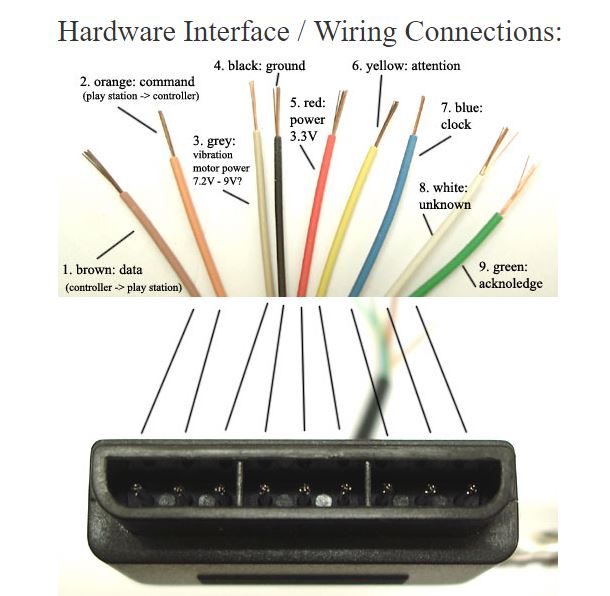
PS2 CONTROL WITH ARDUINO :



Wire colors and its functions:

There are 9 wires and 6 wires are needed at least to communicate with the controller.(clock,data,power,ground,attention).For operating vibration motors motor power is to be given separately.

1. **Brown - Data:** Controller -> PlayStation. This is an [open collector output](http://en.wikipedia.org/wiki/Open_collector) and requires a pull-up resistor (1 to 10k, maybe more). (A pull-up resistor is needed because the controller can only connect this line to ground; it can't actually put voltage on the line).
2. **Orange - Command:** PlayStation -> Controller.
3. **Grey - Vibration Motors Power:** 6-9V? With no controller connected, this meausures about 7.9V, with a controller, 7.6V, most websites say this is 9V (except [playstation.txt](http://www.raphnet.net/electronique/psx_adaptor/Playstation.txt) -> 7.6V), although it will still drive the motors down around 4V, although somewhat slower. When the motors are first engaged, almost 500mA is drawn on this line, and at steady state full power, ~300mA is drawn.
4. **Black - Ground**
5. **Red - Power:** Many sites label this as 5V, and while this may be true for Play Station 1 controllers, we found several wireless brands that would only work at 3.3V. Every controller tested worked at 3.3V, and the actual voltage measured on a live Playstation talking to a controller was 3.4V. [McCubbin](http://www.gamesx.com/controldata/psxcont/psxcont.htm) says that any official Sony controller should work from 3-5V. Most sites say there is a 750mA fuse for both controllers and memory cards, although this may only apply to PS1's since 4 dual shock controllers could exceed that easily.
6. **Yellow - Attention:** This line must be pulled low before each group of bytes is sent / received, and then set high again afterwards. In our testing, it wasn't sufficient to tie this permanently low--it had to be driven down and up around each set. [Digitan](http://www.geocities.com/digitan000/Hardware/22/e22_page.html) considers this a "Chip Select" or "Slave Select" line that is used to address different controllers on the same bus.
7. **Blue - Clock:** 500kH/z, normally high on. The communication appears to be [SPI](http://en.wikipedia.org/wiki/Serial_Peripheral_Interface_Bus)bus. We've gotten it to work from less than 100kHz up through 500kHz (500k bits / second, not counting delays between bytes and packets). When the guitar hero controller is connected, the clock rate is 250kHz, which is also the rate the playstation 1 uses.
8. **White - Unknown**
9. **Green - Acknowledge:** This normally high line drops low about 12us after each byte for half a clock cycle, but not after the last bit in a set. This is a [open collector output](http://en.wikipedia.org/wiki/Open_collector) and requires a pull-up resistor (1 to 10k, maybe more). [playstation.txt](http://www.raphnet.net/electronique/psx_adaptor/Playstation.txt) says that the playstation will consider the controller missing if the ack signal (> 2us) doesn't come within 100us.

This ZIP includes everything you need to get that dusty PS2 Controller talking to your Arduino!

Download PS2X master library

<https://drive.google.com/file/d/0Bzuz9Z6Ay9jzSVkzN3RsYzRPOE0/view?usp=sharing>

Pins 8 and 9 are usually not used . For the 3rd wire which is for vibration is optional uses 7.2V-9V

But we can use 5V from the Arduino. NOTE: Only pin 3 can take over 3.3V.

NOTE: The VCC max for controller is 3.3V.

So we can use female wires for connection .

STEPS:

First of all we connect the VCC and ground. Check if the controller gets switched ON by the indication light.

Next Command wire ->Arduino pin11

Attention ->Arduino Pin 10

PS2 clock ->Arduino Pin 12

**Remember we have to add a pull up resistor to the data line . For this the data line (close to 3.3V) connect it on the breadboard or dot board and add a resistor(10K about) from the 3.3 V.After connecting the pull up resistor connect the data line to pin 8.**

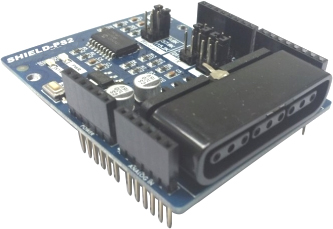
**FOR INSTALLATION AND CONTROLLING OF BOT :**

<https://www.youtube.com/watch?v=xlupRVF_6W8>

**Also we can use the PS2 shield(PS2 shield for Arduino)**

Shield to connect wired or wireless remote control to Arduino main boards  
• Offers a standard connector for SONY PS2 controller to plug-in  
• 5V powered for low current consumption  
• Simple to use UART protocol

The **PS2 Shield for Arduino** is an Arduino compatible shield which is compatible with Arduino UNO, Arduino Duemilanove, Arduino Mega, Arduino Leonardo and possibly other pin compatible main boards. Cytron PS2 Shield offers a compact yet reliable PS2 Controller Converter for user. Cytron PS2 Shield is powered from Arduino main board. with Cytron PS2 Shield Reading Joy-stick and button’s state of PS2 controller will be as easy as reading UART data. It offers a standard connector for SONY PS2 controller to plug-in, either wired or wireless.



**Features**  
• Vibrator motor on PS2 is controllable  
• Wired and Wireless PS2 controller is supported  
• PS2 Controller will automatically operate in analog mode  
• Status LED  
• Jumper selector to select different UART Baud Rate (4800, 9600, 57600, 115200)  
• Jumper selectors to select different digital pin as UART TX and RX pin

**Hardware Introduction**

In this project, [Cytron PS2 Shield](http://www.cytron.com.my/p-shield-ps2) will be used to create an interface between the CT-UNO and the PS2 Controller. Cytron PS2 Shield enables the user to utilize all the buttons from the PS2 controller including the two joystick. By using this shield, decoding the PS2 controller will be as easy as reading UART data. Furthermore, this shield allows the user to use either the hardware serial from Arduino itself (Pin 0 and 1) or software serial (pin 2,3,8,9,10 and 11).

Did I mention that decoding data from PS2 controller is also an easy job. Here, we provide you with the Cytron PS2 Shield library, which helps you to read and write to the controller easily. You can get the library from [here](https://github.com/CytronTechnologies/Cytron_PS2Shield).

In this article, I am going to divide it into two parts. The first part explains on how to use the Cytron PS2 Shield library and decodes all the button from PS2 controller. In the subsequent part, I will be explaining how you can use the PS2 to control you Arduino Edubot.

**Part 1: Using Cytron PS2 Shield Library**

Before I start explaining about the library, make sure you have downloaded the library. Go to [here](https://github.com/CytronTechnologies/Cytron_PS2Shield), then download the zip file of library. Make sure you install the library into you Arduino IDE. While working with the PS2 shield, it is a good practice to refer the [User’s Manual](https://docs.google.com/document/d/1cifT1XqOquLffnu5J961wpKakSO8_LpV35v4sKSaIQU/view) to understand more.

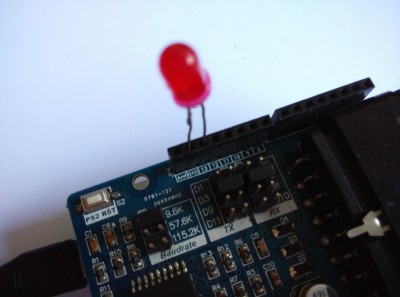
**Circuit Setup:**

For the first part, the setup is quite simple, just attach the Cytron PS2 shield onto the CT-UNO. Next, connect your PS2 controller to the PS2 connector of the shield.



**1. To read the buttons from the PS2 controller**

For this simple example, I have connected a LED on pin 13 of Arduino as the indicator.



To read the digital buttons of PS2 controller, you can use the function:

readButton(button);

The *button* is the key you are interested to read from the PS2 controller. The list of available buttons are shown in table below:

|  |  |  |  |
| --- | --- | --- | --- |
| PS2\_SELECT | PS2\_JOYSTICK\_LEFT | PS2\_JOYSTICK\_RIGHT | PS2\_START |
| PS2\_LEFT | PS2\_RIGHT | PS2\_UP | PS2\_DOWN |
| PS2\_LEFT1 | PS2\_LEFT2 | PS2\_RIGHT1 | PS2\_RIGHT2 |
| PS2\_TRIANGLE | PS2\_CIRCLE | PS2\_CROSS | PS2\_SQUARE |

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/joystick.jpg)Button Label

Below is the code for this example:

#include <SoftwareSerial.h>

#include <Cytron\_PS2Shield.h>

Cytron\_PS2Shield ps2(2,3);

#define led 13

void setup()

{

ps2.begin(9600);

pinMode(led,OUTPUT);

digitalWrite(led,LOW);

}

void loop()

{

if(ps2.readButton(PS2\_SELECT)==0)

{

digitalWrite(led,HIGH);

}

else

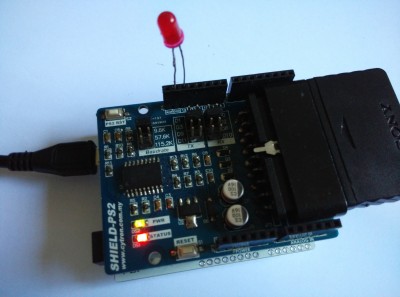
{

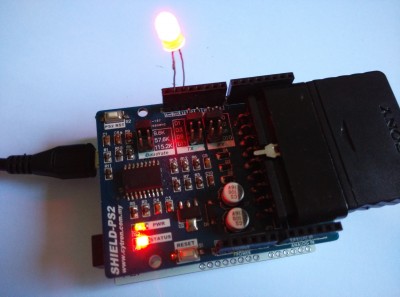
digitalWrite(led,LOW);

}

}

In this example, whenever you pressed the SELECT button on the PS2 controller, it will light up the LED on pin 13.





**2. To read the analog joystick from PS2 controller**

For this example, I will show you how to read the analog value from the joystick. For your information, the joystick is simply a 2 axispotentiometer inside. Therefore, the shield will decode it and return analog value ranged between 0 to 255.

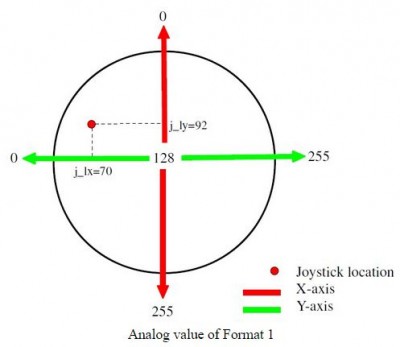
To read the analog joystick value, you can use the function similar to previous example, except a little changes on the *button* variable.

readButton(button)

The available joystick *button* are:

|  |  |  |
| --- | --- | --- |
| PS2\_JOYSTICK\_LEFT\_X\_AXIS | PS2\_JOYSTICK\_LEFT\_Y\_AXIS | PS2\_JOYSTICK\_RIGHT\_X\_AXIS |
| PS2\_JOYSTICK\_RIGHT\_Y\_AXIS | PS2\_JOYSTICK\_RIGHT\_UP | PS2\_JOYSTICK\_RIGHT\_DOWN |
| PS2\_JOYSTICK\_RIGHT\_LEFT | PS2\_JOYSTICK\_RIGHT\_RIGHT | PS2\_JOYSTICK\_LEFT\_UP |
| PS2\_JOYSTICK\_LEFT\_DOWN | PS2\_JOYSTICK\_LEFT\_LEFT | PS2\_JOYSTICK\_LEFT\_RIGHT |

The analog output format of joystick is illustrated by the figure below:

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/joystick_output_format.jpg)

Below is the code for this example:

#include <SoftwareSerial.h>

#include <Cytron\_PS2Shield.h>

Cytron\_PS2Shield ps2(2,3);

void setup()

{

ps2.begin(9600);

Serial.begin(9600);

}

void loop()

{

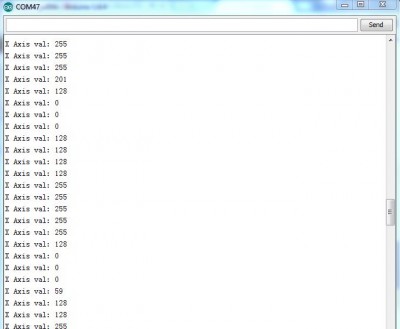
Serial.print("X Axis val: ");

Serial.println(ps2.readButton(PS2\_JOYSTICK\_LEFT\_X\_AXIS));

delay(50);

}

The results can be viewed using the Arduino’s Serial Monitor. When the left joystick is push towards the x axis, the output value will change accordingly.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/Capture.jpg)

**3. To control the vibrator motor from PS2 controller**

In the last example, I will show you how you can control the vibrator motor in the PS2 controller using your Arduino.

\* I need to highlight, not all PS2 controller comes with vibrator motor, so don be surprise if this code cannot works.

To control vibrator motor, you can use this function:

vibrate(motor,value);

Below is the code for this example:

#include <SoftwareSerial.h>

#include <Cytron\_PS2Shield.h>

Cytron\_PS2Shield ps2(2,3);

void setup()

{

ps2.begin(9600);

}

void loop()

{

/\* Function: vibrate(motor, value);

motor = 0; value = 0 to 255 //bigger vibrator

motor = 1; value = 0 or 1 //smaller vibrator

\*/

ps2.vibrate(1,1);

delay(500);

ps2.vibrate(1,0);

delay(500);

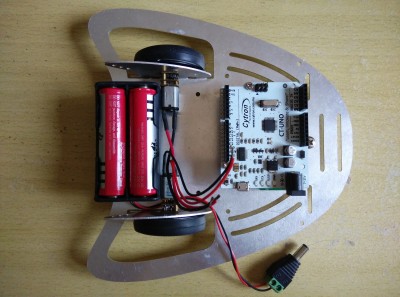
}

**Part 2: Integrating Cytron PS2 Shield with Arduino Edubot**

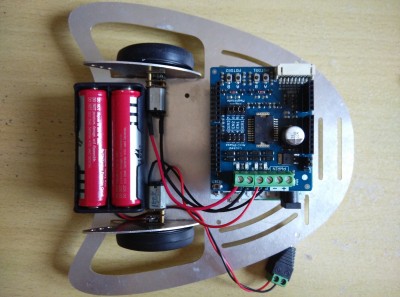
For this part, we will combine both the PS2 Shield and the Arduino Edubot. The PS2 controller will be used to control the movement of the Edubot.

**Project Setup:**

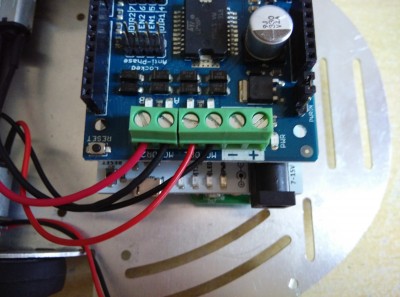
1. First, attach the CT-UNO to the aluminium chassis.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.04.43.jpg)

2. Next, add the 2A Motor Driver Shield onto the CT-Uno.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.05.47.jpg)

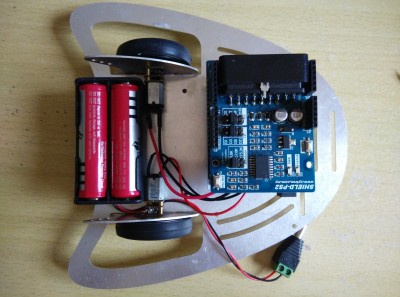
3. Connect the motor connection to the motor shield. Motor 1 for right wheel, motor 2 for left wheel.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.05.50.jpg)

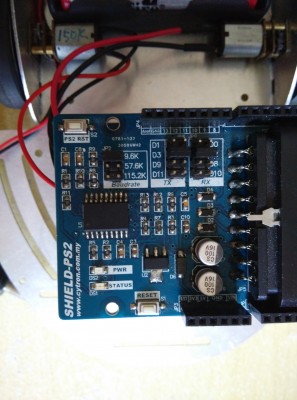
4. Make sure the selection jumper are connected for the **signed magnitude** option.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.05.54.jpg)

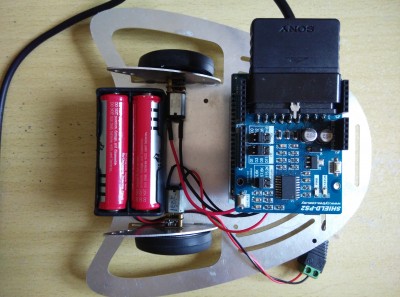
5. After that, stack the PS2 shield on top of the motor shield.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.06.22.jpg)

6. Make sure the selection jumper are connected for **9.6k Baud rate**, **D2 for Rx** and **D3 for Tx**.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.06.27.jpg)

7. Connect the PS2 controller to the PS2 shield.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.06.38.jpg)

8. Finally, insert the DC jack to power up the robot.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.06.48.jpg)

This is how the overall robot looks like.

[](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/2015-07-06-15.10.32.jpg)

After setting up the hardware, let us move on to the software part of the Arduino Edubot. Here I have prepared the example sketch, [PS2\_Edubot\_Joystick](http://tutorial.cytron.com.my/wp-content/uploads/2015/07/PS2_Edubot_Joystick.rar) which you can used for this project. This sketch will read the buttons (TRIANGLE, SQUARE, CIRCLE, CROSS) and the right joystick to control the movement.

REFERENCES:

<http://tutorial.cytron.com.my/2015/07/23/using-cytron-ps2-shield-with-arduino-edubot-2/>

<https://store.open-electronics.org/PS2SHIELD>

Features:

**Features:**

* 5V powered, low current consumption.
* Simple to use UART protocol
* Vibrator motor on PS2 is controllable.
* Wired and Wireless PS2 controller is supported.
* PS2 Controller will automatically operate in analog mode.
* A status LED
* Jumper selector to select different UART Baud Rate (9600, 57600, 115200).
* Jumper selectors to select different digital pin as UART TX and RX pin.