Revisiting device_probe Implementations for USB Drivers

This article is supplementary to the book *FreeBSD Device Drivers*. It assumes you've read up to and including Chapter 15: USB Drivers.

Looking Back

Recall from Chapter 15 that the function `ulpt_probe()`, which is the `device_probe` implementation for `ulpt(4)`, is defined as follows:

```
static int
ulpt_probe(device_t dev)
{
        ❶struct usb_attach_arg *uaa = device_get_ivars(dev);

        ❷if (uaa->usb_mode != USB_MODE_HOST)
                return (ENXIO);

        ❸if ((uaa->info.bInterfaceClass == UICLASS_PRINTER) &&
            (uaa->info.bInterfaceSubClass == UISUBCLASS_PRINTER) &&
            ((uaa->info.bInterfaceProtocol == UIPROTO_PRINTER_UNI) ||
             (uaa->info.bInterfaceProtocol == UIPROTO_PRINTER_BI) ||
             (uaa->info.bInterfaceProtocol == UIPROTO_PRINTER_1284)))
                return (BUS_PROBE_SPECIFIC);

        return (ENXIO);
}
```

Listing 1: ulpt_probe() in FreeBSD 8.0

This function's structure is typical for a USB driver's `device_probe` implementation in FreeBSD 8.0. The following paragraph, which is taken from *FreeBSD Device Drivers*, describes this function.

`ulpt_probe()` begins by ❷ ensuring that the USB host controller is in host mode, which is needed to initiate data transfers. Then `ulpt_probe()` ❸ determines whether `dev` is a USB printer. Note that ❶ `struct usb_attach_arg` contains `dev`'s instance variables.

Moving Forward

Starting with FreeBSD 8.0, `device_probe` implementations for USB drivers can be structured differently than what's shown in Listing 1. In FreeBSD 8.3, `ulpt_probe()` looks like this:

```
static const ❶STRUCT_USB_HOST_ID ulpt_devs[] = {
        /* Unidirectional USB printer.  */
        { USB_IFACE_CLASS(UICLASS_PRINTER),
          USB_IFACE_SUBCLASS(UISUBCLASS_PRINTER),
          USB_IFACE_PROTOCOL(UIPROTO_PRINTER_UNI) },
```

```
        /* Bidirectional USB printer.   */
        { USB_IFACE_CLASS(UICLASS_PRINTER),
          USB_IFACE_SUBCLASS(UISUBCLASS_PRINTER),
          USB_IFACE_PROTOCOL(UIPROTO_PRINTER_BI) },

        /* 1284 USB printer.            */
        { USB_IFACE_CLASS(UICLASS_PRINTER),
          USB_IFACE_SUBCLASS(UISUBCLASS_PRINTER),
          USB_IFACE_PROTOCOL(UIPROTO_PRINTER_1284) },
};

static int
ulpt_probe(device_t dev)
{
        struct usb_attach_arg *uaa = device_get_ivars(dev);
        int error;

        if (uaa->usb_mode != USB_MODE_HOST)
                return (ENXIO);

        error = ❷usbd_lookup_id_by_uaa(❸ulpt_devs, sizeof(ulpt_devs), ❹uaa);
        if (error)
                return (error);

        return (BUS_PROBE_GENERIC);
}
```

Listing 2: ulpt_probe() in FreeBSD 8.3

The difference between this version of `ulpt_probe()` and the one shown in Listing 1 is that here ❷ `usbd_lookup_id_by_uaa()` is used to determine whether `dev` is a USB printer instead of doing it "by hand."

The function `usbd_lookup_id_by_uaa()` takes an ❸ array of `usb_device_id` structures, which is defined using the ❶ `STRUCT_USB_HOST_ID` macro, and returns `0` if any element matches the data in a ❹ `usb_attach_arg` structure (which should contain `dev`'s instance variables).

The benefit of using `usbd_lookup_id_by_uaa()` instead of doing it by hand is that it automatically exports the device's ID to */usr/src/tools/tools/bus_autoconf/* and */etc/devd/usb.conf*.