Pairing and connecting to the PS4 controller

Library usage notes

1. Copy the files ps4\_driver.c and ps4\_driver.h into the project you are working on.
2. Include ps4\_driver.h as part of the project you are working on.
3. Pthreads library is used. Thus, pthread library/linker must be added for compilation.
   1. If using the eclipse IDE, go to Project Properties > C/C++ Build > Setting > Libraries (under Cross GCC Linker). Use the green add arrow in the Libraries pane and type in pthread and add that.
   2. If compiling command-line add –lpthread as a gcc linker option.
4. Call gp\_init() at program initialization.
5. The PS4 controller event handler is set up as a pthread and so, no calls to any library function is required to obtain information. Instead check the fields in the gp\_event struct.

The following constants defined in **ps4\_driver.h** can be changed by the user.

|  |  |  |
| --- | --- | --- |
| Constant | Description | Default value |
| MAX\_CONNECTION\_ATTEMPTS | At the beginning of program execution, connection to the gamepad is attempted by opening the event file. This constant describes the number of times to try to attempt a file open before giving up. File open attempts happen every 500ms. | 10 |
| GAMEPAD\_EVENT\_FILENAME | This is the event file that is opened for connecting with the gamepad. It is typically going to be /dev/input/eventX | /dev/input/event2 |

The struct gp\_event holds the information about the last event that occurred.

|  |  |  |
| --- | --- | --- |
| **Field** | **Data type** | **Description** |
| type | int | Describes if the event is a key press or axis change |
| value | int | Describes the value associated with the specific type of event |
| count | uint64 | Incremented by one when a new value is written |

|  |  |
| --- | --- |
| **type** | **value** |
| BTN\_SQUARE | 1 |
| BTN\_CROSS | 1 |
| BTN\_CIRCLE | 1 |
| BTN\_TRIANGLE | 1 |
| BTN\_LB | 1 |
| BTN\_RB | 1 |
| BTN\_LT | 1 |
| BTN\_RT | 1 |
| BTN\_SHARE | 1 |
| BTN\_OPTIONS | 1 |
| BTN\_LSTICK | 1 |
| BTN\_RSTICK | 1 |
| BTN\_PS | 1 |
| BTN\_TOUCHPAD | 1 |
| LEFT\_ANALOG\_X | -128 to +127  -128 when left analog stick is at leftmost position  +127 when left analog stick is at rightmost position  Linearly interpolated in between |
| LEFT\_ANALOG\_Y | -128 to +127  -128 when left analog stick is at bottommost position  +127 when left analog stick is at topmost position  Linearly interpolated in between |
| RIGHT\_ANALOG\_X | -128 to +127  -128 when right analog stick is at leftmost position  +127 when right analog stick is at rightmost position  Linearly interpolated in between |
| RIGHT\_ANALOG\_Y | -128 to +127  -128 when right analog stick is at bottommost position  +127 when right analog stick is at topmost position  Linearly interpolated in between |
| LEFT\_BUMPER | 0 to 255  0 when left bumper unpressed  255 when left bumper pressed all the way down  Linearly interpolated in between |
| RIGHT\_BUMPER | 0 to 255  0 when right bumper unpressed  255 when right bumper pressed all the way down  Linearly interpolated in between |
| DIGITAL\_X | 1 or -1  1 when right digital pad button pressed  -1 when left digital pad button pressed |
| DIGITAL\_Y | 1 or -1  1 when top digital pad button pressed  -1 when bottom digital pad button pressed |

How the key mappings were obtained