PS3 Controller/BD Remote

https://pimylifeup.com/raspberry-pi-playstation-controllers/

https://kodi.wiki/view/HOW-TO:Set_up_PS3_BD_Remote

https://koditips.com/retro-games-kodi/

https://www.google.com/url?sa=t&source=web&rct=j&url=https://m.youtube.com/watch%3Fv%3Di3oZfa6LAMo&ved=2ahUKEwiZuvqGsuLhAhUJXRUIHbETA4gQwqsBMAB6BAgDEAU&usg=AOvVaw3Lb7s-i7zTo-B97t5nfQeq

£12 Third party PS3 controller (SHANWAN) from Amazon:
https://www.amazon.co.uk/gp/product/B07L6WX6JS

The DualShock 3, DualShock 4 and Sixaxis controllers work out of the box when plugged in via USB (the PS button will need to be pushed to begin).

The generic Bluetooth driver is the btusb kernel module. To load:

```
sudo modprobe btusb
sudo modprobe bluetooth
```

check modprobe blacklist in case bluetooth was disabled /etc/modprobe.d/blacklist.conf ##Disable Bluetooth #blacklist btusb #blacklist bluetooth

connect via USB

Install the bluez and bluez-tools packages, which includes the sixaxis plugin. Bluez v5.48 and kernel 4.15 is required which include patches for third party PS3 controllers.

```
sudo apt-get install bluez bluez-tools
```

Then start bluetooth.service,

```
service bluetooth start
```

plug the controller in via USB, and the plugin should program your PC's bluetooth address into the controller automatically. If not, the sixpair.c file can be compiled and used.

You can now disconnect your controller. The next time you hit the PlayStation button it will connect without asking anything else.
Alternatively, you can hold the select button and the PlayStation button simultaneously (for a few seconds) to put the gamepad in pairing mode, and pair as you would normally.
Remember to disconnect the controller when you are done as the controller will stay on when connected and drain the battery.
Note: If the controller does not connect, make sure the bluetooth interface is turned on and the controllers have been trusted. (See Bluetooth)

```
sudo bluetoothctl
```

```
scan on
```

```
sudo apt-get install libusb-dev libusb-0.1-4
gcc -o sixpair sixpair.c -lusb
sudo ./sixpair
No controller found on USB busses.

# plug in controller via usb

sudo ./sixpair
Current Bluetooth master: d0:d1:d2:d3:d4:d5
Setting master bd_addr to 30:e3:7a:d3:50:a3

# disconnect from usb

sudo bluetoothctl
power on
discoverable on
advertise on
scan on
#switch on controller via PS button
[NEW] Device 04:11:5D:90:63:25 PLAYSTATION(R)3Conteroller-PANHAI
trust 04:11:5D:90:63:25
connect 04:11:5D:90:63:25
pair 04:11:5D:90:63:25
```

connect via usb, wait 15 seconds
disconnect from usb
press select button while all leds flashing
paired up, one led flashing only

sixpair.c

```
/*
 * sixpair.c version 2007-04-18
 * Compile with: gcc -o sixpair sixpair.c -lusb
 */

#include <string.h>
#include <unistd.h>
#include <stdio.h>
#include <usb.h>

#define VENDOR 0x054c
#define PRODUCT 0x0268

#define USB_DIR_IN 0x80
#define USB_DIR_OUT 0

void fatal(char *msg) { perror(msg); exit(1); }
```

```
void show_master(usb_dev_handle *devh, int itfnum) {
  printf("Current Bluetooth master: ");
  unsigned char msg[8];
  int res = usb_control_msg
    (devh, USB_DIR_IN | USB_TYPE_CLASS | USB_RECIP_INTERFACE,
     0x01, 0x03f5, itfnum, (void*)msg, sizeof(msg), 5000);
  if ( res < 0 ) { perror("USB_REQ_GET_CONFIGURATION"); return; }
  printf("%02x:%02x:%02x:%02x:%02x:%02x\n",
     msg[2], msg[3], msg[4], msg[5], msg[6], msg[7]);
}

void set_master(usb_dev_handle *devh, int itfnum, int mac[6]) {
  printf("Setting master bd_addr to %02x:%02x:%02x:%02x:%02x:%02x\n",
     mac[0], mac[1], mac[2], mac[3], mac[4], mac[5]);
  char msg[8]= { 0x01, 0x00, mac[0],mac[1],mac[2],mac[3],mac[4],mac[5]
};
  int res = usb_control_msg
    (devh,
     USB_DIR_OUT | USB_TYPE_CLASS | USB_RECIP_INTERFACE,
     0x09,
     0x03f5, itfnum, msg, sizeof(msg),
     5000);
  if ( res < 0 ) fatal("USB_REQ_SET_CONFIGURATION");
}

void process_device(int argc, char **argv, struct usb_device *dev,
            struct usb_config_descriptor *cfg, int itfnum) {
  int mac[6], have_mac=0;

  usb_dev_handle *devh = usb_open(dev);
  if ( ! devh ) fatal("usb_open");

  usb_detach_kernel_driver_np(devh, itfnum);

  int res = usb_claim_interface(devh, itfnum);
  if ( res < 0 ) fatal("usb_claim_interface");

  show_master(devh, itfnum);

  if ( argc >= 2 ) {
    if ( sscanf(argv[1], "%x:%x:%x:%x:%x:%x",
        &mac[0],&mac[1],&mac[2],&mac[3],&mac[4],&mac[5]) != 6 ) {

      printf("usage: %s [<bd_addr of master>]\n", argv[0]);
      exit(1);
    }
  } else {
    FILE *f = popen("hcitool dev", "r");
    if ( !f ||
     fscanf(f, "%*s\n%*s %x:%x:%x:%x:%x:%x",
        &mac[0],&mac[1],&mac[2],&mac[3],&mac[4],&mac[5]) != 6 ) {
```

```
      printf("Unable to retrieve local bd_addr from `hcitool dev`.\n");
      printf("Please enable Bluetooth or specify an address
manually.\n");
      exit(1);
    }
    pclose(f);
  }
  set_master(devh, itfnum, mac);

  usb_close(devh);
}

int main(int argc, char *argv[]) {

  usb_init();
  if ( usb_find_busses() < 0 ) fatal("usb_find_busses");
  if ( usb_find_devices() < 0 ) fatal("usb_find_devices");
  struct usb_bus *busses = usb_get_busses();
  if ( ! busses ) fatal("usb_get_busses");

  int found = 0;

  struct usb_bus *bus;
  for ( bus=busses; bus; bus=bus->next ) {
    struct usb_device *dev;
    for ( dev=bus->devices; dev; dev=dev->next) {
      struct usb_config_descriptor *cfg;
      for ( cfg = dev->config;
        cfg < dev->config + dev->descriptor.bNumConfigurations;
        ++cfg ) {
    int itfnum;
    for ( itfnum=0; itfnum<cfg->bNumInterfaces; ++itfnum ) {
      struct usb_interface *itf = &cfg->interface[itfnum];
      struct usb_interface_descriptor *alt;
      for ( alt = itf->altsetting;
        alt < itf->altsetting + itf->num_altsetting;
        ++alt ) {
        if ( dev->descriptor.idVendor == VENDOR &&
         dev->descriptor.idProduct == PRODUCT &&
         alt->bInterfaceClass == 3 ) {
          process_device(argc, argv, dev, cfg, itfnum);
          ++found;
        }
      }
    }
      }
    }
  }

  if ( ! found ) printf("No controller found on USB busses.\n");
  return 0;
```

```
}
```