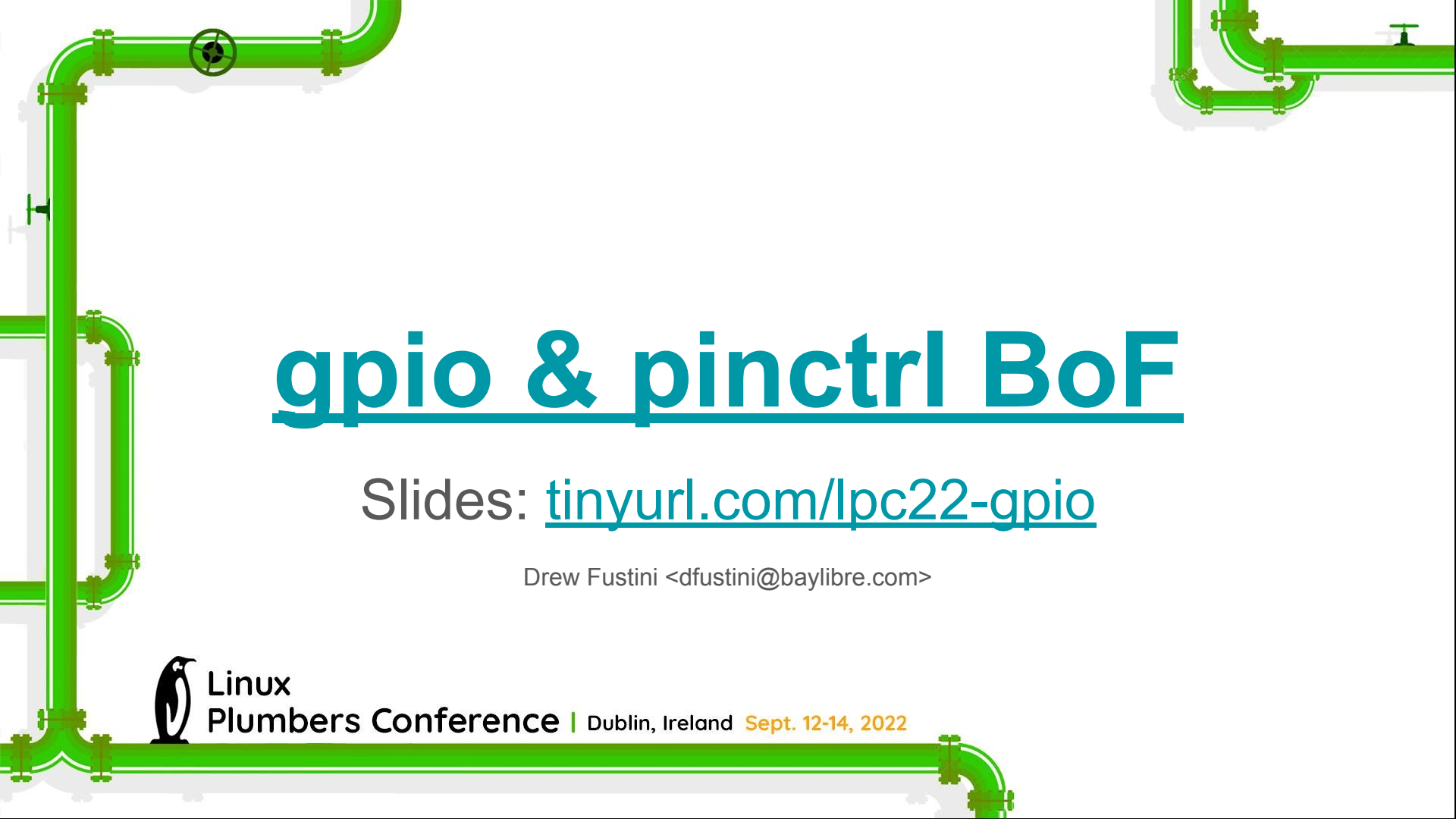


gpio & pinctrl BoF

Slides: tinyurl.com/lpc22-gpio

Drew Fustini <dfustini@baylibre.com>

Linux Plumbers Conference | Dublin, Ireland Sept. 12-14, 2022

Related presentations

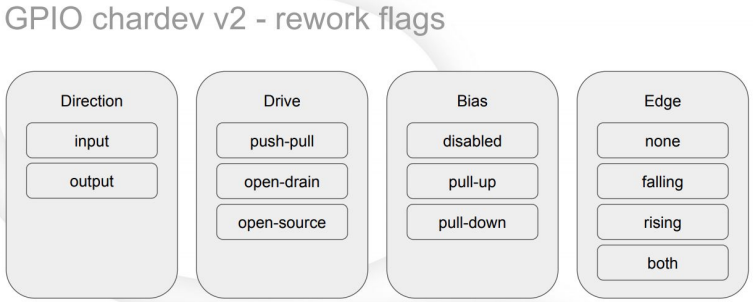
- gpio and pinctrl BoF at Linaro Connect Virtual 2020
  - Video / Slides
- Introduction to pin muxing and GPIO control *(Neil Armstrong)*
  - Video / Slides
- “Plan to Throw One Away” – Pitfalls of API Design for Low-level User-space Libraries and Kernel Interfaces *(Bartosz Golaszewski)*
  - Video / Slides
- libgpiod V2: New Major Release with a Ton of New Features *(Bartosz)*
  - Slides. This is part of “IoTs a 4-Letter Word” MC **this Wednesday morning**

Status of GPIO userspace API (uAPI)

- Writing drivers for devices using GPIOs is encouraged…
  - Subsystem drivers using GPIO
- …. but uAPI needed when no kernel device drivers provided/possible
  - Using GPIO Lines in Linux
- /sys/class/gpio is the legacy uAPI and is **deprecated**
- gpio character device *(aka gpiod)* is the current uAPI since Linux 4.8

GPIO character device uAPI v2

- gpio cdev uAPI v2 by Kent Gibson was merged in Linux 5.10
  - line handle and event requests are merged into a single request, the line request
  - allows for multiple lines producing edge events on the same line handle.
  - only two types of file handle to be concerned with, the chip and the line, and it is clearer which ioctls apply to which type of handle
  - Flag fields are collapsed to one filed: gpio_v2_line_flag



Export GPIO consumer's PID to userspace

- New patch series by Bartosz to allow userspace to know the PID of a process holding GPIO lines in case they have the permissions and need to kill them.

```
struct gpio_v2_line_info {
        __u32 num_attrs;
        __aligned_u64 flags;
        struct gpio_v2_line_attribute attrs[GPIO_V2_LINE_NUM_ATTRS_MAX];
+       __s32 consumer_pid;
        /* Space reserved for future use. */
-       __u32 padding[4];
+       __u32 padding[3];
 };
```

Export GPIO consumer's PID to userspace

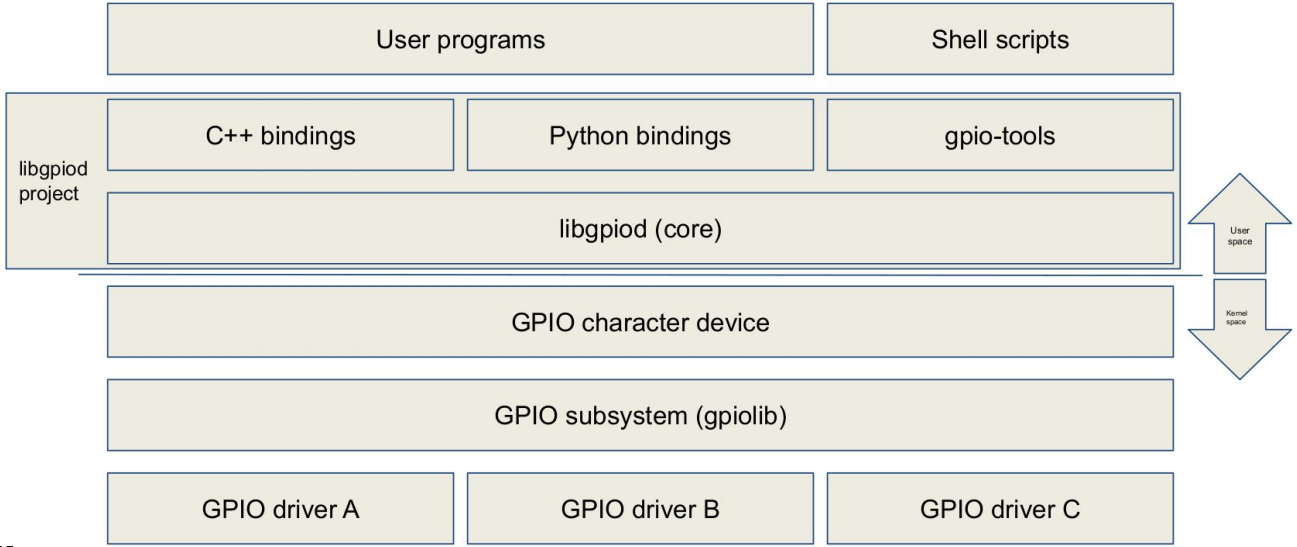
- Set consumer PID to the process ID for user-space consumers and 0 for kernel-space ones.
- Andy: Why not -1? [..] the usual way of telling "don't use this PID"
- Kent: make the pid unsigned, as we never pass a negative PID.
  - Keeping in mind that the existing kernel will return 0 for this field
  - 0 needs to be excluded from valid PIDs anyway.
  - If return -1, then 0 would mean "old kernel", while -1 would mean "kernel held".
  - libgpiod will have to convert 0 to -1 when returning the PID to user-space as pid_t
  - uAPI using 0 to mean "no PID available" for all cases.

Export GPIO consumer's PID to userspace

- Any security considerations?
- By default any user - even one who doesn't have permissions to access /dev/gpiochip* - can already figure out by browsing /proc/$PID/fd that a process does have some lines requested - but not which exactly.
- This provides that additional bit of knowledge to users who already do have permissions to call ioctl() on /dev/gpiochip*

libgpiod: library and tools for GPIO cdev

- Created and maintained by Bartosz

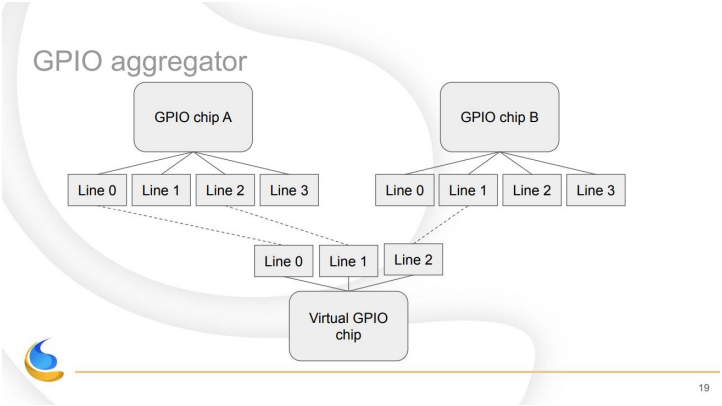


libgpiod: status of v2 API?

- Development happens on the linux-gpio mailing list
- Branch next/libgpiod-2.0 contains current v2 API
- Rust bindings being developed by Viresh Kumar
- [libgpiod v2][RFC PATCH] treewide: rework line configuration
  - We're adding a new structure - line_settings. It's a basic data class that stores a set of line properties. The line_config object is simplified and becomes a storage for the mappings between offsets and line_settings.
- Slides by Bartosz for "IoTs a 4-Letter Word" MC **this Wednesday morning at 11:20 Dublin time**
  - libgpiod V2: New Major Release with a Ton of New Features

GPIO Aggregator

- Provides a mechanism to aggregate GPIOs, and expose them as a new gpio_chip
- **Anyone have problems or ideas to discuss?**



pinctrl topic: pinmux-select

- Drew Fustini added pinmux-select added to pinctrl debugfs files in Linux 5.13
- Motivation: allow userspace to change pinmux for BeagleBone headers pin when prototyping (e.g. breadboarding)

```
Add "pinmux-select" to debugfs which will activate a pin function for a given pin group:

  echo "<group-name function-name>" > pinmux-select

The write operation pinmux_select() handles this by checking that the names map to valid selectors and then calling
ops->set_mux().

The existing "pinmux-functions" debugfs file lists the pin functions registered for the pin controller. For example:

  function: pinmux-uart0, groups = [ pinmux-uart0-pins ]
  function: pinmux-mmc0, groups = [ pinmux-mmc0-pins ]
  function: pinmux-mmc1, groups = [ pinmux-mmc1-pins ]
  function: pinmux-i2c0, groups = [ pinmux-i2c0-pins ]
  function: pinmux-i2c1, groups = [ pinmux-i2c1-pins ]
  function: pinmux-spi1, groups = [ pinmux-spi1-pins ]

To activate function pinmux-i2c1 on group pinmux-i2c1-pins:

  echo "pinmux-i2c1-pins pinmux-i2c1" > pinmux-select
```