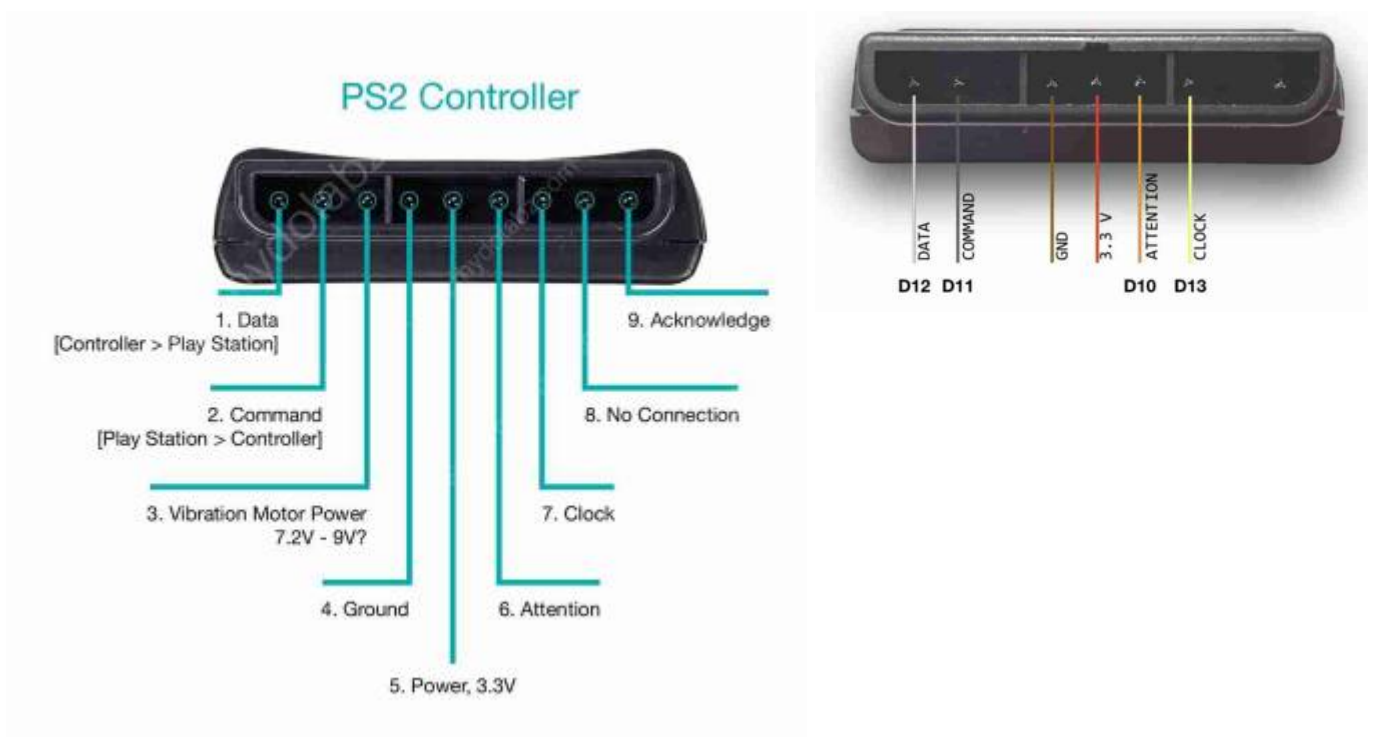


Manettes PS2 et Arduino

<https://www.rhydolabz.com/wiki/?p=12663>

La manette sans fil PS2 est une manette standard pour la PlayStation 2 et est identique à la manette DualShock d'origine pour la console PlayStation. Il comporte douze boutons analogiques (sensibles à la pression) (X, O, Π, Δ, L1, R1, L2, R2, Haut, Bas, Gauche et Droite), cinq boutons numériques (L3, R3 Start, Select et le mode analogique bouton) et deux sticks analogiques. Le contrôleur comporte également deux moteurs de vibration, celui de gauche étant plus gros et plus puissant que celui de droite. Il est alimenté par deux piles AAA. Il communique avec la console en utilisant le protocole RF 2,4 GHz.



Explications broches de sorties

1. - DONNÉES : il s'agit de la ligne de données entre la manette et la PS2. Il s'agit d'une sortie à collecteur ouvert et nécessite une résistance pull-up (1 à 10k, peut-être plus). (Une résistance de rappel est nécessaire car le contrôleur ne peut connecter cette ligne qu'à la terre ; il ne peut pas réellement mettre de tension sur la ligne).
2. - COMMANDE : Il s'agit de la ligne de données de la PS2 à la manette.
3. - PUISSANCE DU MOTEUR DE VIBRATION
4. - GND : masse
5. - VCC : VCC peut varier de 5 V à 3 V.
6. - ATT : ATT est utilisé pour attirer l'attention du contrôleur. Cette ligne doit être tirée vers le bas avant que chaque groupe d'octets ne soit envoyé/reçu, puis replacée vers le haut par la suite. Cette broche est considérée comme une ligne "Chip Select" ou "Slave Select" qui est utilisée pour adresser différents contrôleurs sur le même bus.
7. - CLK : 500 kHz, normalement activé. La communication semble être un bus SPI.
8. - Pas connecté

Nous avons interfacé la manette PS2 avec un Arduino. À chaque pression sur un bouton, l'Arduino reçoit le signal PS2 et l'affiche sur l'interface Serie. Nous avons suivi le protocole standard PS2 pour réaliser l'algorithme de communication, identique au protocole SPI. Notre programme sur l'arduino détecte et lit uniquement les pressions sur les boutons, les valeurs de pression ne sont pas lues. Les valeurs d'état du stick analogique sont affichées en continu sur l'interface serie.

Détails de connexion :

La ligne CLK et les lignes ATT du récepteur PS2 sont maintenues normalement hautes. L'ATT fonctionne comme la ligne Slave Select sous SPI. Vous le tirez vers le bas pour dire au contrôleur que vous lui parlez, puis le renvoyez vers le haut une fois qu'un cycle de communication est terminé. CMD est la ligne de données vers le contrôleur et DATA est les données provenant du contrôleur. Ici, dans notre application, nous n'utilisons pas la broche de reconnaissance.

PS2_Arduino.ino

```
#include <PS2X_lib.h>                                /* PS2 Controller Library
*/
PS2X ps2x;                                           /* create PS2 Controller
Class*/
byte Type = 0;
byte vibrate = 0;
int rx=0,ry=0,lx=0,ly=0;

void setup(){
  Serial.begin(9600);
  ps2x.config_gamepad(13,11,10,12, true, true); /* setup pins and
settings: GamePad(clock, command, attention, data, Pressures?,
Rumble?) check for error*/
  Type = ps2x.readType();                            /* Reading type of the
PS2 Ccontroller */
  if(Type==1){                                       /* Type 1 is Duel
shock controller */

    Serial.println("  DualShock  ");                /* display if the
controller is duel shock*/

    Serial.println("Controller Found");
    delay(1000);

  }
}
void loop(){

  ps2x.read_gamepad(false, vibrate); /* read controller and set
large motor to spin at 'vibrate' speed */

  Serial.println("Stick values:  ");                /* Display analog stick
values */
```

```
ly = ps2x.Analog(PSS_LY);
lx = ps2x.Analog(PSS_LX);
ry = ps2x.Analog(PSS_RY);
rx = ps2x.Analog(PSS_RX);

if((ly <= 9))
checking less than 10 */
    Serial.println("00");
display as "005" in Serial */
    if((ly >= 9 && ly <= 99))
checking between 10-99 */
        Serial.print("0");
display as "055" in Serial */
        Serial.println(ly,DEC);
Y axis */
        Serial.print(",");
*/
    if((lx <= 9))
checking less than 10 */
        Serial.print("00");
display as "005" in Serial */
        if((lx >= 9 && lx<=99))
checking between 10-99 */
            Serial.println("0");
display as "055" in Serial */
            Serial.println(lx,DEC);
X axis */
            Serial.print(",");
*/
    if((ry <= 9))
checking less than 10 */
        Serial.println("00");
display as "005" in Serial */
        if((ry >= 9 && rx<=99))
checking between 10-99 */
            Serial.println("0");
display as "055" in Serial */
            Serial.println(ry,DEC);
stick Y axis */
            Serial.println(",");
comma */
    if((rx <= 9))
checking less than 10 */
        Serial.println("00");
display as "005" in Serial */
        if((rx >= 9 && rx <= 99))
checking between 10-99 */
            Serial.println("0");
display as "055" in Serial */

/* Reading Left stick Y axis */
/* Reading Left stick X axis */
/* Reading Right stick Y axis */
/* Reading Right stick X axis */

/* standardize to 3 digit by
    /* eg: if ly= 5 then it

/* standardize to 3 digit by
    /* eg: if ly= 55 then it

/* display left analog stick

/* separate values using comma

/* standardize to 3 digit by
    /* eg: if lx= 5 then it

/* standardize to 3 digit by
    /* eg: if lx= 55 then it

/* display left analog stick

/* separate values using comma

/* standardize to 3 digit by
    /* eg: if rx= 5 then it

/* standardize to 3 digit by
    /* eg: if rx= 55 then it

/* display Right analog

/* separate values using

/* standardize to 3 digit by
    /* eg: if RX= 5 then it

/* standardize to 3 digit by
    /* eg: if RX= 55 then it
```

```
Serial.println(rx,DEC);                                /* display Right analog
stick X axis */
Serial.println(" ");
if(ps2x.NewButtonState()) {                            /* will be TRUE if any button
changes state */

    if(ps2x.Button(PSB_START))                        /* will be TRUE as long START
button is pressed */
        Serial.println("START PRESSED ");
    if(ps2x.Button(PSB_SELECT))                      /* will be TRUE as long
SELECT button is pressed */
        Serial.println("SELECT PRESSED ");
    if(ps2x.Button(PSB_PAD_UP))                      /* will be TRUE as long
as UP button is pressed */
        Serial.println("UP PRESSED ");
    if(ps2x.Button(PSB_PAD_RIGHT))                   /* will be TRUE as long
as UP button is pressed */
        Serial.println("RIGHT PRESSED ");
    if(ps2x.Button(PSB_PAD_LEFT))                   /* will be TRUE as long
as LEFT button is pressed */
        Serial.println("LEFT PRESSED ");
    if(ps2x.Button(PSB_PAD_DOWN))                   /* will be TRUE as long
as DOWN button is pressed */
        Serial.println("DOWN PRESSED ");
    if(ps2x.Button(PSB_L1))                          /* will be TRUE as long
as L1 button is pressed */
        Serial.println("L1 pressed ");
    if(ps2x.Button(PSB_R1))                          /* will be TRUE as long
as R1 button is pressed */
        Serial.println("R1 pressed ");
    if(ps2x.Button(PSB_L2))                          /* will be TRUE as long
as L2 button is pressed */
        Serial.println("L2 pressed ");
    if(ps2x.Button(PSB_R2))                          /* will be TRUE as long
as R2 button is pressed */
        Serial.println("R2 pressed ");
    if(ps2x.Button(PSB_L3))                          /* will be TRUE as long
as L3 button is pressed */
        Serial.println("L3 pressed ");
    if(ps2x.Button(PSB_R3))                          /* will be TRUE as long
as R3 button is pressed */
        Serial.println("R3 pressed ");
    if(ps2x.Button(PSB_GREEN))                      /* will be TRUE as long
as GREEN/Triangle button is pressed */
        Serial.println("Triangle pressed");
    if(ps2x.Button(PSB_BLUE))                      /* will be TRUE as long
as BLUE/CROSS/X button is pressed */
        Serial.println("X pressed ");
    if(ps2x.Button(PSB_RED))                        /* will be TRUE as long
as RED/Circle button is pressed */
        Serial.println("Circle pressed ");
```

```
if(ps2x.Button(PSB_PINK)) /* will be TRUE as long  
as PINK/Square button is pressed */  
    Serial.println("Square pressed ");  
    delay(700);  
}  
else;  
}
```

Liens Manette PS2 <--> Arduino

[manette-ps2-et-arduino-ps2-controler](#)

[Librairie Arduino-PS2X](#)

PS2 - Touches



Les fonctions les plus pratiques de cette librairie sont :

- **ps2x.config_gamepad(clock, command, attention, data, Pressures? Rumble?);**

La fonction définit la broche du contrôleur et la sensibilité à la pression et aux vibrations des moteurs. Si vous voulez que les touches soient insensibles à la pression ou que les moteurs n'aient pas de vibrations, définissez Pressures et Rumble sur false. Cette fonction renvoie une valeur pour l'erreur

- `prêt();` fonction détermine le type de contrôleur détecté. 0 signifie que le contrôleur n'est pas détecté correctement, 1 signifie la détection du contrôleur DualShock et 2 signifie la détection du contrôleur GuitarHero.
- `read_gamepad(boolean motor1, byte motor2);` fonction commencer à lire l'état des touches lorsque l'état de la vibration du moteur est déterminé. (le moteur 2 est le plus gros.)
- `Bouton (mais tapez);` la fonction renvoie 1 lorsque la touche spécifique de l'argument de la fonction est enfoncée. Dans DualShock, les touches du contrôleur sont nommées comme suit :

Key	Function	Digital/Analog
PSB_SELECT	OK	Digital
PSB_START	OK	Digital
PSB_PAD_UP	UP	Analog
PSB_PAD_DOWN	DOWN	Analog
PSB_PAD_LEFT	LEFT	Analog
PSB_PAD_RIGHT	RIGHT	Analog
PSB_BLUE	X	Analog
PSB_GREEN	Triangle	Analog
PSB_PINK	Square	Analog
PSB_RED	Circle	Analog
PSB_L3	L3	Digital
PSB_R3	R3	Digital
PSB_L2	L2	Analog
PSB_R2	R2	Analog
PSB_L1	L1	Analog
PSB_R1	R1	Analog
PSB_RX	Joystick right x	Analog
PSB_RY	Joystick right y	Analog
PSB_LX	Joystick left x	Analog
PSB_LY	Joystick left y	Analog

From:

<https://chanterie37.fr/fablab37110/> - **Castel'Lab le Fablab MJC de Château-Renault**

Permanent link:

https://chanterie37.fr/fablab37110/doku.php?id=start:arduino:manette_ps2&rev=1659283775



Last update: **2023/01/27 16:08**