Device Driver Gardening -
transplant Linux drivers fast but gently

Stefan Kalkowski
<Brussels FOSDEM 2023>
Outline

1. Motivation
2. Linux kernel ports revisited
3. Short Demo
4. Results
5. Q & A
Why to re-use Linux drivers

- Increasing complexity of hardware
Why to re-use Linux drivers

- Increasing complexity of hardware
- Poor hardware documentation
Why to re-use Linux drivers

- Increasing complexity of hardware
- Poor hardware documentation
- Hardware bugs and necessary quirks
Why to re-use Linux drivers

- Increasing complexity of hardware
- Poor hardware documentation
- Hardware bugs and necessary quirks
- Linux is open & runs everywhere
Why to re-use Linux drivers

- Increasing complexity of hardware
- Poor hardware documentation
- Hardware bugs and necessary quirks
- Linux is open & runs everywhere

Simply an economic decision
Opposed approaches

Purely the driver code

- Emulate needed semantic only
- Less sharing of emulation code
- Low complex
- Lots of manual efforts
- Deep knowledge of driver needs
Opposed approaches

Purely the driver code
- Emulate needed semantic only
- Less sharing of emulation code
- Low complex
- Lots of manual efforts
- Deep knowledge of driver needs

Maximum re-usage
- Likewise original runtime
- Sharing of emulation code possible
- Bigger codebase
- Less manual efforts
- Deep knowledge of Linux internals

Device Driver Gardening - transplant Linux drivers fast but gently
Opposed approaches

Purely the driver code

- Emulate needed semantic only
- Less sharing of emulation code
- Low complex
- Lots of manual efforts
- Deep knowledge of driver needs

Maximum re-usage

- Likewise original runtime
- Sharing of emulation code possible
- Bigger codebase
- Less manual efforts
- Deep knowledge of Linux internals

High efforts ⇒ Tendency to keep code
Display controller & connectors i.MX 8MQ

- HDMI for EVK board: 3 PM
Still too much effort

Display controller & connectors i.MX 8MQ

- HDMI for EVK board: 3 PM
- MIPI DSI with Touchscreen for EVK board: 3 PM
Still too much effort

Display controller & connectors i.MX 8MQ

- HDMI for EVK board: 3 PM
- MIPI DSI with Touchscreen for EVK board: 3 PM
- MIPI DSI with eDP bridge and Panel for MNT Reform2: wrong version
Still too much effort

Display controller & connectors i.MX 8MQ
- HDMI for EVK board: 3 PM
- MIPI DSI with Touchscreen for EVK board: 3 PM
- MIPI DSI with eDP bridge and Panel for MNT Reform2: **wrong version**

**Turning point ⇒ Need for change**
Goals for new approach

- Reduce manual work for driver-specific environment
Goals for new approach

- Reduce manual work for driver-specific environment
- Meet original semantic as close as possible
Goals for new approach

- Reduce manual work for driver-specific environment
- Meet original semantic as close as possible
- Simplify correlation to original driver
Goals for new approach

- Reduce manual work for driver-specific environment
- Meet original semantic as close as possible
- Simplify correlation to original driver
- Consolidate commonly used emulation parts
1. Motivation

2. Linux kernel ports revisited

3. Short Demo

4. Results

5. Q & A
Minimal, executable Linux kernel

make tinyconfig

LX_ENABLE = PCI PCI_MSI
LX_ENABLE += WLAN CFG80211 MAC80211 RFKILL
LX_ENABLE += WLAN_VENDOR_ATH ATH_COMMON ATH9K ATH9K_PCI ATH9K_DEBUGFS
LX_DISABLE = CC_HAS_ASM_GOTO

scripts/config --file .config $(addprefix --enable ,$(LX_ENABLE))
scripts/config --file .config $(addprefix --disable ,$(LX_DISABLE))

make olddefconfig
Device Driver Gardening - transplant Linux drivers fast but gently
tool/dts/extract --select dcss --select edp_bridge --select lcdif
tool/dts/extract --select dcss --select edp_bridge --select lcdif
Initial set of compilation units

```c
compatible = "fsl,imx8mq-lcdif", "fsl,imx28-lcdif";
compatible = "fsl,imx8m-irqsteer", "fsl,imx-irqsteer";
compatible = "fsl,imx8mq-nwl-dsi";
compatible = "fsl,imx8mq-mipi-dphy";
compatible = "nxp,imx8mq-dcss";
compatible = "ti,sn65dsi86";
compatible = "innolux,n125hce-gn1", "simple-panel";

grep -r "fsl,imx8mq-lcdif" drivers  # delivered no hit
grep -r "fsl,imx28-lcdif" drivers  # then try the second one
```
Initial set of compilation units

drivers/gpu/drm/bridge/nwl-dsi.c
drivers/gpu/drm/bridge/ti-sn65dsi86.c
drivers/gpu/drm/imx/dcss/dcss-driv.c
drivers/gpu/drm/mxsfb/mxsfb_drv.c
drivers/gpu/drm/panel/panel-simple.c
drivers/irqchip/irq-imx-irqsteer.c
drivers/phy/freescale/phy-fsl-imx8-mipi-dphy.c
- Include unmodified Linux kernel header
- Include unmodified Linux kernel header
- No manual definition rewriting anymore
Include unmodified Linux kernel header

- No manual definition rewriting anymore
- Exception proves the rule, example initcalls:

```
#include_next <linux/init.h>
#include <lx_emul/init.h>

#undef __define_initcall
#define __define_initcall ...
```
Find further compilation units

- Lots of undefined references!
Find further compilation units

- Lots of undefined references!
- Tooling for identification & generation
Find further compilation units

- Lots of undefined references!
- Tooling for identification & generation

create_dummies <command> [VARIABLES]

--- available commands ---
show - shows missing symbols of given TARGET
generate - generates DUMMY_FILE for given TARGET
Strict API discipline

Linux Shadow Copies
- C/ASM
- LX Emul
  - C/C++
- LX Kit
  - C++
  - Common Services
    - Core API
      - ROM
      - PD
      - Log
      - ...
    - Platform API
      - I/O Mem
      - IRQ
      - DMA Mem
      - ...
- Linux Kernel Code
  - C/ASM
- Platform API
  - I/O Mem
  - IRQ
  - DMA Mem
  - ...
- Driver Services
  - Block
  - Event
  - NIC
  - USB

Device Driver Gardening - transplant Linux drivers fast but gently
1. Motivation
2. Linux kernel ports revisited
3. Short Demo
4. Results
5. Q & A
Driver ports within one year

**PC universe**
- USB *HCl Controllers
- Intel Display Engine
- Intel HD Audio
- Intel Touchpad
- WIFI (Intel, Realtek, Atheros)

+ Architecture independent port of WireGuard

**ARM SoC landscape**
- (e)MMC (Zynq 7000, A64, i.MX8)
- Ethernet (A64, i.MX 5/6/7/8)
- USB Host Controller (A64, i.MX8)
- Mali GPU (A64)
- Vivante GPU (i.MX8)
- Display Engine (A64, i.MX8)
- Camera over CSI (A64)
Outcome

- Initial driver port time: ~15%
Outcome

- Initial driver port time: ~15%
  - Tooling reduces manual work
Outcome

- Initial driver port time: ~15%
  - Tooling reduces manual work
  - Debugging aid: tinykernel correlation
Outcome

- Initial driver port time: \(~15\%\)
  - Tooling reduces manual work
  - Debugging aid: tinykernel correlation
  - Driver update resp. version change faster
Outcome

- Initial driver port time: ~15%
  - Tooling reduces manual work
  - Debugging aid: tinykernel correlation
  - Driver update resp. version change faster

- Drivers better meet all-purpose
Outcome

- Initial driver port time: ~15%
  - Tooling reduces manual work
  - Debugging aid: tinykernel correlation
  - Driver update resp. version change faster

- Drivers better meet all-purpose

- Compiled codebase: ~200-300%
Outcome

- Initial driver port time: ~15%
  - Tooling reduces manual work
  - Debugging aid: tinykernel correlation
  - Driver update resp. version change faster
- Drivers better meet all-purpose
- Compiled codebase: ~200-300%
- Manually code to maintain: ~20%
Further reading

Genodians.org

GENODE
Operating System Framework 22.05
Platforms
Norman Feske

Device Driver Gardening - transplant Linux drivers fast but gently