ARM Flattened Device Tree status report

Grant Likely
Secret Lab Technologies Ltd.

April 14, 2010
Embedded Linux Conference
San Francisco
Overview

- Device Tree Overview
- Integration with the Linux device model
- rework
- ARM specific
Terminology

- **OpenFirmware (OF) Device Tree (DT)**
  - Device representation exported by Open Firmware
  - This presentation is **not** about Open Firmware
- **Flattened Device Tree (FDT)**
  - Firmware-independent device tree
- **Device Tree Compiler (DTC)**
  - Convert between .dts and .dtb
  - Device Tree Source (.dts)
  - Device Tree Blob (.dtb)
    - Tokenized form; used by kernel
- **Binding**
  - Documentation of how the DT describes hardware
- **Too Many Acronyms (TMA)**
FDT – 3 minute overview

- Data structure for describing hardware
- Passed to kernel at boot
- Alternative to hard-coded platform details
Device Tree Model

- Tree with nodes & properties
  - Nodes give structure
  - Properties add detail
    - Key-value pairs
- 'compatible' property
  - Each 'compatible' value associated with a 'binding'.
- 'phandles'
  - Secondary connections between nodes
  - irqs, gpios, mdio, i2s, etc
FDT – Why?

- Multiplatform
- Variances
- Simplify board ports
- Firmware → Kernel data passing
FDT – Why not?

- Complexity (Does it buy me anything?)
- Learning curve
- More work?
It took a lot of work, but this latest Linux patch enables support for machines with 4,096 CPUs, up from the old limit of 1,024.

Do you have support for smooth full-screen Flash video yet?

No, but who uses that?
FDT – What it is not?

- Doesn't replace board-specific code
  - Good for common-case
  - Special cases are still special
- Doesn't add features to your platform
FDT – Alternatives

- BIOS/ACPI/DSDT
- UEFI
- Open Firmware

- Why isn't u-boot/CFE/favorite firmware in this list?
Model – Firmware

- Firmware obtains .dtb and passes to kernel
- How? Options:
  - Load and pass verbatim
  - Load and modify
  - Generate from scratch
    - (Here there be Dragons!)
Model - Booting

- Device Tree passed in ATAG_DEVTREE
  - Physical address of .dtb (tentative)
  - May carry devtree as ATAG payload

- Early init
  - Determine memory
  - Determine machine
  - Boot to mm set up
  - Issue: access during early init?
Model – Booting

- Unflatten .dtb
  - Allocate space for unpacked form
  - Can now directly dereference tree
  - Same representation as for 'real' Open Firmware
- Boot to machine_init
- Register devices
Digression: Linux device model

- Busses
- Devices
- Drivers
- Hierarchy of Devices
Digression: Linux device model

Drivers registered against bus types

- platform
  - spi
  - bridge
    - uart1
  - i2c bus
    - eeprom
    - temp
    - rtc
  - pci bus
    - ethernet

- platform_bus_type
- i2c_bus_type
- pci_bus_type
DT & Linux device model – historical

● SPARC
  – Walk entire tree, register of_device for each node
  – drivers may bind against any node
  – Other bus registrations mirror of_device hierarchy (ie. PCI)
    • Duplicate 'struct device'
DT & Linux device model - historical

- PowerPC/Microblaze
  - Subset of tree registered as of_devices
    - Typically only mmio devices
  - Drivers may create child busses of different types
  - No device-tree integration with non-of_devices
    - OF wrappers for other busses
DT & Linux device model – new approach

• Conceptual flaws of of_platform_bus
  – Duplicate of platform bus
  – DT data applicable to more than just platform devices
  – platform_driver won't bind against an of_device
DT & Linux device model – new approach

- DT is *support data*
  - Make available to all devices
- Move probe data
  - *of_node → struct device*
  - *of_match_table → struct device driver*
- Generalize OF-style binding functions
  - Available to any bus type
- Eliminate of_platform_bus_type
  - Merge with platform_bus_type
Demo Time
Current State

- Infrastructure works
  - Register mapping
  - IRQ mapping
  - Platform bus integration
- Board support (minimal)
  - Versatile on QEMU
  - OMAP3
  - i.MX51
- Registering platform devices
- Binding against drivers
Next Steps

- Finish board support
- I2c, spi, and others
  - Need to add bus binding code
- Documentation
  - How to use it
  - Bindings
Resources

- Secret Lab git tree
  - git://git.secretlab.ca/git/linux-2.6
  - Branch: test-devicetree
  - Unified code, bindings
- Jeremy Kerr's tree
  - git://git.can
  - Early init support, boot wrapper
- Web sites
  - http://fdt.secretlab.ca
  - https://wiki.ubuntu.com/KernelTeam/ARMDeviceTrees
Acknowledgements

- CE Linux Forum
- IBM/Freescale/Canonical
- Too many people to mention