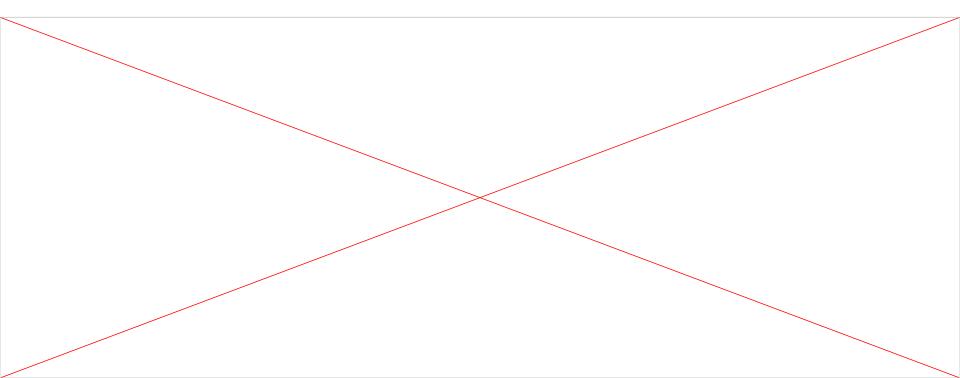


Common clock Device Tree bindings

Mike Turquette, 2013-10-29



Quick note on pull requests

- Number of clock driver changes has increased
- Some of the more active platforms are having lots of merge conflicts amongst their own clock driver changes
- I'm happy to take pull requests, instead of cobbling it all together
 - After requisite patch review
- Also might be good to have one owner/maintainer for each platform's clock driver that marshalls all patches together



Device Tree as clkdev

 The primary benefit that DT brings to clocks is linking clock phandles to consumer devices

- This has been integrated with clk_get
- If DT does absolutely nothing else for clocks, that's OK by me



Terminology

A "clock controller" node in DT has various clock inputs and various clock outputs. #clock-cells is 1. Use a phandle + index to get a clock.

- Documentation/devicetree/bindings/clock/exynos5250-clock.txt
- arch/arm/boot/dts/exynos5250.dtsi

A "per-clock" node corresponds to a single clock entity, often with verbose clock data. #clock-cells is 0. Use just the phandle to get it.

http://www.spinics.net/lists/arm-kernel/msg282479.html



DT clock bindings

Two approaches - each at opposite ends of the spectrum

- Model a clock controller as a node in DTS
 - Least controversial
 - Little to no hardware description, no per-clock data in DTS
 - Clocks are enumerated in the binding definition
 - Only purpose is to link clock phandles to clock consumer nodes
 - Sometimes referred to as the "clock mapping" approach
 - Results in lots of struct clk object



DT clock bindings, cont'd

- Model per-clock data in DTS
 - Most controversial
 - There should be no clock data in the kernel source at all
 - Benefit is moving data out of the kernel makes Linus happy?
 - Only used in some rare cases fixed-rate clock, OMAP
 - Results in a single struct clk object



Why model clock data in DTS?

- Off-SoC clocks such as oscillators
 - Things that change on a per-board basis
- Intermediate solution for new platforms
 - If bootloader enables clocks then it is possible to use fixed-rate clocks defined in DTS as placeholder clocks
 - This is helpful for drivers that expect to get clocks and know their rate
 - Analogous to a dummy regulator
 - Eventually replace this data in DTS with the appropriate clock controller node binding



Why model clock data in DTS?

- The only reason is to move 100% of clock data out of the kernel source
 - o Flamebait: Is that what Linus wants in The Future?

- If we cannot remove 100% of the static clock data from the kernel then do not bother putting any of it in DTS
 - Just use a clock controller mapping scheme



Modeling clock data in DTS

- Easy enough for off-chip fixed-rate crystals and oscillators
- What about complex on-SoC clocks?
 - We need to model register-level data such as bitmasks
 - In some cases, create one node per clock



Misc. note on clock-names

- Please use the clock-names property to list the input name for the clock with respect to the consuming node
- Not the name of the clock output name from the data sheet!





More about Linaro: http://www.linaro.org/about/

More about Linaro engineering: http://www.linaro.org/engineering/

How to join: http://www.linaro.org/about/how-to-join

Linaro members: www.linaro.org/members