Device-Tree / ACPI compatibility

David Woodhouse <David.Woodhouse@intel.com>
Kernel Recipes 2015
Origins of “Device Tree” properties

• IEEE1275 Open Firmware
• Power PC Reference Platform (PreP)
• Common Hardware Reference Platform (CHRP)
• Embedded Power Architectures Platform Requirements (ePAPR)
Standard device properties

• “compatible”
• “interrupts”
• “reg”
• “device_type”
Device-specific properties

• “clock-frequency”
• “current-speed”
• “reg-shift”
• “fifo-size”
Linux and device-tree

• Query OpenFirmware at run time (SPARC)
• Query OpenFirmware at boot time (PowerPC)
• No OpenFirmware at all (ARM) → “Flattened Device Tree”
uart@3,1 {
    compatible = "ns16550a", "ns16450";
    reg = <3 0x100 8>; /* CS3, offset 0x100, IO size 8 */
    bank-width = <2>;
    reg-shift = <1>;
    reg-io-width = <1>;
    interrupt-parent = <&gpio4>;
    interrupts = <6 IRQ_TYPE_EDGE_RISING>; /* gpio102 */
    clock-frequency = <1843200>;
    current-speed = <115200>;
};
ACPI device description

• Hardware ID (_HID): “ABCD0001”
• Compatibility ID (_CID)
  – No standard way to convey more detail
ACPI 5.1: _DSD

Name (_HID, “ABCD0001”)
Name (_DSD, Package () {
    ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
    Package () {
        Package {"a-string-property", "A string"},
        Package {"a-cell-property", Package {1, 2, 3, 4}};
    }
})
Linux: Generic property APIs

• `of_property_…() → device_property_…()`

```c
/* Check for registers offset within the devices address range */
- if (of_property_read_u32(np, "reg-shift", &prop) == 0)
+ if (device_property_read_u32(&pdev->dev, "reg-shift", &prop) == 0)
   port->regshift = prop;

/* Check for fifo size */
- if (of_property_read_u32(np, "fifo-size", &prop) == 0)
+ if (device_property_read_u32(&pdev->dev, "fifo-size", &prop) == 0)
   port->fifosize = prop;
```
Matching Device-Tree devices

```c
static const struct of_device_id of_match_table[] = {
    { .compatible = "foo" },
    {};
MODULE_DEVICE_TABLE(of, of_match_table);
```
Matching ACPI devices

static const struct acpi_device_id acpi_match_table[] = {
    { .id = "ABCD0001" },
    {};
MODULE_DEVICE_TABLE(acpi, acpi_match_table);
But we don't want to have to do that!

Name (_HID, “ABCD0001”)
Name (_CID, “PRP0001”)
Name (_DSD, Package () {
    ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
    Package () {
        Package{"compatible", "foo"},
        Package{"a-string-property", "A string"},
        Package{"a-cell-property", Package {1, 2, 3, 4}};
    }
})
Representing GPIO properties in Device Tree

```c
gpio0: gpio-controller@f1000000 {
    compatible = "foo-gpio-controller";
    #gpio-cells = <2>;
};

foo {
    compatible = "foo";
    reset-gpio = <&gpio0 30 GPIO_ACTIVE_LOW>
};
```
Representing GPIO properties in _DSD

Device (FOO)
{
    Name (_HID, "PRP0001")
    Name (_CRS, ResourceTemplate ()
    {
        GpioIo (Exclusive, PullUp, 0, 0, IoRestrictionInputOnly, "\\_SB.GPO0", 0, ResourceConsumer) {15}
    }
    Name (_DSD, Package ()
    {
        ToUUID("daffd814-6eba-4d8c-8a91-bc9bbf4aa301"),
        Package ()
        {
            Package (){"reset-gpio", Package() {^FOO, 0, 0, 0 }},
            /* ref, idx, pin, active-low */
        }
    })
}
Driver GPIO code

```c
struct gpio_desc *gpio = gpiod_get(dev, "reset-gpio", 0);
```
ACPI 5.1 _DSD properties + “PRP0001” ID

• Re-use of existing device bindings
• No need to modify drivers (after migration from old OF-specific API)
• Seamless transition between ACPI and DT firmwares
• Preserves native representation where available
Q & A