DOs and DON'Ts for designing and writing Devicetree bindings

This is a list of common review feedback items focused on binding design. With every rule, there are exceptions and bindings have many gray areas.

For guidelines related to patches, see Documentation/devicetree/bindings/submitting-patches.rst

Overall design

- DO attempt to make bindings complete even if a driver doesn't support some features. For example, if a device has an interrupt, then include the 'interrupts' property even if the driver is only polled mode.
- DON'T refer to Linux or "device driver" in bindings. Bindings should be based on what the hardware has, not what an OS and driver currently support.
- DO use node names matching the class of the device. Many standard names are defined in the DT Spec. If there isn't one, consider adding it.
- DO check that the example matches the documentation especially after making review changes.
- DON'T create nodes just for the sake of instantiating drivers. Multi-function devices only need child nodes when the child nodes have their own DT resources. A single node can be multiple providers (e.g. clocks and resets).
- DON'T use 'syscon' alone without a specific compatible string. A 'syscon' hardware block should have a compatible string unique enough to infer the register layout of the entire block (at a minimum).

Properties

- DO make 'compatible' properties specific. DON'T use wildcards in compatible strings. DO use fallback compatibles when devices are the same as or a subset of prior implementations. DO add new compatibles in case there are new features or bugs.
- DO use a vendor prefix on device-specific property names. Consider if properties could be common among devices of the same class. Check other existing bindings for similar devices.
- DON'T redefine common properties. Just reference the definition and define constraints specific to the device.
- DO use common property unit suffixes for properties with scientific units. Recommended suffixes are listed at https://github.com/devicetree-org/dt-schema/blob/master/schemas/property-units.yaml
- DO define properties in terms of constraints. How many entries? What are possible values? What is the order?

Board/SoC .dts Files

- DO put all MMIO devices under a bus node and not at the top-level.
- DO use non-empty 'ranges' to limit the size of child buses/devices. 64-bit platforms don't need all devices to have 64-bit address and size.