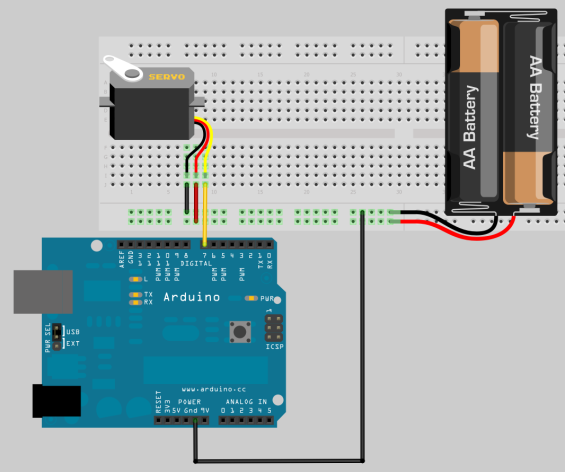
SERVO:



#include <Servo.h>

Servo myservo; // create servo object to control a servo

// twelve servo objects can be created on most boards

int pos = 0; // variable to store the servo position

void setup() {

myservo.attach(9); // attaches the servo on pin 9 to the servo object

}

void loop() {

for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees

// in steps of 1 degree

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees

myservo.write(pos); // tell servo to go to position in variable 'pos'

delay(15); // waits 15ms for the servo to reach the position

}

myservo.write(180); // tell servo to go to position in variable 'pos'

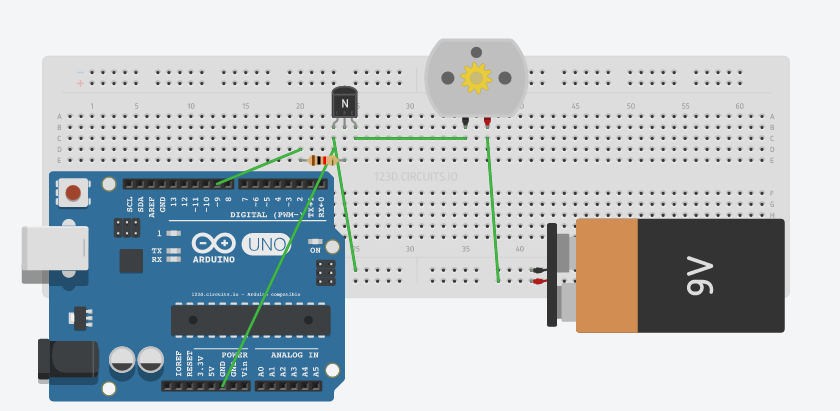
delay(2000);

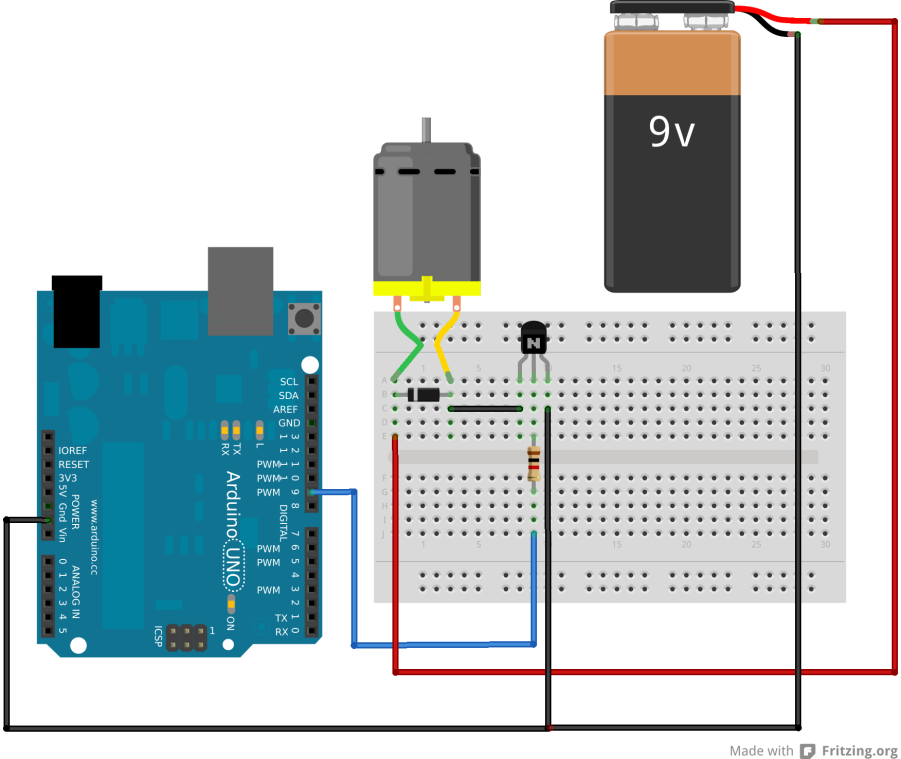
myservo.write(90); // tell servo to go to position in variable 'pos'

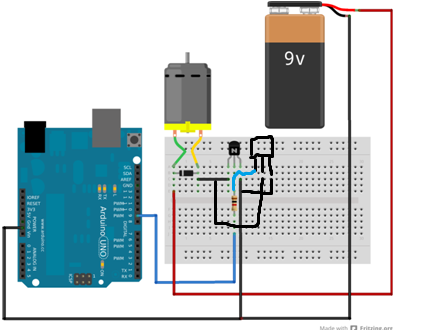
delay(2000);

}

MOTOR DC con PWM:







int led = 9; //Pin donde se encuentra el LED, salida

int pos = 0;

//------------------------------------

//Funcion principal

//------------------------------------

void setup () // Se ejecuta cada vez que el Arduino se inicia

{

pinMode(led, OUTPUT); //Configurar el LED como una salida

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop () // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

// Escritura analoga (PWM) en el LED escribo el valor de brillo

for (pos = 0; pos <= 255; pos += 1) { //el maximo para el puerto analogo es 255

// in steps of 1 degree

Motor1(led, pos);

delay(15);

}

for (pos = 255; pos >= 0; pos -= 1) {

Motor1(led, pos);

delay(15);

}

Motor1(led, pos);

delay(2000);

}

void Motor1(int pin, int velocidad){

analogWrite(pin, velocidad);

}

BUZZER: Melodía:

int tempo = 350;

int longitud = 15; // Número de notas de la canción

/\*El array notas[] tiene las notas de la canción

Se emplea la notación americana:

do=c, re=d, mi=e, fa=f, sol=g, la=a, si=b

Un espacio representa un silencio \*/

char notas[] = "ccggaagffeeddc ";

/\*Duración de cada nota de la canción.

"2" dura el doble que "1", “4” dura el doble que “2”, etc\*/

int pulsacion[] = {1,1,1,1,1,1,2,1,1,1,1,1,1,2,4};

//Nombres de las notas de una escala

char nombres[] = { 'c', 'd', 'e', 'f', 'g', 'a', 'b', 'C' };

//Sus correspondientes frecuencias

int tonos[] = { 523, 587, 659, 698, 783, 880,98, 1047 };

void setup() {

pinMode(8, OUTPUT);

}

void loop() {

int i;

for (i = 0; i < longitud; i++) {

if (notas[i] == ' ') {

//Si hay silencio, se espera

delay(pulsacion[i] \* tempo);

} else {

//Si no, emite la nota

playNota(notas[i]);

//con la duración especificadas

delay(pulsacion[i]\*tempo);

noTone(8);

}

}

}

void playNota(char nota) {

int i;

/\*Emite la frecuencia correspondiente al nombre de la nota. Para

ello, busco en el array nombres[] si la nota especificada como

parámetro está allí. \*/

for (i = 0; i < 8; i++) {

if (nombres[i] == nota) {

tone(8,tonos[i]);

}

}

}

GENERACION PULSOS PWM:

int led = 9; //Pin donde se encuentra el LED, salida

int pos = 0;

//------------------------------------

//Funcion principal

//------------------------------------

void setup () // Se ejecuta cada vez que el Arduino se inicia

{

pinMode(led, OUTPUT); //Configurar el LED como una salida

}

//------------------------------------

//Funcion ciclicla

//------------------------------------

void loop () // Esta funcion se mantiene ejecutando

{ // cuando este energizado el Arduino

// Escritura analoga (PWM) en el LED escribo el valor de brillo

Motor1(led, 255);

delay(10); //determina el tiempo mas alto de la señal cuadrada

Motor1(led, 0);

delay(40); //determina el tiempo mas bajo de la señal cuadrada

}

void Motor1(int pin, int velocidad){

analogWrite(pin, velocidad);

//analogWrite(pin, 0);

}

LECTURA DE monitor serie prender 3 LED:

char color; //Es importante que sea de tipo char y no byte

byte brillo;

byte LedRojo = 5;

byte LedVerde = 6;

byte LedAzul = 9;

void setup() {

Serial.begin(9600);

pinMode(LedRojo, OUTPUT);

pinMode(LedVerde, OUTPUT);

pinMode(LedAzul, OUTPUT);

analogWrite(LedRojo, 0); //Establezco un brillo inicial medio

analogWrite(LedVerde, 0);

analogWrite(LedAzul, 0);

}

void loop () {

//Leo el primer carácter recibido por el canal serie

if (Serial.available()>0) {

color=Serial.read();

if( color == 'r' || color == 'g' || color == 'b' ) {

delay(5);

//Extraigo el número que sigue a la primera letra

brillo=byte(Serial.parseInt());

if(color == 'r') {

analogWrite(LedRojo, brillo);

} else if(color == 'g'){

analogWrite(LedVerde, brillo);

}else if(color == 'b'){

analogWrite(LedAzul, brillo);

}

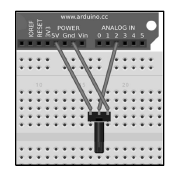
}

}

delay(100); //Me espero para recibir los datos serie bien

}

Lectura de Potenciometro PIN Analogo enciendo 3 led:



int valorPot=0;

char color; //Es importante que sea de tipo char y no byte

byte brillo;

byte LedRojo = 5;

byte LedVerde = 6;

byte LedAzul = 9;

void setup() {

Serial.begin(9600);

pinMode(LedRojo, OUTPUT);

pinMode(LedVerde, OUTPUT);

pinMode(LedAzul, OUTPUT);

analogWrite(LedRojo, 0); //Establezco un brillo inicial medio

analogWrite(LedVerde, 0);

analogWrite(LedAzul, 0);

}

void loop () {

valorPot=analogRead(2);

Serial.println(valorPot);

delay(100);

if( valorPot < 50 ) {

prenderled(LedRojo);

} else if(valorPot > 50 && valorPot < 500){

prenderled(LedVerde);

}else if( valorPot > 500){

prenderled(LedAzul);

}

delay(100); //Me espero para recibir los datos serie bien

}

void prenderled(int pin){

analogWrite(LedRojo, 0);

analogWrite(LedVerde, 0);

analogWrite(LedAzul, 0);

Serial.println(pin);

analogWrite(pin, 255);

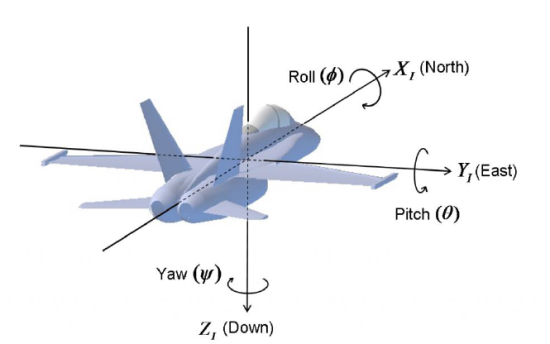
}

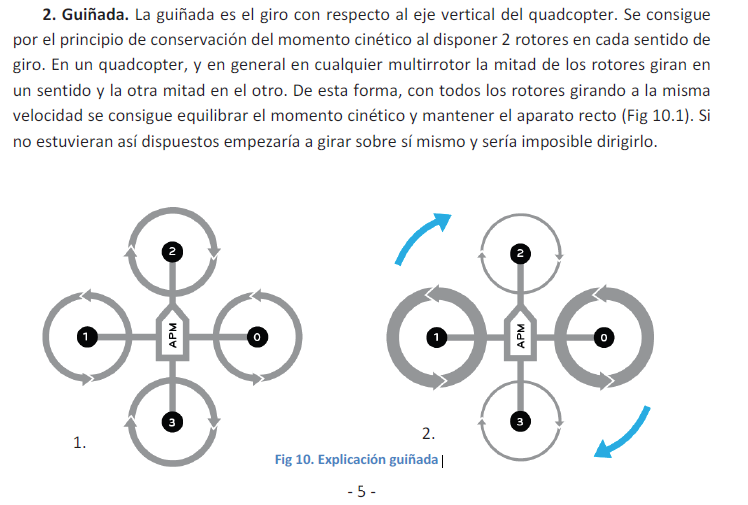
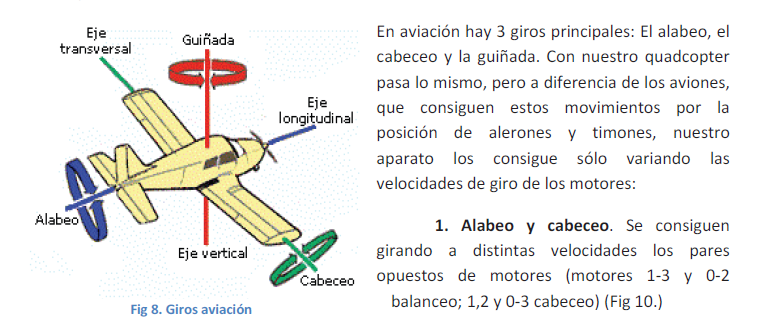
Lectura de Gyroscopio mpu-6050:

X = roll o alabeo.

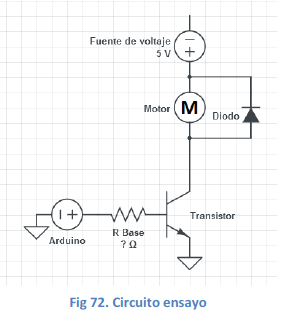
y = pitch o cabeceo.

Z = Yaw o giñana.





Circuito para probar los motores:



Código eficiente a medias , no mide z con #include "MPU6050.h"

Otro ejemplo es el que viene con arduino utiliza otra librería..

#include "I2Cdev.h"

#include "MPU6050.h"

#include "Wire.h"

MPU6050 accelgyro;

int16\_t ax, ay, az, gx, gy, gz;

double timeStep, time, timePrev;

double arx, ary, arz, grx, gry, grz, gsx, gsy, gsz, rx, ry, rz;

int i;

double gyroScale = 131;

void setup() {

Wire.begin();

Serial.begin(9600);

accelgyro.initialize();

time = millis();

i = 1;

}

void loop() {

// set up time for integration

timePrev = time;

time = millis();

timeStep = (time - timePrev) / 1000; // time-step in s

// collect readings

accelgyro.getMotion6(&ax, &ay, &az, &gx, &gy, &gz);

// apply gyro scale from datasheet

gsx = gx/gyroScale;

gsy = gy/gyroScale;

gsz = gz/gyroScale;

// calculate accelerometer angles

arx = (180/3.141592) \* atan(ax / sqrt(square(ay) + square(az)));

ary = (180/3.141592) \* atan(ay / sqrt(square(ax) + square(az)));

arz = (180/3.141592) \* atan(sqrt(square(ay) + square(ax)) / az);

// set initial values equal to accel values

if (i == 1) {

grx = arx;

gry = ary;

grz = arz;

}

// integrate to find the gyro angle

else{

grx = grx + (timeStep \* gsx);

gry = gry + (timeStep \* gsy);

grz = grz + (timeStep \* gsz);

}

// apply filter

/\*

rx = (0.1 \* arx) + (0.9 \* grx);

ry = (0.1 \* ary) + (0.9 \* gry);

rz = (0.1 \* arz) + (0.9 \* grz);

\*/

rx = (0.96 \* arx) + (0.04 \* grx);

ry = (0.96 \* ary) + (0.04 \* gry);

rz = (0.96 \* arz) + (0.04 \* grz);

// print result

/\*Serial.print(i); Serial.print(":i\t");

Serial.print(timePrev); Serial.print(":timePrev\t");

Serial.print(time); Serial.print(":time\t");

Serial.print(timeStep, 5); Serial.print("\t\t");

Serial.print(ax); Serial.print(":ax\t");

Serial.print(ay); Serial.print(":ay\t");

Serial.print(az); Serial.print(":az\t");

Serial.print(gx); Serial.print(":gx\t");

Serial.print(gy); Serial.print(":gy\t");

Serial.print(gz); Serial.print(":gz\t");

\*/

//Se imprime x-roll o alabeo

Serial.print(arx); Serial.print(" \t"); Serial.print(" :x-roll o alabeo\t");

//Se imprime y-pitch o cabeceo

Serial.print(ary); Serial.print(" \t"); Serial.print(" :y-pitch o cabeceo\t");

Serial.print(arz); Serial.print(" \t"); Serial.print(" :arz\t");

Serial.print(grx); Serial.print(" \t"); Serial.print(":gyro rx\t");

Serial.print(gry); Serial.print(" \t"); Serial.print(":gyro ry\t");

Serial.print(grz); Serial.print(" \t"); Serial.print(":gyro rz\t");

/\*

Serial.print(rx); Serial.print(":rx\t");

Serial.print(ry); Serial.print(":ry\t");

Serial.println(rz);

\*/

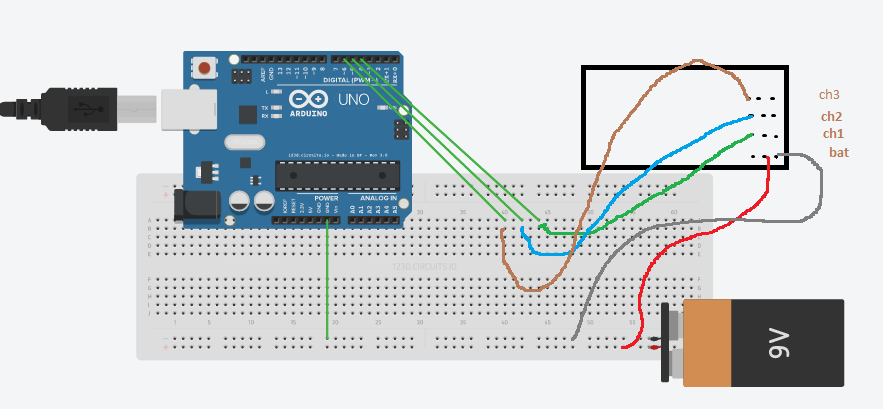
Serial.println("fin");

i = i + 1;

delay(1050);

}

**Lector PPM para Arduino mejorado con filtro antisalto (anti-glitch):**



Código sin filtrado de la señal:

#define channumber 3 //Cuantos canales tiene tu radio???????/How many channels have your radio???

#define filter 3 // Filtro anti salto/ Glitch Filter

int channel[channumber]; //Valores de canales leidos/ readed Channel values

int lastReadChannel[channumber]; //Ultima lectura obtenida/ Last values readed

int conta=0; //Contador/couter

int valor=0;

int pinCanal[channumber];

int pinComienzo=4;

void setup()

{

Serial.begin(9600); //Iniciamos com serial/ Serial Begin

pinMode(13, OUTPUT); // Led pin 13

for(int i = 0; i <= channumber-1; i++) //lee los pulsos de los canales / Read the pulses of the channels

{

pinCanal[i]=i+pinComienzo; //Se coloca el 1er canal en el pin 4 en forma creciente.

pinMode(i+pinComienzo, INPUT); //Setea los pines de lectura comienza en el cuatro hasta N channumber

}

}

void loop()

{

valor=pulseIn(pinCanal[0], HIGH); //Lee uno de los canales para ver si esta prendido el emisor o radio.

if(valor <= 2000 && valor >= 1100) //Si el pulso leido no se encuentra dentro del rango es porque el radio esta apagado.

{

for(int i = 0; i <= channumber-1; i++) //Se leen los N canales guardando los mismos en el vector de valores.

{

channel[i]=pulseIn(pinCanal[i], HIGH);

Serial.print("valor sin filtrar ");

Serial.print(" Canal " );

Serial.print( i+1 );

Serial.print( "-");

Serial.println(channel[i]);

}

Serial.println( " ");

}else{

Serial.print("Radio sin señal!!");

Serial.println(valor);

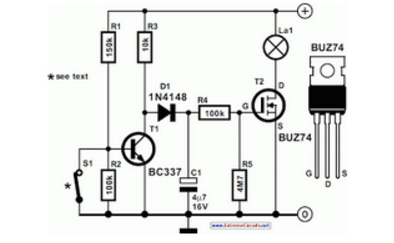
}

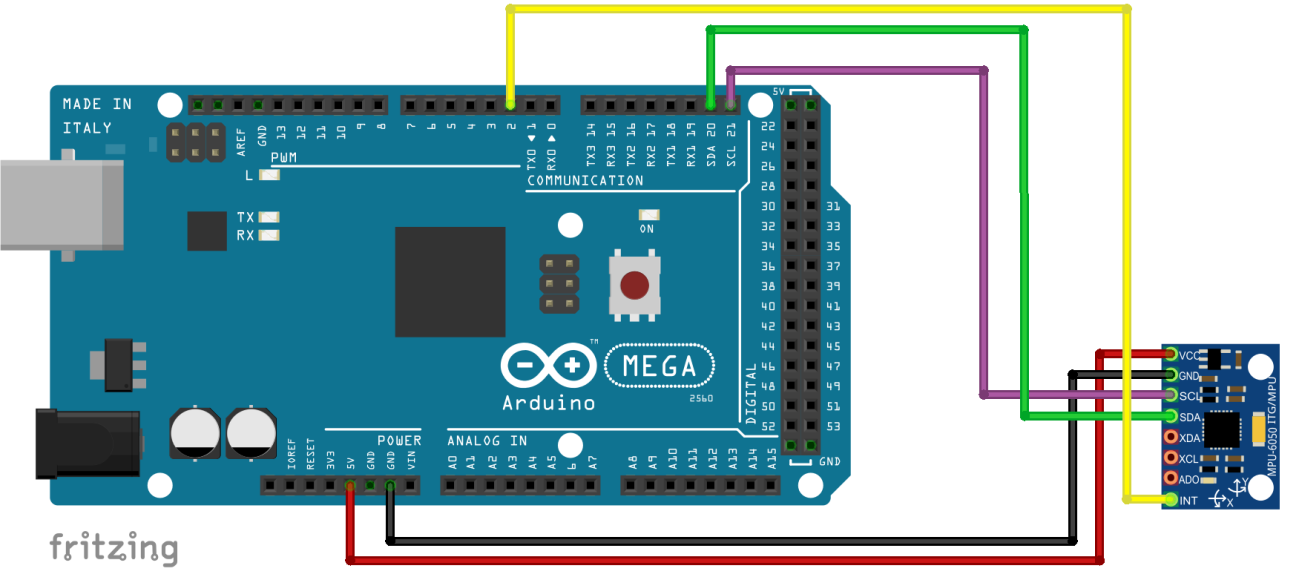
delay(400); //Delay

conta=0;//Reinicia el contador/ Restart couter.

}

Nivelar motores balancín con el giroscopio: mpu6050+pwm





//ESTE EJEMPLO ESTA SACADO DE UN PDF DE QUADCOPTERS y modificada la logica.

//Aes\_quad bueno bien explicado\_con codigo.pdf

#include "I2Cdev.h"

#include "MPU6050.h"

#include "Wire.h"

MPU6050 mpu;

int16\_t ax, ay, az, gx, gy, gz;

float axx, ayy, azz; //variables calculadas a angulos.

float alfa=0.8;

float axx2 = 0, ayy2 = 0, azz2 = 0;

int estado=1, angulo, anguloabs;

int factorEstabilidad = 5;

int motor1=100; //valor minimo para que el motor encienda.

int motor2=100; //valor minimo para que el motor encienda.

int acelerador=150; //acelerado al maximo 255.

void setup() {

Wire.begin(); //comando necesario para la comunicacion serial con el mpu.

Serial.begin(9600);

pinMode(5,OUTPUT); //PIN DE MOTOR Y DERECHO motor1

pinMode(6,OUTPUT); //PIN DE MOTOR Y IZQUIERDO moto2

mpu.initialize(); //inicializacion del acelerometro

mpu.setSleepEnabled(false); //Nos aseguramos de que el acelerometro no este en modo sleep.

}

void loop() {

if(Serial.available()>0){

estado = Serial.read(); //leo el primer dato y lo guardo en variable.

Serial.flush(); //borro lo que hay en el buffer del puerto serie.

}

if(estado== 1){

/\*obtencion del angulo\*/

ayy2=ayy;

mpu.getMotion6(&ax, &ay, &az, &gx, &gy, &gz); //leo los valores del giro y los guardo en la variable que he declarado al principio.

ayy=(1-alfa)\*ay+alfa\*ayy2;

angulo=ayy/166.44;

/\* end angulo \*/

//se va hacia la izquierda acelero motor2 y desaselero motor1.

if(angulo<0){

anguloabs=abs(angulo);

Serial.println(anguloabs);

Serial.println(" angulo negativo ");

motor2=acelerador+(anguloabs\*factorEstabilidad); //como el valor es negativo - \* - = + acelero motor

motor2=constrain(motor2,50,255);

motor1=acelerador-(anguloabs\*factorEstabilidad); // como el valor es negativo + \* - = - desacelero motor.

motor1=constrain(motor1,50,255);

}else if(angulo>0){

Serial.println(" angulo positivo ");

motor1=acelerador+(angulo\*factorEstabilidad);

motor1=constrain(motor1,50,255);

motor2=acelerador-(angulo\*factorEstabilidad);

motor2=constrain(motor2,50,255);

}else{

motor2=acelerador;

motor1=acelerador;

}

analogWrite(5,motor1);

analogWrite(6,motor2);

/\*

motor1=(angulo\*80/30)+175; //Le doy a motor1 el valor de potencia del motor5?.

motor1=constrain(motor1,100,255); //limito motor1 para que la potencia maxima y minima este entre -30ª y 30ª

motor2=255-motor1; //le doy a motor2 el valor que le corresponde para cumplir la condicion de que la media de los motores sea 200.

\*/

/\*

Serial.print("ax:");

Serial.print(ax); Serial.print("\t");

Serial.println(" ");

\*/

Serial.print(ayy); Serial.print(" ayy \t");

Serial.print(angulo); Serial.print(" angulo\t");

Serial.print(motor1); Serial.print(" motor1\t");

Serial.print(motor2); Serial.print(" motor2\t");

Serial.println(" ");

/\*

Serial.print("az:");

Serial.print(az); Serial.print("\t");

Serial.println(" ");

Serial.print("gx:");

Serial.print(gx); Serial.print("\t");

Serial.println(" ");

Serial.print("gy:");

Serial.print(gy); Serial.print("\t");

Serial.println(" ");

Serial.print("gz:");

Serial.print(gz); Serial.print("\t");

\*/

Serial.println(" ");

Serial.println(" ");

Serial.println(" ");

Serial.println(" ");

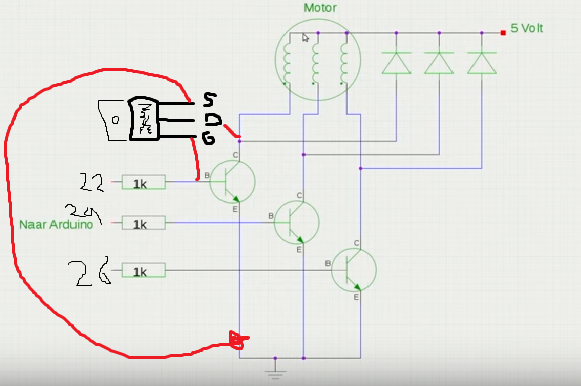
Serial.println(" ");

delay(150);

}

}

Motor brushless o paso a paso con alimentación externa:



int phase1 = 26;

int phase2 = 24;

int phase3 = 22;

int led = 13;

unsigned long stepLength = 50000;

int minStepLength = 100;

int steps = 100;

void setup() {

pinMode(led, OUTPUT);

pinMode(phase1, OUTPUT);

pinMode(phase2, OUTPUT);

pinMode(phase3, OUTPUT);

digitalWrite(led, LOW);

Serial.begin(9600);

}

void loop() {

switchStep(1);

switchStep(2);

switchStep(3);

Serial.print(stepLength); Serial.println(" stepLength");

if(stepLength > minStepLength)

{

stepLength = stepLength - steps;

} else {

// set the minimum pulse length

stepLength=minStepLength;

}

if (stepLength < 39950) {

digitalWrite(led, HIGH); // second gear

steps = 300;

}

if (stepLength < 20000) {

digitalWrite(led, LOW); // third gear

steps = 50;

}

if (stepLength < 3000) {

digitalWrite(led, HIGH); // fourth gear

steps = 2;

}

}

void switchStep(int stage)

{

switch(stage)

{

case 1:

//digitalWrite(phase1, HIGH);

//digitalWrite(phase2, LOW);

//digitalWrite(phase3, LOW);

digitalWrite(phase1, HIGH);

digitalWrite(phase2, LOW);

digitalWrite(phase3, LOW);

myDelay(stepLength);

break;

case 2:

//digitalWrite(phase1, LOW);

//digitalWrite(phase2, HIGH);

//digitalWrite(phase3, LOW);

digitalWrite(phase1, LOW);

digitalWrite(phase2, HIGH);

digitalWrite(phase3, LOW);

myDelay(stepLength);

break;

default:

//digitalWrite(phase1, LOW);

//digitalWrite(phase2, LOW);

//digitalWrite(phase3, HIGH);

digitalWrite(phase1, LOW);

digitalWrite(phase2, LOW);

digitalWrite(phase3, HIGH);

myDelay(stepLength);

break;

}

}

void myDelay(unsigned long p) {

if (p > 16380) {

delay (p/1000);

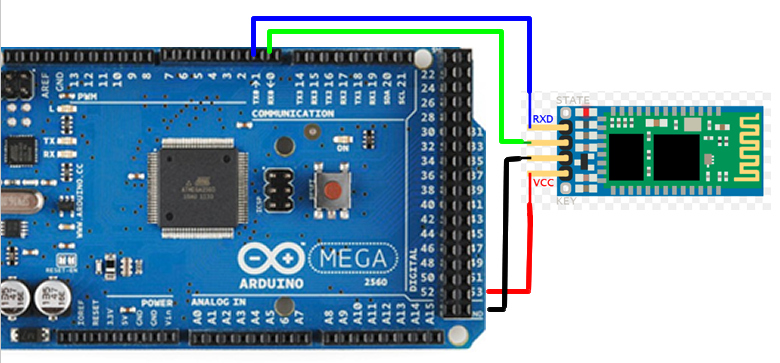
} else {

delayMicroseconds(p);

}

}

## INTERACION CON MODULO BLUETHOOT (CONFIGURACION MODULO)



Algunos comandos AT:  
AT+RESET - Reinicia el módulo  
AT+ORGL - Valores de fabrica  
AT+NAMExxxxx - Cambiar nombre  
AT+BAUDx - Cambiar velocidad  
AT+PINxxxx - Cambiar PIN de conexion

const int LED = 13;

const int BTPWR = 53;

char nombreBT[10] = "TechKrowdS";

char velocidad ='4'; //9600

char pin[5]= "0000";

void setup(){

pinMode(LED, OUTPUT);

pinMode(BTPWR, OUTPUT);

digitalWrite(LED, LOW);

digitalWrite(BTPWR, HIGH);

Serial.begin(9600);

Serial.print("AT");

delay(1000);

Serial.print("AT+NAME");

Serial.print(nombreBT);

delay(1000);

Serial.print("AT+BAUD");

Serial.print(velocidad);

delay(1000);

Serial.print("AT+PIN");

Serial.print(pin);

delay(1000);

digitalWrite(LED, HIGH);

}

void loop(){

}

## INTERACION CON MODULO BLUETHOOT (conectado al puerto serial trd y rxd)

//Projeto Arduino+Bluetooth+Android - Evandro Sperandiu Gonçaves

//evandrosg@gmail.com

char buffer[67];

byte Pino02 = 2;

byte Pino03 = 3;

byte Pino04 = 4;

void setup(){

Serial.begin(9600);

Serial.flush();

pinMode(Pino02, OUTPUT);

pinMode(Pino03, OUTPUT);

pinMode(Pino04, OUTPUT);

}

void loop(){

if (Serial.available()>0){

int index=0;

delay(100);//deixe o buffer encher

int numChar = Serial.available();

if(numChar>65){

numChar=65;

}

while(numChar--){

buffer[index++] = Serial.read();

}

splitString(buffer);

}

}

void splitString(char\* data){

Serial.print("Dados inseridos: ");

Serial.println(data);

char\* parameter;

parameter= strtok (data, " ,");

while(parameter != NULL){

setLED(parameter);

parameter = strtok (NULL, " ,");

}

//Limpa o texto e os buffers seriais

for (int x=0; x<66; x++){

buffer[x]='\0';

}

Serial.flush();

}

void setLED(char\* data){

//Se procede a parcear los datos ingresados.

Serial.print("Dados inseridos: ");

Serial.println(data[10]);

if ((data[10] =='a') || (data[10] == 'A')){

int Ans = strtol(data+1, NULL, 10);

Ans = constrain(Ans,0,1);

Serial.print("valor: ");

Serial.println(Ans);

digitalWrite(Pino02, Ans);

Serial.print("Pino02 esta em: ");

Serial.println(Ans);

}

if ((data[11] =='b') || (data[11] == 'B')){

int Ans = strtol(data+1, NULL, 10);

Ans = constrain(Ans,0,255);

analogWrite(Pino03, Ans);

Serial.print("Pino03 esta em: ");

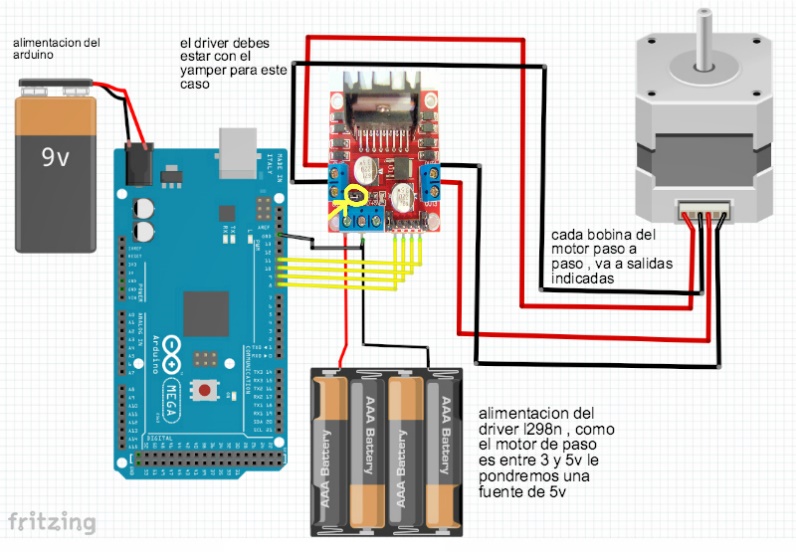
Serial.println(Ans);

}

}

## MOTORES PASO A PASO (Movimiento forma manual)

## [**Modulo L298n Puente H Driver**](http://articulo.mercadolibre.com.ar/MLA-618683287-modulo-l298n-puente-h-driver-motor-paso-a-paso-dc-insys-_JM)



/\* Motor Paso a Paso ajustado a grados

by: www.elprofegarcia.com

Gira los grados que se le indiquen por el monitor serial o bluetooth

Arduino Driver ULN200

8 IN1

9 IN2

10 IN3

11 IN4

Tienda en Linea: http://dinastiatecnologica.com/

\*/

int retardo=5; // Tiempo de retardo en milisegundos (Velocidad del Motor)

int dato\_rx; // valor recibido en grados

int numero\_pasos = 0; // Valor en grados donde se encuentra el motor

String leeCadena; // Almacena la cadena de datos recibida

void setup() {

Serial.begin(9600); // inicializamos el puerto serie a 9600 baudios

pinMode(11, OUTPUT); // Pin 11 conectar a IN4

pinMode(10, OUTPUT); // Pin 10 conectar a IN3

pinMode(9, OUTPUT); // Pin 9 conectar a IN2

pinMode(8, OUTPUT); // Pin 8 conectar a IN1

}

void loop() {

while (Serial.available()) { // Leer el valor enviado por el Puerto serial

delay(retardo);

char c = Serial.read(); // Lee los caracteres

leeCadena += c; // Convierte Caracteres a cadena de caracteres

}

if (leeCadena.length()>0){

dato\_rx = leeCadena.toInt(); // Convierte Cadena de caracteres a Enteros

Serial.print(dato\_rx); // Envia valor en Grados

Serial.println(" Grados");

delay(retardo);

dato\_rx = (dato\_rx \* 1.4222222222); // Ajuste de 512 vueltas a los 360 grados

}

while (dato\_rx>numero\_pasos){ // Girohacia la izquierda en grados

paso\_izq();

numero\_pasos = numero\_pasos + 1;

}

while (dato\_rx<numero\_pasos){ // Giro hacia la derecha en grados

paso\_der();

numero\_pasos = numero\_pasos -1;

}

leeCadena = ""; // Inicializamos la cadena de caracteres recibidos

apagado(); // Apagado del Motor para que no se caliente

} ///////////////////// Fin del Loop ///////////////////////////

void paso\_der(){ // Pasos a la derecha

digitalWrite(11, LOW);

digitalWrite(10, LOW);

digitalWrite(9, HIGH);

digitalWrite(8, HIGH);

delay(retardo);

digitalWrite(11, LOW);

digitalWrite(10, HIGH);

digitalWrite(9, HIGH);

digitalWrite(8, LOW);

delay(retardo);

digitalWrite(11, HIGH);

digitalWrite(10, HIGH);

digitalWrite(9, LOW);

digitalWrite(8, LOW);

delay(retardo);

digitalWrite(11, HIGH);

digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(8, HIGH);

delay(retardo);

}

void paso\_izq() { // Pasos a la izquierda

digitalWrite(11, HIGH);

digitalWrite(10, HIGH);

digitalWrite(9, LOW);

digitalWrite(8, LOW);

delay(retardo);

digitalWrite(11, LOW);

digitalWrite(10, HIGH);

digitalWrite(9, HIGH);

digitalWrite(8, LOW);

delay(retardo);

digitalWrite(11, LOW);

digitalWrite(10, LOW);

digitalWrite(9, HIGH);

digitalWrite(8, HIGH);

delay(retardo);

digitalWrite(11, HIGH);

digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(8, HIGH);

delay(retardo);

}

void apagado() { // Apagado del Motor

digitalWrite(11, LOW);

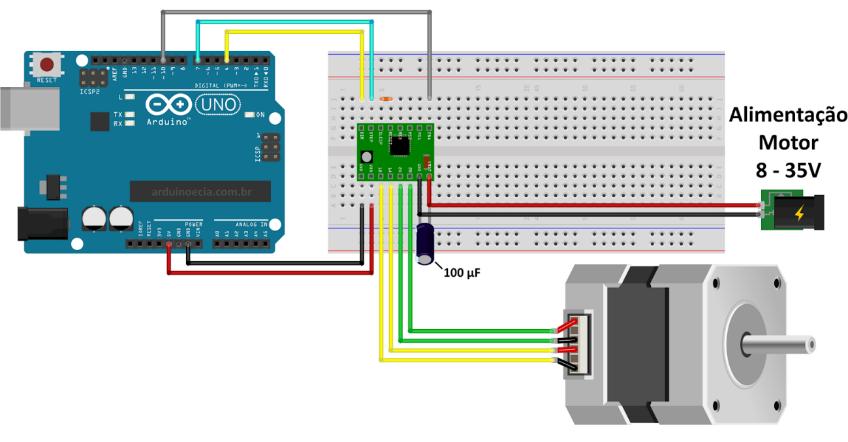
digitalWrite(10, LOW);

digitalWrite(9, LOW);

digitalWrite(8, LOW);

}

## MOTOR PASO A PASO (MODULO A4988)



// https://www.youtube.com/watch?v=gzNP6-GI7Cc&t=4s

// Gm25 Electrônica y Mas !!

// le presentamos un programa basico para el uso del dirver A4988

// dale me gusta a nuestros video nos ayudara de mucho, y suscribete

//para mas tutuoriales !!

const int steppin = 7; // declaramos el pin 10 del arduino como step

const int dirpin = 4; // declaramos el pin 12 del arduino con dir

int tiempo = 10; // declaramos la variable tiempo como (5 milisegundos) cuanto menos va mas rapido.

int pasos = 12; //propiedad del motor, en 48 pasos del motor da un giro completo.

int reset = 10;

int pasosTotal = 0;

int GiroDerecha = 1;

void setup() {

pinMode(steppin,OUTPUT); // declaramos como salidas

pinMode(dirpin,OUTPUT); // declaramos como salida

}

void loop() {

GiroDerecha = 1;

//digitalWrite(dirpin,GiroDerecha); // ponemos en alto o 1 logico a pin dir para que gire hacia un sentido

Girar(pasos,GiroDerecha);

delay(1000); // cuando termine el lazo for , descanzara 1 segundo o 1000 milisegundos

GiroDerecha = 0;

//digitalWrite(dirpin,GiroDerecha); // pones en bajo , o en 0 logico para que gire el el sentido opuesto

Girar(pasos\*2,GiroDerecha);

delay(5000);

}

void Girar(int pasosAdar, int GiroDerecha )

{

digitalWrite(dirpin,GiroDerecha);

for(int x = 0; x < pasosAdar; x++){ // creamos un lazo for , en nuetro caso el motor es de 200 paso, para dar una vuelta !

digitalWrite(steppin,HIGH); // las intrucciones que se encuntran dentro del lazo

delay(tiempo); // son las que aran que nuestro motor se mueva ,es la parte donde van los pulsos

digitalWrite(steppin,LOW); // necesarios para que el motor se mueva

delay(tiempo);

pasosTotal++;

}

}