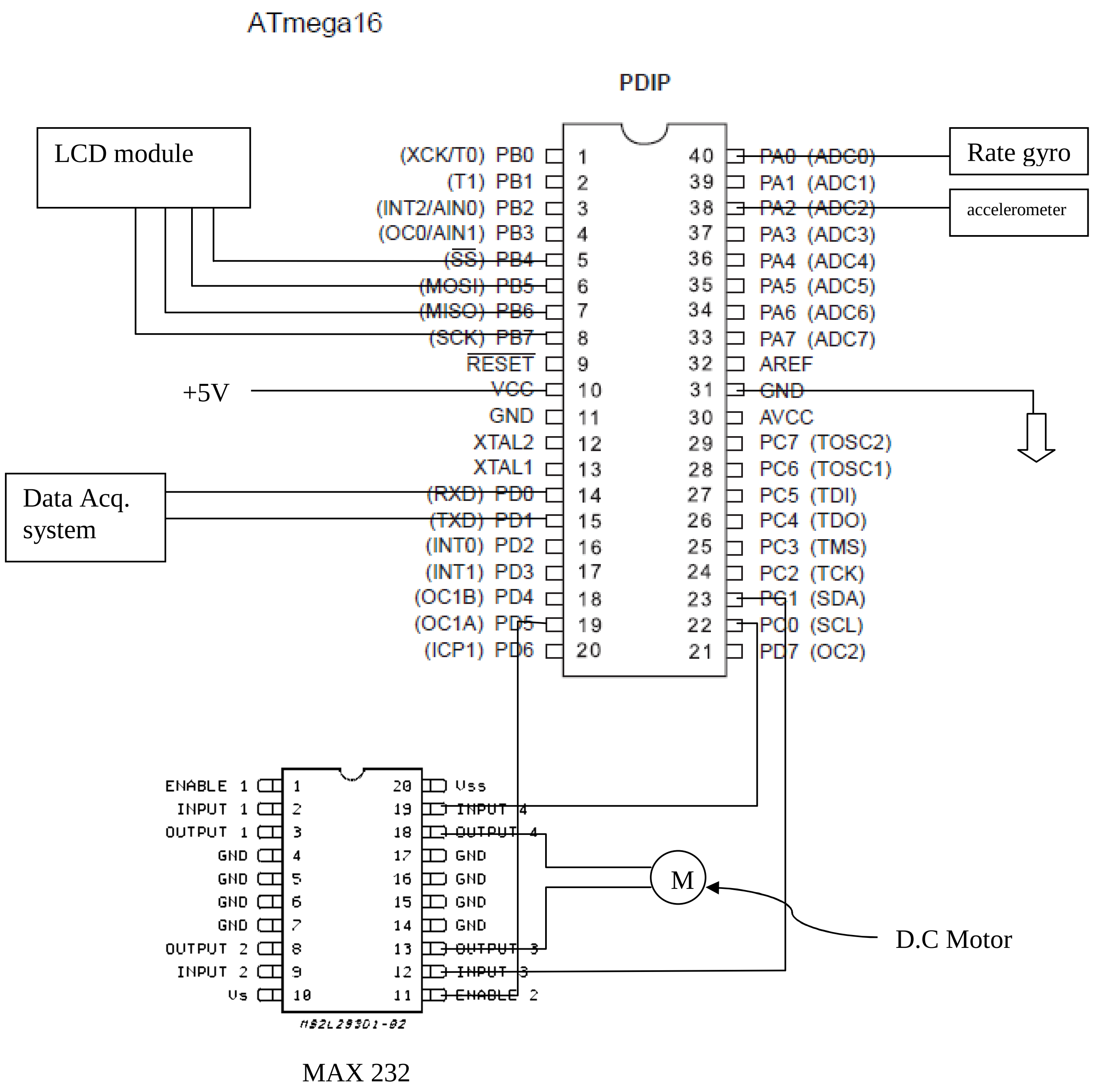
The processing unit used is Atmel ATMega16,8-bit microcontroller unit which is a versatile EEPROM. It has four I/O ports, onboard ADC and two PWM outputs. It can be programmed easily with minimum hardware requirements which make it extremely popular in robotics applications. Here it performs the following functions:

- ADC conversion of outputs of Rate Gyro and Accelerometer
- Processing the input signals
- Periodic recalibration of gyro
- Display of angle & other data.
- Control of actuator unit

Schematic of the processing unit is as shown:



The circuit diagram is as shown:



MAX 232

The microcontroller is clocked at 12MHz frequency by connecting a crystal across pins 12-13.The complete specification is mentioned in appendix-3

ALGORITHM

The algorithm for the controller is as follows:

Step1:

Initialize the values of rate gyro bias and accelerometer bias for zero value of θ & ω.

Step2:

Measure the value of voltage of gyro and accelerometer outputs and store those values as ω and A_x.

Step3:

Integrate the rate gyro reading by: $\theta_t = \theta_{t-1} + \omega * \delta t$

Differentiate the rate gyro reading by: $\acute{\omega}_t = (\omega_t - \omega_{t-1}) / \delta t$

δt in each case is assumed to be constant and is equal to the cycle time.

Step4:

Calculate the raw angle and gravity angle and compute the error value.

Step5:

Update the raw angle by:

$\theta_{\text{stabilized}} = \theta_{\text{raw angle}} + K_2 (\theta_{\text{gravity angle}} - \theta_{\text{raw angle}})$

K_2 is taken as .2

Step6:

Calculate the torque required to restore:

$\tau = K_3 * \sin\theta$

for small angles:

$\tau = K_3 * \theta$

Step7:

Obtain the direction of rotation of the reaction wheel:

If $\theta < 0$; counterclockwise

If $\theta > 0$; clockwise

Step8:

Send the output to the DC motor attached to the reaction wheel.

Step9:

Repeat steps (2-8)

Details of the hardware:

Operating system:Microsoft XP service Pack 2

Programming platform:WinAVR,Atmel Studio

Programmer:AVRdude.

Microcontroller:ATMega16L,40 PIN DIP

Code in C language:

```
/*-----------------------------------------------------------------
 Description:
 This example runs lcd in 4bit mode using only 7 I\O pins of AVR.

 Note:

 Change following parameters as per requirement in lcd.h file.

 Default values are as below.

 #define XTAL 1000000

 #define LCD_CONTROLLER_KS0073 0

 #define LCD_PORT         PORTB       port for the LCD lines
 #define LCD_DATA0_PORT   LCD_PORT      port for 4bit data bit 0
 #define LCD_DATA1_PORT   LCD_PORT      port for 4bit data bit 1
 #define LCD_DATA2_PORT   LCD_PORT      port for 4bit data bit 2
 #define LCD_DATA3_PORT   LCD_PORT      port for 4bit data bit 3
 #define LCD_DATA0_PIN    4           pin for 4bit data bit 0
 #define LCD_DATA1_PIN    5           pin for 4bit data bit 1
 #define LCD_DATA2_PIN    6           pin for 4bit data bit 2
 #define LCD_DATA3_PIN    7            pin for 4bit data bit 3
 #define LCD_RS_PORT      LCD_PORT      port for RS line
 #define LCD_RS_PIN       0            pin  for RS line
 #define LCD_RW_PORT      LCD_PORT      port for RW line
 #define LCD_RW_PIN       1            pin  for RW line
 #define LCD_E_PORT       LCD_PORT      port for Enable line
 #define LCD_E_PIN        3            pin  for Enable line

 Connect 7 I\O pins as shown before with corresponding lcd pins.

 _delay_ms() function parameter passed should not exeed
 262.14ms / F_CPU in mhz.

 so for 16MHz the maximum paramer should be 16ms.

 For 1MHz it can be upto 262ms (250 is used for this example).

 Must be changed to get accurate delay at higher MHz.

 The timing will differ if the operating frequency is changed.

---------------------------------------------------------------------*/

/*-----------------------------------------------------------------
-----------------HEADER FILES-------------------------------------
-----------------------------------------------------------------*/

#include <stdlib.h>
#include<math.h>
#include <avr/io.h>
#include <avr/interrupt.h>
#include <avr/pgmspace.h>
#include <compat/deprecated.h>          //HEADER FILE FOR FUNCTIONS LIKE SBI AND CBI
#include <util/delay.h>                 //HEADER FILE FOR DELAY

#include "C:\Documents and Settings\richi\Desktop\LCD\lcd.h"
#include "C:\Documents and Settings\richi\Desktop\LCD\lcd.c"
#define F_CPU 12000000
#define OC1_DDR DDRD            // OC1 DDR
#define OC1A_PIN PD5            // OC1A pin
#define OC1B_PIN PD6            // OC1B pin
/*-----------------------------------------------------------------
-----------------CONSTANTS----------------------------------------
-----------------------------------------------------------------*/

static const PROGMEM unsigned char copyRightChar[] =
{
      0x07, 0x08, 0x13, 0x14, 0x14, 0x13, 0x08, 0x07,
      0x00, 0x10, 0x08, 0x08, 0x08, 0x08, 0x10, 0x00
};

/*-----------------------------------------------------------------
-----------------MAIN PROGRAM-------------------------------------
-----------------------------------------------------------------*/
```

```
//This function is used to initialize the USART
//at a given UBRR value
void USARTInit(uint16_t ubrr_value)
{

  //Set Baud rate

  UBRRL = ubrr_value;
  UBRRH = (ubrr_value>>8);

  /*Set Frame Format


  >> Asynchronous mode
  >> No Parity
  >> 1 StopBit

  >> char size 8

  */

  UCSRC=(1<<URSEL)|(3<<UCSZ0);


  //Enable The receiver and transmitter

  UCSRB=(1<<RXEN)|(1<<TXEN);


}


//This function is used to read the available data
//from USART. This function will wait untill data is
//available.

char USARTReadChar()
{
  //Wait untill a data is available

  while(!(UCSRA & (1<<RXC)))
  {
    //Do nothing
  }

  //Now USART has got data from host
  //and is available is buffer

  return UDR;
}


//This fuction writes the given "data" to
//the USART which then transmit it via TX line
void USARTWriteChar(char data)
```

```
{
  //Wait untill the transmitter is ready

  while(!(UCSRA & (1<<UDRE)))
  {
    //Do nothing
  }

  //Now write the data to USART buffer

  UDR=data;
}


void InitPWM()
{
OC1_DDR |= _BV(PD5);                              // set OC1A pin as output, required for output
toggling

   TCCR1A =  _BV(WGM11) |_BV(COM1A1);             // enable 8 bit PWM, select inverted PWM
          // timer1 running on 1/8 MCU clock with clear timer/counter1 on Compare Match
   // PWM frequency will be MCU clock / 8 / 512, e.g. with 1Mhz Crystal 244 Hz.
   TCCR1B = _BV(CS11) | _BV(WGM12) | _BV(WGM13);
          ICR1=2499;//TOP


}
int main(void)
{
     void move(int a)
          {
          PORTD=0x01;
          _delay_us(a);
          PORTD=0x00;
          _delay_ms(20);
          }


     DDRB=0XFB;                                //SET DATA DIRECTION REGISTER
                                             //SET 1 for OUTPUT PORT
                                             //SET 0 FOR INPUT PORT
                                             //PB.2 IS  INPUT
                                             //ALL OTHERS ARE OUTPUT

     DDRC=0xFF;                                //SET DATA DIRECTION REGISTER
                                             //SET 1 for OUTPUT PORT
                                             //SET 0 FOR INPUT PORT
                                             //PD.1, PD.2 AND PD.3 ARE INPUT
                                             //ALL OTHERS ARE OUTPUT

     sbi(PORTB,2);                                        //ENABLE PULL UP FOR SWITCH INT2
          cbi(PIND,7);
     cbi(PIND,6);

                              //ENABLE PULL DOWN FOR SWITCH
   lcd_init(LCD_DISP_ON);                              /* initialize display, cursor off */

void InitADC()
{
ADMUX=(1<<REFS0);                         // For Aref=AVcc;
ADCSRA=(1<<ADEN)|(1<<ADPS2)|(1<<ADPS1)|(1<<ADPS0); //Rrescalar div factor =128
}

uint16_t ReadADC(uint8_t ch)
{
            //Select ADC Channel ch must be 0-7
   ch=ch&0b00000111;
   ADMUX &= 0b11111000; // clear bottom 3 bits
   ADMUX|=ch;  // set to new value

  //Start Single conversion
  ADCSRA|=(1<<ADSC);

  //Wait for conversion to complete
  while(!(ADCSRA & (1<<ADIF)));

  //Clear ADIF

  ADCSRA|=(1<<ADIF);

  return(ADC);
}

  int adc_resultg,adc_resulta,del,dif_g;
  float x,z,y=0;
  char buffer_g[10];
  char buffer_a[10],data;
  unsigned char cnt=0,cnt2=0;
  uint8_t servo=162;


  //initialize USART
  USARTInit(38);
  //Initialize LCD
  lcd_clrscr();

  //Initialize ADC
  InitADC();
  //Initialize PWM
  InitPWM();
  //-Wl,-u,vfprintf -lprintf_flt -lm  (add those to atmelstudio,else float wont work!)
            //collect the constants
            lcd_gotoxy(0,0);
            lcd_puts("place the gyro still & press int2");
            while(bit_is_set(PINB,2));
            adc_resultg=ReadADC(0);
            itoa(adc_resultg , buffer_g, 10);
            lcd_gotoxy(0,1);
```

```
        lcd_puts("gyro_const=");
        lcd_puts(buffer_g);
        int gyro_const=adc_resultg;
        _delay_ms(16);
        sbi(PORTB,2);

        lcd_clrscr();
        lcd_gotoxy(0,0);
        lcd_puts("place the acc vertical & press int2");
        while(bit_is_set(PINB,2));
        adc_resulta=ReadADC(2);
        itoa(adc_resulta , buffer_g, 10);
        lcd_gotoxy(0,1);
        lcd_puts("acc_const=");
        lcd_puts(buffer_g);
        _delay_ms(16);
        _delay_ms(16);
        _delay_ms(16);
        int acc_const=adc_resulta;
        lcd_clrscr();
        int swp=162;
        void send(int x)
                {
                char buffer[10];
                itoa(x , buffer, 10);
                for(int i=0;i<strlen(buffer);i++)
                USARTWriteChar(buffer[i]);
                USARTWriteChar(';');
                }
        void sendf(float x)
                {
                char buffer[10];
                sprintf(buffer, "%3.6f", x);
                for(int i=0;i<strlen(buffer);i++)
                USARTWriteChar(buffer[i]);
                USARTWriteChar(';');
                }

        void ckwise()
        {
        sbi(PORTC,0);
        cbi(PORTC,1);
        }

        void cckwise()
        {
        sbi(PORTC,1);
        cbi(PORTC,0);
        }



while(1)
{
```

```
            adc_resultg=ReadADC(0);          // Read Analog value from channel-0
            adc_resulta=ReadADC(2);          // Read Analog value from channel-2
           x=.7*(adc_resulta-acc_const);
            sendf(x);
            dif_g=gyro_const-adc_resultg;
            //if((dif_g<-1) || (dif_g>1))
            y+=(dif_g)*.036;
            sendf(y);


            if(y<0)
            {
            ckwise();
            if(ReadADC(0)<adc_resultg) OCR1A=(x[2]-.001)*(-2400);
            else OCR1A=x[2]*(-2400);
            }
            else
            {
            cckwise();
            if(ReadADC(0)>adc_resultg) OCR1A=(x[2]+.001)*2400;
            else OCR1A=x[2]*2400;
            }
            send(OCR1A);
            USARTWriteChar('\r')


  }


}
```

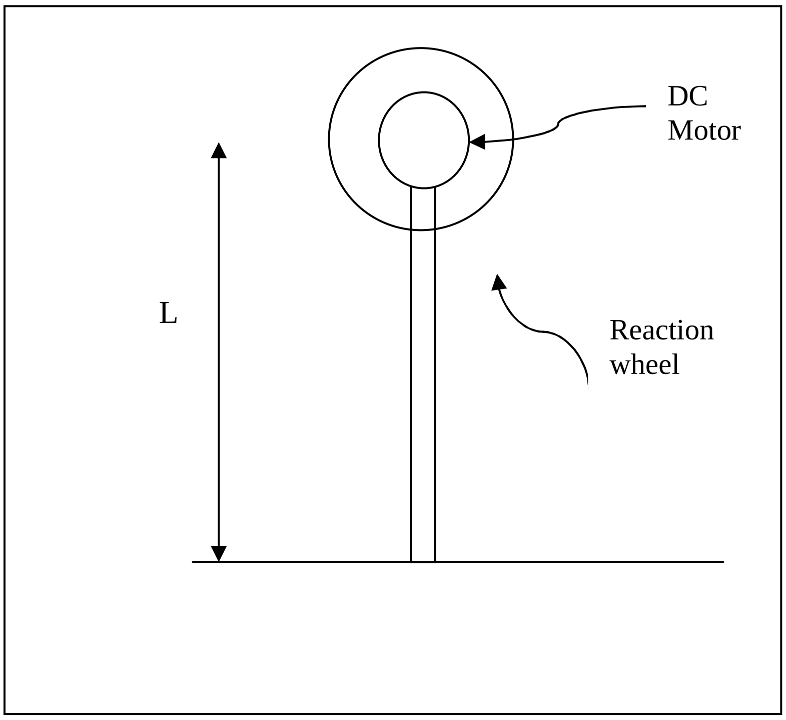
CHAPTER 5:

ACTUATOR UNIT

As the robot tilts, we require to apply a restoring force to return the robot to vertical position.A reaction wheel pendulum model is followed for the balancing purpose.The components used are:

- High torque 12V DC motor
- A metallic reaction wheel
- 1µf ,15V capacitor

The model is as follows



Mathematical model of the system:

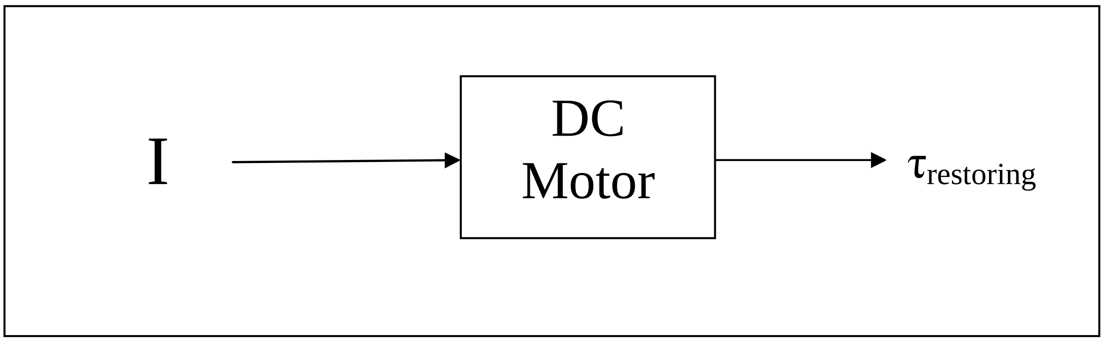
For a displacement of θ;Torque

$\tau = MgL_{c.g}\sin\theta+(m_{motor}+m_{reaction\ wheel})L\sin\theta$

For a DC motor:

$\tau = K_4 I - f$;where f is the loss on the account of friction

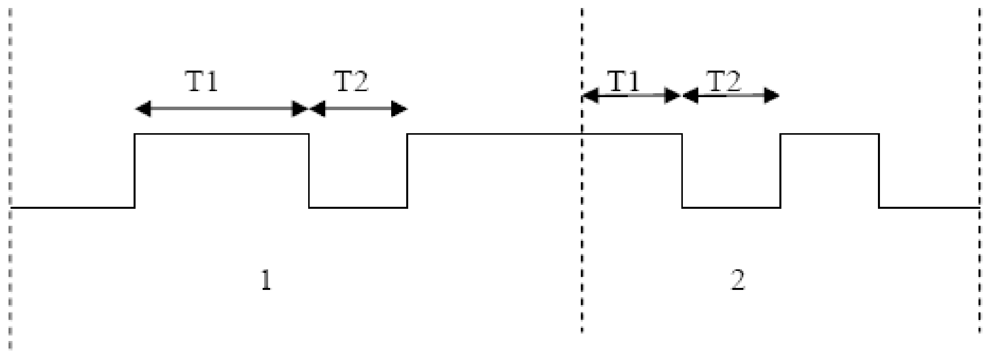
So the equivalent model of the reaction wheel'



Motor Control

Pulse Width Modulation

PWM output is basically a series of pulses with varying size in pulse width. This PWM signal is output from the h-bridge circuit to control the wiper motor. The difference in pulse length shows the different output of h-bridge circuit controlling the output speed of the motor.



Pulse Width Modulation waveform

Figure above shows the varying pulse length of the pulse width modulation (PWM) scheme. Let’s say that the PWM frequency is about 50 Hertz, with a period cycle of 20ms. Therefore assuming that the T1 and T2 length values are 15ms and 5ms respectively, the duty cycle can be calculated as below:

Duty cycle = T1/(T1+T2) * 100%

= 15/20 * 100%

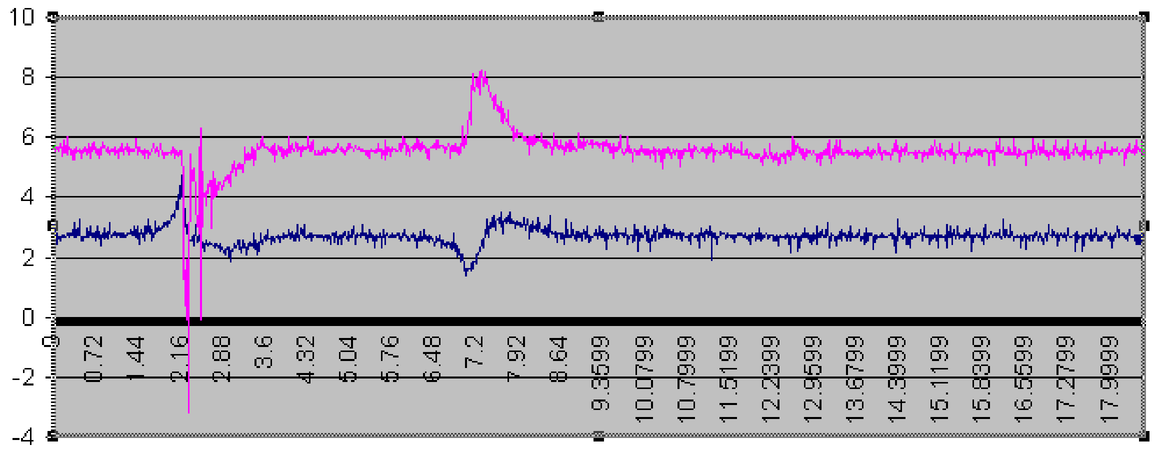
= 75%

The capacitor connected across the motor charges and discharges during the on and off time respectively,thus behaving like an integrator.The torque generated by the motor is a function of the average value of current supplied to the motor.

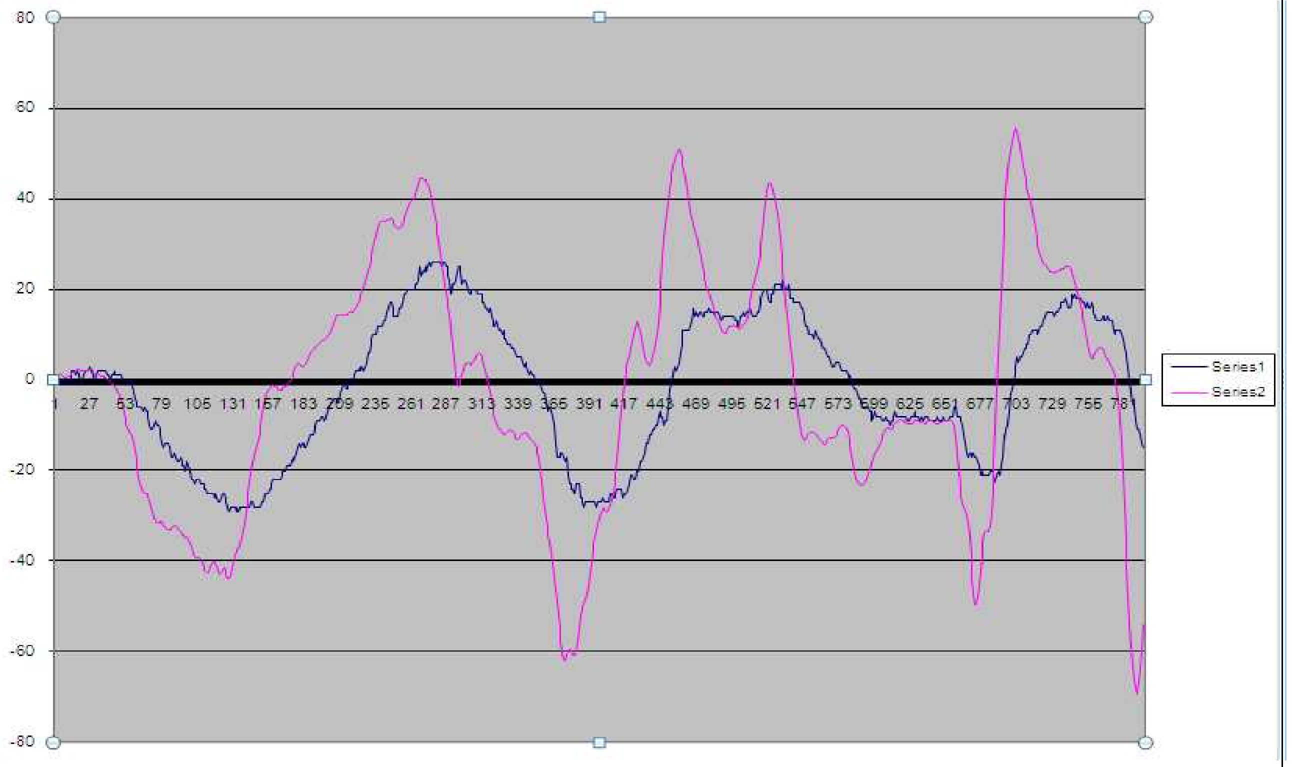
CHAPTER 6:

RESULTS

Individual outputs of accelerometer and gyro:



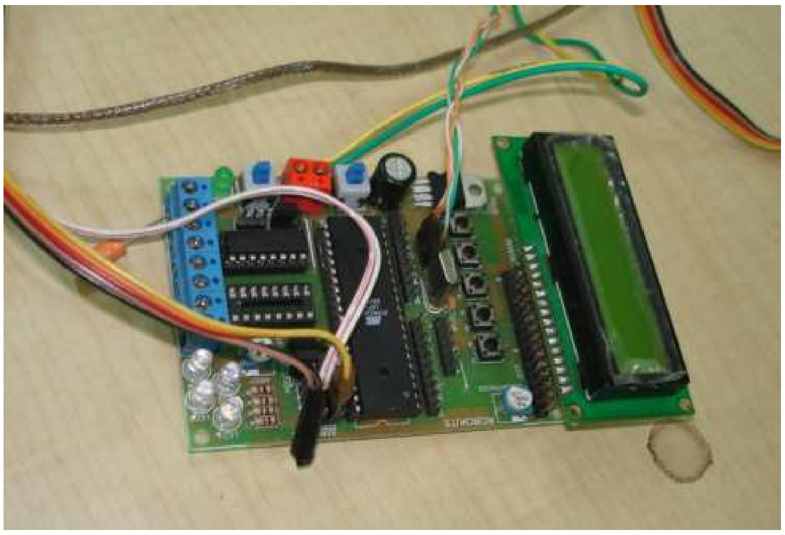
Combined outputs



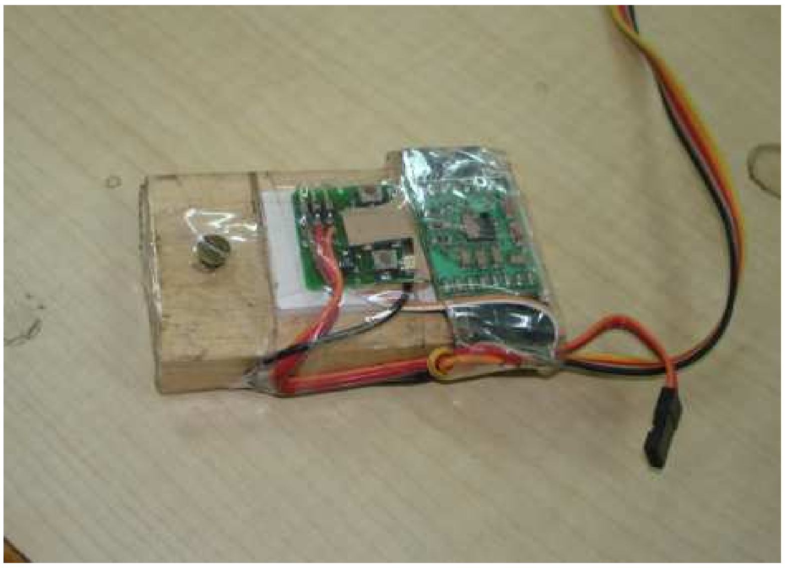
Series1-gravity angle(only accelerometer reading)

Series2-stabilized angles(combined readings)

Reaction wheel on motor

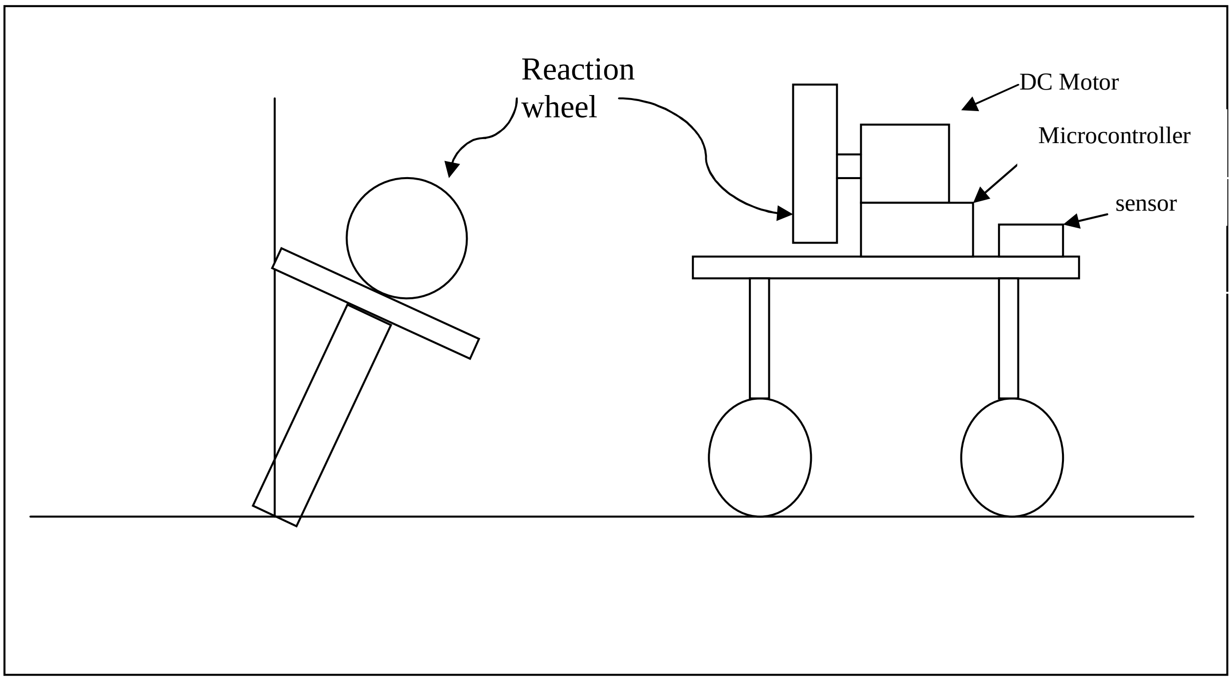


Microcontroller unit



Rate gyro & accelerometer

Schematic of robot:



As seen from the graphs,the gyro responds to rotation instantly while the accelerometer has a phase lag response as determined by Albert-Jan Baervaeldt and Robert Klang, in ‘A Low-cost and Low-weight Attitude Estimation System for an Autonomous Helicopter’, Halmstad University, Sweden.This is corrected in the second graph which shows the combined readings.

CHAPTER 7:

CONCLUSIONS

Researchers could build on what is researched until now. There are a few experiments that are unaccomplished. That is the main drawback that hampers the overall project as concrete result its unable to attain. Therefore appropriate conclusions are not able to achieve. The problem with the oscillation still remains with the system and future work has to be done to achieve a stable solution.

FUTURE IMPROVEMENT

The stabilization provided by the reaction wheel is limited be the torque provided by the reaction wheel motor. Subsequent plan is to use a rotating disc and its gyroscopic precession for balancing. This would provide a more stable design capable of providing higher restoring torque .In such a case particular attention should be paid to any rotary axes, their alignment, and how they are fixed to the model, to the position and alignment of brackets, and to the mounting and fastening of any flexible couplings.

In addition to this, fuzzy logic controller can also be implemented to provide flexibility and accuracy in control.

CHAPTER 8:

REFERENCES

[1]Brennan, L. (1905) U.S. Patent No. 796, 893. Washington, D.C.: U.S. Patent and Trademark Office.

[2]Carter, De Rubis, Guiterrez, Schoellig, Stolar. "Gyroscopically Balanced Monorail System Final Report" (2005) Columbia University.

[3]E. Ferreira, S. Tsai, C. Paredis, and H. Brown Advanced Robotics, Vol. 14, No. 6, June, 2000, pp. 459 - 475.

[4]C.H. Ross, J. C. Hung, "Stabilization of an Unmanned Bicycle," Proc. IEEE Region III Convention, 1968, pp. 17.4.1-17.4.8.

[5]Gallaspy, J. "Gyroscopic Stabilization of an Unmanned Bicycle." Ph.D. Thesis, Auburn University (2000).

[6] Anderson, D.P, 'Nbot, a two wheel balancing robot', <http://www.geology.smu.edu/~dpa-www/robo/nbot>

[7] Steve Hassenplug, 2002, 'Steve's Legway', <http://www.teamhassenplug.org/robots/legway/>

[8] Dean Kamen ,2001,<http://www.segway.com>

[9] John Green,David Krakauer, March 2003, New iMEMS Angular Rate Sensing Gyroscope,<http://www.analog.com/library/analogDialogue/archives/37-03/gyro.html>

[10] Peter Hemsley, 32-bit signed integer maths for PICS, <http://www.piclist.com/techref/microchip/math/32bmath-ph.htm>

[11] Mosfets and Mosfet's drivers, <http://homepages.which.net/~paul.hills/SpeedControl/Mosfets.html>

[12] Rick Bickle, 11 July 2003,'DC motor control systems for robot applications', <http://www.dprg.org/tutorials/2003-10a/motorcontrol.pdf>

[13] Carnegie Mellon, 26 August 1997, 'Control Tutorials for Matlab', The University of Michigan,<http://www.engin.umich.edu/group/ctm/PID/PID.html>

[14] <http://www.boondog.com/tutorials/mouse/mouseHack.htm>

[15] Martin Rowe, 11 January 2001, Measuring PWM motor efficiency, Test & Measurement World,<http://www.reed-electronics.com/tmworld/article/CA180848.html >

[16] Gerry, 6 Ferbruary , Tilt sensors for your Robot, <http://www.roboticsindia.com/modules.php?name=News&file=article&sid=90>

[17] (c) 1998, 2001 EME Systems, Berkeley CA U.S.A., Pulse Width Modulation, http://www.emesystems.com/BS2PWM.htm

[18] Dennis Clark and Michael Owings, 'Building Robot Drive Trains', McGraw Hill Companies.

[19] Naoji Shiroma, Osamu Matsumoto, Shuji Kajita, Kazuo Tani, 'Cooperative Behavior of a Wheeled Inverted Pendulum for Object Transportation', Proceedings of the 1996

[20] IEEE/RSJ International conference on Intelligent Robots and Systems '96, IROS 96,volume:2, 4-8Nov. 1996 Pg(s): 396-401 vol.

[21] Grasser, Felix, Alonso D'Arrigo, Silvio Colombi & Alfred C. Rufer, 2002, 'JOE: A Mobile, Inverted Pendulum', IEEE Transactions on Industrial Electronics, vol 49.

[22] Young Soo Suh, 'Attitude Estimation using Low Cost Accelerometer and Gyroscope',

[23] Proceedings of the 7th Korea-Russia International Symposium, KORUS 2003,Pg(s) 423-427.

[24] Albert-Jan Baervaeldt and Robert Klang, 'A Low-cost and Low-weight Attitude

Estimation System for an Autonomous Helicopter', Halmstad University, Sweden, Pg(s) 391-395.

[25] Yongjun Hou, Greg R.Luecke, October 5-8 2003, 'Control of the Tight Rope Balancing Robot', Proceedings of the 2003 IEEE International Symposium on Intelligent Control,Houston, Texas, Pg(s): 896-901.

[26] Alessio Salerno and Jorge Angles, 'The Control of Semi-Autonomous Two-Wheeled Robots Undergoing Large Payload-Variations', Proceedings of the 2004 IEEE International Conference on Robotics & Automation, New Orleans, LA, Pg(s): 1740-1745.

[27] Kiyoshi Komoriya and Eimei Oyama, 'Position Estimation of a Mobile Robot Using Optical Fiber Gyroscope (OFG)', Pg(s): 143-149.

[28] Prof. John Billingsley, Mechatronics Practice Unit,<http://www.usq.edu.au/course/material/eng3905/>

[29] Dr.Tony Ah Fock, Power Electronics study book 1 and 2, University of Southern Queensland.

[30] Paul E.Sandin , "Robot Mechanisms and Mechanical Devices Illustrated" , The McGrawHill companies.

[31] J.J. D'Azzo, C.H. Houpis, Feedback Control System Analysis and Synthesis, second edition, McGraw-Hill International Editions. (ISBN 0-07-Y85150-6) Pg11

[32] 'Inverted Pendulum', Microrobot NA, http://www.microrobotna.com/pendulum.htm

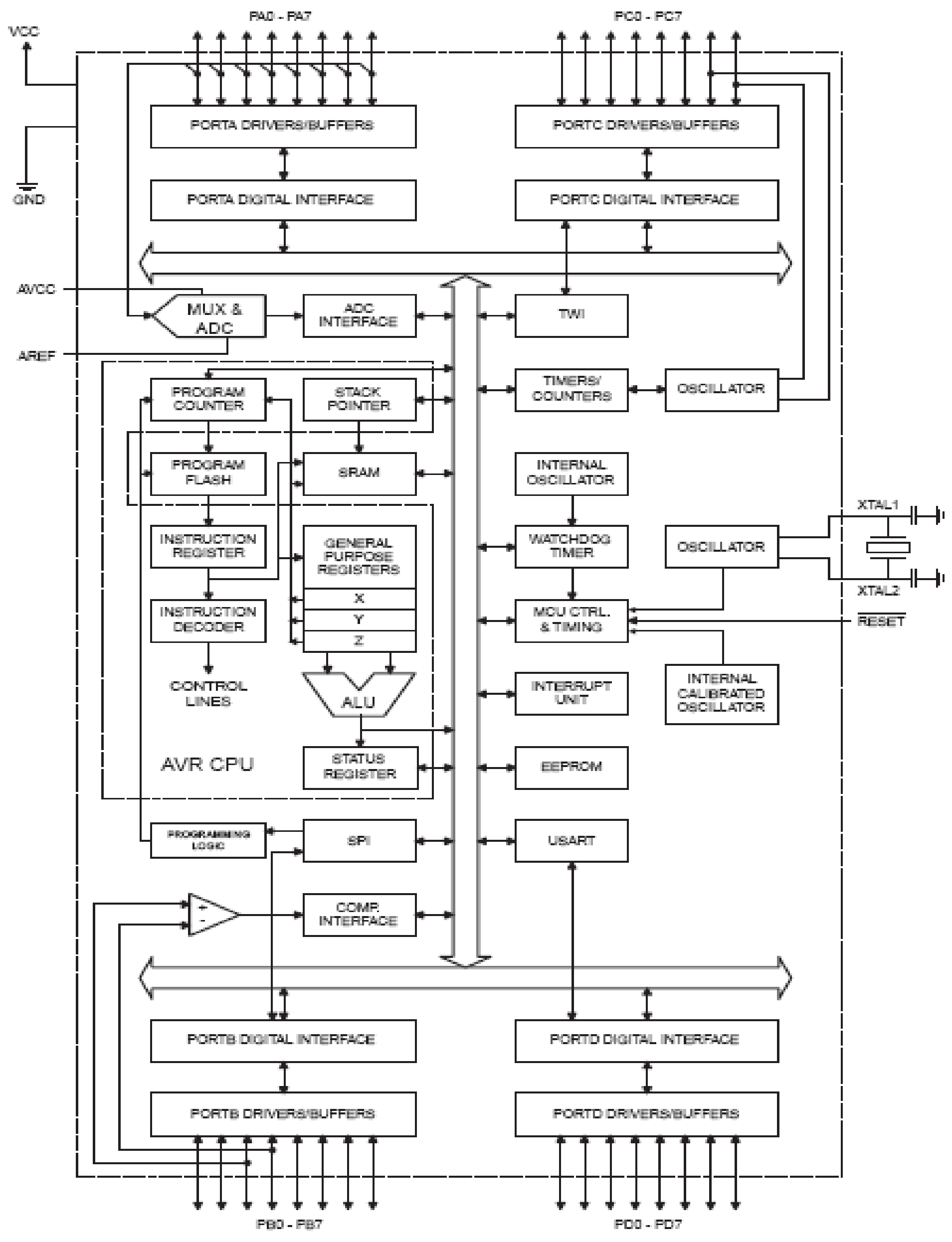
[33] R.E Kalman, 1960, 'A New Approach to Linear Filtering and Prediction Problems', Transactions of the ASME- Journal of Basic Engineering, 82(Series D), pp35-45.

APPENDIX-1
Features of ATMega-16

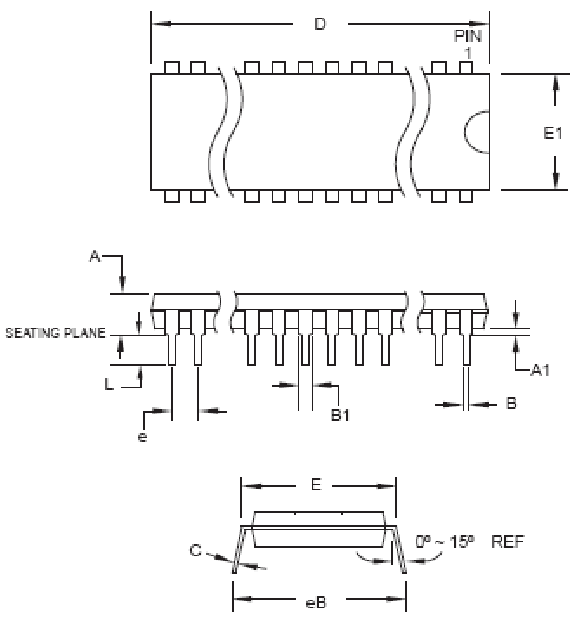
• High-performance, Low-power AVR® 8-bit Microcontroller
• Advanced RISC Architecture
– 131 Powerful Instructions – Most Single-clock Cycle Execution
– 32 x 8 General Purpose Working Registers
– Fully Static Operation
– Up to 16 MIPS Throughput at 16 MHz
– On-chip 2-cycle Multiplier
• High Endurance Non-volatile Memory segments
– 16K Bytes of In-System Self-programmable Flash program memory
– 512 Bytes EEPROM
– 1K Byte Internal SRAM
– Write/Erase Cycles: 10,000 Flash/100,000 EEPROM
– Data retention: 20 years at 85℃/100 years at 25° C(1)
– Optional Boot Code Section with Independent Lock Bits
In-System Programming by On-chip Boot Program
True Read-While-Write Operation
– Programming Lock for Software Security
• JTAG (IEEE std. 1149.1 Compliant) Interface
– Boundary-scan Capabilities According to the JTAG Standard
– Extensive On-chip Debug Support
– Programming of Flash, EEPROM, Fuses, and Lock Bits through the JTAG Interface
• Peripheral Features
– Two 8-bit Timer/Counters with Separate Prescalers and Compare Modes
– One 16-bit Timer/Counter with Separate Prescaler, Compare Mode, and Capture Mode
– Real Time Counter with Separate Oscillator
– Four PWM Channels
– 8-channel, 10-bit ADC
8 Single-ended Channels
7 Differential Channels in TQFP Package Only
2 Differential Channels with Programmable Gain at 1x, 10x, or 200x
– Byte-oriented Two-wire Serial Interface
– Programmable Serial USART
– Master/Slave SPI Serial Interface
– Programmable Watchdog Timer with Separate On-chip Oscillator
– On-chip Analog Comparator
• Special Microcontroller Features
– Power-on Reset and Programmable Brown-out Detection
– Internal Calibrated RC Oscillator
– External and Internal Interrupt Sources
– Six Sleep Modes: Idle, ADC Noise Reduction, Power-save, Power-down, Standby and Extended Standby
• I/O and Packages
– 32 Programmable I/O Lines
– 40-pin PDIP, 44-lead TQFP, and 44-pad QFN/MLF
• Operating Voltages
– 2.7 - 5.5V for ATmega16L
– 4.5 - 5.5V for ATmega16
• Speed Grades
– 0 - 8 MHz for ATmega16L
– 0 - 16 MHz for ATmega16
• Power Consumption @ 1 MHz, 3V, and 25℃ for ATmega16L
– Active: 1.1 mA
– Idle Mode: 0.35 mA
– Power-down Mode: < 1 μA

Internal block diagram of ATMega16

Figure 2. Block Diagram



Packaging Information



COMMON DIMENSIONS
(Unit of Measure = mm)

SYMBOL	MIN	NOM	MAX	NOTE
A	–	–	4.826	
A1	0.381	–	–	
D	52.070	–	52.578	Note 2
E	15.240	–	15.875	
E1	13.462	–	13.970	Note 2
B	0.356	–	0.559	
B1	1.041	–	1.651	
L	3.048	–	3.556	
C	0.203	–	0.381	
eB	15.494	–	17.526	
e	2.540 TYP			

Notes: 1. This package conforms to JEDEC reference MS-011, Variation AC.
2. Dimensions D and E1 do not include mold Flash or Protrusion.
Mold Flash or Protrusion shall not exceed 0.25 mm (0.010").

09/28/01

	TITLE	DRAWING NO.	REV.
2325 Orchard Parkway San Jose, CA 95131	**40P6**, 40-lead (0.600"/15.24 mm Wide) Plastic Dual Inline Package (PDIP)	40P6	B

APPENDIX-2

Accelerometer module specification

Chip-freescale semiconductors
Board-Robokits

Board size - 28mm X 23mm

5 pin interface (VCC, GND, Xout, Yout, Zout)

Selectable Sensitivity (1.5g/2g/4g/6g) and Sleep Mode Selectable through jumpers or microcontroller

Current Consumption: 500 µA

Low Voltage Operation: 2.6V to 5V

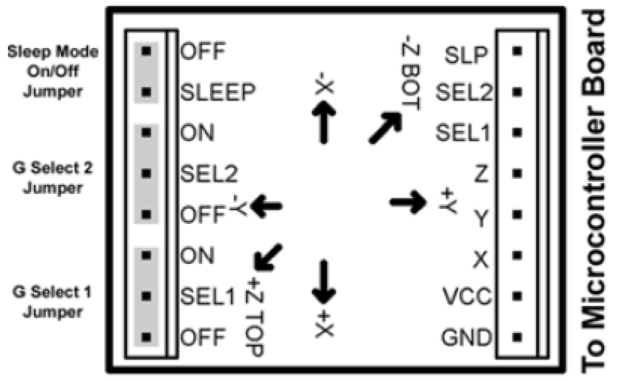
High Sensitivity (800 mV/g @ 1.5g) for small movements

Fast Turn On Time

Integral Signal Conditioning with Low Pass Filter

Robust Design, High Shocks Survivability

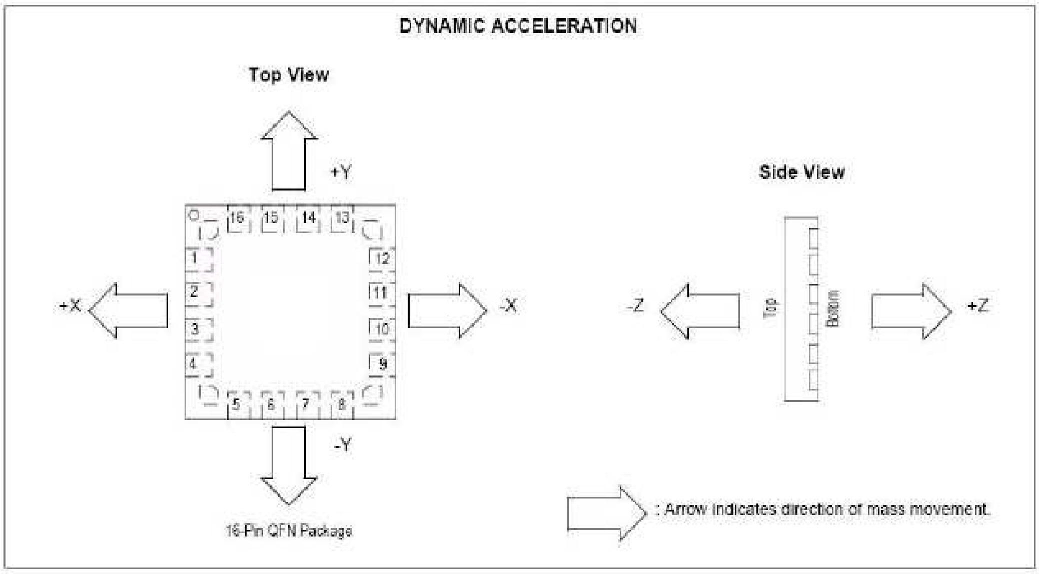
Board Top Layout

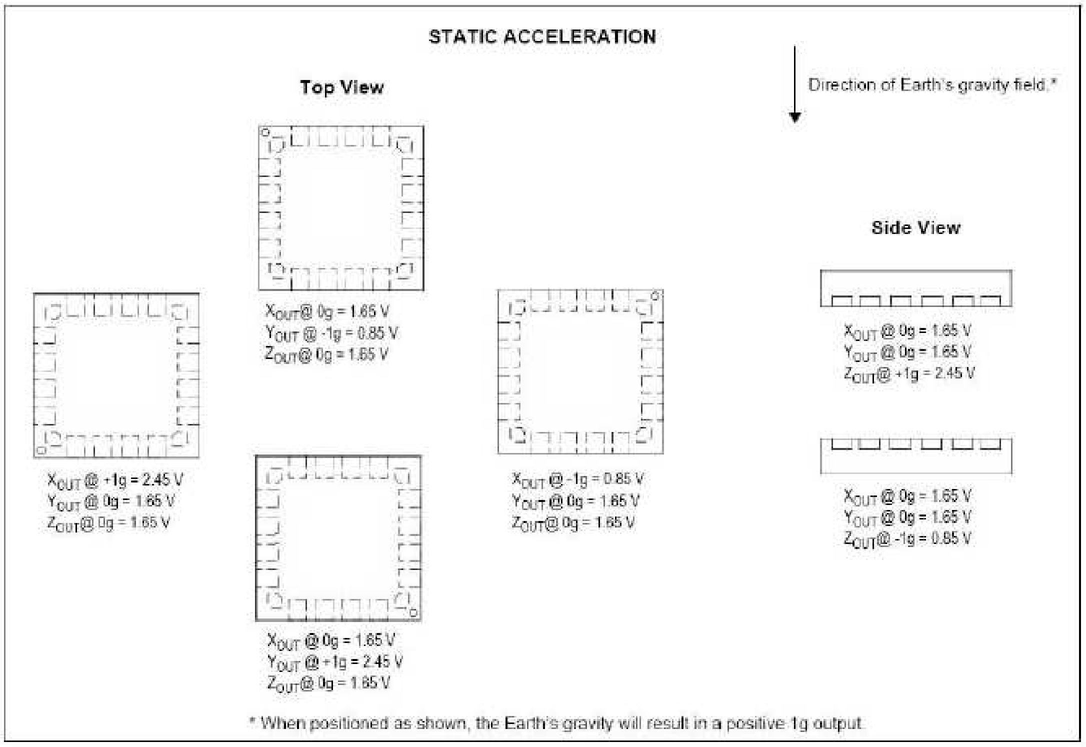


SLP – Sleep Mode*
SEL2 – g-Select2*
SEL1 – g-Select1*
Z – Z Axis analog Output
Y – Y Axis analog Output
X – X Axis analog Output
VCC – 2.6V to 5V input
GND - Ground

*Do not provide more than 3.6V input to these pins. If not controlled by microcontroller use Jumpers on other side to select mode and leave these pins open.

g-Select2	g-Select1	g-Range	Sensitivity
0	0	1.5g	800 mV/g
0	1	2g	600 mV/g
1	0	4g	300 mV/g
1	1	6g	200 mV/g

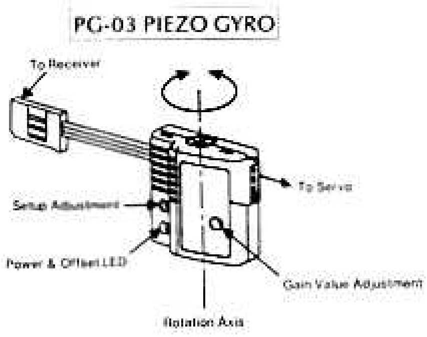

DYNAMIC ACCELERATION
Top View
+Y
16 15 14 13
1 2 3 4
12 11 10 9
5 6 7 8
+X
-X
-Y
16-Pin QFN Package
Side View
-Z
Top
Bottom
+Z
: Arrow indicates direction of mass movement.


STATIC ACCELERATION
Top View
Direction of Earth's gravity field.*
X_{OUT}@ 0g = 1.65 V
Y_{OUT} @ -1g = 0.85 V
Z_{OUT}@ 0g = 1.65 V
X_{OUT} @ +1g = 2.45 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT}@ 0g = 1.65 V
X_{OUT} @ -1g = 0.85 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT}@ 0g = 1.65 V
X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ +1g = 2.45 V
Z_{OUT}@ 0g = 1.65 V
Side View
X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT}@ +1g = 2.45 V
X_{OUT} @ 0g = 1.65 V
Y_{OUT} @ 0g = 1.65 V
Z_{OUT}@ -1g = 0.85 V
* When positioned as shown, the Earth's gravity will result in a positive 1g output.

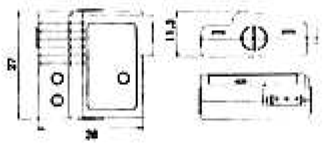
APPENDIX-3

GWS PG-03 PIEZO GYRO INSTRUCTIONS

Note: Please read the instruction manual thoroughly before operation.

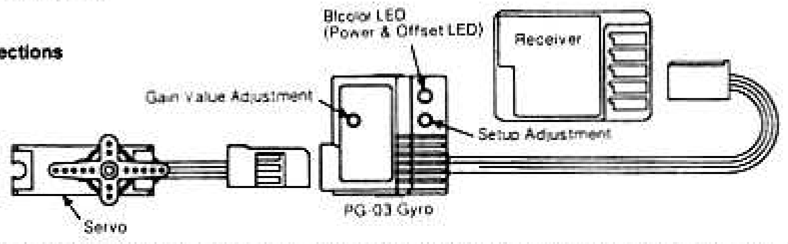


Specifications:
Dimension: 26.0 x 27.0 x 11.3 mm
Weight: **7.0 g** (0.285 oz) with Plastic Case
4.8 g (0.169 oz) w/o Plastic Case
Power Supply: 4.8 ~ 6.0 Volts
Current Draw: 10mA (4.8V)
Gain Adjustment: Single rate, non-remote
Operating Temperature: - 5℃ ~ 60℃
Applicable R/C System: Futaba, JR, Hi-tec, Sanwa/Airtronics, Multiplex, GWS



INTRODUCTION
Thank you for choosing Grand Wing Servo-Tech (GWS) PG-03 Piezo Gyro System. The PG-03 has been designed to use new micro piezo sensor, which has exclusively been developed for this state-of-the-art gyro system that has better and quicker response and is much stronger resistance in crash or unexpected external impact. Also, it has better temperature characteristic and neutral stability owing to new circuit design. The PG-03 ultra-light weight and super-compact size allows it to be installed in all size helicopters, airplanes, cars and boats. Please read the instruction manual completely before you attempt to operate the system.

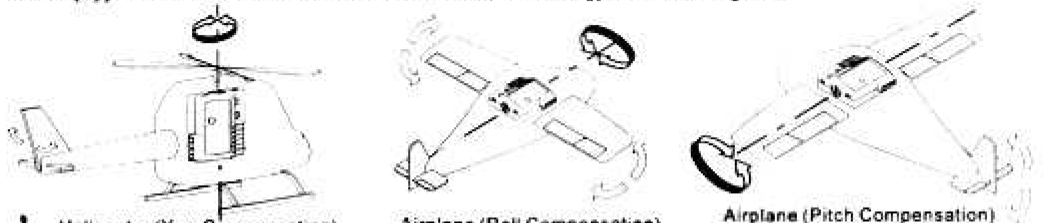
INSTALLATION
Wiring and Connections



Disconnect the servo to be compensated from the receiver and plug the connector from the gyro there, then insert the servo connector to the gyro port. Be sure to observe the proper polarity on the connector and the gyro port (socket). If the wires on the components are excessively dragged each other, it is not possible to get best vibration absorption with the mounting foam tape, also it may cause possible disconnection during use.

Location
Find most ideal location on your model to mount the gyro, area in low vibration or follow the manufacturer recommended location of your model. Please make sure that the setup and gain value adjustment trimmers are accessible for future adjustment.

Mounting
Attach the supplied double-sided adhesive foam tape to the bottom of the gyro and mount it to the ideal position of your model. A conventional adhesive tape for a gyro mounting is not suitable because it can not absorb vibration enough for the ultra-light PG-03 gyro. It is always essential to check if the gyro is compensating in the proper direction as no reversing switch equipped on the PG-03. If the direction is reversed, rotate the gyro itself 180 degrees.



• Helicopter (Yaw Compensation)
RX-PG-03-Rudder Servo

Airplane (Roll Compensation)
RX-PG-03-Aileron Servo

Airplane (Pitch Compensation)
RX-PG-03-Elevator Servo

P1