Identifying and Supporting"X-compatible" Hardware Blocks

Chen-Yu Tsai / wens@csie.org

Chen-Yu Tsai

- Software engineer at CloudMosa, Inc. in Taipei
  - Manages Linux servers
  - Write tooling for management
- Embedded Linux hobbyist since 2011
  - Co-maintainer of kernel support for Allwinner SoC

Overview

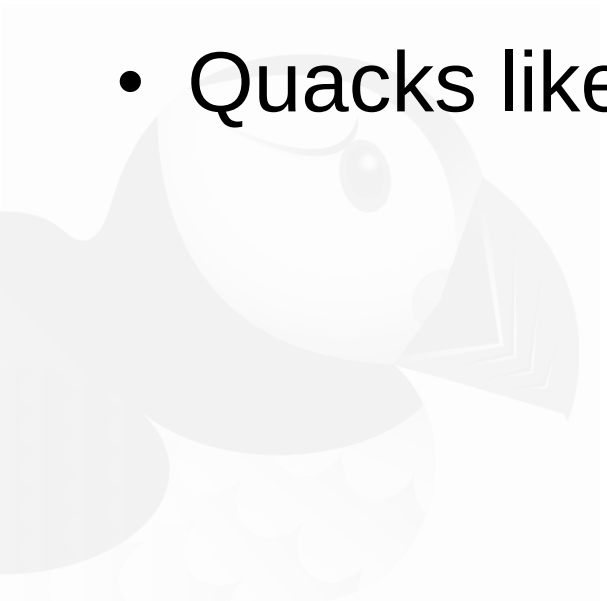
- Introduction
- Identifying hardware
- Driver implementations
- Examples

Ancient PC history

- VGA
- 8250 UART
- NE2000
- Sound Blaster

Duck Test

- Looks like a duck
- Swims like a duck
- Quacks like a duck



Compatible How?

- Looks like a duck
  - Has the same interface
- Swims like a duck
  - Has the same logic
- Quacks like a duck
  - Gives the same response

Examples

- Standard controller interfaces
  - AHCI, EHCI, OHCI, xHCI
- Licensed IP cores
- Clones

AHCI, EHCI, OHCI, xHCI

- Standard interface to access the core
  - Performance varies
- Vendor specific glue layer
  - Clocks, resets, PHYs

IP Vendors

- Synopsys (DesignWare)
- Mentor Graphics
- Cadence

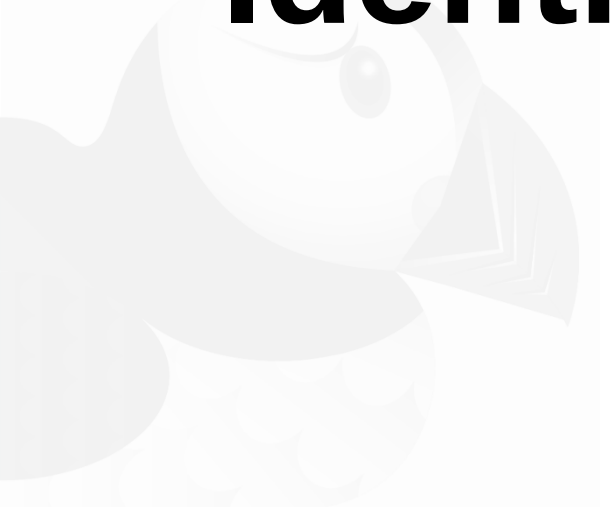
Licensed IP Cores

- Core IP block
  - May even be for HCIs
- Glue layer
- PHYs

Clones

- Same logic
- Missing registers
- Rearranged registers
- Obfuscated registers

Identifying Hardware



Resources

- Datasheets
- Vendor Kernels

Good Public Datasheets

- NXP i.MX6 Reference Manual
- TI KeyStone II User Guide Library
- Zynq UltraScale+ MPSoC Register Reference

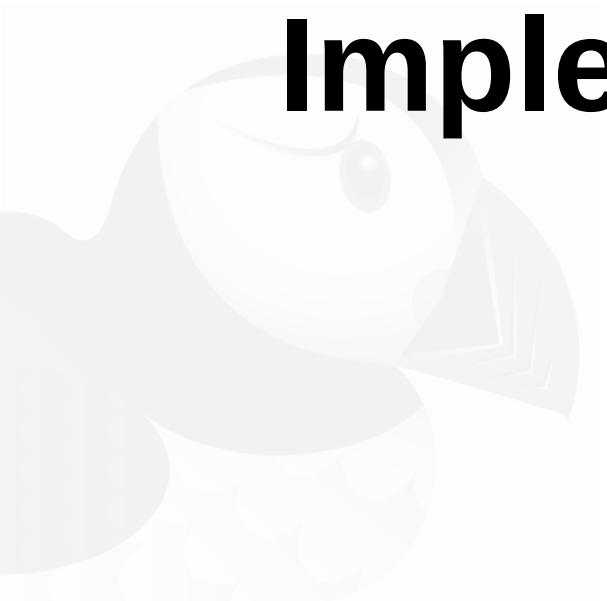
Identifiers

- Register layout
  - Register names
- DMA engine
  - Logic
  - Descriptor format

Experience & Luck

- Needle in a sea of needles
  - One can only recognize something they've seen before
- Senior members of the community may help
- Might only be recognizable after driver is finished
- YMMV

Implementing Drivers



Driver Libraries

- Core logic as a library
- Platform driver
  - Extra resource management
  - Extra setup
  - Presents parameters and callbacks to library

drivers/net/ethernet/stmicro/stmmac/dwmac-sunxi.c

```
static int sun7i_gmac_probe(struct platform_device *pdev)
{
        stmmac_get_platform_resources(pdev, &stmmac_res);
        stmmac_probe_config_dt(pdev, &stmmac_res.mac);

        /* platform specific stuff ... */
        plat_dat->init = sun7i_gmac_init;
        plat_dat->exit = sun7i_gmac_exit;

        stmmac_dvr_probe(&pdev->dev, plat_dat, &stmmac_res);
}
```

Dealing with Register Changes

- Register offset table
- `regmap` & `regmap_field`
- Custom I/O callbacks

Register Offset Table

drivers/i2c/busses/i2c-mv64xxx.c:

```
struct mv64xxx_i2c_regs {
        u8      addr;
        u8      ext_addr;
        u8      data;
        u8      control;
        u8      status;
        u8      clock;
        u8      soft_reset;
};

writel(0, drv_data->reg_base + drv_data->reg_offsets.soft_reset);
```

regmap_field

drivers/gpu/drm/sun4i/sun4i_hdmi_enc.c:

```
static const struct sun4i_hdmi_variant sun6i_variant = {
…
        .field_ddc_en           = REG_FIELD(SUN4I_HDMI_DDC_CTRL_REG, 31, 31),
        .field_ddc_start        = REG_FIELD(SUN4I_HDMI_DDC_CTRL_REG, 30, 30),
        .field_ddc_reset        = REG_FIELD(SUN4I_HDMI_DDC_CTRL_REG, 0, 0),
        .field_ddc_addr_reg     = REG_FIELD(SUN4I_HDMI_DDC_ADDR_REG, 0, 31),
        .field_ddc_slave_addr   = REG_FIELD(SUN4I_HDMI_DDC_ADDR_REG, 0, 6),
…
};
```

regmap_field

drivers/gpu/drm/sun4i/sun4i_hdmi_enc.c:

```
static const struct sun4i_hdmi_variant {
…
        /* Regmap fields for I2C adapter */
        struct regmap_field     *field_ddc_en;
        struct regmap_field     *field_ddc_start;
        struct regmap_field     *field_ddc_reset;
        struct regmap_field     *field_ddc_addr_reg;
        struct regmap_field     *field_ddc_slave_addr;
…
};
```

regmap_field

```
regmap_field_write(hdmi->field_ddc_fifo_tx_thres,
                   hdmi->variant->ddc_fifo_thres_incl ? 0 : 1);
regmap_field_write(hdmi->field_ddc_fifo_rx_thres, RX_THRESHOLD);
regmap_field_write(hdmi->field_ddc_fifo_clear, 1);
if (regmap_field_read_poll_timeout(hdmi->field_ddc_fifo_clear,
                                   reg, !reg, 100, 2000))
        return -EIO;
```

Custom I/O Functions

drivers/usb/musb/musb_io.h:

```
/**
 * struct musb_io - IO functions for MUSB
 * @quirks:     platform specific flags
 * @ep_offset:  platform specific function to get end point offset
 * @ep_select:  platform specific function to select end point
 * @fifo_offset: platform specific function to get fifo offset
 * @read_fifo:  platform specific function to read fifo
 * @write_fifo: platform specific function to write fifo
 * @busctl_offset: platform specific function to get busctl offset
 */
```

Examples

8250 UART

- De-facto standard
  - 8250 driver library
- Platform specifics
  - Extra features
  - Quirks or bugs

Synopsys DesignWare APB UART

- Extra “busy” interrupt
  - Fires on attempts to write to LCR when UART is busy
  - “irq XX: nobody cared”
- Extra “UART status register”
- 8250_dw driver wraps 8250 library

AHCI

- `libahci & libahci_platform`
  - Driver library
  - Handles PHYs, clocks, regulators
- Platform specific `ahci_*`
- Generic `ahci_platform`

Mentor Graphics MUSB

- USB On-The-Go IP core
  - Seen on a few platforms
  - Has all related signals (ID & VBUS detect)
- Separate USB PHY

Allwinner USB OTG

- MUSB with rearranged registers
  - Custom I/O functions
- Very weird USB PHY interface
  - Generic PHY framework
- ID & VBUS detect lines not routed outside
  - Controlled through custom register
  - Use GPIO lines externally
  - extcon framework

DesignWare MAC

- Many hardware revisions
  - Later ones have feature register
- Found on 12+ platforms
- Called “`stmmac`” in the kernel
  - Driver library
  - Platform provides config and callbacks

A20 GMAC

- Old revision of DWMAC
- No DMA feature register
  - Dug out feature set from vendor kernel
- Simple glue layer

A64/H3 EMAC

- DWMAC with rearranged registers
  - MDC bitfield meaning is different
  - Chained DMA descriptors only
- DMA descriptor format is the same
  - A few fields are marked as “reserved”

DesignWare HDMI

- Found on 3+ platforms
  - With either DW-HDMI-PHY or 3rd party PHY
- “`dw-hdmi`” in the kernel
  - Library-ish driver
  - Platform provides config and callbacks

A64/H3 HDMI

- Obfuscated register addresses
  - Obfuscation can be turned off :)
    - Found in some vendor kernel release
- Custom PHY
  - Reverse engineered init code
    - PHY settings limited to certain clock divisors

Questions?