



Linux
Plumbers
Conference | Richmond, VA | Nov. 13-15, 2023

Make sync_state/handoff work for the common clk framework





Background

clk_disable_unused

Agenda

- Background
 - `clk_disable_unused`
 - Handoff
 - Use Cases
- Proposed Solutions
- Brainstorming

`clk_disable_unused()`

for all clks

if struct clk_ops::is_enabled() && !struct clk_core::enable_count

struct clk_ops::disable()

`late_initcall_sync(clk_disable_unused)`



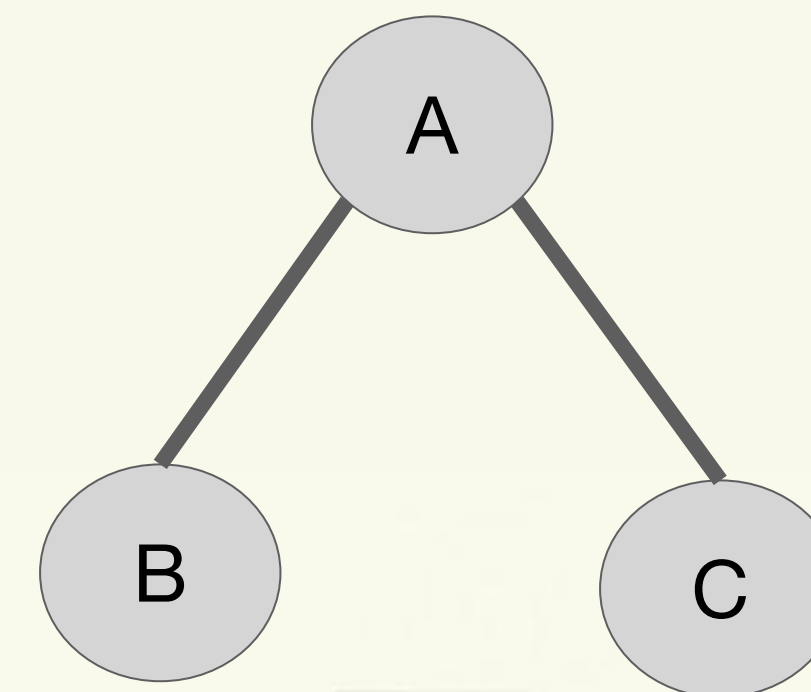
Background

Handoff

What clks are enabled and/or prepared when they are registered with `clk_register()`?

Agenda

- **Background**
 - `clk_disable_unused`
 - **Handoff**
 - Use Cases
- Proposed Solutions
- Brainstorming





Background

Use Cases

Agenda

- **Background**
 - clk_disable_unused
 - Handoff
 - **Use Cases**
- Proposed Solutions
- Brainstorming

- Boot splash screen maintained until display driver probes
- Save power by disabling clks that bootloader leaves enabled
- Save power by disabling clks for devices without a driver





Linux
Plumbers
Conference | Richmond, VA | Nov. 13-15, 2023

Proposed Solutions





Proposed Solutions

Add `sync_state()` support to clock framework [1]

Agenda

- Background
- **Proposed Solutions**
 - `sync_state()`
 - generic callback
 - CLK_HANDOFF
- Brainstorming

If enabled at `clk_register()` hold that enable until `clk_sync_state()`

Was the clk registered with a struct device? If so, skip clk during disabling of unused clks.

When `sync_state` logic triggers for a device, call `clk_sync_state()` which iterates over the entire clk tree for any clks registered with that device and call struct `clk_ops::disable()` if enabled and unused



Proposed Solutions

Add `sync_state()` support to clock framework

Agenda

- Background
- **Proposed Solutions**
 - `sync_state()`
 - generic callback
 - `CLK_HANDOFF`
- Brainstorming

Rejection Reasons

- Enables clks during registration when they're already enabled
- Keeps clks enabled until `sync_state` stage (potentially long time)
- Relies on clks to be registered with struct device for `sync_state()` callback
- Increases software `enable_count`, leading to possible underflow issues of count by other consumers



Proposed Solutions

clk: Add generic sync_state callback for disabling unused clocks [2]

Mostly same as before, with some differences

Agenda

- Background
- **Proposed Solutions**
 - sync_state()
 - **generic callback**
 - CLK_HANDOFF
- Brainstorming

- Don't hold enable state from registration time
- Allow sync_state callback to be anything in case drivers want to override



Proposed Solutions

clk: Add generic sync_state callback for disabling unused clocks

Agenda

- Background
- **Proposed Solutions**
 - sync_state()
 - generic callback
 - CLK_HANDOFF
- Brainstorming

Rejection Reasons

- Avoids enabling clks at registration time
- Doesn't keep clks enabled from boot (doesn't solve hand off)
- Clks can be disabled in the middle of the tree affecting leafs with sync_state
- Relies on clks to be registered with struct device for sync_state() callback



Proposed Solutions

CLK_ENABLE_HAND_OFF clk flag [3]

Add struct clk_core booleans needs_prepare_handoff and needs_enable_handoff

Add enable/prepare counts to struct clk

Agenda

- Background
- **Proposed Solutions**
 - sync_state()
 - generic callback
 - **CLK_HANDOFF**
- Brainstorming

clk_enable()

struct clk::enable_count++

if struct clk_core::needs_enable_handoff

clear bool and return

clk_register()

if CLK_ENABLE_HAND_OFF

set bool to true, call clk_core_enable()



Proposed Solutions

CLK_ENABLE_HAND_OFF clk flag

Agenda

- Background
- **Proposed Solutions**
 - sync_state()
 - generic callback
 - **CLK_HANDOFF**
- Brainstorming

Rejection Reasons

- Requires marking clks with clk flag to opt-in
- Doesn't check enable state to know if clk would like to opt in to flag



Linux
Plumbers
Conference | Richmond, VA | Nov. 13-15, 2023

Brainstorming





Brainstorming

What Do Other Frameworks Do?

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - **Other Frameworks**
 - Kconfig
 - Hand off
 - Read Hardware

Regulator

- Read hardware for enable state
- DT property for boot enabled
- Wait 30 seconds after late init and disable unused regulators
- Not usually a complex tree
 - 10s not 100s of regulators
 - Not always in a tree



Brainstorming

What Do Other Frameworks Do?

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - **Other Frameworks**
 - Kconfig
 - Hand off
 - Read Hardware

Interconnect

- Scan DT and count number of interconnect providers during device_initcall
- Use sync_state() callback for interconnect providers
 - Framework mandates struct device during registration
- Iterate over all interconnects and drop bandwidth for unused ones once all interconnect providers call sync_state API



Brainstorming

Kconfig for `clk_ignore_unused=true`

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - Other Frameworks
 - **Kconfig**
 - Hand off
 - Read Hardware
- Make a config option to set `clk_ignore_unused` to true
- Don't ever disable clks from the clk framework because we don't know when to do so
- Invert logic so that commandline is needed to opt-in to ignore unused behavior when config enabled



Brainstorming

Hand off enable state

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - Other Frameworks
 - Kconfig
 - **Hand off**
 - Read Hardware

- During registration, read enabled state, mark 'boot_enabled' flag if enabled
- During clk reparenting, migrate flag to parent if child is boot enabled
- `clk_enable()` checks flag and only increments count if boot_enabled
- `clk_disable()` checks flag and clears when enable count reaches zero





Brainstorming

Hand off enable state - Problems

Agenda

- Background
 - Proposed Solutions
 - **Brainstorming**
 - Other Frameworks
 - Kconfig
 - **Hand off**
 - Read Hardware
- When is it safe to disable clk in middle of tree?
 - Need to special case CLK_OPS_PARENT_ENABLE



Brainstorming

Stop Caching Hardware State

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - Other Frameworks
 - Kconfig
 - Hand off
 - **Read Hardware**

```
clk_enable()
```

```
if !struct clk_core::enable_count && struct clk_ops::is_enabled()
```

```
    struct clk_core::enable_count++
```

```
    return
```





Proposed Solutions

Stop Caching Hardware State

Agenda

- Background
- Proposed Solutions
- **Brainstorming**
 - Kconfig
 - Hand off
 - **Read Hardware**

- Simple to implement
- Doesn't fix side swipe problem
- CLK_OPS_PARENT_ENABLE needs special care
 - Need to know where enable is coming from, provider or consumer
- Augment `clk_core_is_enabled()` to check children for enable state recursively
 - Many clk drivers don't implement `is_enabled` `clk_op`



Linux
Plumbers
Conference | Richmond, VA | Nov. 13-15, 2023

References

- [1] <https://lore.kernel.org/r/20210407034456.516204-1-saravanak@google.com>
- [2] <https://lore.kernel.org/r/20221227204528.1899863-1-abel.vesa@linaro.org>
- [3] <https://lore.kernel.org/r/1455225554-13267-1-git-send-email-mturquette@baylibre.com>

