```
#include "ros/ros.h"
#include "std_msgs/String.h"
#include "std_msgs/Float32.h"
#include "std_msgs/Int32.h"
#include "std_msgs/Int8.h"
#include <thread>
#include <unistd.h>
#include <iostream>
#include <chrono>
#include <signal.h>
#include <stdlib.h>
#include <stdio.h>
#include <move_base_msgs/MoveBaseAction.h>
#include <actionlib/client/simple_action_client.h>
#include <tf/transform_broadcaster.h>

#include "Arduino-Serial/ArduinoSerial.h"

using namespace std;
int colorSelect = 0;
int colorChoose = 0;
int matchStatus  = 0;
int gatepos = 0;
int flagpos = 0;
int rightSpeed = 0;
int leftSpeed = 0;
int leftWheelVal = 0;
int rightWheelVal = 0;

serialPort arduino("/dev/serial/by-id/usb-1a86_USB2.0-Serial-if00-port0");

void rin(const std_msgs::Int8ConstPtr &msg){
  rightSpeed = (int)msg->data;
}

void lin(const std_msgs::Int8ConstPtr &msg){
  leftSpeed = (int)msg->data;
}


void flagFunc(const std_msgs::Int32ConstPtr &msg){
  flagpos = (int)msg->data;
    arduino.moveFlag(flagpos);
}
void gateFunc(const std_msgs::Int32ConstPtr &msg){
  gatepos = (int)msg->data;
    arduino.moveFlag(gatepos);

}
void leftWheelFunc(const std_msgs::Int32ConstPtr &msg){
    leftWheelVal = msg->data;
}
void rightWheelFunc(const std_msgs::Int32ConstPtr &msg){
    rightWheelVal = msg->data;
}


int main(int argc, char **argv){
    ros::init(argc, argv, "arduino_talker");
    ros::NodeHandle n;
    ros::Rate loop_rate(40);  //1 Hz
    ros::Subscriber lsub = n.subscribe<std_msgs::Int8>("rmotor_cmd", 1,lin);
    ros::Subscriber rsub = n.subscribe<std_msgs::Int8>("lmotor_cmd", 1,rin);
    ros::Subscriber flag = n.subscribe<std_msgs::Int32>("flag_cmd", 1,flagFunc);
    ros::Subscriber gate = n.subscribe<std_msgs::Int32>("gate_cmd", 1,gateFunc);
    ros::Subscriber leftWheel = n .subscribe<std_msgs::Int32>("lwheel",1,leftWheelFunc);
    ros::Subscriber rightWheel = n .subscribe<std_msgs::Int32>("rwheel",1,rightWheelFunc);

    ros::Publisher colorSelectPub = n.advertise<std_msgs::Int32>("colorSelectFunc",1,true);
    ros::Publisher startMatchPub = n.advertise<std_msgs::Int32>("startMatchFunc",1,true);
    ros::Time current_time = ros::Time::now();

    while(ros::ok()) {
        if(matchStatus == 1){
            startMatchPub.publish(matchStatus);
            matchStatus++;
        }

        std_msgs::String msg;
      msg.data = std::string("Hello Fuck Shits");


  if(arduino.getButtonState()){
    startMatchPub.publish(1);
  }
        ros::spinOnce();
    arduino.updateLCD("Fuck");
    arduino.drive(rightSpeed,leftSpeed);
    arduino.updateArduino();
    loop_rate.sleep();
    }
    return 0;
}

```