



designwest
center of the engineering universe

Device Trees – A Database Approach
to Describing Hardware
ESC 331
Doug Abbott

Intellimetrix
Accelerating Product Development and Innovation

April 22-25, 2013
McEnery Convention Center
San Jose, CA
www.ubmdesign.com



Problem



- How to describe hardware to OS?
 - Build description into drivers
 - CONFIG_ variables
 - Create a “Board Support Package”
 - mach-* and plat-* directories in arch/arm/
 - Database approach
- OS and bootloader may have different views of hardware

“Gaah! Guys, this whole ARM thing is a _____ pain in the ass.”

Linus Torvalds

Platform vs. “discoverable” devices



- Discoverable devices
 - Connected to well-defined bus like PCI or USB
 - Can be interrogated to determine what they are
- Platform devices
 - Generally connect directly to the CPU
 - Lack ability to be interrogated
 - These are the tough ones!

Device Tree Background



- Came from Open Firmware standard
 - IEEE 1275
 - Replaced by UEFI (Unified Extensible Firmware Interface)
- Widely used in Power PC world
 - Now required for all PPC Linux implementations
- Gaining traction in ARM world
 - See linux/arch/arm/boot/dts

Device Tree



- Data Structure for Describing Hardware Configuration
 - Standard interface between bootloader and OS
 - Avoid hard coding platform details
- Open Firmware specification
- Framework that can support multiple SoC devices with single kernel image

Device Tree Syntax



```
/ memreserve / 0 x20000000-0 x21FFFFFF ;  
/{  
    model = " MyBoard ";  
    compatible = " acme, MyBoard";  
    #address - cells = <1>;  
    #size - cells = <1>;  
    interrupt-parent = <&intc>;  
  
    cpus {  
        #address - cells = <1>;  
        #size - cells = <0>;  
        cpu@0 {  
  
        };  
    };  
    intc: Interrupt_controller @10140000 {  
  
    };  
};
```

Device Tree Elements



- Tree -- hierarchy of nodes beginning at /
- Node -- name { properties };
- Name
 - Up to 31 characters
 - Represents *type* of device, not specific implementation
 - Format: <string>[@<unit_address>]
 - <unit_address>: how the node is referenced

Node Properties



- name = value;
- Value:
 - String -- "MyBoard"
 - Cells -- <0 8000 f0000000>
 - Bytestring -- [1234abcdef]
 - Reference -- &/path_to_node
 - Empty

Memory-mapped Device Node



```
serial@101f0000 {  
    compatible = "arm, pl011";  
    reg = <0x101f0000 0x1000 >;  
    interrupts = <3 0>;  
}
```

Bus-mapped Device Node



```
external-bus {  
    #address-cells = <2>;  
    #size-cells = <1>;  
  
    i2c@1,0 {  
        compatible = "acme, a1234-i2c-bus";  
        #address-cells = <1>;  
        #size-cells = <0>;  
        reg = <1 0 0x1000>;  
        rtc@58 {  
            compatible = "maxim, ds1338";  
            reg = <58>;  
        };  
    };  
};
```

Interrupt controller Device Node



```
intc: interrupt-controller@10140000 {  
    compatible = "arm, pl190";  
    reg = <0x10140000 0x1000 >;  
    interrupt-controller;  
    #interrupt-cells = <2>;  
};
```

"aliases" node



- Associate short name with full path to node
- Must be at root level

```
aliases {  
    i2c = "/external_bus/i2c@1,0";  
    rtc = "/external_bus/i2c@1,0/rtc@58";  
};
```

“chosen” node



- Doesn't represent a device
- Used to pass variable data from bootloader to kernel
 - Often left blank in dts. Filled in at boot time
- Must be at root level

```
chosen {  
    bootargs = "root=/dev/nfs rw nfsroot=192.168.1.50  
    console=ttyS0, 115200";  
};
```

compatible Property



- Link between device and driver
- In device tree
 - compatible = "device_name";
- In driver code
 - struct of_device_id my_of_match[] = {
 - { .compatible = "device_name"
 - }
 - MODULE_DEVICE_TABLE (of,
my_of_match)

Property Names



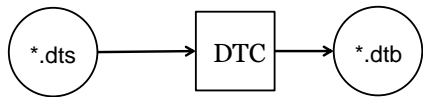
- Defined by “bindings”
- Bindings defined by:
 - Original IEEE 1275 specification
 - Not available free
 - ePAPR
 - See References slide at end
 - Kernel source tree
 - linux/Documentation/devicetree/bindings

Device Tree and Linux



- Configuration: Boot Options
 - Device Tree Support
 - CONFIG_USE_OF
- /proc/device-tree
 - Node -> directory
 - Property -> file
- On entry to kernel (ARM)
 - R2 -> device tree blob
- Device tree support code in drivers/of

Flattened Device Tree



Device Tree Source Device Tree Compiler Flattened Device Tree Blob

DTC at linux/scripts/dtc
DTS files in arch/<arch>/boot/dts

make <board_name>.dtb

FDT and u-boot



- #define CONFIG_OF_LIBFDT
- fdt command
 - List tree
 - Make new nodes
 - Set properties
 - Remove nodes or properties
- bootm <kernel_addr> <initrd_addr> <dtb_addr>

Accessing the dtb



```
#include of.h

struct device_node – Represents a node in the device tree. Linked list
struct property – Represents a property of a device node. Linked list

struct device_node *of_find_node_by_name (struct device_node *from,
const char *name);
struct device_node *of_find_node_by_type (struct device_node *from,
const char *type);
struct device_node *of_get_parent (const struct device_node *node);

struct property *of_find_property (const struct device_node *np,
const char *name, int *lenp);
int of_n_addr_cells (struct device_node *np);
int of_n_size_cells (struct device_node *np);
```

Device Tree Does Not...



- Replace board-specific code
 - Handles the common case
 - Odd hardware still requires special treatment
- Add features
- Boot fast
 - Increases kernel footprint

Example



- `linux/drivers/tty/serial/of_serial.c`
- `linux/arch/arm/boot/dts/kirkwood.dtsi`
- `linux/arch/arm/boot/dts/kirkwood-dns320.dts`

Summary



- Problem: how to describe system hardware to the OS?
 - Make sure bootloader and OS have consistent view
- Device tree = database for describing hardware
 - Device Tree Compiler (DTC) translates source to binary “blob”

Summary II



- Bootloader passes blob to OS at boot time
- Kernel and device drivers interrogate blob to determine configuration
 - Kernel and drivers can now be “generic” rather than unique to each board

References



- Standard for Embedded Power Architecture Platform Requirements (ePAPR)
 - www.power.org/resources/downloads/Power_ePAPR_APPROVED_v1.0.pdf
- Device Tree Usage
 - www.devicetree.org/Device_Tree_Usage
- Kernel source tree
 - `linux/Documentation/devicetree`
 - `usage-model.txt`