It’s Broken!

Fixing the DT binding process

It’s Broken!

Fixing the DT binding process

The Problem

- Not everything is discoverable.
  - Isn’t just an ARM or embedded problem
    - servers and desktops
  - Isn’t just a platform_device problem
    - PCI and USB
- Topology between devices is hard
- Becoming *more* prevalent, not less
- Takes time to see *how* to describe hardware
  - What are the common patterns?

The Problem - part 2

- DT/ACPI describes what is non-discoverable
- Binding defines how to describe specific devices
- Stability Tension:
  - Don’t yet know how hardware should be described
  - Don’t break users
- Infrastructure Tension:
  - Common patterns
  - Device specific schema
  - Conversion from board centric view
  - Strict adherence to historic DT principles

Proposed Solution (Policy)

- By default, treat bindings as stable
  - **BUT,** don’t get trapped creating the “perfect” binding
- Bindings can change - when done correctly
  - We can refactor as patterns emerge
- Configuration data are acceptable
- We will provide:
  - Facility for unstable bindings
  - Documentation on best practices and process
  - Tools for validation

Statement on DT process

- **Bindings MUST be documented**
  - A binding is a schema for hardware description
  - `Documentation/devicetree/bindings`
  - Ideally in a separate patch
  - cc'd to devicetree@vger.kernel.org
  - In near future will be enforced via tooling
- **Binding Acks:**
  - MUST be acked from subsystem maintainer
  - SHOULD be acked from DT maintainer
    - Subsystem maintainer MAY make decision in case of sleepy DT maintainers.
  - Generic subsystem bindings require higher scrutiny
- **Merge bindings via subsystem trees**
  - Merge .dts changes via ARCH tree

Questions?

Configuration data

- Describes intended operation point of device
- Entirely reasonable to encode in device tree
  - Just try not to encode Linux implementation details
- Guidelines will be published

Making a compatible update

- Old kernels MUST continue to function with updated DT
  - Properties DEPRECATED, not removed
  - New optional properties may be added
- Guidelines (review checklist) on how to design future-proof bindings will be published

Making an incompatible update

- Submit to public flogging
- New compatible string (v2..)
- Driver support both old and new bindings
  - Old binding may be removed after sufficient time
- DTS should have ***only*** old or new instance

Unstable bindings

- Analogous to CONFIG_STAGING
  - In tree, but not stable
    - Not guaranteed to work in future
  - Must be explicitly enabled (UNSTABLE_DT?)
  - Either stabilized or removed
- Not a shortcut around review / correctness
  - Exceptional cases only (e.g. new device class)
    - MUST meet usual hygiene standards
  - Taint flag if used (?)

General Binding Review Rules

- Check that code matches schema
- Check the schema matches the hardware
  - Configuration options
  - Missing or ambiguous reg, interrupts, clocks, resets, regulators, power domains…
- Err on the side of simplicity
  - You can always add stuff later
  - Be explicit from start — which interrupt is which?

schema design goals

- bindings without schema should warn, not error (unless strict is requested)
- Schema encodes both documentation and binding
- Flag properties/nodes that aren’t in schema

Other Notes

- Representation (DT or ACPI) is irrelevant - schema policy is the same
- “Failover cascade” is important
  - Aim for ABI stability
  - Extend with optional properties
  - Change compatible when old binding is broken
  - Use quirks when really really broken